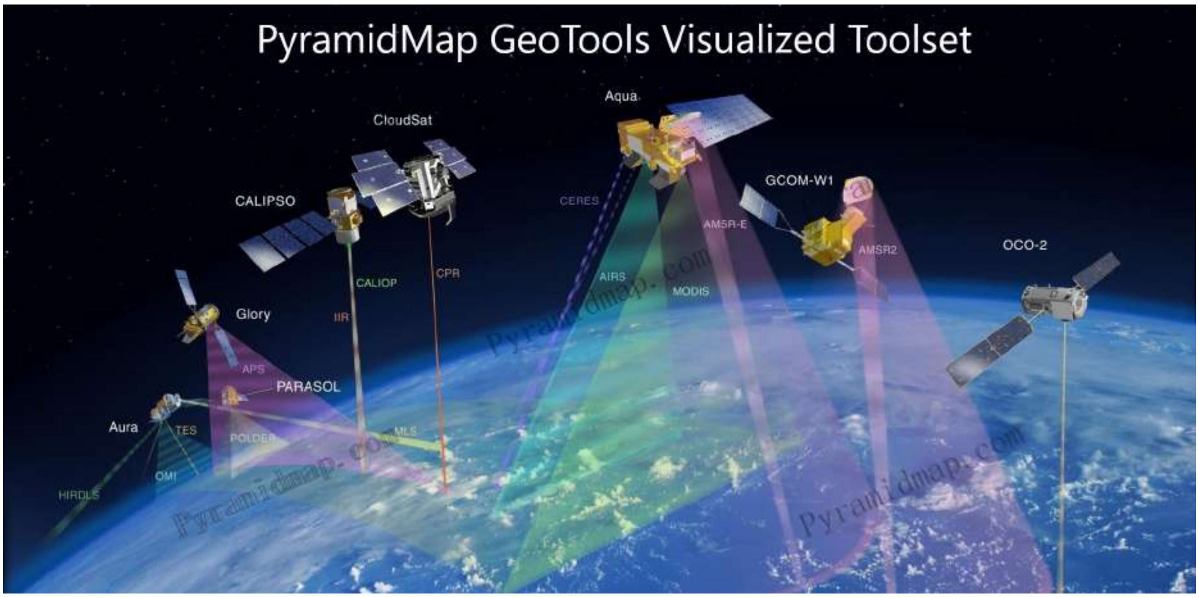


PyramidMap Geotools Visualized Toolset Instructions

1 Introduction

1.1: Function overview

As the world enters the information and digital era, the development of communication and Internet technology has completely changed the way people communicate and perceive the world. GIS (Geographic Information System) has developed from original paper map to holographic digital electronic map. GIS is synchronized with the software and hardware support of operating system, communication interconnection and mobile terminal, and has penetrated into all aspects of social production and people's life. From space launch, satellite remote sensing, express delivery and bus service, almost every link is shining with the glory of GIS. GIS can integrate spatial and dynamic information into holographic digital visual effects, and provide timely and accurate visual services for geographical research, geographical decision-making and residents' lives. The rich and colorful services of GIS come from its massive data support and powerful analysis and processing capabilities. However, the production, storage, analysis, processing and service provision of map data is an extremely professional, arduous and complex work, which has professional requirements for workflow and operators, and the technical cost is very high. Therefore, many enterprises with this demand but weak technology are discouraged. PyramidMap is a simple and practical GIS data processing workflow platform for small and medium-sized enterprises to solve this problem.

PyramidMap GeoTools visual map tool set is launched by PyramidMap (www.Pyramidmap. com). It is based on GeoTools component technology, which can be used as an independent map processing tool, as well as oriented to GeoServer and used as its client. It constitutes a simple and easy-to-use map service system covering mapping, editing and assignment, rendering, database storage, service release and management. It is simple and easy to use, covering mapping, editing and assignment, rendering, database storage, service publishing, and management. The purpose of PyramidMap is to transform the professional and even daunting GIS data processing into an approachable, easy to use, relaxed and pleasant workflow. On this basis, the original complex process of GeoServer and Geodatabase connection and map service publishing is transformed into a fully guided visualization process, reducing the user threshold and improving the efficiency of map operation, Thus, GIS users are given the ability to independently complete the whole process from map mapping to web map services. Its ultimate goal is to provide efficient data processing process for WebGIS applications. That is to say, PyramidMap tool provides you with the most basic data assurance processing process for WebGIS applications. It belongs to the core layer of map data processing and is also the most complex process. PyramidMap is fully qualified for this work. The process is shown in Figure 1-1.

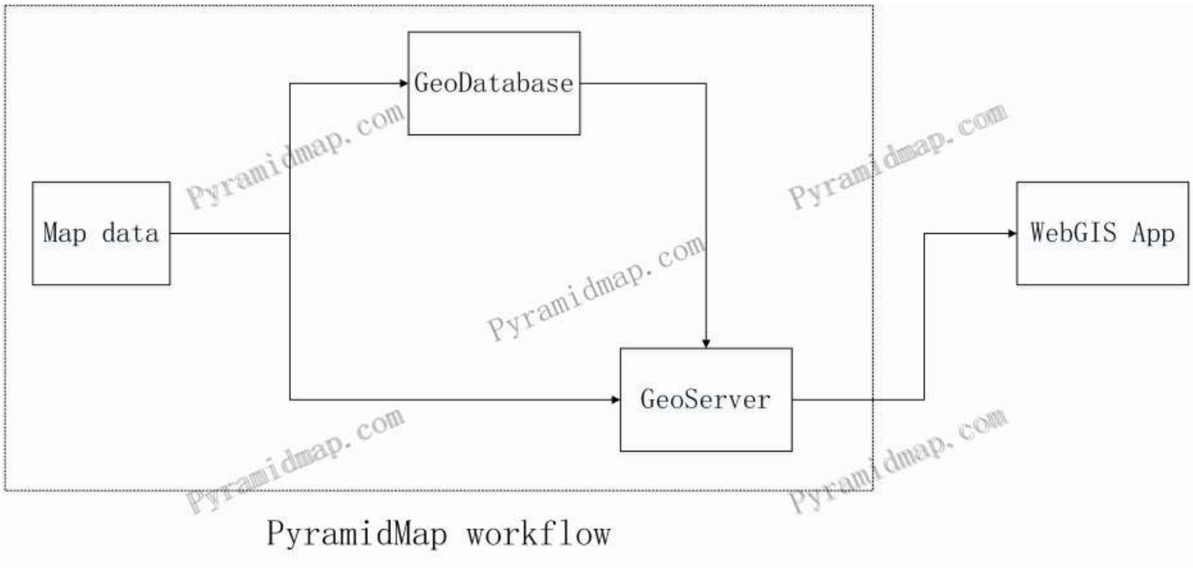


Figure 1-1: Diagram of PyramidMap Total Energy

PyramidMap is positioned at the key link of the whole WebGIS implementation process, and completes the processing and storage of key map data before the feasibility of WebGIS. Traditionally, it is highly dependent on the skills of professional GIS personnel. Now it can be easily completed by PyramidMap, which greatly reduces the access threshold for GIS users and improves the efficiency of map data processing.

1.2 Contact us

PyramidMap focuses on GIS application services and is committed to building a complete ecosystem of efficient GIS data processing, enterprise level geographic data storage, distributed map services, web and mobile map applications. It can help you achieve map data processing more easily and quickly, until the terminal displays the entire process of the application as a streamlined service. Welcome visiting to: http://www.pyramidmap.com obtaining beta software and technical information. For more technical support, please contact PyramidMap studio.

Email: service@pyramidmap.com

Tele: (086)0535-82957588

Mobile: (086)18660789051

WeChat: A18660789051

QQ: 29862351

2 Software deployment

2.1 Deployment mode

PyramidMap integrates all the dependencies required for operation internally. You only need to decompress the software to use it (Chinese path is not supported for the moment). Double click the PyramidMapViewer.exe executable to run it. The deployment mode is shown in Figure 2-1.

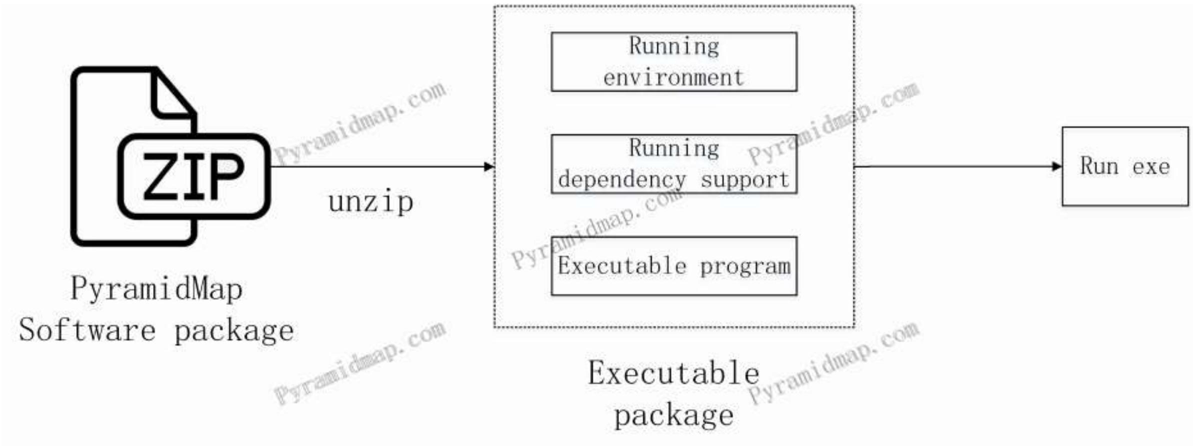


Figure 2-1: Schematic Diagram of PyramidMap Deployment Operation Mode

The PyramidMap GeoTools visualization toolset is only supported running in a Windows environment currently , with a minimum resolution requirement of 1920 * 1200.

2.2 Running mode

- Client/Server mode

PyramidMap can complete independent mapping, map editing, symbol definition rendering, Geodatabase geographic database connection, access, map data input and output, GeoServer map server space management, map service publishing, hosting and access and other full process processing. PyramidMap can independently complete all functions from mapping to editing, support multiple spatial geographic databases, including but not limited to Oracle, Postgre, MySQL, and complete the input, output, access and storage of map data. PyramidMap can be seamlessly connected with GeoServer map server, and can be used as a visualization client tool of GeoServer to complete workspace and data storage management of remote server, layer publishing, map symbol production and publishing, server layer and symbol data management, preview and other serialized operations on the client. To sum up, PyramidMap provides map users with a full process function from mapping to Web side use. PyramidMap supports deployment in Internet and intranet environments. It has all the dependencies required for integrated operation. It can be used after decompression without installation (Chinese path is not supported temporarily).The role and operation mode of PyramidMap in WebGIS network architecture are shown in Figure 2-2.

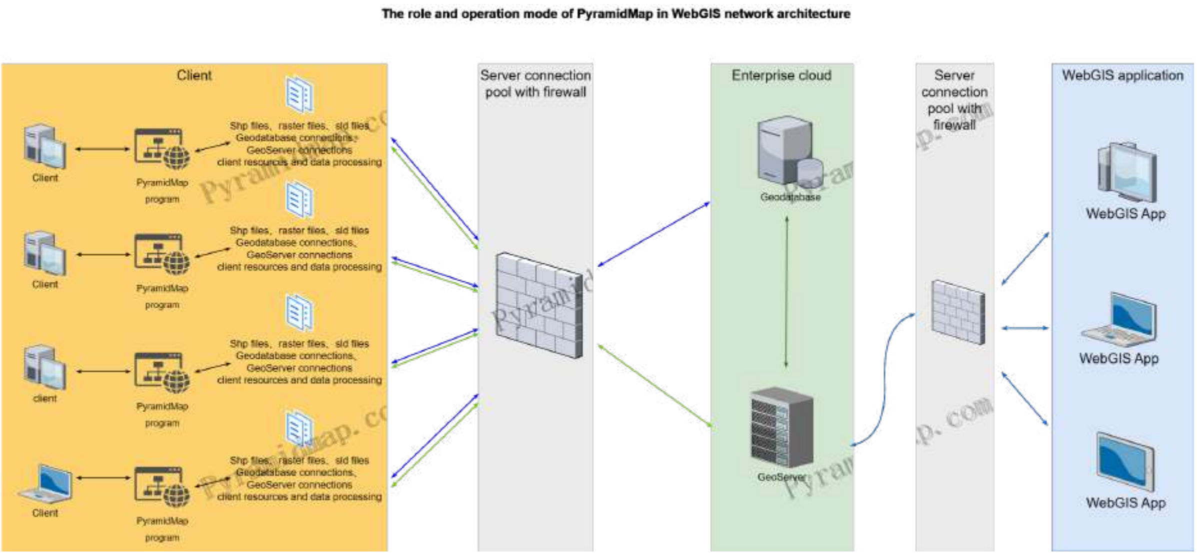


Figure 2-2: The role and operation mode of PyramidMap in WebGIS network architecture

2.3 System function

Functions	Products				Application Scenario
	Free	Basic	Standard	Professional	
Map display	●	●	●	●	Displaying various map such as vector, raster, geodatabase, GeoServer and WMTS.
Map resource management	●	●	●	●	Vector, raster, geodatabase, GeoServer, WMS, WFS/WMTS layers list.
Shp creation and editing	●	●	●	●	Design and create the original Shape file and initialize the data table.
Map feature selection	●	●	●	●	The selecting interface exchange of map points, lines, and surface features.
Map feature drawing	●	●	●	●	Create points, lines, and polygons on the map and perform custom rendering.
Map data editing	●	●	●	●	Add, delete, and modify map features and their attribute data.
Map data query	●	●	●	●	Build visual query analyzer, implement multi condition constructed queries for attribute data, and achieve specific rendering of result sets.
Map data conversion			●	●	The bidirectional exchange between heterogeneous data such as Shp, Geodatabase, GeoServer, and Excel, Csv, Kml, GeoJson.
Map symbol management		●	●	●	Create SLD symbols and manage in pool locally, and achieve bidirectional synchronization with GeoServer.
Coordinate System Conversion			●	●	Coordinate systems conversion for vector and raster, supporting the vast majority of EPSG standardized geographic and projection coordinate systems.
Vector layers tile	Level 10	Level 12	Level 16	Level 24	Build TMS, Google XYZ, MVT (Mapbox Vector Tile), MBTile (Mapbox Tile) tiles.
Raster layers tile	Level 10	Level 12	Level 16	Level 24	Build TMS, Google XYZ, MBTile (Mapbox Tile) tiles.
Raster data processing			●	●	Raster compression, slicing, merging, NoData processing.
GeoDatabase connections pool	●	●	●	●	Supports Oracle, PostGIS, MySQL, and SQL Server connections pools.
Import and export of GeoDatabase			●	●	Import layers into goodatabase and export out such as Shp, Csv, Kml, GeoJson.
GeoServer connections pool	●	●	●	●	Create GeoServer connections and maintain in pool mode.
GeoServer workspace management				●	Creates and obtains workspaces for GeoServer on client side.
GeoServer datastorage management				●	Creates and obtains data storages for GeoServer on client side.
Workspace Sync with GeoServer				●	Sync workspaces between clinet and GeoServer.
DataStorage Sync with GeoServer				●	Sync datastorages between clinet and GeoServer.
SLD Sync with GeoServer			●	●	Sync slds between clinet and GeoServer.
Layers published to GeoServer				●	Publish vector and raster layers to GeoServer on the client side.

Table 2-1: PyramidMap release functions list

3 Map view

The main map interface is divided into four functional areas, as shown in Figure 3-1. They are the main map view and the display container of all map data. They accept the loading and display of various map resource data from Shp files, image files, Geodatabase, GeoServer, online maps, etc; The online layer node on the left is the layer data source node that has been loaded into the main map view. It is classified according to the path layer, and has the corresponding level menu to complete the operation as instructed; On the upper right is the Geodatabase data source node, which is classified according to the data connection layer table, and has the corresponding level menu. The operation can be completed according to the instructions. The layer table can be dragged to the map view area or displayed by double clicking the mouse, and automatically added to the layer display node on the left; The lower right side is the GeoServer workspace and layer data source node, which are classified according to GeoServer connection - workspace - layer, and have the corresponding level menu. The operation is completed according to the instructions. The layer can be dragged to the map view area or displayed by double clicking the mouse, and automatically added to the layer display node on the left.

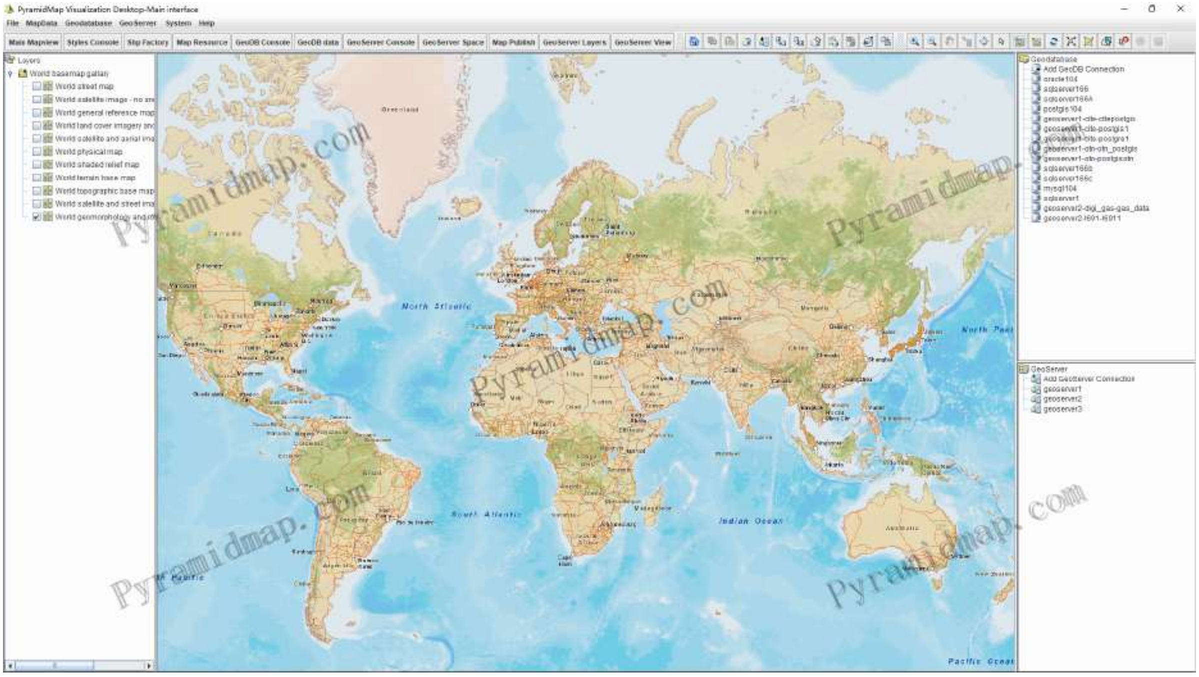


Figure 3-1: Main map viewer

3.1 Load basemap

The base map provides a reference map for the system and an environment for the content to be displayed in the map. When creating a new map, you can select an underlay to use. PyramidMap currently supports many base map resource based on Web Mercator coordinate system and others standard coordinate system. You can change the base map of the current map at any time: you can select the base map from the base map library. With the help of the base map, accurate spatial location calibration can be carried out, map data related to location can be processed, dot, line and picture can be plotted on the map, and accurate positioning and track query can be carried out. The main interface loads and switches the base map through the shortcut menu in the toolbar, as shown in Figure 3-2.

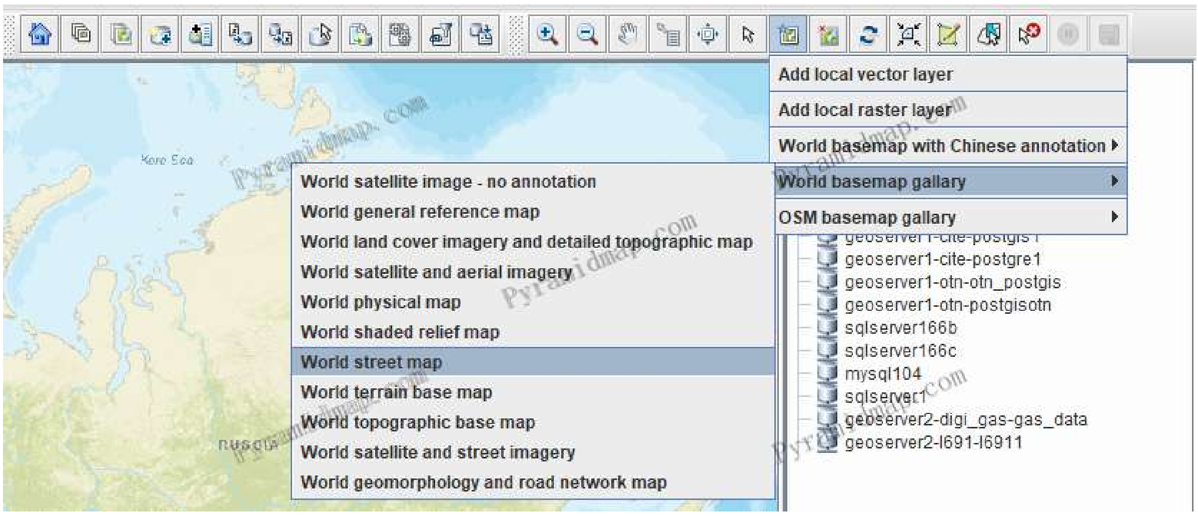


Figure 3-2: Main viewer base map menu

PyramidMap supports vector and raster type base map resources， This base map is designed to be used as a general reference map for informational purposes as well as for GIS professionals and other users to creating web maps and web mapping applications.

The world general reference map includes administrative boundaries, cities, protected areas, highways, roads, railways, water features, buildings and landmarks, overlaid on shaded relief and land cover imagery for added context. This reference map display effect is shown in Figure 3-3.

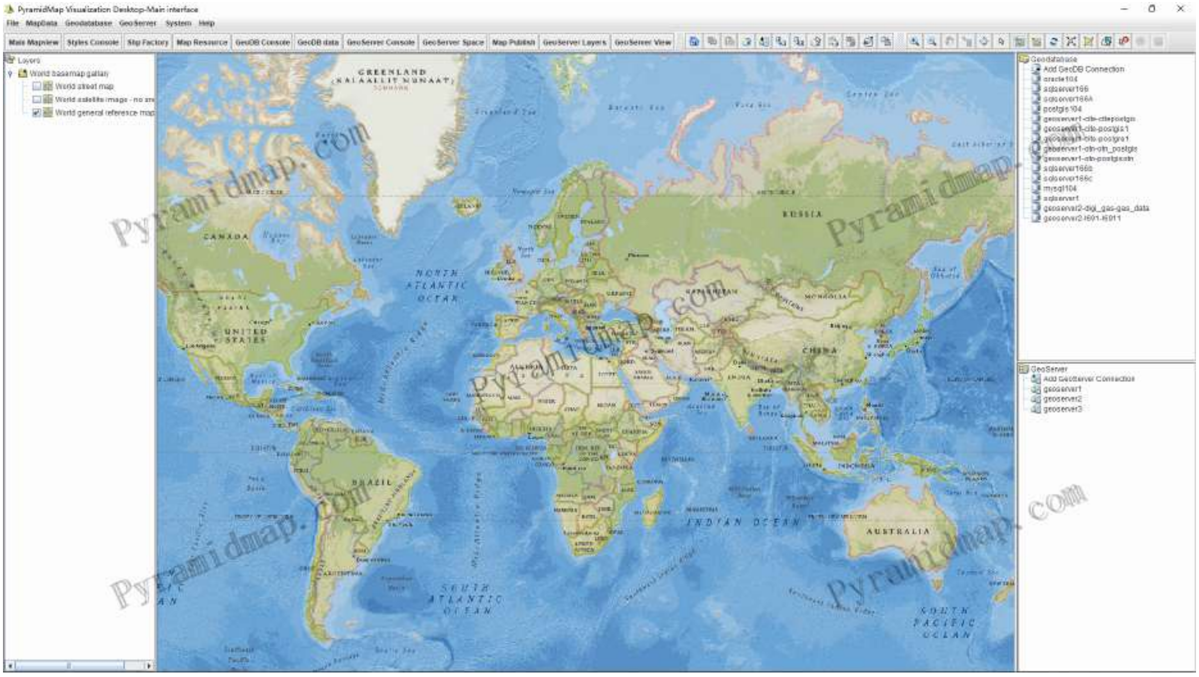


Figure 3-3：The world general reference map display effect

The local details somewhere for the world general reference map are shown in Figure 3-4.

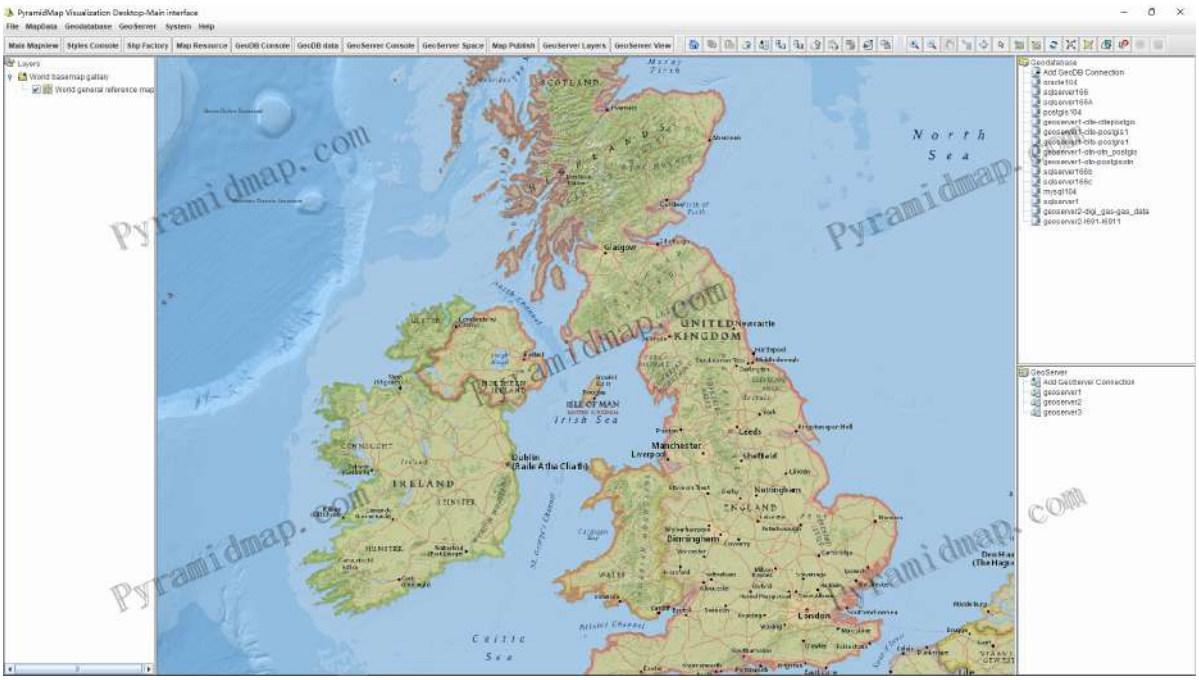


Figure 3-4：The world general reference map somewhere local details

The world land cover imagery and detailed topographic map presents land cover imagery and detailed topographic maps for the world. This map display effect is shown in Figure 3-5.

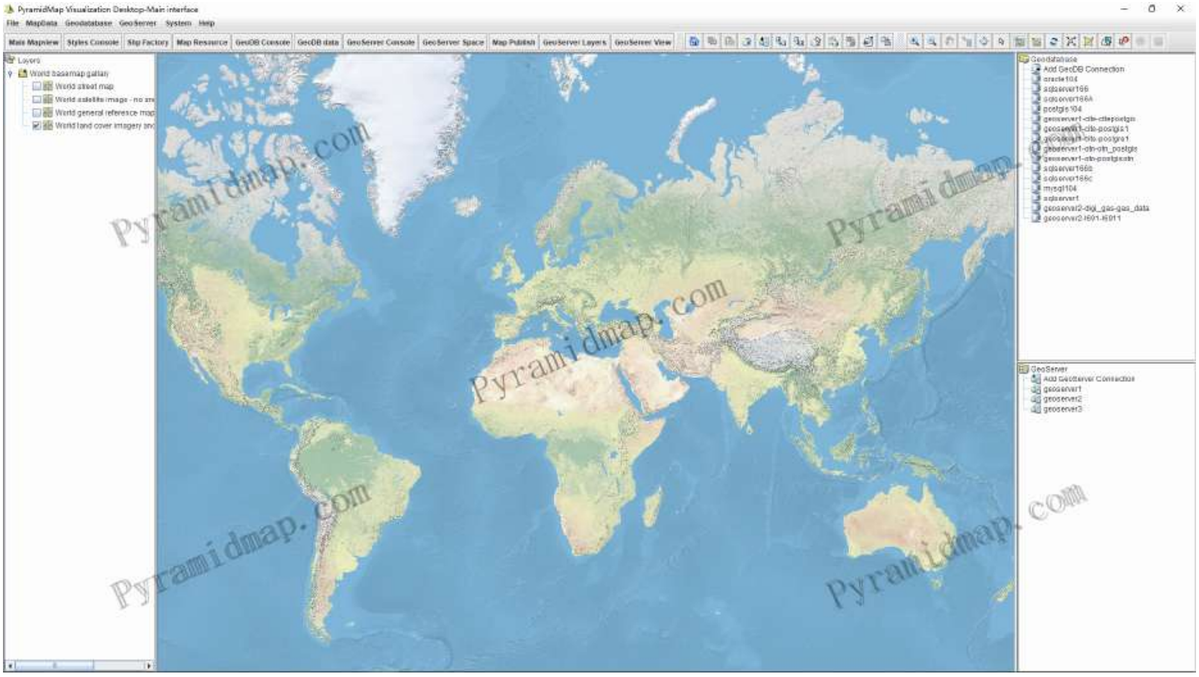


Figure 3-5：The world land cover imagery and detailed topographic map display effect

The local details somewhere for the world land cover imagery and detailed topographic map are shown in Figure 3-6.

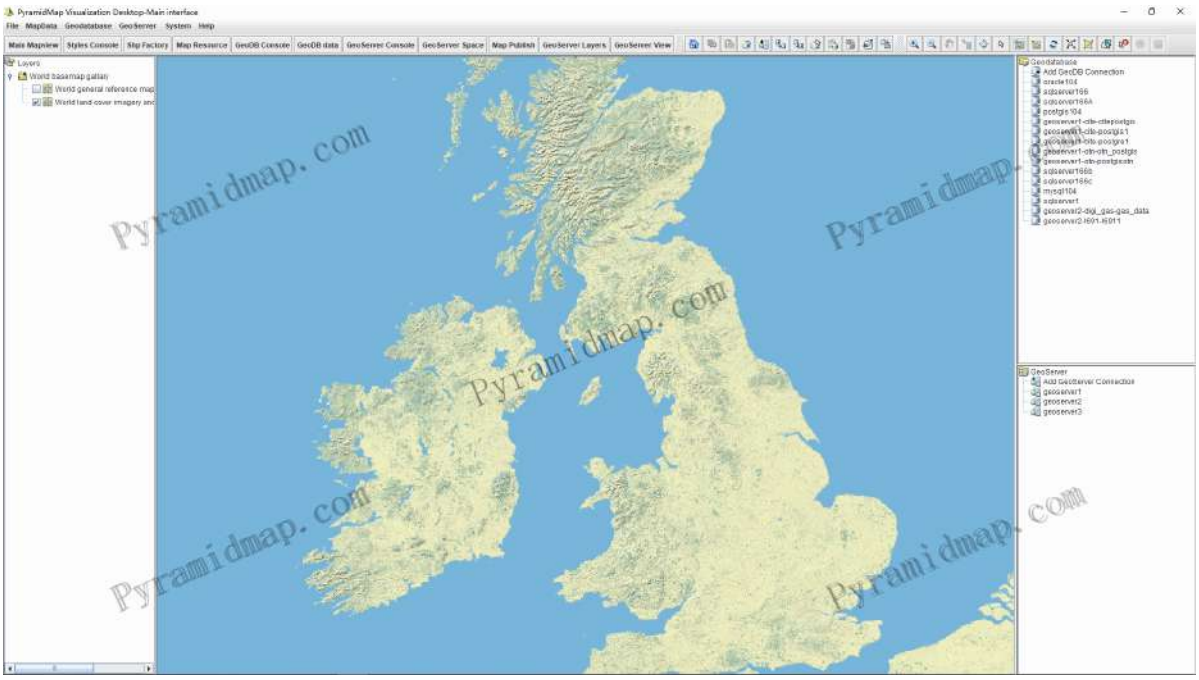


Figure 3-6：The world land cover imagery and detailed topographic map somewhere local details

The world satellite and aerial imagery map provides one meter or better satellite and aerial imagery in many parts of the world and lower resolution satellite imagery worldwide. The map includes 15m TerraColor imagery at small and mid-scales (~1:591M down to ~1:288k) for the world. The map features Maxar imagery at 0.3m resolution for select metropolitan areas around the world, 0.5m resolution across the United States and parts of Western Europe, and 1m resolution imagery across the rest of the world.The world satellite and aerial imagery map features high-resolution aerial photography contributed by the GIS User Community. This imagery ranges from 0.3m to 0.03m resolution (down to ~1:280 in select communities). The world satellite image display effect is shown in Figure 3-7.

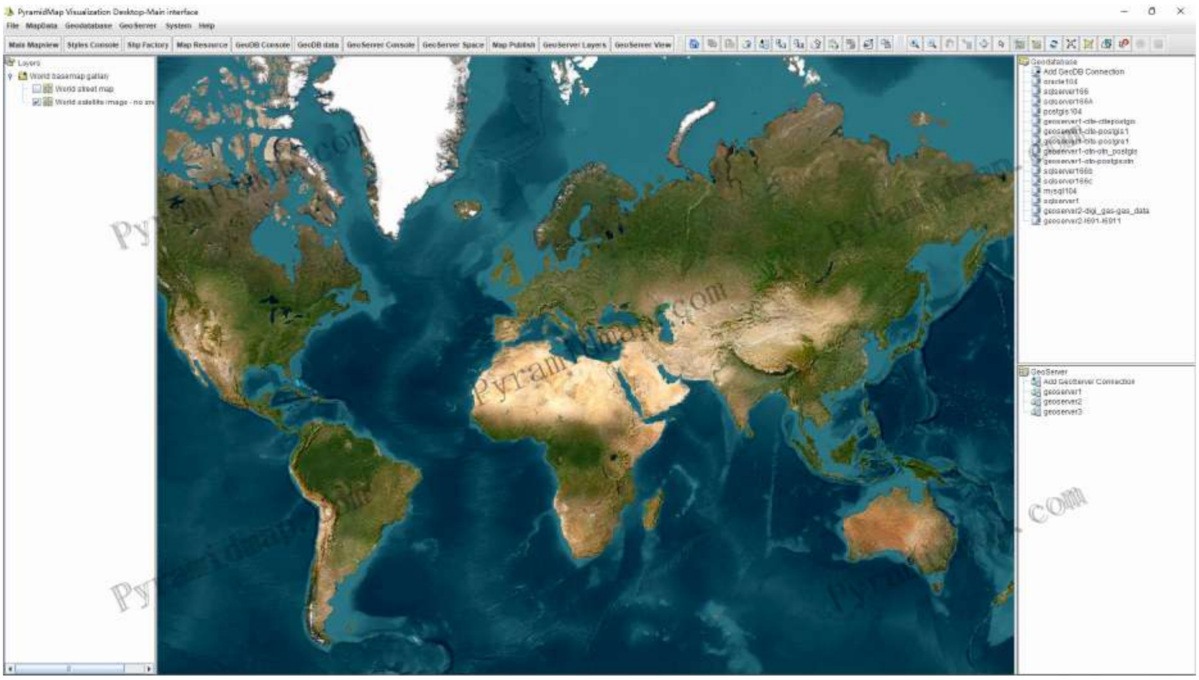


Figure 3-7：The world satellite image map display effect

The local details somewhere for the world satellite and aerial imagery map are shown in Figure 3-8.

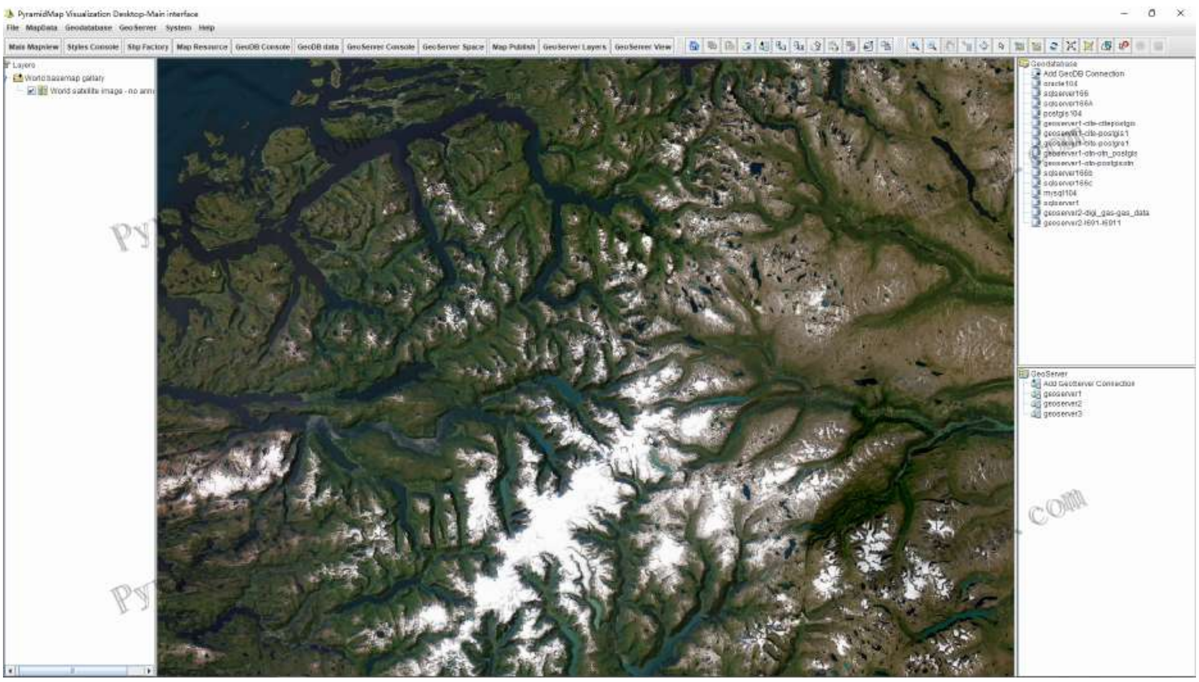


Figure 3-8：The world satellite image map somewhere local details

The world physical map presents the Natural Earth physical map at 1.24km per pixel for the world and 500m for the coterminous United States. This map display effect is shown in Figure 3-9.

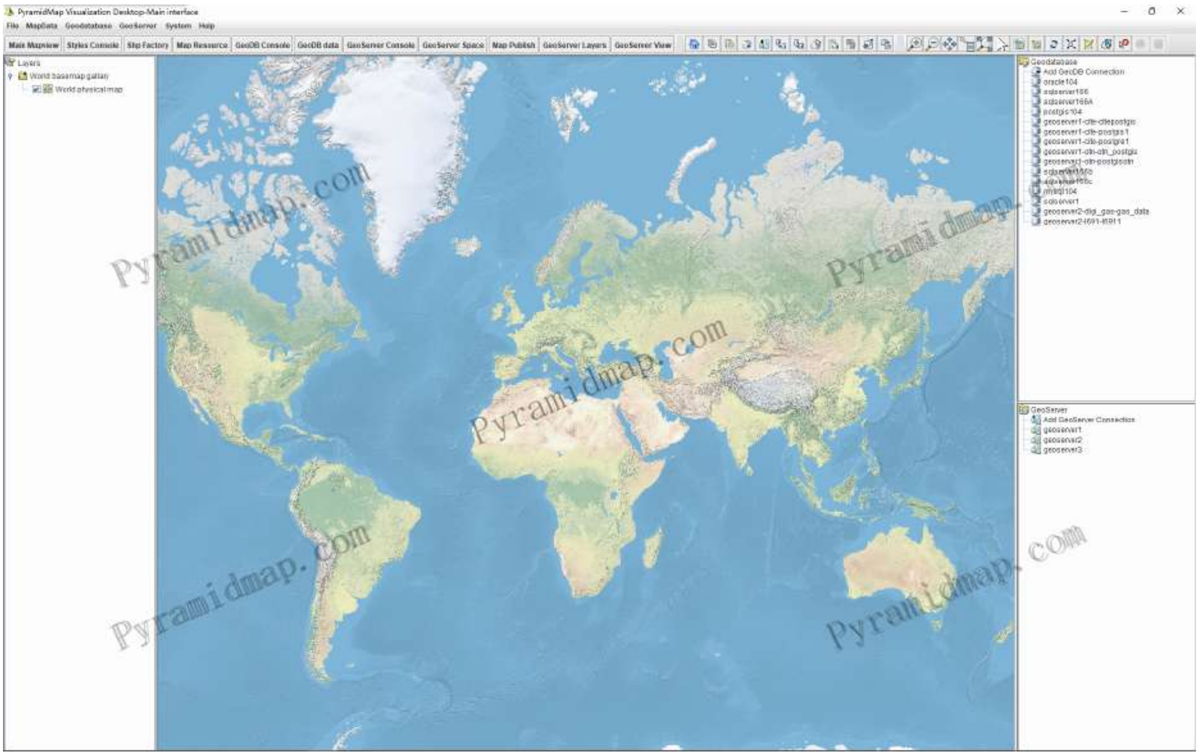


Figure 3-9：The world physical map display effect

The local details somewhere for the world physical map are shown in Figure 3-10.

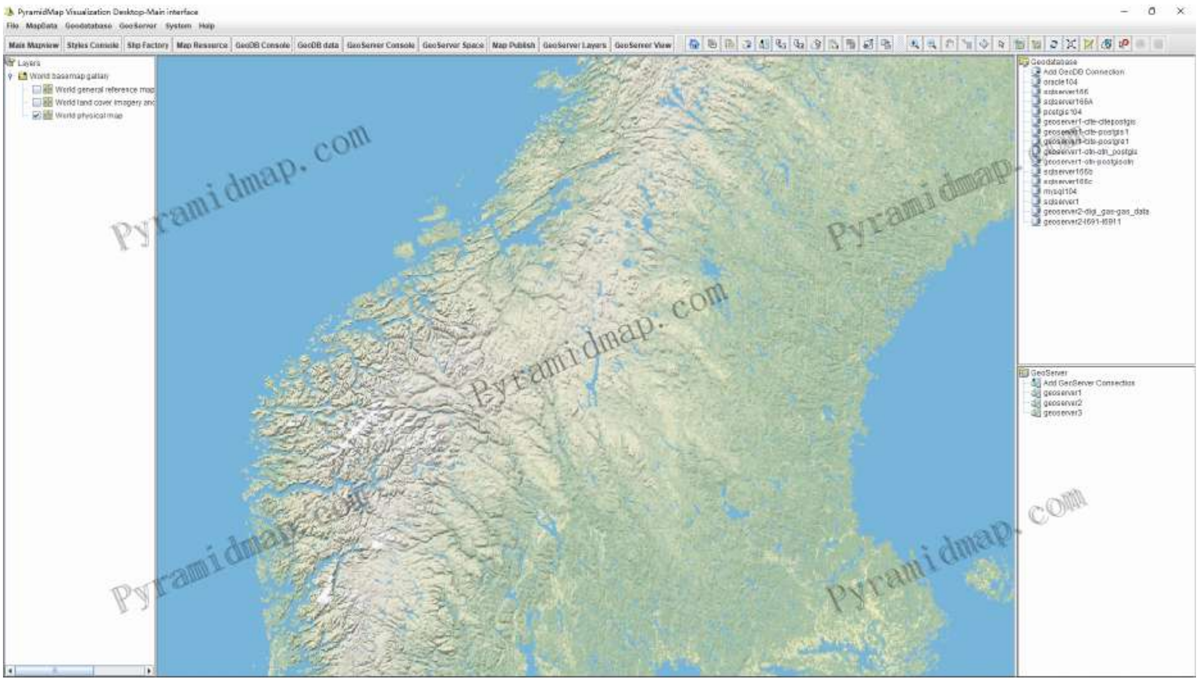


Figure 3-10：The world physical map somewhere local details

The world shaded relief map portrays surface elevation as shaded relief. This map is used as a basemap layer to add shaded relief to other GIS maps, such as the ArcGIS Online World Street Map. It is especially useful in maps that do not contain orthoimagery. The map resolution (cell size) is as follows: 30 Meters for the U.S. 90 Meters for all land areas between 60 degrees north and 56 degrees south latitude. 1 KM resolution above 60 degrees north and 56 degrees south. The shaded relief imagery was developed by Esri using GTOPO30, Shuttle Radar Topography Mission (SRTM), and National Elevation Data (NED) data from the USGS. The world shaded relief map display effect is shown in Figure 3-11.

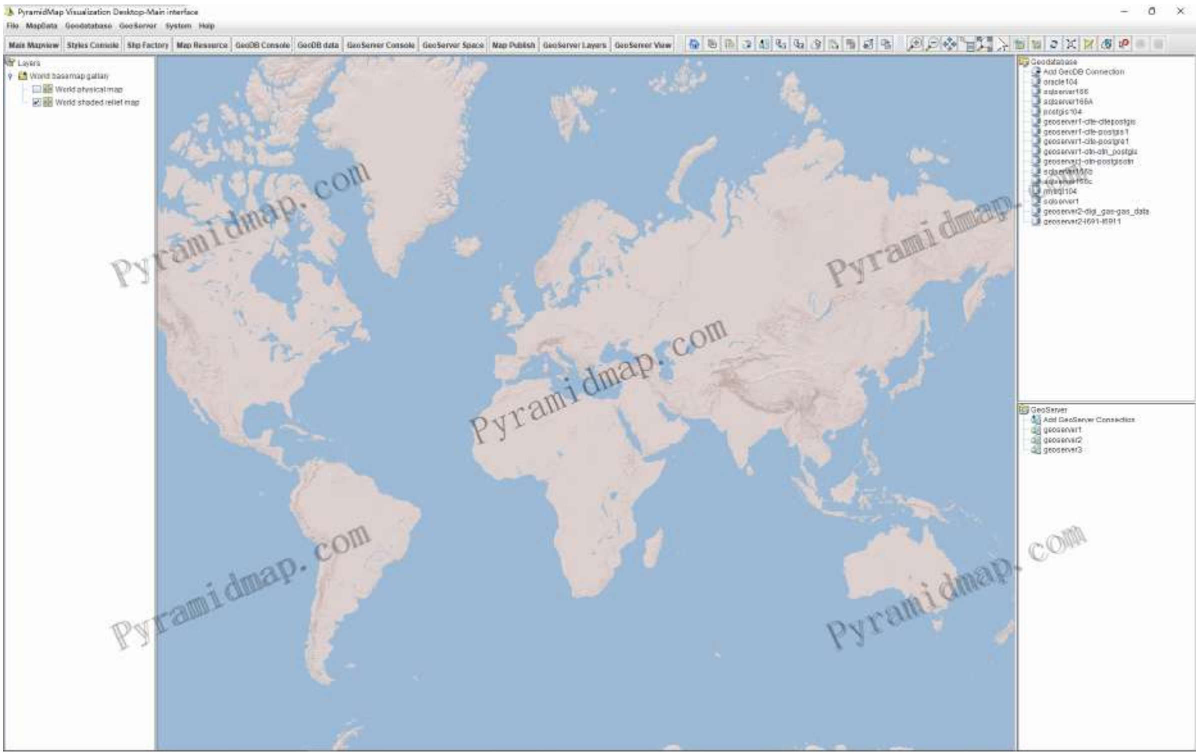


Figure 3-11： The world shaded relief map display effect

The local details somewhere for the world shaded relief map are shown in Figure 3-12.

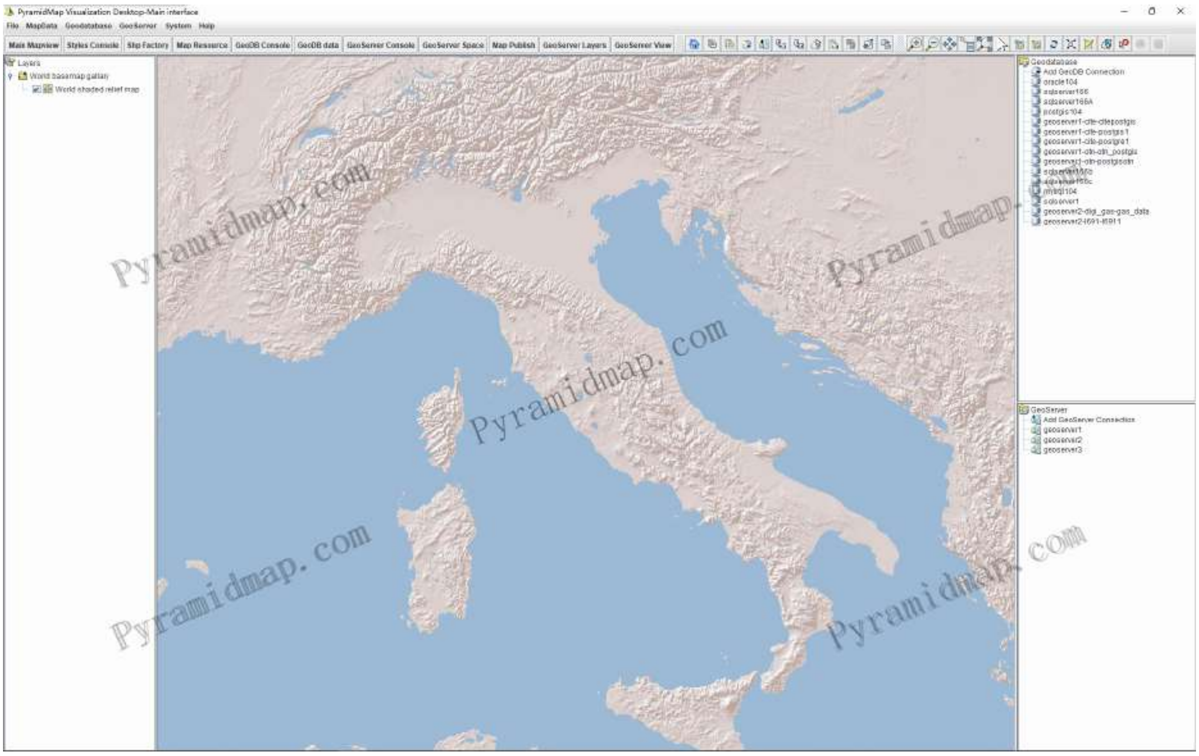


Figure 3-12： The world shaded relief map somewhere local details

The world street map presents highway-level data for the world. Street-level data includes the United States; much of Canada; Mexico; Europe; Japan; Australia and New Zealand; India; South America and Central America; Africa; and most of the Middle East. This comprehensive street map includes highways, major roads, minor roads, one-way arrow indicators, railways, water features, administrative boundaries, cities, parks, and landmarks, overlaid on shaded relief imagery for added context. Display effect is shown in Figure 3-13.

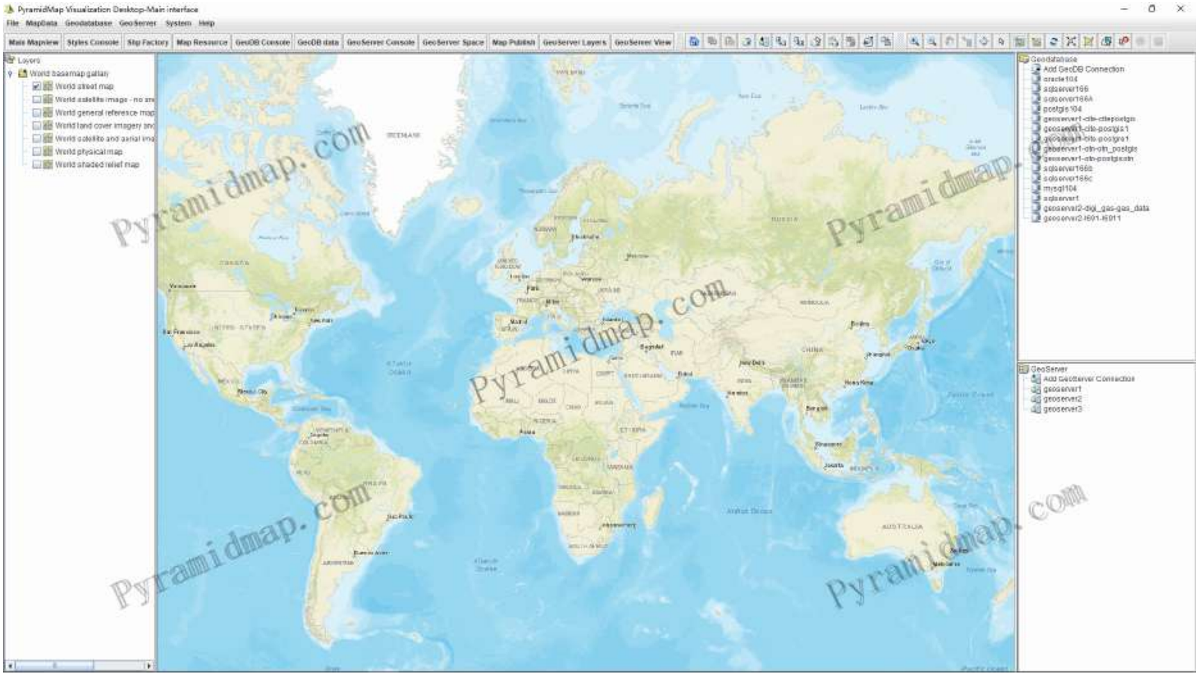


Figure 3-13：The world street map display effect

The local details somewhere for the world street map are shown in Figure 3-14.

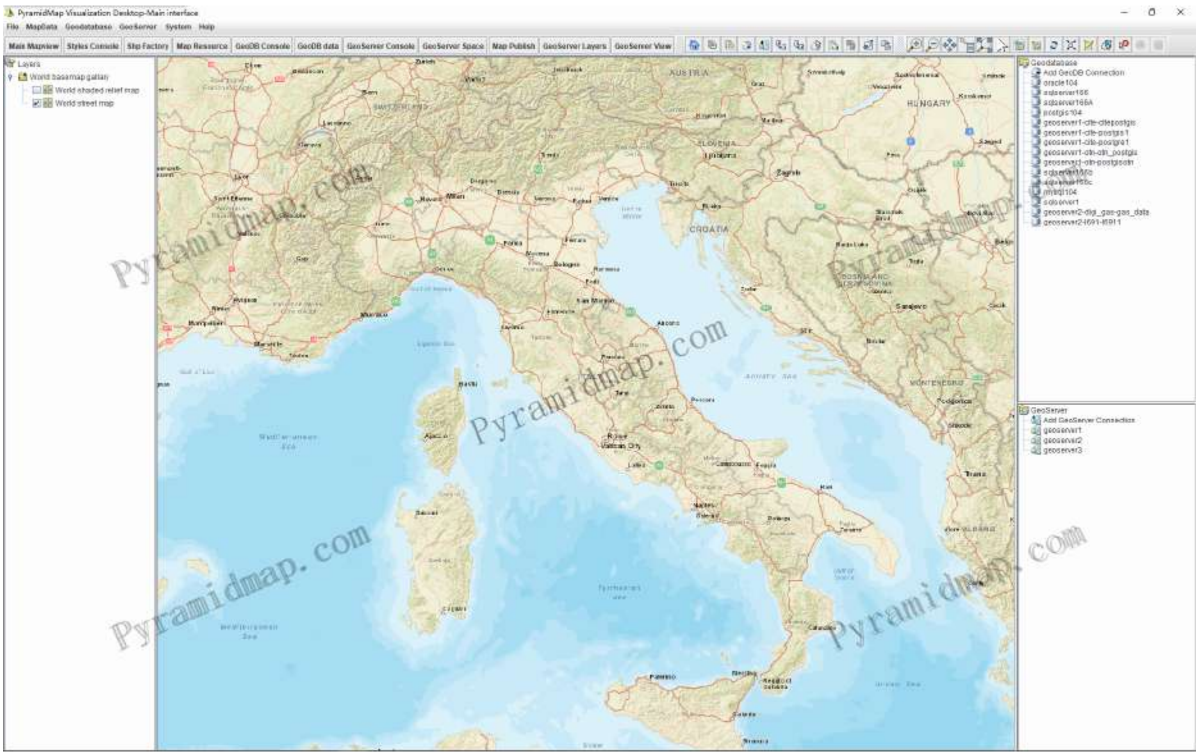


Figure 3-14：The world street map somewhere local details

The world terrain base map is designed to be used as a base map by GIS professionals to overlay other thematic layers such as demographics or land cover. The base map features shaded relief imagery, bathymetry, and coastal water features designed to provide a neutral background for other data layers. The base map currently provides coverage for the world down to a scale of ~1:1m.The world terrain base map display effect is shown in Figure 3-15.

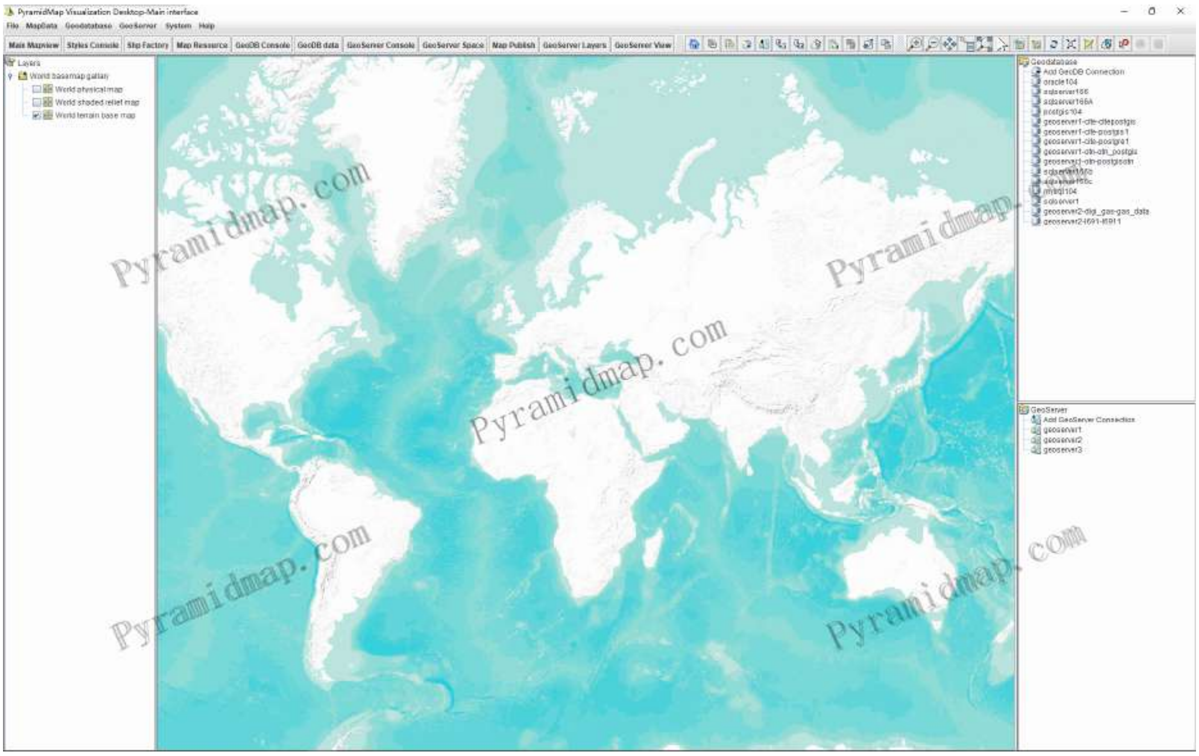


Figure 3-15：The world terrain base map display effect

The local details somewhere for the world terrain base map are shown in Figure 3-16.

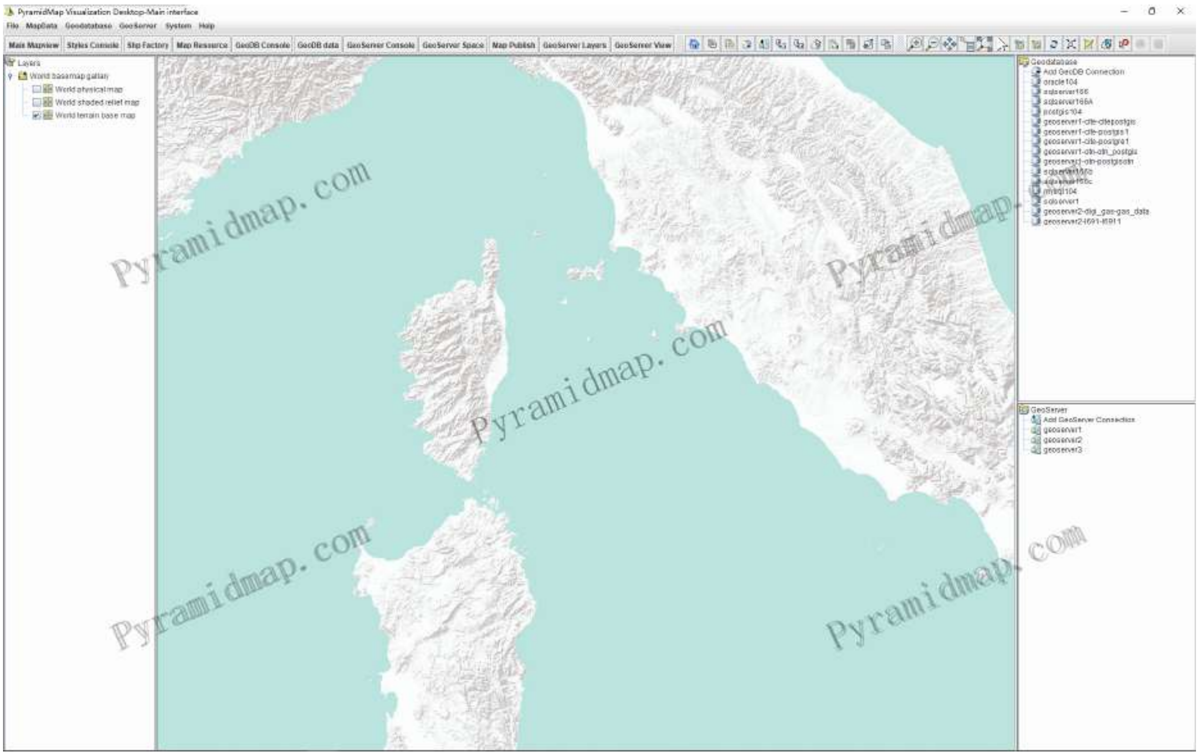


Figure 3-16：The world terrain base map somewhere local details

The world topographic base map is designed to be used as a basemap by GIS professionals and as a reference map by anyone. The map includes administrative boundaries, cities, water features, physiographic features, parks, landmarks, highways, roads, railways, and airports overlaid on land cover and shaded relief imagery for added context. The world topographic base map display effect is shown in Figure 3-17.

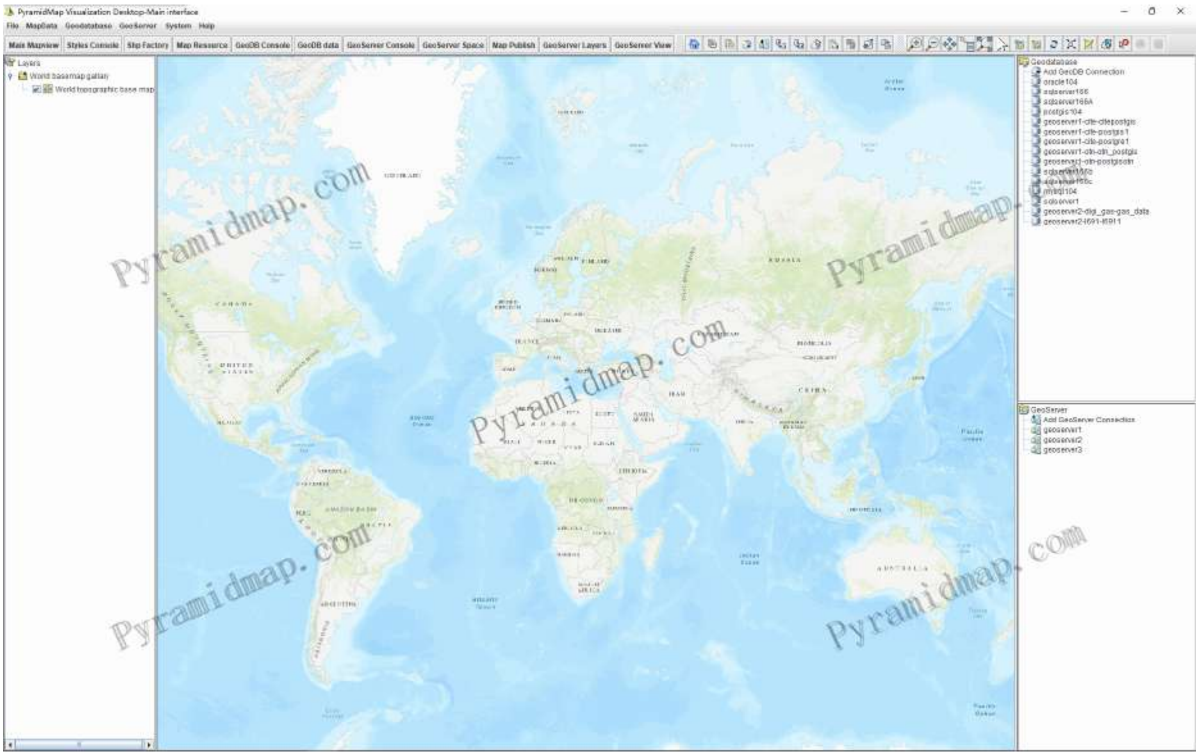


Figure 3-17: The world topographic base map display effect

The local details somewhere for the world topographic base map are shown in Figure 3-18.

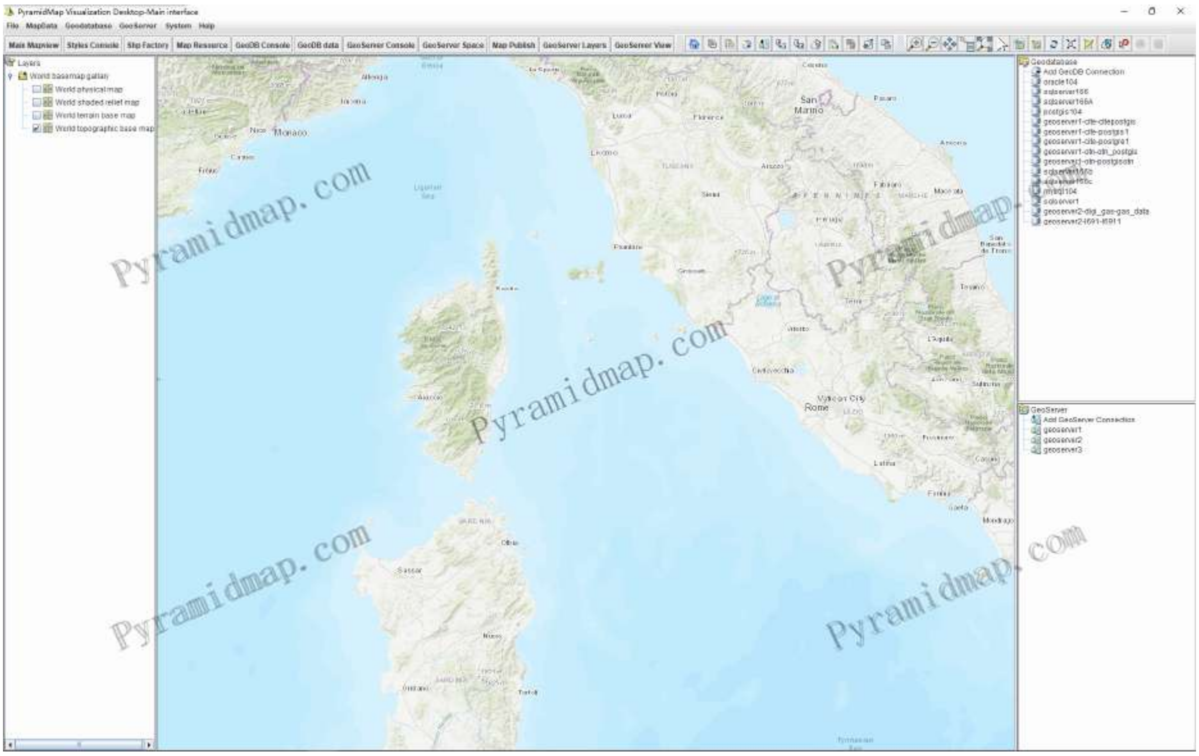


Figure 3-18: The world topographic base map somewhere local details

The world geomorphology and road network map presents highway-level data for the world. Street-level data includes the United States; much of Canada; Japan; Europe; Australia and New Zealand; India; South America and Central America; most of the Middle East; Egypt and Morocco; and parts of southern Africa including Botswana, Lesotho, Namibia, South Africa, and Swaziland. This comprehensive street map includes highways, major roads, minor roads, one-way arrow indicators, railways, water features, administrative boundaries, cities, parks, and landmarks, overlaid on shaded relief imagery for added context. This map display effect is shown in Figure 3-19.

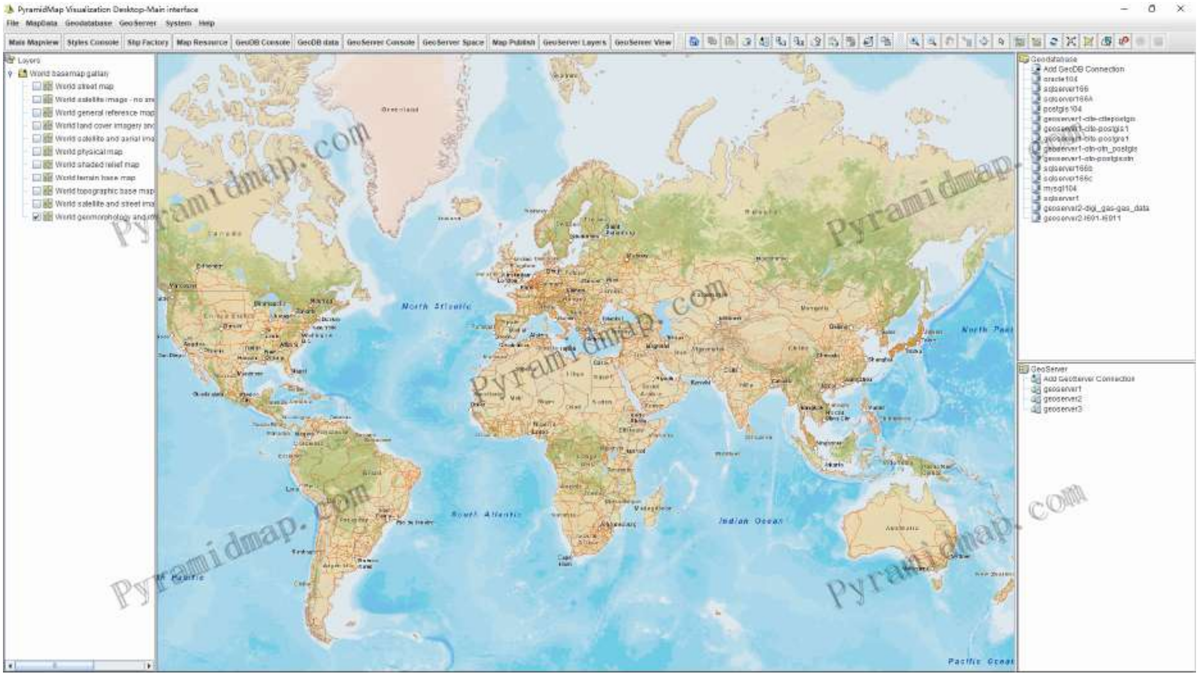


Figure 3-19：The world geomorphology and road network display effect

The local details somewhere for the world geomorphology and road network map are shown in Figure 3-20.

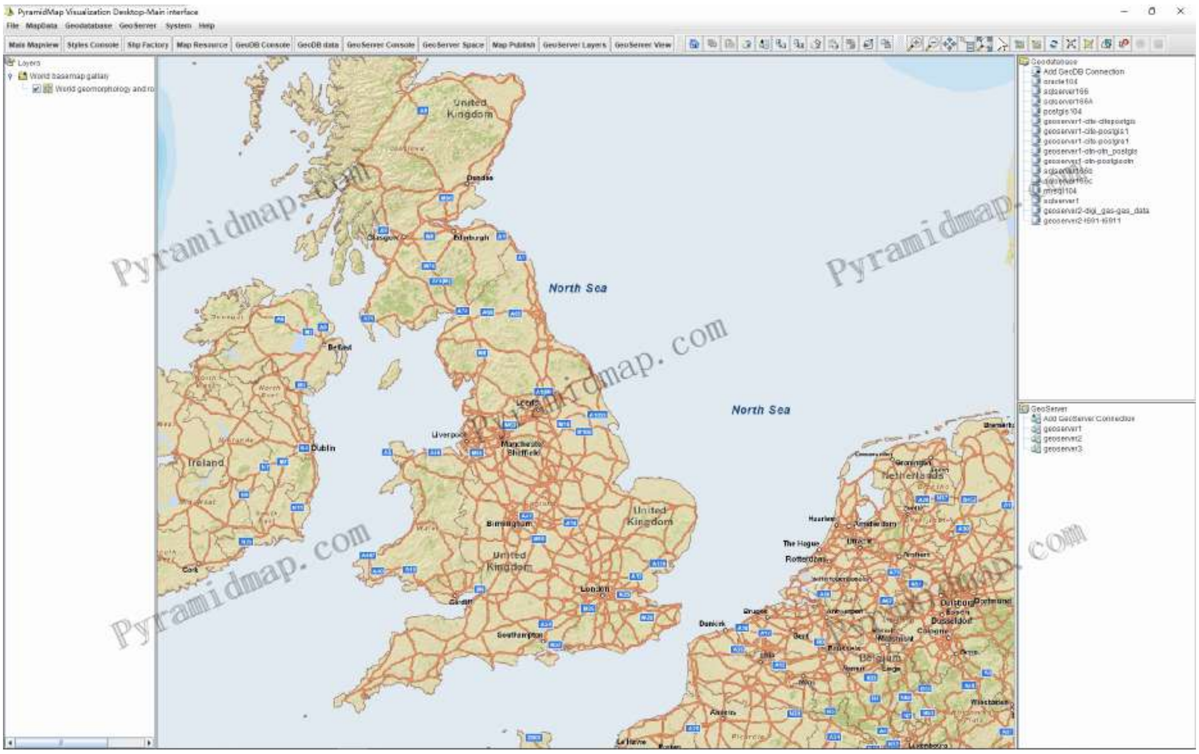


Figure 3-20：The world geomorphology and road network map somewhere local details

3.2 Load business layer

The map view supports the following map data: Shp file type, Geodatabase geographic database type, WMS, WFS, WCS, WMTS and other standardized map services from GeoServer, and various online map resources that follow standardized specifications. The layers added to the view are classified according to the data source path and displayed in the layer node on the left. The hierarchical operation is implemented through the right-click shortcut menu.

3.2.1 Load local Shp layer

Select "Load local vector layer" in the toolbar, as shown in Figure 3-21.

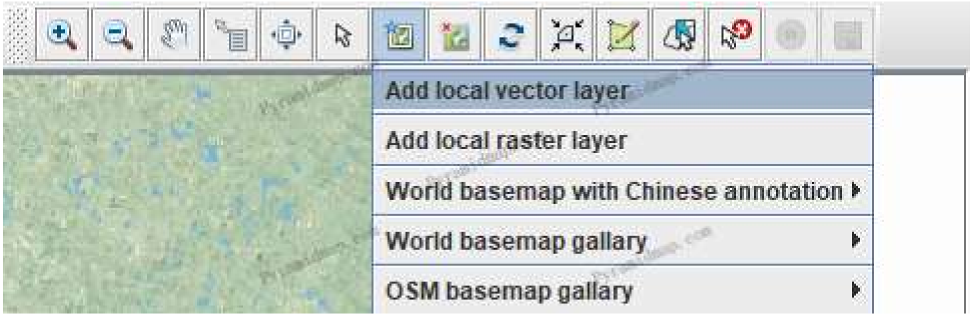


Figure 3-21：Load local vector layer

Loading the required layer through file broswer. After successful loading, it will be displayed in the map view, and the layers mount node on the left will be formed, as shown in Figure 3-22.

