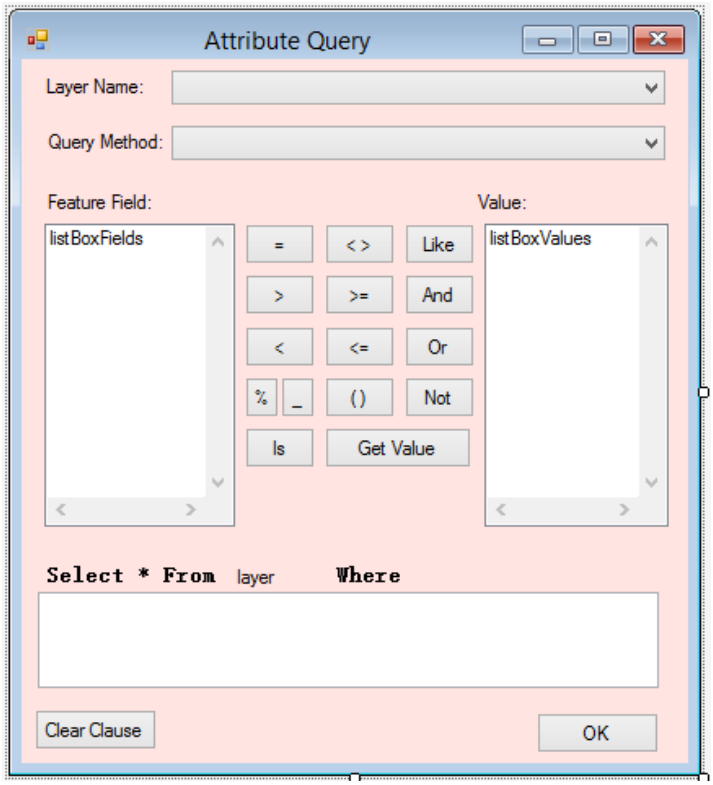
Chapter 10 Query By Attributes

1. Add a Windows Form “SelectByAttributes” to the project:



GUI implementation detail could be found at “SelectByAttributes.Designer.cs” file.

2. Implementation of SelectByAttributes:

- Class member and Construction method:

```
IHookHelper m_hookhelper;
IMap m_map;
IActiveView m_activeview;
public SelectByAttribute(IHookHelper hookhelper)
{
    InitializeComponent();
    m_hookhelper = hookhelper;
    m_map = hookhelper.FocusMap;
    m_activeview = m_hookhelper.ActiveView;
}
```

- Add Load event of form:

```
private IEnumLayer GetLayers()
{
    UID uid = new UIDClass();
    uid.Value = "{40A9E885-5533-11d0-98BE-00805F7CED21}"; //IFeatureLayer
    IEnumLayer layers = m_map.get_Layers(uid, true);
    return layers;
}

private void SelectByAttribute_Load(object sender, EventArgs e)
{
    IEnumLayer layers = GetLayers();
    layers.Reset();
    ILayer layer = layers.Next();
    while (layer != null)
    {
        comboBoxLayers.Items.Add(layer.Name.ToString());
        layer = layers.Next();
    }
}
```

- Add selectedIndexChanged event of “comboBoxLayers”:

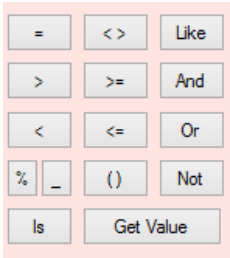
```
private void comboBoxLayers_SelectedIndexChanged(object sender, EventArgs e)
{
    listBoxFields.Items.Clear();
    listBoxValues.Items.Clear();
    string strSelectedLayerName = comboBoxLayers.Text;
    IFeatureLayer pFeatureLayer;
    try
    {
        for (int i = 0; i < m_map.LayerCount; i++)
        {
            if (m_map.get_Layer(i).Name == strSelectedLayerName)
            {
                if (m_map.get_Layer(i) is IFeatureLayer)
                {
                    pFeatureLayer = (IFeatureLayer)m_map.get_Layer(i);
                    for (int j = 0; j < pFeatureLayer.FeatureClass.Fields.FieldCount; j++)
                    {
                        listBoxFields.Items.Add(pFeatureLayer.FeatureClass.Fields.get_Field(j).Name);
                    }
                    labelDescription2.Text = strSelectedLayerName;
                }else
                {
                    MessageBox.Show("This Layer could be queried! Please choose another");
                    break;
                }
            }
        }
    }
    catch (Exception ex){
        MessageBox.Show(ex.Message);
    }
}
```