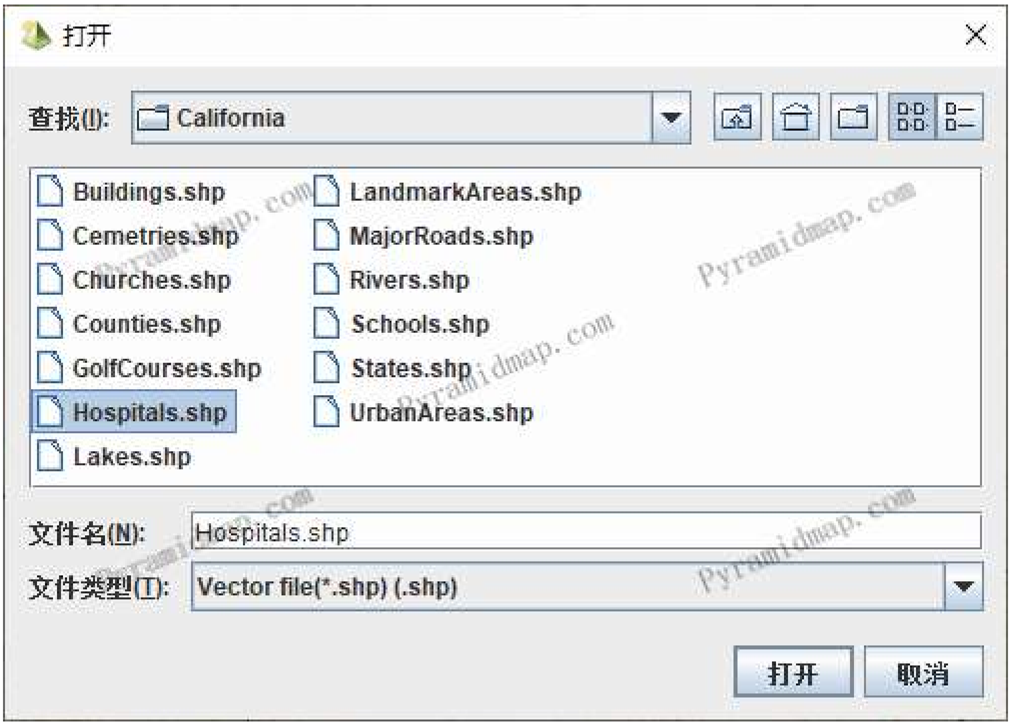


Figure 3-22：Load local shp file broswer

Then the shp selected display in the map viewer,as shown in Figure 3-23.

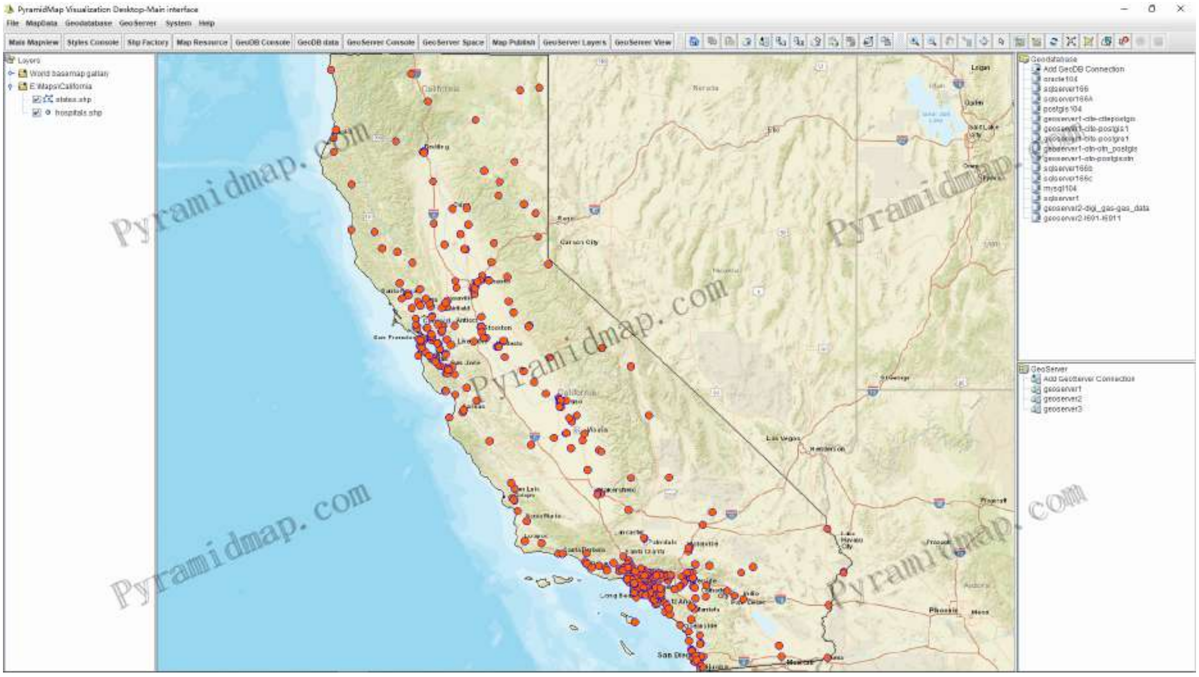


Figure 3-23：Load local shp file and display in the map viewer

At the same time, PyramidMap provides a more convenient operation, assuming the map initialization view is shown in Figure 3-24.

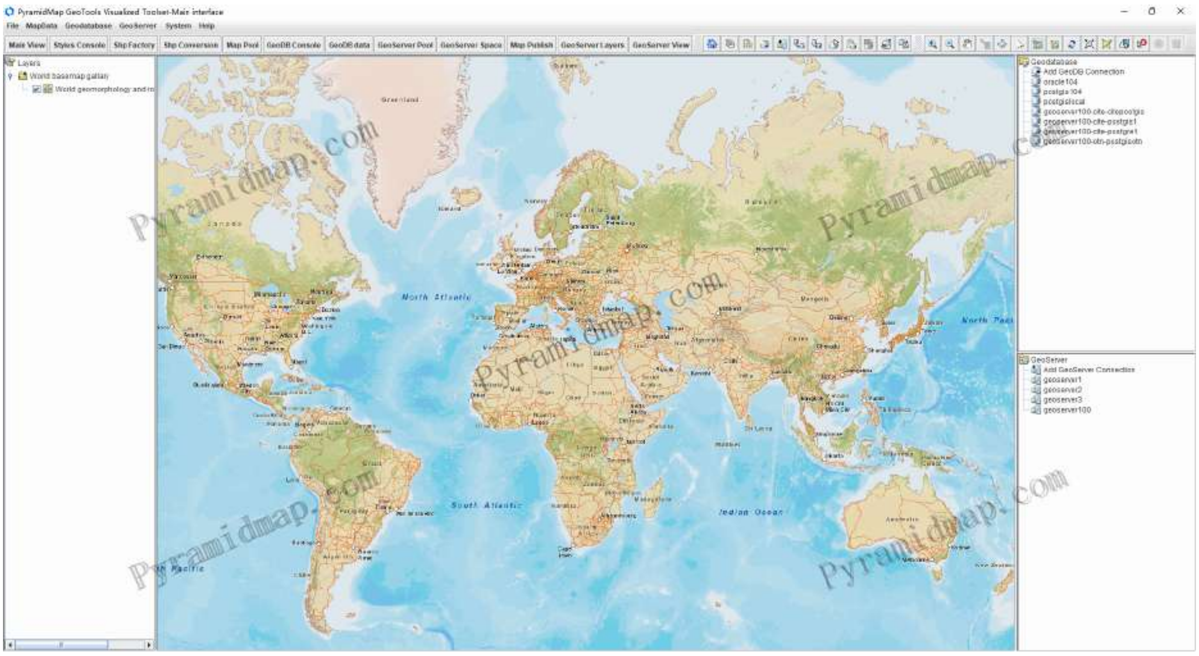


Figure 3-24：The map viewer initialization

You can drag and drop the shapefiles to the map view displaying directly, as shown in Figure 3-25.

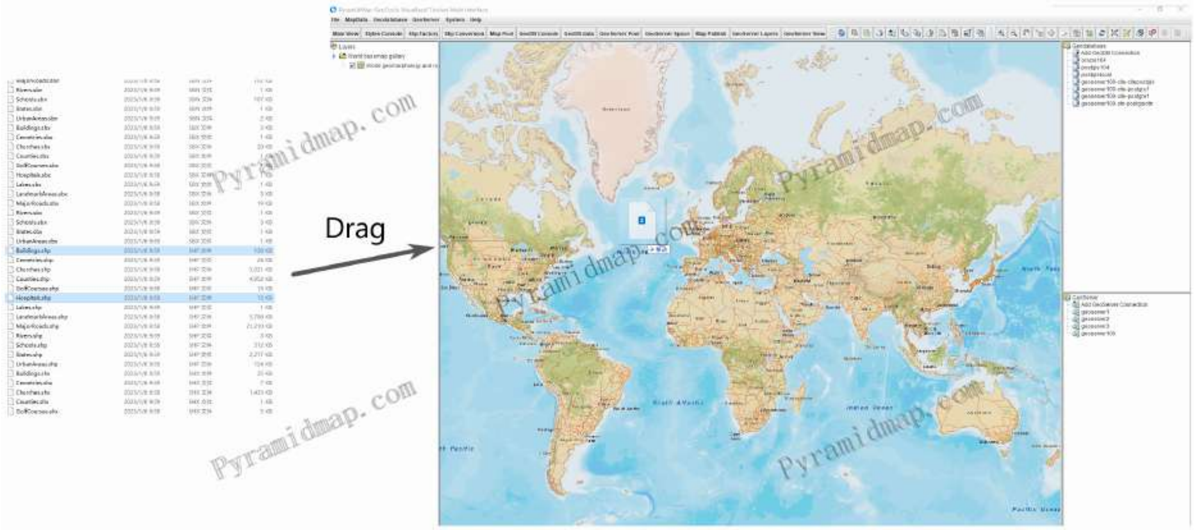


Figure 3-25： Drag and drop the shapefiles to the map view displaying directly

The display effect is shown in Figure 3-26.

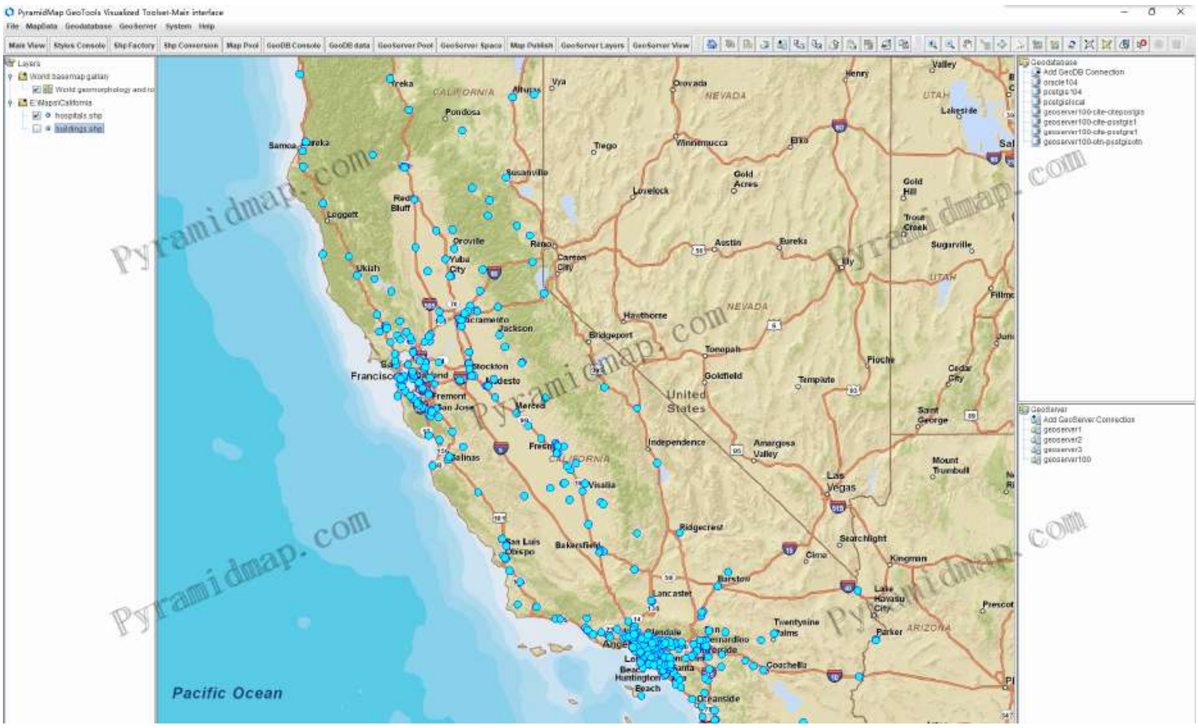


Figure 3-26： Drag and drop the shapefiles displaying

3.2.2 Load Geodatabase layer

In the node tree of the database connection pool on the main interface, double-click the database node to dynamically load its internal layer. Drag the layer node to the map view or double-click the mouse to display it. At the same time, the layer mount node on the left is formed, as shown in Figure 3-27.

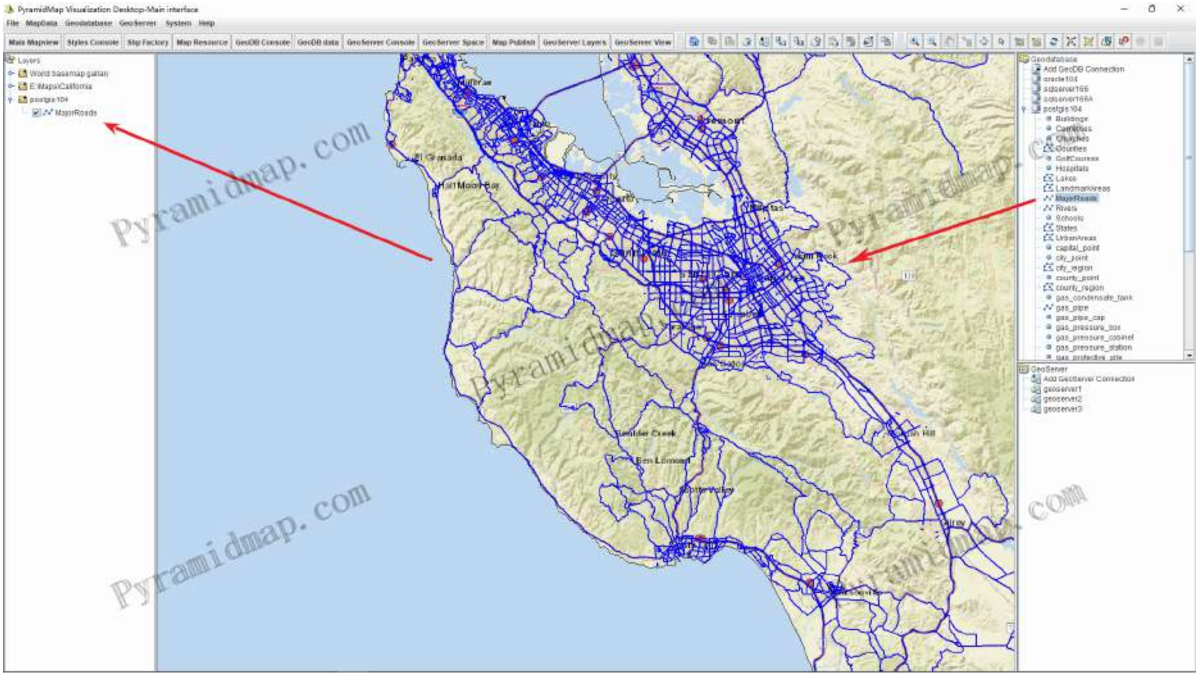


Figure 3-27：Loading and displaying vector layers in Geodatabase

3.2.3 Geodatabase node catalog and operation

On the upper right side of the main interface, there is a Geodatabase connection pool node, which provides a Geodatabase data source. Double click the database node to dynamically load its internal layers. The layer nodes can be dragged or double clicked to displaying in the map viewer. Different levels of nodes have corresponding shortcut menus. The database node menu completes database connection test, editing, layer list management, and deletion operations, as shown in Figure 3-28.

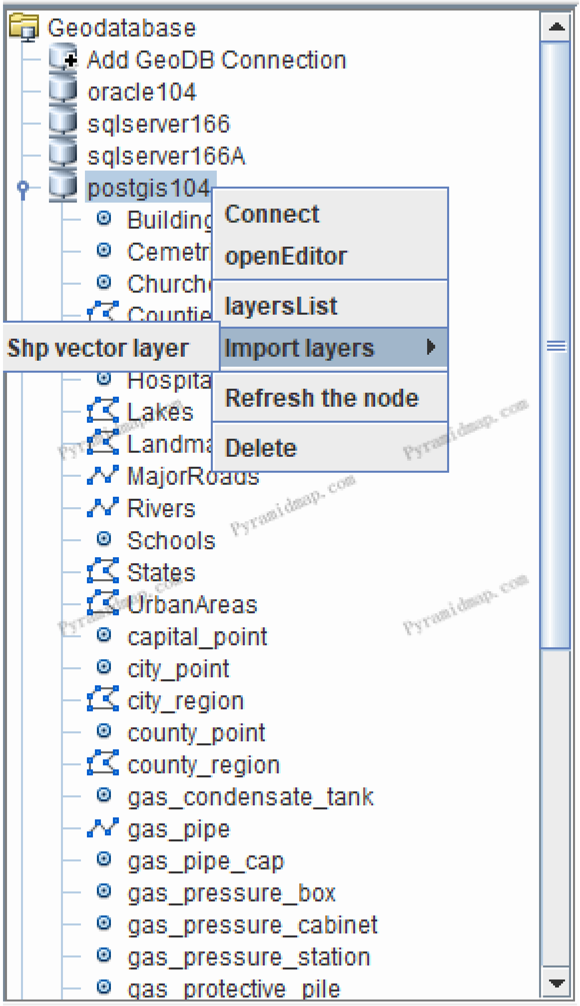


Figure 3-28： Geodatabase node popup menu

The layer node menu supports exporting database layers to Shp, Kml, Csv, GeoJson and other formats or being deleted, as shown in Figure 3-29.

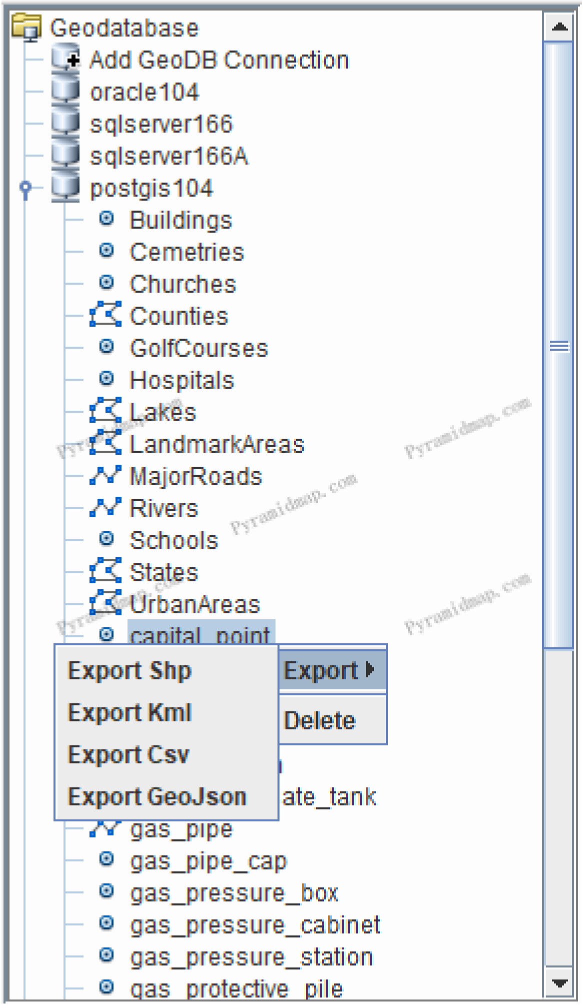


Figure 3-29： Geodatabase layer node popup menu

3.2.4 Load GeoServer layer

In the node tree of the GeoServer connection pool in the main interface, double-click the GeoServer node to dynamically load its internal workspace and its layers. The layer nodes can be dragged or double clicked to displaying in the map viewer, at the same time, the layer mount node on the left is formed, as shown in Figure 3-30.

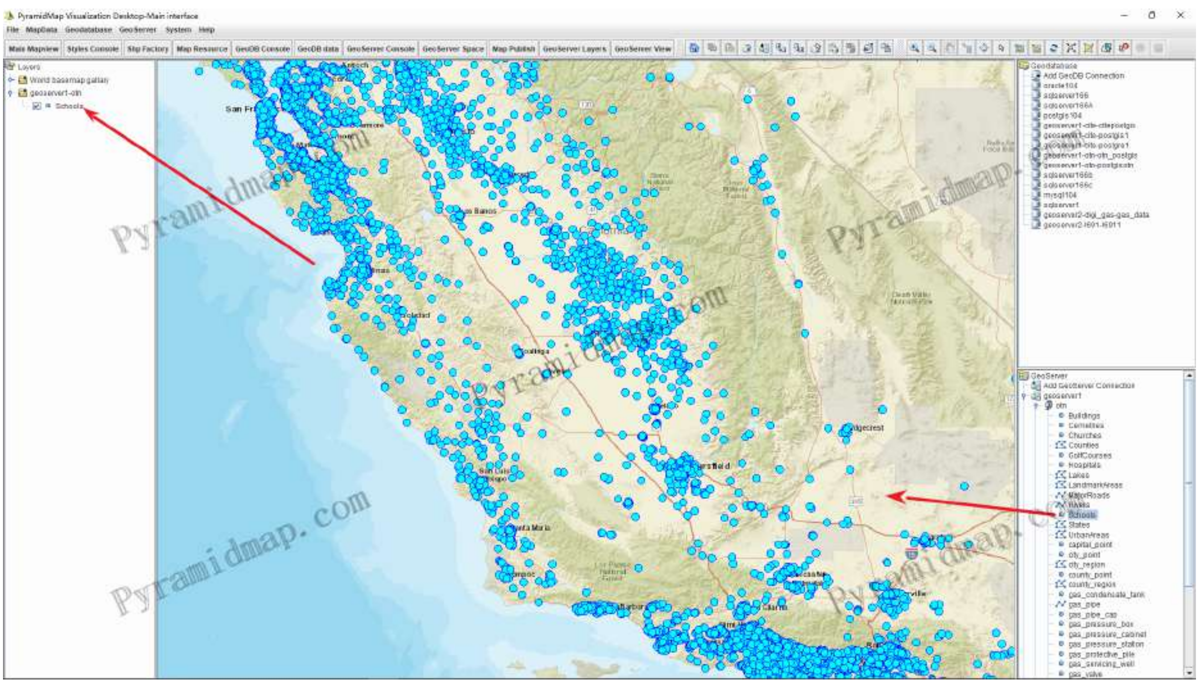


Figure 3-30： Loading and displaying vector layers in GeoServer

3.2.5 GeoServer node catalog and operation

On the lower right side of the main interface, the GeoServer connects to the pool node, providing the GeoServer data source. Double click the GeoServer node, then dynamic loading the internal workspace and layer nodes. Nodes at different levels have corresponding shortcut menus. The GeoServer node menu completes the server connection test, connection editing, obtaining server synchronization data (configuring the GeoServer workspace and data storage and its database connection), workspace management (localizing and modifying the workspace parameters and synchronizing them to the server), refreshing the layer list, connection deletion, and other operations, as shown in Figure 3-31.

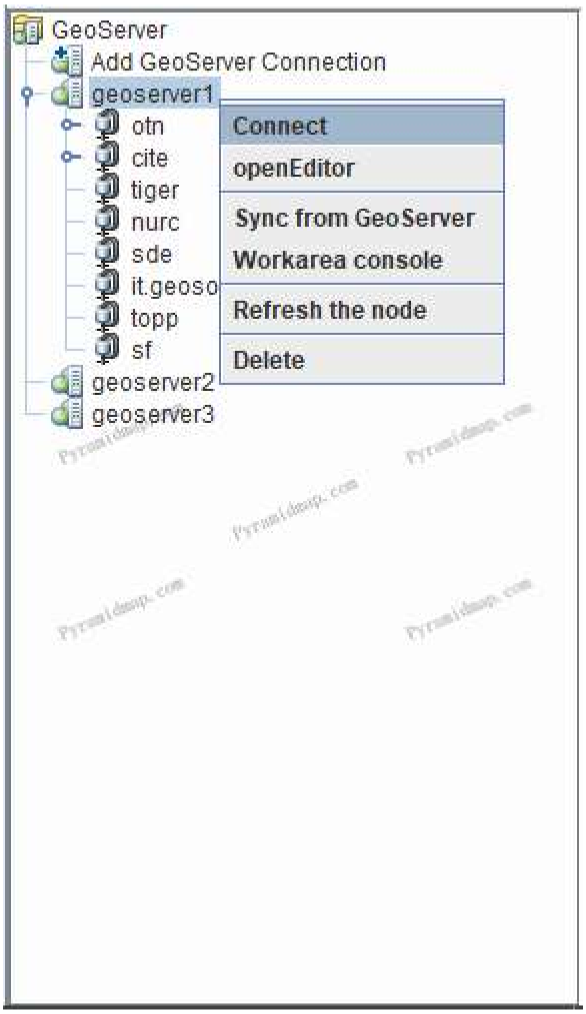


Figure 3-31： GeoServer node popup menu

Under the GeoServer node is the workspace node. The workspace node menu completes data storage management, layer node refresh, map import, workspace deletion and other operations, as shown in Figure 3-32.

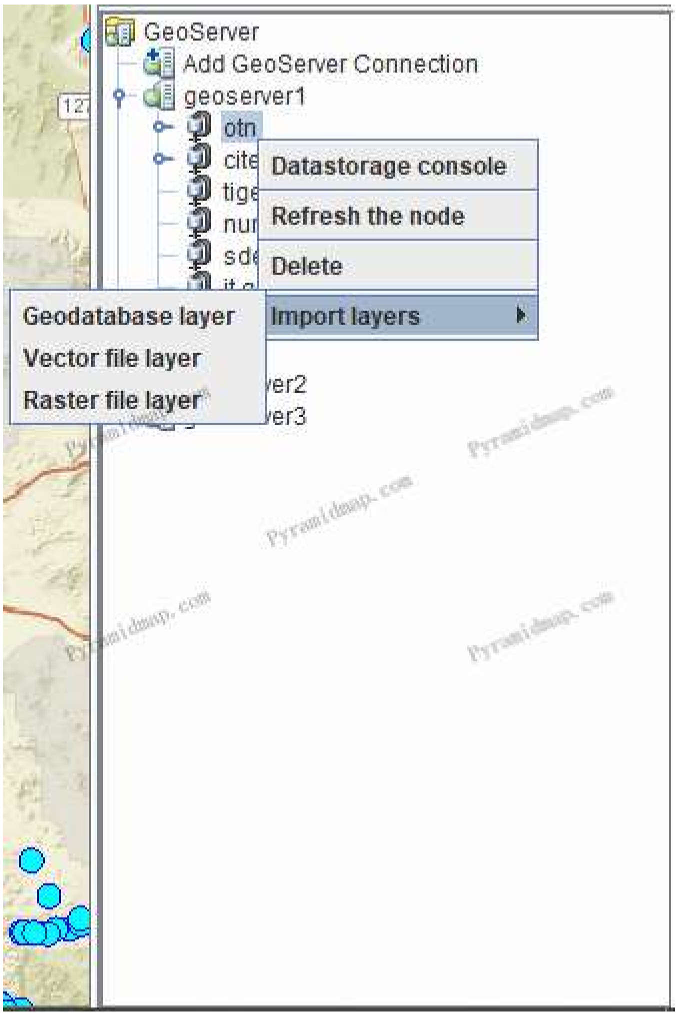


Figure 3-32： GeoServer workspace node popup menu

In particular, in the data storage list, you can edit and modify the database connection and maintain the layer list for each item in the list, including the details of the layer and the addition and deletion maintenance. The layer nodes can be dragged or double clicked to displaying in the map viewer and can be setted sld symbols, exported and deleted, as shown in Figure 3-33.

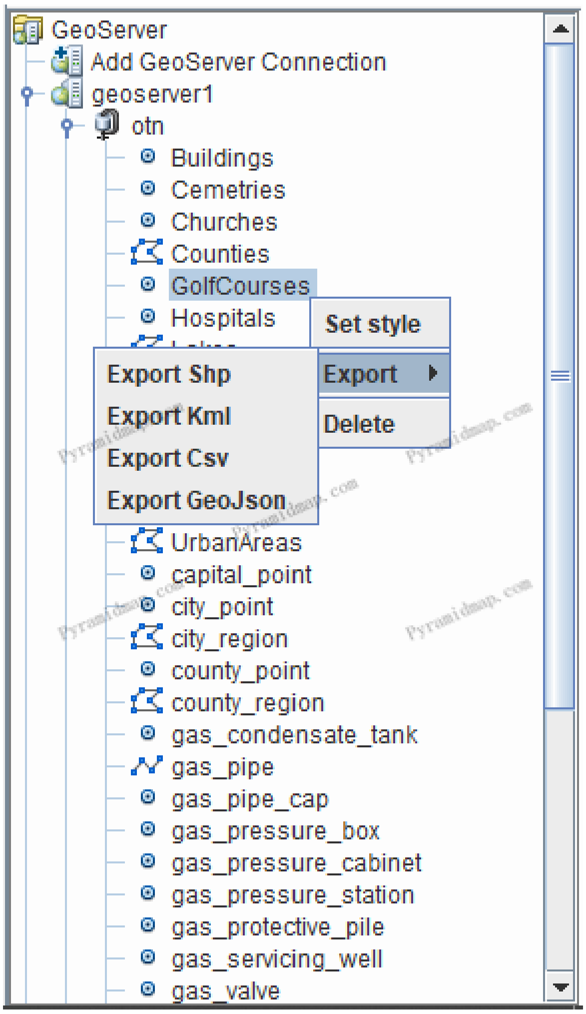


Figure 3-33： GeoServer layer node popup menu

Especially in the layer node under the GeoServer workspace, right click to set the binding relationship between the layer and the sld display symbol in the GeoServer, as shown in Figure 3-34.

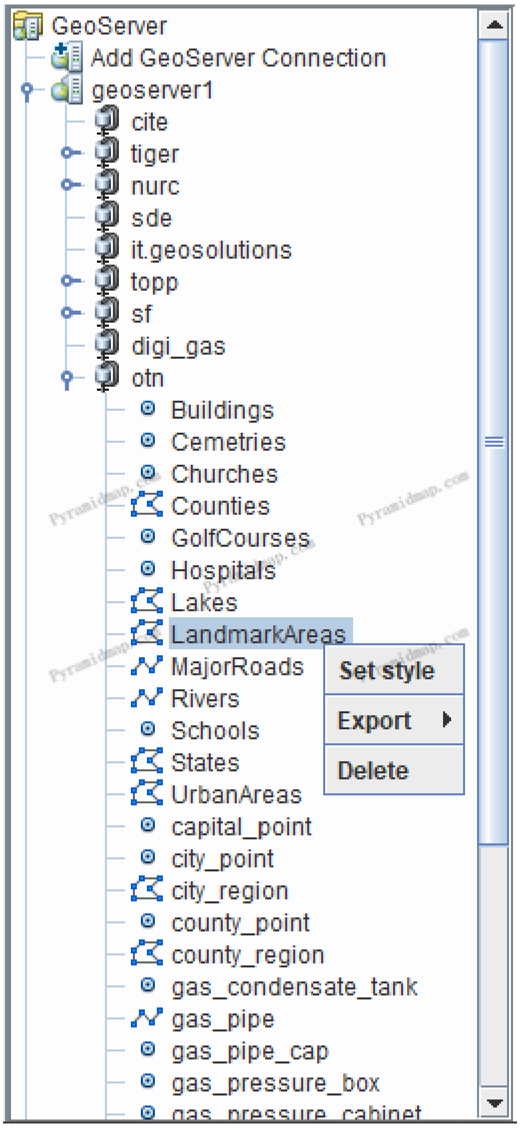


Figure 3-34：Set the binding relationship between the layer and the sld symbol in the GeoServer

PyramidMap will return the sld symbol selection list of the same geometry type on the server according to the selected layer, as shown in Figure 3-35.

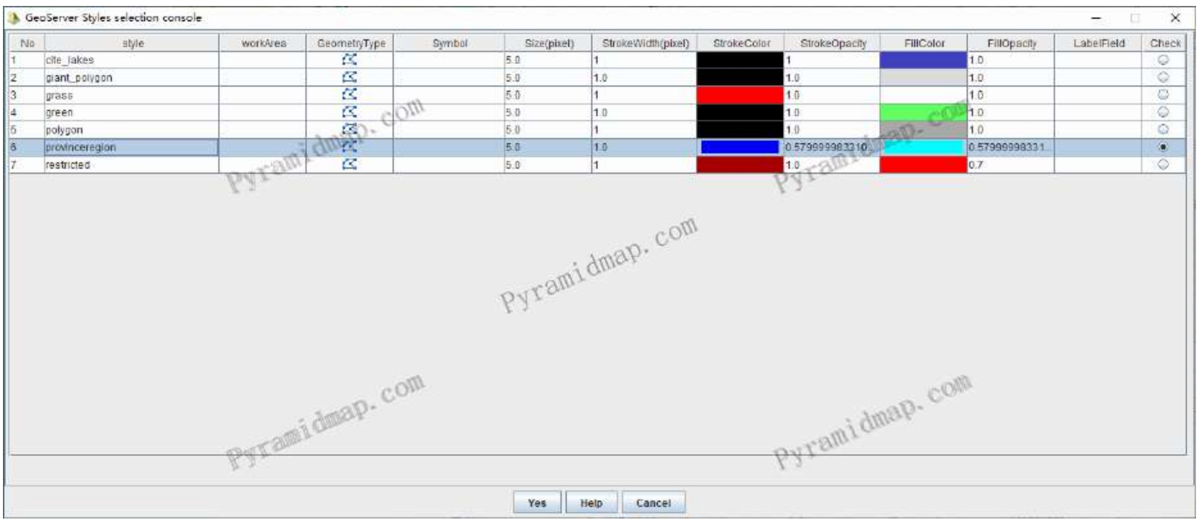


Figure 3-35：Return sld symbol list of the same geometry type on the server according to the selected layer

Click "OK", the selected layer and sld symbol are bound successfully. Subsequently, the WMS service of this layer output by the GeoServer server will be rendered according to this symbol, as shown in Figure 3-36.

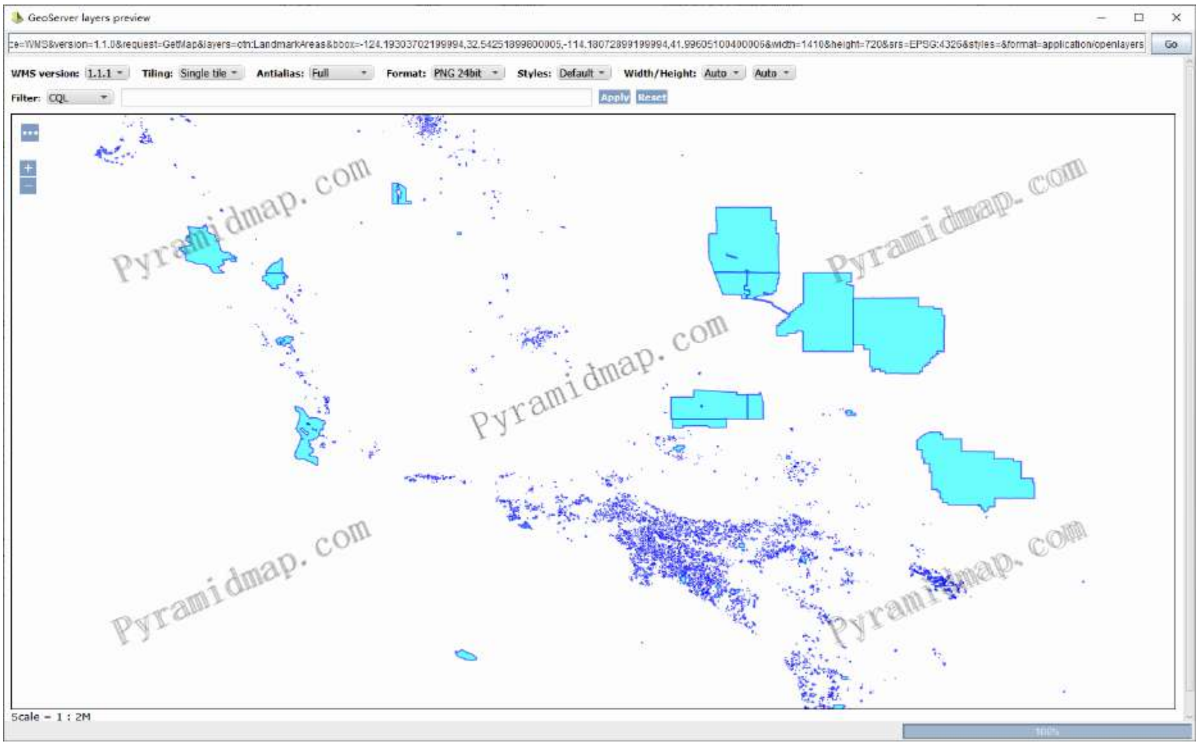


Figure 3-36： The WMS service of this layer output by the GeoServer server will be rendered according to this symbol

3.3 Visible layer node operations

3.3.1 Layer control

The layers in the main map view are classified and managed in checkbox tree nodes. It supports the popup menu and checked visible controlling on each layer node, and the corresponding operations are implemented according to the layer type. The following is layer checkbox node classification and operation for vector layers, as shown in Figure 3-37.

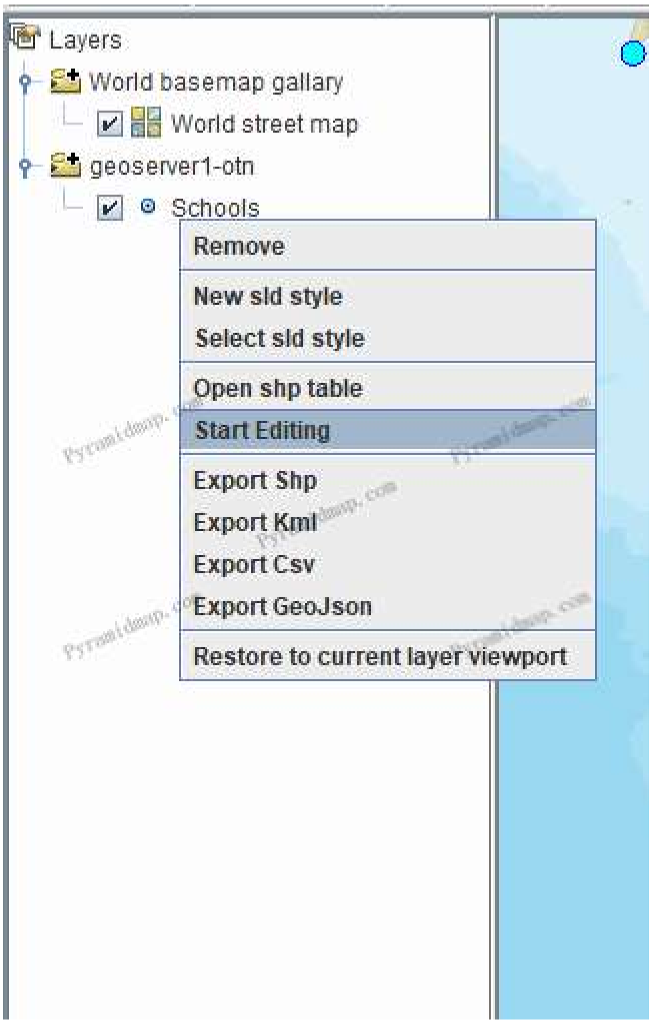


Figure 3-37： Vector layer checkbox node classification and operation

The following is layer checkbox node classification and operation for raster layer, as shown in Figure 3-38.

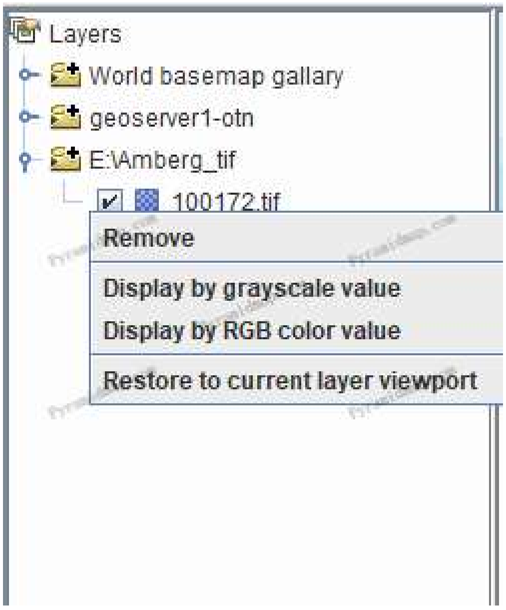


Figure 3-38： Raster layer checkbox node classification and operation

You can control the visibility of layers and the data processing of different types of layers with the classification and operation of layer checkbox nodes.

3.3.2 Symbol rendering

Select New sld Symbol from the shortcut menu of the vector layer display node to enter the sld symbol definition interface, as shown in Figure 3-39.

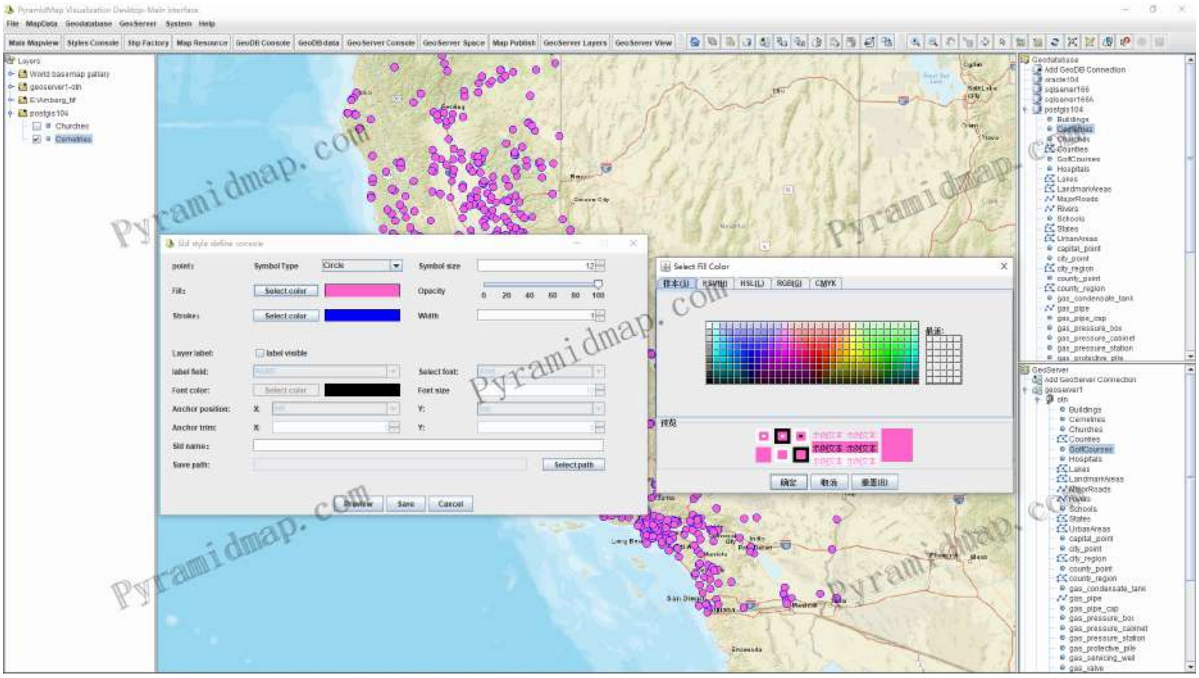


Figure 3-39：Definition and effect of map Sld symbols

With this operation, create a style definition file for point (Point, MultiPoint), line (LineString, MultiLineString), polygon (Polygon, MultiPolygon) type layers, and set the annotation field and font type, size, color, and annotation location. After the effect preview is satisfactory, save it as an sld file and include it in the PyramidMap resource pool for maintenance, The SLD resource pool maintained by PyramidMap client can provide feature symbol selection for layers. Select the SLD symbol on the layer node, as shown in Figure 3-40.

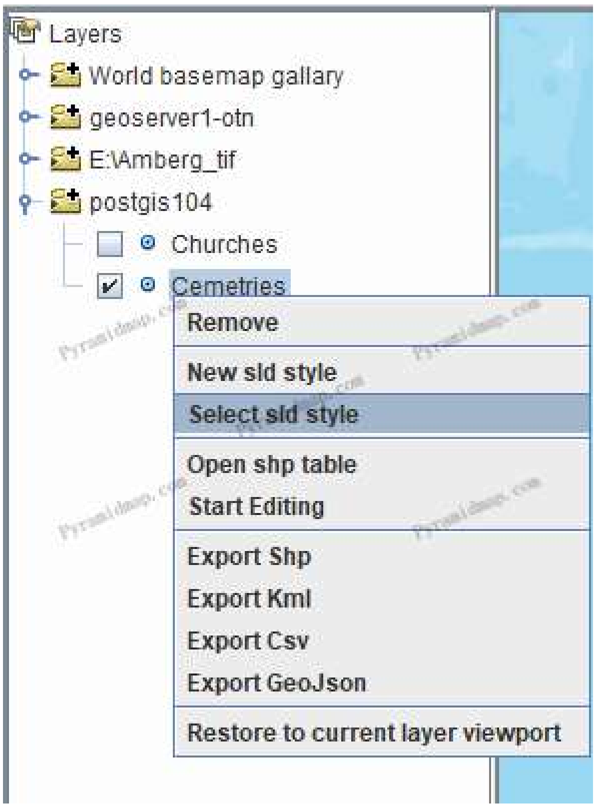


Figure 3-40：Select sld symbol on layer node

PyramidMap will automatically match the Sld data according to the geometry type of the selected layer, and provide a selection list of sld resources of the same geometry type, as shown in Figure 3-41.

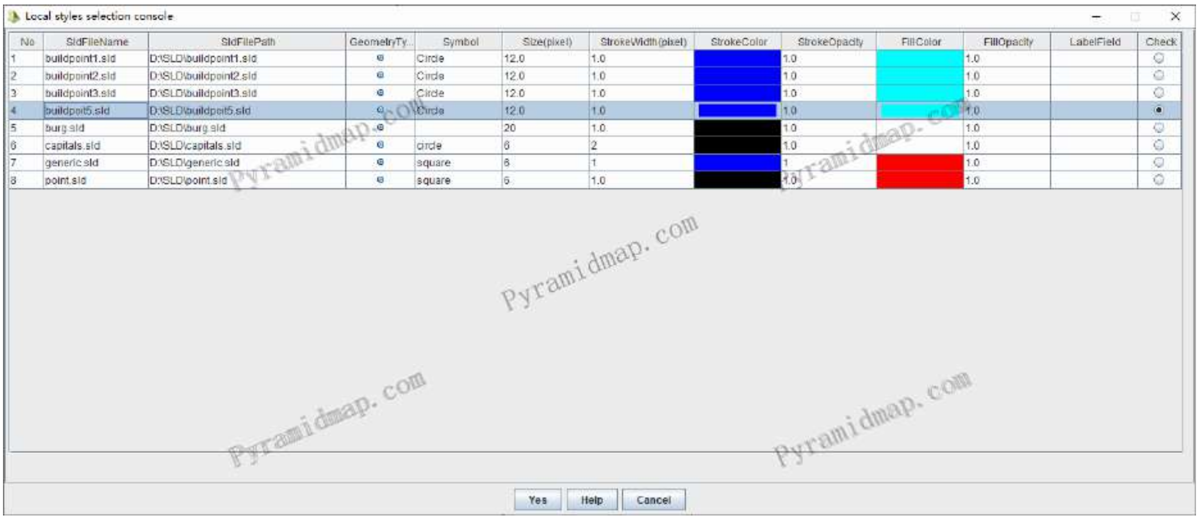


Figure 3-41： SLD resource list of the same geometry type as the selected layer

The rendering effect of the selected sld is applied to the layer, as shown in Figure 3-42.

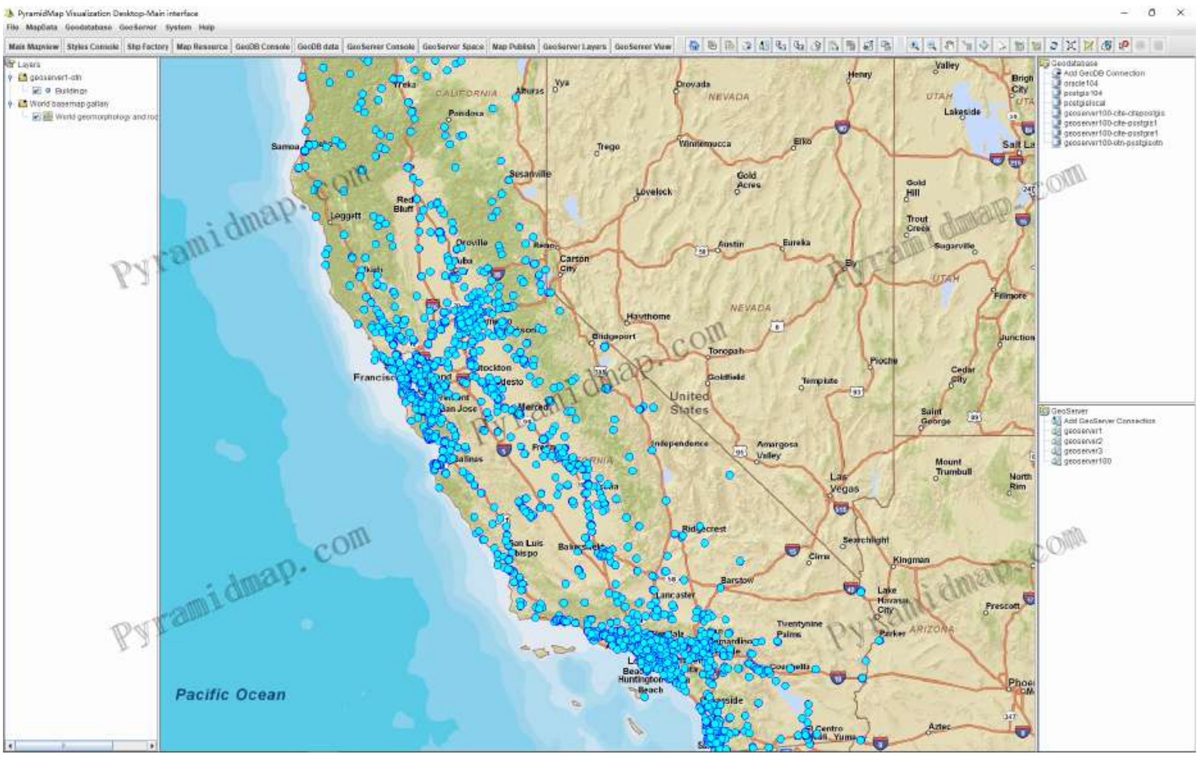


Figure 3-42： Rendering effect of layer after applying the selected sld

3.3.3 Feature data table

Select the "Open shp table" option in the layer node on the left, as shown in Figure 3-43.

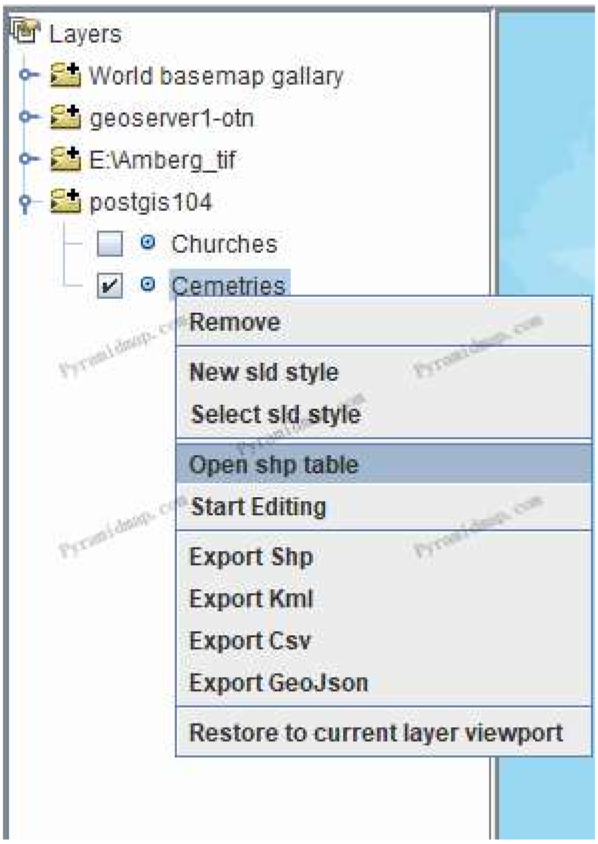


Figure 3-43：Open shp table

The feature data table and query page of the selected layer are shown in Figure 3-44.

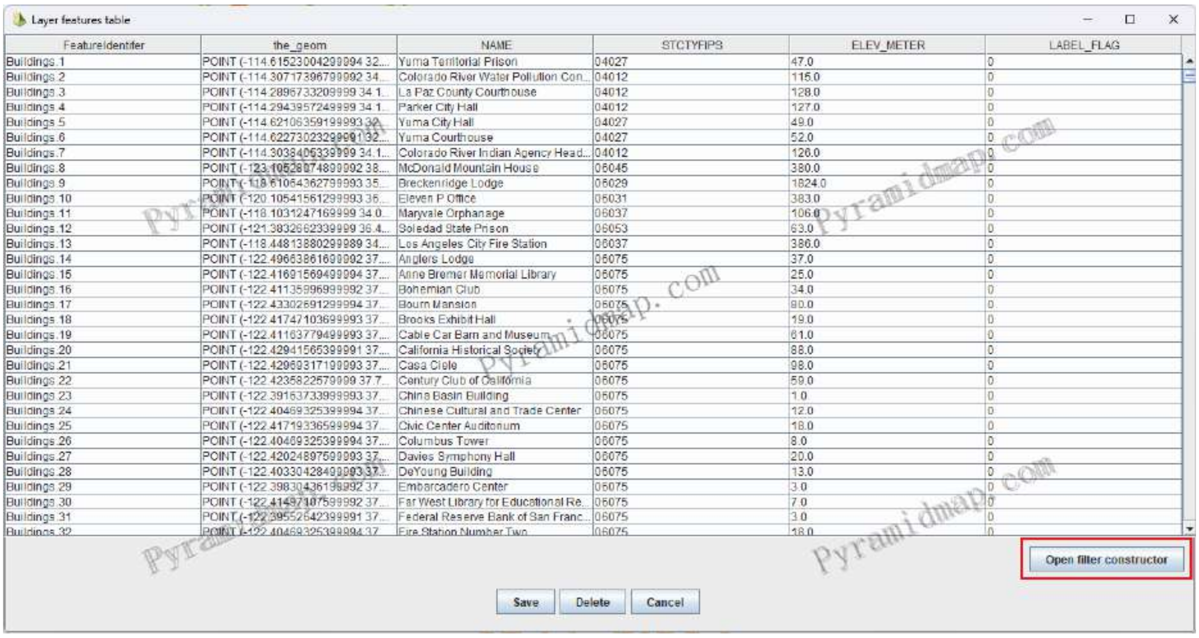


Figure 3-44：Shp table and query interface

In the layer data table interface, you can build a combined query condition based on all fields through the query tool to realize simple and complex queries on layers, as shown in Figure 3-45.

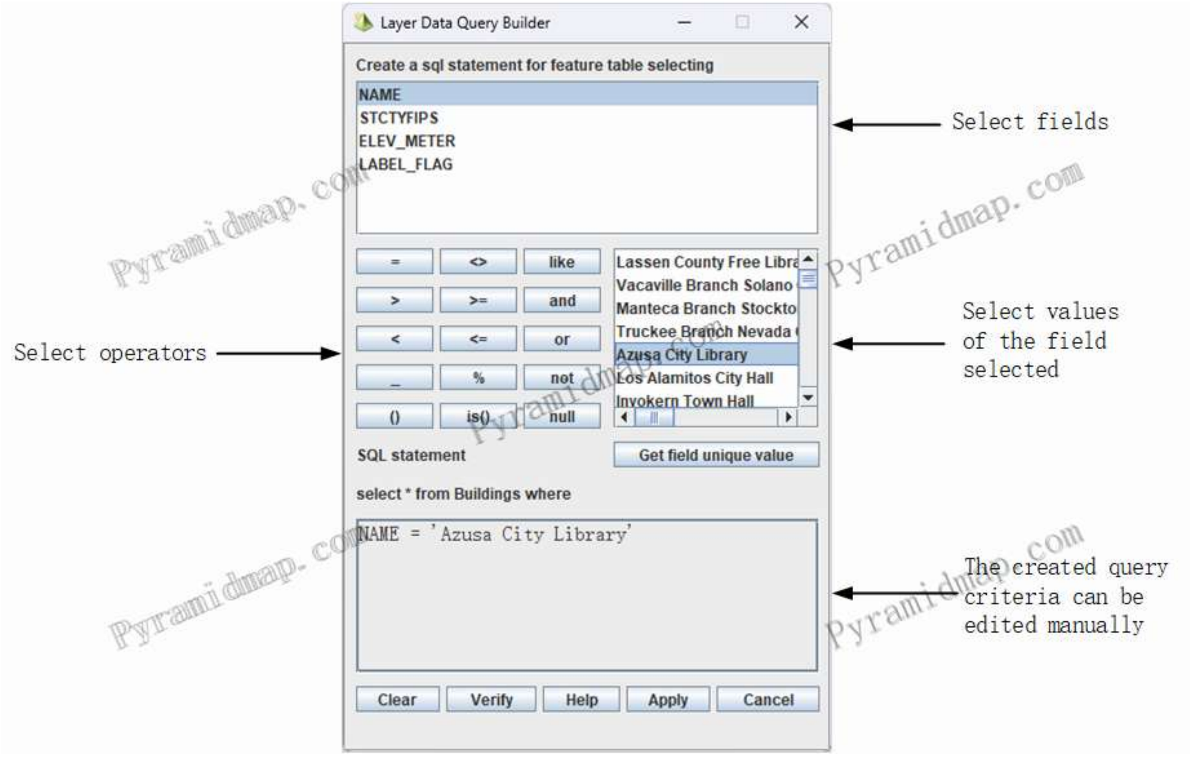


Figure 3-45： Layer table query constructor

In the query constructor, you can verify the validity of the built query statements, as shown in Figure 3-46.

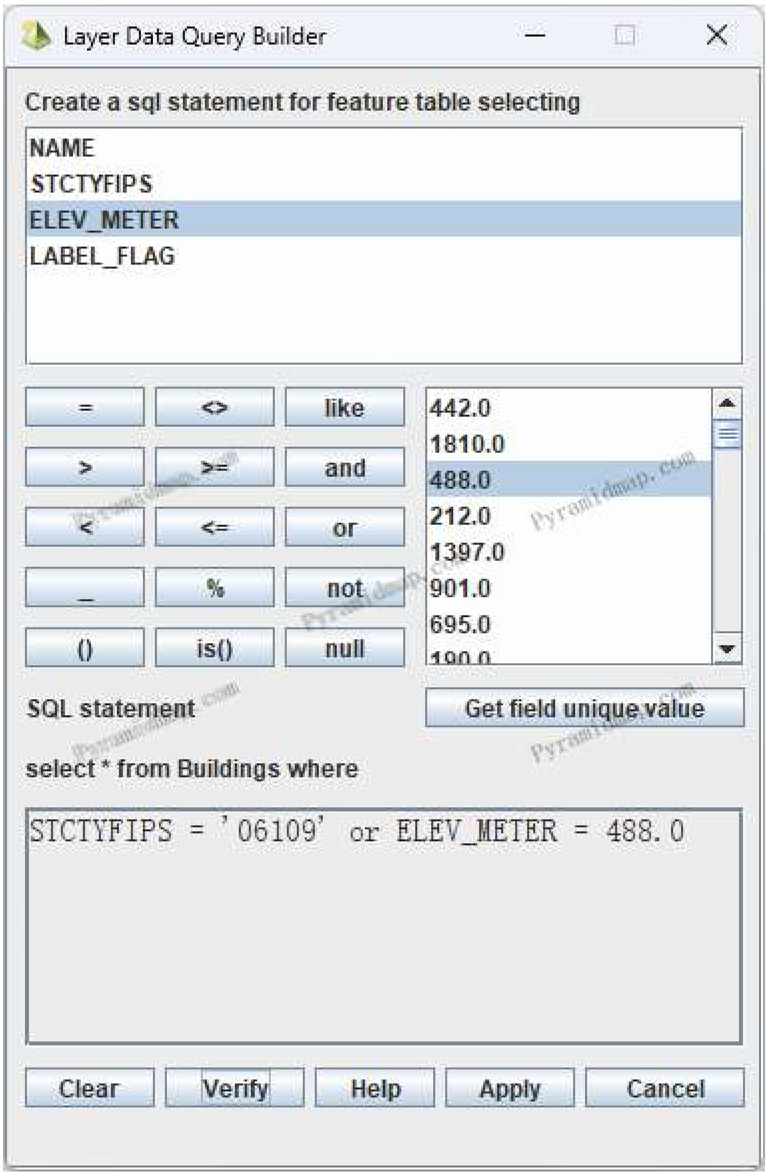
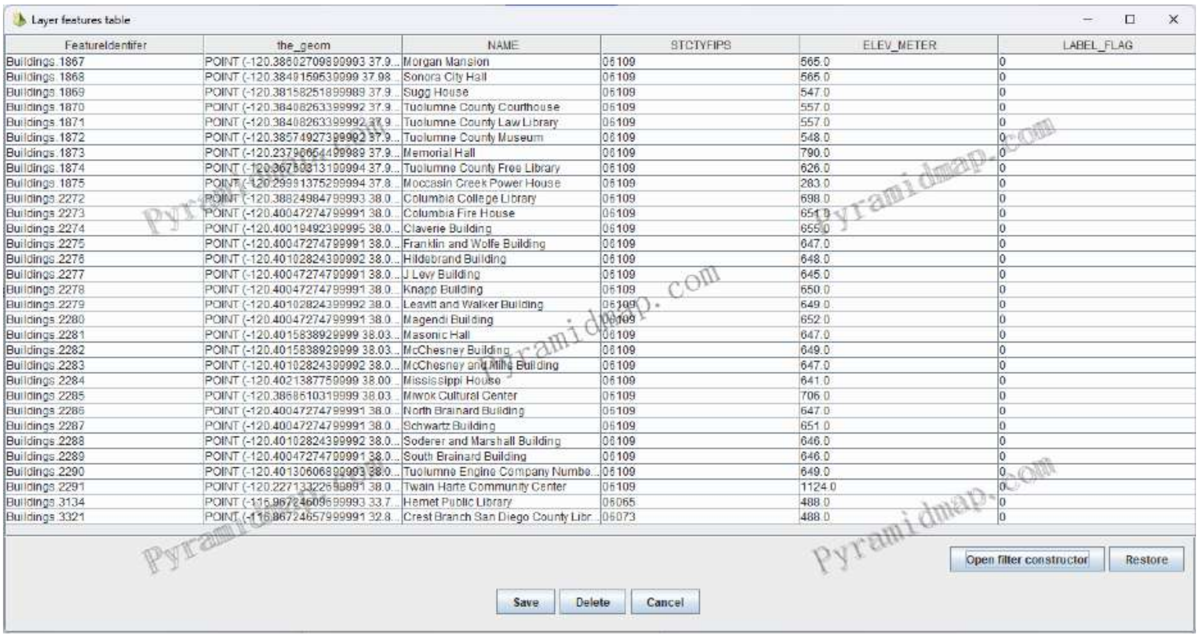


Figure 3-46：Query statement validation

The validation results are shown in Figure 3-47.

Figure 3-47：Query statement validation results

Click "Apply" to execute the query. Based on the above construction conditions, the data table query results are shown in Figure 3-48.



Layer features table

FeatureIdentifer	the_geom	NAME	STCTYFIPS	ELEV_METER	LABEL_FLAG
Buildings.1867	POINT (-120.38602709899993 37.9...	Morgan Mansion	06109	565.0	0
Buildings.1868	POINT (-120.3849159539999 37.98...	Sonora City Hall	06109	565.0	0
Buildings.1869	POINT (-120.38158251899989 37.9...	Sugg House	06109	547.0	0
Buildings.1870	POINT (-120.38408263399992 37.9...	Tuolumne County Courthouse	06109	557.0	0
Buildings.1871	POINT (-120.38408263399992 37.9...	Tuolumne County Law Library	06109	557.0	0
Buildings.1872	POINT (-120.38574927399992 37.9...	Tuolumne County Museum	06109	548.0	0
Buildings.1873	POINT (-120.23796654499989 37.9...	Memorial Hall	06109	790.0	0
Buildings.1874	POINT (-120.36758313199994 37.9...	Tuolumne County Free Library	06109	626.0	0
Buildings.1875	POINT (-120.29991375299994 37.8...	Moccasin Creek Power House	06109	283.0	0
Buildings.2272	POINT (-120.38824984799993 38.0...	Columbia College Library	06109	698.0	0
Buildings.2273	POINT (-120.40047274799991 38.0...	Columbia Fire House	06109	651.0	0
Buildings.2274	POINT (-120.40019492399995 38.0...	Claverie Building	06109	655.0	0
Buildings.2275	POINT (-120.40047274799991 38.0...	Franklin and Wolfe Building	06109	647.0	0
Buildings.2276	POINT (-120.40102824399992 38.0...	Hildebrand Building	06109	648.0	0
Buildings.2277	POINT (-120.40047274799991 38.0...	J Levy Building	06109	645.0	0
Buildings.2278	POINT (-120.40047274799991 38.0...	Knapp Building	06109	650.0	0
Buildings.2279	POINT (-120.40102824399992 38.0...	Leavitt and Walker Building	06109	649.0	0
Buildings.2280	POINT (-120.40047274799991 38.0...	Magendi Building	06109	652.0	0
Buildings.2281	POINT (-120.40158389299999 38.03...	Masonic Hall	06109	647.0	0
Buildings.2282	POINT (-120.40158389299999 38.03...	McChesney Building	06109	649.0	0
Buildings.2283	POINT (-120.40102824399992 38.0...	McChesney and Mills Building	06109	647.0	0
Buildings.2284	POINT (-120.4021387759999 38.00...	Mississippi House	06109	641.0	0
Buildings.2285	POINT (-120.38686103199999 38.03...	Miwok Cultural Center	06109	706.0	0
Buildings.2286	POINT (-120.40047274799991 38.0...	North Brainard Building	06109	647.0	0
Buildings.2287	POINT (-120.40047274799991 38.0...	Schwartz Building	06109	651.0	0
Buildings.2288	POINT (-120.40102824399992 38.0...	Soderer and Marshall Building	06109	646.0	0
Buildings.2289	POINT (-120.40047274799991 38.0...	South Brainard Building	06109	646.0	0
Buildings.2290	POINT (-120.40130606899993 38.0...	Tuolumne Engine Company Numbe...	06109	649.0	0
Buildings.2291	POINT (-120.22713322699991 38.0...	Twain Harte Community Center	06109	1124.0	0
Buildings.3134	POINT (-116.96724609699993 33.7...	Hemet Public Library	06065	488.0	0
Buildings.3321	POINT (-116.86724657999991 32.8...	Crest Branch San Diego County Libr...	06073	488.0	0

Open filter constructor Restore

Save Delete Cancel

Figure 3-48： The data table query results

The query results are displayed on the map synchronously, as shown in Figure 3-49.

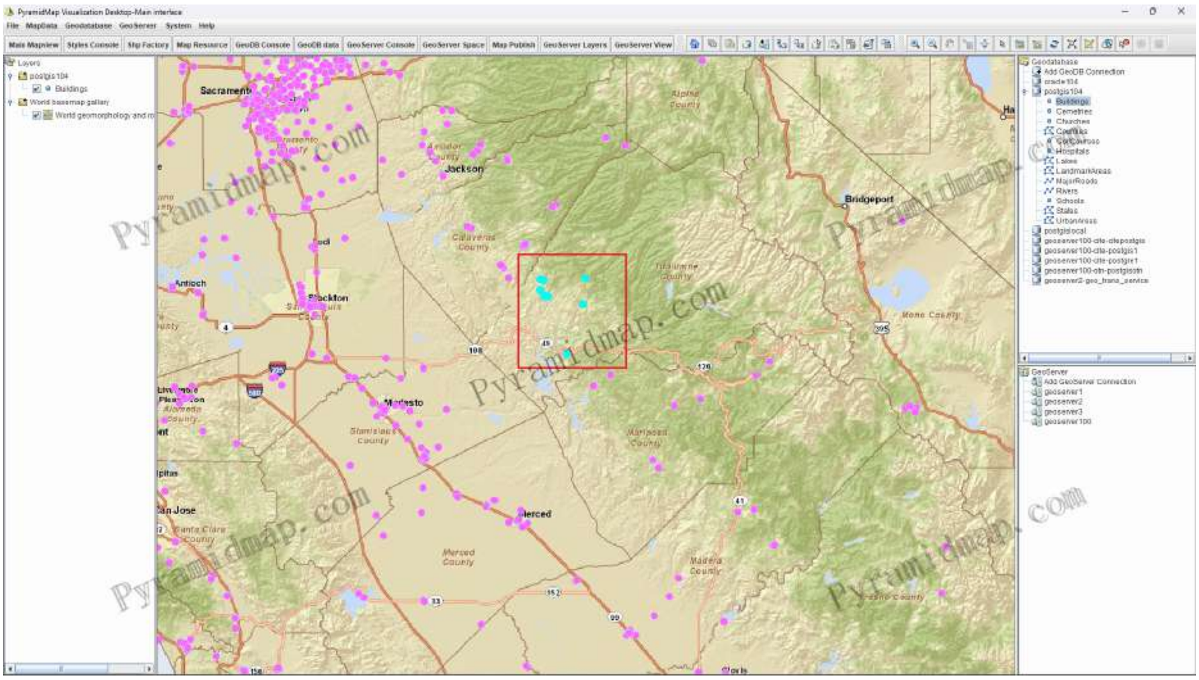


Figure 3-49： The query results are displayed on the map synchronously

In particular, the feature data table interacts with the map. Further, the editing, modification and deletion of feature attribute data can be completed in the data table, and the data can be submitted and saved according to the layer data source type. It supports the Shp vector file type, Geodatabase geographic database type, and GeoServer map server type.

4 Map editing

4.1 Create Shp

In PyramidMap, you can create your own Sh-style vector layer. Users can independently select the feature geometry type, map coordinate system, create layer fields and data types, and create vector layers that fully meet their needs. The design interface is shown in Figure 4-1.

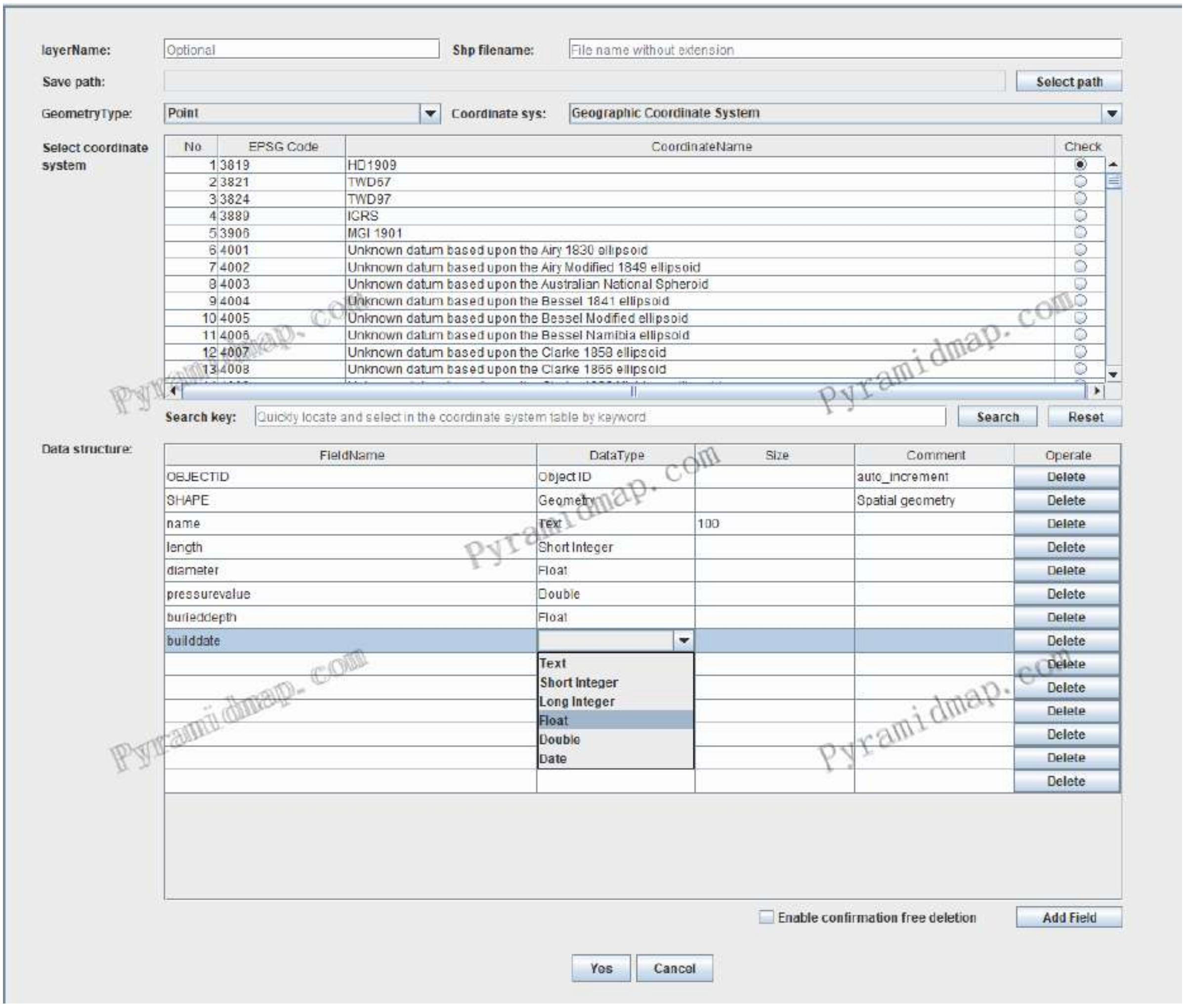


Figure 4-1： Shp layer data structure design

The Shp layer data structure includes three aspects: geometric type, coordinate system, and attribute data. The geometric type indicates the geometric shape of map features, including point (Point/MultiPoint), line (LineString/MultiLineString), and polygon (Polygon/MultiPolygon) types; The coordinate system is responsible for defining and describing the coordinate positions of map features, so as to display them in the correct positions and ranges. At the same time, digital maps not only have spatial data, but also can carry various types of attribute data, which is the unity of spatial and attribute data. PyramidMap covers most commonly used standard coordinate systems at present, including the GCS geographic coordinate system and the PCS projection coordinate system. GCS is responsible for the description of the spatial three-dimensional position, such as WGS84. PCS can understand the projection of the spatial position on the plane, and is responsible for the description of the plane position, such as WGS84 Web Mercator, which is the main coordinate system adopted by the current Web map. PyramidMap realizes the selection of coordinate system through classification list, and provides keyword mode to realize quick query and selection, as shown in Figure 4-2.

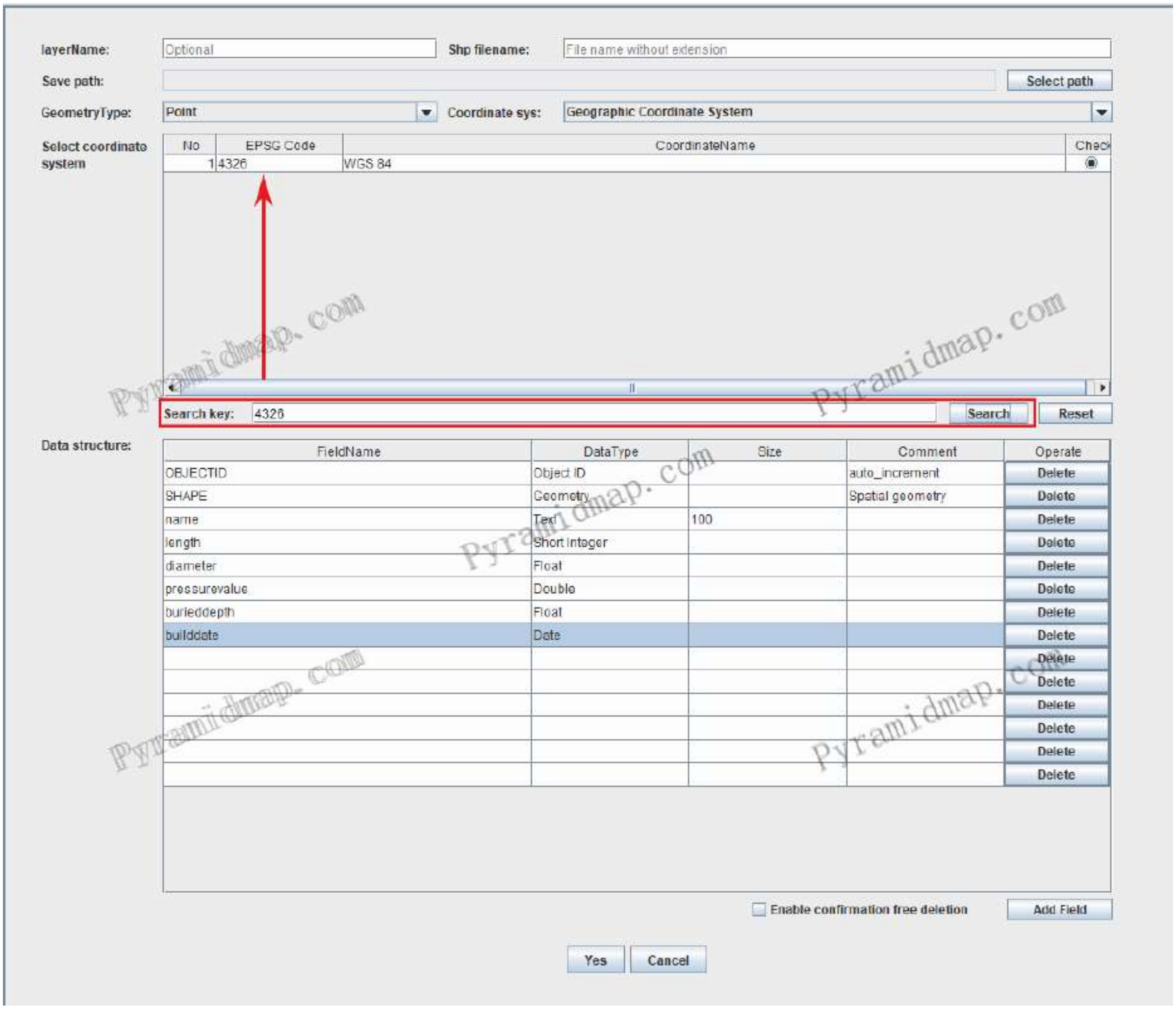


Figure 4-2： Fast selection of coordinate system through keywords

For a Shp type layer, it can have field information, including field naming, data type, and so on, like common database tables, to store all common types of data except spatial data, and realize the attribute information description of a map feature. Attribute data types include text, long integer, short integer, floating point type, double precision type and date time type. The creation process is shown in Figure 4-3.

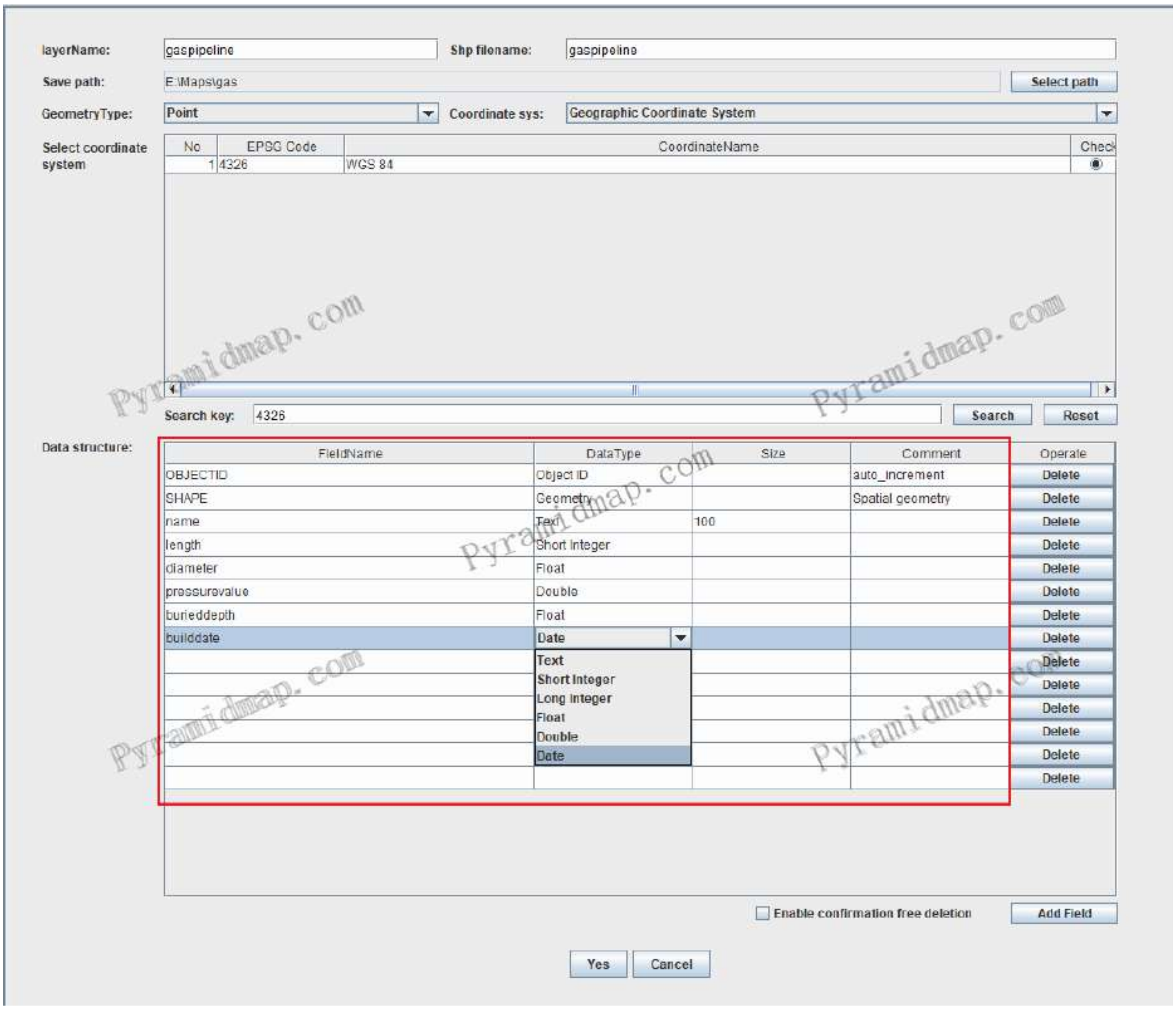


Figure 4-3: Design the field structure of Shp layer table

The created Shp layer is automatically added to the layer resource pool of PyramidMap for unified management, and can be selected from the resource pool list for all-purpose , as shown in Figure 4-4.

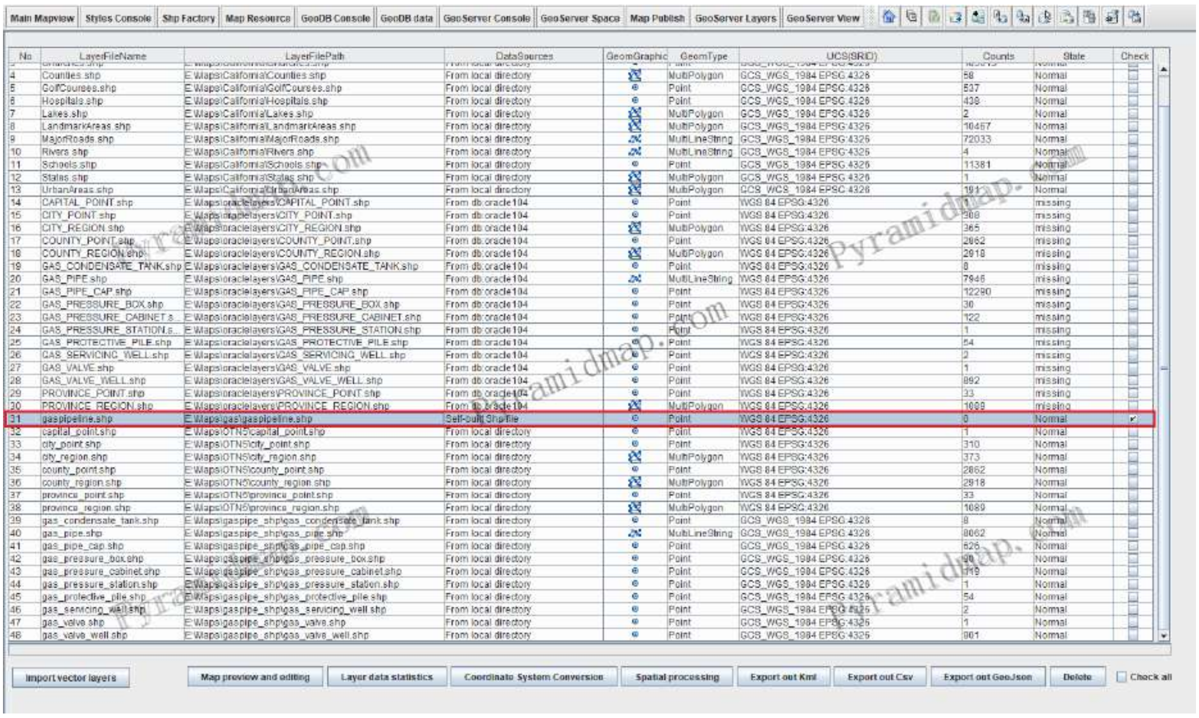


Figure 4-4: The created Shp layer is added to the map resource pool

Select a layer in the resource pool list to draw map elements and assign data values, as shown in Figure 4-5.

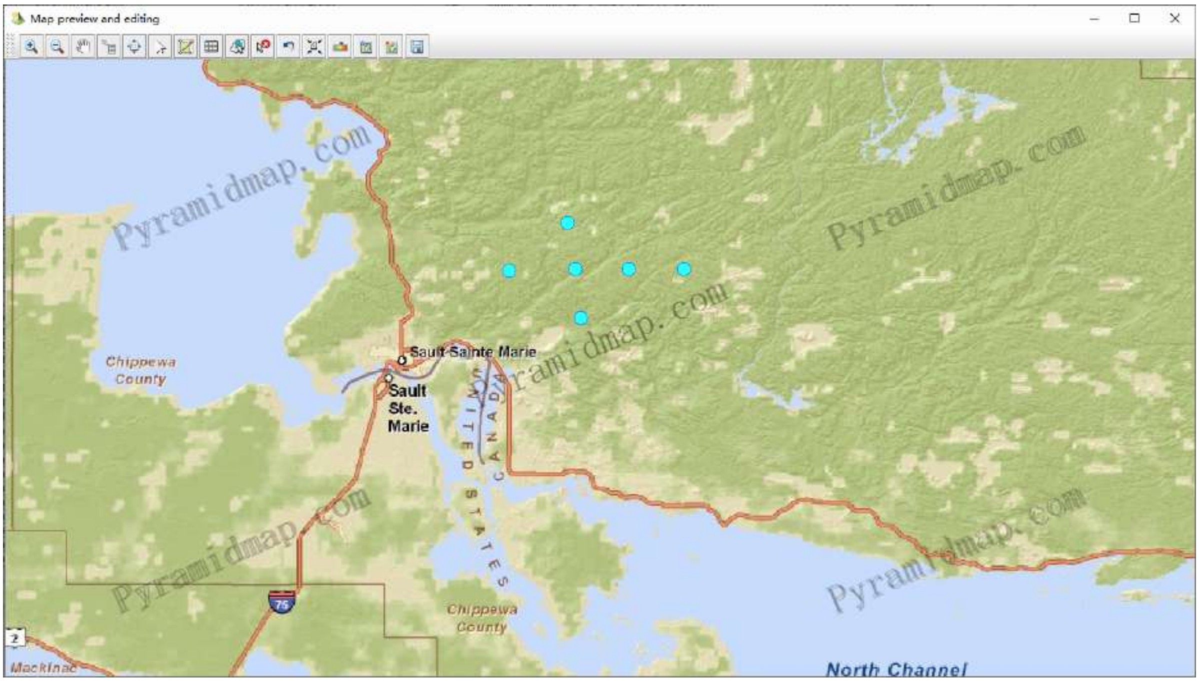


Figure 4-5： Map feature drawing and data assignment for the created Shp layer

This is the complete workflow of creating a map instance and adding it to the resource management pool, mapping, and adding map data.

4.2 Graphical Editing

Map data consists of geometric shapes representing space vectors and feature attributes data. Therefore, map editing includes two parts, they are graphical editing and attributes data editing.

4.2.1 Main view editor

PyramidMap supports the implementation of mouse dotting, line and polygon drawing, and submitting the saving according to the geometric type of the current layer in the main map view and each independent map view. Support Shp vector file, Geodatabase layer, wfs data in GeoServer and other data types. In the main view interface, activate the editing status of the selected layer through the layer node shortcut menu "Open Editing" option, as shown in Figure 4-6.

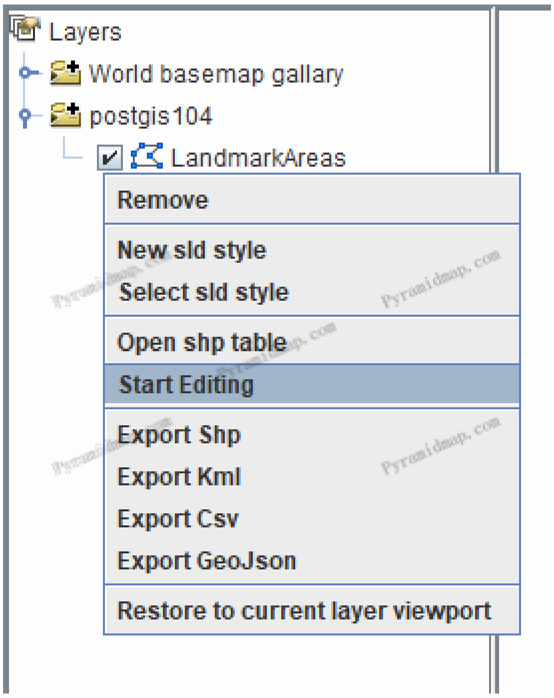


Figure 4-6：Layer on editing

Click the "Draw" button in the toolbar, as shown in Figure 4-7.

Figure 4-7：For vector layer editing, the "Draw" button in the toolbar

The selected layer enters the editing state, interacts with the mouse, and implements operations such as dotting, drawing lines, and drawing polygons according to the geometric type of the layer, as shown in Figure 4-8.

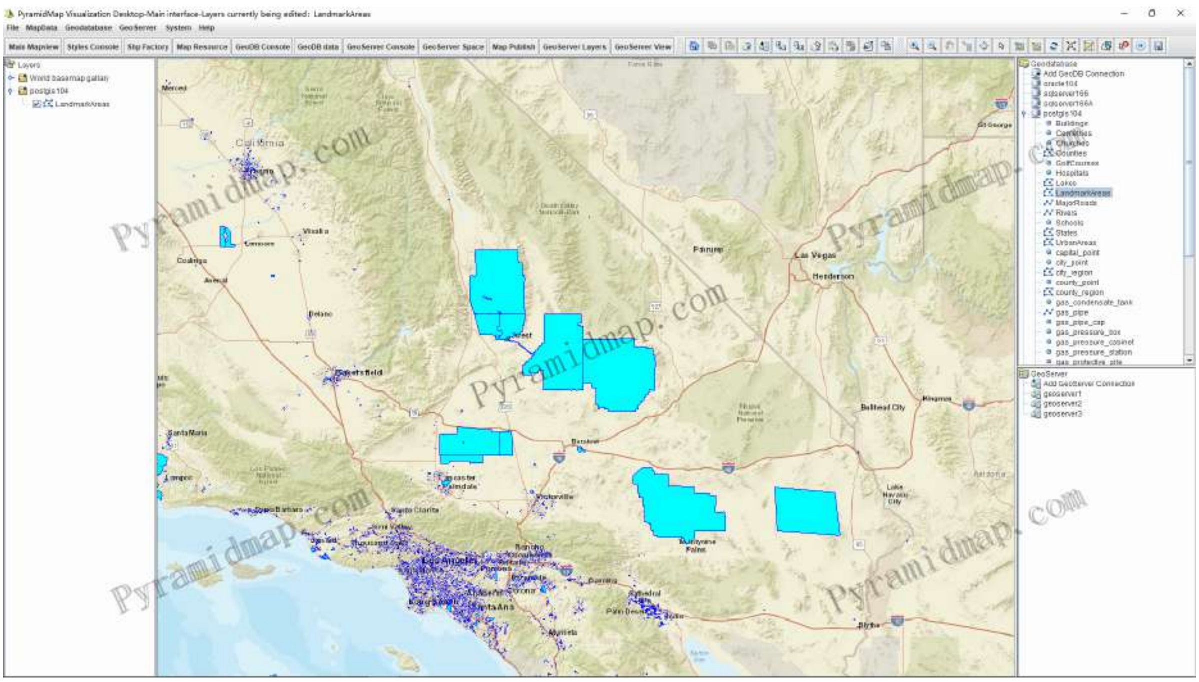


Figure 4-8：Drawing feature graphic on map

After drawing, click the "Save" button in the toolbar, as shown in Figure 4-9.

Figure 4-9：For the edited map data, the "Save" button in the toolbar

PyramidMap saves and submits the drawn feature data to the layer data source, and supports Shp vector file, Geodatabase vector layer, and GeoServer WFS layer. After successful saving, the system returns a prompt, as shown in Figure 4-10.

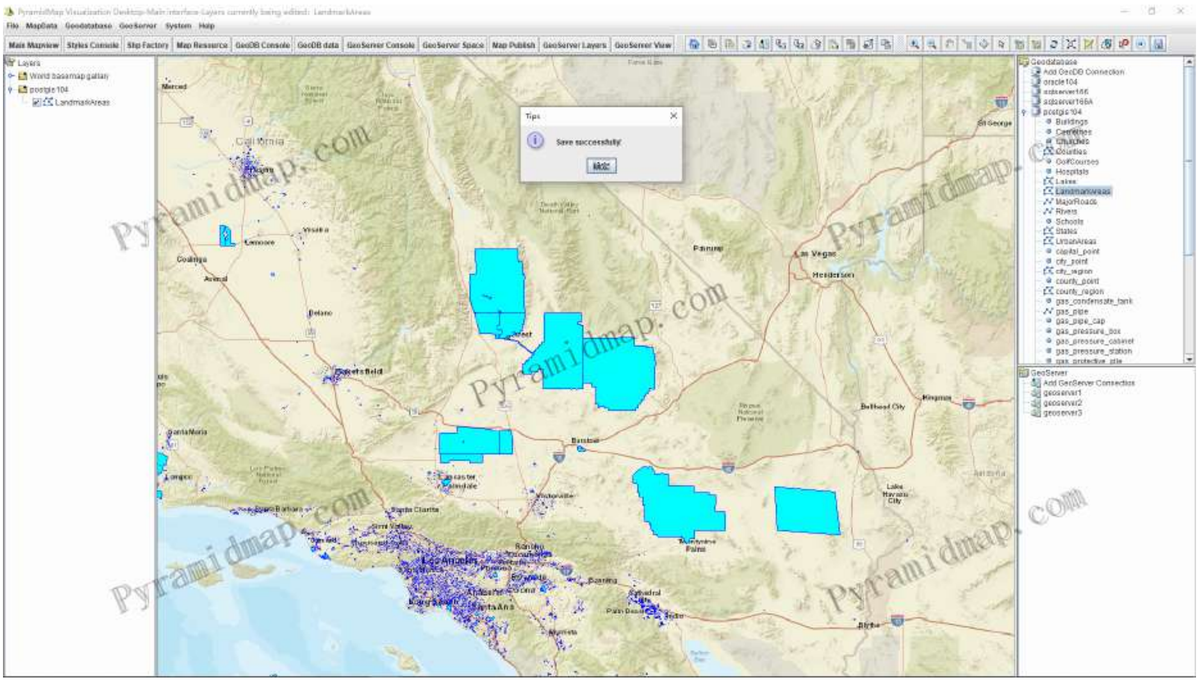


Figure 4-10：Drawing and saving map features

4.2.2 Independent view editor

In the Shp layer resource management and Geodatabase layer resource management modules, the independent display view of each layer can be opened through the "Preview and Edit" button to complete the editing operation function, as shown in Figure 4-11.

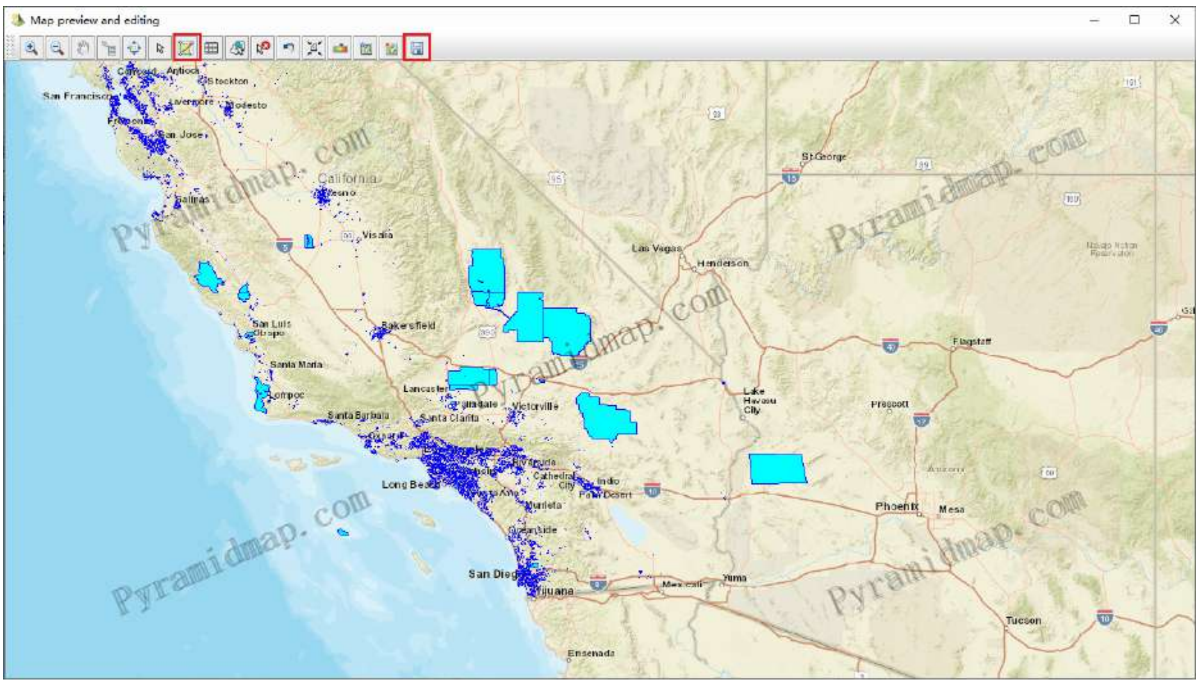


Figure 4-11：Layer independent display editing function page

Click the drawing button to realize the interactive drawing of the mouse on the map view. After that, click the "Save" button to update and save the data to the data source corresponding to the layer, Shp to the layer file, and the database layer to the feature table.

4.3 Attribute data editing

PyramidMap implements digital editing of the layer attribute table, which is respectively implemented in the main interface visualization layer node and the preview and editing in the Shp or Geodatabase geographical database layer list.

4.3.1 Editing in main map view

Visualize the "Open shp table" option in the right-click shortcut menu of the layer node on the main interface, as shown in Figure 4-12.

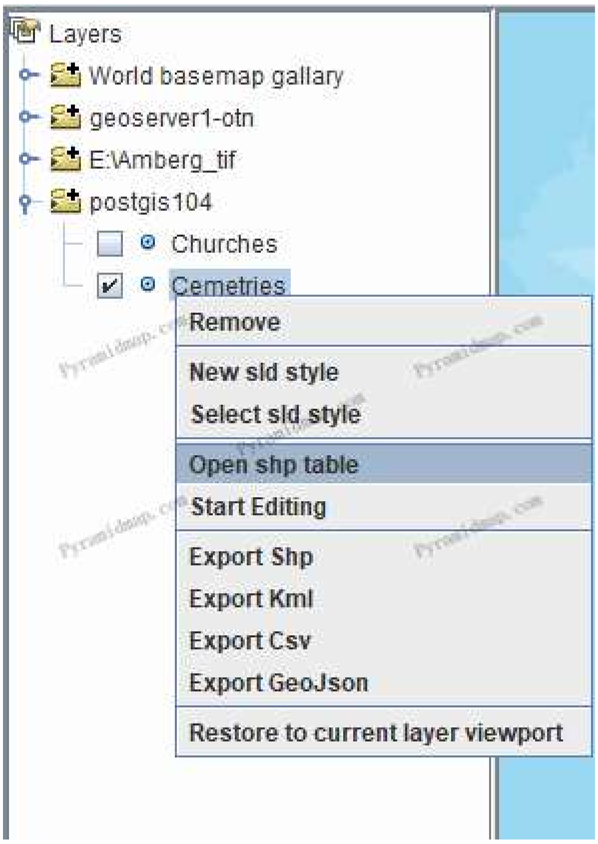
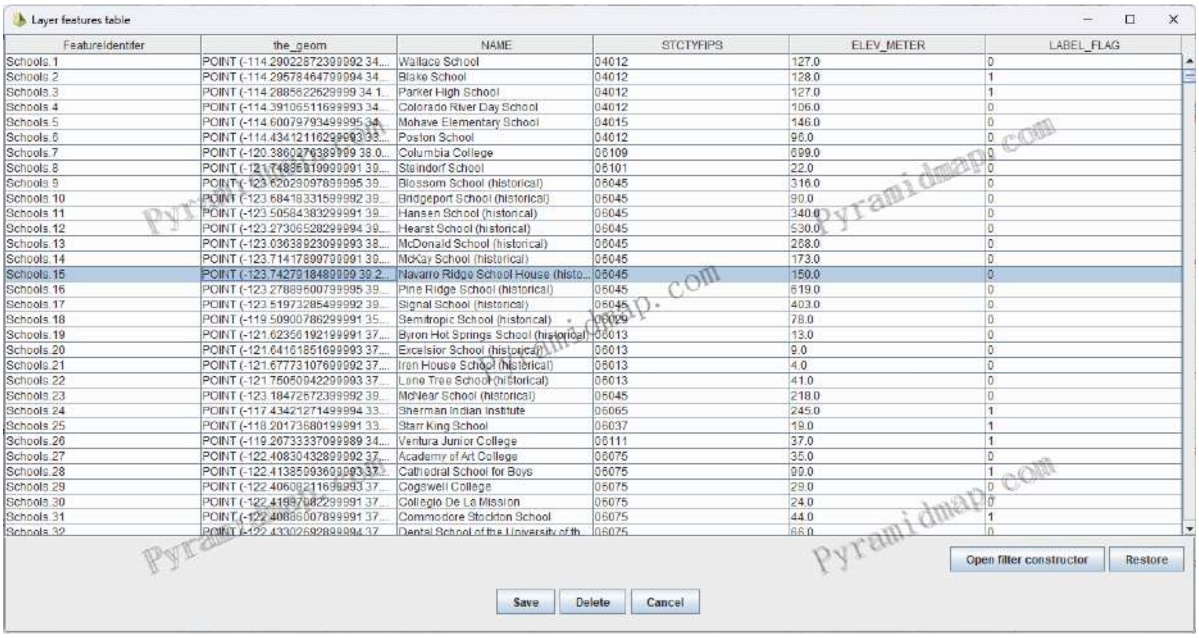


Figure 4-12：Visualize the "Open shp table" in the right-click shortcut menu of the layer node

The opened layer data table is shown in Figure 4-13.



Layer features table

FeatureIdentifer	the_geom	NAME	STCTYFIPS	ELEV_METER	LABEL_FLAG
Schools.1	POINT (-114.29022872399992 34....	Wallace School	04012	127.0	0
Schools.2	POINT (-114.29578464799994 34....	Blake School	04012	128.0	1
Schools.3	POINT (-114.2885622629999 34.1...	Parker High School	04012	127.0	1
Schools.4	POINT (-114.39106511699993 34....	Colorado River Day School	04012	106.0	0
Schools.5	POINT (-114.60079793499995 34....	Mohave Elementary School	04015	146.0	0
Schools.6	POINT (-114.43412116299993 33....	Poston School	04012	96.0	0
Schools.7	POINT (-120.38602763899999 38.0...	Columbia College	06109	699.0	0
Schools.8	POINT (-121.74886519999991 39....	Steindorf School	06101	22.0	0
Schools.9	POINT (-123.62029097899995 39....	Blossom School (historical)	06045	316.0	0
Schools.10	POINT (-123.68418331599992 39....	Bridgeport School (historical)	06045	90.0	0
Schools.11	POINT (-123.50584383299991 39....	Hansen School (historical)	06045	340.0	0
Schools.12	POINT (-123.27306528299994 39....	Hearst School (historical)	06045	530.0	0
Schools.13	POINT (-123.03638923099993 38....	McDonald School (historical)	06045	268.0	0
Schools.14	POINT (-123.71417899799991 39....	McKay School (historical)	06045	173.0	0
Schools.15	POINT (-123.74279184899999 39.2...	Navarro Ridge School House (histo...	06045	150.0	0
Schools.16	POINT (-123.27889600799995 39....	Pine Ridge School (historical)	06045	619.0	0
Schools.17	POINT (-123.51973285499992 39....	Signal School (historical)	06045	403.0	0
Schools.18	POINT (-119.50900786299991 35....	Semitropic School (historical)	06029	78.0	0
Schools.19	POINT (-121.62356192199991 37....	Byron Hot Springs School (historical)	06013	13.0	0
Schools.20	POINT (-121.64161851699993 37....	Excelsior School (historical)	06013	9.0	0
Schools.21	POINT (-121.67773107699992 37....	Iron House School (historical)	06013	4.0	0
Schools.22	POINT (-121.75050942299993 37....	Lone Tree School (historical)	06013	41.0	0
Schools.23	POINT (-123.18472672399992 39....	McNear School (historical)	06045	218.0	0
Schools.24	POINT (-117.43421271499994 33....	Sherman Indian Institute	06065	245.0	1
Schools.25	POINT (-118.20173680199991 33....	Starr King School	06037	19.0	1
Schools.26	POINT (-119.26733337099989 34....	Ventura Junior College	06111	37.0	1
Schools.27	POINT (-122.40830432899992 37....	Academy of Art College	06075	35.0	0
Schools.28	POINT (-122.41385093699993 37....	Cathedral School for Boys	06075	90.0	1
Schools.29	POINT (-122.40608211699993 37....	Cogswell College	06075	29.0	0
Schools.30	POINT (-122.41997082299991 37....	Collegio De La Mission	06075	24.0	0
Schools.31	POINT (-122.40886007899991 37....	Commodore Stockton School	06075	44.0	1
Schools.32	POINT (-122.43302692899994 37...	Dental School of the University of th...	06075	66.0	0

Open filter constructor | Restore

Save | Delete | Cancel

Figure 4-13： Edit attribute data in the shp data table

Double click the table Cell (except for the Id and Geometry fields) to switch to the editing and input status, and the selected features are highlighted on the map at the same time, as shown in Figure 4-14.

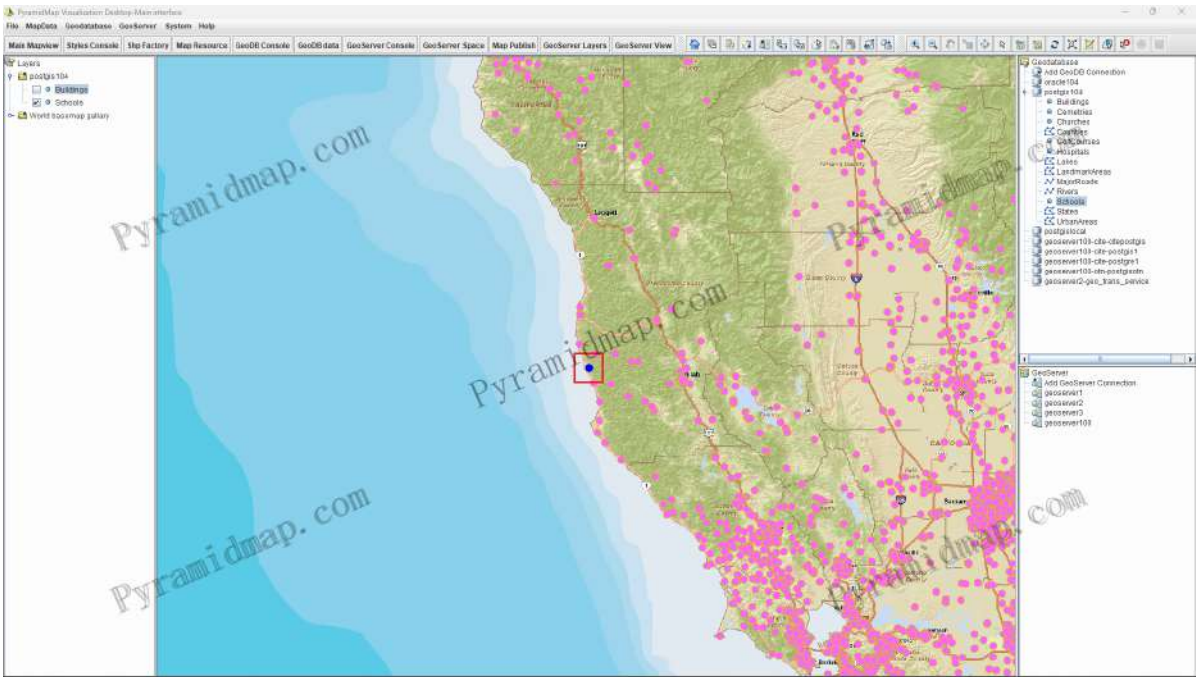


Figure 4-14： The selected features in the data table are highlighted on the map

Batch modification, submission and saving are supported.

4.3.2 Editing in individual map view

Layers in the Shp resource pool and database resource pool can be previewed and edited in an independent display interface, as shown in Figure 4-15.

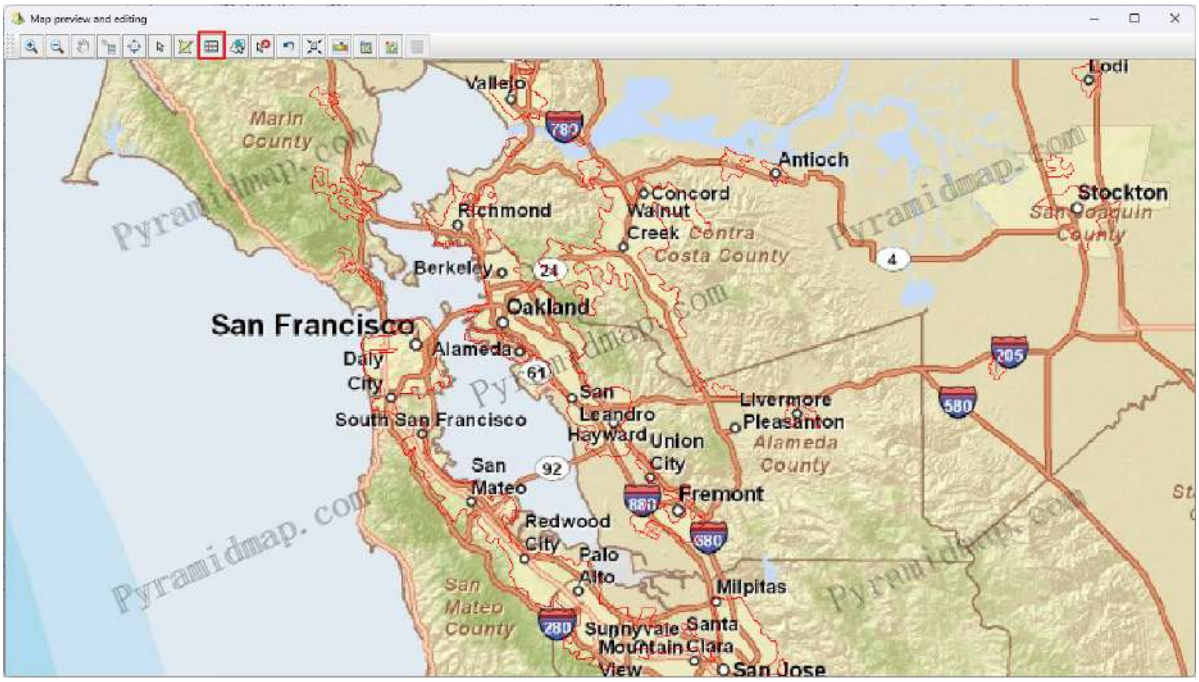
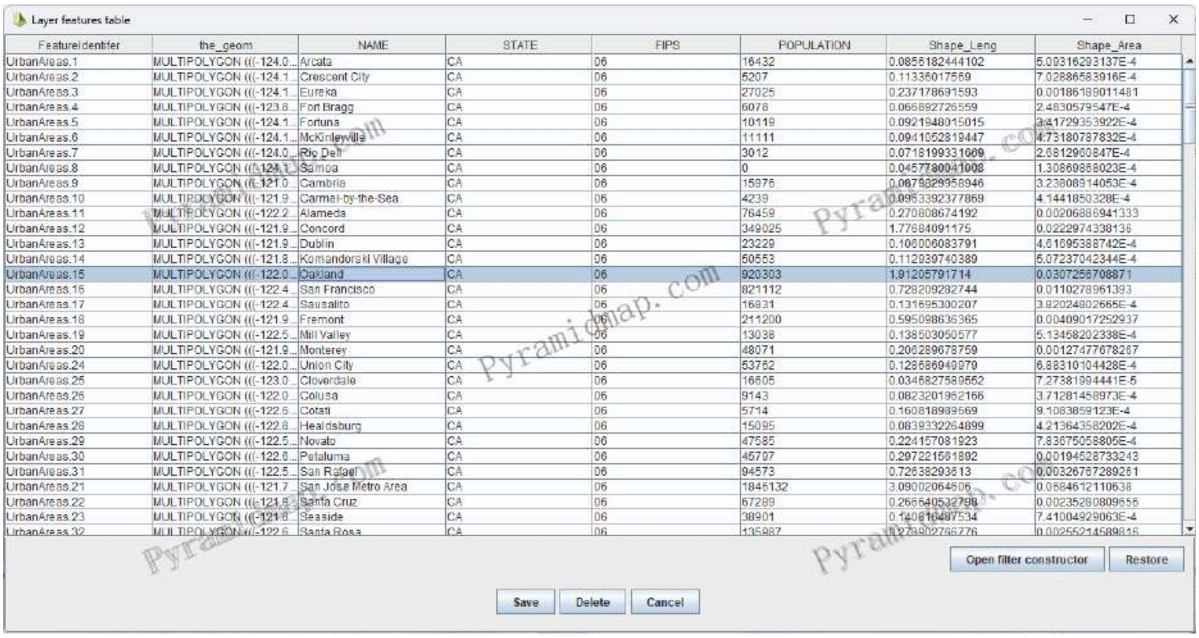


Figure 4-15：Open the data table in the independent view of each layer

Click "Open shp table" to open the layer feature data table, as shown in Figure 4-16.


Layer features table

FeatureIdentifer	the_geom	NAME	STATE	FIPS	POPULATION	Shape_Leng	Shape_Area
UrbanAreas.1	MULTIPOLYGON (((-124.0...	Arcata	CA	06	16432	0.0856182444102	5.09316293137E-4
UrbanAreas.2	MULTIPOLYGON (((-124.1...	Crescent City	CA	06	5207	0.11336017569	7.02886583916E-4
UrbanAreas.3	MULTIPOLYGON (((-124.1...	Eureka	CA	06	27025	0.237178691593	0.00186189011481
UrbanAreas.4	MULTIPOLYGON (((-123.8...	Fort Bragg	CA	06	6078	0.066892726559	2.4830579547E-4
UrbanAreas.5	MULTIPOLYGON (((-124.1...	Fortuna	CA	06	10119	0.0921948015015	3.41729353922E-4
UrbanAreas.6	MULTIPOLYGON (((-124.1...	McKinleyville	CA	06	11111	0.0941052819447	4.73180787832E-4
UrbanAreas.7	MULTIPOLYGON (((-124.0...	Rio Dell	CA	06	3012	0.0718199331669	2.6812960847E-4
UrbanAreas.8	MULTIPOLYGON (((-124.1...	Samoa	CA	06	0	0.0457780041008	1.30869868023E-4
UrbanAreas.9	MULTIPOLYGON (((-121.0...	Cambria	CA	06	15976	0.0879829958946	3.23808914053E-4
UrbanAreas.10	MULTIPOLYGON (((-121.9...	Carmel-by-the-Sea	CA	06	4239	0.0953392377869	4.1441850328E-4
UrbanAreas.11	MULTIPOLYGON (((-122.2...	Alameda	CA	06	76459	0.270808674192	0.00206886941333
UrbanAreas.12	MULTIPOLYGON (((-121.9...	Concord	CA	06	349025	1.77684091175	0.0222974338136
UrbanAreas.13	MULTIPOLYGON (((-121.9...	Dublin	CA	06	23229	0.106006083791	4.61695388742E-4
UrbanAreas.14	MULTIPOLYGON (((-121.8...	Komandorski Village	CA	06	50553	0.112839740389	5.07237042344E-4
UrbanAreas.15	MULTIPOLYGON (((-122.0...	Oakland	CA	06	920303	1.91205791714	0.0307256708871
UrbanAreas.16	MULTIPOLYGON (((-122.4...	San Francisco	CA	06	821112	0.728209282744	0.0110278961393
UrbanAreas.17	MULTIPOLYGON (((-122.4...	Sausalito	CA	06	16831	0.131695300207	3.82024802665E-4
UrbanAreas.18	MULTIPOLYGON (((-121.9...	Fremont	CA	06	211200	0.595098636365	0.00409017252937
UrbanAreas.19	MULTIPOLYGON (((-122.5...	Mill Valley	CA	06	13038	0.138503050577	5.13458202338E-4
UrbanAreas.20	MULTIPOLYGON (((-121.9...	Monterey	CA	06	48071	0.206289678759	0.00127477678267
UrbanAreas.24	MULTIPOLYGON (((-122.0...	Union City	CA	06	53762	0.128686940979	6.88310104428E-4
UrbanAreas.25	MULTIPOLYGON (((-123.0...	Cloverdale	CA	06	16605	0.0346827589552	7.27381994441E-5
UrbanAreas.26	MULTIPOLYGON (((-122.0...	Colusa	CA	06	9143	0.0823201962166	3.71281458973E-4
UrbanAreas.27	MULTIPOLYGON (((-122.6...	Cotati	CA	06	5714	0.160818989569	9.1083859123E-4
UrbanAreas.28	MULTIPOLYGON (((-122.8...	Healdsburg	CA	06	15095	0.0839332264899	4.21364358202E-4
UrbanAreas.29	MULTIPOLYGON (((-122.5...	Novato	CA	06	47585	0.224157081923	7.83675058805E-4
UrbanAreas.30	MULTIPOLYGON (((-122.6...	Petaluma	CA	06	45797	0.297221561892	0.00194528733243
UrbanAreas.31	MULTIPOLYGON (((-122.5...	San Rafael	CA	06	94573	0.72638293613	0.00326767289261
UrbanAreas.21	MULTIPOLYGON (((-121.7...	San Jose Metro Area	CA	06	1846132	3.09002064606	0.0684612110638
UrbanAreas.22	MULTIPOLYGON (((-121.8...	Santa Cruz	CA	06	67289	0.265640532798	0.00235280809656
UrbanAreas.23	MULTIPOLYGON (((-121.8...	Seaside	CA	06	38901	0.140816487534	7.41004929063E-4
UrbanAreas.32	MULTIPOLYGON (((-122.6...	Santa Rosa	CA	06	135987	0.279902766776	0.00255214589816

Open filter constructor | Restore

Save | Delete | Cancel

Figure 4-16：Edit attribute data in the shp independent view

Double click the table Cell (except for the Id and Geometry fields) to switch to the editing input status, and the selected features will be highlighted on the map view at the same time, as shown in Figure 4-17.

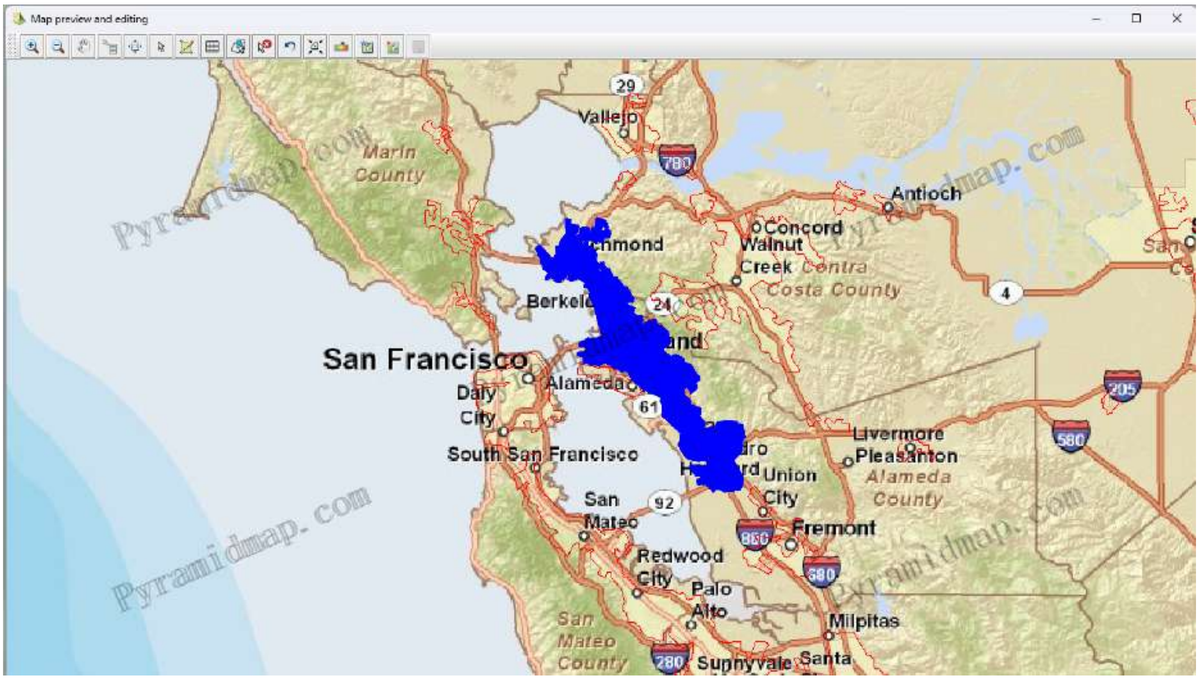


Figure 4-17： Layer feature highlight by being selected in independent view

Batch modification, submission and saving are supported.

5 Geodatabase and GeoServer

5.1 Geodatabase connection pool

PyramidMap supports direct connection access with geographic databases to realize the import, export, storage and conversion of map data, and supports (but is not limited to) Oracle, PostGIS (PostgreSQL+GIS function extension), MySQL, SQLServer and other databases. After PyramidMap creates a database connection, it is managed in the database connection pool mode to import, export, preview, edit and other operations of the map. PyramidMap provides two ways to create a geographic database connection: 1. create a database connection through the map database node on the main map display interface; 2. create a database connection on the database configuration page through "System" - "Geodatabase console" entrance.

5.1.1 Geodatabase data source node

In the geographic database node of the main interface, double-click New Database Connection, as shown in Figure 5-1.

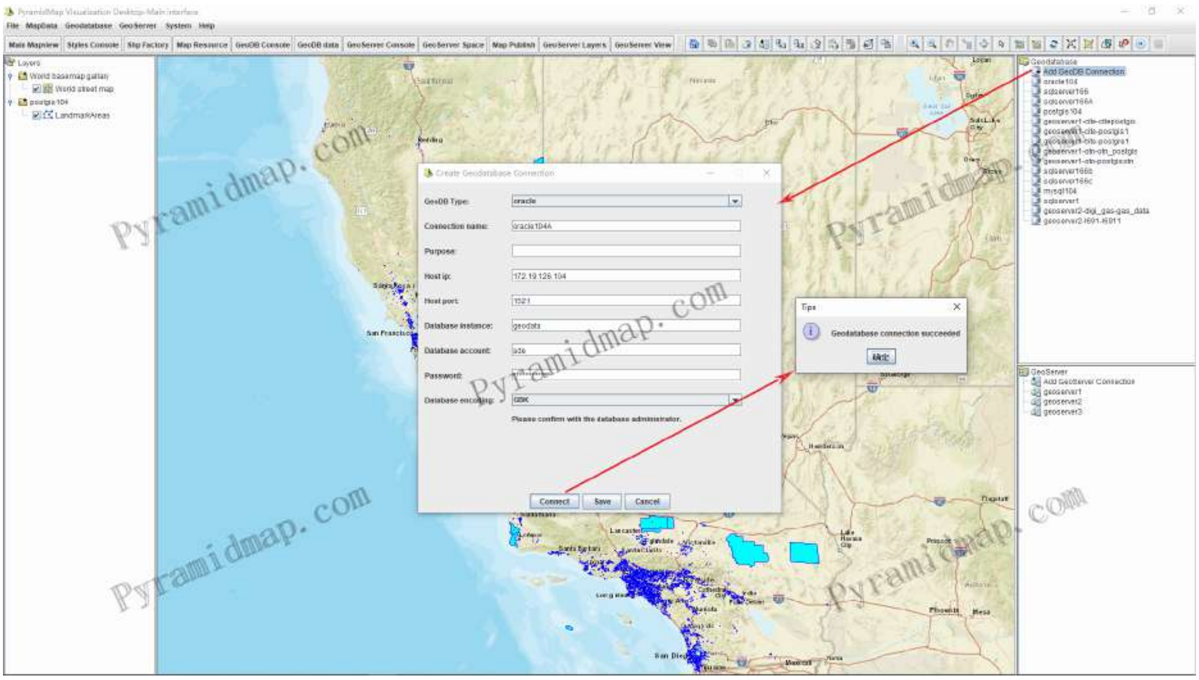


Figure 5-1： Main interface create new Geodatabase connection

Open the geographic database connection configuration interface, select the geographic database type, and enter the correct connection parameters. After the connection test is successful, it can be saved and created as an effective geographic database connection, included in the PyramidMap geographic database resource connection pool, and dynamically added to the geodatabase node and the inside layers node, as shown in Figure 5-2.

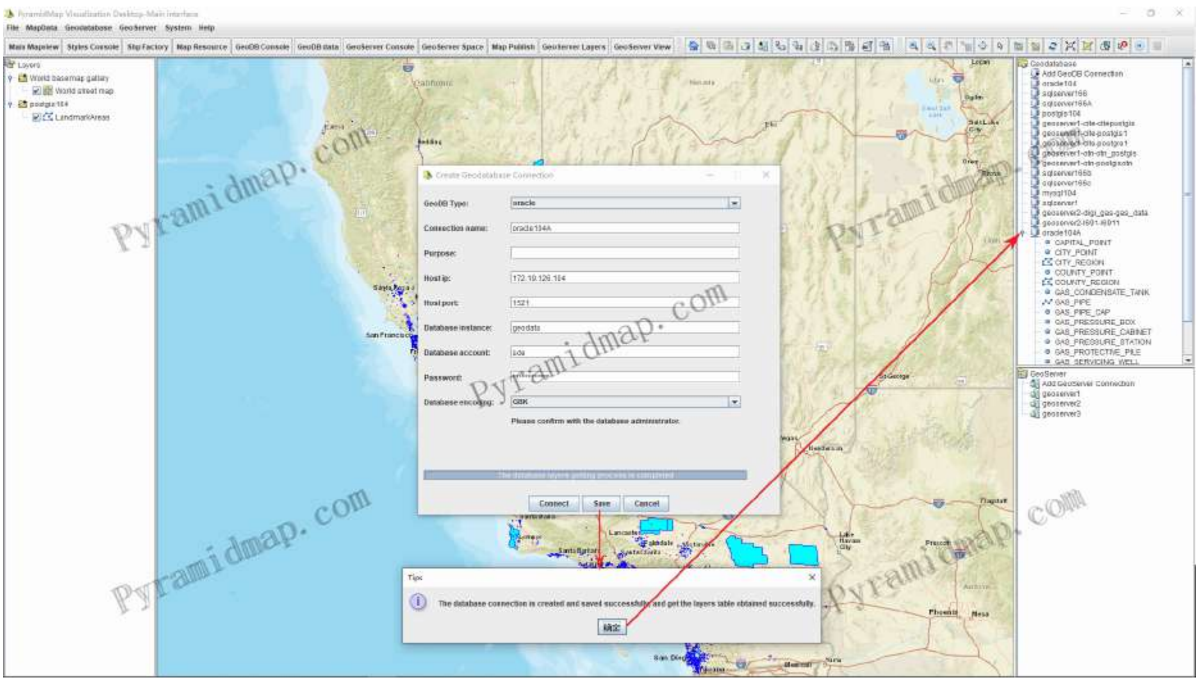


Figure 5-2： Geodatabase connection created successfully

Description of interface input parameters:

- GeoDB type: Select a database type (Oracle, PostGIS, MySQL)
- Connection name: Customize the Geodatabase connection name.
- Purpose: Customize the Geodatabase connection purpose describe.
- Host IP： IP address of the server where the Geodatabase is hosted.

- Port： The port number used by the database. (Default port number: Oracle 1521, Postgre 5432, MySQL 3306)
- Database instance: the name of the database's service instance to connect to.
- Schema： The schema to be connected is mainly for PostGIS, and others can be ignored. In Oracle, this parameter is consistent with the user name.
- The connection button: test whether the newly created database connection is valid.
- The save button: save the new database connection.
- The cancel button: Exit the interface without performing any operation.

5.1.2 Geodatabase connection pool

Open the Geodatabase connection pool interface through "System" - "Geodatabase console", or directly click the "GeoDB console" button in the toolbar， as shown in Figure 5-3.

Figure 5-3： Create a Geodatabase connection through the system settings portal

The geodatabase connection configuration page is shown in Figure 5-4.

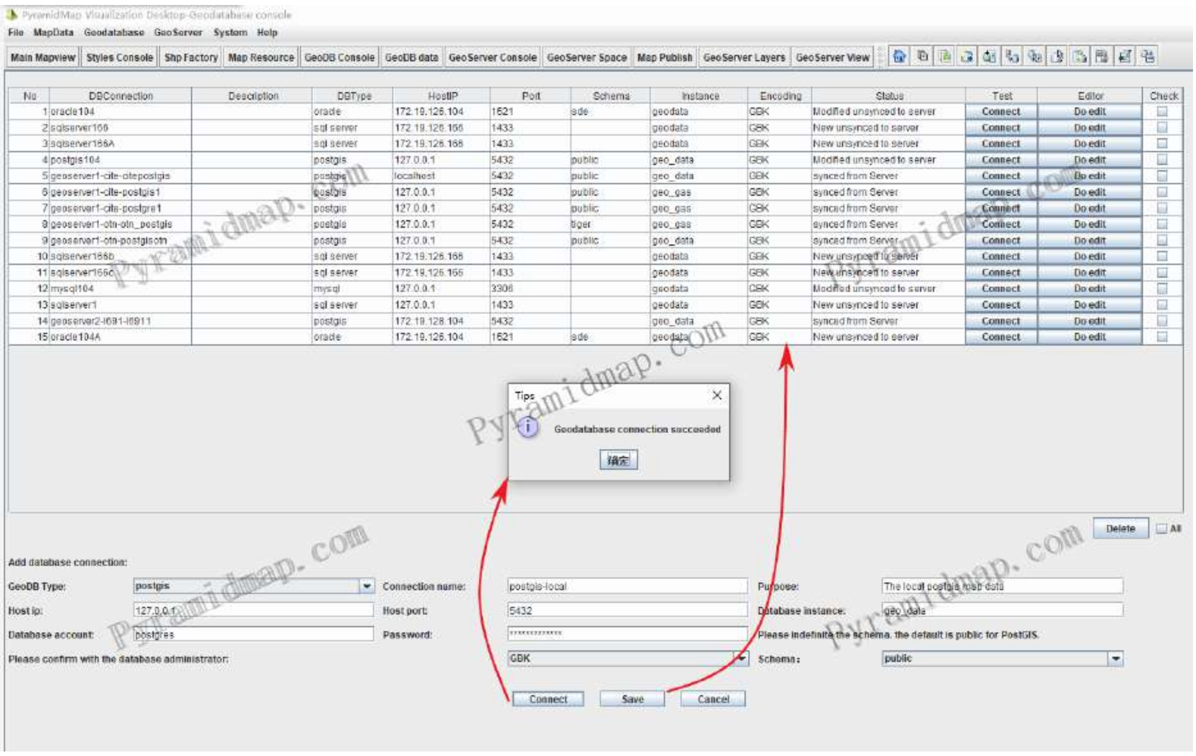


Figure 5-4： Geodatabase Connection Configuration and pool

The connections in the pool can be retested, edited and deleted, as shown in Figure 5-5.

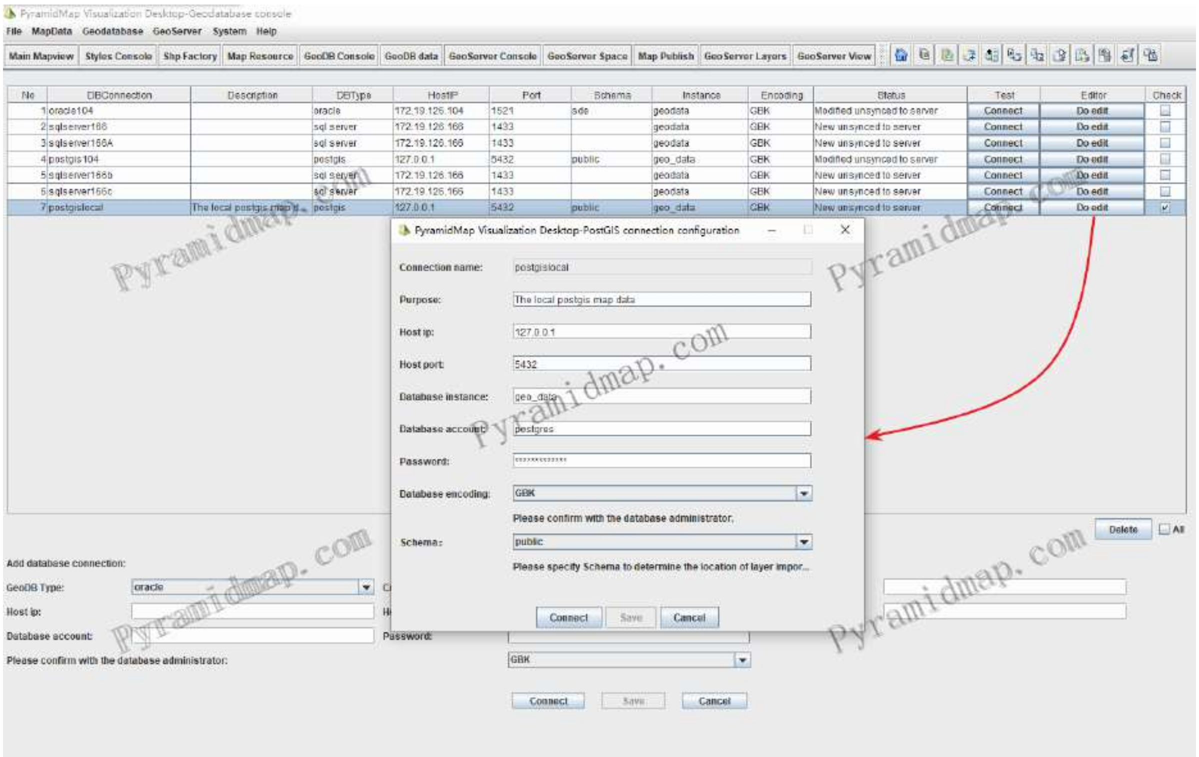


Figure 5-5： The connections in the pool can be retested, edited and deleted

Use the connection pool list to test, edit, modify, and delete the Geodatabase connection. Each connection has two button options : Connect and Do Edit.

.Connect: test whether the database connection is successful.

.Do edit: Re edit the Geodatabase connection to make it available.

5.2 GeoServer connection pool

PyramidMap supports direct connection access with GeoServer, realizes multi type layer publishing function, and provides layer service interface for WebGIS. After the connection parameters are configured and tested successfully, create and maintain the GeoServer resource connection pool. PyramidMap provides two ways to create a GeoServer connection: 1. Through the GeoServer tree node on the main interface; 2. Implement it on the GeoServer connection configuration module through "System" - "GeoServer console".

5.2.1 GeoServer data node

In the GeoServer connection node of the main interface, double-click Add GeoServer Connection, as shown in Figure 5-6.

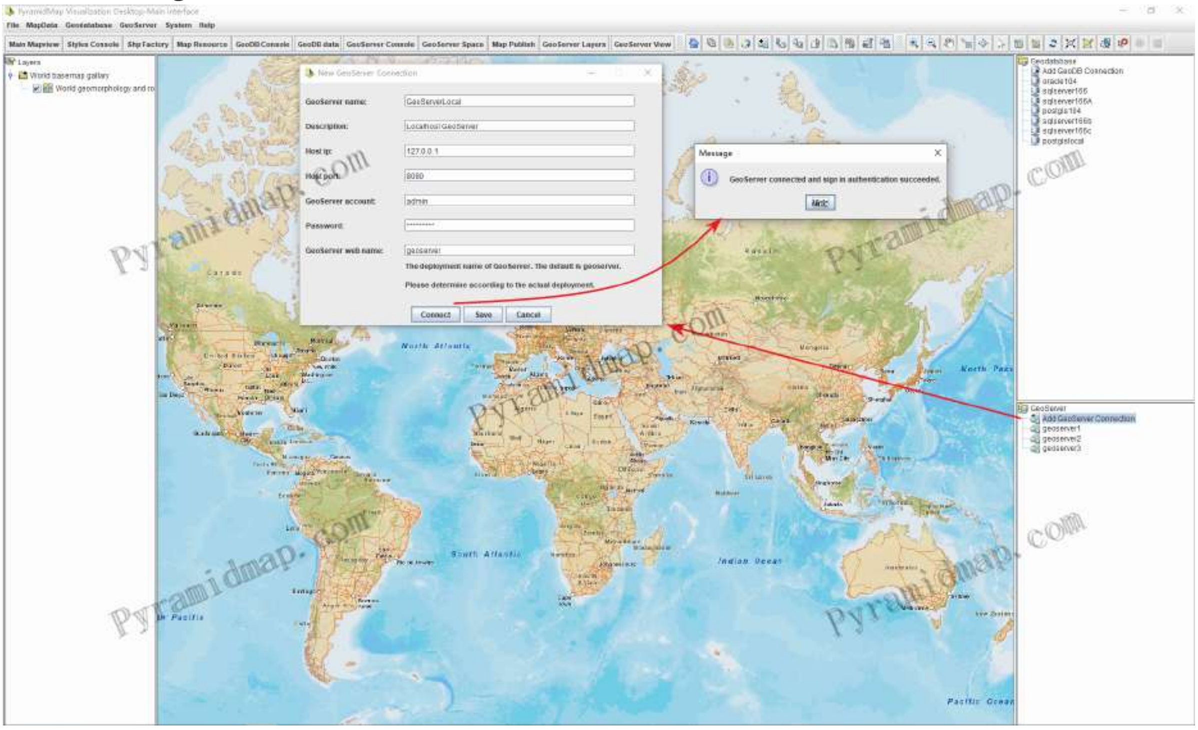


Figure 5-6： Create GeoServer connection through double-clicking on the GeoServer node in the main interface

Open the GeoServer connection configuration interface, enter the correct connection parameters, after the connection testing successful, you can save and create a valid GeoServer connection and hosting the connection in resource pool, and dynamically add it to the GeoServer connection node. Description of interface input parameters:

- GeoServer name: Customize the GeoServer server connection naming.
- Description： Customize the GeoServer connection purpose describe.
- Host IP： IP address of the server where the GeoServer is hosted.
- Port： GeoServer port number. (The port number of the web server hosting GeoServer, such as Tomcat)
- GeoServer account： GeoServer administrator account, default admin, please contact the web administrator for confirmation.
- Password: GeoServer administrator password, please contact the web administrator for confirmation.
- GeoServer web name： The deployment name of GeoServer in the web service. The default is geoserver. Please contact the web administrator for confirmation.

5.2.2 GeoServer connection pool

Open the GeoServer connection pool management interface through "System" - "GeoServer console", or directly click the "GeoServer console" button in the toolbar, as shown in Figure 5-7.

Figure 5-7： Create GeoServer connection menu

In the GeoServer connection pool management interface, create a new GeoServer connection and incorporate it into the unified resource management pool. The workflow is shown in Figure 5-8.

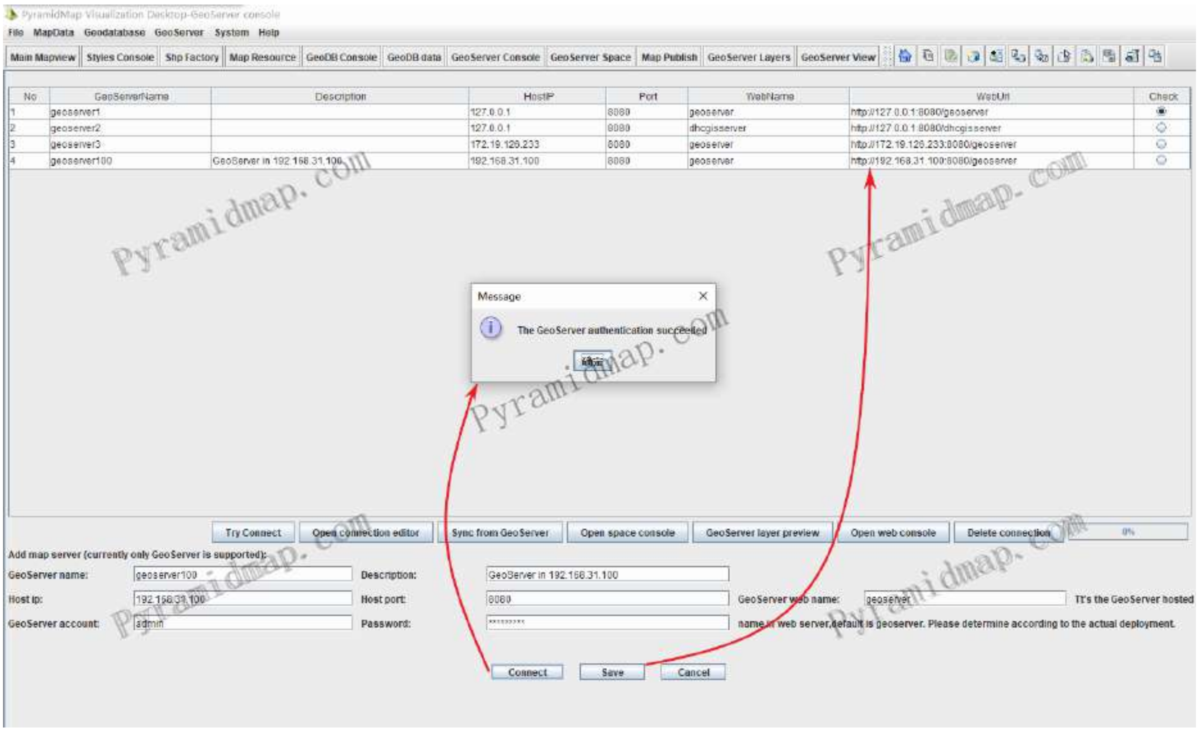


Figure 5-8： Create GeoServer connection and connections pool

The GeoServer connection created successfully is added to the resource connection pool.

5.2.3 Manage GeoServer connections

PyramidMap maintain the GeoServer connections in resource pool, manage the server internal workspace, data storage and layers in the function

modules, as shown in Figure 5-9.

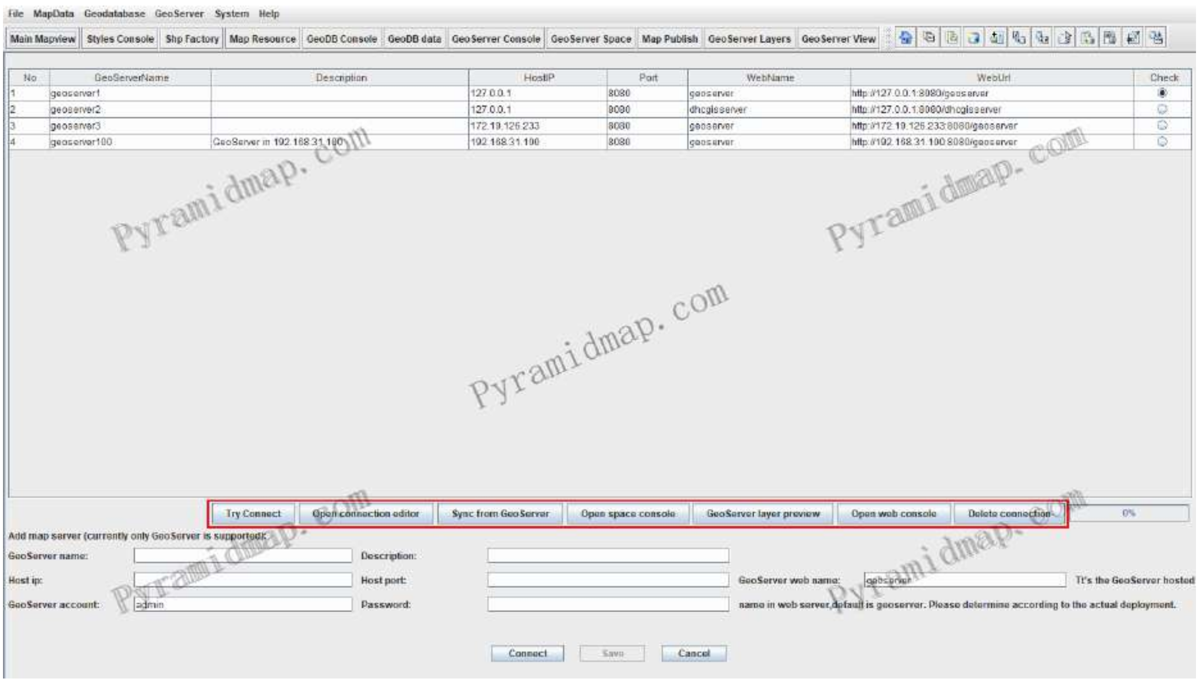


Figure 5-9： Internal management of GeoServer

GeoServer internal management identify：

- Try connect: test whether the selected map server connection is available.
- Open Connection editor: Edit the selected GeoServer connection properties.
- Sync from GeoServer: obtain the selected GeoServer workspace, data storage and database connection properties data, and synchronizing them to the client.
- Open space console: query, edit and modify, delete the workspace and data storage of the selected GeoServer.
- GeoServer layer preview: get the list of selected GeoServer layers and preview them.
- Delete connection: Delete the selected GeoServer connection.

5.3 GeoServer internal operate

The internal space of GeoServer is divided into three levels: workspaces, data storages and layers. As the client of GeoServer, PyramidMap can visually maintain these modules. PyramidMap provides multiple forms of access management to GeoServer server space.

5.3.1 GeoServer Workspace

Through the GeoServer node tree in the main interface, you can right-click on the GeoServer node layer and the workspace node layer to provide management access to the workspace and data storage. The management entry of the GeoServer node to the workspace is shown in Figure 5-10.

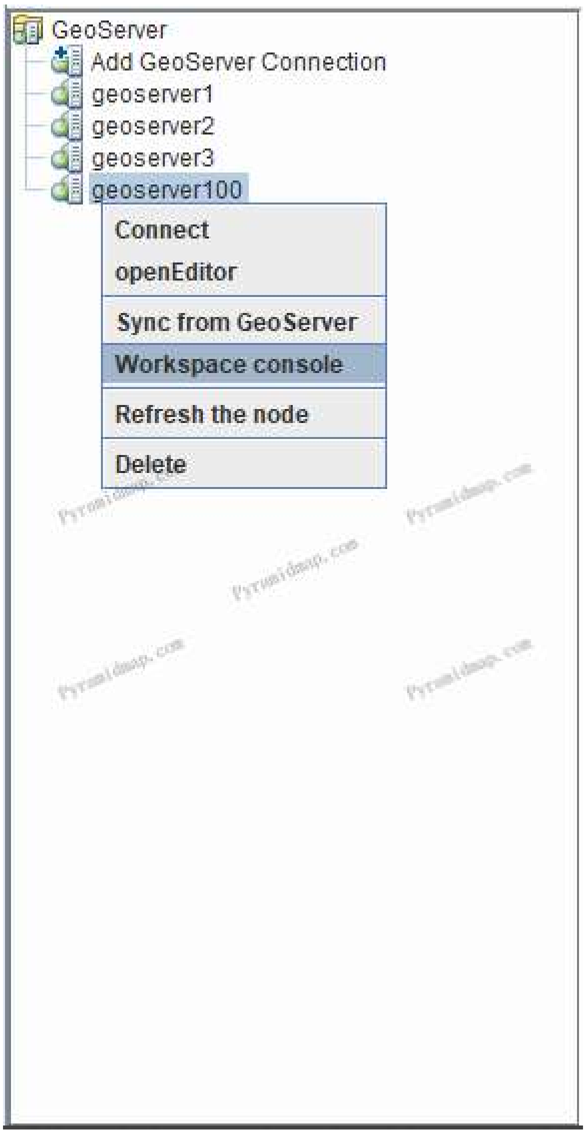


Figure 5-10：GeoServer node entrance to workspace

Enter the workspace management module, as shown in Figure 5-11.

Figure 5-11： The GeoServer's workspaces pool manage interface

In the workspace module, you can perform the following operations: create a new workspace, synchronize the server workspace to the client, and submit the client workspace to the server. In particular, in the workspace list management pool, you can provide independent attribute editors and delete each workspace in the list table. The new workspace interface and and data interaction with GeoServer is shown in Figure 5-12.

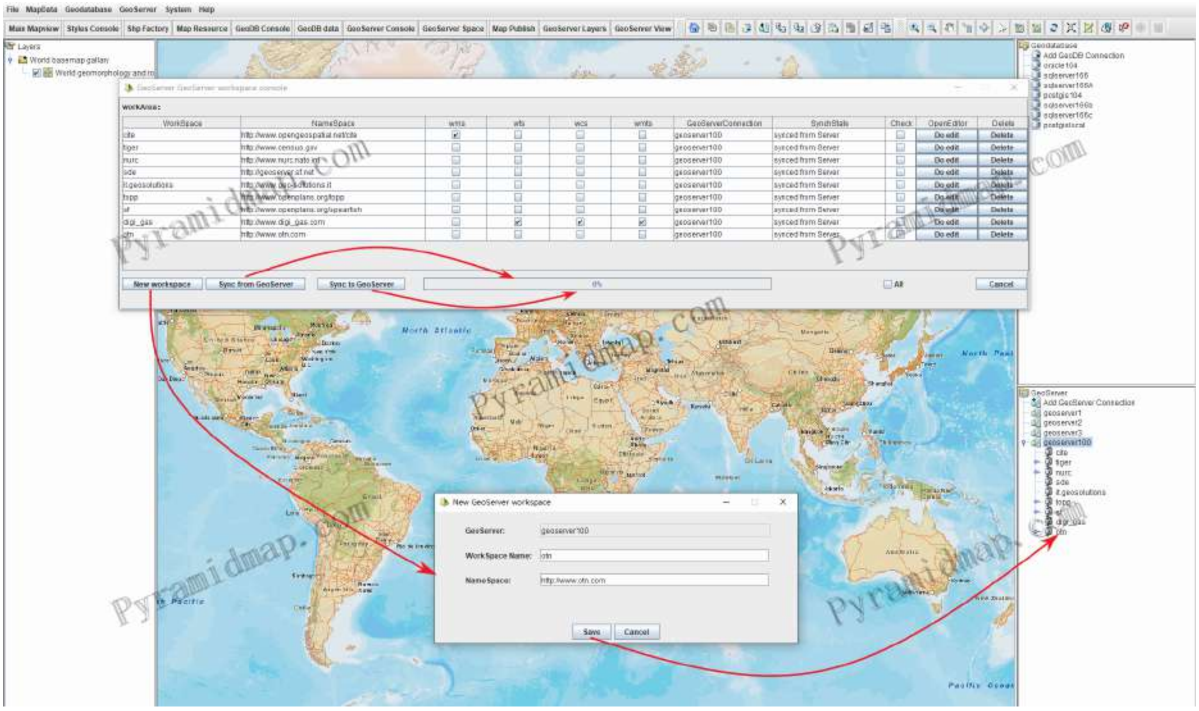


Figure 5-12： The new workspace interface and and data interaction with GeoServer

The new workspace will be automatically attached to the GeoServer node.

5.3.2 GeoServer data storage

In the workspace node, a data storage entry is provided, as shown in Figure 5-13.