- SelectedIndexChanged event of comboBoxMethod:

```
private esriSelectionResultEnum selectmethod = esriSelectionResultEnum.esriSelectionResultNew;
private void comboBoxMethod_SelectedIndexChanged(object sender, EventArgs e)
{
    switch (comboBoxMethod.SelectedIndex)
    {
        case 0: selectmethod = esriSelectionResultEnum.esriSelectionResultNew; break;
        case 1: selectmethod = esriSelectionResultEnum.esriSelectionResultAdd; break;
        case 2: selectmethod = esriSelectionResultEnum.esriSelectionResultSubtract; break;
        case 3: selectmethod = esriSelectionResultEnum.esriSelectionResultAnd; break;
    }
}
```

- clauseElementClicked method, and define as the every button's click event in the following graph (except "Get Value" and "()"):

```
private void clauseElementClicked(object sender, EventArgs e)
{
    textBoxWhereClause.SelectedText = ((Button)sender).Text;
}
```



- Click event of "()" and DoubleClick event of listBoxFields:

```
private void buttonBrace_Click(object sender, EventArgs e)
{
    textBoxWhereClause.SelectedText = "(  )";
    textBoxWhereClause.SelectionStart = textBoxWhereClause.Text.Length - 2;
}

private void listBoxFields_DoubleClick(object sender, EventArgs e)
{
    textBoxWhereClause.SelectedText = listBoxFields.SelectedItem.ToString() + " ";
}
```

- GetUniqueValuesCount, getUniqueValue, and GetLayerByName:

```
private int GetUniqueValuesCount(IFeatureClass featureClass, string strField)
{
    ICursor cursor = featureClass.Search(null, false) as ICursor;
    IDataStatistics dataStatistics = new DataStatistics();
    dataStatistics.Field = strField;
    dataStatistics.Cursor = cursor;
    System.Collections.IEnumerator enumerator = dataStatistics.UniqueValues;
    return dataStatistics.UniqueValueCount;
}
private System.Collections.IEnumerator GetUniqueValues(
    IFeatureClass featureClass, string strField)
{
    ICursor cursor = (ICursor)featureClass.Search(null, false);
    IDataStatistics dataStatistics = new DataStatistics();
    dataStatistics.Field = strField;
    dataStatistics.Cursor = cursor;
    System.Collections.IEnumerator enumerator = dataStatistics.UniqueValues;
    return enumerator;
}
```

```
private ILayer GetLayerByName(string strLayerName)
{
    ILayer pLayer = null;
    for (int i = 0; i < m_map.LayerCount; i++)
    {
        pLayer = m_map.get_Layer(i);
        if (strLayerName == pLayer.Name) { break; }
    }
    return pLayer;
}
```

- Click Event of “Get Value”:

```
private void buttonGetValue_Click(object sender, EventArgs e)
 {
     if (listBoxFields.Text == "")
     {
         MessageBox.Show("Please choose a field"); return;
     }
     string strSelectedFieldName = listBoxFields.Text;
     listBoxValues.Items.Clear();
     valueCounts.Text = "";
     if (strSelectedFieldName == null) return;
     IFeatureClass pFeatureClass = ((IFeatureLayer)GetLayerByName
         (comboBoxLayers.Text)).FeatureClass;
     if (pFeatureClass == null) return;
     int fieldIndex = pFeatureClass.Fields.FindField(strSelectedFieldName);
     IField field = pFeatureClass.Fields.get_Field(fieldIndex);
     try
     {
         System.Collections.IEnumerator uniqueValues = GetUniqueValues
             (pFeatureClass, strSelectedFieldName);
         if (uniqueValues == null) return;
         if ((field.Type == esriFieldType.esriFieldTypeDouble) ||
             (field.Type == esriFieldType.esriFieldTypeInteger) ||
             (field.Type == esriFieldType.esriFieldTypeSingle) ||
             (field.Type == esriFieldType.esriFieldTypeSmallInteger))
         {
             System.Collections.Generic.List<double> valuesList =
                 new System.Collections.Generic.List<double>();
             while (uniqueValues.MoveNext())
             {
                 valuesList.Add(double.Parse(uniqueValues.Current.ToString()));
             }
             valuesList.Sort();
             foreach (object uniqueValue in valuesList)
             {
                 listBoxValues.Items.Add(uniqueValue.ToString());
             }
         }
         else
         {
             System.Collections.Generic.List<object> valuesList =
                 new System.Collections.Generic.List<object>();
             while (uniqueValues.MoveNext())
             {
                 valuesList.Add(uniqueValues.Current);
             }
             valuesList.Sort();
```

```
                foreach (object uniqueValue in valuesList)
                {
                    listBoxValues.Items.Add(uniqueValue.ToString());
                }
            }
            valueCounts.Text = GetUniqueValuesCount(pFeatureClass,
                strSelectedFieldName).ToString() + " values";
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
        }
    }
```