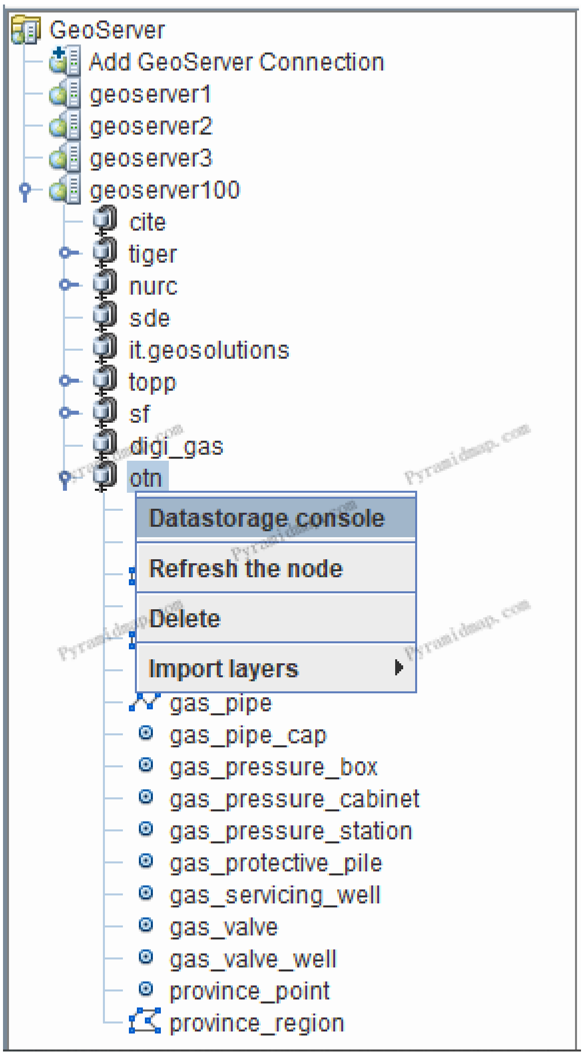


Figure 5-13：Workspace node to data storage management entrance

Then enter the data storage management module, as shown in Figure 5-14.

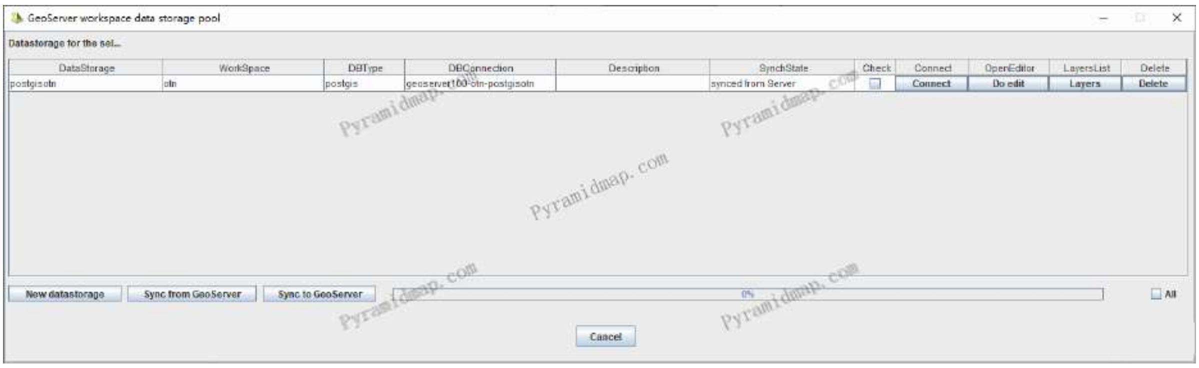


Figure 5-14：The data storage pool interface from the workspace node

In the data storage module, you can perform the following operations: create a new data storage, synchronize the server side data storage to the client, and submit the client side data storage cache to the server. In particular, in the data storage list management pool, you can provide independent editing and deletion operations for each data storage, as shown in Figure 5-15.

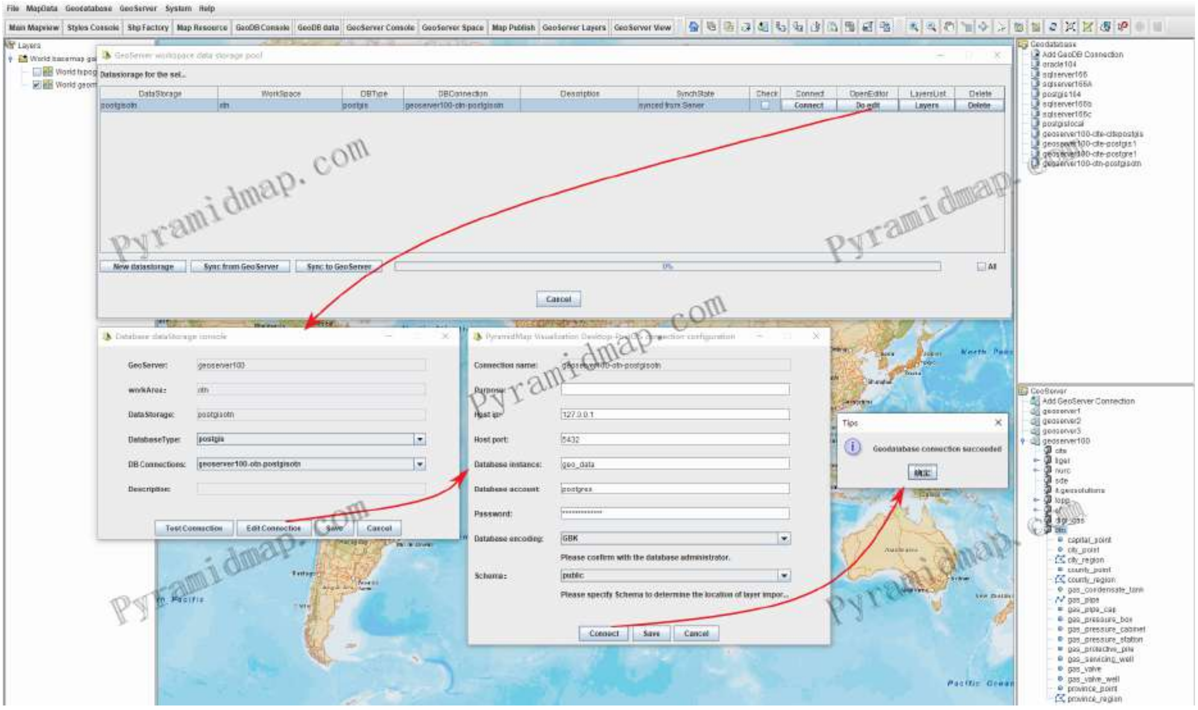


figure5-15： Data storage management pool of GeoServer workspace

In the New Data Storage module. You can select the database type corresponding to the data storage, and select the geodatabase connection that matches the type to automatically bring in the configuration of database connection parameters. The selected database connection can be dynamically edited, as shown in Figure 5-16.

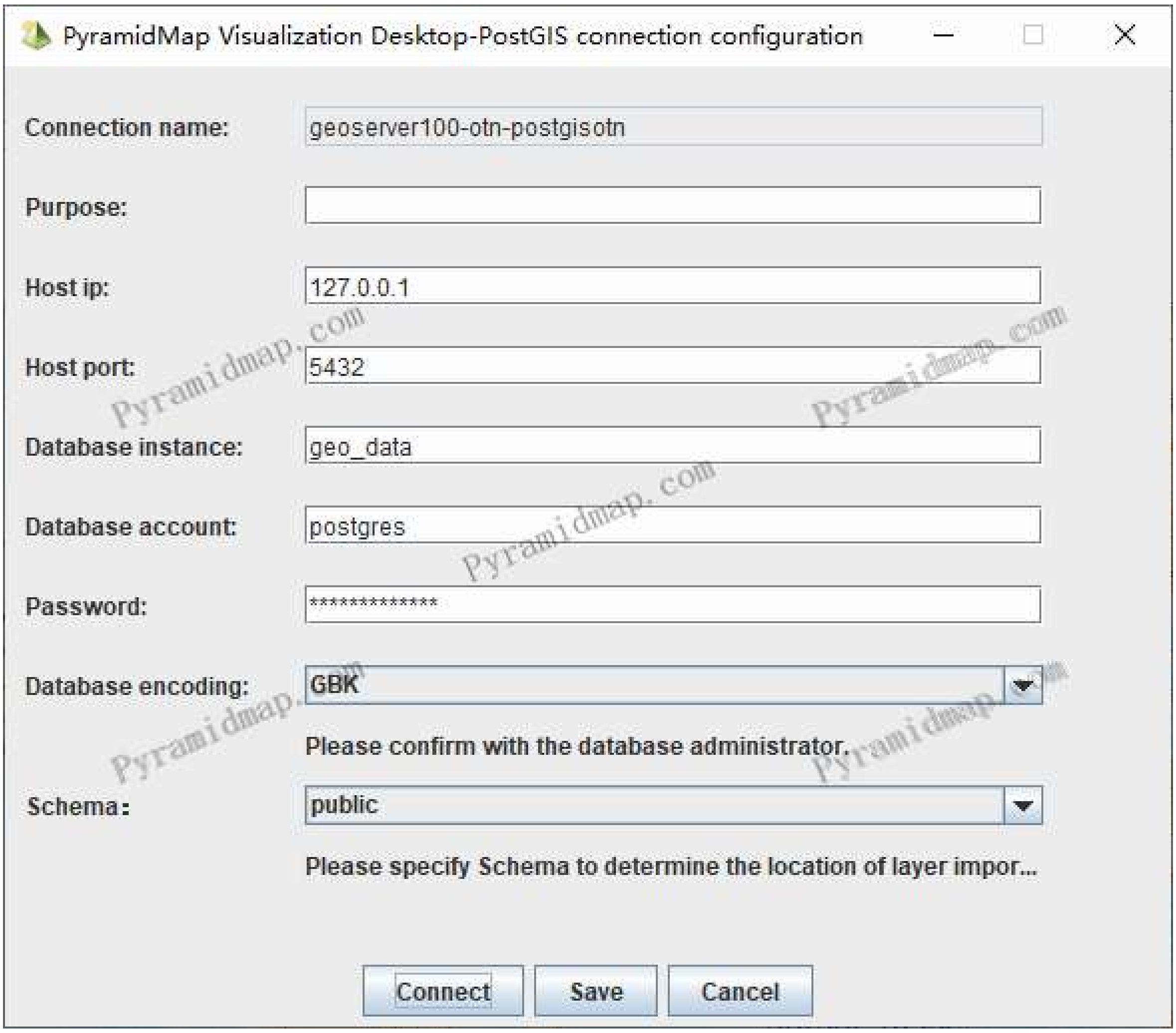


Figure 5-16： Edit the database connection for the datastorage

The predefined geodatabase connection serves as an independent resource management pool to provide data sources for GeoServer data storage.

5.3.3 GeoServer pool console

PyramidMap centrally manages GeoServer connections in the form of resource management pools, as shown in Figure 5-17.

Figure 5-17： GeoServer connections resource management pool entrance

PyramidMap creates and maintains the GeoServer resource connection pool, and manages and maintains its internal service space for each GeoServer connection. The module is shown in Figure 5-18.

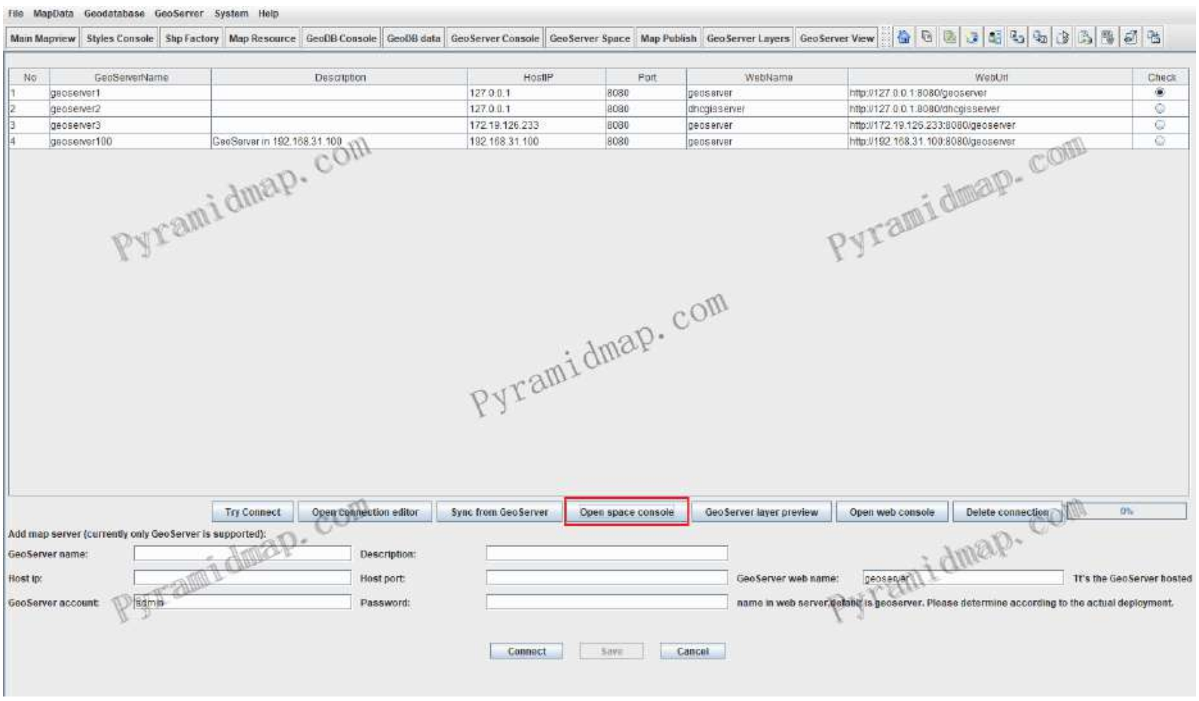


图5-18： GeoServer connections pool and operation options

Select the GeoServer and click “open space console“ button, as shown in Figure 5-19.

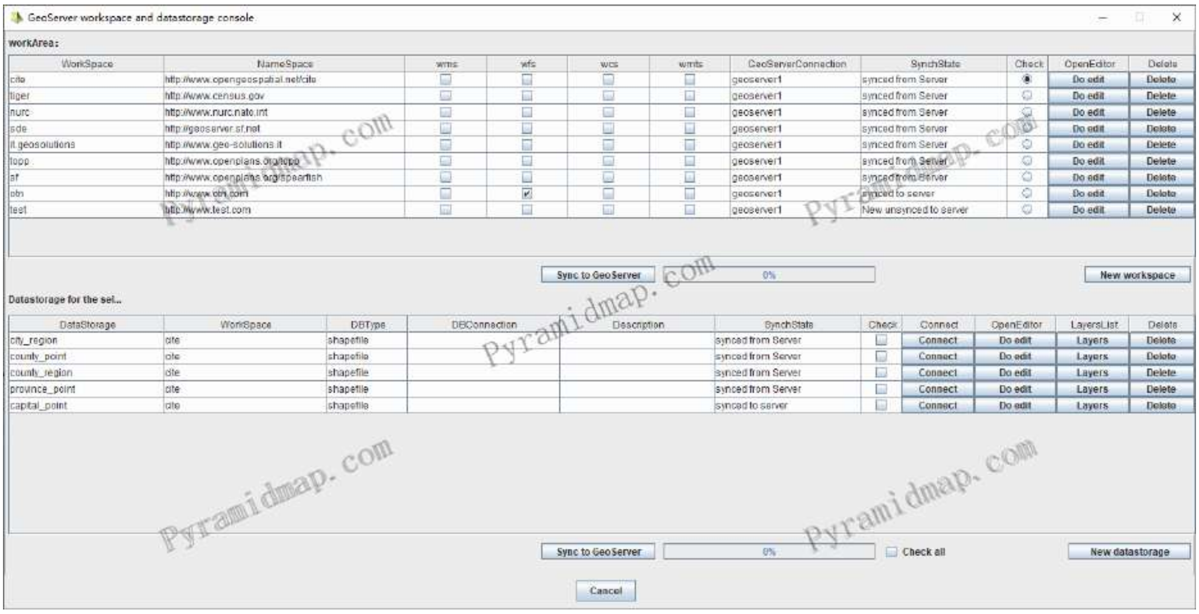


Figure 5-19： GeoServer workspace and data storage managing pool

The GeoServer workspace and data storage managing pool module displays the list of all workspaces of the server and the data storages list of each workspace. The client side and the server side maintain two-way synchronization to achieve visual management of the server side data by the client side.

In particular, PyramidMap implements a client cache maintenance mechanism for GeoServer. New and modified workspaces and data stores are temporarily stored on the client. When necessary, they are submitted to the GeoServer server in batches. At the same time, the GeoServer server data can be synchronized to the client at any time. This is the two-way synchronization mechanism between PyramidMap and GeoServer.

6 Map data and conversion

PyramidMap supports five types of map data sources, namely Shp file vector layer, file image layer, Geodatabase vector layer, GeoServer vector layer and raster tile layer.

6.1 Vector layers pool

PyramidMap imports the client's Shp file vector layers into the resource pool through a file browser to form a Shp management list pool and supports various operations with corresponding buttons, as shown in Figure 6-1.

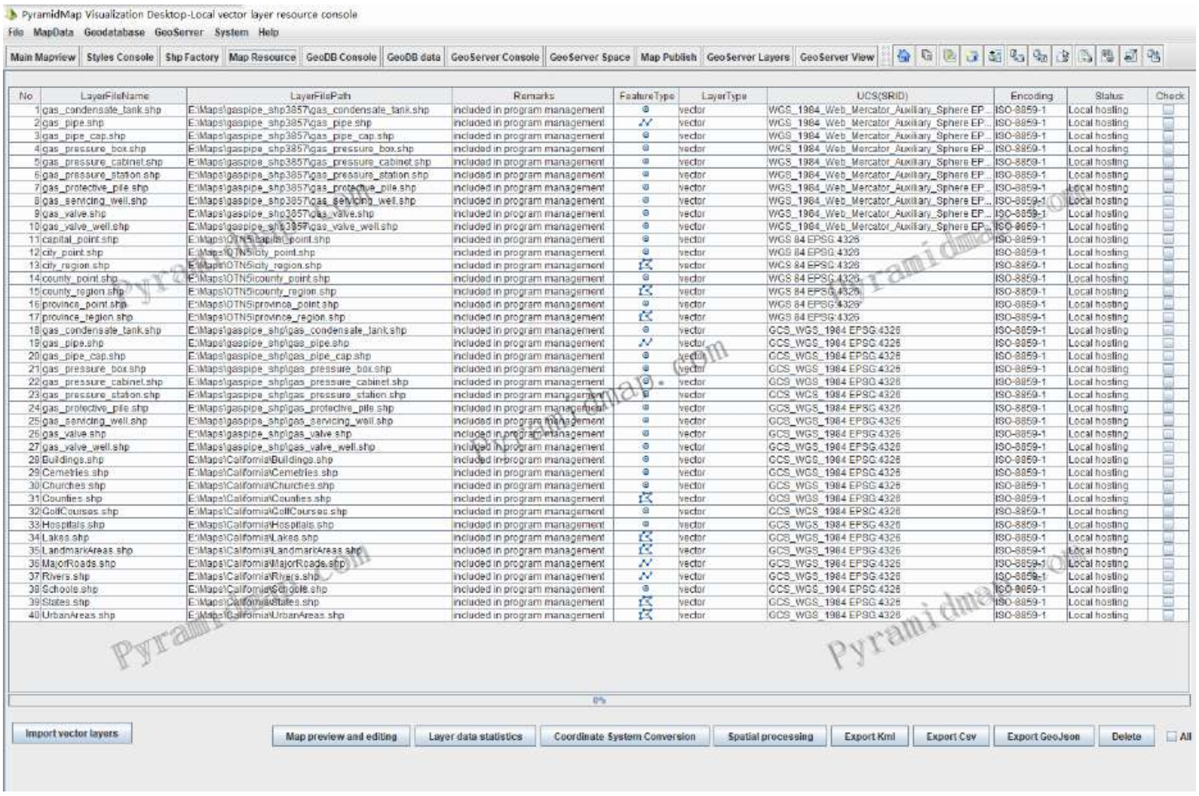


Figure 6-1：Shp layers resource pool

The vector layers pool displays information of every layer in a list format, including:

- Layer name.
- Layer file path.
- Data source types, including: local files, database layers, and GeoServer layers.
- Geometric indicate: Indicate the geometric type of the feature using an icon.
- Geometric type，include：Point/MultiPoint、LineString/MultiLineString、Polygon/MultiPolygon。
- Coordinate System (SRID): The definition of the coordinate system for a layer, including the coordinate system code.
- Data volume: The number of features in a layer.
- Status: The existence status of the current layer resource.

6.1.1 Layer preview

Through the map preview and editing options, the selected layer file will be opened in a separate map view to achieve layer preview and editing, as shown in Figure 6-2.

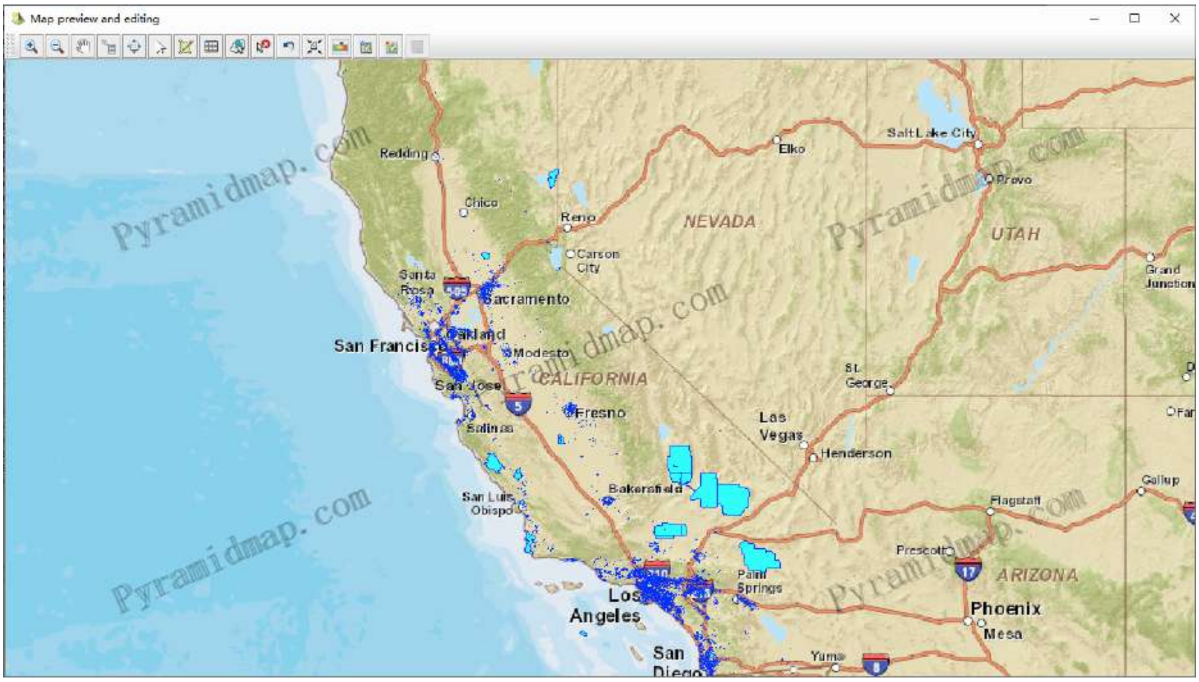


Figure 6-2：Open independent preview and editing view of single shp vector layer

Independent map view provides each layer with separate display, rendering, base map selection and overlay, graphical editing, attribute data editing in table, feature selection and deletion operations. It is a comprehensive map service for a single layer.

6.1.2 Data processing

Pyramidmap manages and operates layers uniformly through resource pooling, including layer resource import, layer preview, coordinate system conversion, spatial processing, layer slicing, layer data export, and other operations. As shown in Figure 6-3.

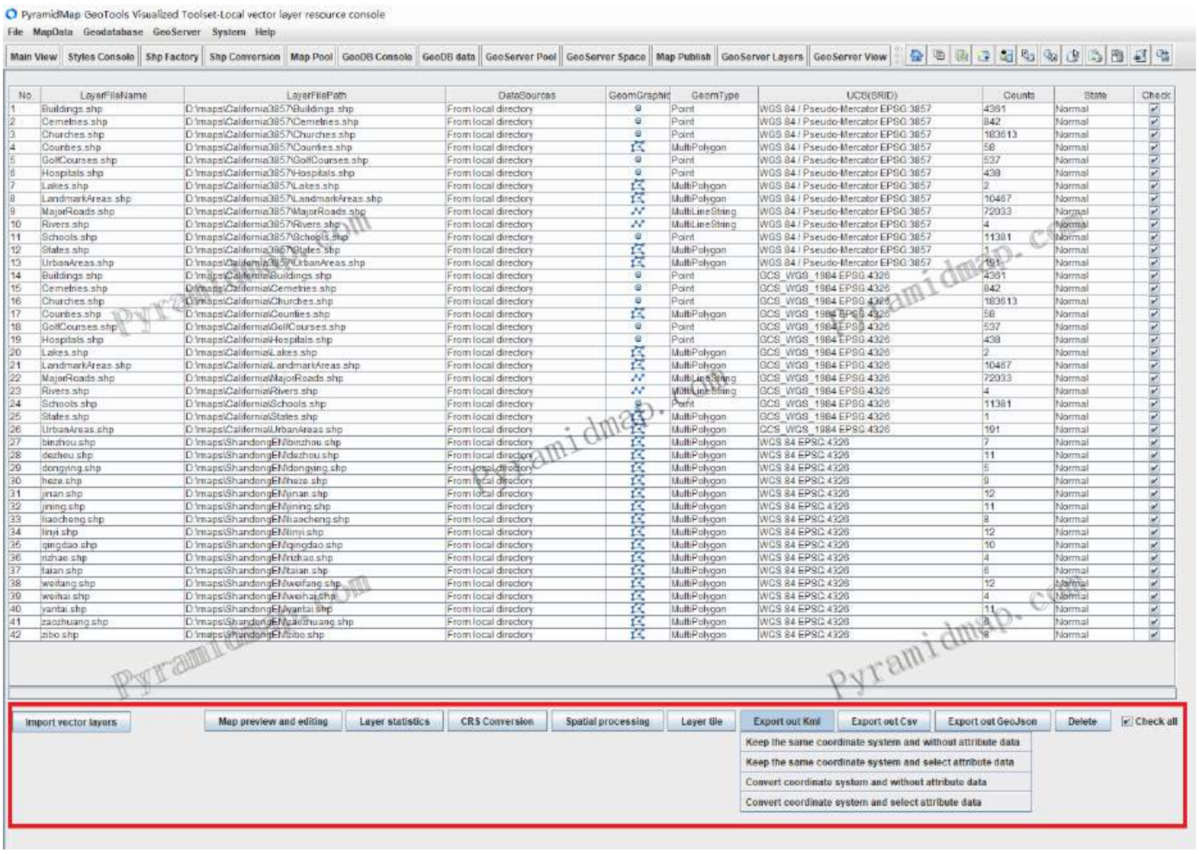


Figure 6-3: Data Processing Options for the vector layers pool

Support various operations on vector layers, including:

- Import vectorlayers: import client Shp files and GeoJson files into the resource management pool.
- Map preview and editing: preview and edit the Shp layer in independent view of each layer in the list pool.
- Layer data statistics: perform the features data statistics of the shp layer in the list pool.
- Coordinate system conversion: perform coordinate system conversion on the shp layer in the list pool.
- Spatial processing: perform spatial processing on shp layers in the list pool, including merging and intersection.
- Build layer tile: For vector layers, TMS standard tiles, XYZ standard tiles, MVT standard tiles, and MBTiles standard tiles are generated according to the preset tile format scale level.
- Data conversion: Shp to Kml, Shp to Csv, Shp to GeoJson.
- Delete: Delete the shp layer in the list pool.

6.1.3 Data Processing Overview

PyramidMap vector layer data processing covers coordinate system conversion, layer slicing, Shp and other heterogeneous spatial data, including mutual conversion between kml, csv, and Geojson. Technical details and examples will be explained in detail in [6.6 Coordinate System Conversion], [6.7 Data Conversion], and [6.8 Map tile].

6.2 Raster layers pool

PyramidMap imports raster image layers of client file types into the raster resource pool through a file browser to form an raster file management list, which supports various operations with corresponding buttons, as shown in Figure 6-4.

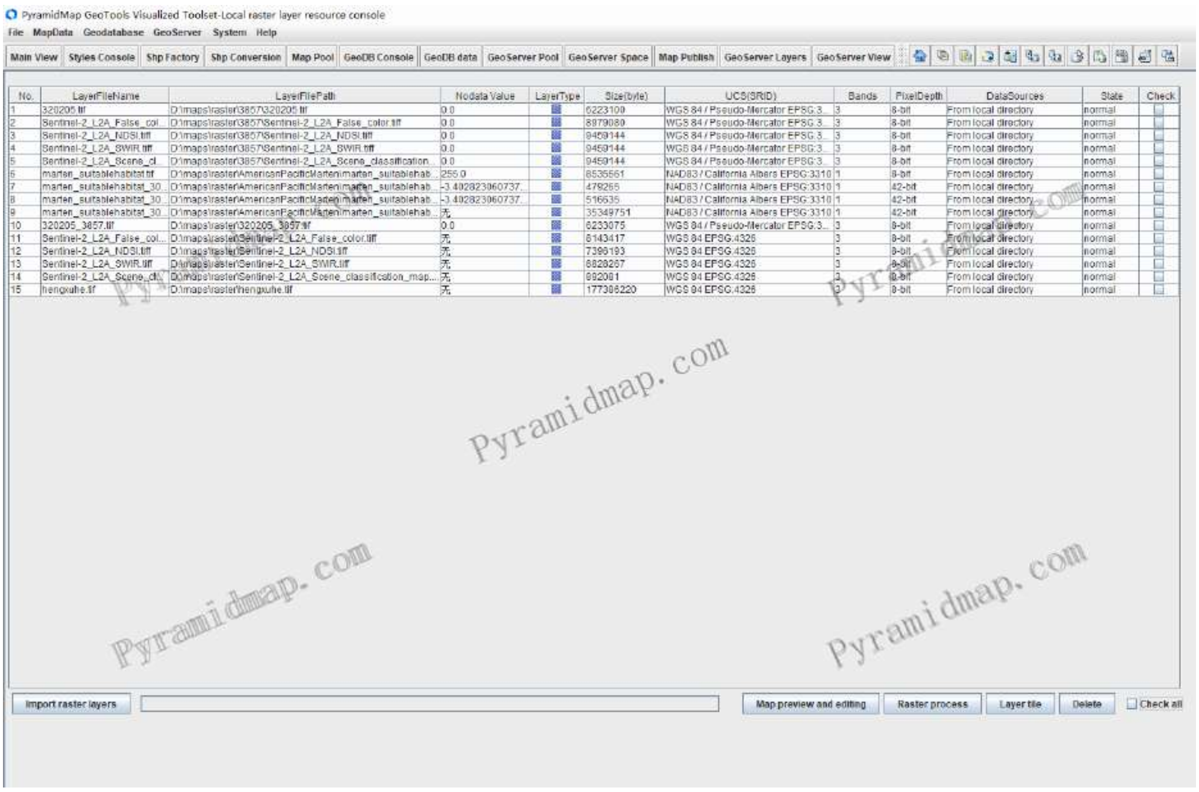


Figure 6-4： Raster layers pool

The grid layer resource pool displays layer information for unified resource management in a list format, including:

- Layer name.
- Layer file path.
- NoData value: NoData indicates whether the pixel position information in the grid is available. If a pixel position information is not available, that position will be designated as NoData. Please note that NoData is different from 0, which is a valid numerical value.
- Layer Type: Display the grid layer type in icon mode.
- Size: The data size of the raster layer.
- Coordinate System (SRID): The definition of the coordinate system for a layer, including the coordinate system code.
- Band number: The number of bands in the raster layer.
- Pixel bit depth: The bit depth (pixel depth) of a pixel determines the range of values that a particular raster file can store, which is based on the formula 2n (where n is the bit depth). For example, an 8-bit raster can have 256 unique values that range from 0 to 255.
- Data source types, including: local grid layer, database export layer, and GeoServer export layer.
- Status: The existence status of the current layer resource.

6.2.1 Layer preview

Through the map preview and editing options, the selected layer file will be opened in a separate map view to achieve layer preview and editing, as shown in Figure 6-5.

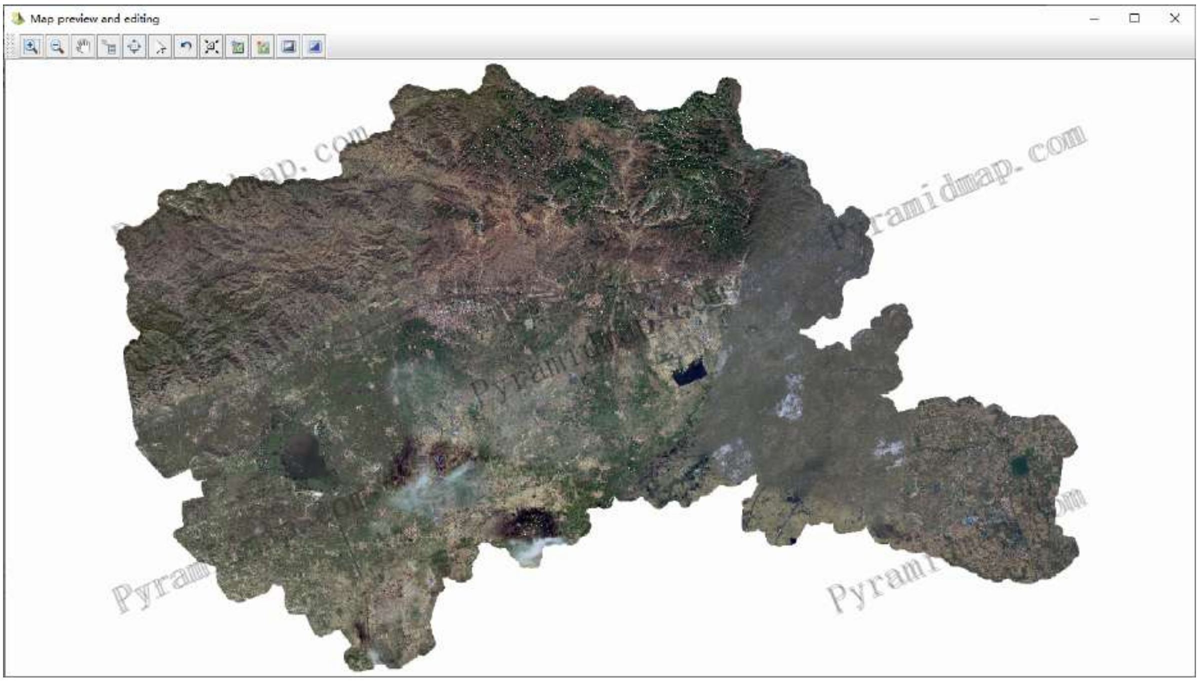


Figure 6-5: Map preview of the raster layer

The independent view provides separate display, rendering, and overlay for each layer, and is a comprehensive map service window for a single layer.

6.2.2 Data processing

Pyramidmap manages and operates layers uniformly through resource pooling, including layer resource import, layer preview, coordinate system conversion, spatial processing, layer tile, layer data export, and other operations. As shown in Figures 6-6.

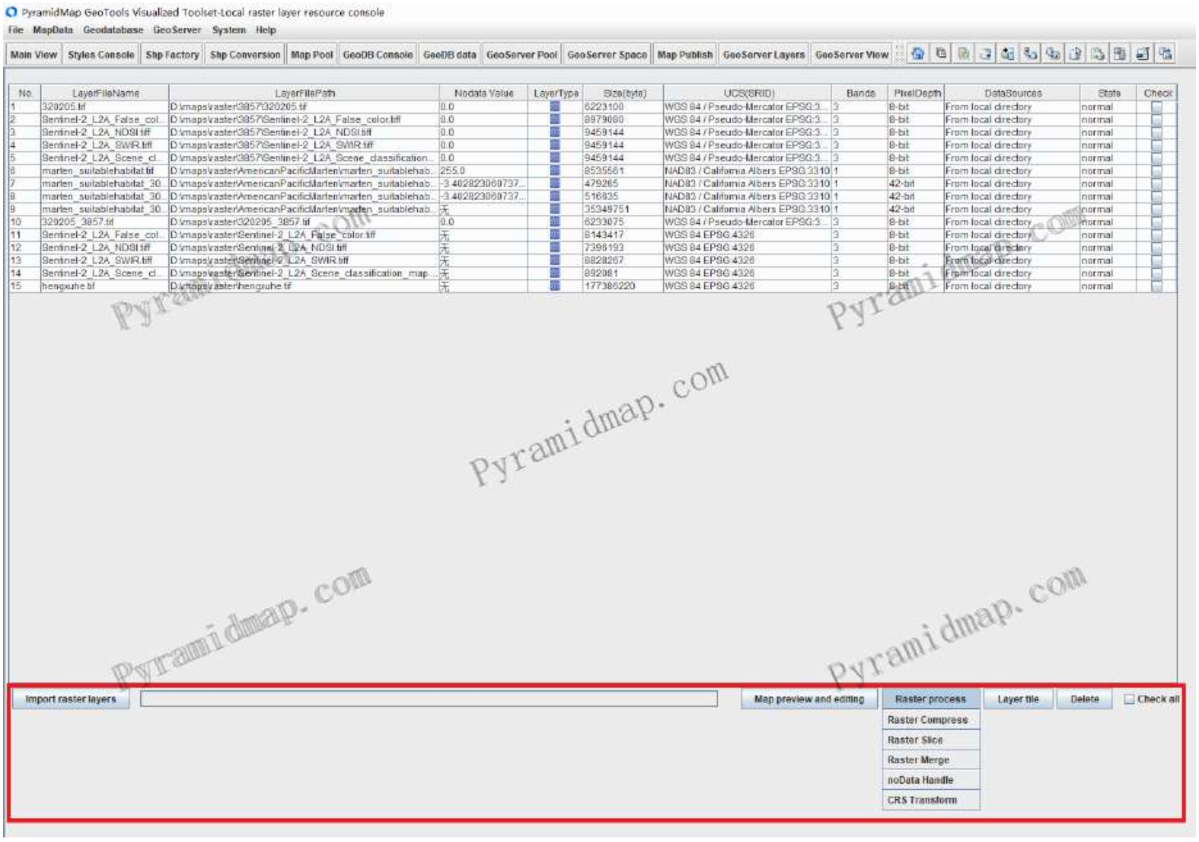


Figure 6-6： Grid Layer Resource Pool Data Processing Options

Support various operations on raster layer resources pool, including:

- Import raster Layer: import the client raster layer file into the resource management pool.
- Map preview and editing: preview and edit the raster layers in the list pool.
- Raster compression: Compress the raster layers according to the specified compression scheme.
- Raster slice: This is a process that reverses with the raster Mosaic, divide a whole image into multiple slice images according to rules.
- Raster merge: A mosaic combines multiple raster images to obtain a seamless raster.
- Deal noData: process the invalid value of noData for the raster layer in the list pool, mainly to remove the black background and other invalid pixels of the image.
- Coordinate system conversion: perform coordinate system conversion on raster layers in the list pool.
- Build layer tile:: Generate TMS standard tiles, XYZ standard tiles, and MBTiles standard tiles according to the preset tile format scale level for raster.
- Delete: Delete the raster layer in the list pool.

6.2.3 Raster compress

The main advantage of image compression is that the compressed data requires less storage space, and because less information is transmitted, it accelerates the display of the data. The PyramidMap image compression interface is shown in Figures 6-7.

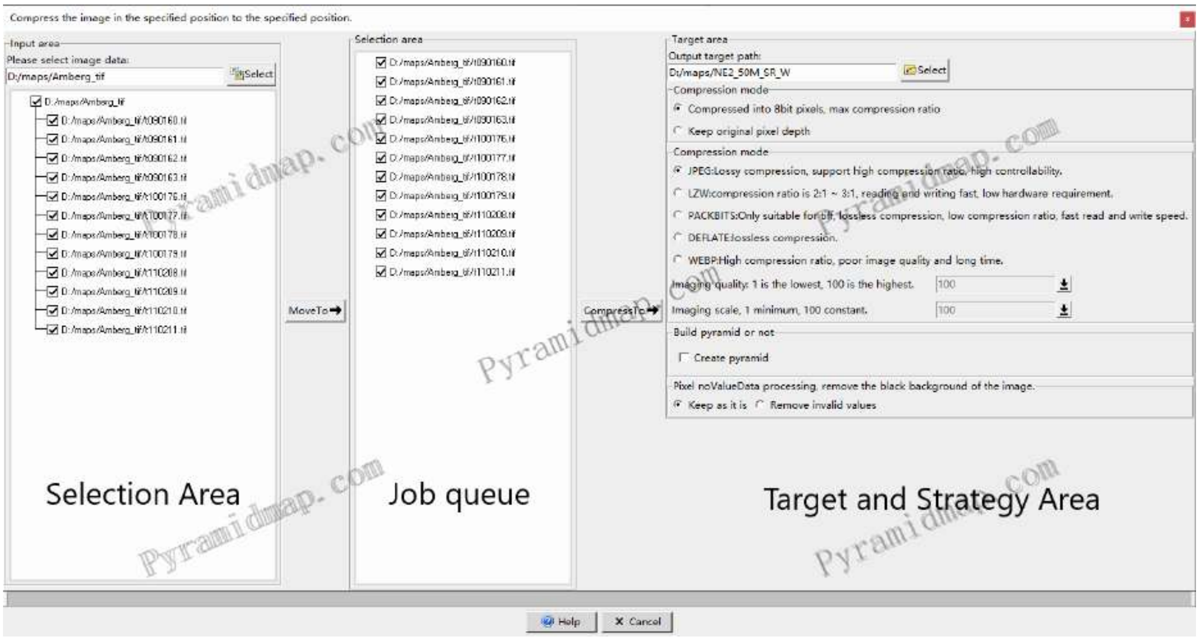


Figure 6-7: Performing raster Compression

PyramidMap supports configuring the same compression strategy for batch data, achieving one click completion. PyramidMap configures image compression strategies through parameters:

- Compression depth: By compressing the pixel depth of the original image to 8 bits, the maximum compression ratio can be achieved, but at the cost of reducing the image color scale, grayscale, and image quality. The original pixel depth can be retained.
- Compression method: The data compression method can be lossy compression (JPEG/JPEG 2000, WEBP) or lossless compression (LZW, PackBits, and DEFLATE). Lossless compression means that the pixel values in the raster dataset will not change or be lost. If the pixel values of the raster dataset will be used for analyzing or deriving other data products, lossless compression or no compression should be chosen. The amount of compression depends on the data and compression quality. The more similar data, the higher the compression ratio. The lower the compression quality, the higher the compression ratio. Compared to lossless compression, lossy compression usually produces a higher compression ratio.
- Imaging quality: When selecting the **JPEG** compression method, the compression quality must be specified. The effective range of compression quality values is 1 to 100, with 75 being the default value.
- Imaging Scale: Specify the width/height ratio of the compressed image to the original image.
- Build a pyramid: Option whether to build a pyramid structure on the compressed image. Building a pyramid can perform hierarchical indexing on the image, which will accelerate the image loading speed. This is very effective in improving the loading and display speed of big data images.
- Pixel noData processing: Remove unnecessary values created around raster data. The specified value is different from other useful data in the raster dataset. For example, values with zero on the grid boundary are different from zero values within the grid dataset. The specified pixel value will be set to NoData in the output grid dataset. For file based grids, to

ignore the background value, it must be set to the same value as **NoData**. Invalid values in the image can cause a black background. Choosing the invalid value processing option will automatically filter out invalid pixels during the compression process, thereby improving image quality and simplifying the processing process.

The processing process will save the log, and a message prompt will be given after the processing is completed.

6.2.4 Raster slice

Raster slice is the process of dividing a raster dataset into smaller segment based on the features in blocks or faces. The schematic diagram of image slicing is shown in Figures 6-8.

Figure 6-8: Schematic diagram of raster slice

PyramidMap can perform batch slicing operations on rasters in the resource pool, and the raster slicing dialog is shown in Figure 6-9.

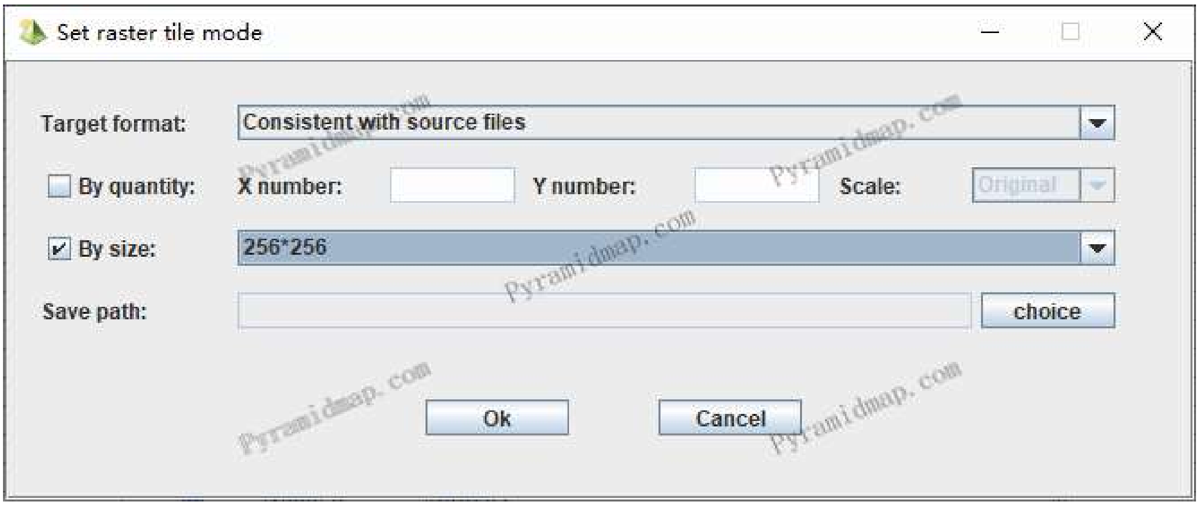


Figure 6-9: Raster Slicing procssing

Select the tile file format, which can be consistent with the source file, or select Tiff, png, jpg, webp and other different data formats. You can creating tiles according to the number and size strategies, and finally output to the specified save path.

With the NE2_ 50M_ SR_ W.tif in the resource pool as an example, the whole image is shown in Figure 6-10.

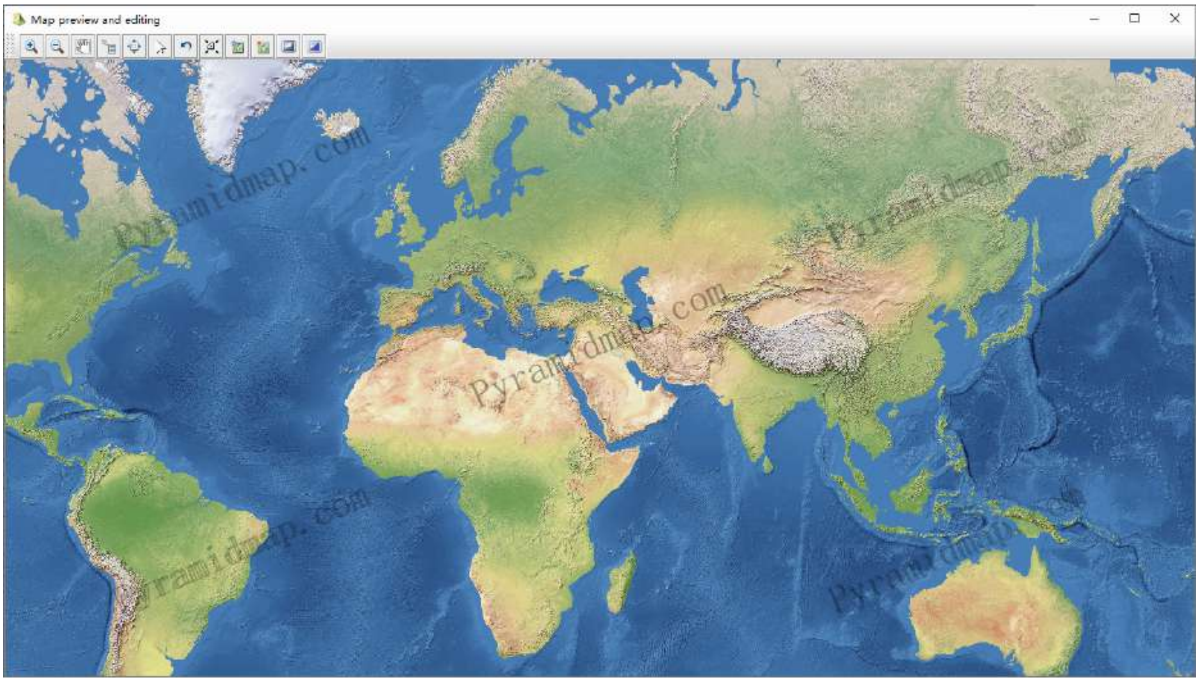


Figure 6-10： the whole raster image

Taking the 256 * 256 (pixel) slicing strategy as an example, the generated tiles is shown in Figure 6-11.

Figure 6-11：The effect of tiles generated by the whole raster

6.2.5 Raster merge

In a sense, raster mosaicing is the reverse process of the raster tiling. Mosaicing is to combine multiple raster images through mosaic to obtain a seamless raster image. The image merging diagram is shown in Figure 6-12.

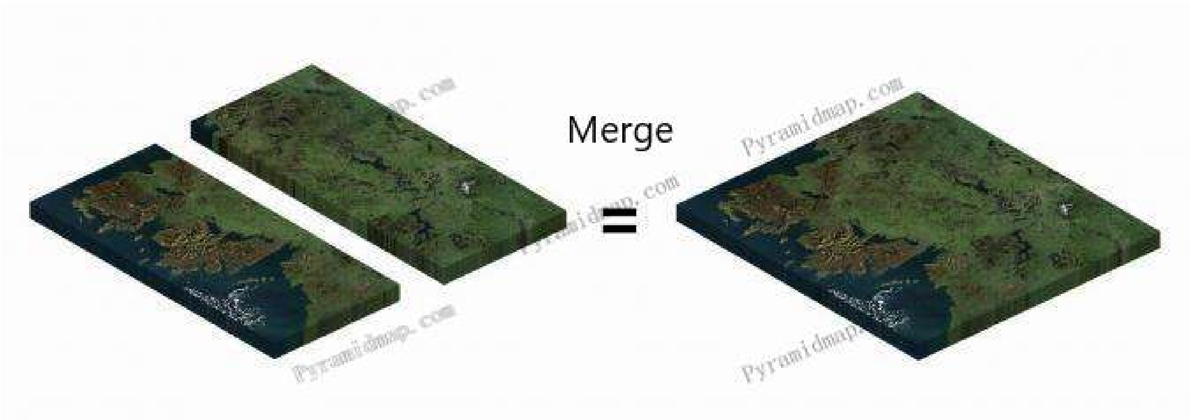


Figure 6-12：Schematic diagram of raster mosaicing

PyramidMap batch mosaicing rasters in the resource pool, as shown in Figure 6-13.

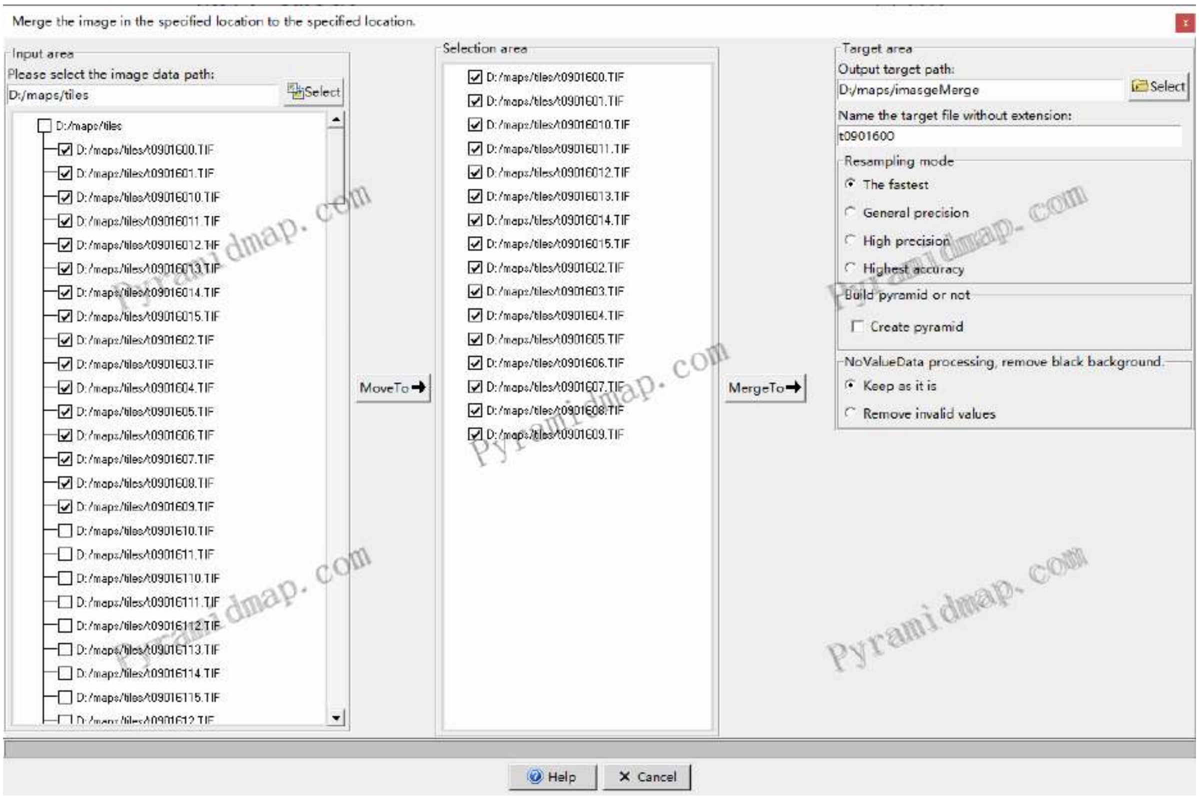


Figure 6-13：PyramidMap mosaicing rasters in the resource pool in batches

The effect of the merged raster is shown in Figure 6-14.

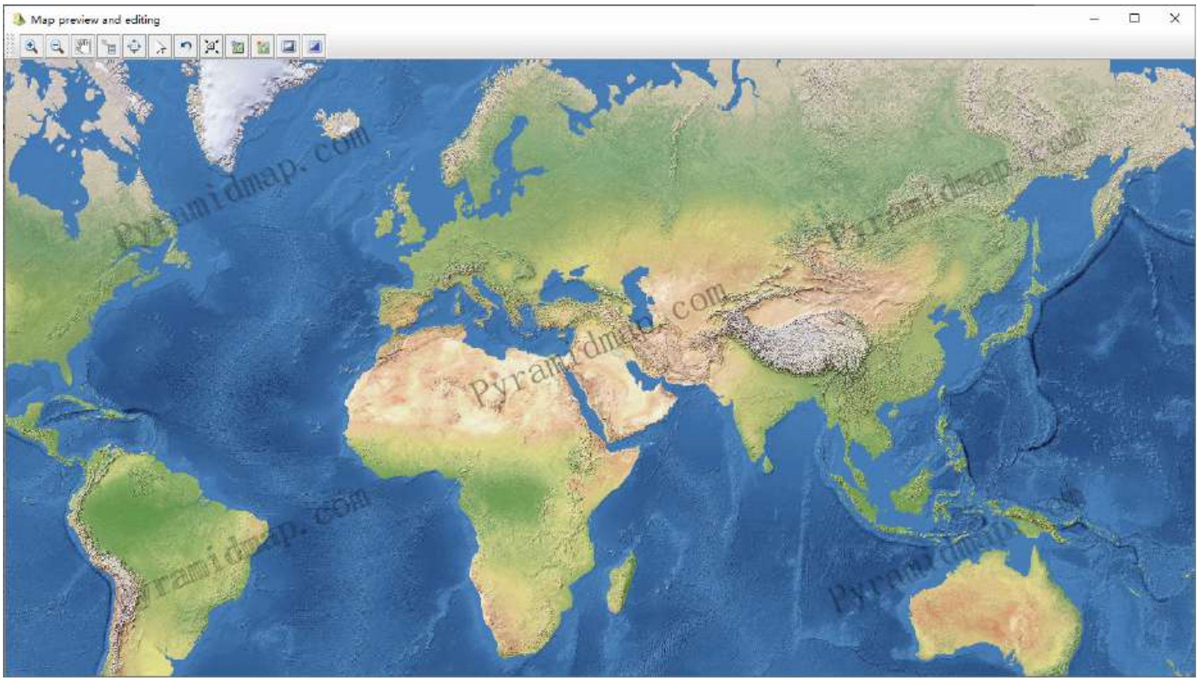


Figure 6-14: The effect of rasters mosaicing

Two common methods for combining adjacent or overlapping raster layers are MERGE and MOSAIC. The following is an explanation of the differences between these two types of work. The MOSAIC and MERGE functions work similarly, but there are also differences between them. When the input grids do not have any overlapping area, there is no difference in the output from either MERGE or MOSAIC. The difference is only in the processing of overlapping areas. In the case of MERGE, the sequence of input grids determines the values assigned to cells, giving priority to those entered first. The merge processing mode is shown in Figure 6-15.

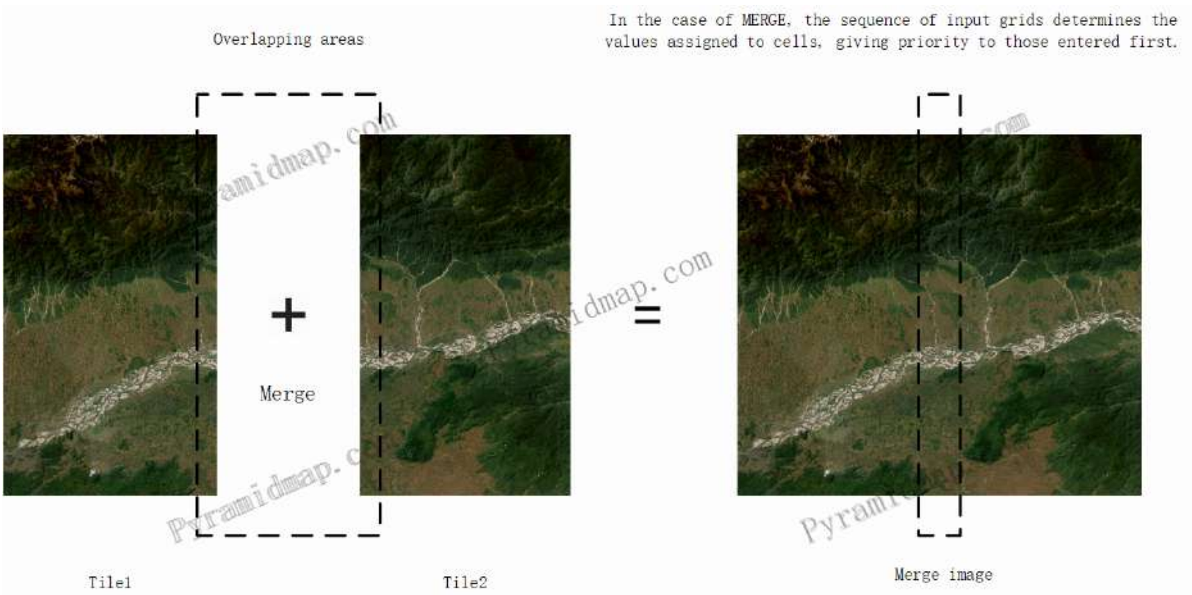


Figure 6-15: The merge processing mode

6.2.6 NoData processing

In general, raster images will have NoData values. These NoData are invalid values that do not have actual value and will interfere with image data analysis. Black image frame is a common NoData error, which is particularly common for UAV images and raster data sets after geographical reference processing. In this example, we remove the black border of raster image through NoData invalid value processing. The processing process is shown in Figure 6-16.

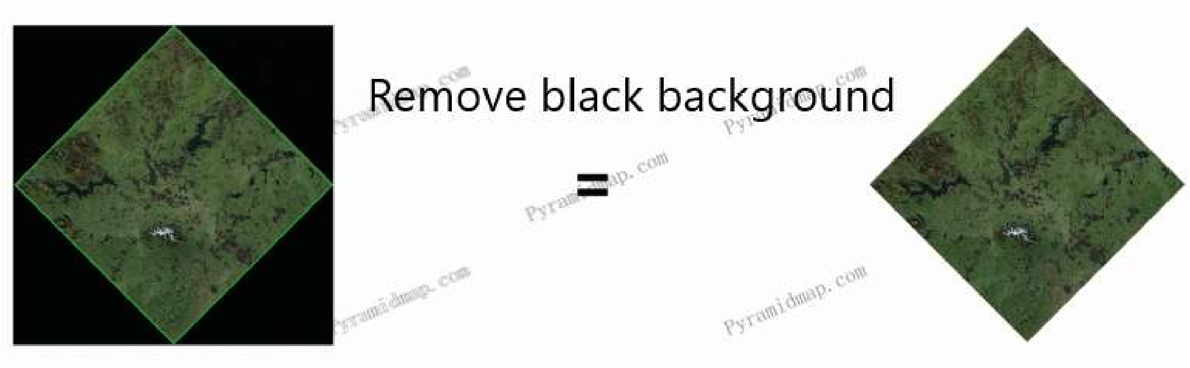


Figure 6-16：Remove the black border of raster image through NoData invalid value processing

PyramidMap perform batch NoData processing on images in the resource pool, as shown in Figure 6-17.

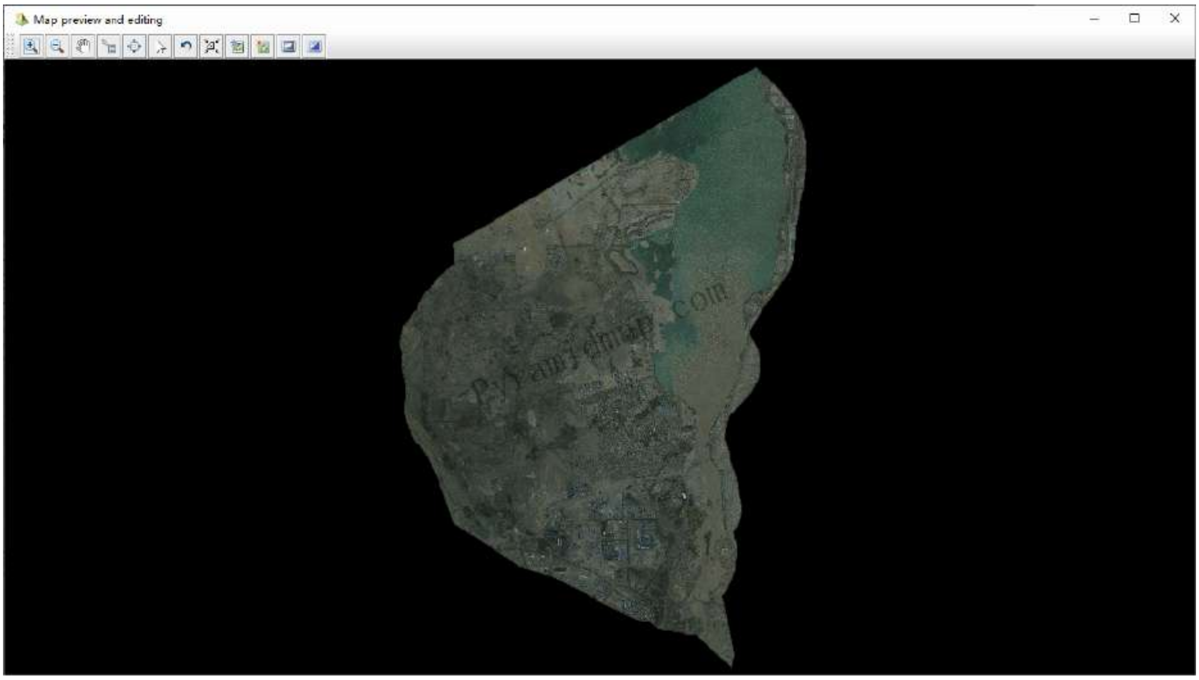


Figure 6-17：Batch NoData processing of images in the resource pool

The image with black background before processing is shown in Figure 6-18.

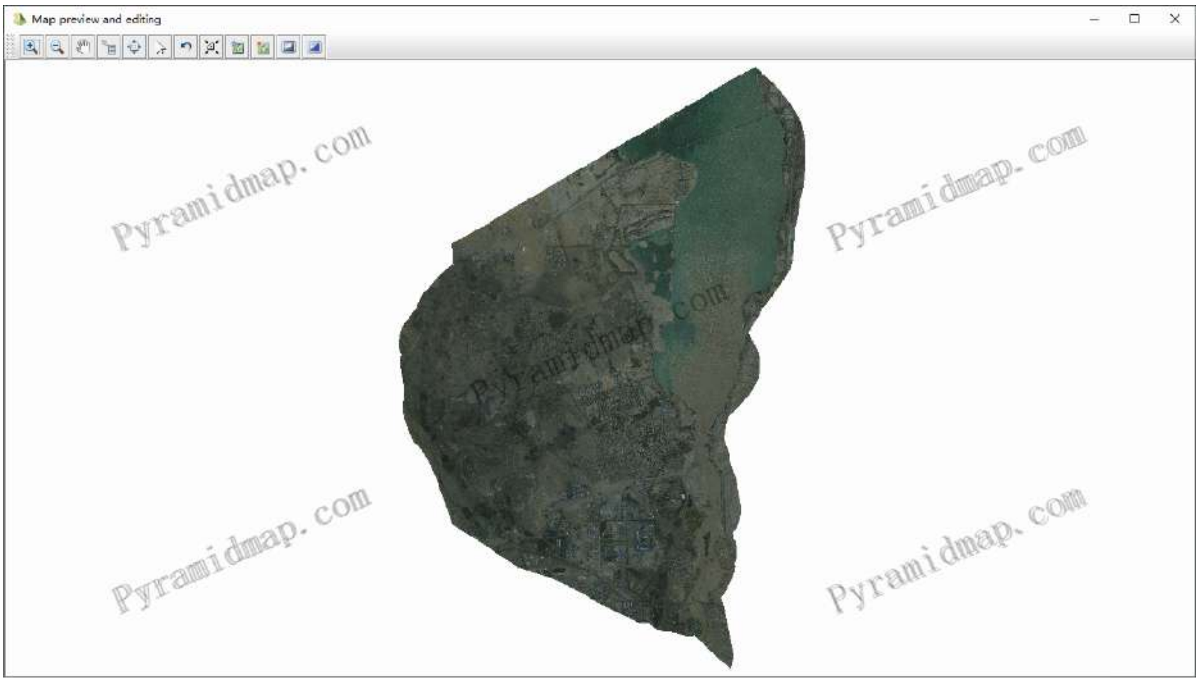


Figure 6-18：The image with black background before processing

6.2.7 Coordinate System Conversion

PyramidMap支持对Shp矢量文件类型和栅格文件类型图层的坐标系转换，在上述两种类型的图层资源列表中选择图层文件，选择“坐标系转换”，如图6-19所示。

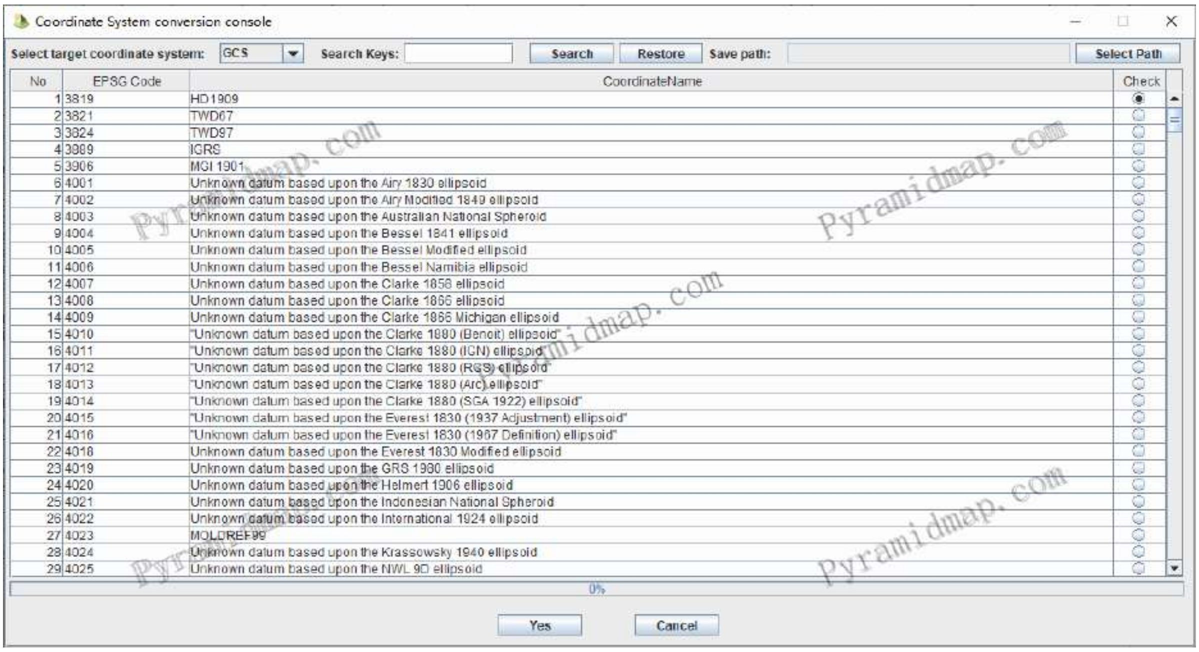


Figure 6-19：Select in coordinate system resource pool for conversion

In the coordinate system conversion interface, select the type of target coordinate system, which is divided into two types: spatial coordinate system and plane coordinate system, as well as many standardized coordinate systems to which it belongs. It supports global search by name and EPSG code. Taking WGS84 as an example, we can perform global keyword search according to coordinate system code 4326 to accurately obtain the target coordinate system we want.

6.3 Geodatabase layers pool

PyramidMap imports client shp vector layers into Geodatabase through database connection pool, obtains layer list through database connection pool, and supports various operations with corresponding buttons.

6.3.1 Importing Shp into Geodatabase

PyramidMap maintains the Geodatabase connection pool and the Shp vector layer resource pool to import the Shp vector layer into the Geodatabase in batches. It supports but is not limited to Oracle, PostGIS and MySQL. The import interface and process are shown in Figure 6-20.

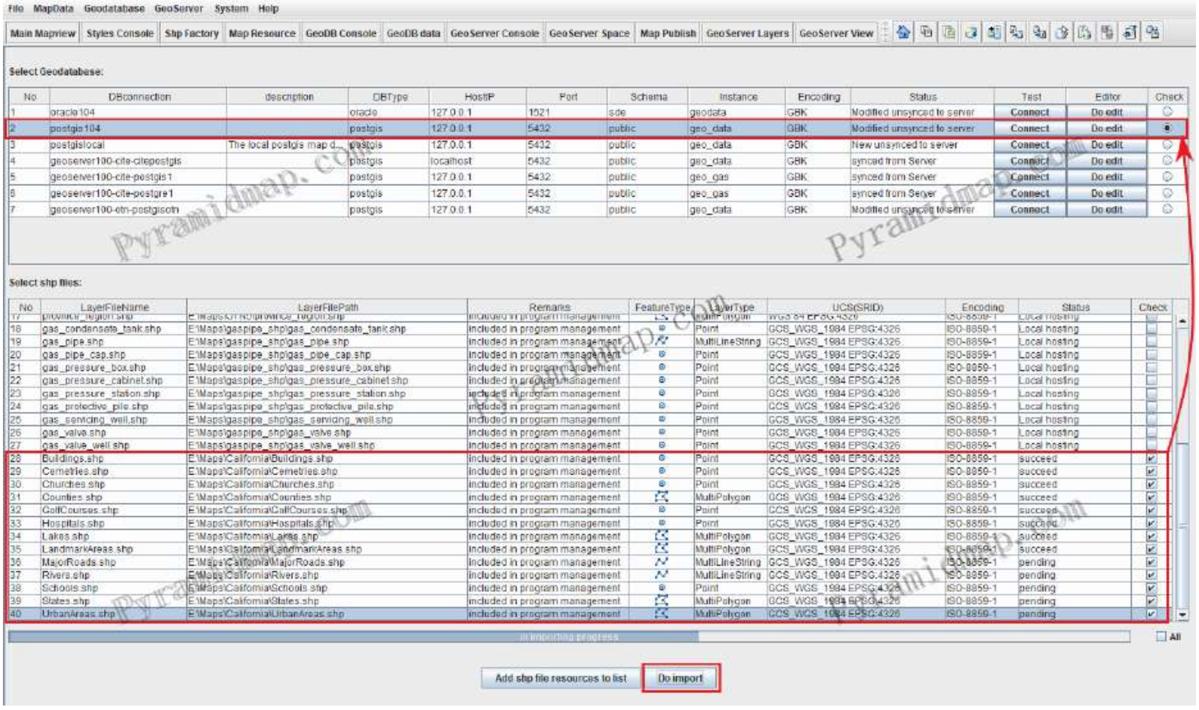


Figure 6-20： Shp vector layerS imported to Geodatabase workflow

In this module, users can selectively import Shp vector layers into Geodatabase in batches. After importing, each Shp layer generates a layer feature table with the same name. As a reciprocal process, the layer feature table in Geodatabase can also be exported to different geographic feature data such as Shp, Csv, Kml, GeoJson, etc.

6.3.2 Geodatabase exporting out shp

As a reciprocal process, the features table in Geodatabase can also be exported to different geographical element data such as Shp, Csv, Kml, GeoJson, etc. The operation interface and flow are shown in Figure 6-21.

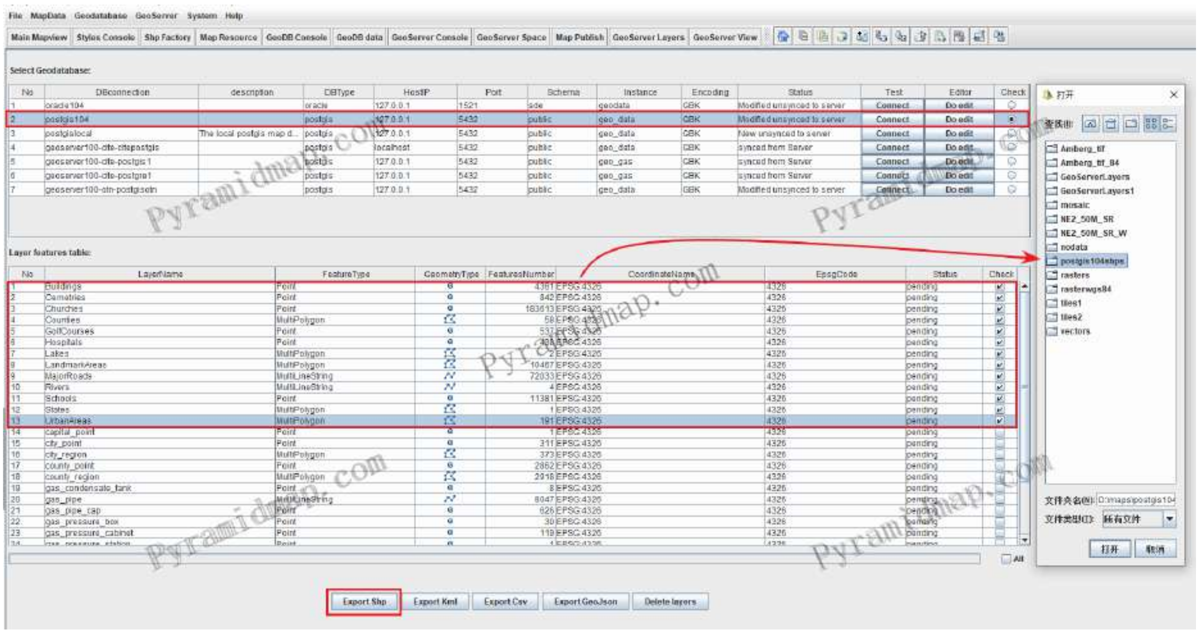


Figure 6-21： Geodatabase layer table export interface and workflow chart

In this module, users can selectively export the layer tables in Geodatabase as Shp, Csv, Kml and GeoJson under the specified path. Taking the exported Shp file as an example, the export process, export status and export file are shown in Figure 6-22.

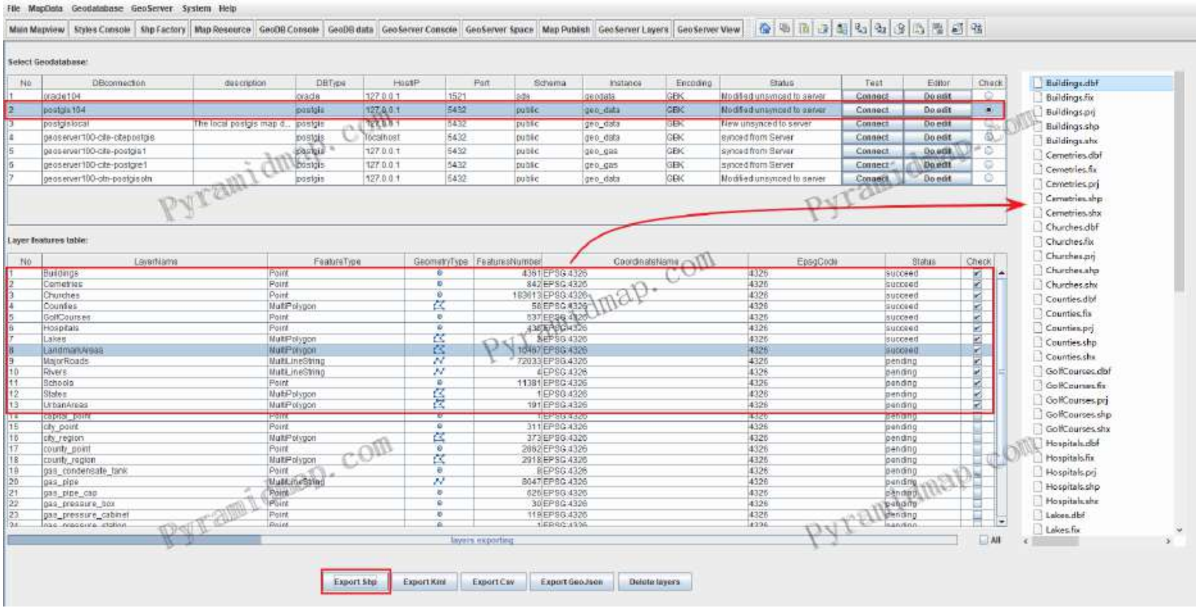


Figure 6-22：Geodatabase layer table export process, export status and export file flow chart

Select the database connection, dynamically obtain its internal layer resources, perform corresponding processing, and support various operations on layer resources with corresponding buttons, including:

- Export Shp：Export Geodatabase layers to shp files.
- Export Kml：Export Geodatabase layers to kml files.
- Export Csv：Export Geodatabase layers to csv files.
- Export GeoJson：Export Geodatabase layers to geojson files.
- Delete layers：Bulk delete the selected Geodatabase layers.

Reload and display the exported Shp layer in the map view, as shown in Figure 6-23.

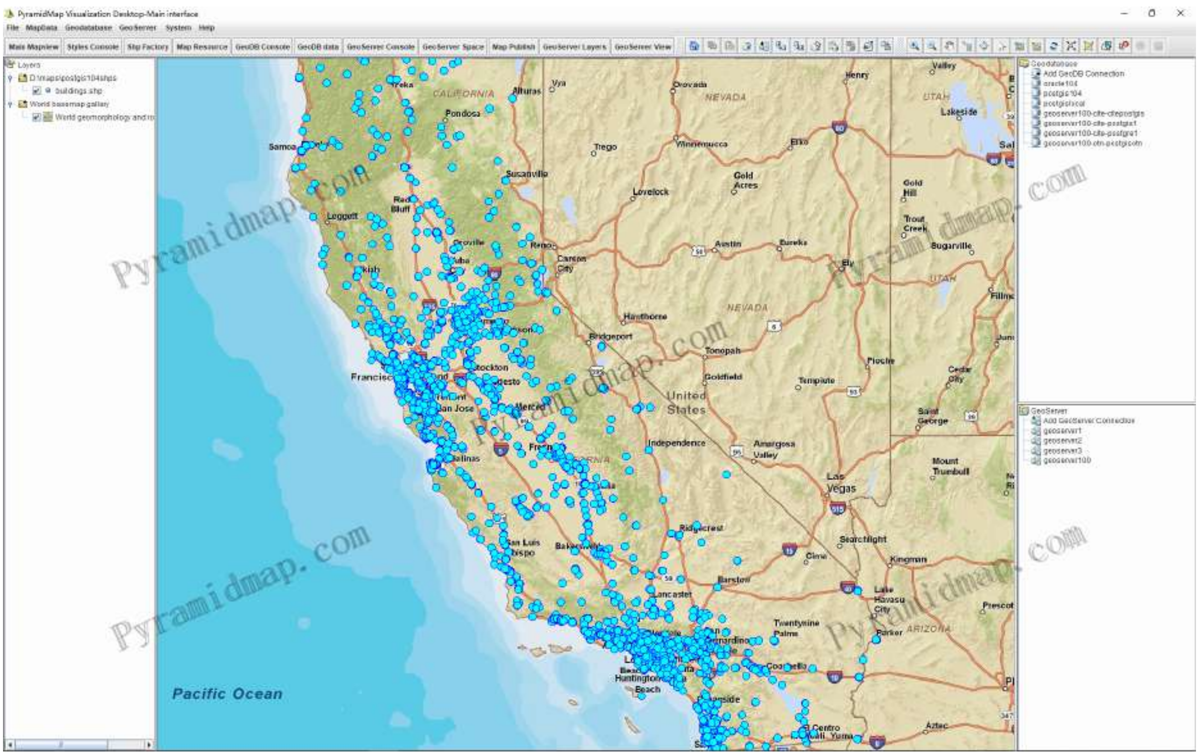


Figure 6-23：Display of Shp layer exported from Geodatabase

6.3.3 Geodatabase layers preview and edit

The layer table in Geodatabase can be queried and previewed directly. The operation interface and flow diagram are shown in Figure 6-24.

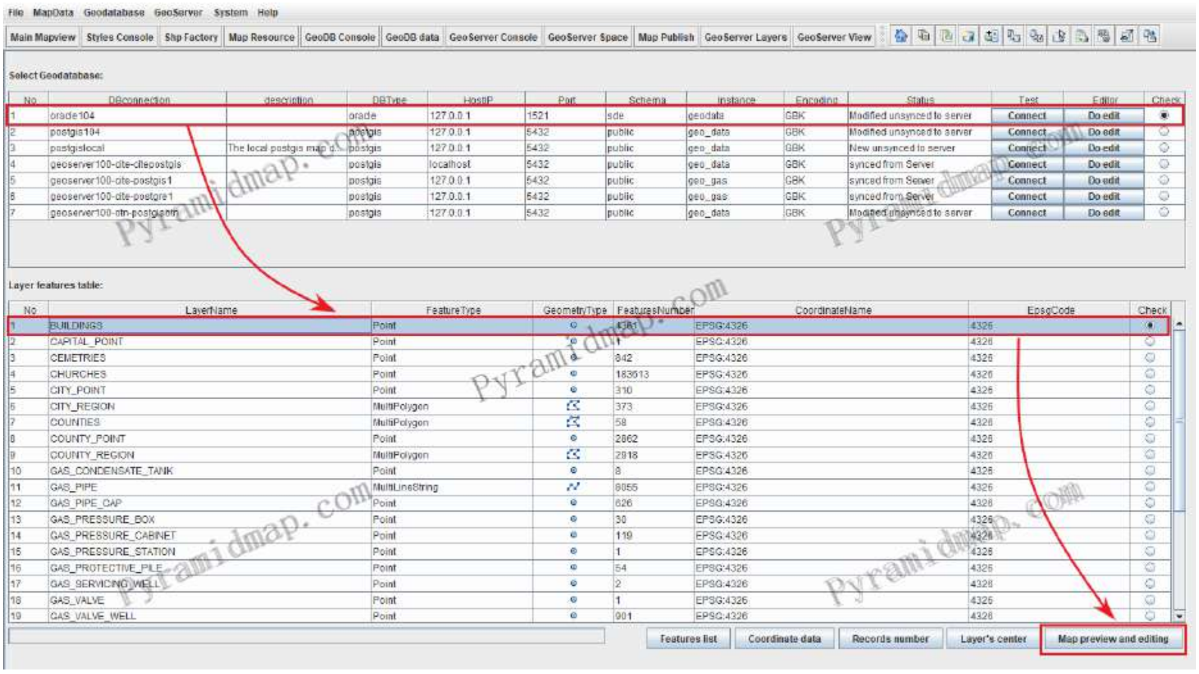


Figure 6-24： Geodatabase layer table query and preview

Select the database connection, dynamically obtain its internal layer resources, perform corresponding processing, and support various operations on layer resources with corresponding buttons, including:

- Feature List: forms the feature list of the selected layer.
- Coordinate data: form a list of geographical coordinates of the selected layer.
- Records number: perform data statistics on the features in the layer.
- Layer's center: calculates the center point location of the layer.
- Map preview and edit: preview and edit the layers in the management pool.

In particular, through the map preview and editing options, the selected layer file will be opened in an independent map view to preview and edit the layer, as shown in Figure 6-25.

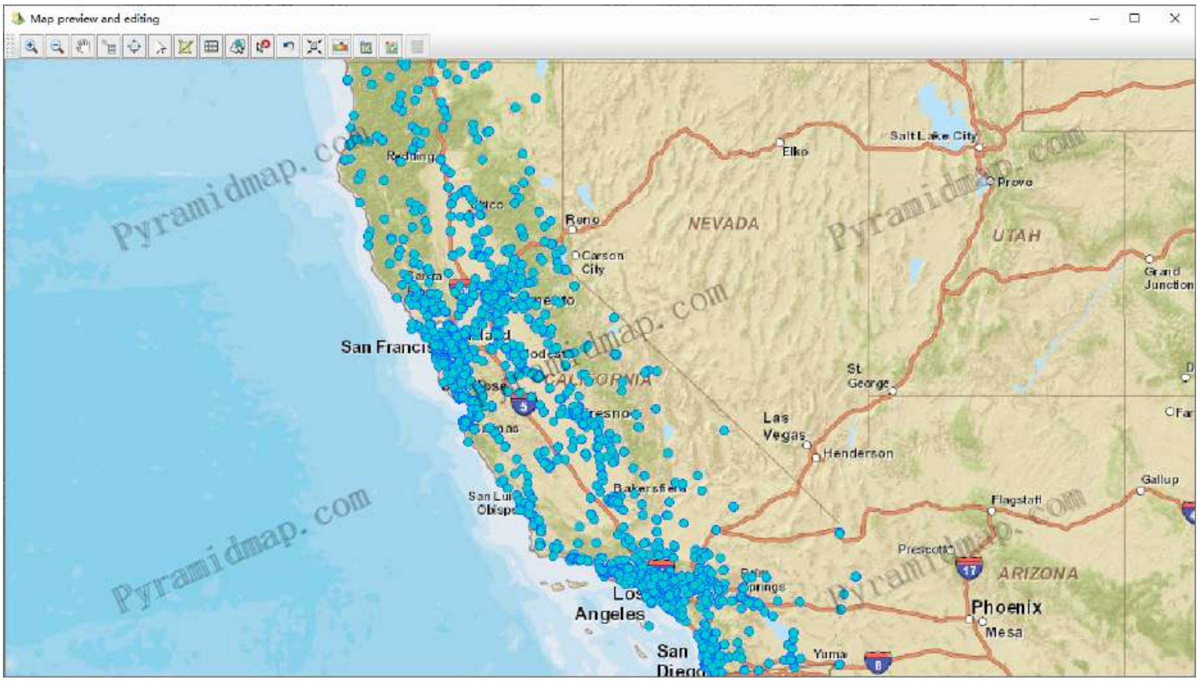


Figure 6-25：Open the independent preview and editing view of the single geodatabase layer

Independent map view provides each layer with separate display, rendering, base map selection and overlay, image-based editing, data table editing, feature selection and deletion operations. It is a comprehensive map service for a single layer.

6.4 GeoServer vector layers pool

PyramidMap publishes vector layers of client Shp file to GeoServer through GeoServer connection, and obtains the Internal layers through GeoServer connection in the pool, supporting various operations with corresponding buttons.

6.4.1 GeoServer layers preview

TSelect the GeoServer connection to dynamically acquire its internal layers and perform corresponding processing., as shown in Figure 6-26.

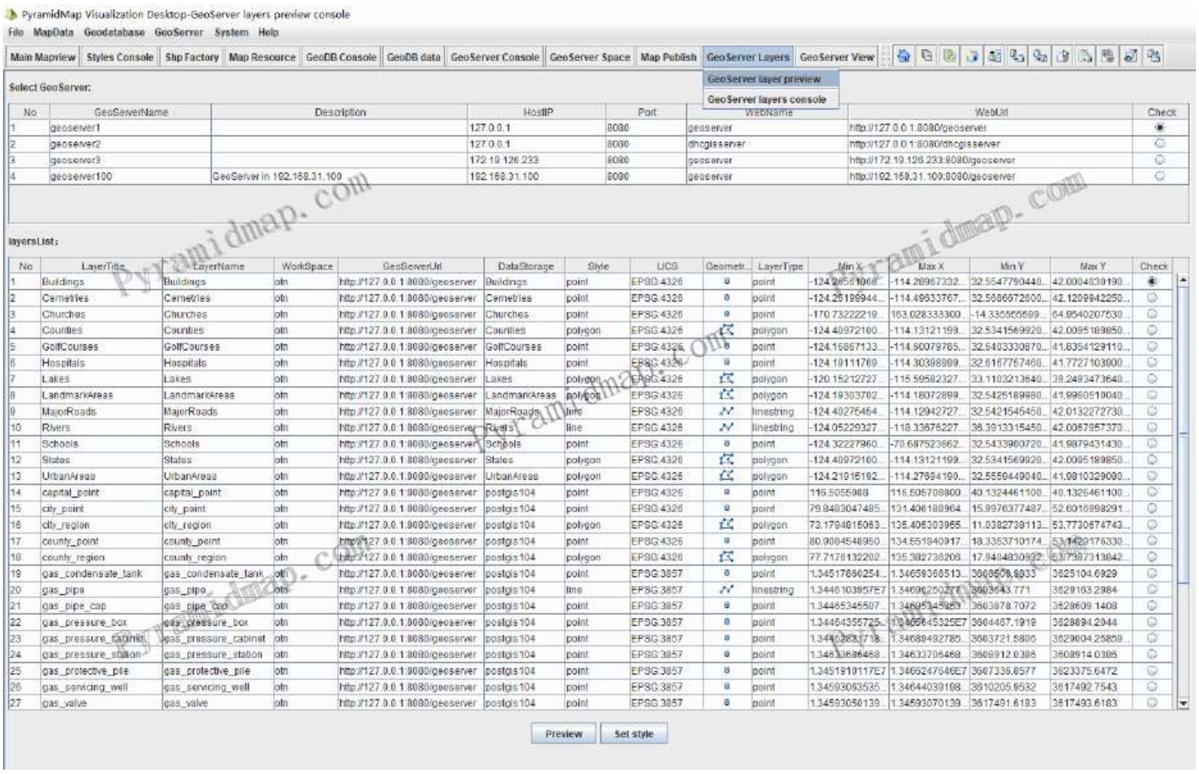


Figure 6-26： GeoServer connections pool and its internal layers pool

Various operations on layer resources are supported, including:

- Preview: preview the selected layer through WMS service mode.
- Set Style: set the sld display symbol matching its geometric type for the selected layer, as shown in Figure 6-27.

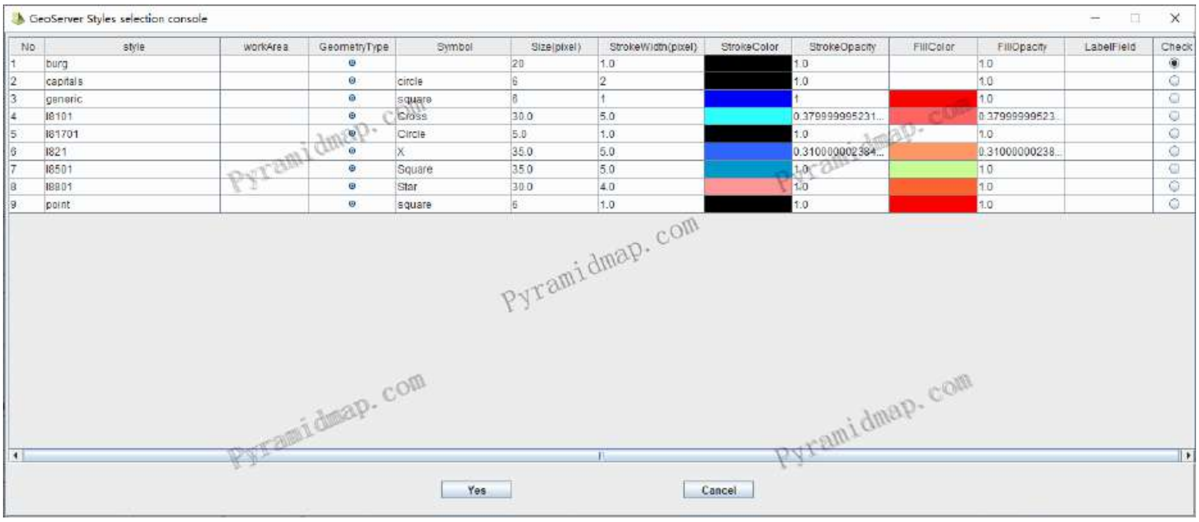


Figure 6-27： Get the internal styles in the GeoServer selected and form styles list pool

Select style in the list and assign it to the selected layer.

In particular, through the map preview and editing options, the selected layer file will be opened in the independent GeoServer map view to preview the layer. The editing function is not supported temporarily, as shown in Figure 6-28.

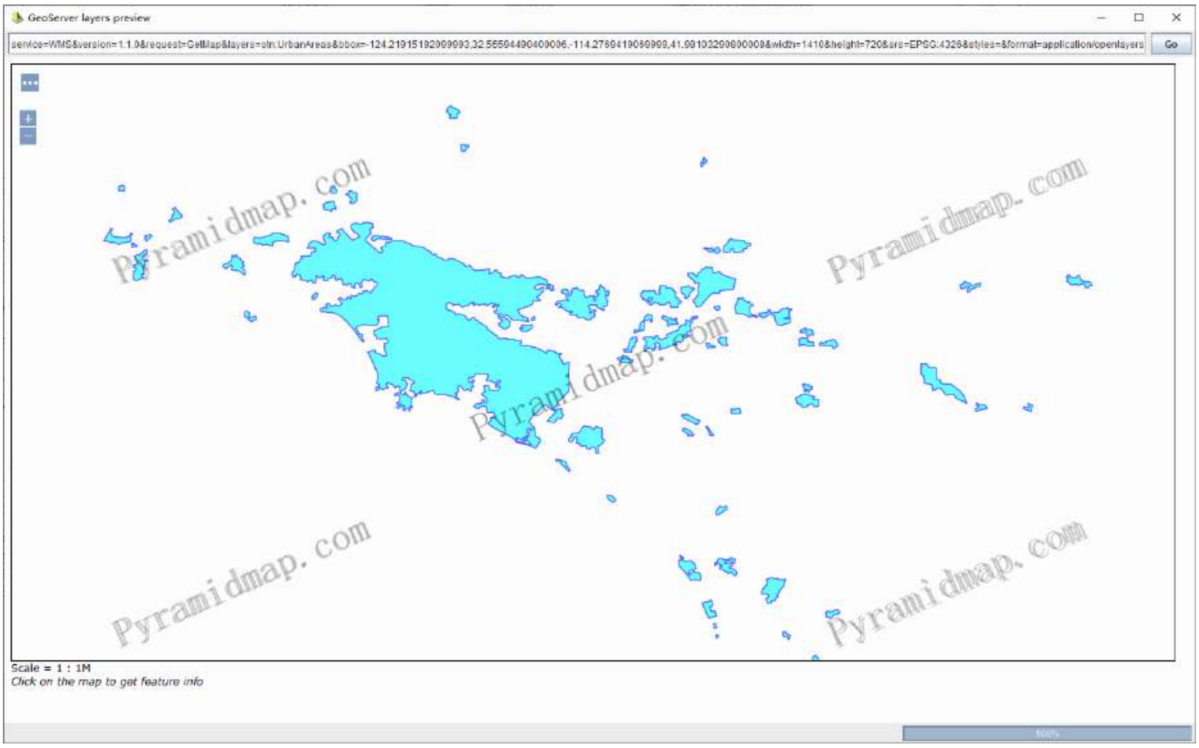


Figure 6-28： The GeoServer vector layer is previewed according to the preset sld style

6.4.2 GeoServer layers exporting

As a reciprocal process with layers publishing, the layers hosted in GeoServer can be exported to different geographic feature data such as Shp, Csv, Kml, GeoJson, etc. The operation interface and flow are shown in Figure 6-29.

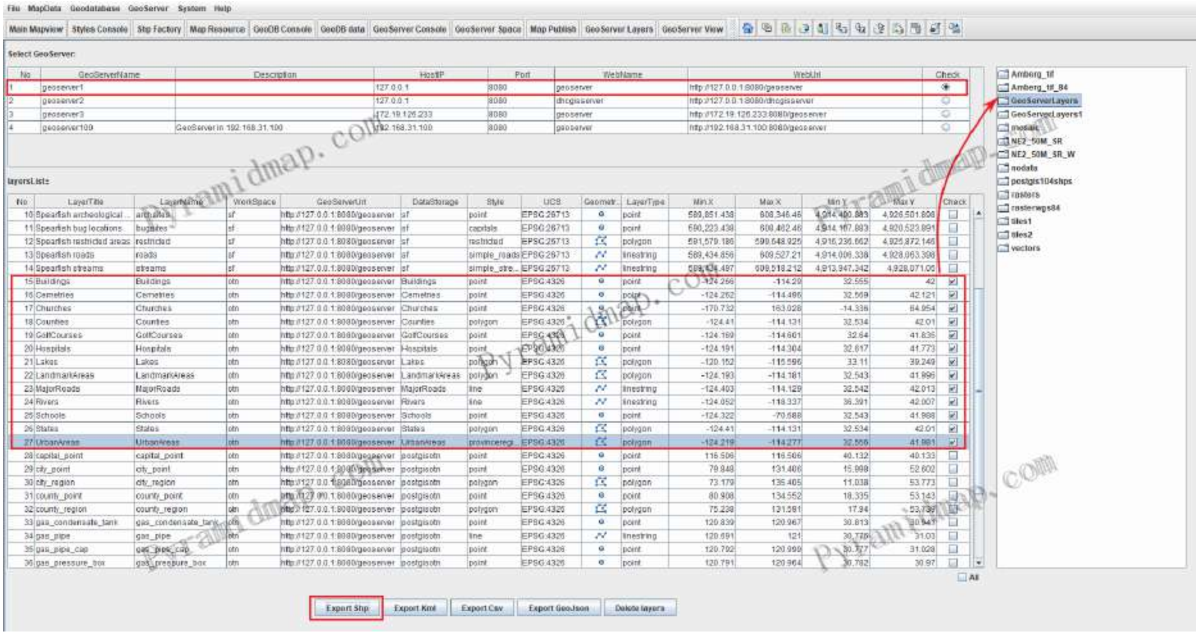


Figure 6-29： GeoServer layer export selection and specified export path

In the GeoServer layer resource list, select a layer and export it to the specified path in batch. The export process is displayed with a progress bar, and the flow diagram is shown in Figure 6-30.

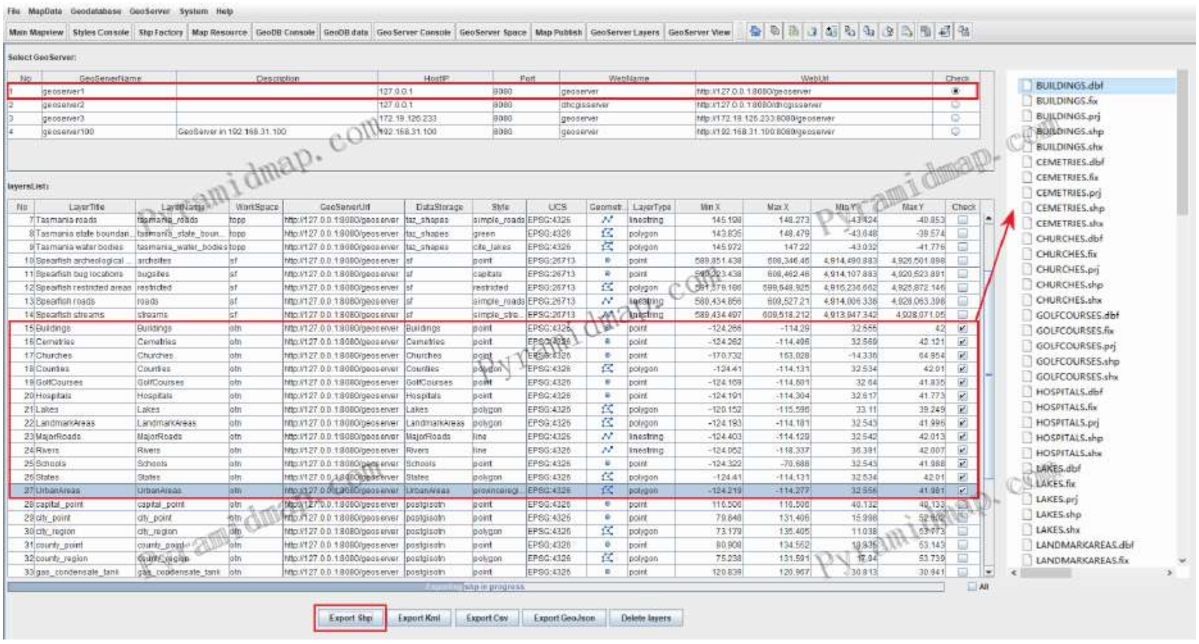


Figure 6-30： GeoServer layer export process diagram

The selected layer is exported to the specified target path, and the export progress is displayed through the progress bar. The exported map is reloaded into the map view in the form of Shp vector file, as shown in Figure 6-31.

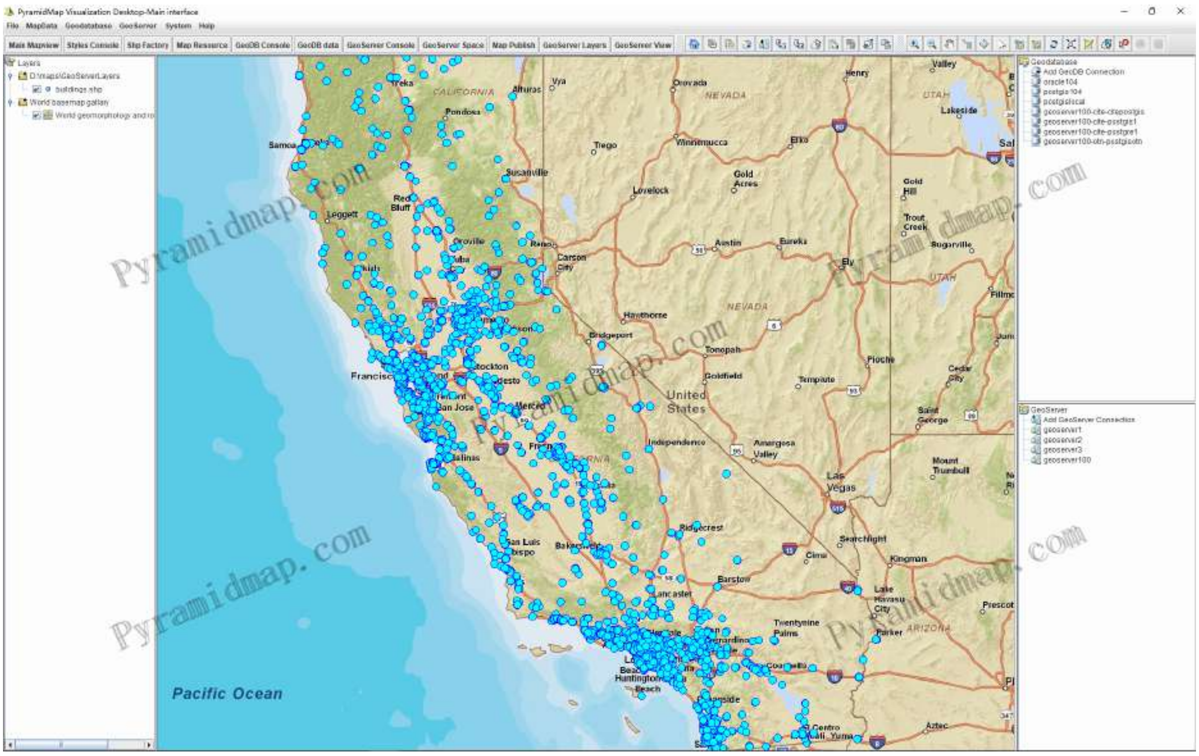


Figure 6-31： The exported Shp vector layer of GeoServer is reloaded to the map view for display

6.5 GeoServer raster layer pool

Select the raster layer in the Figure 6-8 GeoServer map server connection pool and its list, and the selected layer file will be opened in the independent GeoServer map view to preview the layer. The editing function is not supported temporarily, as shown in Figure 6-32.

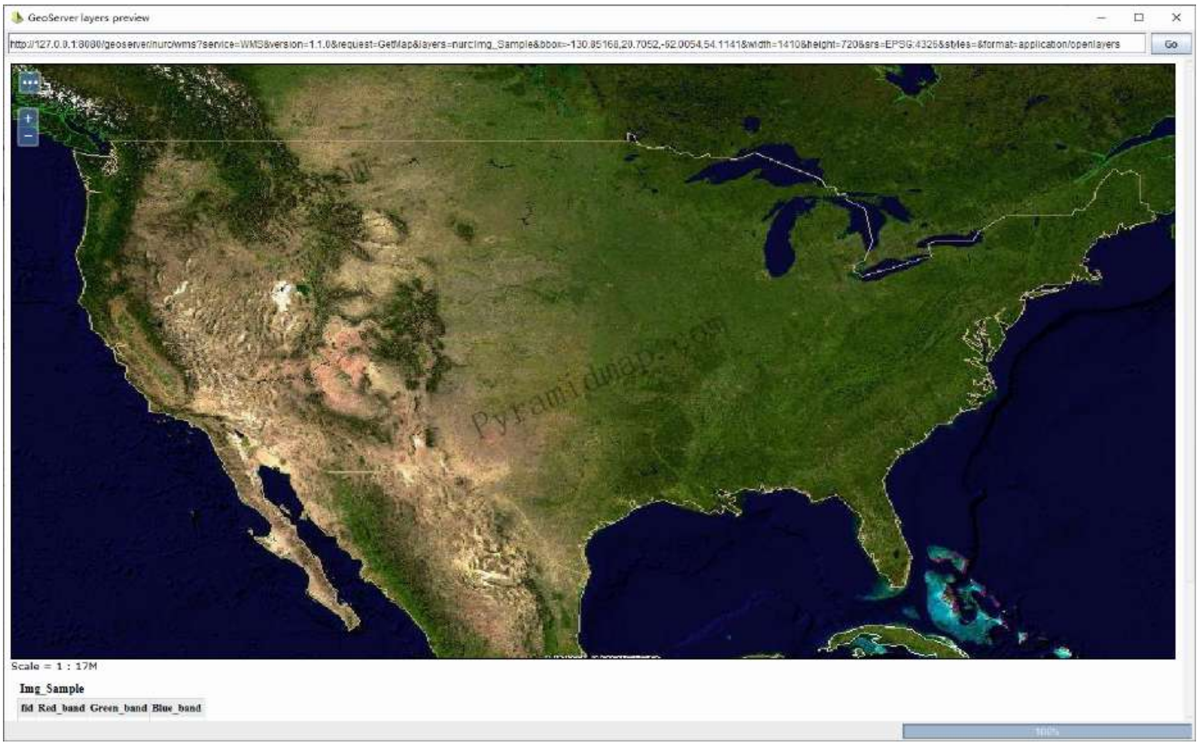


Figure 6-32： The GeoServer raster layer is previewed

6.6 Coordinate System Conversion

PyramidMap supports coordinate system conversion of Shp vector file and raster file layers. Select a layer from the above two types in the resource pool, and select "Coordinate System Conversion" button, as shown in Figure 6-33.

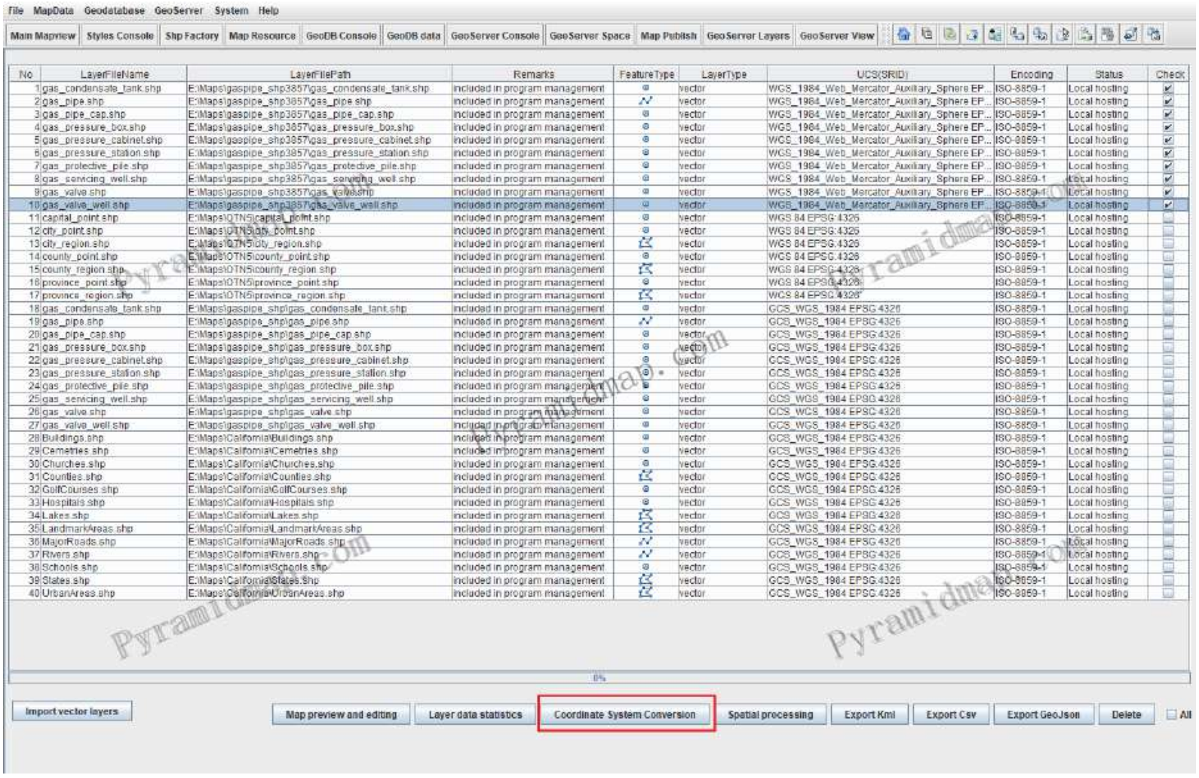


Figure 6-33：Select a layer for coordinate system conversion

The coordinate system conversion interface pops up, as shown in Figure 6-34.

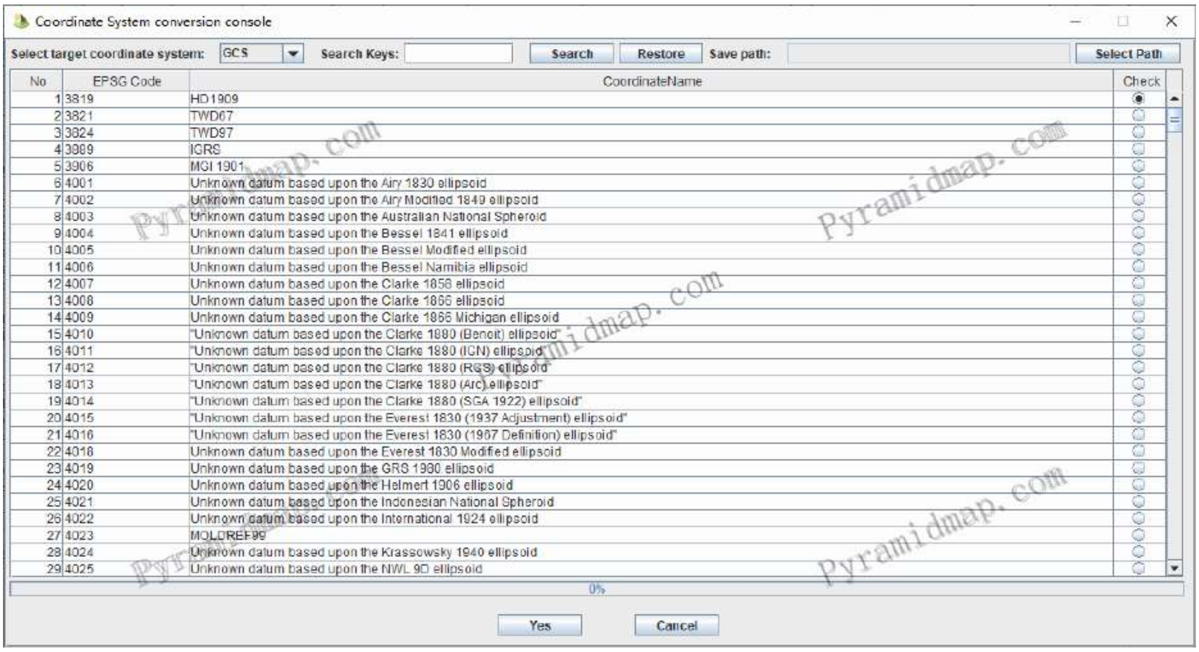


Figure 6-34：Select in coordinate system resource pool for conversion

In the coordinate system conversion interface, select the type of target coordinate system, which is divided into two types: spatial coordinate system and plane coordinate system, as well as many standardized coordinate systems to which it belongs. It supports global search by name and EPSG code. Taking WGS84 as an example, we can perform global keyword search according to coordinate system code 4326 to accurately obtain the target coordinate system we want, as shown in Figure 6-35.

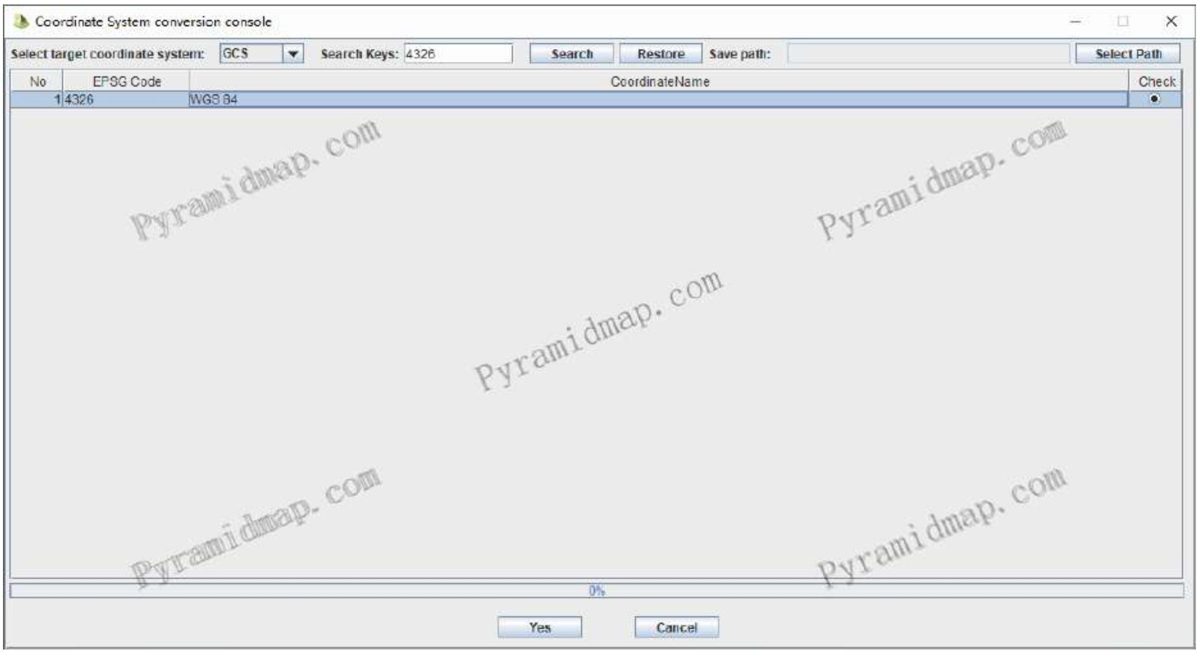


Figure 6-35：Select in coordinate system with global keyword searching according to coordinate system code

Select the target storage path and click "OK" to perform coordinate system conversion. The selected layers will be converted and saved to the target path according to the specified coordinate system. The progress bar displays the conversion progress, as shown in Figure 6-36.

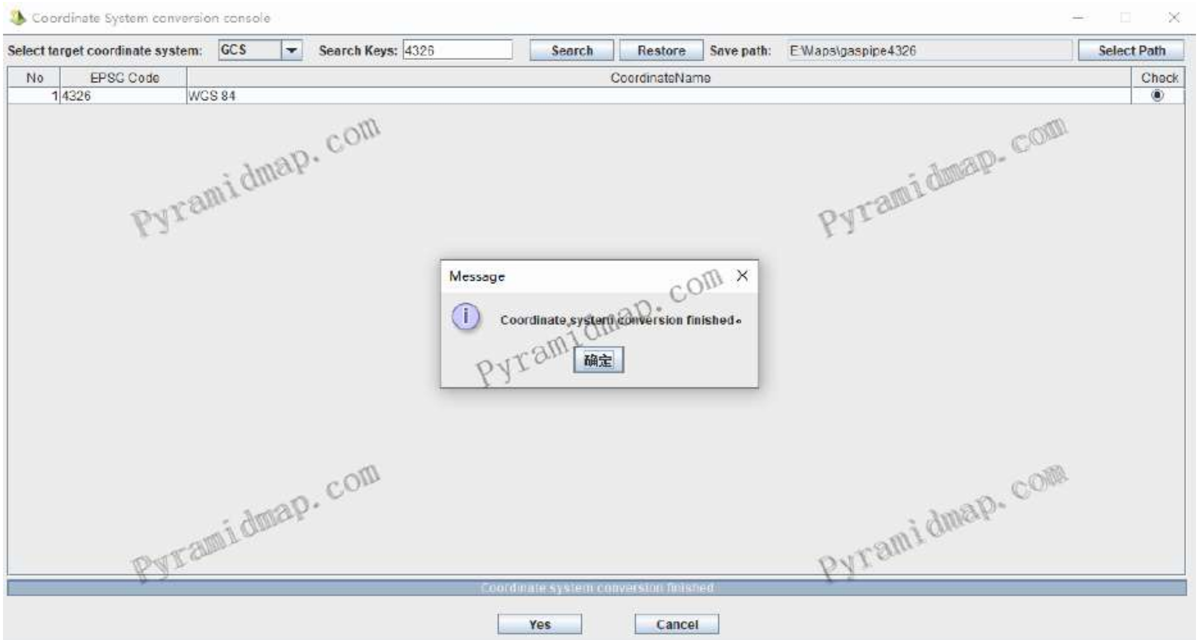


Figure 6-36：Converts the selected layer to the specified coordinate system under the target path

6.7 Data conversion

PyramidMap supports the mutual conversion between multi-source heterogeneous data and Shp layers, giving map application systems the ability to diversify data sources and convert maps into diversified production data. PyramidMap supports bidirectional conversion between Shp and Csv, Excel, GeoJson, as well as unidirectional conversion from Shp to Kml/Kmz. The functional entrances for converting Csv, Excel, and GeoJson to Shp are shown in Figure 6-37.

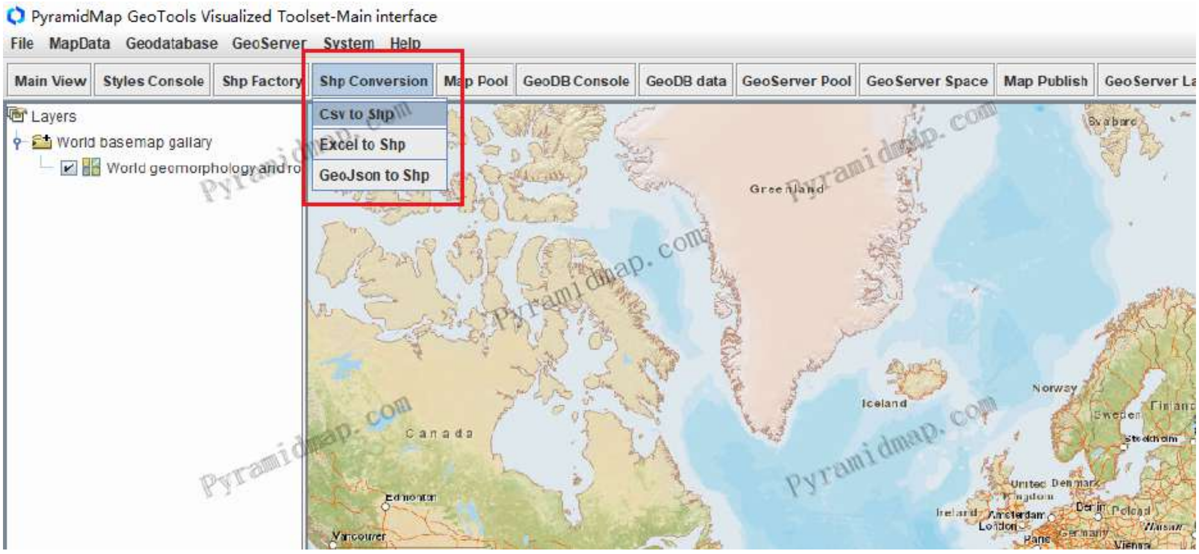


Figure 6-37: Csv, Excel, GeoJson to Shp Function Entry

6.7.1 Csv to Shp

PyramidMap supports the conversion of CSV which meets contractual specifications into Shp graphics, and the data source supports feature types such as Point/MultiPoint, LineString/MultiLineString, Polygon/MultiPolygon, etc. PyramidMap parses CSV according to the agreed specifications and converts it into SHP. The PyramidMap converter reads from CSV format and uses a column named WKT as geometric data. The WKT column contains geometry as well known text. If you are writing data to CSV format, make sure to create geometry and write it to the WKT column. The WKT data format that meets the specifications is shown in Figures 6-38.

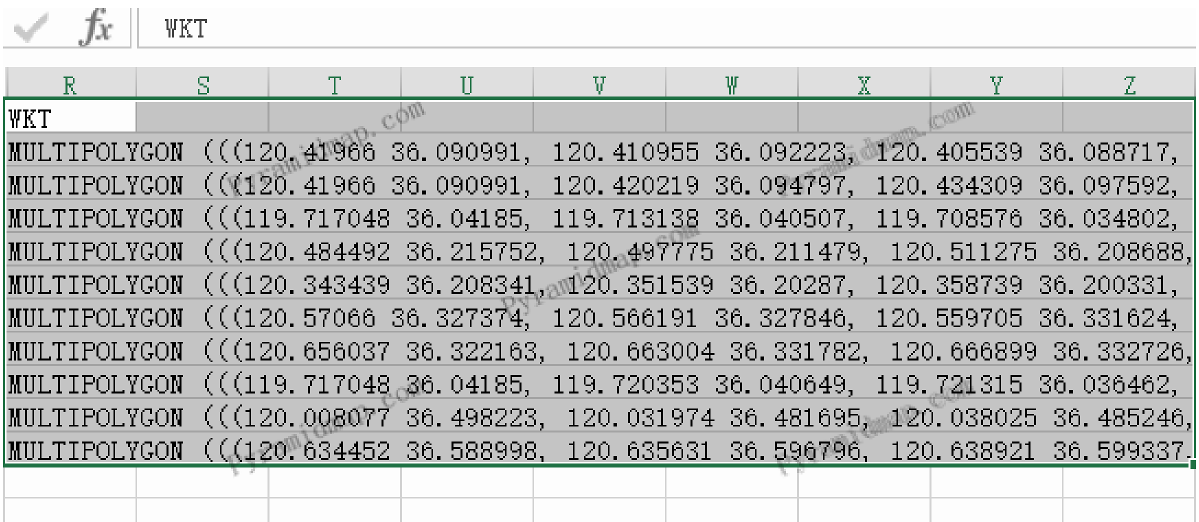


Figure 6-38: WKT data format that meets specifications

Other columns will be written to Shp as attribute data. Specifically, for Point type files, PyramidMap also supports replacing wkt with longitude/latitude fields to represent coordinate values. In other words, for Point type CSV files that represent coordinate data in longitude/latitude (case insensitive) fields instead of WKT format, PyramidMap also supports conversion to Shp. The data format example is shown as below.

```
LATITUDE, LONGITUDE, CITY, NUMBER
46.066667,11.116667,Trento,140
44.9441,-93.0852,St Paul,125
13.752222,100.493889,Bangkok,150
45.420833,-75.69,Ottawa,200
```

```
44.9801,-93.251867,Minneapolis,350
46.519833,6.6335,Lausanne,560
48.428611,-123.365556,Victoria,721
-33.925278,18.423889,Cape Town,550
-33.859972,151.211111,Sydney,436
41.383333,2.183333,Barcelona,914
39.739167,-104.984722,Denver,869
52.95,-1.133333,Nottingham,800
45.52,-122.681944,Portland,840
37.5667,129.681944,Seoul,473
50.733992,7.099814,Bonn,700
```

The Csv to Shp conversion view is shown in Figure 6-39.

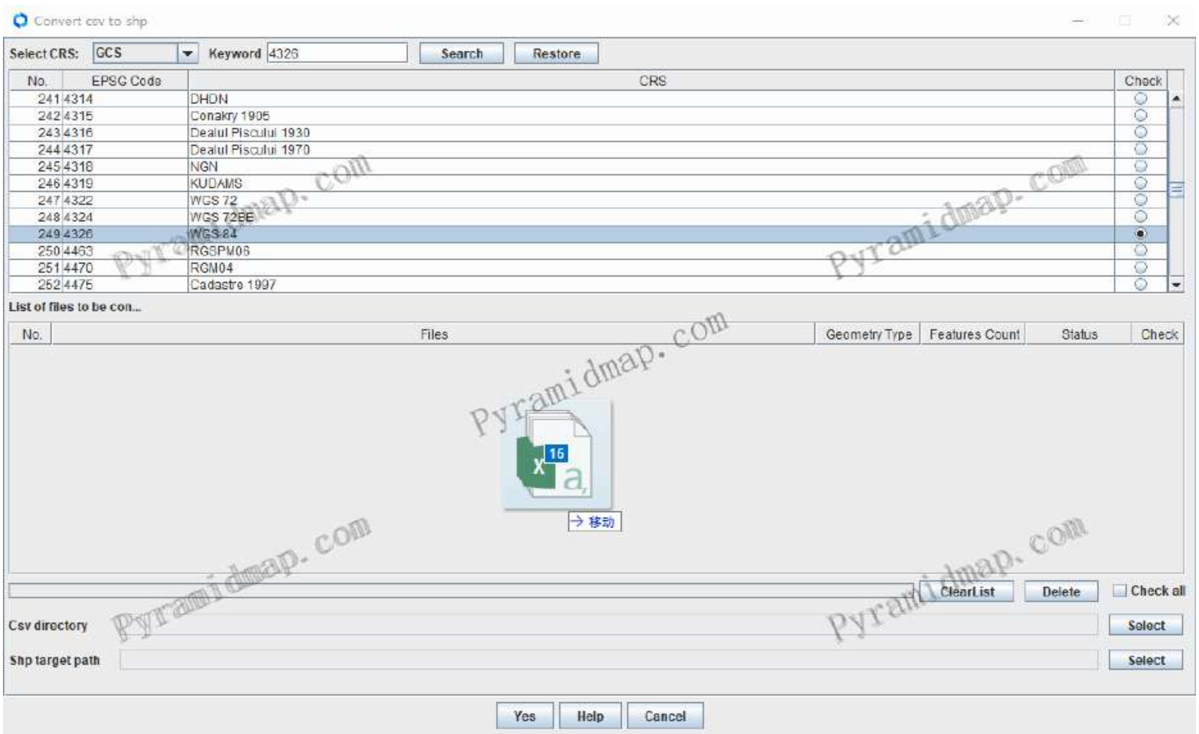


Figure 6-39: Csv to Shp Conversion Interface

PyramidMap supports two methods of selecting CSV data: file selector and drag and drop. Taking the drag and drop mode as an example, it is shown in Figure 6-40.

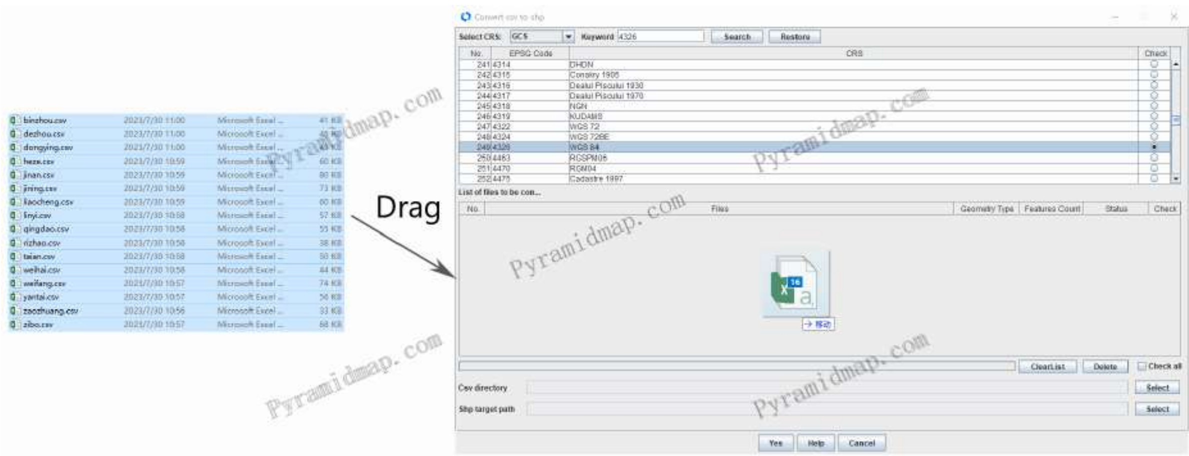


Figure 6-40: Selecting and dragging CSV files to the conversion interface

Form a list of files to be converted, as shown in Figure 6-41.

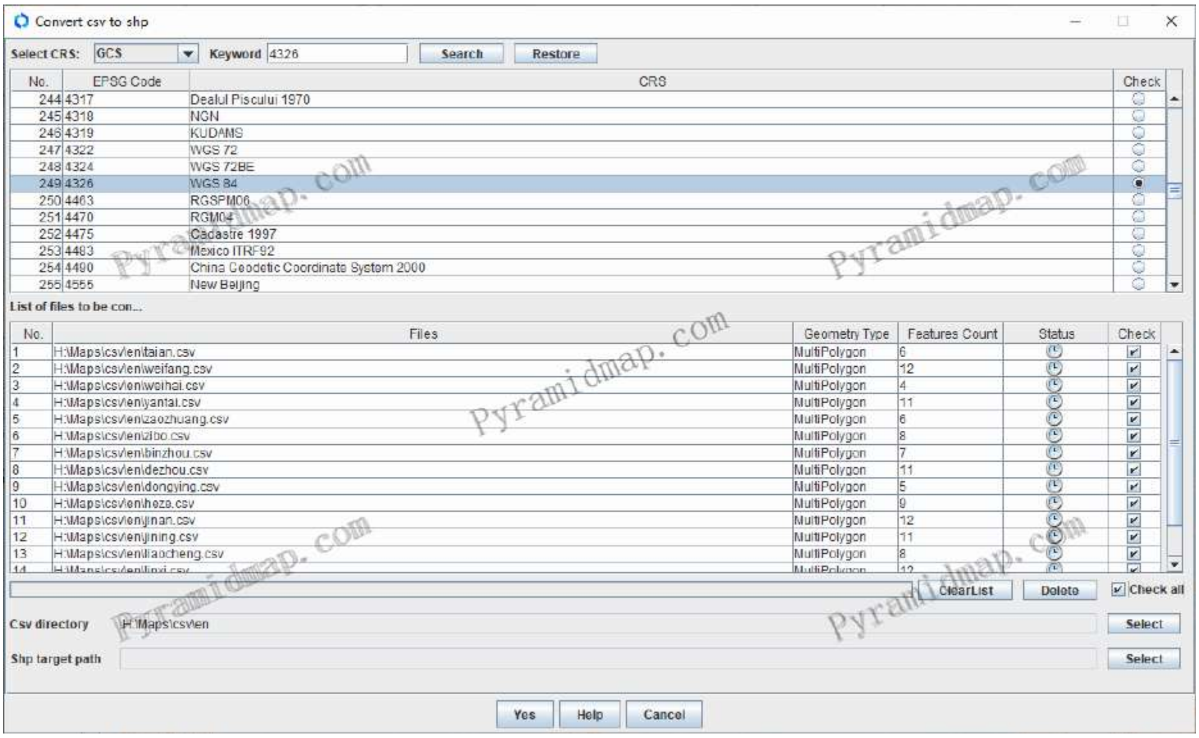


Figure 6-41: Dragging and Dropping CSV Files to Form a Conversion List

Specifically, in general, the wkt data in CSV does not explicitly indicate coordinate system information, so the output file should be specified with a * * Spatial Reference System * * (SRS). PyramidMap supports a wide range of SRSs (approximately 5000), and we are constantly updating them. Specify the * * Spatial Reference System * * (SRS) for the output file, as shown in Figures 6-42.

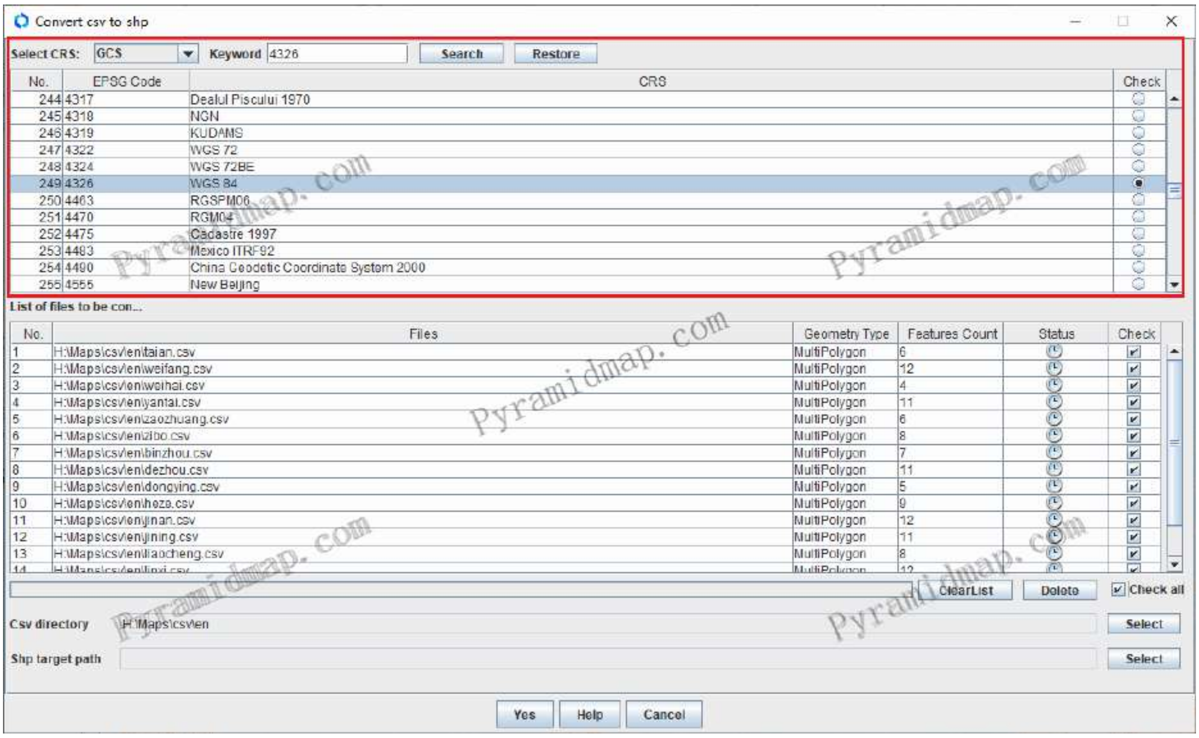


Figure 6-42: Specifying a Spatial Reference System (SRS) for Output Files

The spatial reference system (SRS) is divided into the Geographic coordinate system (GCS) representing three-dimensional space and the projection coordinate system (PCS) representing right angles to the plane. We classify them in these two ways, and conduct Full-text search through keywords. Perform the conversion process, as shown in Figure 6-43.

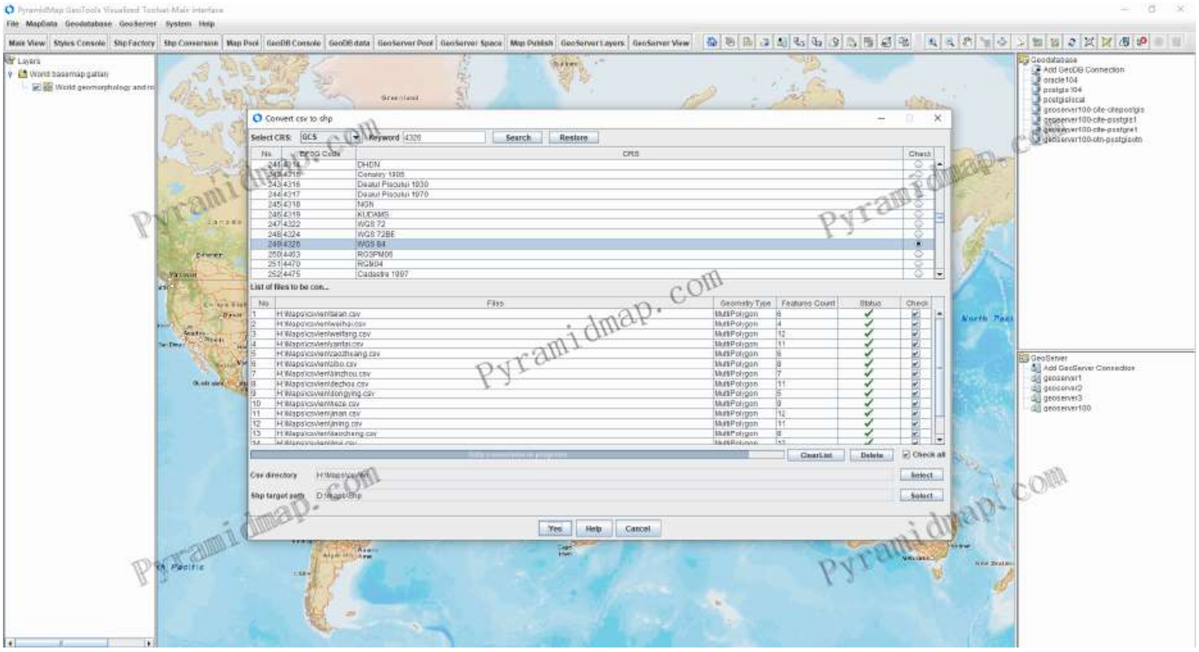


Figure 6-43: Performing conversion on CSV file list

The conversion is complete, as shown in Figure 6-44.

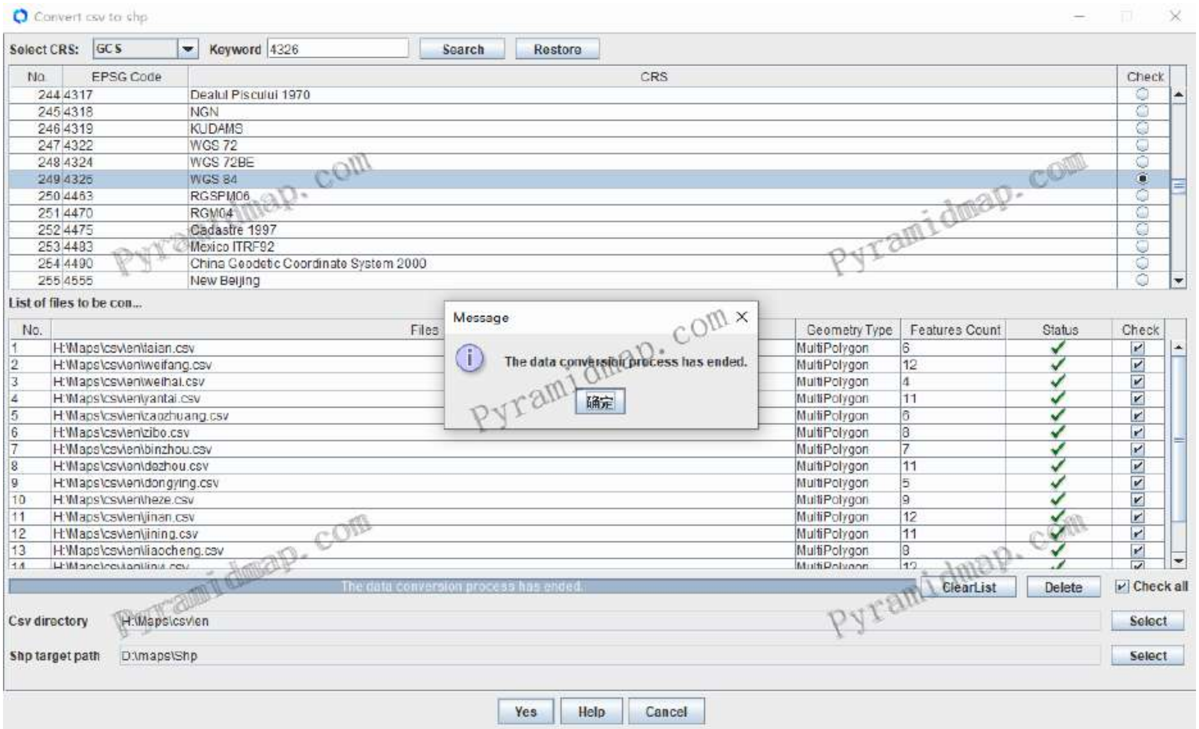


Figure 6-44: CSV file list conversion completed

Form a shp and overlay it onto the map view. The preview effect is shown in Figure 6-45.

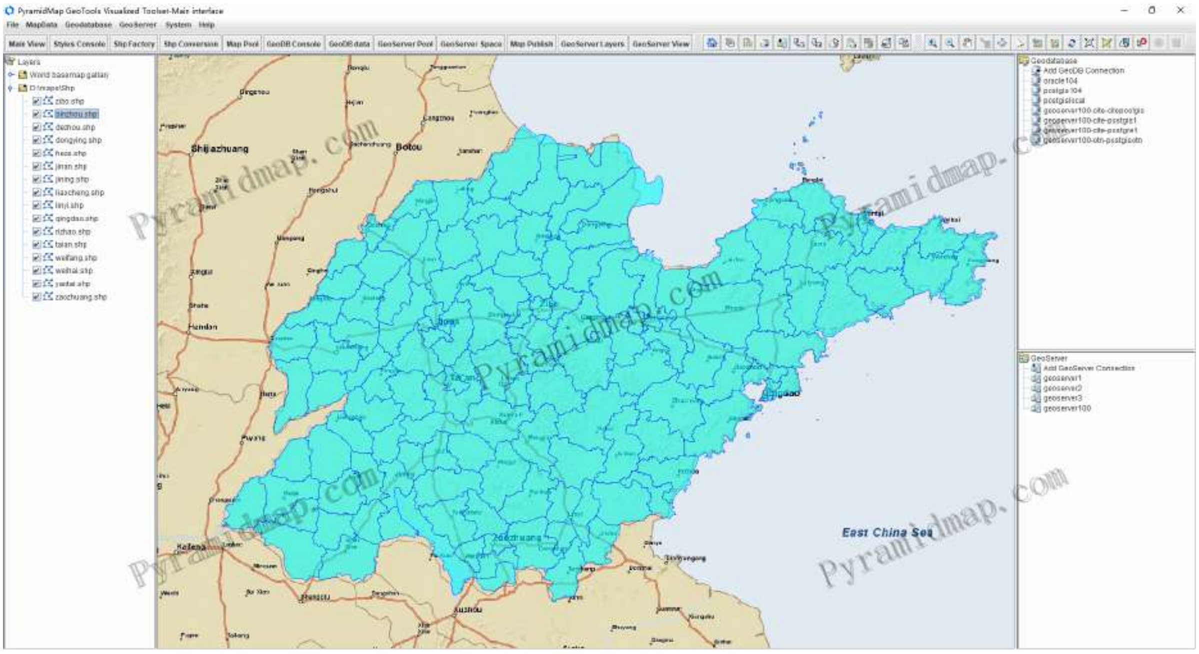
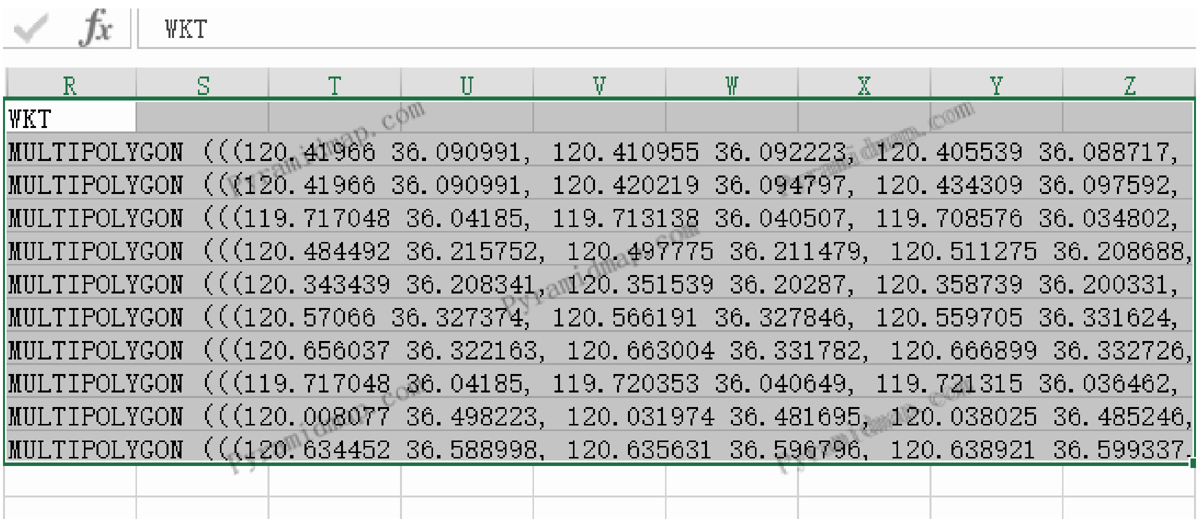


Figure 6-45: Superimposing the formed shp onto the map view, preview effect

6.7.2 Excel to Shp

PyramidMap supports Excel conversion to Shp graphics that meet contractual specifications, and the data source supports feature types such as Point/MultiPoint, LineString/MultiLineString, Polygon/MultiPolygon. PyramidMap parses Excel according to the agreed specifications and converts it into shp. The PyramidMap converter reads from Excel format and uses a column named WKT as geometric data. The WKT column contains geometry as well known text. If you are writing data in Excel format, please ensure that the geometry is created and written into the WKT column. The WKT data format that meets the specifications is shown in Figures 6-46.


fx WKT

R
WKT
MULTIPOLYGON (((120.41966 36.090991, 120.410955 36.092223, 120.405539 36.088717,
MULTIPOLYGON (((120.41966 36.090991, 120.420219 36.094797, 120.434309 36.097592,
MULTIPOLYGON (((119.717048 36.04185, 119.713138 36.040507, 119.708576 36.034802,
MULTIPOLYGON (((120.484492 36.215752, 120.497775 36.211479, 120.511275 36.208688,
MULTIPOLYGON (((120.343439 36.208341, 120.351539 36.20287, 120.358739 36.200331,
MULTIPOLYGON (((120.57066 36.327374, 120.566191 36.327846, 120.559705 36.331624,
MULTIPOLYGON (((120.656037 36.322163, 120.663004 36.331782, 120.666899 36.332726,
MULTIPOLYGON (((119.717048 36.04185, 119.720353 36.040649, 119.721315 36.036462,
MULTIPOLYGON (((120.008077 36.498223, 120.031974 36.481695, 120.038025 36.485246,
MULTIPOLYGON (((120.634452 36.588998, 120.635631 36.596796, 120.638921 36.599337,

Figure 6-46: WKT data format that meets specifications

Other columns will be written to Shp as attribute data. Specifically, similar to the section on converting Csv to Shp in 6.7.1, PyramidMap also supports replacing wkt with longitude/latitude fields to indicate the coordinate values for Point type files. In other words, PyramidMap also supports conversion to Shp for Point type Excel files that represent coordinate data in longitude/latitude (case insensitive) fields instead of WKT format. The Excel to SHP conversion interface is shown in Figure 6-47.

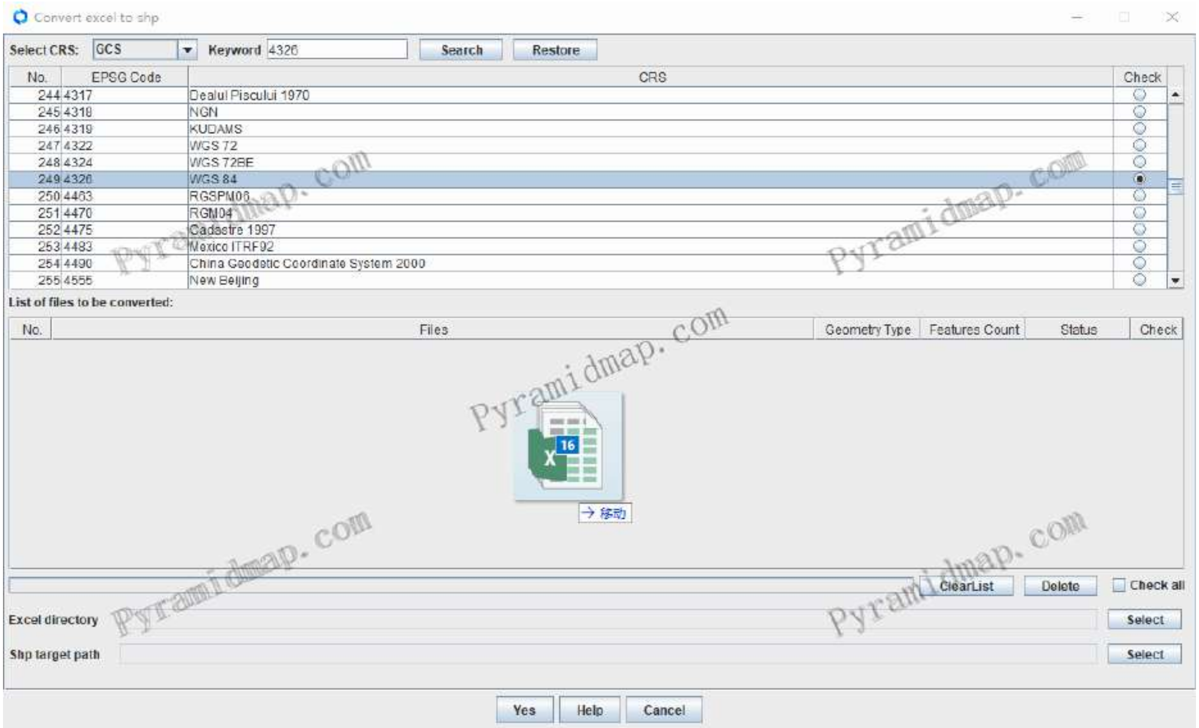


Figure 6-47: Excel to SHP Conversion Interface

PyramidMap supports two methods of selecting Excel data: file selector and drag and drop. Taking the drag and drop mode as an example, it is shown in Figure 6-48.

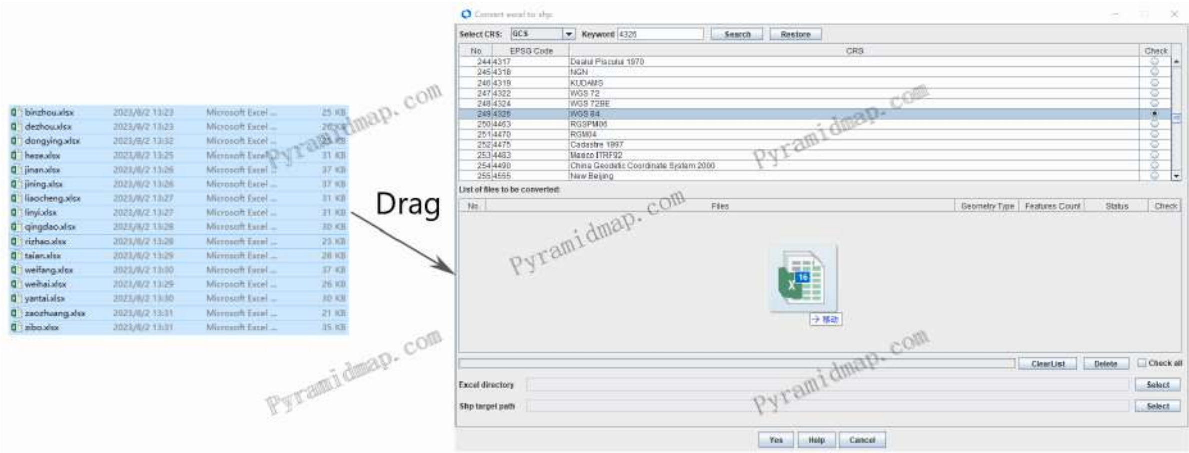


Figure 6-48: Select and drag an Excel file to the conversion interface

Form a list of files to be converted, as shown in Figures 6-49.