- DoubleClick event of listBoxValues and Click event of “Clear Clause”:

```
private void listBoxValues_DoubleClick(object sender, EventArgs e)
{
    textBoxWhereClause.SelectedText = " " + listBoxValues.SelectedItem.ToString();
}

private void buttonClear_Click(object sender, EventArgs e)
{
    textBoxWhereClause.Clear();
}
```

- ExecuteAttributeSelect:

```
private int ExecuteAttributeSelect()
{
    try
    {
        IQueryFilter pQueryFilter = new QueryFilter() as IQueryFilter;
        IFeatureLayer pFeatureLayer = null;
        pQueryFilter.WhereClause = textBoxWhereClause.Text;
        ILayer targetLayer = GetLayerByName(comboBoxLayers.Text);
        pFeatureLayer = (IFeatureLayer)targetLayer;
        pFeatureSelection = (IFeatureSelection)pFeatureLayer;
        pFeatureSelection.SelectFeatures(pQueryFilter, selectmethod, false);
        if (pFeatureSelection.SelectionSet.Count == 0)
        {
            MessageBox.Show("Could not find");
            return 0;
        }
        m_activeview.PartialRefresh(esriViewDrawPhase.esriViewGeoSelection, null, null);
        return pFeatureSelection.SelectionSet.Count;
    }
    catch
    {
        MessageBox.Show("Error may exist in query clause, please input again");
        return -1;
    }
}
```

- Add Click event of “OK”:

```
private void buttonOk_Click(object sender, EventArgs e)
{
    if (textBoxWhereClause.Text == string.Empty)
    {
        MessageBox.Show("Please Build SQL query clause");
        return;
    }
    int result = ExecuteAttributeSelect();
    if (result == -1)
    {
        labelResult.Text = "Error!";
        return;
    }
    labelResult.Text = string.Format("Find {0} values", result);
}
```

- Add FormClosing event of the form:

```
private void SelectByAttribute_FormClosing(object sender, FormClosingEventArgs e)
{
    if (pFeatureSelection != null)
    {
        System.Runtime.InteropServices.Marshal.ReleaseComObject(pFeatureSelection);
    }
}
```

- Define map selection change changes and add “SetupEvents()” in the end of Load method:

```
IActiveViewEvents_Event activeViewEvent = null;
IActiveViewEvents_SelectionChangedEventHandler mapSelectionChanged;
private void SetupEvents()
{
    activeViewEvent = m_activeview as IActiveViewEvents_Event;
    mapSelectionChanged = new
        IActiveViewEvents_SelectionChangedEventHandler(OnMapSelectionChanged);
    activeViewEvent.SelectionChanged += mapSelectionChanged;
}

private void OnMapSelectionChanged()
{
    m_activeview.PartialRefresh(esriViewDrawPhase.esriViewGeoSelection,
        null, m_activeview.Extent);
}
```

3. Add A Base Command Class AttributeQueryCmd, and implement OnClick method:

```
public override void OnClick()
{
    // TODO: Add AttributeQueryCmd.OnClick implementation
    if (m_hookHelper.Hook == null) return;
    if (m_hookHelper.FocusMap.LayerCount != 0)
    {
        SelectByAttribute frmAttributeQuery = new SelectByAttribute(m_hookHelper);
        frmAttributeQuery.Show(m_hookHelper as System.Windows.Forms.IWin32Window);
    }
}
```

4. Back to MainForm, add a menu content and its click event:

```
private void queryByAttributeToolStripMenuItem_Click(object sender, EventArgs e)
{
    ICommand command = new AttributeQueryCmd();
    command.OnCreate(m_mapControl.Object);
    command.OnClick();
}
```