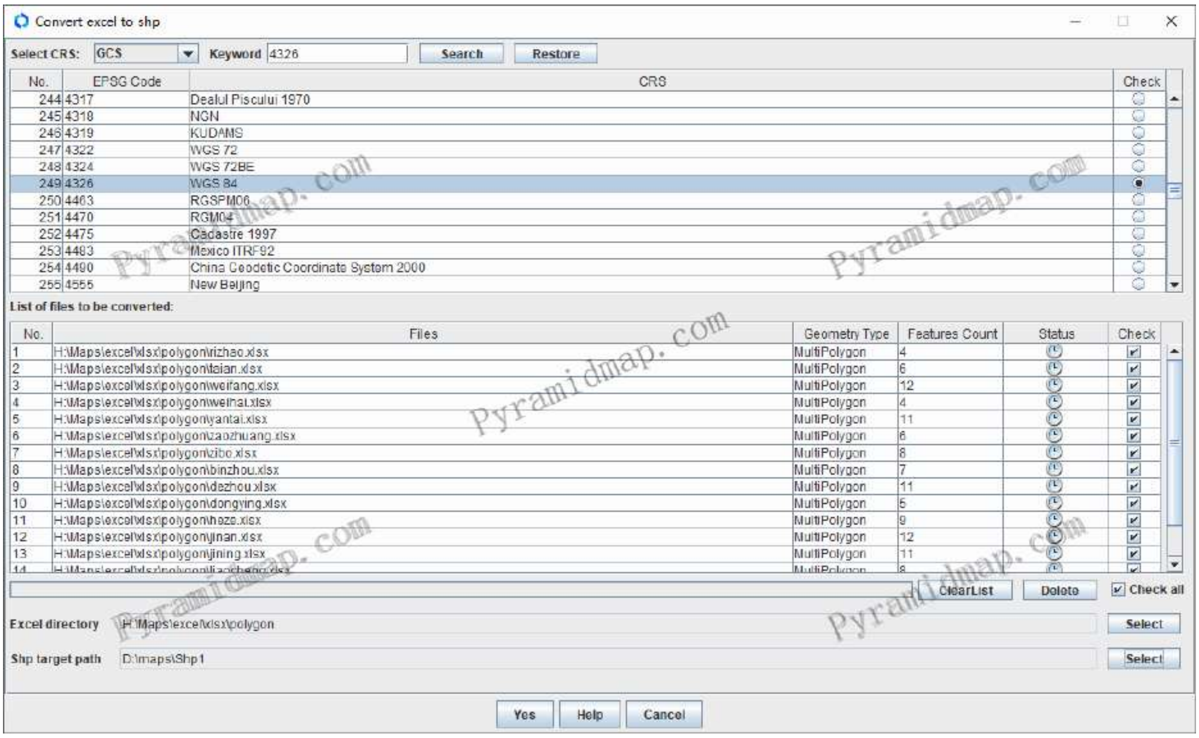


Figure 6-49: Dragging and Dropping Excel Files to Form a Conversion List

Specifically, in general, the wkt data in Excel does not explicitly indicate coordinate system information, so a * * Spatial Reference System * * (SRS) should be specified for the output file. PyramidMap supports a wide range of SRSs (approximately 5000), and we are constantly updating them. Specify the * * Spatial Reference System * * (SRS) for the output file, as shown in Figures 6-50.

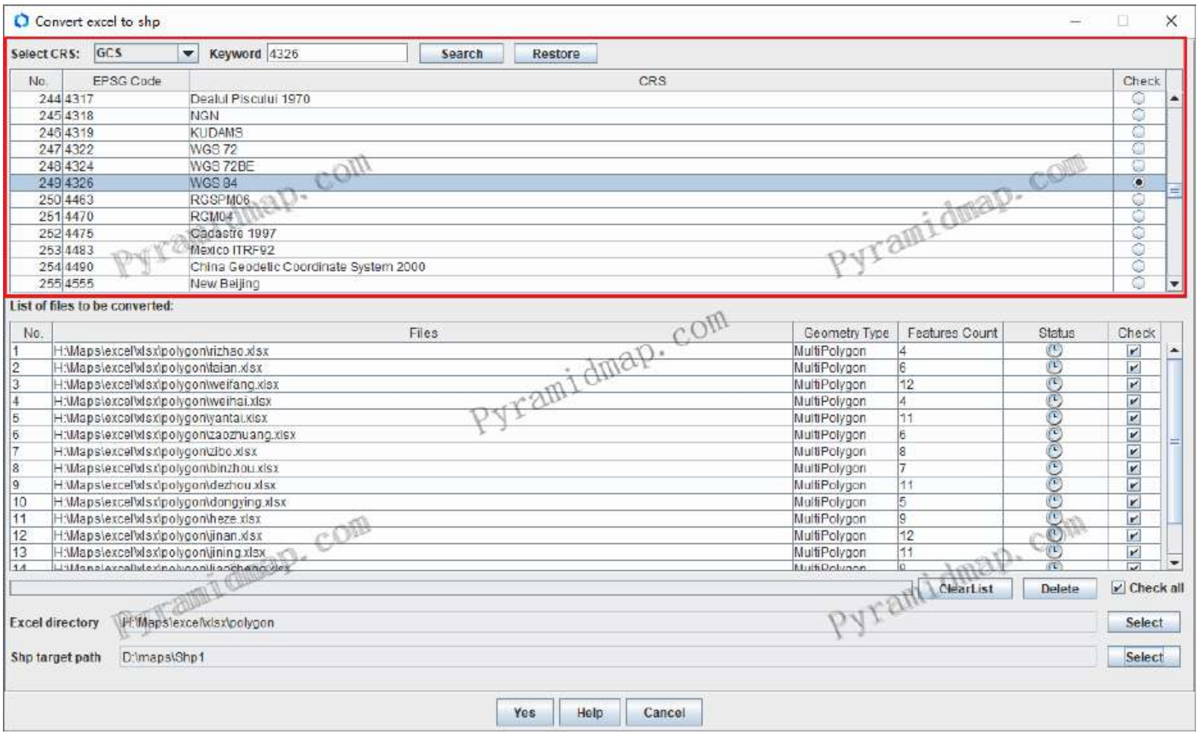


Figure 6-50: Specifying a Spatial Reference System (SRS) for Output Files

The spatial reference system (SRS) is divided into the Geographic coordinate system (GCS) representing three-dimensional space and the projection coordinate system (PCS) representing right angles to the plane. We classify them in these two ways, and conduct Full-text search through keywords. Perform the conversion process, as shown in Figure 6-51.

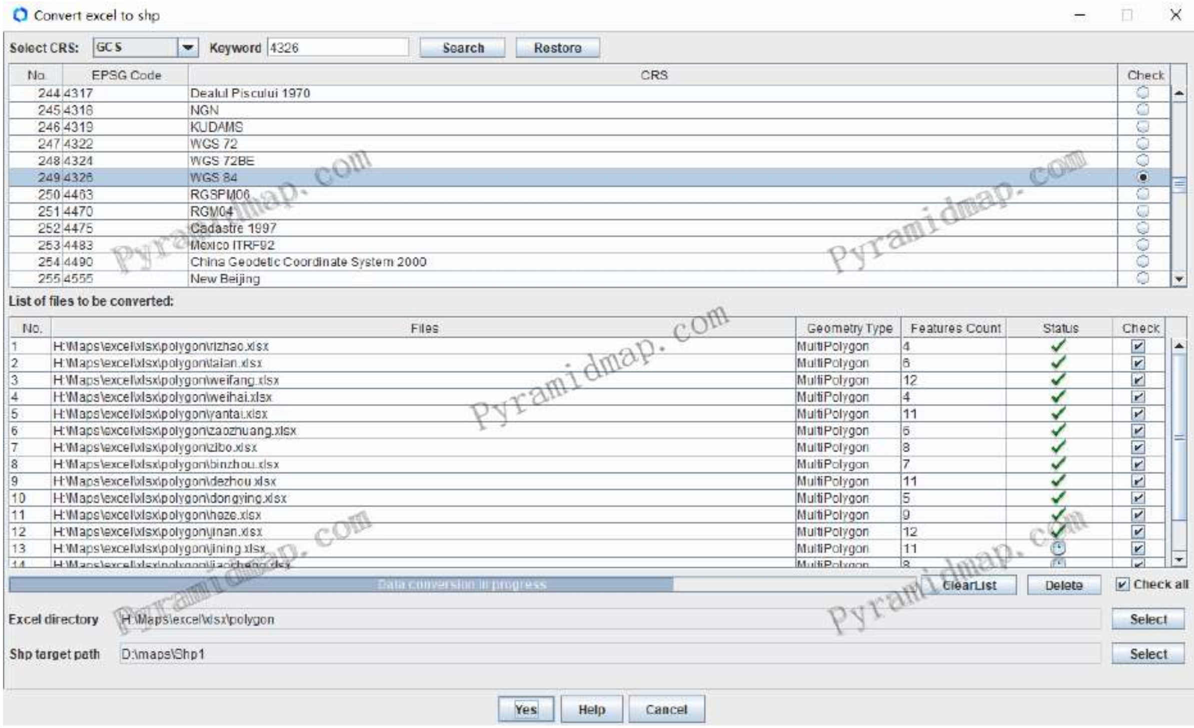


Figure 6-51: Performing Conversion on Excel File List

The conversion is complete, as shown in Figure 6-52.

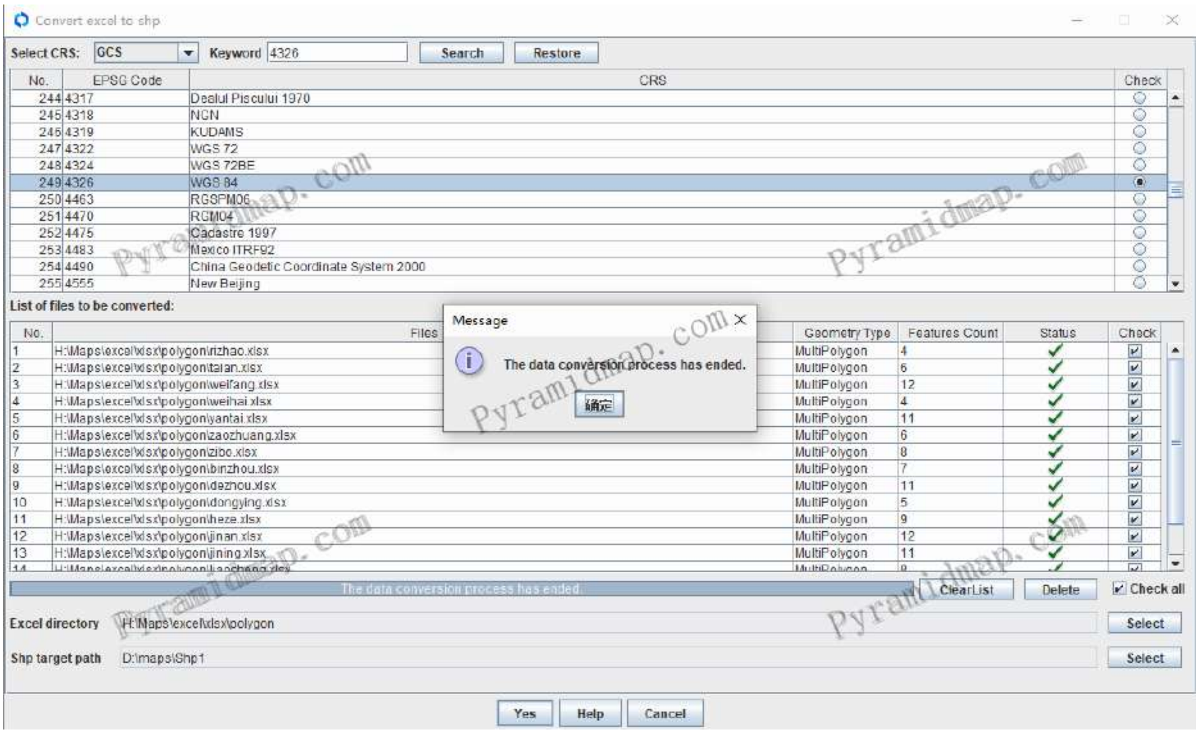


Figure 6-52: Excel file list conversion completed

Form a shp and overlay it onto the map view. The preview effect is shown in Figure 6-53.

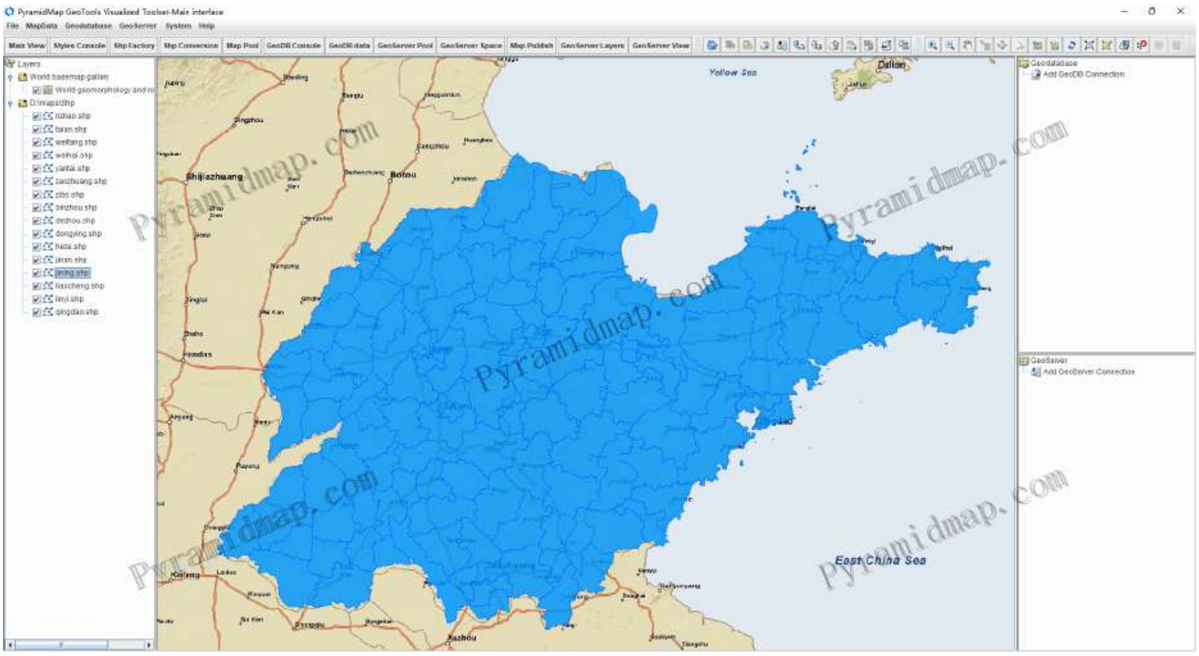


Figure 6-53: Superimposing the formed shp onto the map view, preview effect

6.7.3 GeoJSON to Shp

GeoJSON is an open standard geospatial data exchange format that can represent simple geographic features and their non-spatial attributes. It has become a de facto data standard and is increasingly widely used in data exchange between GIS systems and platforms. GeoJSON is based on JavaScript Object Representation (JSON) and is a format used for encoding various geographic data structures. GeoJSON uses geographical coordinate Frame of reference to define six basic types of geometry: Point, LineString, Polygon, MultiPoint, MultiLineString, and MultiPolygon. GeoJSON geometry can be combined with attribute data to define the features of features and provide data support for conversion to Shp format. The following is an example of a valid GeoJSON file:

```
{
    "type": "FeatureCollection",
    "features": [{
        "type": "Feature",
        "properties": {
            "adcode": 370281,
            "name": "***city",
            "center": [120.006202, 36.285878],
            "centroid": [119.953343, 36.24043],
            "childrenNum": 0,
            "level": "district",
            "acroutes": [100000, 370000, 370200],
            "parent": {
                "adcode": 370200
            }
        },
        "geometry": {
            "type": "MultiPolygon",
            "coordinates": [
                [
                    [
                        [119.717048, 36.041844],
```

```
                        [119.72035, 36.040649],
                        [119.721314, 36.036459],
                        [119.724365, 36.037674],
                        [119.729332, 36.037267],
                        [119.730267, 36.035008],
                        [119.731875, 36.035606],
                        [119.733143, 36.034548],
                        [119.734037, 36.032421],
                        [119.734633, 36.028226],
                        [119.717048, 36.041844]
                    ]
                ]
            ]
        }
    }]
}
```

PyramidMap will parse and convert GeoJSON into shp according to standard specifications, verify the validity of the file, and display error messages when the file is invalid. To treat a GeoJSON file as a valid file in PyramidMap, it must start with named "type" as key and with the value of 'FeatureCollection'. You can convert GeoJSON into Shp for loading in project or layer form in the map. Specifically, if the coordinate system CRS parameter is not specified in the data, the coordinate data will be processed as WGS84:4326 by default.

The GeoJSON to shp conversion interface is shown in Figure 6-54.

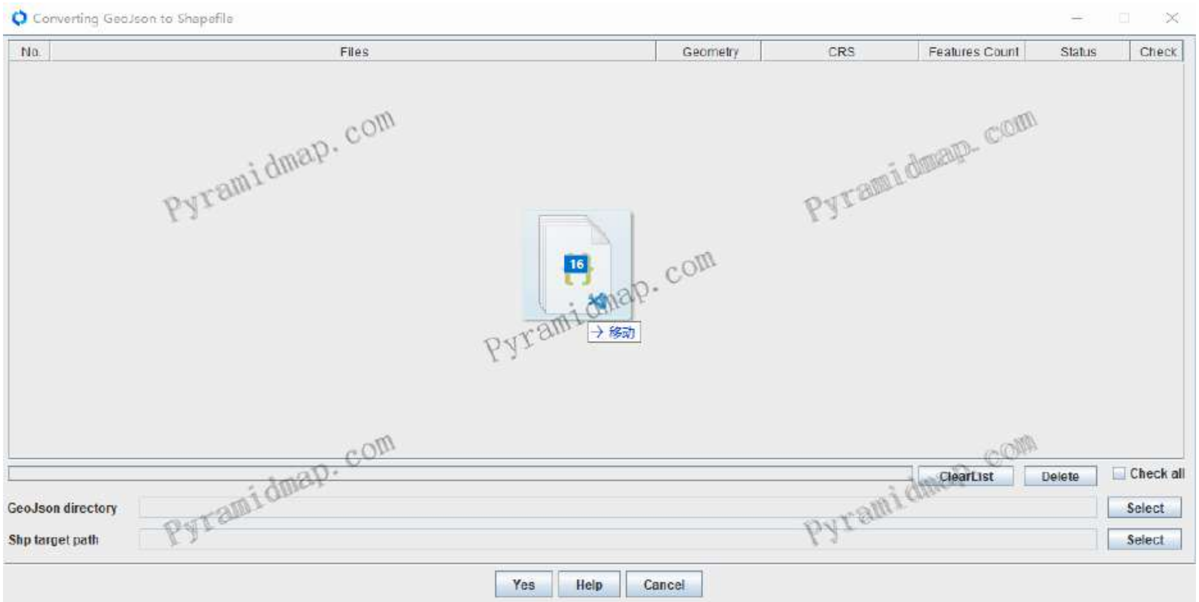


Figure 6-54: GeoJSON to shp conversion interface

PyramidMap supports two methods of selecting GeoJSON data: file selector and drag and drop. Taking the drag and drop mode as an example, it is shown in Figure 6-55.

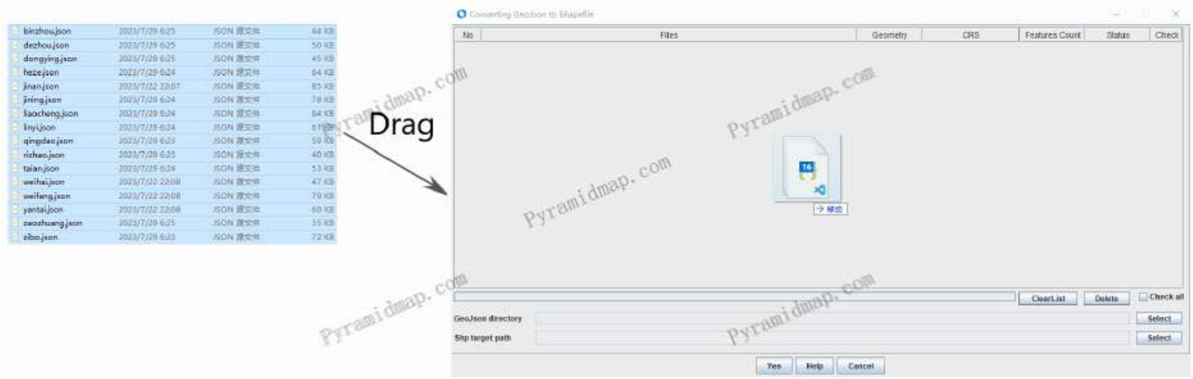


Figure 6-55: Selecting and dragging GeoJSON files to the conversion interface

Form a list of files to be converted, as shown in Figure 6-56.

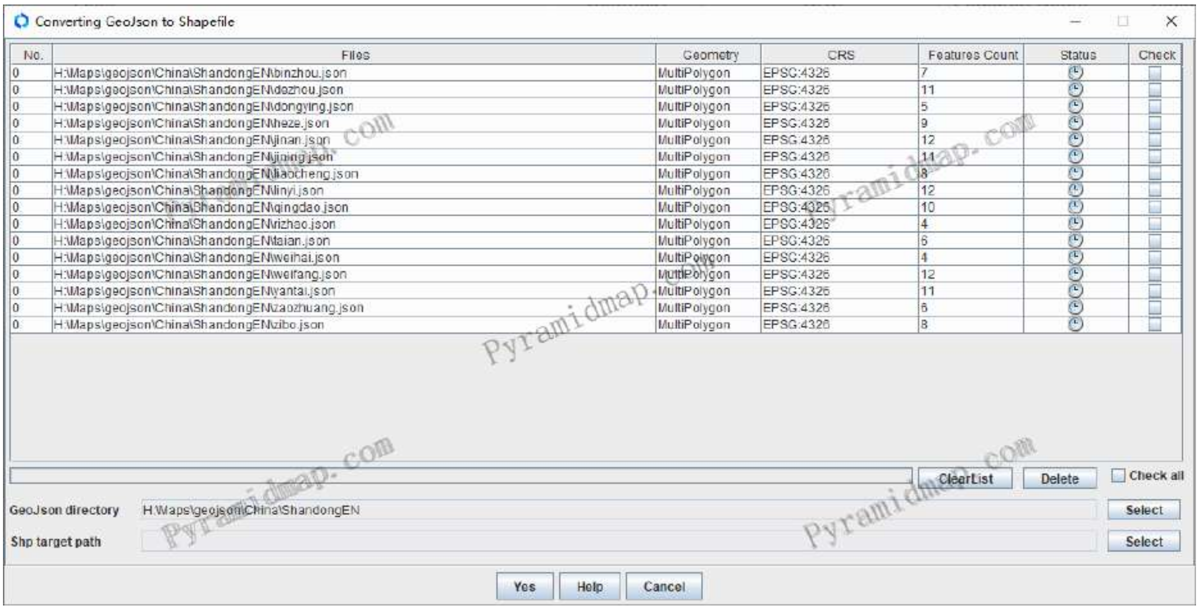


Figure 6-56: Dragging and Dropping GeoJSON Files to Form a Conversion List

GeoJSON uses geographical coordinate Frame of reference to define six basic types of geometry: Point, LineString, Polygon, MultiPoint, MultiLineString, and MultiPolygon. GeoJSON geometry can be combined with attribute data to define the features of features and provide data support for conversion to Shp format. Perform the conversion process, as shown in Figures 6-57.

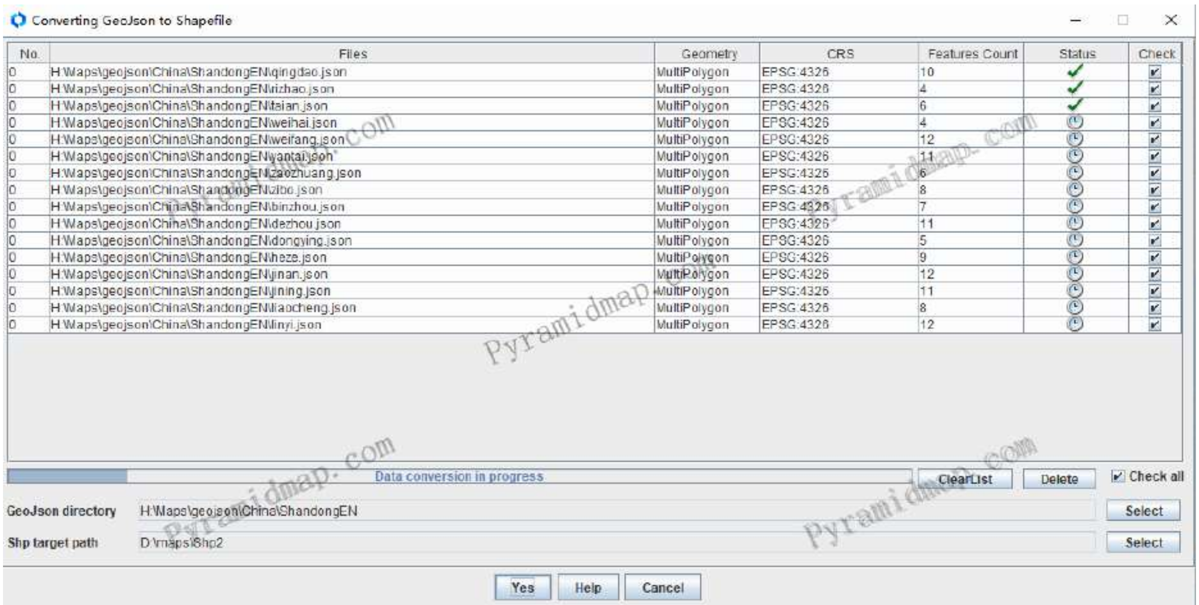


Figure 6-57: Performing conversion on GeoJSON file list

The conversion is complete, as shown in Figure 6-58.

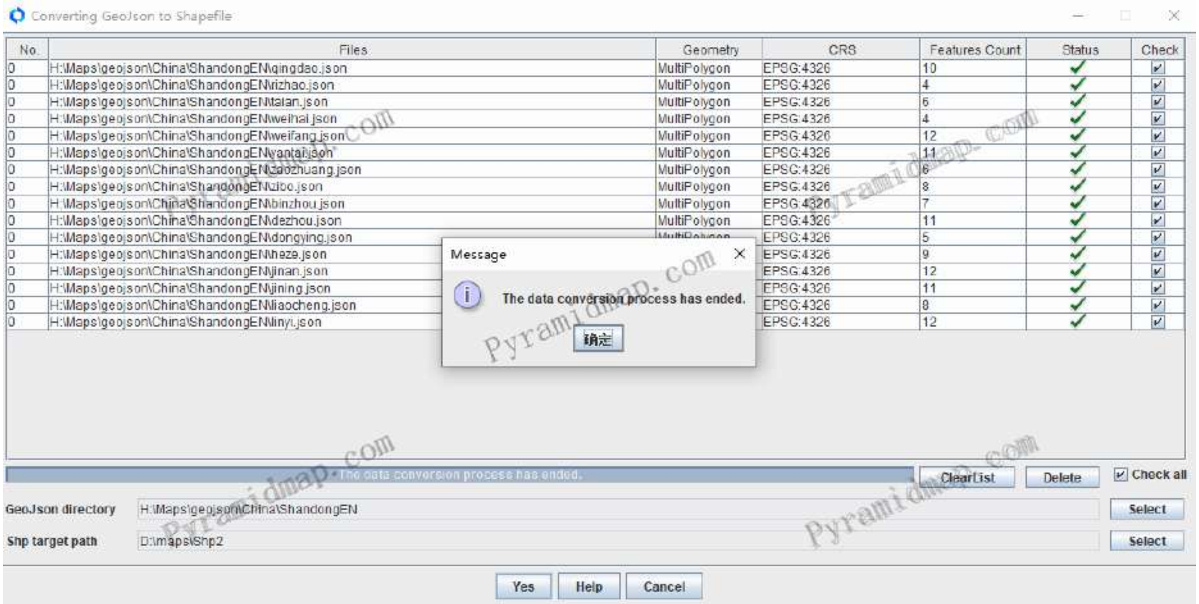


Figure 6-58: GeoJSON file list conversion completed

Form a shp and overlay it onto the map view. The preview effect is shown in Figure 6-59.

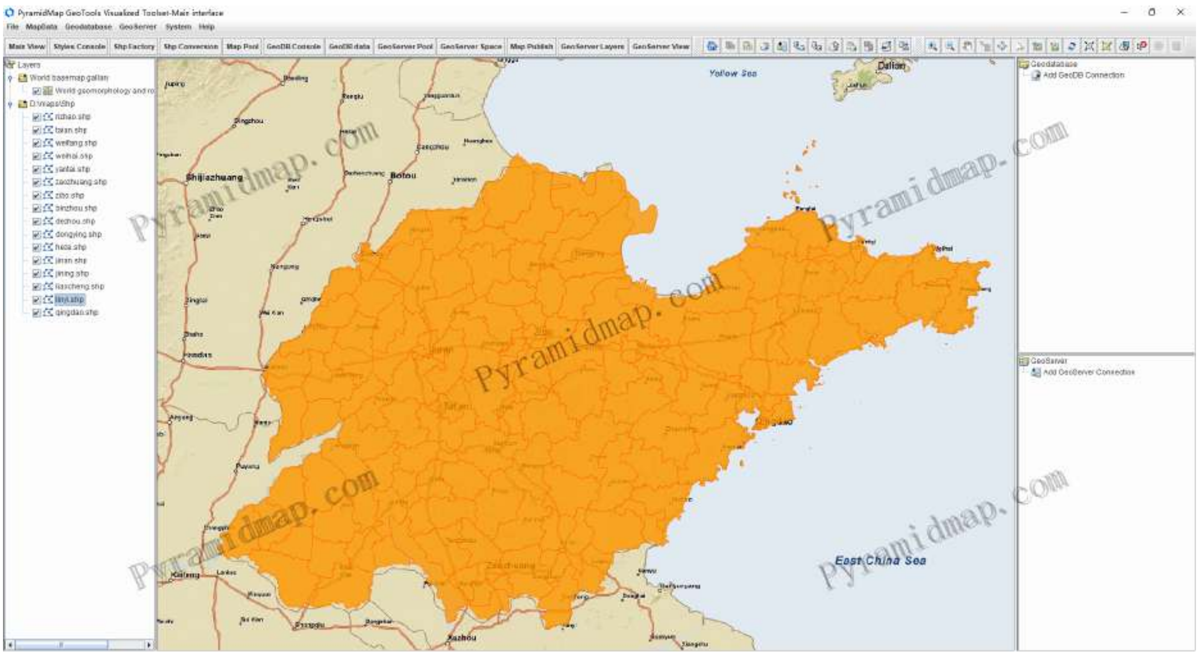


Figure 6-59: Superimposing the formed shp onto the map view, preview effect

6.7.4 Shp to Csv and GeoJSON

PyramidMap supports the conversion of Shp vector layers to structured data types such as Kml, Csv, GeoJson, etc. On the vector layer node of PyramidMap Home screen, the data conversion operation is realized through the right-click menu, as shown in Figure 6-60.

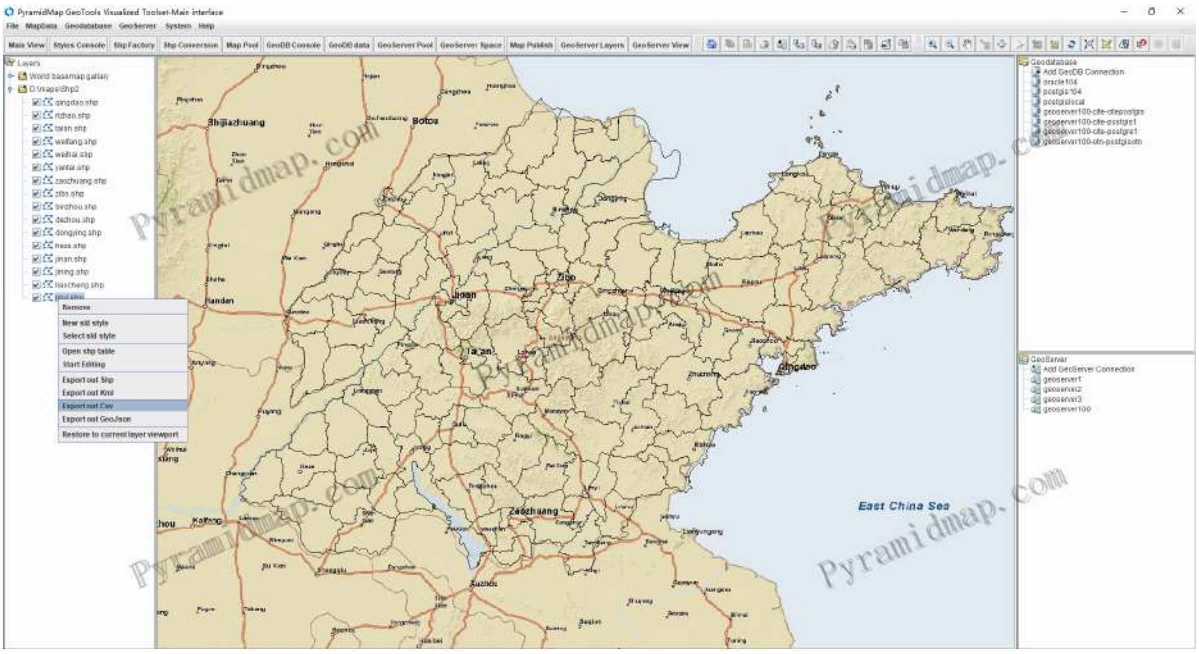


Figure 6-60: PyramidMap vector layer node right-click to converting operation

By using the loaded vector layer nodes, it is easy to export maps to multiple types of structured data such as Shp, Csv, Kml, GeoJSON, etc. The operation of the same industry can also be completed in the vector map resource pool, as shown in Figure 6-61.

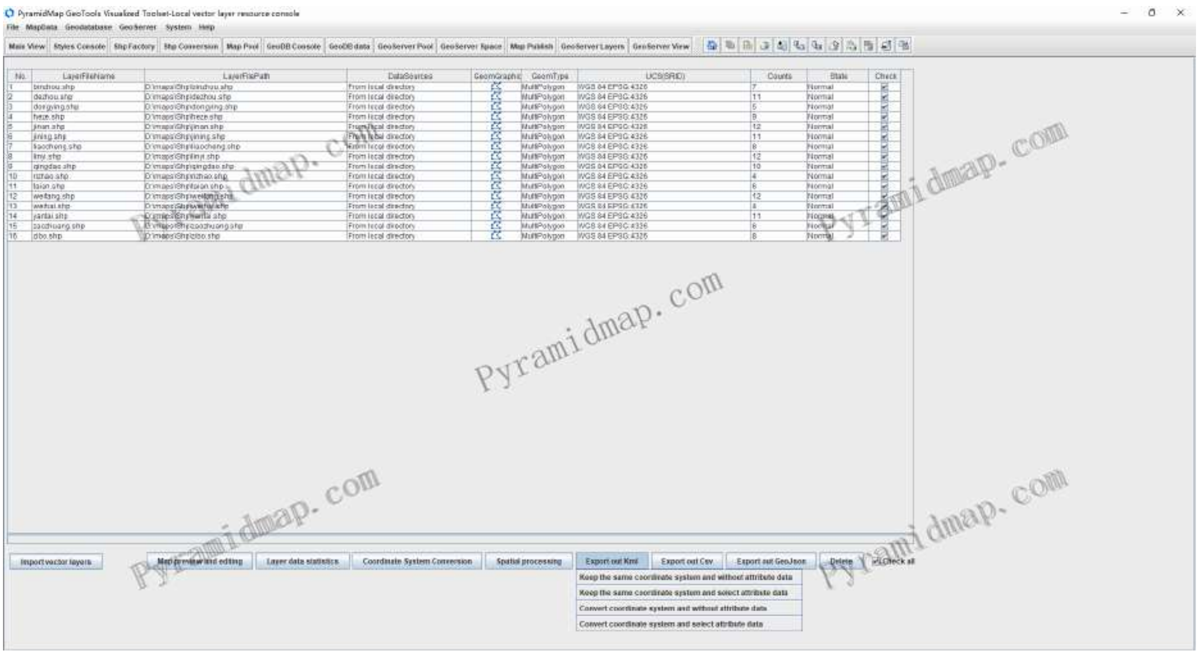


Figure 6-61: Vector layer resource pool and data conversion

Taking Csv data as an example, the transformed data fragments are shown below.

```
LATITUDE, LONGITUDE, CITY, NUMBER
46.066667, 11.116667, Trento, 140
44.9441, -93.0852, St Paul, 125
13.752222, 100.493889, Bangkok, 150
45.420833, -75.69, Ottawa, 200
44.9801, -93.251867, Minneapolis, 350
46.519833, 6.6335, Lausanne, 560
48.428611, -123.365556, Victoria, 721
-33.925278, 18.423889, Cape Town, 550
-33.859972, 151.211111, Sydney, 436
```

```
41.383333, 2.183333, Barcelona, 914
39.739167, -104.984722, Denver, 869
52.95, -1.133333, Nottingham, 800
45.52, -122.681944, Portland, 840
37.5667,129.681944,Seoul,473
50.733992,7.099814,Bonn,700,2016
```

6.7.5 Shp to Kml

Keyhole Markup Language (KML) is an XML based format used to store geographic data and related content, and is an official Open Geospatial Consortium (OGC) standard. The KML format is easy to publish on the Internet and can be viewed through many free applications such as Google Earth and ArcGIS Explorer, making it commonly used for sharing geographic data with non GIS users. KML files have an extension of * *. kml * * or * *. kmz * * (representing compressed KML files).

Same as the method of exporting Csv and GeoJSON, PyramidMap implements data conversion from vector layer to Kml through layer nodes and map resource pool selection list in the Home screen. The layer node conversion of the Home screen is shown in Figure 6-62.

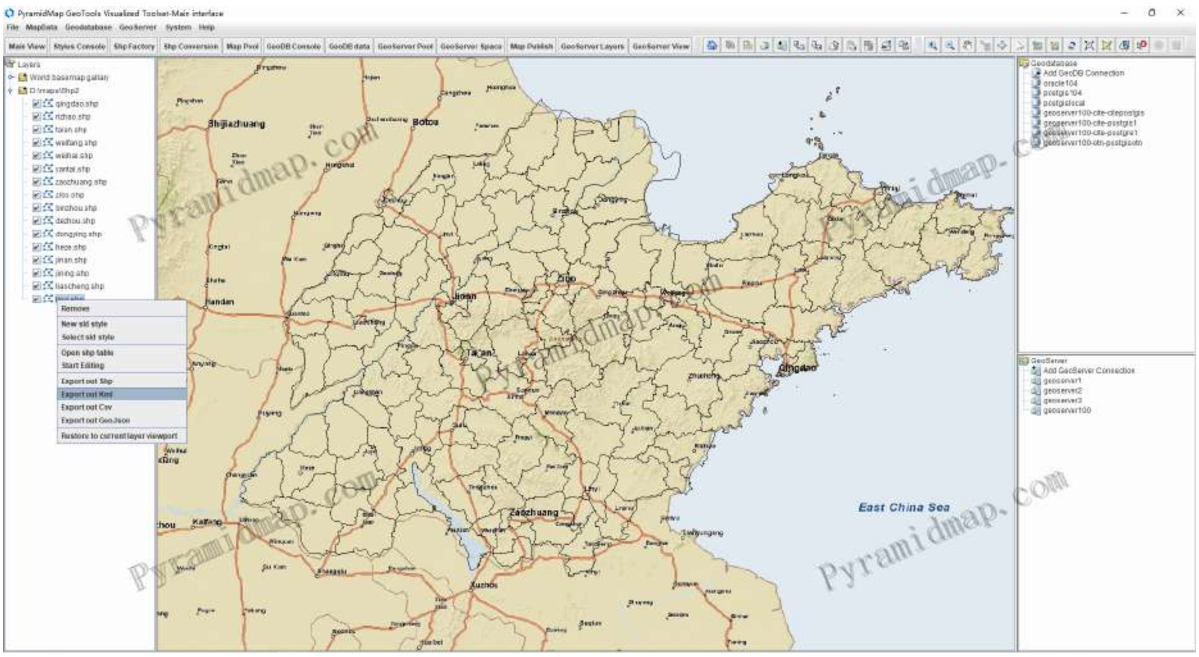


Figure 6-62: PyramidMap vector layer node right-click kml conversion operation

The same conversion operation can also be completed in the vector map resource pool, as shown in Figure 6-63.

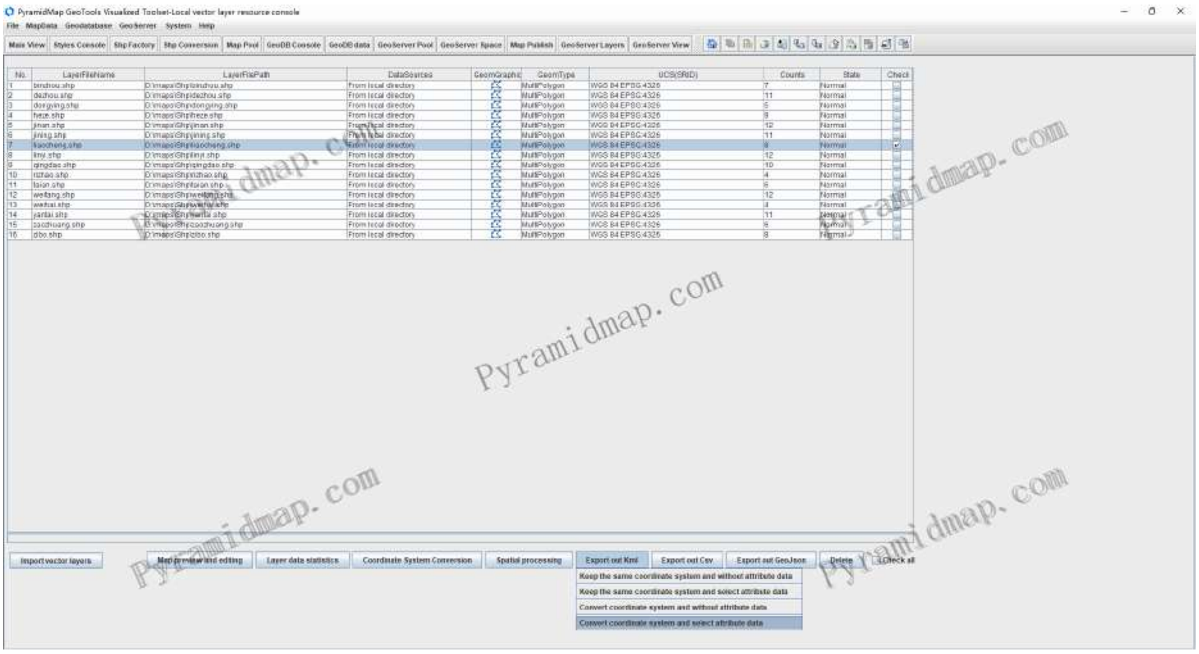


Figure 6-63: Vector layer resource pool and kml data conversion

Select the Shp layer in the list, select the target data format and options, and take the "Convert Coordinate System to Select Attribute Data" option as an example. The execution interface is shown in Figure 6-64.

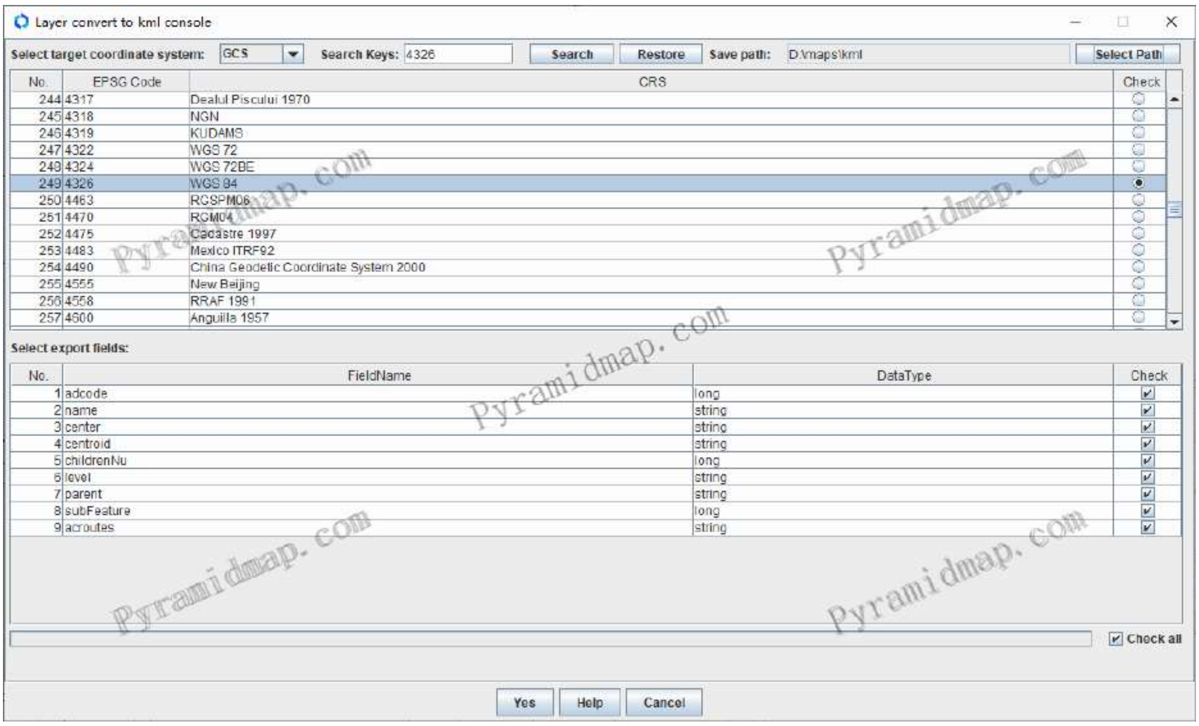


Figure 6-64: Shp to Kml Setting Coordinate System and Attribute Data Options

Select the target coordinate system and the fields to be exported, click "OK", and the selected shp layer will be selected according to the specified coordinate system and fields, and exported as a kml file under the target path. The exported kml data fragment is as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<kml
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:kml="http://earth.google.com/kml/2.1">
    <Document id="featureCollection">
        <Placemark id="fid-79657dd2_189aaf861b2_-7ffd">
```

```
            <name>**city</name>
            <MultiGeometry>
                <Polygon>
                    <outerBoundaryIs>
                        <LinearRing>
                            <coordinates>121.737819,37.128871 121.740542,37.12828
121.743526,37.12536 121.737819,37.128871</coordinates>
                        </LinearRing>
                    </outerBoundaryIs>
                </Polygon>
                <Polygon>
                    <outerBoundaryIs>
                        <LinearRing>
                            <coordinates>121.484626,36.732883
121.490872,36.738411 121.491364,36.742102 121.484626,36.732883</coordinates>
                        </LinearRing>
                    </outerBoundaryIs>
                </Polygon>
            </MultiGeometry>
        </Placemark>
    </Document>
</kml>
```

6.8 Map Tile

Map tiling is a method of dividing an entire map into grids with row and column numbers according to certain rules. Map tiling is the most effective and primary technical means to improve the loading speed of layers, especially for large-scale remote sensing images. Major GIS manufacturers have done so.
For vector maps, tiling can only draw the corresponding elements within the grid according to the required display range, thereby reducing rendering pressure. For raster maps, it can reduce the amount of data requested (or read) at once, thus displaying the map within the area more efficiently.

How to effectively tiling maps according to standard specifications and make the map tiles dataset suitable for different map engines to display correctly on the web is a technical challenge. In this chapter, we will demonstrate the implementation solutions of mainstream slicing technologies such as TMS, XYZ, MVT, MBTiles, etc. through specific examples.

6.8.1 Tile pyramid model

The concept of map tiling was first proposed and applied by Google Maps, which uses a specific cutting method to slice the world map grid image using Web Mercator projection coordinates. Web Mercator projection is also a scheme created by Google, which simulates the real elliptical Earth as an isosphere for plane projection, making it convenient for computers to calculate, Subsequently, major mainstream WebGIS and internet map applications adopted a method based on this projection coordinate system for tiling.
The map is tiled according to the pyramid model, and the tile map pyramid model is a multi-resolution hierarchical model that stores and displays at different resolutions according to user needs under a unified spatial reference, forming a pyramid structure with resolution ranging from coarse to fine and data volume ranging from small to large. The geographical range represented remains unchanged. The lower the pyramid, the more detailed the map information it represents, and the larger of the scale. The schematic diagram of the principle is shown in Figure 6-65.

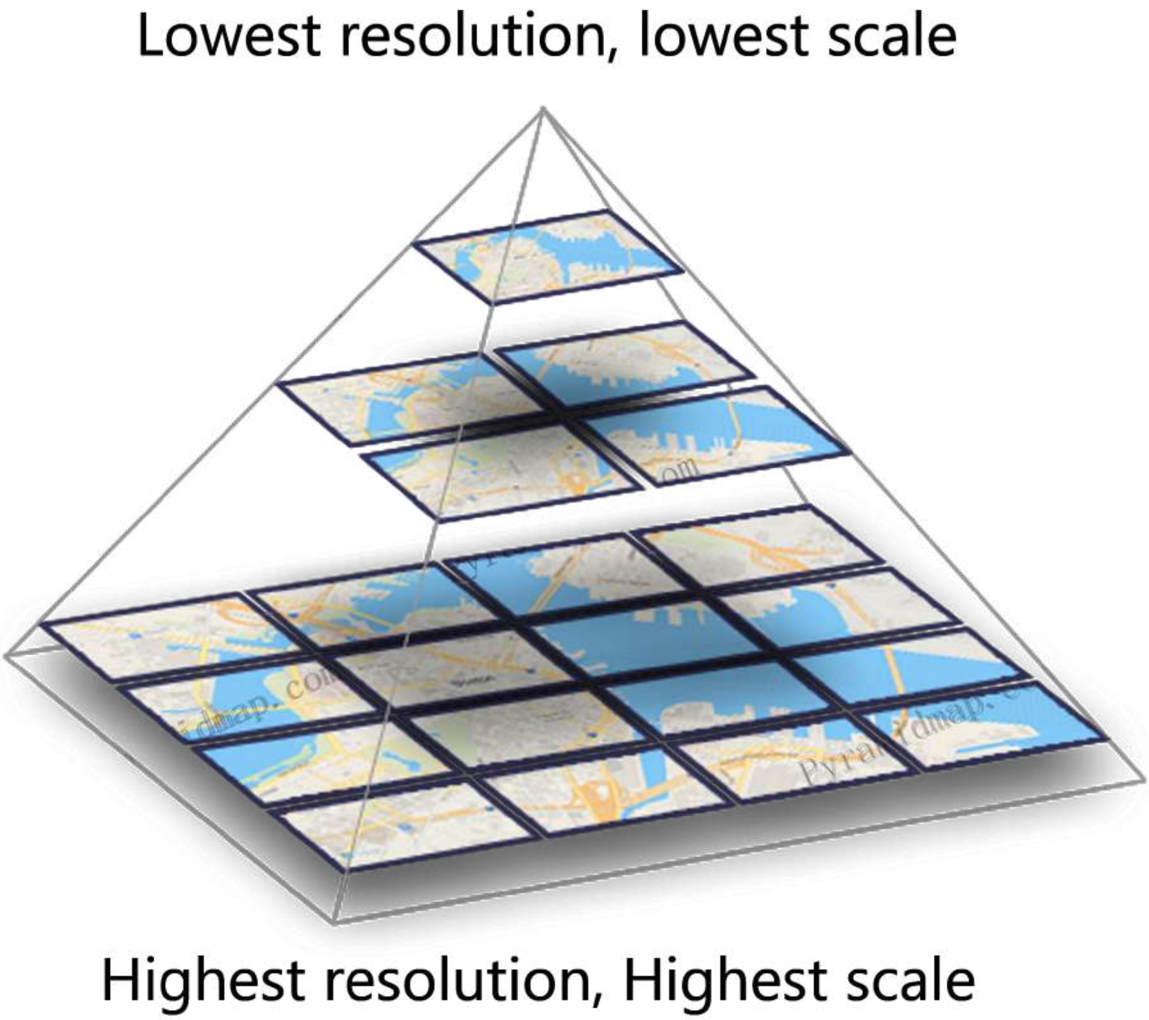


Figure 6-65： Map tile pyramid model

The main principle of a tiling pyramid is to project the Earth of a curved surface onto a two-dimensional plane based on a specific map projection coordinate system, and then divide the two-dimensional plane into multiple scales, which is equivalent to creating multiple digital maps with different resolution levels. Each level corresponds to corresponding codes, and the higher the level, the higher the resolution corresponding to the map; Then, the global spatial range map at each level is gridded using a certain spatial partitioning method, dividing it into fixed size square grid images of several rows and columns. These segmented regular individual grid units are called tiles, and the partitioning method for each level is in the same way.

According to this model, map tiles of corresponding levels can be scheduled based on the displayed range, and corresponding grids can be displayed based on the region, thereby further reducing the amount of data requested (read). Compared to traditional maps, tiling maps reduce the amount of data requested (read) and accelerate rendering efficiency, but it takes up more storage space. It is a typical space for time solution, and the drawbacks of this solution can be almost ignored in today's increasingly low storage costs.

6.8.2 Tile coordinate system

The encoding of all tiles is based on the tile coordinate system, and the origin of the tile coordinate system can be represented in two forms:

①TMS standard: TMS (Tile Map Service) is a tile map service defined by the Open Source Geospatial Foundation (OSGeo). In the TMS specification (GeoServer follows this specification), the YX order is used, where latitude at the first and longitude at the last, with the bottom left corner of the map as the origin and the Y coordinate increasing from south to north. The principle is shown in Figure 6-66.

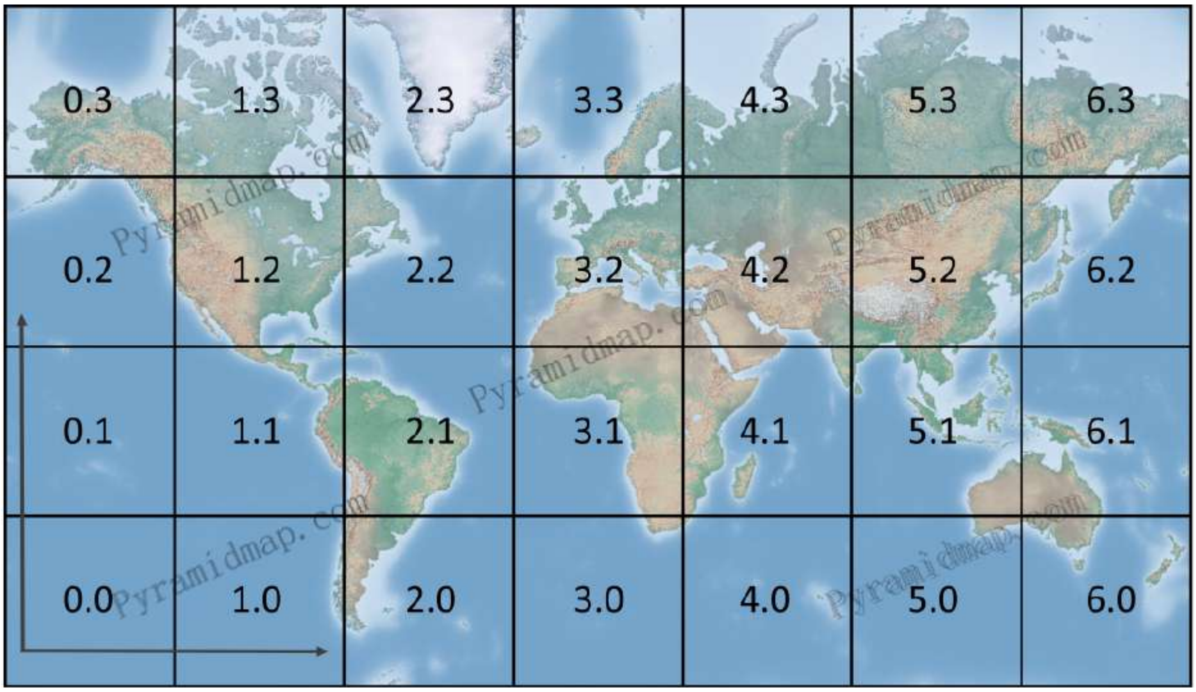


Figure 6-66: Schematic diagram of TMS tile specification

The numbering method for tiles is to start from the bottom left corner and increase the sequence number in ascending/descending order.

②XYZ standard: It is a tiling specification developed by Google/OSM(Open Street Map), using the XY order, where longitude at the first, latitude at the last, and the origin is in the upper left corner of the map. The Y coordinate increases from north to south, as shown in Figure 6-66.

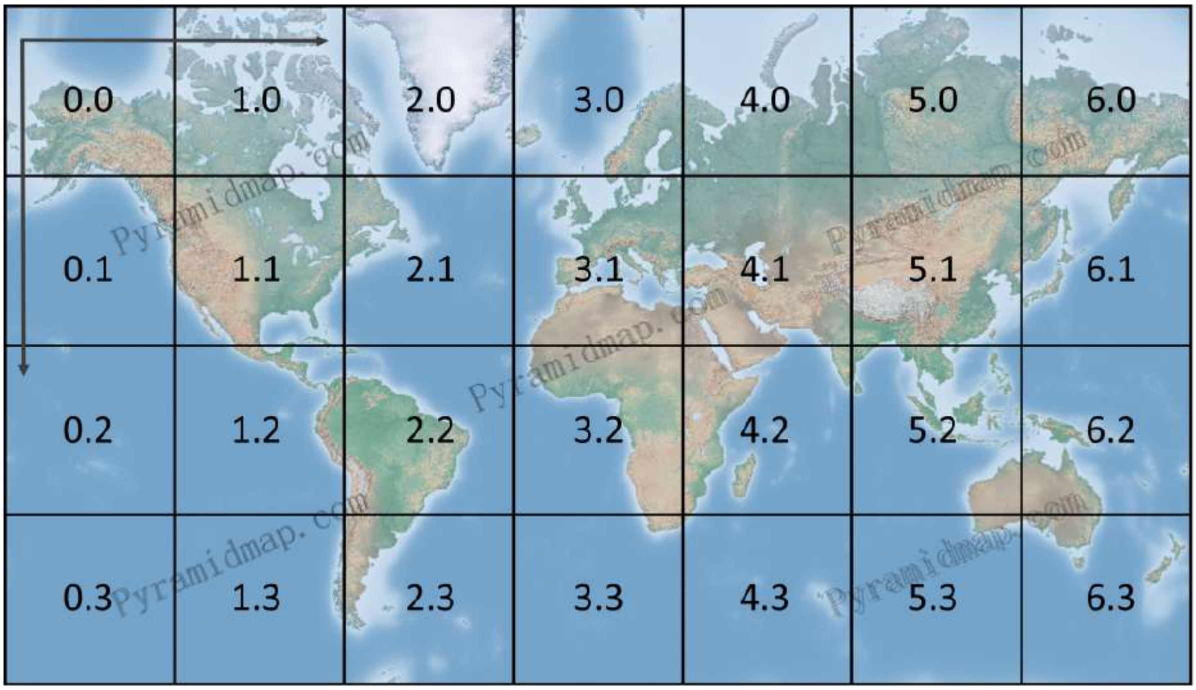


Figure 6-67: Schematic diagram of XYZ tile specification

The numbering method for TILEs is to start from the top left corner and increase the sequence number in descending/rightward order. Tiled layer related attributes, including the coordinate system used, EPSG: numerical code; The width and height of the tiles are 256x256 pixels, and the Origin coordinates are (-180, 90). Each pixel represents a unit of longitude and latitude (units per pixel) of 0.703125/2 ^ n (n is the number of layers of tiles). From this, it can be calculated that when the tile level is level 0, the longitude and latitude across a tile is 180 degrees, indicating that the level 0 is composed of two tiles of 256x256 size side by side.

6.8.3 Tile representation

Example as TMS standard, the representation of map tiles is shown in Figures 6-68.

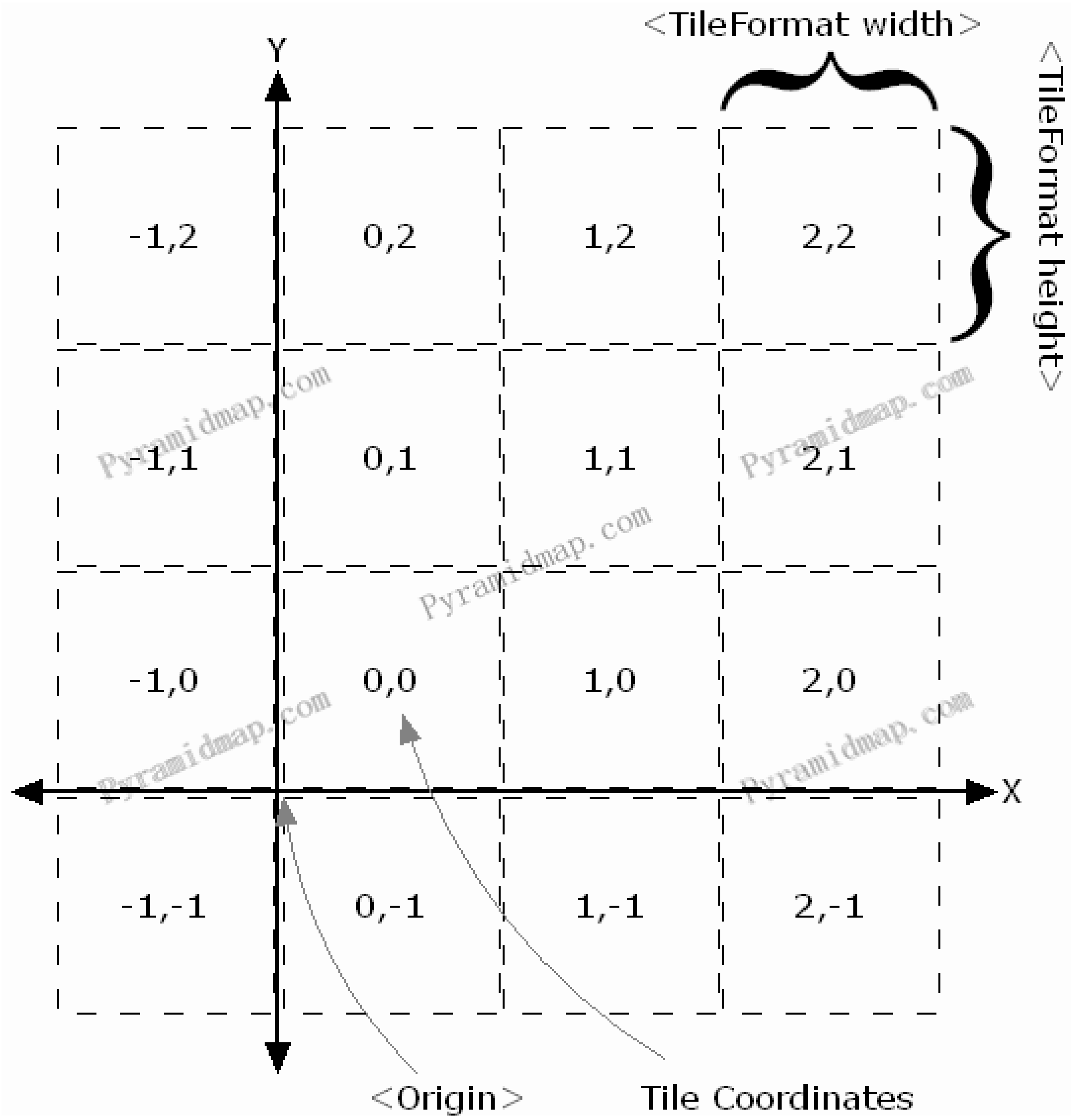


Figure 6-68: Schematic diagram of XYZ tile specification

The attributes related to the tiled layer, including the coordinate system used, (EPSG: numerical code is a coordinate system parameter published and maintained by the European Petroleum Survey Group, and different numerical codes represent different coordinate systems, such as the commonly used WGS84 the EPSG code is 4326); The size of tiles (width&height), such as 256x256 or 512x512 pixels; The origin coordinates in TMS format are (-180, -90), and the origin coordinates in XYZ format are (-180, 90); The number of units per pixel represented by each pixel is 0.703125/2 ^ n (n is the number of layers of tiles). From this, it can be calculated that when the tile level is level 0, the span of a tile is 180 degrees, indicating that level 0 is composed of two tiles of 256x256 size side by side.

The level is represented by z, where the longitude direction of the tiles (refers to the method of changing the longitude of the tiles, that is, east-west direction, east direction is positive) is numbered as x, and the latitude direction (refers to the direction of changing the dimension of the tiles, TMS refers to north-south direction, north direction is positive; XYZ refers to north-south direction, south direction is positive) is numbered as y, and z is the pyramid hierarchy of the current tile, which is the scaling ratio of the current layer. Therefore, each tile can be uniquely described by a three-dimensional tuple (x, y, z). The tile representation of the map at zoom=1 is shown in Figures 6-69.

Figure 6-69: Tile representation of the map at zoom=1

The tile representation of the map at zoom=2 is shown in Figures 6-70.

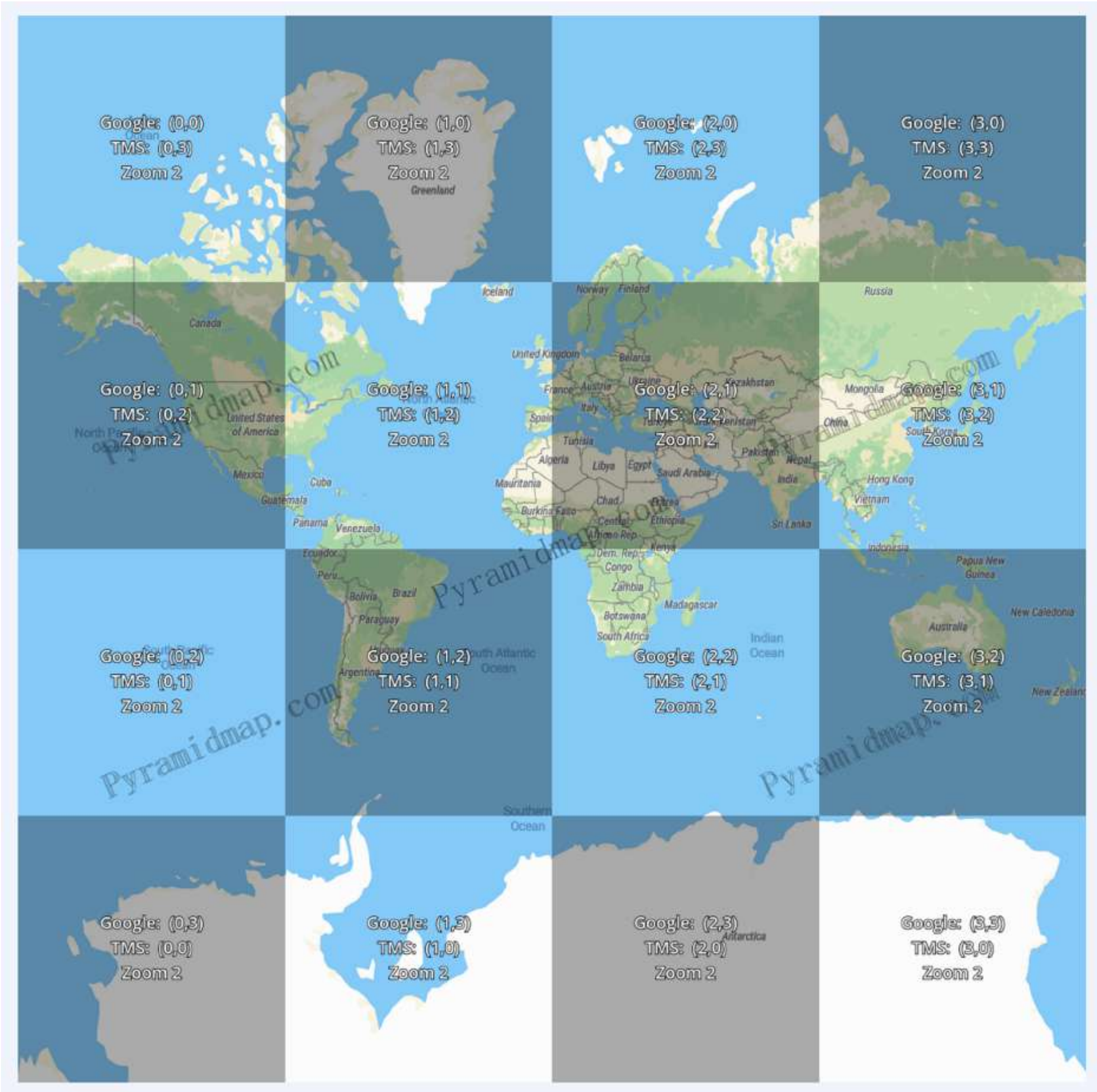


Figure 6-70: Tile representation of the map at zoom=2

In the traditional sense, map tile is oriented towards raster on the server side. After the layers are rendered according to the preset style, they are graded and tiled according to the set coordinate system, extend, and zoom level. Map tiles are stored in the form of static bitmaps in the server path, and the tiled images of the corresponding regions are returned based on the client's request extend and scale.

PyramidMap supports all the standard vector and raster map tile processing mentioned above, including: 1: TMS standard vector tile, 2: XYZ standard vector tile, 3: MVT standard vector tile, 4: MBTiles standard vector tile, 5: TMS standard raster tile, 6: XYZ standard raster tile, and 7: MBTiles standard raster tile. The tile operation entry is shown in Figure 6-71.

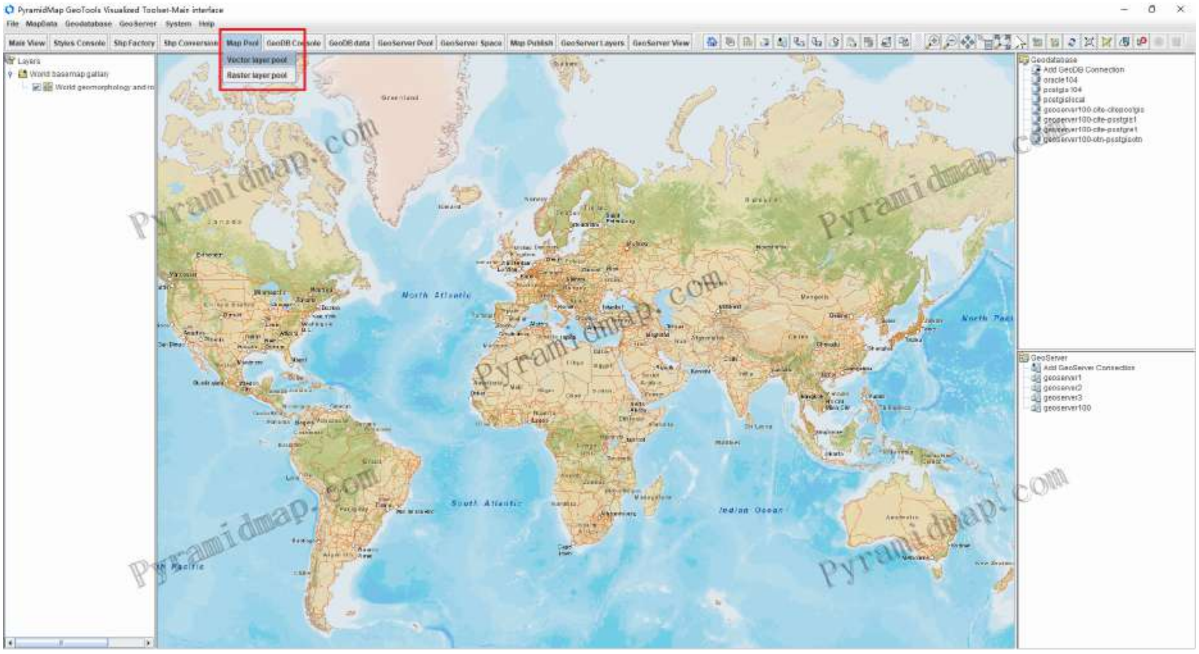


Figure 6-71: PyramidMap Layer Tile Entry

PyramidMap tiling operations are implemented in the vector map resource pool and the raster map resource pool, respectively.

6.8.4 Build Vector TMS tile

TMS (Tile Map Service) is a tile standard defined by the Open Source Geospatial Foundation (OSGeo). TMS defines tile starting from the bottom left corner of the map, with the bottom left corner of the map as the origin of the tile and coding system, and the vertical coordinate (Y value) increasing from south to north. PyramidMap supports TMS tile of vector layers, providing dynamic tile data loading at different resolutions for web GIS applications. In the PyramidMap vector resource pool, select the vector layer in batch mode, as shown in Figure 6-72.

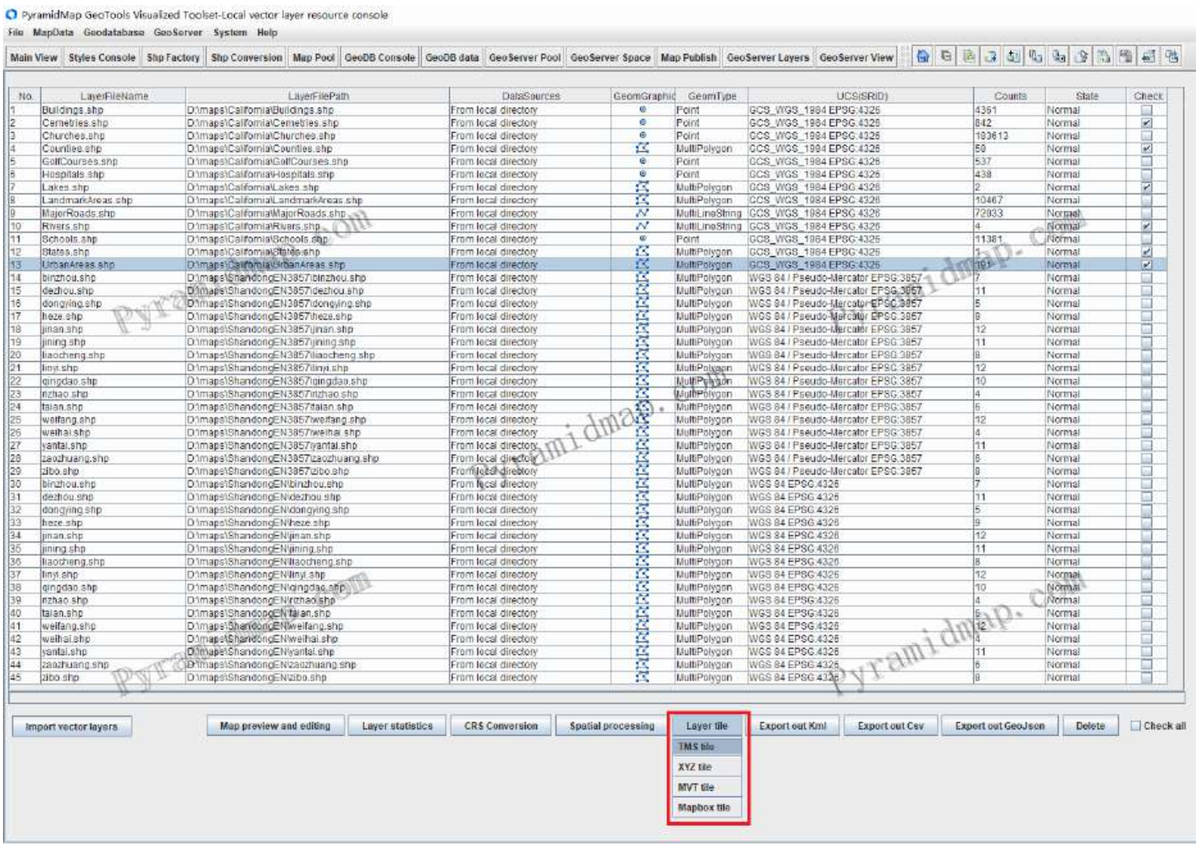


Figure 6-72: Select Vector layers for TMS tile

Select the TMS tile option and then open the tile scheme interface, as shown in Figure 6-73.

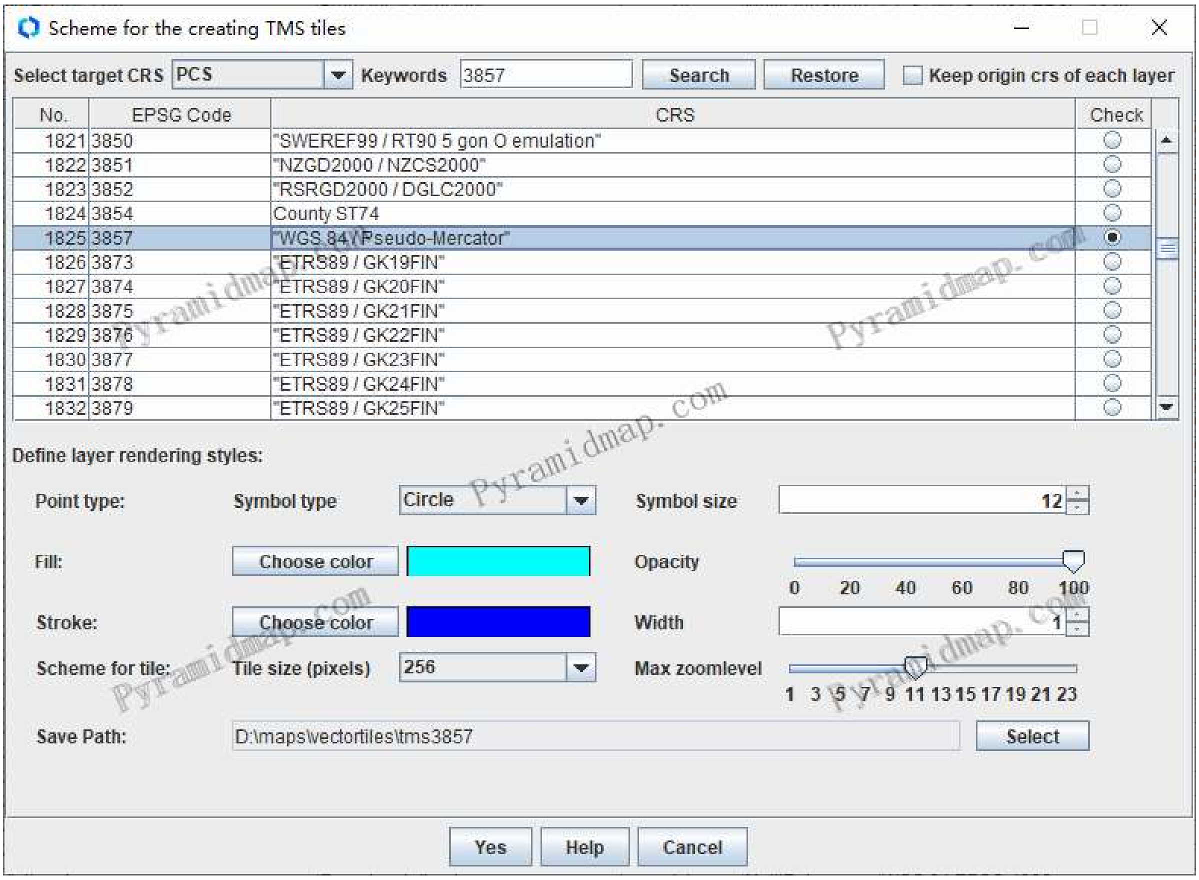


Figure 6-73: TMS tile scheme for vector layer

In the tiling scheme, select the target coordinate system, which is the coordinate system of the target tile dataset. The default option is to keep the original coordinate system of each layer. If need to change it, you can uncheck it and select the desired target coordinate system. The tile operation will automatically convert the coordinate system based on your selection. PyramidMap provides the vast majority of standard coordinate system and is still being continuously updated. It should be noted that although PyramidMap supports tiling operations in all standard coordinate systems, most WebGIS platforms currently use the Web Mercator 3857 coordinate system. Therefore, in general, the Web Mercator 3857 coordinate system is chosen for tile operations.

The rendering scheme defines the rendering mode of the layer and supports rendering configurations for all collection types such as Point/MultiPoint, Linestring/MultiLinestring, Polygon/MultiPolygon, including point type symbols, size, stroke width, stroke and fill colors, and the opacity.

The scheme defines the tile size and the maximum scaling level. The tile size is measured in pixels, representing the size of each target tile.

It should be noted that tile process is a work that consumes system resources and requires high system performance. The tiling speed depends on system performance. Therefore, it is strongly recommended to run the tile process on the workstation level computers.

The tiling process is shown in Figures 6-74.

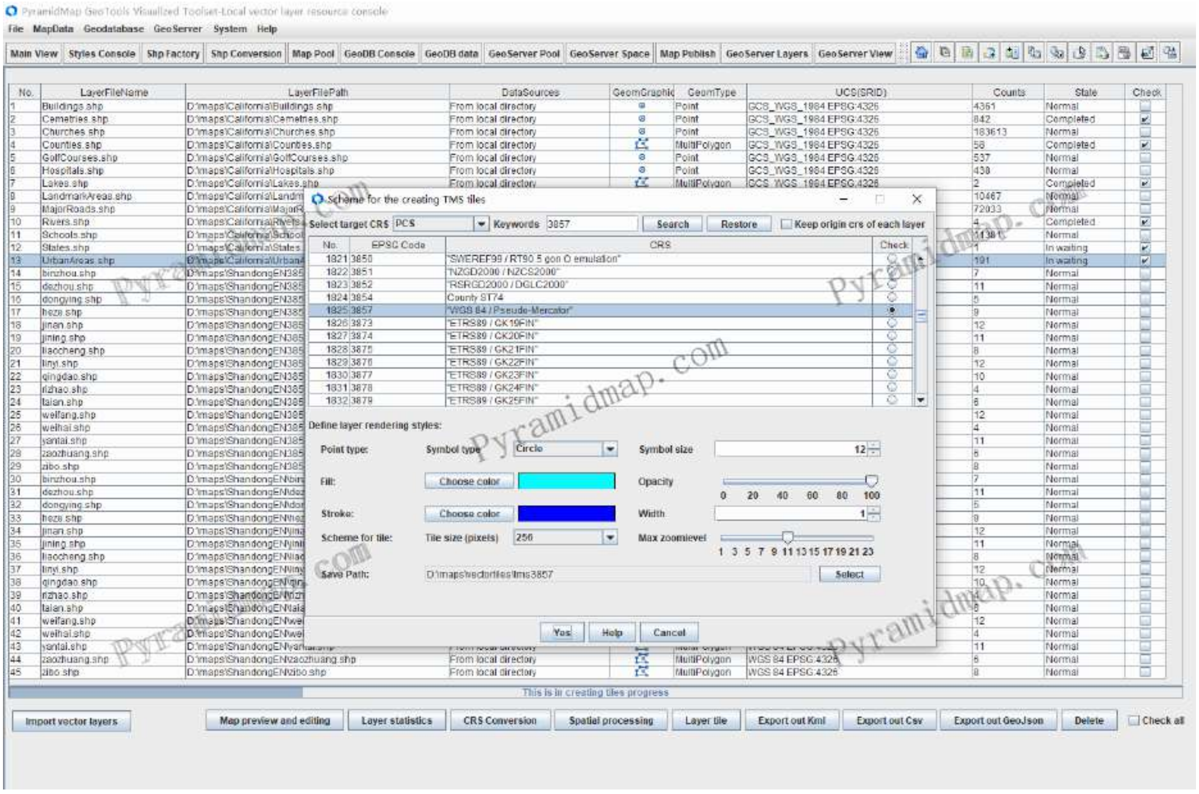


Figure 6-74: TMS tiling process of vector layers

The progress bar indicate the tiling process and status of each selected layer. The completion prompt is shown in Figure 6-75.

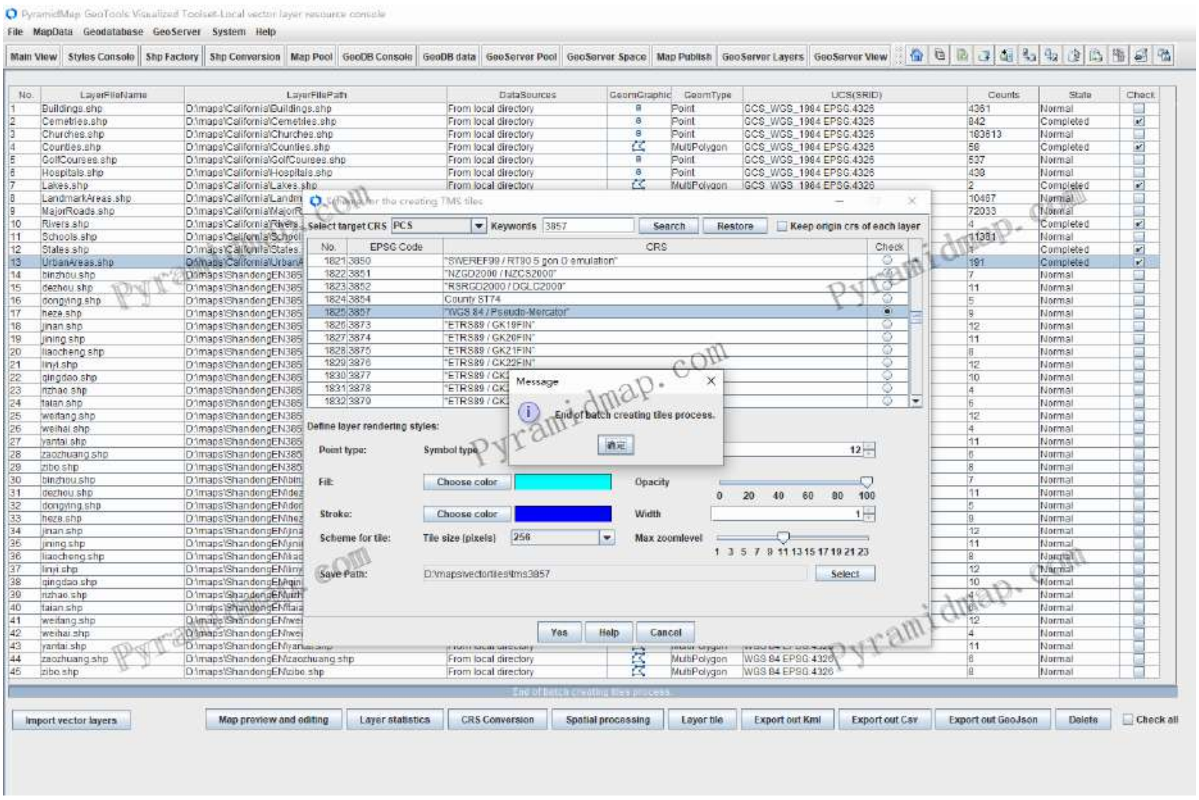


Figure 6-75: TMS tiling completed prompt

The final generated tiles will be classified according to the original layer name, grouped according to the Z value (zoom level) and X value, named according to the Y value, and saved to the corresponding hierarchical directory. At the same level, the tiles will be grouped based on the X value. The tiles with the same X value will be saved to the same path which named with the X value, and the tiles will be named with the Y value. The directory structure is shown as Figure 6-76.

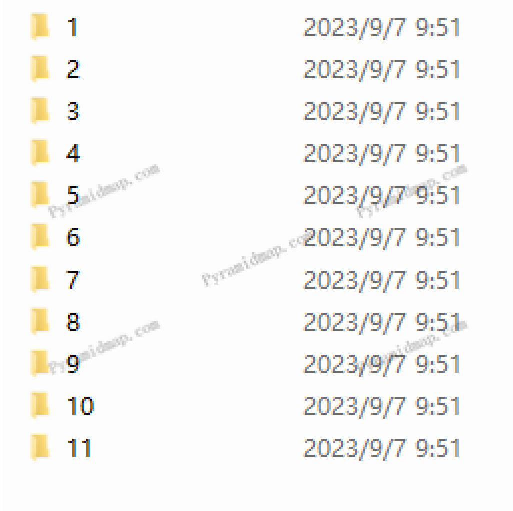


Figure 6-76: The target TMS tiles folder which corresponding to the zoomlevel

View the tile file at a certain level, as shown in Figures 6-77.

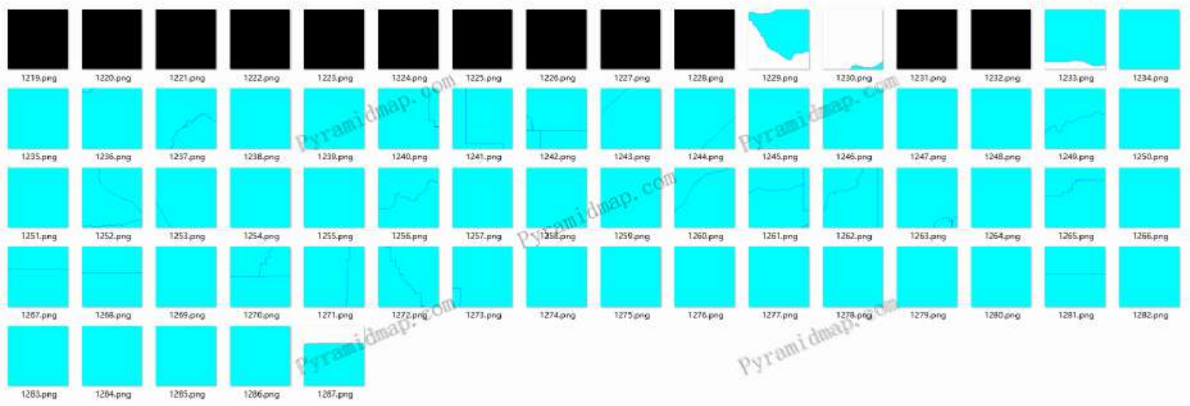


Figure 6-77: Vector layer TMS tiles file

When storing tiles, the code uses a folder structure that follows the TMS (Tile Map Service) standard, namely Z/X/Y.png, where Z represents the zoom level and X and Y represent the coordinates of the tiles. During this process, the code also checks and creates necessary folders. After completing the tiling, deploy the tiles to a web server, such as Apache Tomcat, nothing with the professional mapserver such as GeoServer, that will greatly reduce the difficulty and cost of map deployment and maintenance. You can deploy independently, access tiles data through HTTP URLs, or integrate deployment with projects, and access tiles data through relative paths. As an example, we demonstrate the web gis application through Leaflet, the complete code is as follows:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Leaflet vector TMS tiles Example</title>
    <link rel="stylesheet"
href="https://unpkg.com/leaflet@1.7.1/dist/leaflet.css" />
</head>
<style type="text/css">
    body {
        margin: 0;
        padding: 0;
```

```
    }
    html, body, #map{
        width: 100%;
        height: 100%;
    }
</style>
<body>

<div id="map" ></div>

<script src="https://unpkg.com/leaflet@1.7.1/dist/leaflet.js"></script>
<script>
    var map = L.map('map').setView([31.562710059362658,120.29751401540051], 8);
    // loading the ArcGIS basemap resource
L.tileLayer('https://server.arcgisonline.com/ArcGIS/rest/services/World_Topo_Map/
MapServer/tile/{z}/{y}/{x}').addTo(map);
    // loading the local TMS tiles,Please modify according to the real path of
yourself.
    L.tileLayer('.data/tiles/jinan/{z}/{x}/{y}.png', {
      tms: true,  // Indicates this is a TMS tile
      opacity: 0.7 // The opacity of the polygon tile can be adjusted as needed
    }).addTo(map);
</script>

</body>
</html>
```

Then display on the map as shown in Figures 6-78.

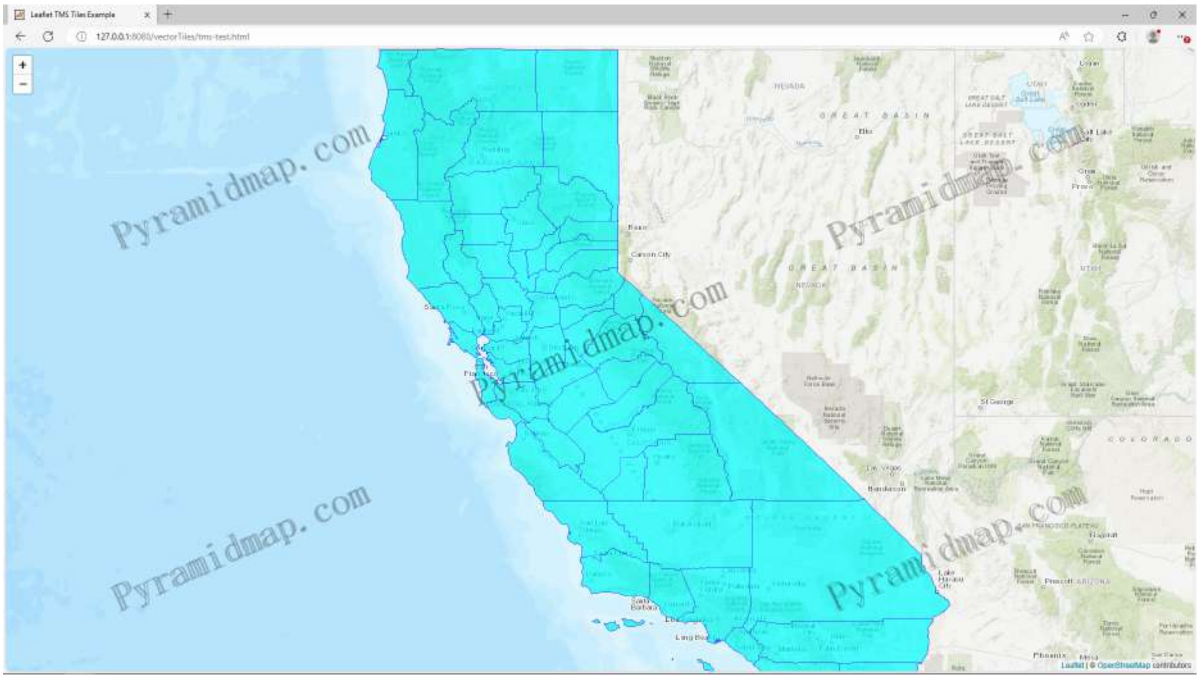


Figure 6-78: The TMS tiles loading in Leaflet

It should be noted that although PyramidMap supports tiling operations in all standard coordinate systems, most WebGIS platforms currently use the web Mercator 3857 coordinate system. Therefore, in general, the Web Mercator 3857 coordinate system should be chosen for tile.

6.8.5 Build Vector XYZ tile

XYZ tile is a standard adopted by Google/OSM (Open Street Map). TMS defines tile as starting from the bottom left corner of the map, with the bottom left corner of the map as the origin of the tile and coding system, and the vertical coordinate (Y value) increasing from south to north. On the contrary, Google uses the top left corner of the map as the origin of the tile and coding system, with the y-axis (Y value) increasing from north to south. In the PyramidMap vector map resource pool, select the vector layer and support batch selection, as shown in Figure 6-79.

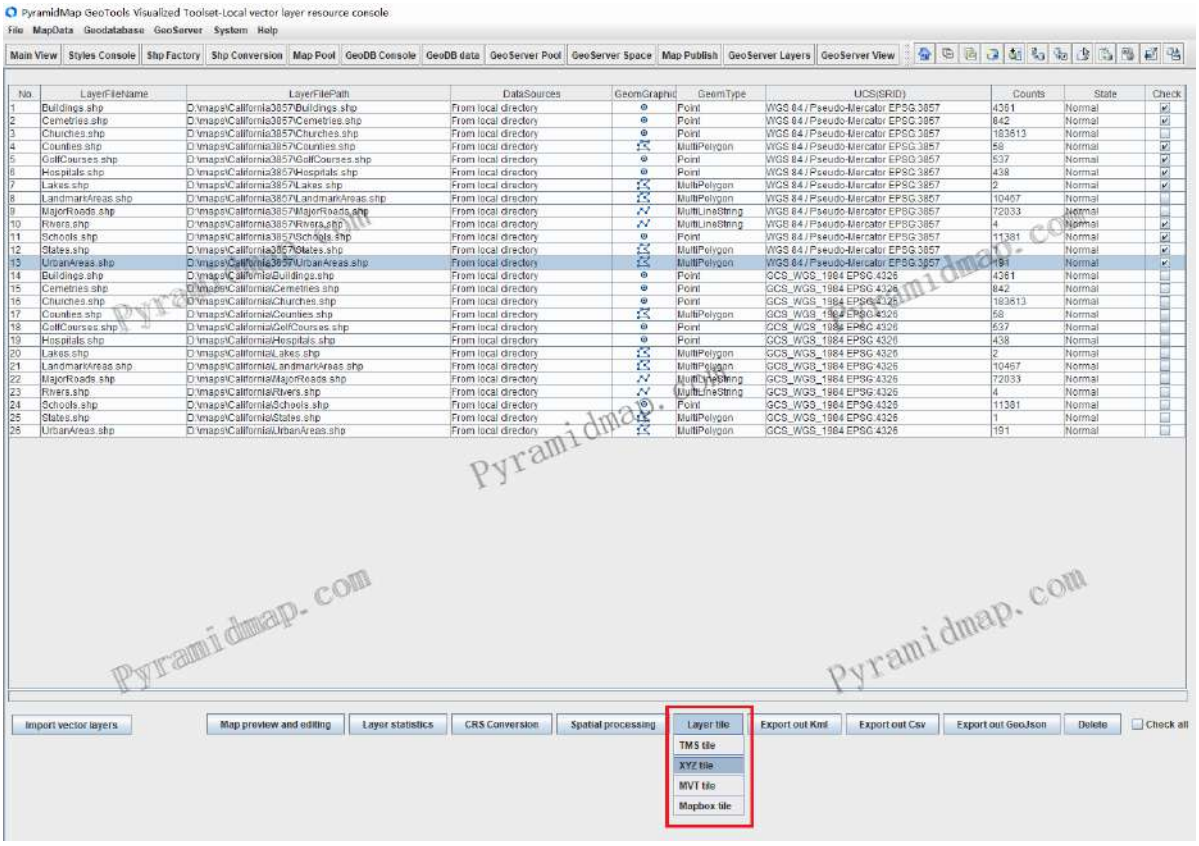


Figure 6-79: Select vector layers for XYZ tile

Select the XYZ format tile and then open the tile scheme interface, as shown in Figure 6-80.

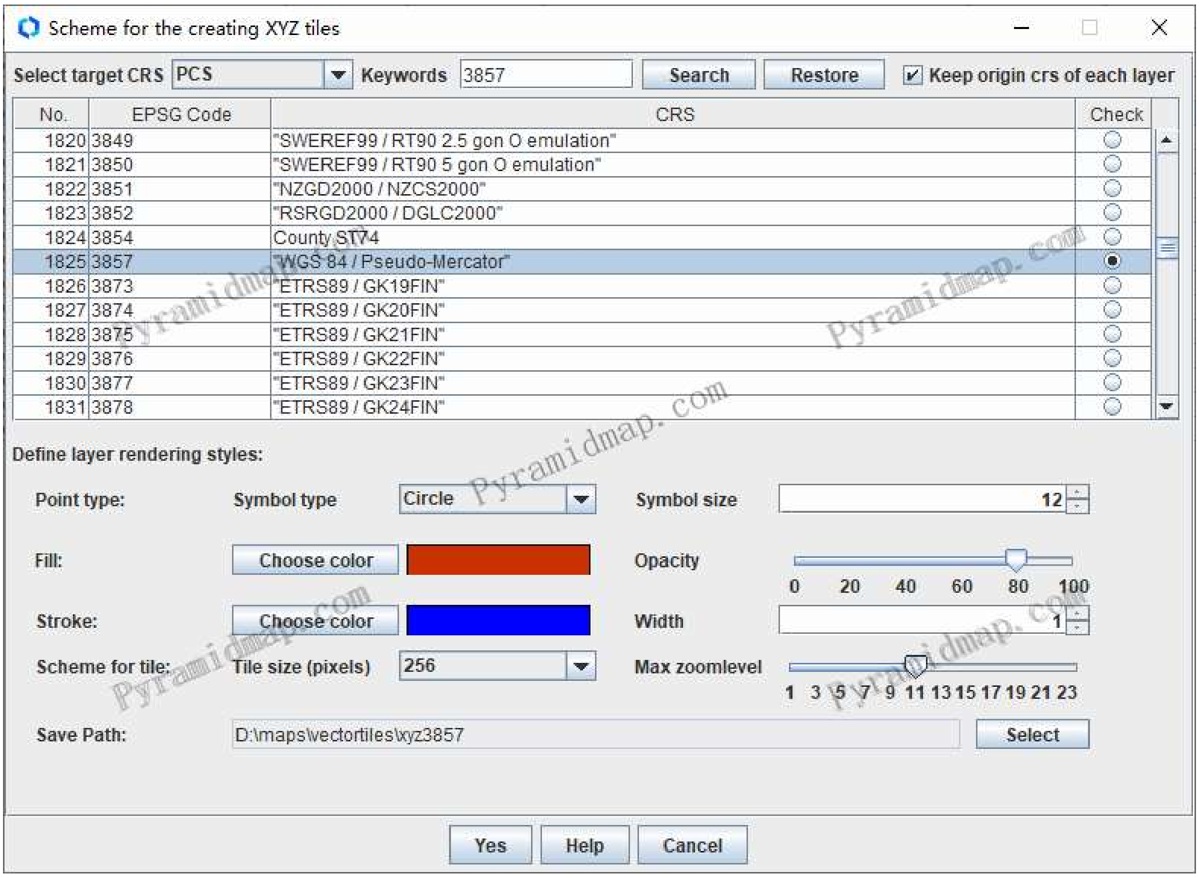


Figure 6-80: XYZ tile scheme for vector layer

In the tiling scheme, select the target coordinate system, which is the coordinate system of the target tile dataset. The default option is to keep the original coordinate system of each layer. If need to change it, you can uncheck it and select the desired target coordinate system. The tile operation will automatically convert the coordinate system based on your selection. PyramidMap provides the vast majority of standard coordinate system and is still being continuously updated. It should be noted that although PyramidMap supports tiling operations in all standard coordinate systems, most WebGIS platforms currently use the Web Mercator 3857 coordinate system. Therefore, in general, the Web Mercator 3857 coordinate system is chosen for tile operations.

The rendering scheme defines the rendering mode of the layer and supports rendering configurations for all collection types such as Point/MultiPoint, Linestring/MultiLinestring, Polygon/MultiPolygon, including point type symbols, size, stroke width, stroke and fill colors, and the opacity.

The scheme defines the tile size and the maximum scaling level. The tile size is measured in pixels, representing the size of each target tile.

It should be noted that tile process is a work that consumes system resources and requires high system performance. The tiling speed depends on system performance. Therefore, it is strongly recommended to run the tile process on the workstation level computers.

The tiling process is shown in Figures 6-81.

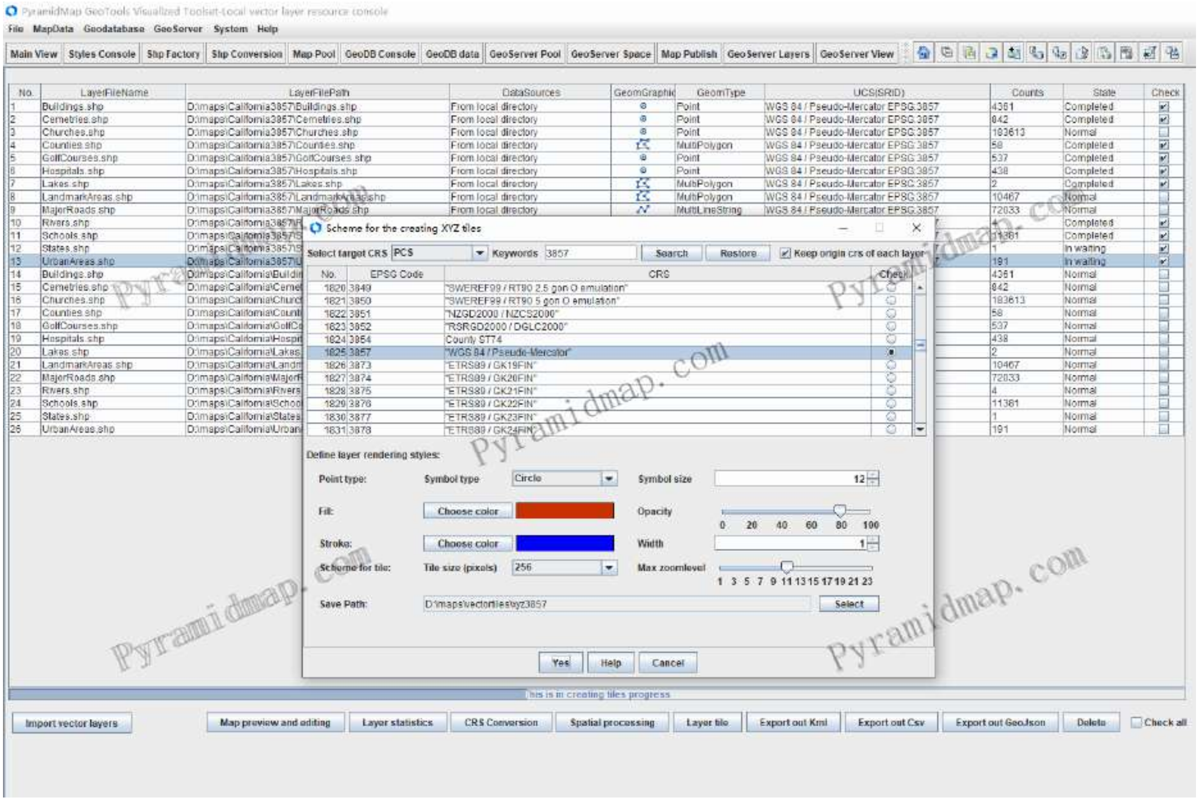


Figure 6-81: XYZ tiling process of vector layers

The progress bar indicate the tiling process and status of each selected layer. The completion prompt is shown in Figure 6-82.

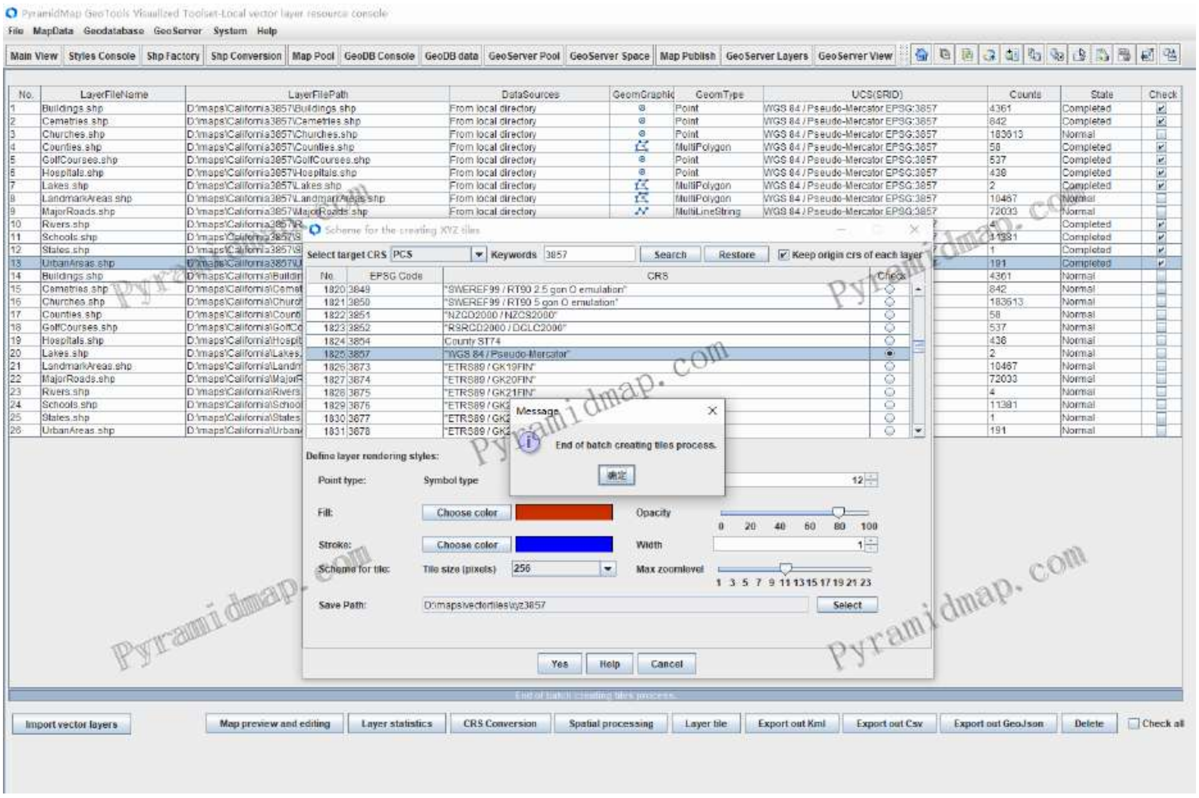


Figure 6-82: XYZ tiling completed prompt

The final generated tiles will be classified according to the original layer name, grouped according to the Z value (zoom level) and X value, named according to the Y value, and saved to the corresponding hierarchical directory. At the same level, the tiles will be grouped based on the X value. The tiles with the same X value will be saved to the same path which named with the X value, and the tiles will be named with the Y value. The directory structure is shown as Figure 6-83.

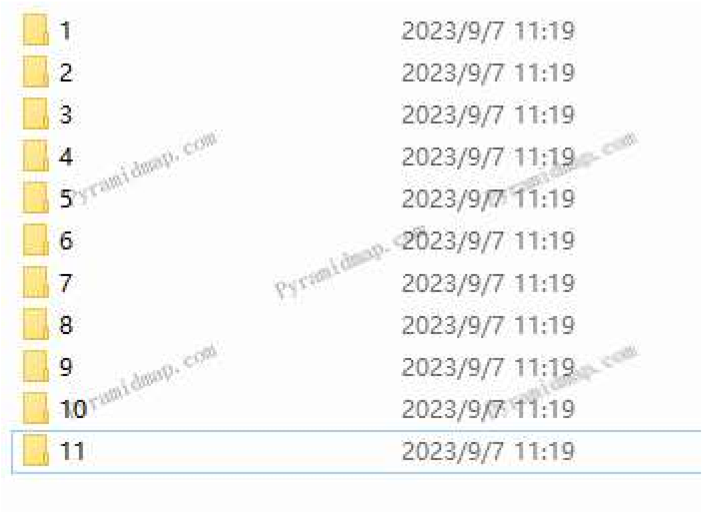


Figure 6-83: The target XYZ tiles folder which corresponding to the zoomlevel

View the tile file at a certain level, as shown in Figures 6-84.

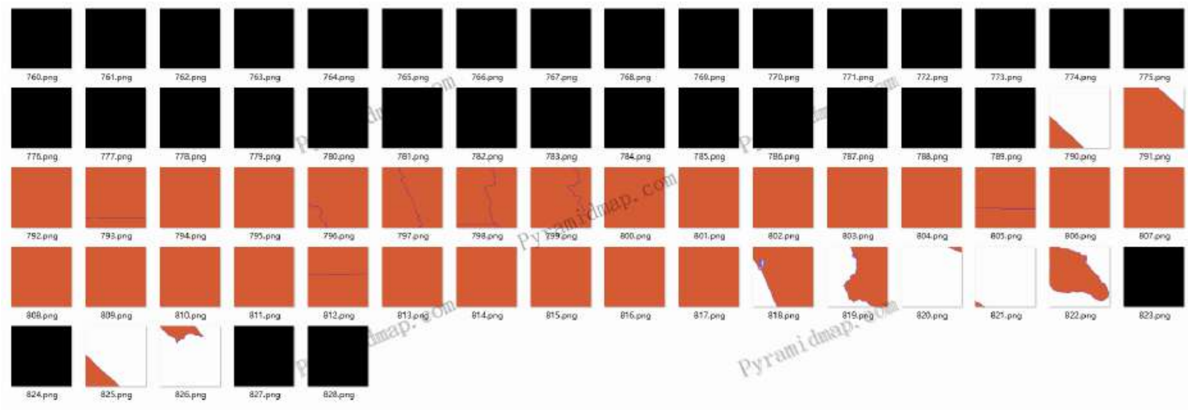


Figure 6-84: Vector layer XYZ tiles file

When storing tiles, the code uses a folder structure that follows the TMS (Tile Map Service) standard, namely Z/X/Y.png, where Z represents the zoom level and X and Y represent the coordinates of the tiles. During this process, the code also checks and creates necessary folders. After completing the tiling, deploy the tiles to a web server, such as Apache Tomcat, nothing with the professional mapserver such as GeoServer, that will greatly reduce the difficulty and cost of map deployment and maintenance. You can deploy independently, access tiles data through HTTP URLs, or integrate deployment with projects, and access tiles data through relative paths. As an example, we demonstrate the web gis application through Leaflet, the complete code is as follows:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Leaflet vector XYZ Tiles Example</title>
    <link rel="stylesheet"
href="https://unpkg.com/leaflet@1.7.1/dist/leaflet.css" />
</head>
<style type="text/css">
    body {
        margin: 0;
        padding: 0;
```

```
    }
    html, body, #map{
        width: 100%;
        height: 100%;
    }
</style>
<body>

<div id="map" ></div>

<script src="https://unpkg.com/leaflet@1.7.1/dist/leaflet.js"></script>
<script>
    //Initialize the map and set the center and zoom levels
    var map = L.map('map').setView([33.923710059362658,-118.23851401540051], 8);
    // Loading ArcGIS online basemap resource

L.tileLayer('https://server.arcgisonline.com/ArcGIS/rest/services/World_Topo_Map/
MapServer/tile/{z}/{y}/{x}').addTo(map);
    // Load local XYZ TILEs, please modify according to your file path.
    L.tileLayer('./data/tiles/xyz3857/UrbanAreas/{z}/{x}/{y}.png', {
       tms: false,  // Indicates this is an XYZ standard tiles
       opacity: 0.7 // The transparency of the tile can be adjusted as needed
    }).addTo(map);
</script>
</body>
</html>
```

The loading effect on the map is shown in Figures 6-85.

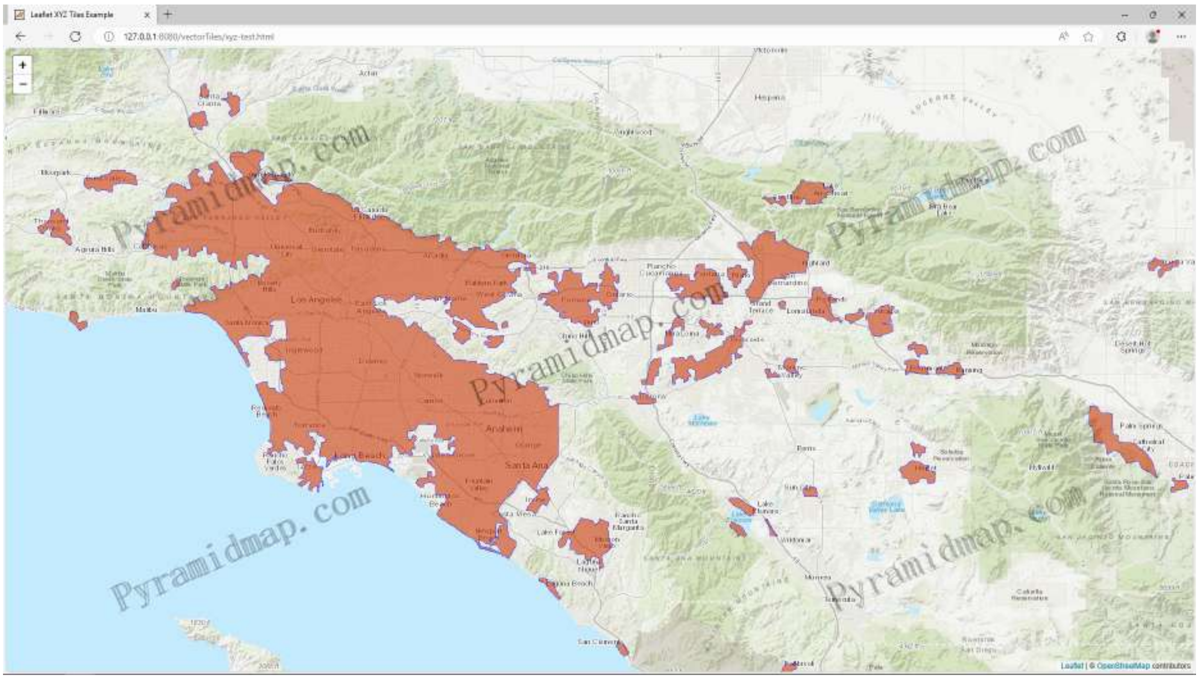


Figure 6-85: The XYZ format tiles loading in Leaflet

It should be noted that although PyramidMap supports tiling operations in all standard coordinate systems, most WebGIS platforms currently use the web Mercator 3857 coordinate system. Therefore, in general, the Web Mercator 3857 coordinate system should be chosen for tile.

6.8.6 Build Vector MVT tile

MVT (Mapbox Vector Tiles), as the name suggests, this is a vector tile standard created by Mapbox. At present, there are three main formats for vector tiles: mvt (pbf), geojson, and topojson, all of these which with the best compression rate and transmission speed being the mvt format. MVT divides the traditional tile mode into two processes which including vector tiles data and style rendering separately, it represents a more novel map service mode. Its characteristic is that the server only stores vector tile data, and the rendering is completed on the client side. MVT vector tile technology inherits the dual advantages of vector data and traditional tile, reflected in the following aspects:

- 1. Lossless storage: Tiles are stored in mvt format, based on each tile number. All sizes are 256 * 256; Smaller granularity and nearly lossless information;
- 2. Smaller volume: MVT has higher compression rate and smaller volume;
- 3. Fast response speed: This is an inherent characteristic of tiled data, and the server returns data according to the client's request level;
- 4. Fast transmission speed: The data and rendering are separated, and the server only returns mvt vector data, which reduces the network load and improves the transmission and loading speed;
- 5. Implement client-side rendering: In the web client, achieve custom style rendering, giving a more flexible and rich display effect;
- 6. Implement more UI interaction: MVT enables client side vectorization operations on maps, such as interactive operations, attribute and spatial queries, independent value rendering, thematic statistics, and other feature based functions, thereby enriching map functionality.
- 7. Make full use of hardware resources: mvt utilizes OpenGL/WebGL drivers, adjusts graphics card and memory performance, adapts to client screens, and performs high-precision vector rendering based on screen resolution. It can be said that for the same tile data, different hardware performance and rendering effects are different, which completely depends on the client's display system.

Based on the above advantages, MVT technology has quickly gained recognition and support from a wide range of map manufacturers. ArcGIS, Leaflet, MapTiler, and OpenLayers have all added support for the MVT specification.

Of course, when viewed dialectically, advantages and disadvantages come with each other, and there is no such thing as having only advantages and no disadvantages. The disadvantage of MVT is its by-product of pursuing advantages, which is its high technical cost. Currently, the using range of MVT is limited by the following points:

- 1. High technical difficulty: The usual approach is to upload the map to Mapbox studio and perform slicing hosting on the Mapbox platform. This requires registering a Mapbox account and customizing the tile mode, which is a complex process;
- 2. High usage cost: The usage cost of online MVT generation and hosting is very high. Taking Mapbox Studio as an example, its service space and traffic are limited, and if it exceeds the limit, fees will be charged. For ordinary users and small and medium-sized enterprises, this is not an ideal option;

PyramidMap provides the ability for users to independently produce, host, and use MVTs. PyramidMap can perform MVT tiles in localization, and deploy the tiles data freely. You can publish data within the programs together, and access tiles data through URL relative paths without any special configuration in the web application, greatly simplifying the map data processing, improving system deployment flexibility, and reducing project implementation costs.

Before tiling, let's first take a look at the original layer in shp format, using Building.shp as an example, which represents the iconic buildings, locations, and elevations of California, as shown in Figures 6-86.

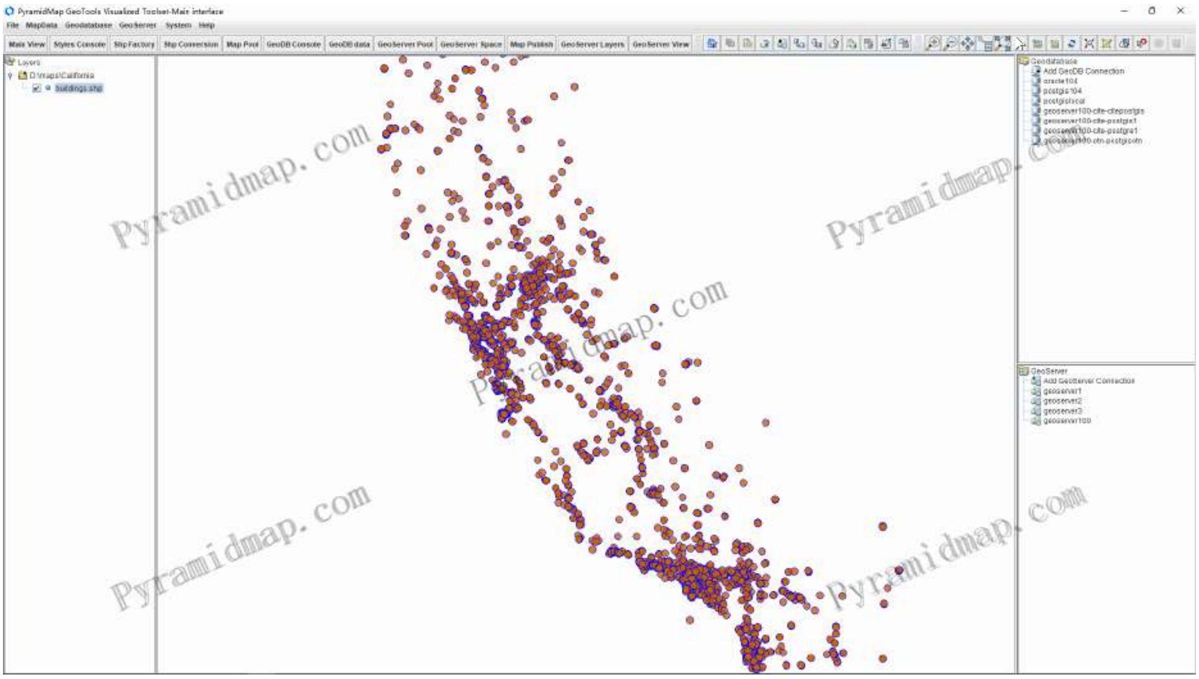


Figure 6-86: The Original Shp for MVT tiles

Its specific location is shown in Figures 6-87.

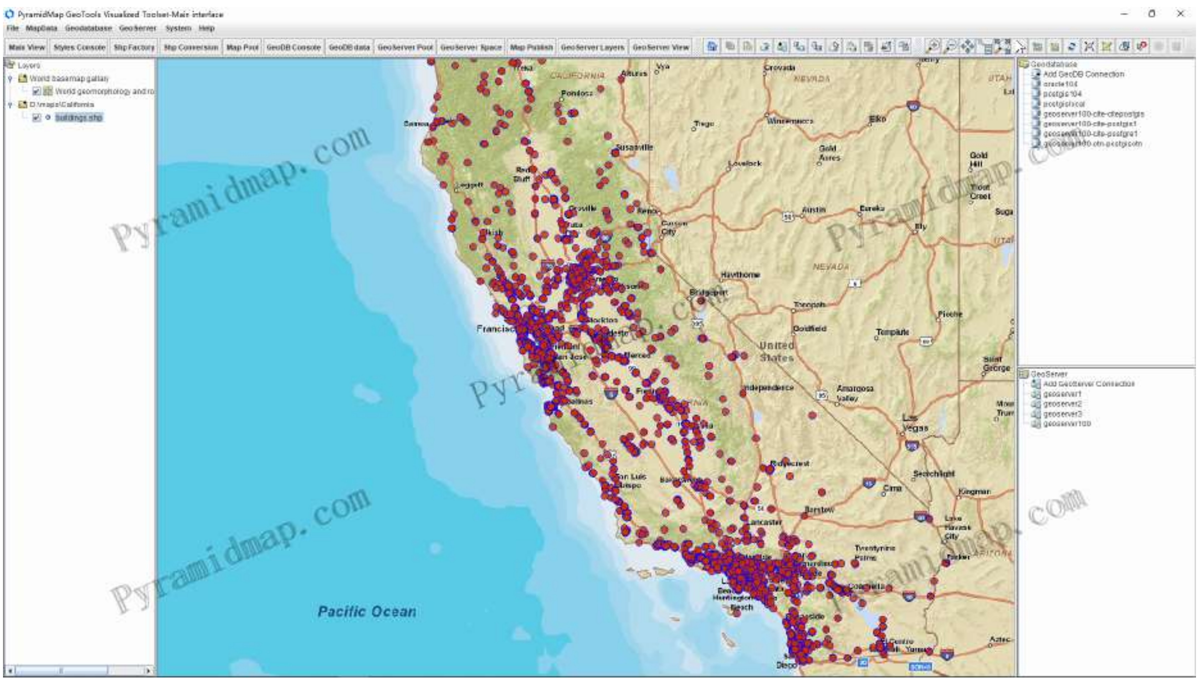


Figure 6-87: The specific location of the original Shp

Open data table of the shp file, as shown in Figure 6-88.

Figure 6-88: Open data table of Buildings.shp

Next, we will tile the vector layers including Buildings.shp in the vector resource pool under batch mode, as shown in Figure 6-89.

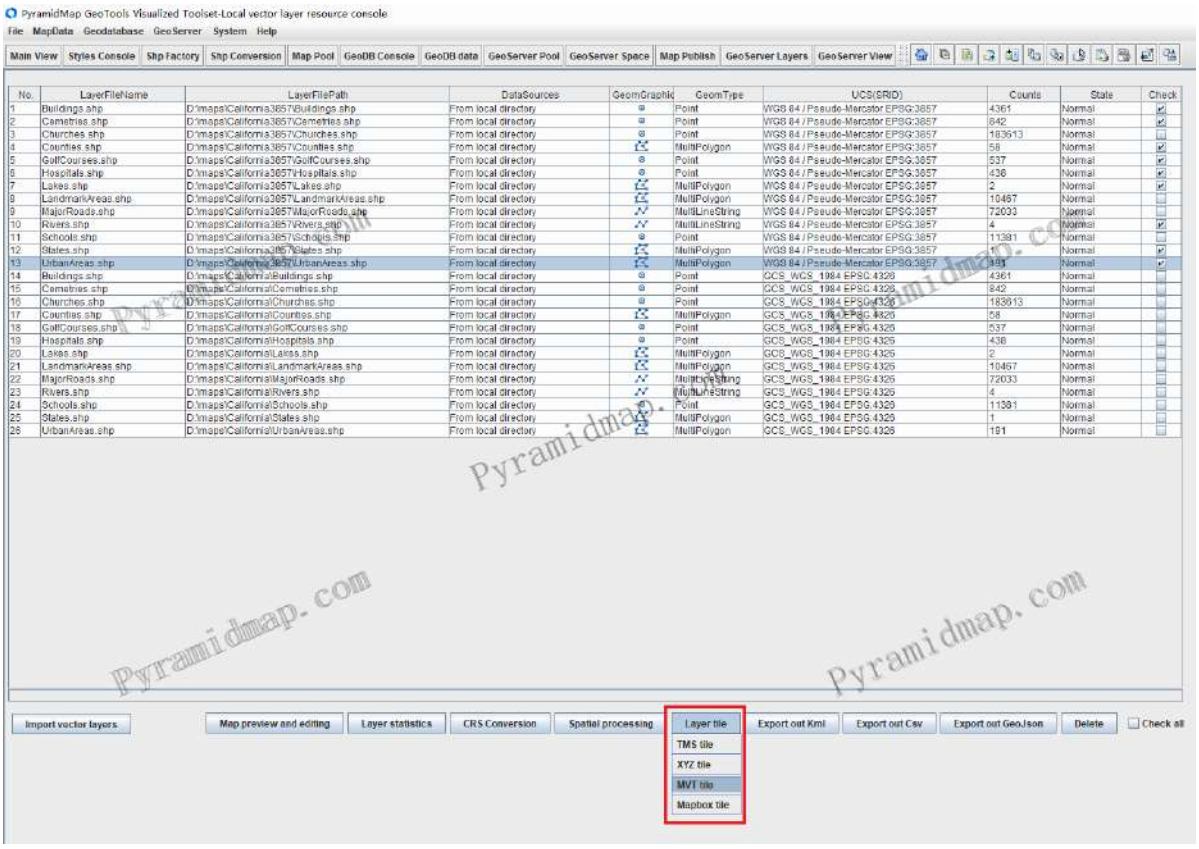


Figure 6-89: Choose Vector layers to MVT tiling

Select the MVT tile and then open the tile scheme interface, as shown in Figure 6-90.

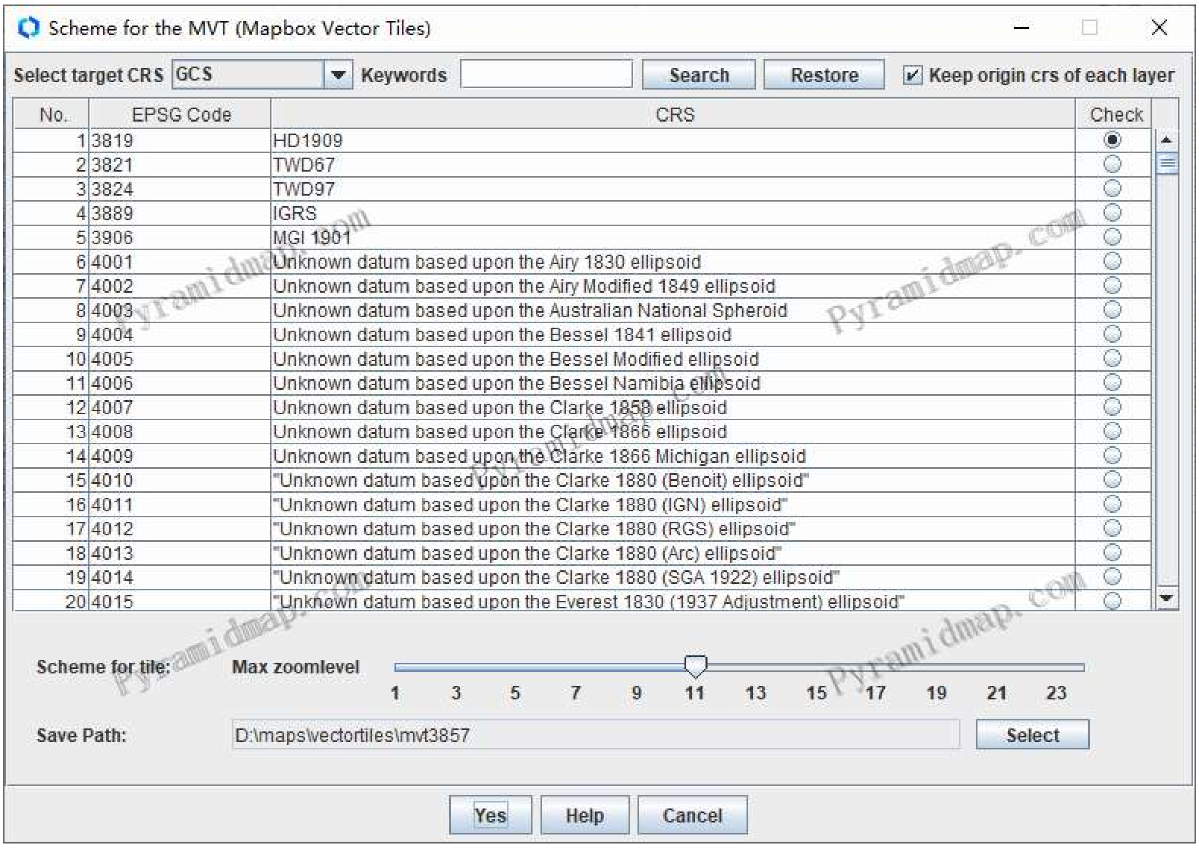


Figure 6-90: Vector layer MVT tiling scheme

In the tiling scheme, select the target coordinate system, which is the coordinate system of the target tile dataset. The default option is to keep the original coordinate system of each layer. If need to change it, you can uncheck it and select the desired target coordinate system. The tile operation will automatically convert the coordinate system based on your selection. PyramidMap provides the vast majority of standard coordinate system and is still being continuously updated. It should be noted that although PyramidMap supports tiling operations in all standard coordinate systems, most WebGIS platforms currently use the Web Mercator 3857 coordinate system. Therefore, in general, the Web Mercator 3857 coordinate system is chosen for tile operations.

Unlike TMS and XYZ tiling, MVT does not require pre set map rendering. In fact, the results of TMS and XYZ format tiling are still png tiles. The tile data is stored in the server as a static image and not rendered on the client. Although the data source is vector, the tiles are non vector. The MVT are only vector data, and map rendering is achieved through user-defined style files on the client, so there is no need to preset tile rendering strategies.

It should be noted that tile process is a work that consumes system resources and requires high system performance. The tiling speed depends on system performance. Therefore, it is strongly recommended to run the tile process on the workstation level computers.

The tiling process is shown in Figures 6-91.

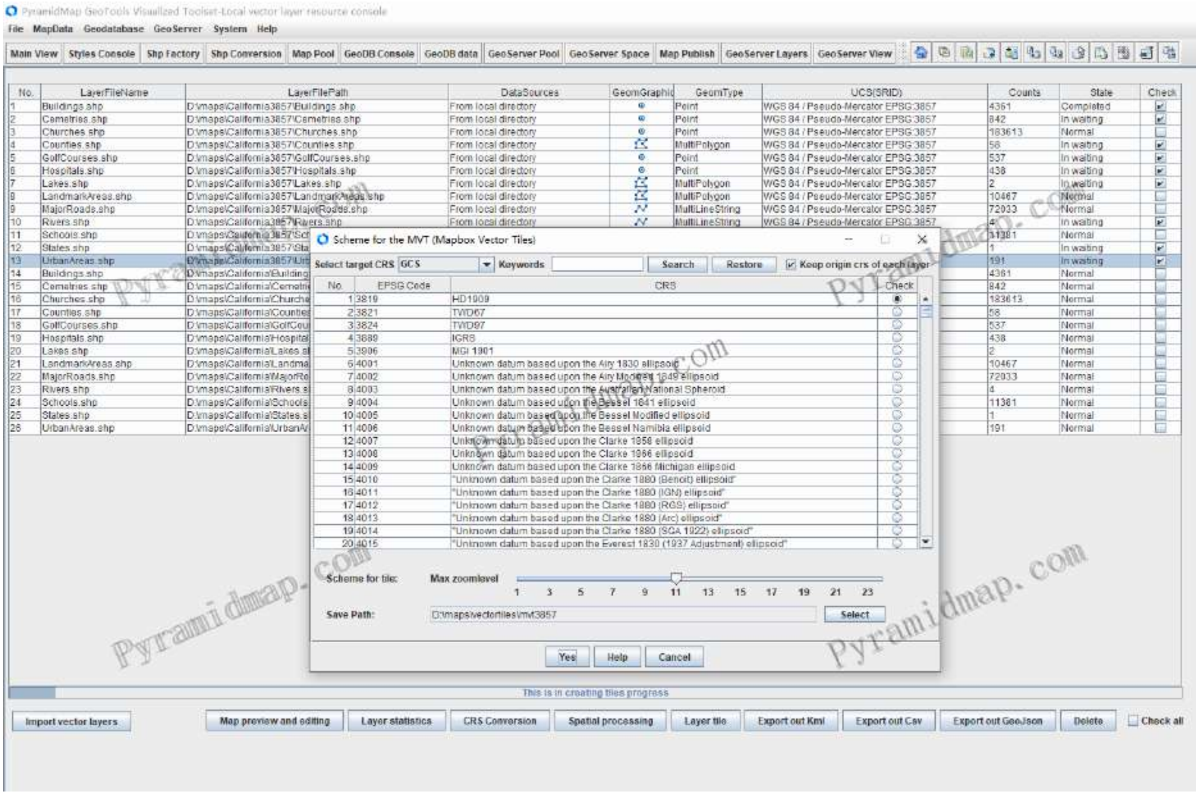


Figure 6-91: MVT tiling process of vector layers

The progress bar indicate the tiling process and status of each selected layer.

The final generated tiles will be classified according to the original layer name, grouped according to the Z value (zoom level) and X value, named according to the Y value, and saved to the corresponding hierarchical directory. At the same level, the tiles will be grouped based on the X value. The tiles with the same X value will be saved to the same path which named with the X value, and the tiles will be named with the Y value.

View the mvt tile files at a certain level. As shown in Figures 6-92.

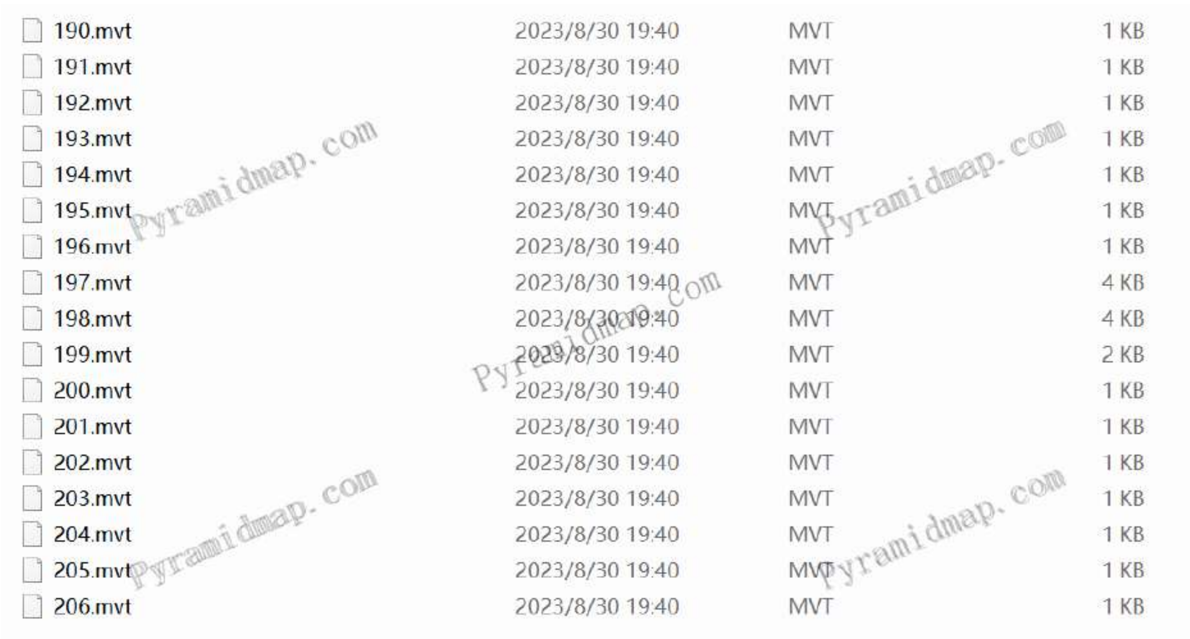
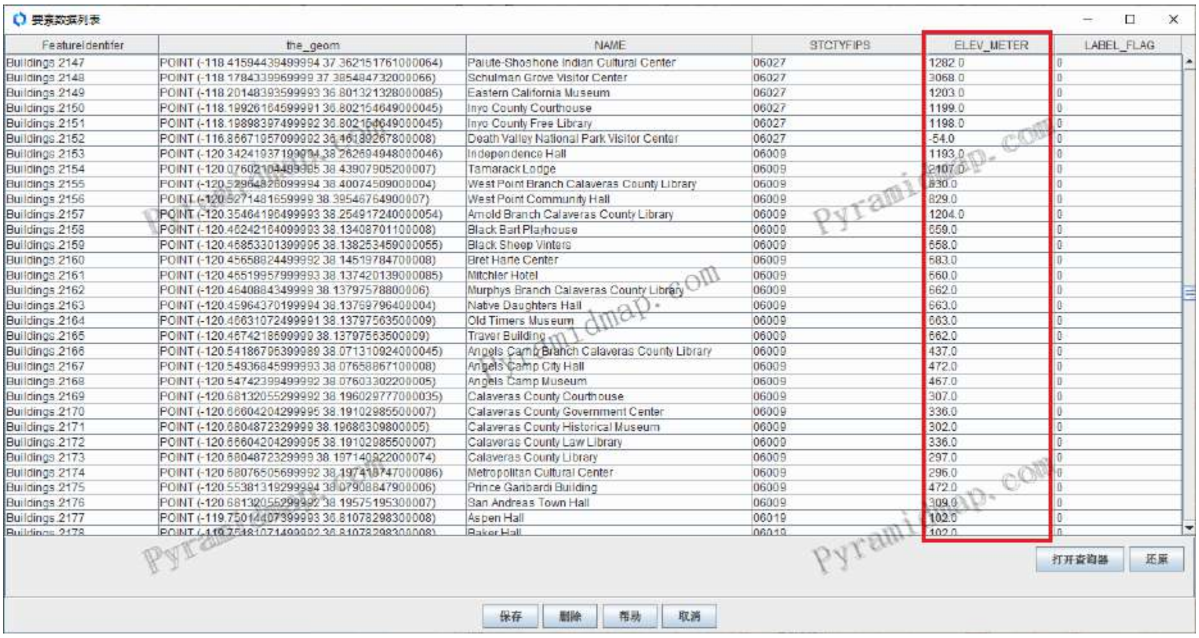


Figure 6-92: View the mvt tile files at a certain level

At this point, the MVT vector tiling process is completed.

When storing tiles, the code uses a folder structure that follows the TMS (Tile Map Service) standard, namely Z/X/Y.png, where Z represents the zoom level and X and Y represent the coordinates of the tiles. During this process, the code also checks and creates necessary folders. After completing the tiling, deploy the tiles to a web server, such as Apache Tomcat, nothing with the professional mapserver such as GeoServer, that will greatly reduce the difficulty and cost of map deployment and maintenance. You can deploy independently, access tiles data through HTTP URLs, or integrate deployment with projects, and access tiles data through relative paths. As an example, we demonstrate the web gis application through Leaflet.

Taking the Buildings.shp as an example, open its data table in PyramidMap, as shown in Figure 6-93.


要素数据列表

FeatureIdentifer	the_geom	NAME	STCTYFIPS	ELEV_METER	LABEL_FLAG
Buildings.2147	POINT (-118.41594439499994 37.36215176100064)	Paiute-Shoshone Indian Cultural Center	06027	1282.0	0
Buildings.2148	POINT (-118.17843399699999 37.38548473200066)	Schulman Grove Visitor Center	06027	3068.0	0
Buildings.2149	POINT (-118.20148393599993 36.80132132800085)	Eastern California Museum	06027	1203.0	0
Buildings.2150	POINT (-118.19926164599991 36.80215464900045)	Inyo County Courthouse	06027	1199.0	0
Buildings.2151	POINT (-118.19898397499992 36.80215464900045)	Inyo County Free Library	06027	1198.0	0
Buildings.2152	POINT (-116.86671957099992 36.46189267800008)	Death Valley National Park Visitor Center	06027	-54.0	0
Buildings.2153	POINT (-120.34241937199984 38.26269494800046)	Independence Hall	06009	1193.0	0
Buildings.2154	POINT (-120.07602104499985 38.43907905200007)	Tamarack Lodge	06009	2107.0	0
Buildings.2155	POINT (-120.52964826099994 38.40074509000004)	West Point Branch Calaveras County Library	06009	830.0	0
Buildings.2156	POINT (-120.5271481659999 38.39546764900007)	West Point Community Hall	06009	829.0	0
Buildings.2157	POINT (-120.35464196499993 38.25491724000054)	Arnold Branch Calaveras County Library	06009	1204.0	0
Buildings.2158	POINT (-120.46242164099993 38.13408701100008)	Black Bart Playhouse	06009	659.0	0
Buildings.2159	POINT (-120.46853301399995 38.13825345900055)	Black Sheep Vinters	06009	658.0	0
Buildings.2160	POINT (-120.45658824499992 38.14519784700008)	Bret Harte Center	06009	583.0	0
Buildings.2161	POINT (-120.46519957999993 38.13742013900085)	Mitchler Hotel	06009	660.0	0
Buildings.2162	POINT (-120.4640884349999 38.13797578800006)	Murphys Branch Calaveras County Library	06009	662.0	0
Buildings.2163	POINT (-120.45964370199994 38.13769796400004)	Native Daughters Hall	06009	663.0	0
Buildings.2164	POINT (-120.46631072499991 38.13797563500009)	Old Timers Museum	06009	663.0	0
Buildings.2165	POINT (-120.4674218699999 38.13797563500009)	Traver Building	06009	662.0	0
Buildings.2166	POINT (-120.54186796399989 38.07131092400045)	Angels Camp Branch Calaveras County Library	06009	437.0	0
Buildings.2167	POINT (-120.54936845999993 38.07658867100008)	Angels Camp City Hall	06009	472.0	0
Buildings.2168	POINT (-120.54742399499992 38.07603302200005)	Angels Camp Museum	06009	467.0	0
Buildings.2169	POINT (-120.68132055299992 38.19602977700035)	Calaveras County Courthouse	06009	307.0	0
Buildings.2170	POINT (-120.66604204299995 38.19102985500007)	Calaveras County Government Center	06009	336.0	0
Buildings.2171	POINT (-120.6804872329999 38.19686309800005)	Calaveras County Historical Museum	06009	302.0	0
Buildings.2172	POINT (-120.66604204299995 38.19102985500007)	Calaveras County Law Library	06009	336.0	0
Buildings.2173	POINT (-120.6804872329999 38.19714092200074)	Calaveras County Library	06009	297.0	0
Buildings.2174	POINT (-120.68076505699992 38.19741874700086)	Metropolitan Cultural Center	06009	296.0	0
Buildings.2175	POINT (-120.55381319299994 38.07908847900006)	Prince Garibardi Building	06009	472.0	0
Buildings.2176	POINT (-120.68132055299992 38.19575195300007)	San Andreas Town Hall	06009	309.0	0
Buildings.2177	POINT (-119.75014407399993 36.81078298300008)	Aspen Hall	06019	102.0	0
Buildings.2178	POINT (-119.75481071499992 36.81078298300008)	Baker Hall	06019	102.0	0

Figure 6-93: MVT raw layer Buildings.shp data table

We will render the features with the ELEV_METER field value by query and highlighting way in Leaflet. This field represents the elevation of the building's location in meters, and the code is as follows:

```
<!DOCTYPE html>
<html lang="ja">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <link rel="stylesheet"
href="//cdnjs.cloudflare.com/ajax/libs/leaflet/0.7.7/leaflet.css"/>
  <style>
    .map {
      width: 100%;
      height: 1100px;
    }
    .info {
        padding: 6px 8px;
        font: 14px/16px Arial, Helvetica, sans-serif;
        background: white;
        background: rgba(255,255,255,0.8);
        box-shadow: 0 0 15px rgba(0,0,0,0.2);
        border-radius: 5px;
```

```
    }
    .info h4 {
        margin: 0 0 5px;
        color: #777;
    }
  </style>
  <script src="//cdnjs.cloudflare.com/ajax/libs/leaflet/0.7.7/leaflet-src.js">
</script>
  <script src="./js/Leaflet.MapboxVectorTile.min.js"></script>
  <title>Leaflet MVT Tiles Example</title>
</head>
<body>
  <div class="map" id="map"></div>
  <script>
      const map = L.map("map", {
        maxZoom: 17
      }).setView([37.627254,-122.365161], 11);
      // Loading ArcGIS online basemap
      const base =
L.tileLayer("https://server.arcgisonline.com/ArcGIS/rest/services/World_Topo_Map/
MapServer/tile/{z}/{y}/{x}").addTo(map);
      // Load MVT layers and construct feature data query criteria for grouping
rendering
      const buildings = new L.TileLayer.MVTSource({
        url: "./data/tiles/mvt3857/Buildings/{z}/{x}/{y}.mvt",
        maxZoom: 18,
        //maxNativeZoom: 15,
        mutexToggle: false,
        onClick: hoveronFeature,
        style: function(feature){
          let color = ((feature.properties.ELEV_METER >= 100) &&
(feature.properties.ELEV_METER <= 300)) ? "#FFFF00" :
                      ((feature.properties.ELEV_METER > 300) &&
(feature.properties.ELEV_METER <= 500)) ? "#FFD700" :
                      ((feature.properties.ELEV_METER > 500) &&
(feature.properties.ELEV_METER <= 700)) ? "#6495ED" :
                      ((feature.properties.ELEV_METER > 700) &&
(feature.properties.ELEV_METER <= 900)) ? "#FF00FF" :
                      ((feature.properties.ELEV_METER > 900) &&
(feature.properties.ELEV_METER <= 1100)) ? "#C71585" :
                      ((feature.properties.ELEV_METER > 1100) &&
(feature.properties.ELEV_METER <= 1300)) ? "#8A2BE2" :
                      ((feature.properties.ELEV_METER > 1300) &&
(feature.properties.ELEV_METER <= 1500)) ? "#7A67EE" :
                      ((feature.properties.ELEV_METER > 1500) &&
(feature.properties.ELEV_METER <= 1700)) ? "#0000FF" :
                      ((feature.properties.ELEV_METER > 1700) &&
(feature.properties.ELEV_METER <= 2000)) ? "#00EE76" :
                       "#0000FF";

          return {color: color, radius: 5, selected: {color: "red", radius: 10}};
        }
      });
      buildings.addTo(map);
      // Building information display components
      const info = L.control();
```

```
        info.onAdd = function (map) {
            this._div = L.DomUtil.create('div', 'info'); // create a div with a
class "info"
            this.update();
            return this._div;
        };
        // Display selected feature attributes
        info.update = function (props) {
            this._div.innerHTML = '<h4>Building Info</h4>' +  (props ?
                '<b>' + props.NAME + '</b>'
                : 'Click circles');
        };
        info.addTo(map);
        function hoveronFeature(e) {
          if(!e.feature) return;
          info.update(e.feature.properties);
        }
    </script>
</body>
</html>
```

The rendering effect of grouping the values of the ELEV_METER field is shown as Figure 6-94.

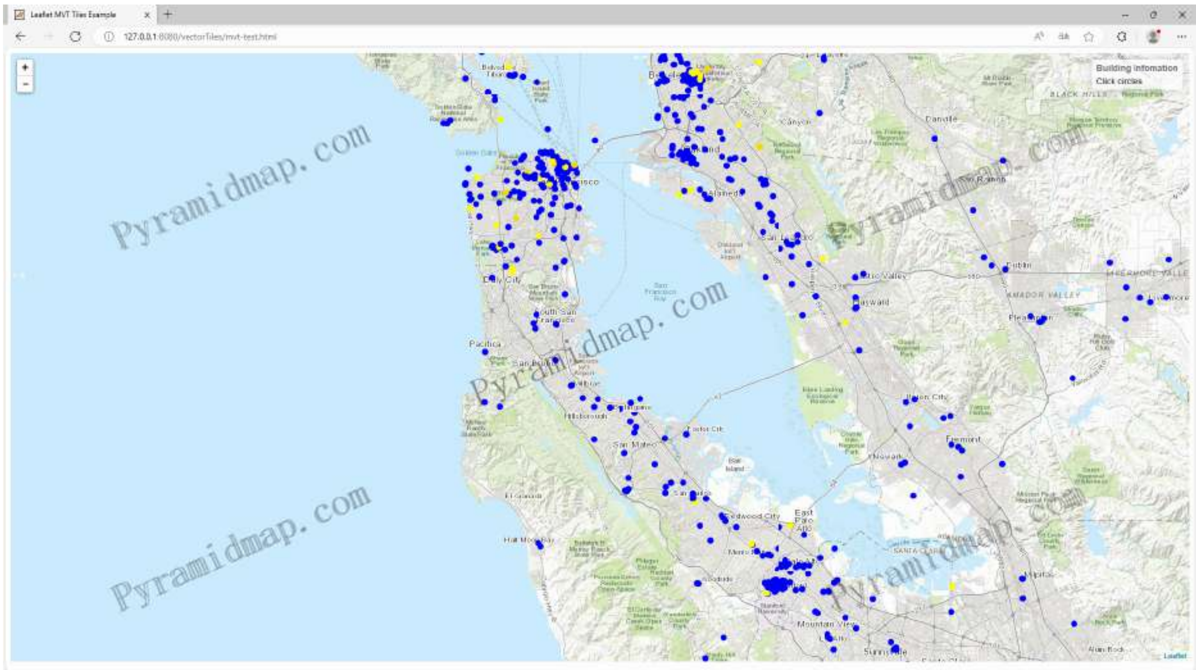


Figure 6-94: Value range grouping rendering and query highlighting for MVT tiles

Select the features, highlight it, the effect is shown as Figure 6-95.

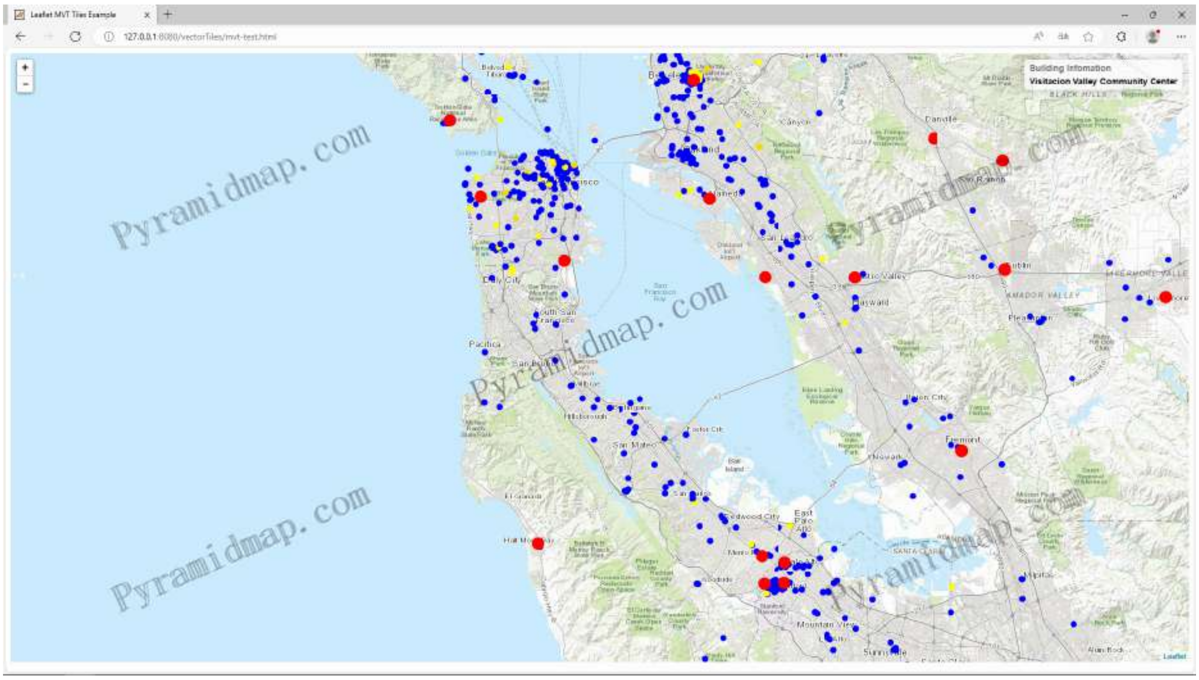


Figure 6-95： highlighting the features selected

Select the feature, display the information, as shown in Figure 6-96.

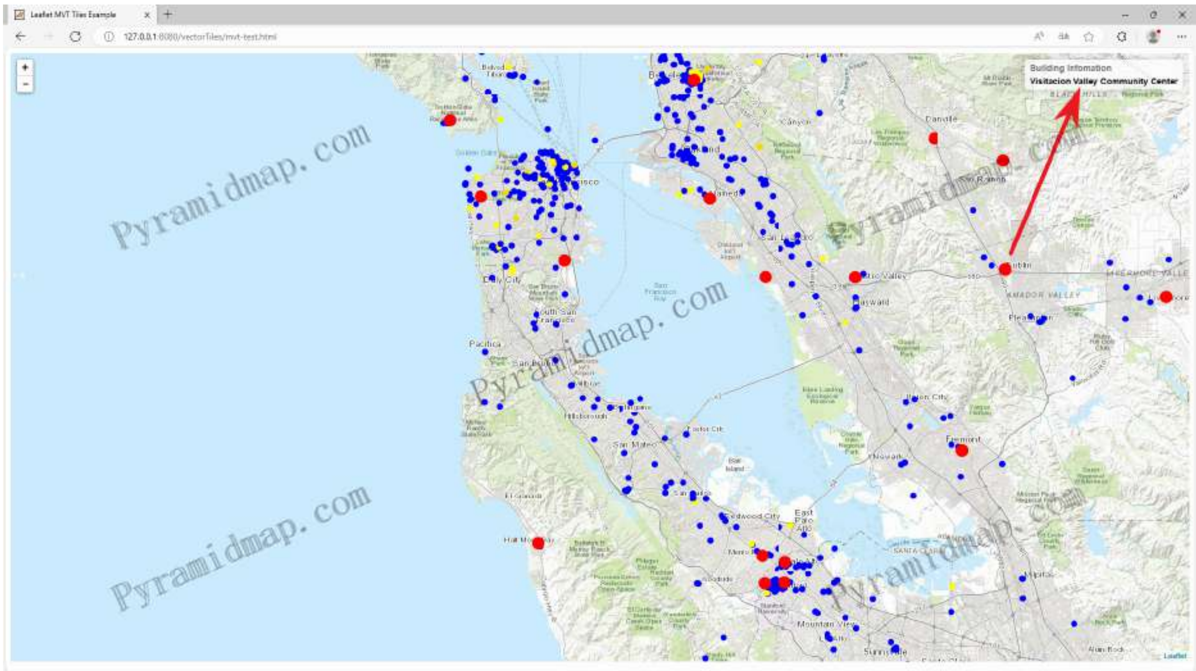


Figure 6-96： Display the information of the feature selected

All the above queries are based on the feature data in the vector tiles. In a sense, vector tile is equivalent to achieving desktop version vector layer visualization on the web, as well as data thematic statistics and rendering based on this.

The above process implements the complete process of MVT vector tile and its application on the web side. From the map, we can see that even when zoomed in to the maximum level (depending on the tile level), the map remains undistorted because the tile data is vector.

So how to determine the application plan of MVT in actual projects? First of all, imagine a scenario where users need to place a map of a certain country on the internet and require it to be accurate to village level administrative areas, roads, and buildings. You have many options to choose from, of course, the best one is definitely vector tile. Firstly, we exclude WMS and WFS, which are relatively inefficient. Which should be choosed between WMTS and vector tile? Although both are tile, the basic data of WMTS is images and does not support operations on layer's feature. Therefore, the vector tile is the optimal and almost the only choice in the large-scale map applications involving map feature data, and the MVT in Mapbox specification is almost the first choice.

It should be noted that although PyramidMap supports tiling operations in all standard coordinate systems, most WebGIS platforms currently use the web Mercator 3857 coordinate system. Therefore, in general, the Web Mercator 3857 coordinate system should be chosen for tile.

6.8.7 Build Vector MBTile tile

MBTiles (Mapbox tiles), as the name suggests, this is an open source tile standard created by Mapbox, with the goal of promoting standardization and efficiency of the tiling. MBTiles supports both vector and raster tiles as well as interactive grid tiles, MBTiles using Web Mercator projection to describe tile coordinate data through metadata, including boundaries, longitude and latitude coordinates, etc. MBTiles internally grades tiles, essentially a map tile dataset based on SQLite, which improves the efficiency of tile retrieval through database indexing, much higher than folder mode tiles. You can build yourself vector MbTiles in PyramidMap, as shown in Figures 6-97.

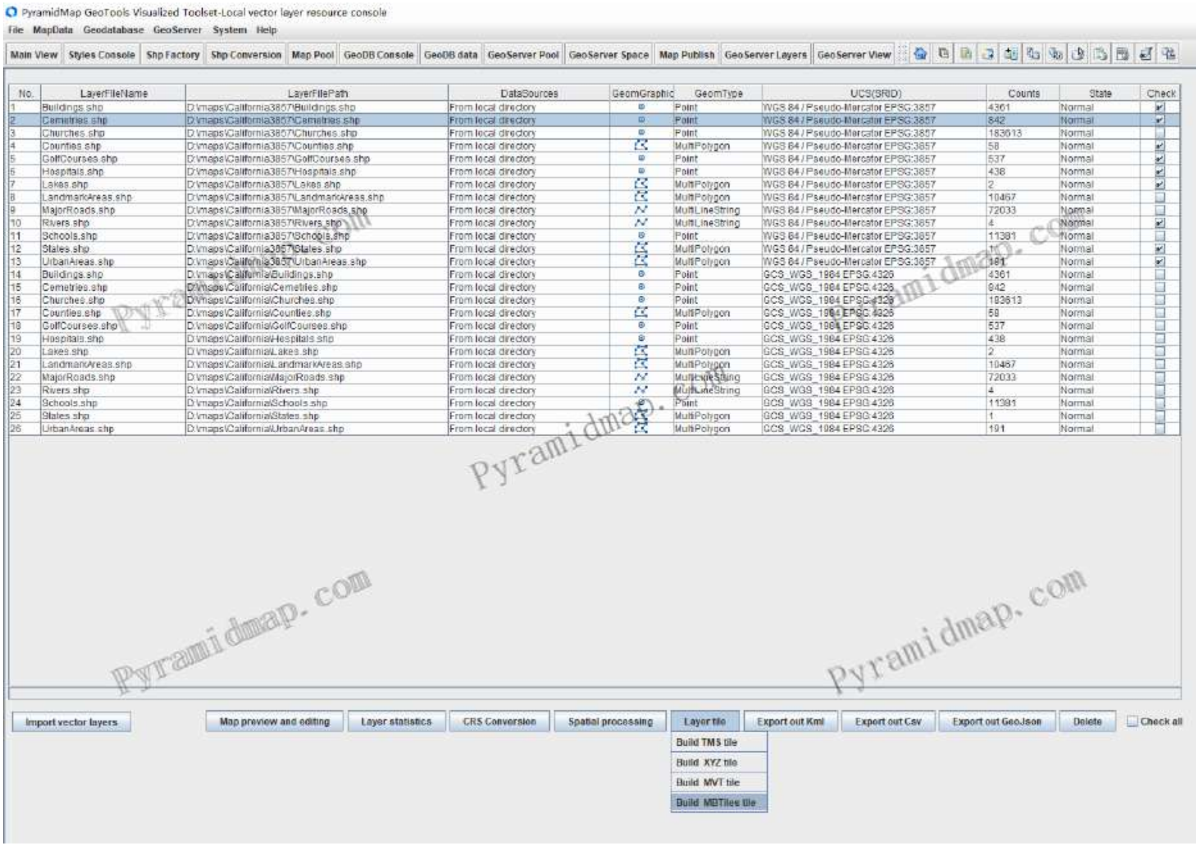


Figure 6-97: Build vector MbTiles in PyramidMap

Select the MBTiles tiles option and then open the tile scheme interface, as shown in Figure 6-98.

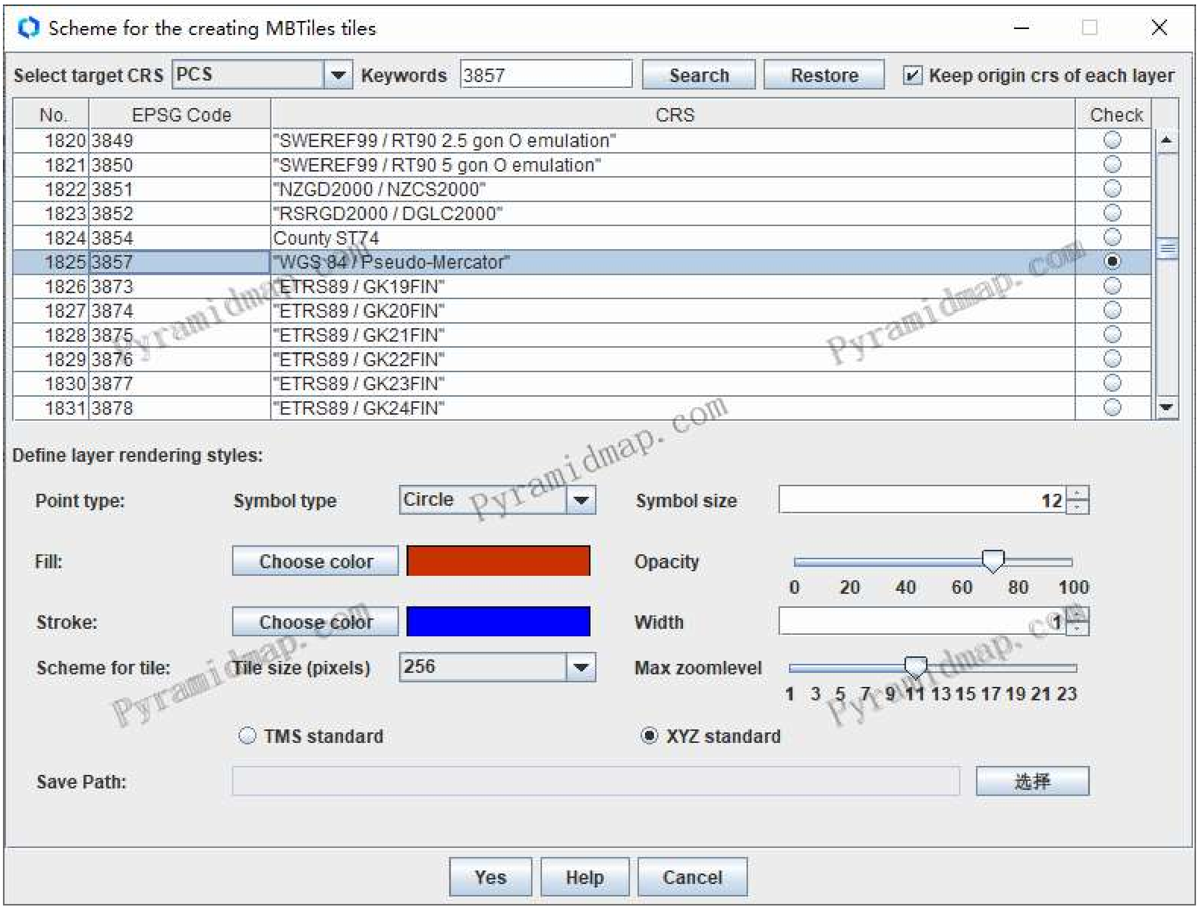


Figure 6-98: Vector MbTiles tile scheme

In the tiling scheme, select the target CRS(coordinate system) for the target tile dataset. The default option is to keep the original CRS of each layer indicated by the checkbox. If you want to transform to another CRS for the target tiles, you should uncheck it and select the desired target coordinate system. The tile operation will automatically convert the coordinate system based on your choosen. PyramidMap provides the vast majority of standard coordinate system and is still being continuously updated. It should be noted that although PyramidMap supports tiling operations in all standard coordinate systems, most WebGIS platforms currently use the Web Mercator 3857 coordinate system. Therefore, in general, the Web Mercator 3857 coordinate system is chosen for tile operations.

The rendering scheme defines the rendering mode of the layer and supports rendering configurations for all collection types such as Point/MultiPoint, Linestring/MultiLinestring, Polygon/MultiPolygon, including point type symbols, size, stroke width, stroke and fill colors, and the opacity.

The scheme defines the tile size and the maximum scaling level. The tile size is measured in pixels, representing the size of each target tile.

The MbTiles scheme supports both TMS and XYZ standards, you can choose as needing.

Select the tile level and target save path, the tile scale can be supported up to 24 zoomlevel.

It should be noted that tile process is a work that consumes system resources and requires high system performance. The tiling speed depends on system performance. Therefore, it is strongly recommended to run the tile process on the workstation level computers.

The tiling process is shown in Figures 6-99.

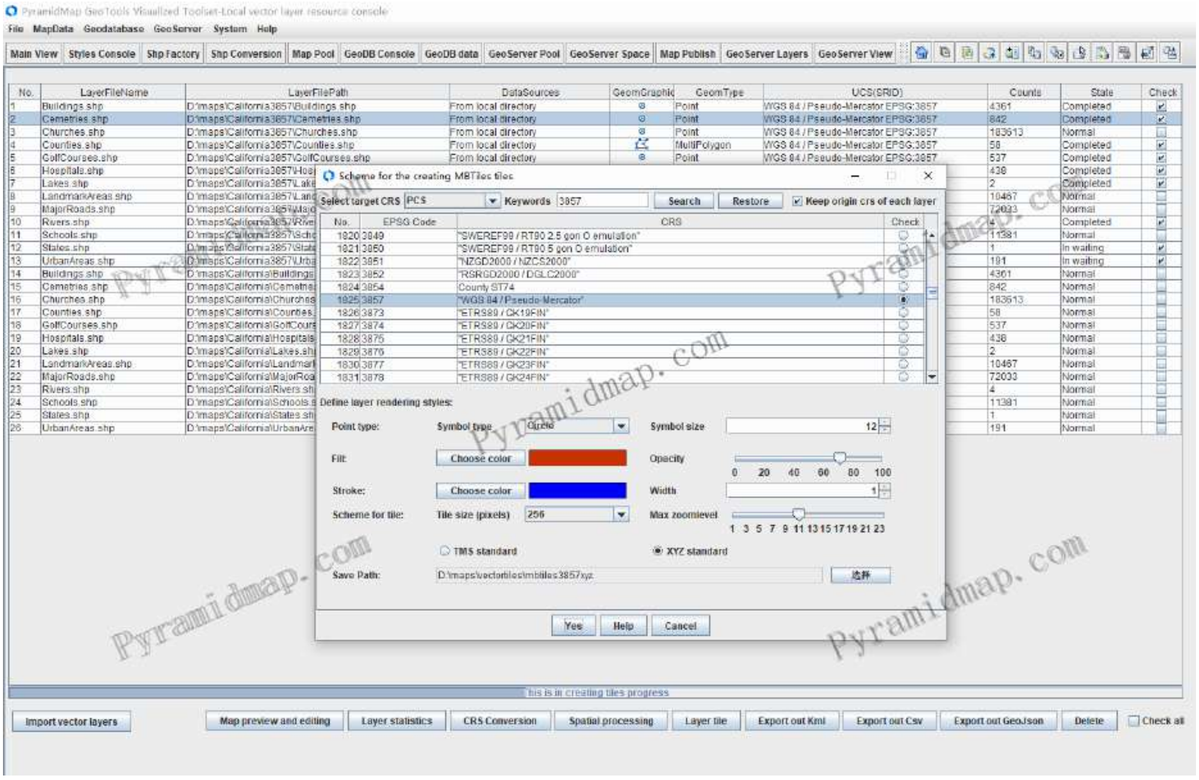


Figure 6-99: Vector MbTiles tile process

The tiling process has a progress bar and the tiled status displayed for each layer and the tiling completion prompt is shown in Figure 6-100.

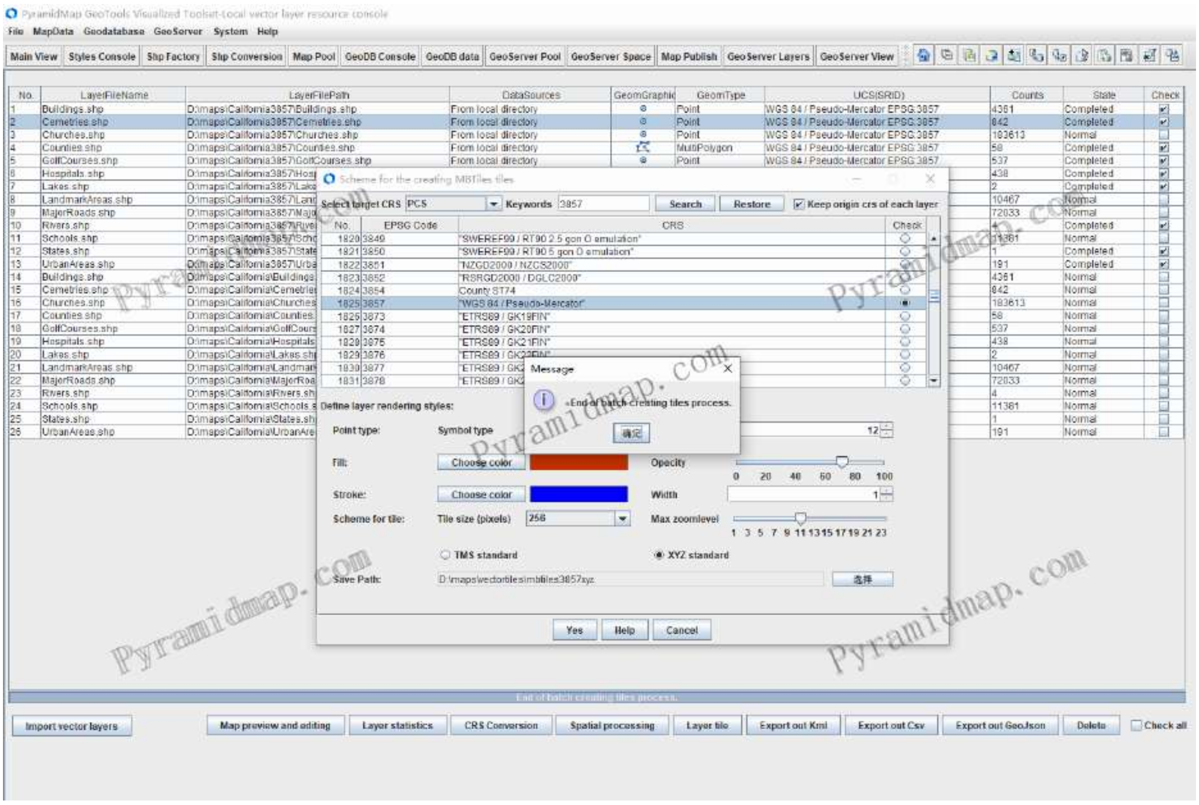


Figure 6-100: Vector MbTiles tiling completed

The generated target tile files is shown as Figure 6-101.

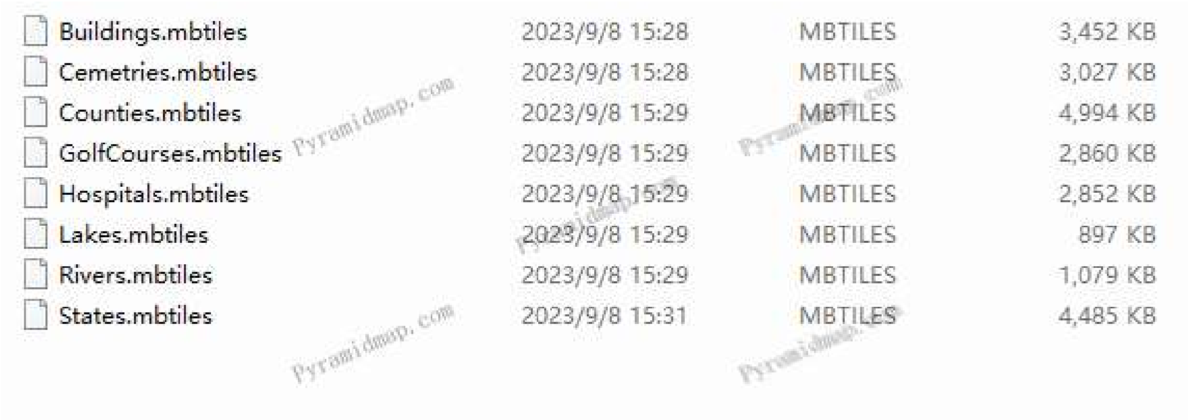


Figure 6-101: Vector MbTiles files

The tile file corresponds to the original layer name, and each vector layer corresponds to its MBtiles file. MBtiles can be freely deployed to web servers, such as Tomcat, without needing professional map servers such as GeoServer, therefore greatly reducing the difficulty of map deployment and maintenance as well as project costs. You can deploy independently, access data through HTTP URLs, or integrate deployment with projects, and access data through relative paths. Taking Leaflet as an example to implement web side MBtiles loading, the complete code is as follows:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Leaflet vector MBTiles Example</title>
    <link rel="stylesheet"
href="https://unpkg.com/leaflet@1.7.1/dist/leaflet.css" />
</head>
<style type="text/css">
    body {
        margin: 0;
        padding: 0;
    }
    html, body, #map{
        width: 100%;
        height: 100%;
    }
</style>
<body>

<div id="map" ></div>

<script src="https://unpkg.com/leaflet@1.7.1/dist/leaflet.js"></script>
<script src="https://unpkg.com/leaflet@1.7.1/dist/leaflet.js"></script>
<script src="https://unpkg.com/sql.js@0.3.2/js/sql.js"></script>
<script
src="https://unpkg.com/Leaflet.TileLayer.MBTiles@1.0.0/Leaflet.TileLayer.MBTiles.
js"></script>
<script>
    var map = L.map('map').fitWorld();
    // Loading ArcGIS online basemap
```

```
L.tileLayer('https://server.arcgisonline.com/ArcGIS/rest/services/World_Topo_Map/
MapServer/tile/{z}/{y}/{x}').addTo(map);
    // Load local MBTiles, please modify according to your file path.
    // Create Counties mbtiles layer
    var mb_Counties =
L.tileLayer.mbTiles('./data/tiles/mbtiles3857xyz/Counties.mbtiles');
    // Create Hospitals mbtiles layer
    var mb_Hospitals =
L.tileLayer.mbTiles('./data/tiles/mbtiles3857xyz/Hospitals.mbtiles');
    // Add Counties mbtiles layer to map
    mb_Counties.addTo(map);
    // Add Hospitals mbtiles layer to map
    mb_Hospitals.addTo(map);
</script>

</body>
</html>
```

The Counties mbtiles and Hospitals mbtiles were added to the map in the Leaflet engine as shown in Figures 6-102.

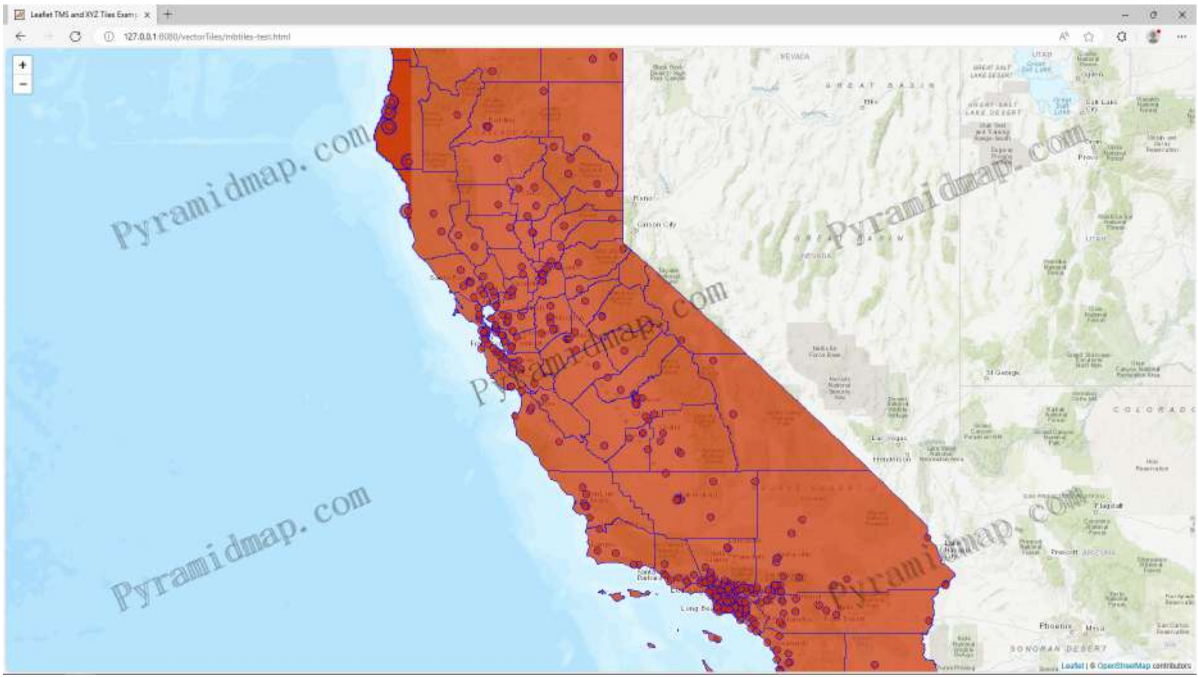


Figure 6-102: loading MbTiles in Leaflet

It should be noted that although PyramidMap supports tiling operations in all standard coordinate systems, most WebGIS platforms currently use the web Mercator 3857 coordinate system. Therefore, in general, the Web Mercator 3857 coordinate system should be chosen for tile.

6.8.8 Build Raster TMS tile

PyramidMap supports TMS, XYZ, and MbTiles standard tile of raster layers, as shown in Figure 6-103.

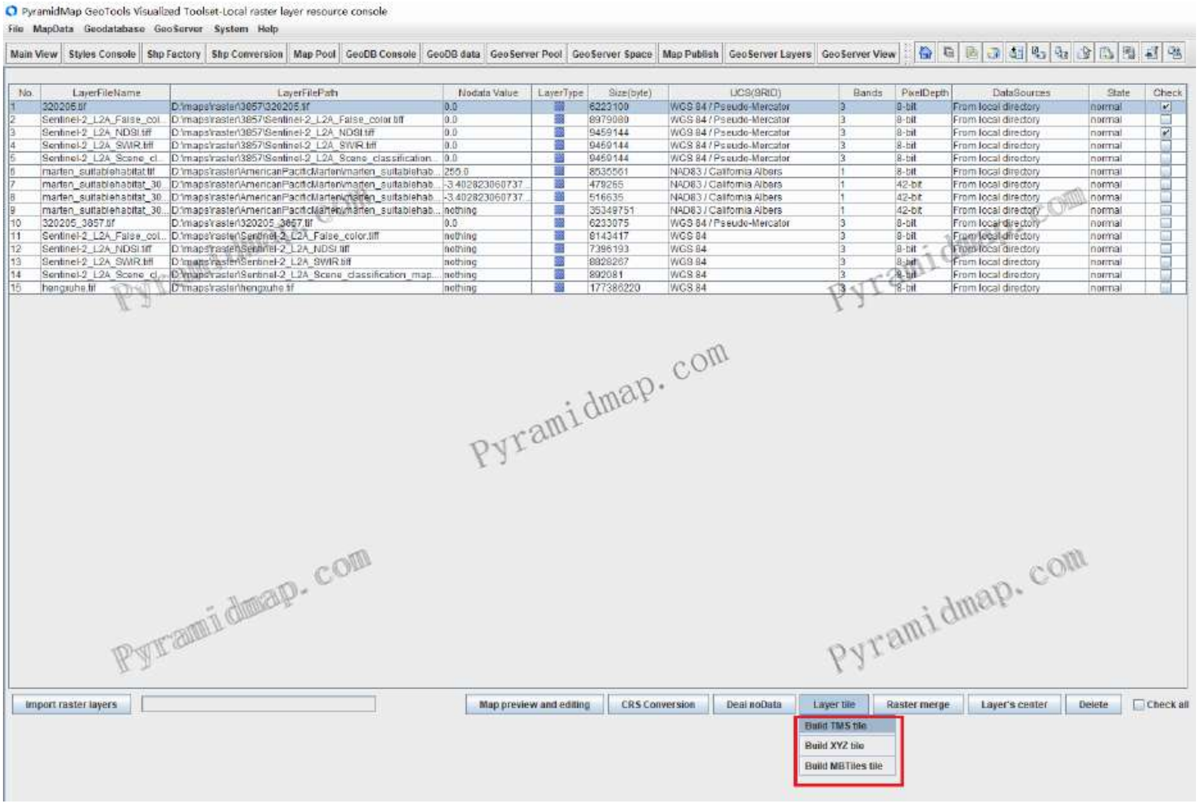


Figure 6-103: PyramidMap raster layer TMS tile

The TMS tile standard has been indentified in the section [6.8.4 Vector TMS tile].

Select the TMS tile and then open the scheme interface, as shown in Figure 6-104.

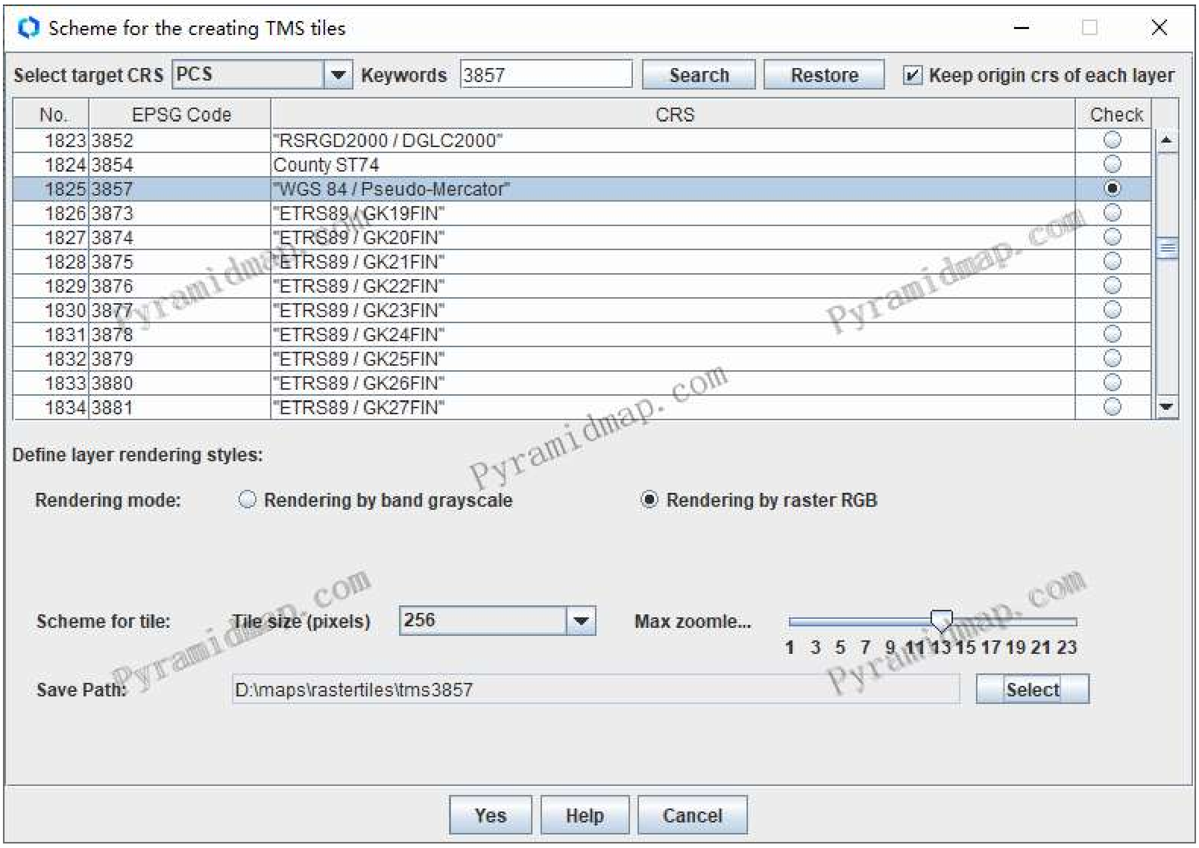


Figure 6-104： Raster TMS tile scheme

In the tiling scheme, select the target coordinate system, which is the coordinate system of the target tile dataset. The default option is to keep the original coordinate system of each layer. If need to change it, you can uncheck it and select the desired target coordinate system. The tile operation will automatically convert the coordinate system based on your selection. PyramidMap provides the vast majority of standard coordinate system and is still being continuously updated. It should be noted that although PyramidMap supports tiling operations in all standard coordinate systems, most WebGIS platforms currently use the Web Mercator 3857 coordinate system. Therefore, in general, the Web Mercator 3857 coordinate system is chosen for tile operations.

Please choose the rendering mode based on the band of the raster layer. For single band layers, please choose to render according to grayscale, and for multi band layers, please choose to render according to RGB raster.

The scheme defines the tile size and the maximum scaling level. The tile size is measured in pixels, representing the size of each target tile.

Select the tile level and target save path, the tile scale can be supported up to 24 zoomlevel.

It should be noted that tile process is a work that consumes system resources and requires high system performance. The tiling speed depends on system performance. Therefore, it is strongly recommended to run the tile process on the workstation level computers.

The execution process is shown in Figures 6-105.

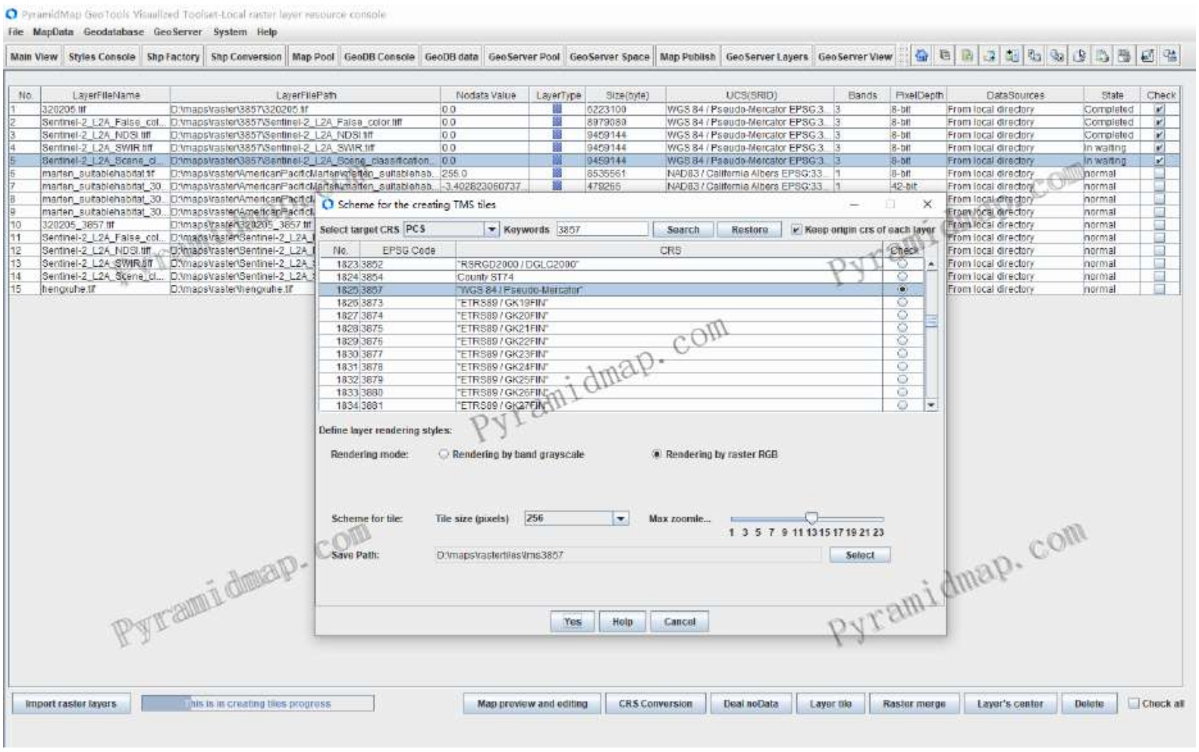


Figure 6-105: Raster TMS tiling process

The progress bar indicate the tiling process and status of each selected layer. The tiling completion prompt is shown in Figure 6-106.

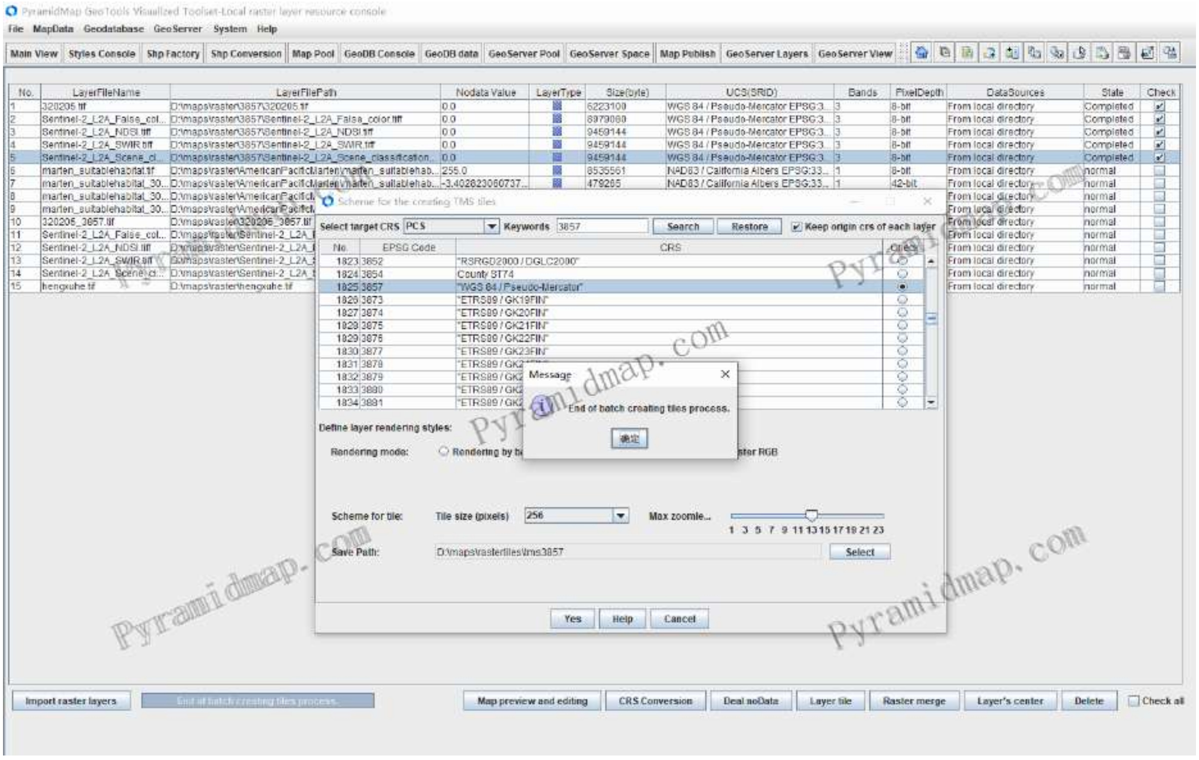


Figure 6-106: Raster TMS tiling completed

The final generated tiles will be classified according to the original layer name, grouped according to the Z value (zoom level) and X value, named according to the Y value, and saved to the corresponding hierarchical directory. At the same level, the tiles will be grouped based on the X value. The tiles with the same X value will be saved to the same path which named with the X value, and the tiles will be named with the Y value. The directory structure is shown as Figure 6-107.

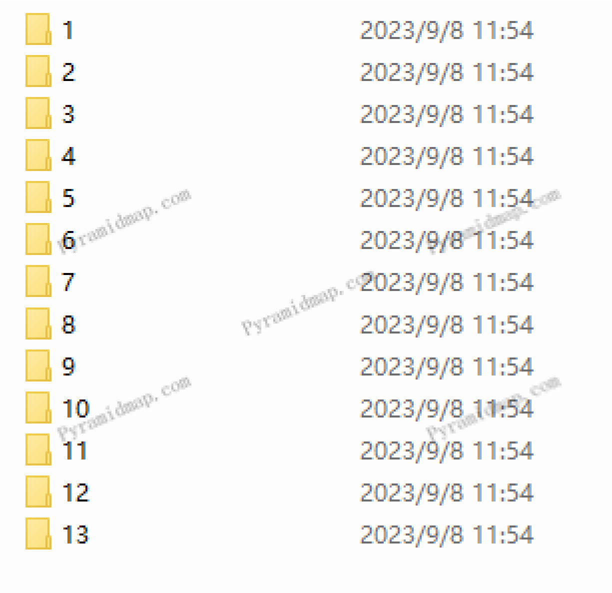


Figure 6-107: TMS tiles Folder

View the slicing file at a certain level, as shown in Figure 6-108.

Figure 6-108: Raster TMS tile files

When storing tiles, the code uses a folder structure that follows the TMS (Tile Map Service) standard, namely Z/X/Y.png, where Z represents the zoom level and X and Y represent the coordinates of the tiles. During this process, the code also checks and creates necessary folders. After completing the tiling, deploy the tiles to a web server, such as Apache Tomcat, nothing with the professional mapserver such as GeoServer, that will greatly reduce the difficulty and cost of map deployment and maintenance. You can deploy independently, access tiles data through HTTP URLs, or integrate deployment with projects, and access tiles data through relative paths. As an example, we demonstrate the web gis application through Leaflet, the complete code is as follows:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Leaflet raster TMS tiles Example</title>
    <link rel="stylesheet"
href="https://unpkg.com/leaflet@1.7.1/dist/leaflet.css" />
</head>
<style type="text/css">
    body {
        margin: 0;
        padding: 0;
    }
    html, body, #map{
        width: 100%;
        height: 100%;
    }
</style>
<body>

<div id="map" ></div>

<script src="https://unpkg.com/leaflet@1.7.1/dist/leaflet.js"></script>
<script>
    var map = L.map('map').setView([31.562710059362658,120.29751401540051], 8);
// 初始化地图并设置中心和缩放级别
    // Loading ArcGIS online basemap
```

```
L.tileLayer('https://server.arcgisonline.com/ArcGIS/rest/services/World_Topo_Map/
MapServer/tile/{z}/{y}/{x}').addTo(map);
    // Load local TMS tiles, please modify according to your file path.
    L.tileLayer('.data/tiles/tms3857/{z}/{x}/{y}.png', {
       tms: true,  // Indicates this is an TMS tile
       opacity: 0.7 // The transparency of the tile can be adjusted as needed
    }).addTo(map);
</script>

</body>
</html>
```

The loading effect on the map is shown in Figures 6-109.

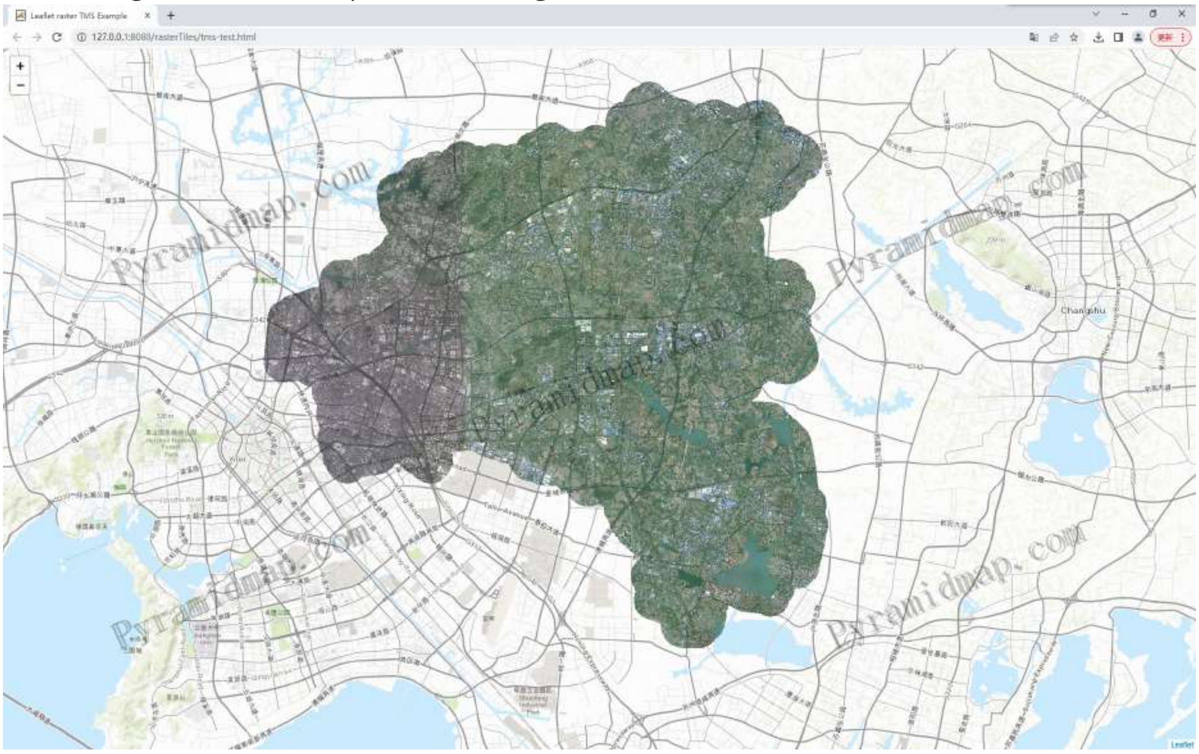


Figure 6-109: Loading raster TMS tiles in Leaflet

It should be noted that although PyramidMap supports tiling operations in all standard coordinate systems, most WebGIS platforms currently use the web Mercator 3857 coordinate system. Therefore, in general, the Web Mercator 3857 coordinate system should be chosen for tile.

6.8.9 Build Raster XYZ Tile

PyramidMap supports TMS, XYZ, and MbTiles standard tile of raster layers, as shown in Figure 6-110.

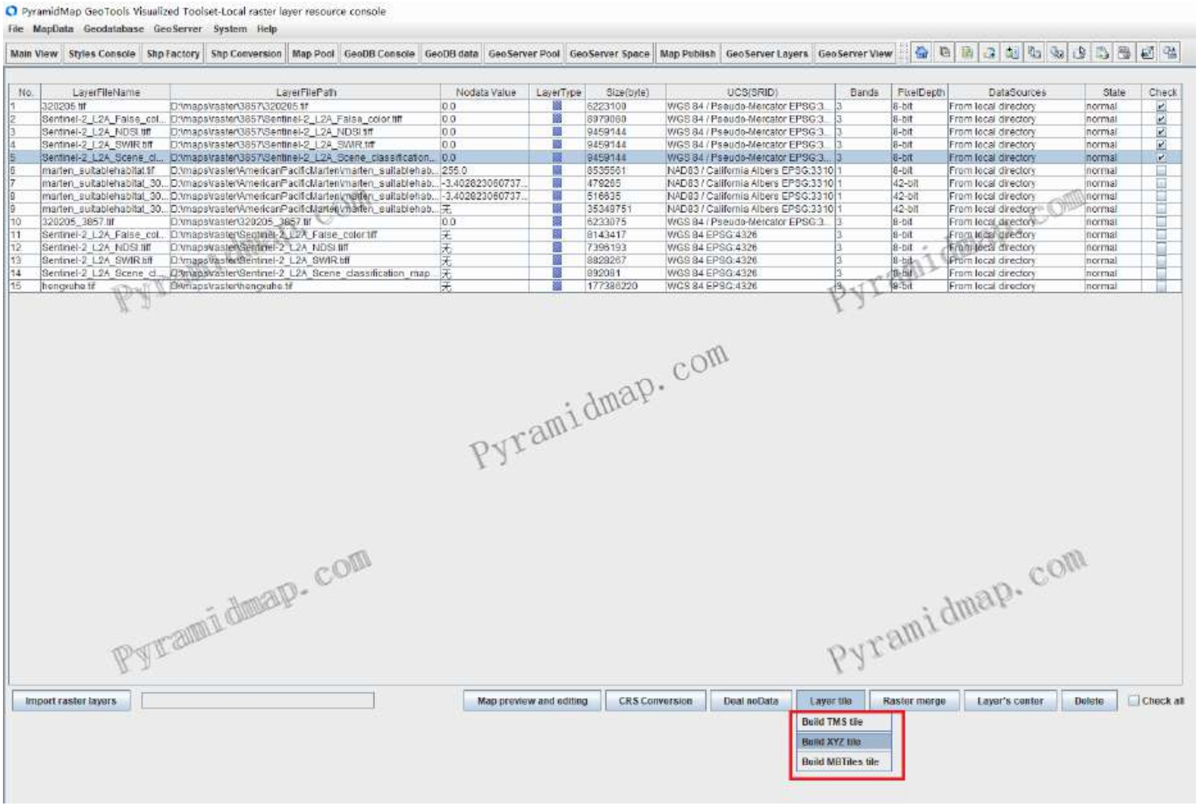


Figure 6-110: PyramidMap raster layer XYZ tile

The XYZ tile standard has been indentified in the section [6.8.5 Vector XYZ tile].

Select the XYZ tile and then open the scheme interface, as shown in Figure 6-111.

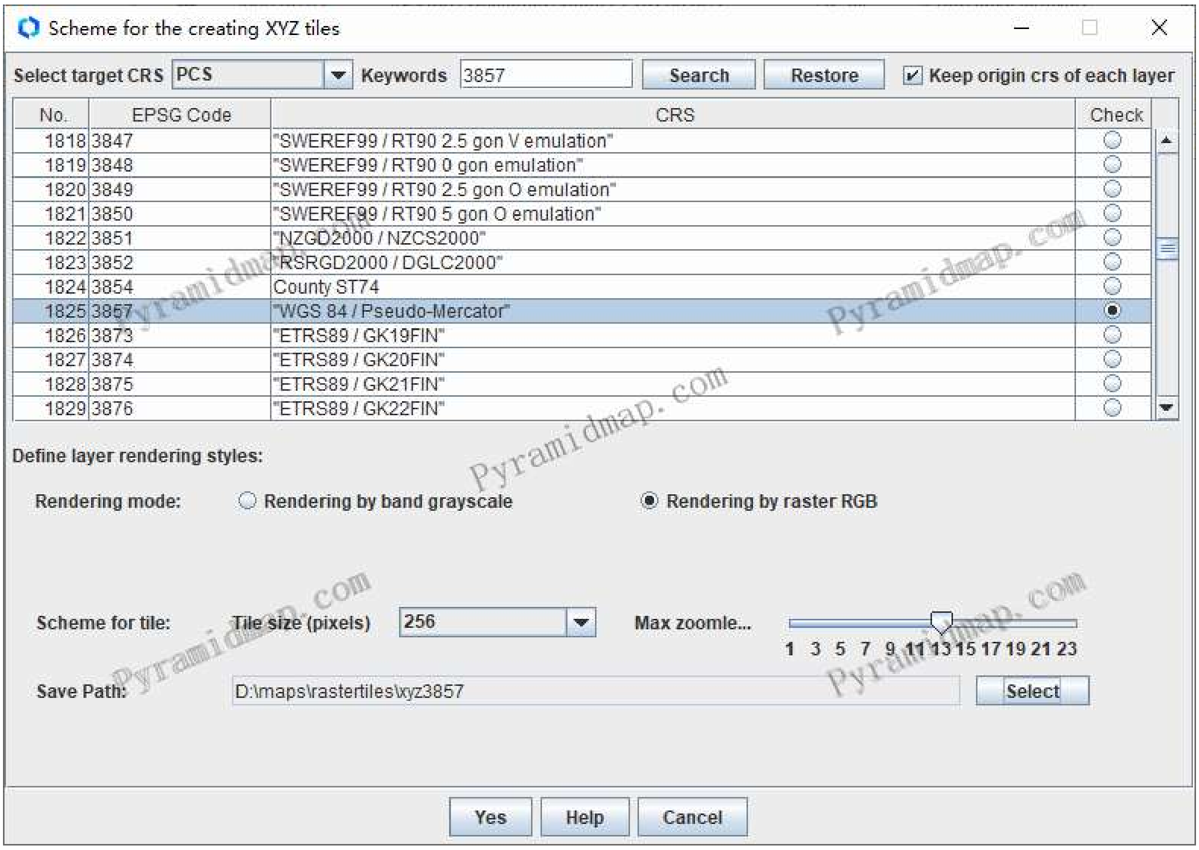


Figure 6-111： Raster TMS tile scheme

In the tiling scheme, select the target coordinate system, which is the coordinate system of the target tile dataset. The default option is to keep the original coordinate system of each layer. If need to change it, you can uncheck it and select the desired target coordinate system. The tile operation will automatically convert the coordinate system based on your selection. PyramidMap provides the vast majority of standard coordinate system and is still being continuously updated. It should be noted that although PyramidMap supports tiling operations in all standard coordinate systems, most WebGIS platforms currently use the Web Mercator 3857 coordinate system. Therefore, in general, the Web Mercator 3857 coordinate system is chosen for tile operations.

Please choose the rendering mode based on the band of the raster layer. For single band layers, please choose to render according to grayscale, and for multi band layers, please choose to render according to RGB raster.

The scheme defines the tile size and the maximum scaling level. The tile size is measured in pixels, representing the size of each target tile.

Select the tile level and target save path, the tile scale can be supported up to 24 zoomlevel.

It should be noted that tile process is a work that consumes system resources and requires high system performance. The tiling speed depends on system performance. Therefore, it is strongly recommended to run the tile process on the workstation level computers.

The execution process is shown in Figures 6-112.

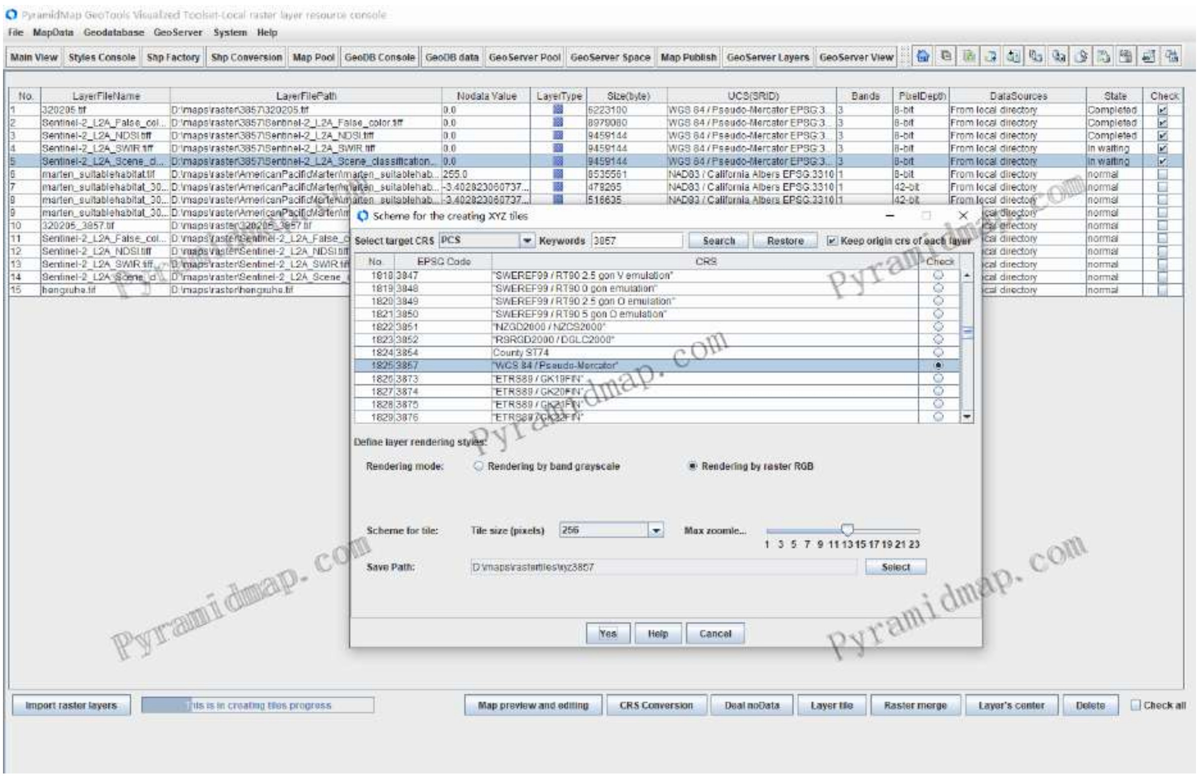


Figure 6-112: Raster TMS tiling process

The progress bar indicate the tiling process and status of each selected layer. The tiling completion prompt is shown in Figure 6-113.

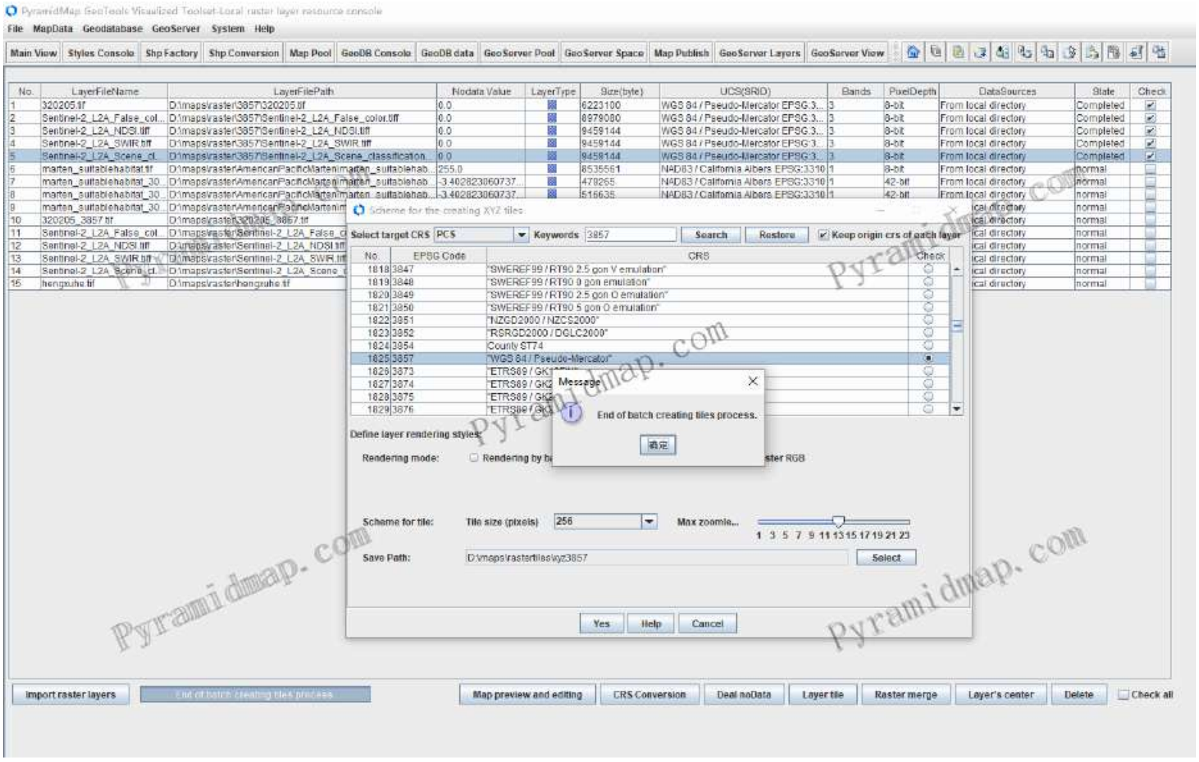


Figure 6-113: Raster XYZ tiling completed

The final generated tiles will be classified according to the original layer name, grouped according to the Z value (zoom level) and X value, named according to the Y value, and saved to the corresponding hierarchical directory. At the same level, the tiles will be grouped based on the X value. The tiles with the same X value will be saved to the same path which named with the X value, and the tiles will be named with the Y value. The directory structure is shown as Figure 6-114.

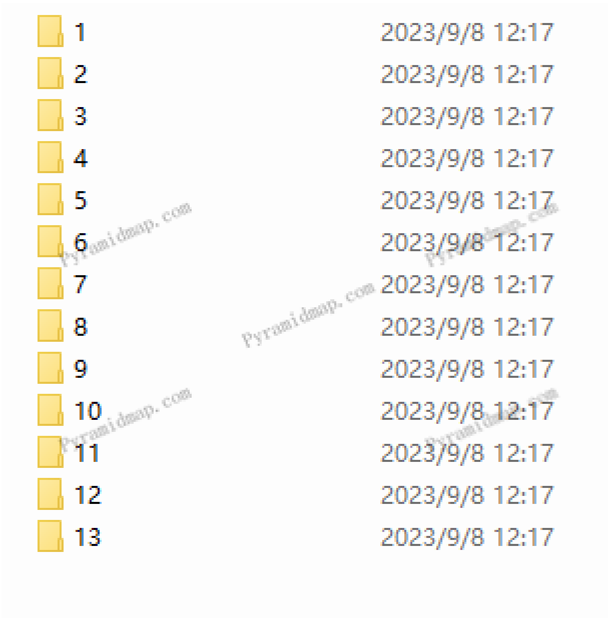


Figure 6-114: XYZ TILE Folder

View the slicing file at a certain level, as shown in Figure 6-115.

Figure 6-108: Raster XYZ tile files

When storing tiles, the code uses a folder structure that follows the TMS (Tile Map Service) standard, namely Z/X/Y.png, where Z represents the zoom level and X and Y represent the coordinates of the tiles. During this process, the code also checks and creates necessary folders. After completing the tiling, deploy the tiles to a web server, such as Apache Tomcat, nothing with the professional mapserver such as GeoServer, that will greatly reduce the difficulty and cost of map deployment and maintenance. You can deploy independently, access tiles data through HTTP URLs, or integrate deployment with projects, and access tiles data through relative paths. As an example, we demonstrate the web gis application through Leaflet, the complete code is as follows:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Leaflet raster XYZ tiles Example</title>
    <link rel="stylesheet"
href="https://unpkg.com/leaflet@1.7.1/dist/leaflet.css" />
</head>
<style type="text/css">
    body {
        margin: 0;
        padding: 0;
    }
    html, body, #map{
        width: 100%;
        height: 100%;
    }
</style>
<body>

<div id="map" ></div>

<script src="https://unpkg.com/leaflet@1.7.1/dist/leaflet.js"></script>
<script>
    var map = L.map('map').setView([31.562710059362658,120.29751401540051], 8);
// 初始化地图并设置中心和缩放级别
```

```
    // Loading ArcGIS online basemap

L.tileLayer('https://server.arcgisonline.com/ArcGIS/rest/services/World_Topo_Map/
MapServer/tile/{z}/{y}/{x}').addTo(map);
    // Load local XYZ tiles, please modify according to your file path.
    L.tileLayer('.data/tiles/xyz3857/{z}/{x}/{y}.png', {
       tms: false,  // Indicates this is an XYZ tile
       opacity: 0.7 // The transparency of the tile can be adjusted as needed
    }).addTo(map);
</script>

</body>
</html>
```

The loading effect on the map is shown in Figures 6-116.

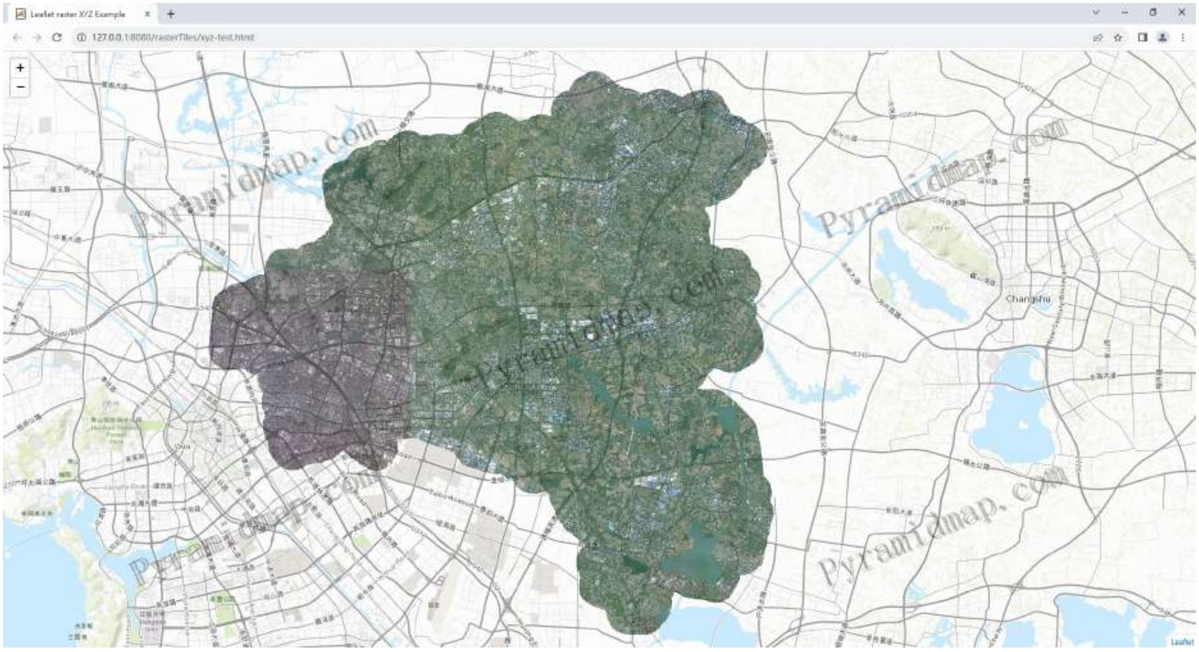


Figure 6-116: Loading raster TMS tiles in Leaflet

It should be noted that although PyramidMap supports tiling operations in all standard coordinate systems, most WebGIS platforms currently use the web Mercator 3857 coordinate system. Therefore, in general, the Web Mercator 3857 coordinate system should be chosen for tile.

6.8.10 Build Raster MBTiles

MBTiles (Mapbox tiles), as the name suggests, this is an open source tile standard created by Mapbox, with the goal of promoting standardization and efficiency of the tiling. MBTiles supports both vector and raster tiles as well as interactive grid tiles, MBTiles using Web Mercator projection to describe tile coordinate data through metadata, including boundaries, longitude and latitude coordinates, etc. MBTiles internally grades tiles, essentially a map tile dataset based on SQLite, which improves the efficiency of tile retrieval through database indexing, much higher than folder mode tiles. You can build yourself raster MbTiles in PyramidMap, as shown in Figures 6-117.

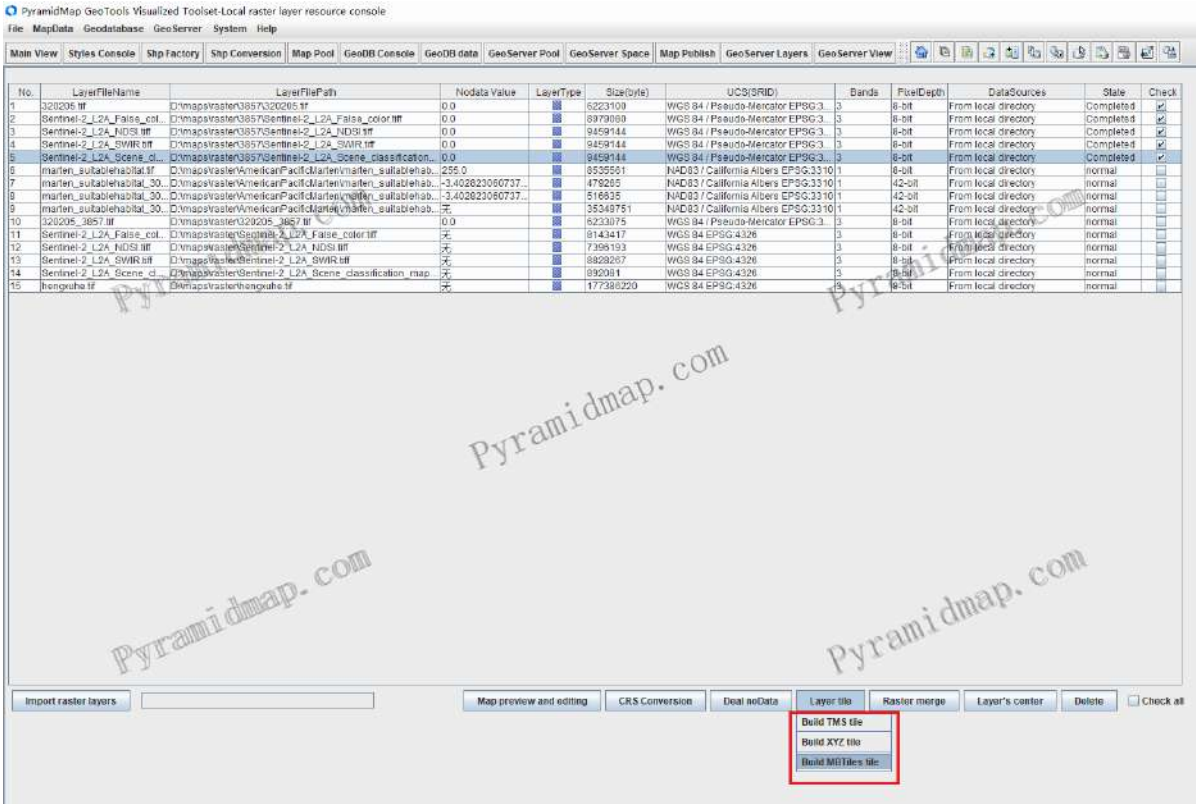


Figure 6-97: Build raster MbTiles in PyramidMap

Select the MBTiles tile and then open the tile scheme interface, as shown in Figure 6-118.

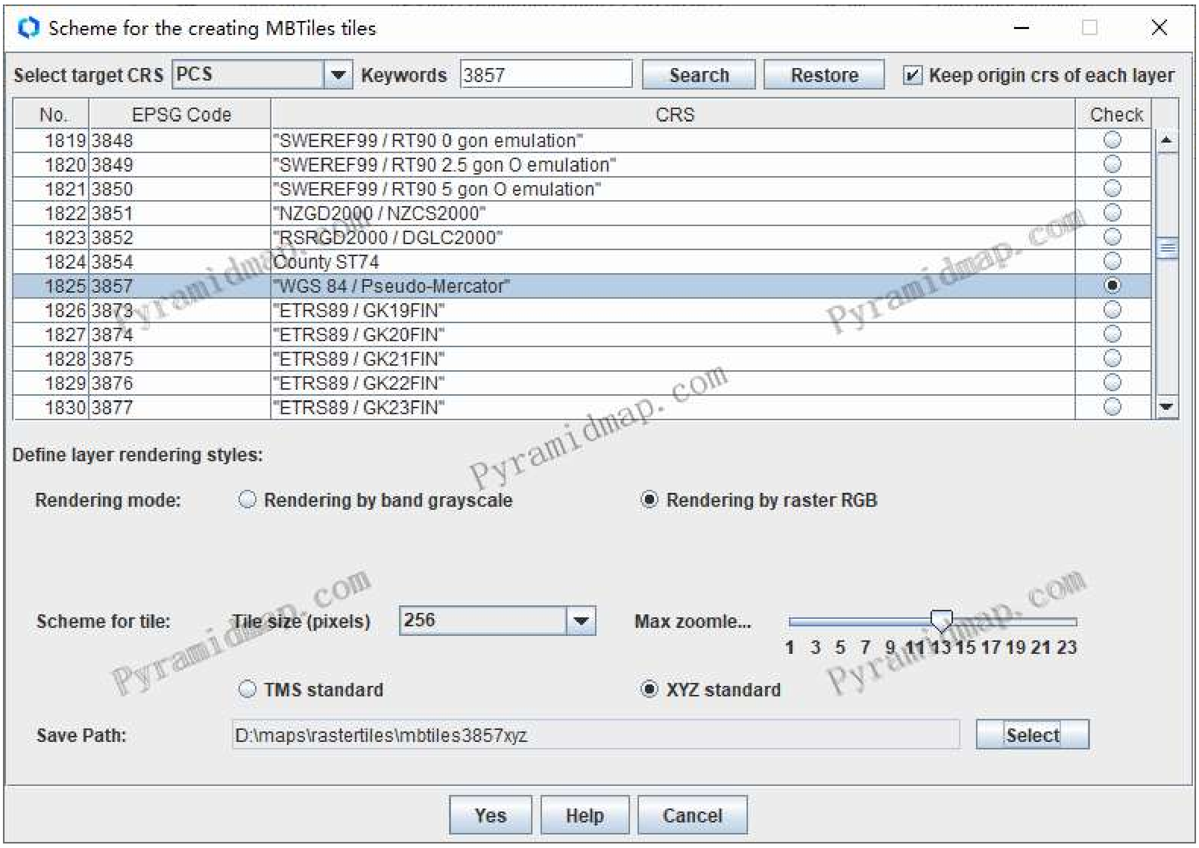


Figure 6-118: Raster MBTile tile scheme

In the tiling scheme, select the target coordinate system, which is the coordinate system of the target tile dataset. The default option is to keep the original coordinate system of each layer. If need to change it, you can uncheck it and select the desired target coordinate system. The tile operation will automatically convert the coordinate system based on your selection. PyramidMap provides the vast majority of standard coordinate system and is still being continuously updated.

It should be noted that although PyramidMap supports tiling operations in all standard coordinate systems, most WebGIS platforms currently use the Web Mercator 3857 coordinate system. Therefore, in general, the Web Mercator 3857 coordinate system is chosen for tile operations.

Please choose the rendering mode based on the band of the raster layer. For single band layers, please choose to render according to grayscale, and for multi band layers, please choose to render according to RGB raster.

The scheme defines the tile size and the maximum scaling level. The tile size is measured in pixels, representing the size of each target tile.

The MbTiles scheme supports both TMS and XYZ standards, you can choose as needing.

Select the tile level and target save path, the tile scale can be supported up to 24 zoomlevel.

It should be noted that tile process is a work that consumes system resources and requires high system performance. The tiling speed depends on system performance. Therefore, it is strongly recommended to run the tile process on the workstation level computers.

The execution process is shown in Figures 6-119.

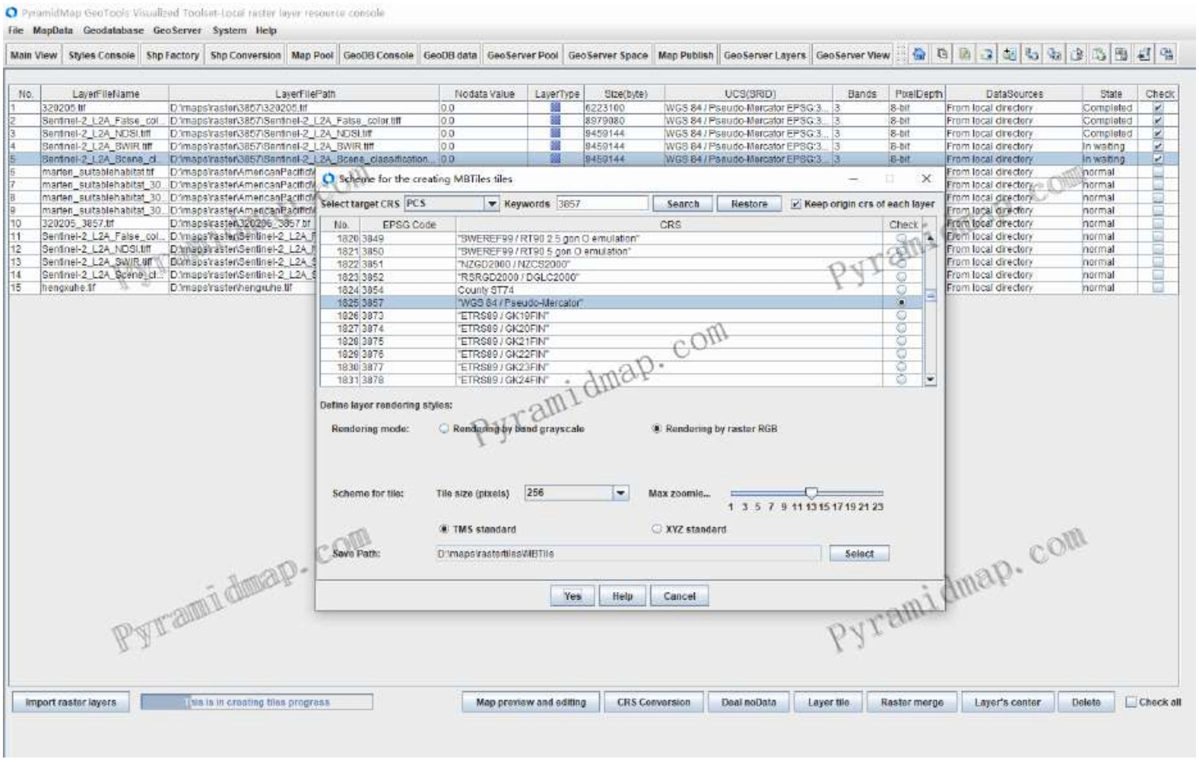


Figure 6-119: Raster MBtiles tiling process

The progress bar indicate the tiling process and status of each selected layer. The tiling completion prompt is shown in Figure 6-120.

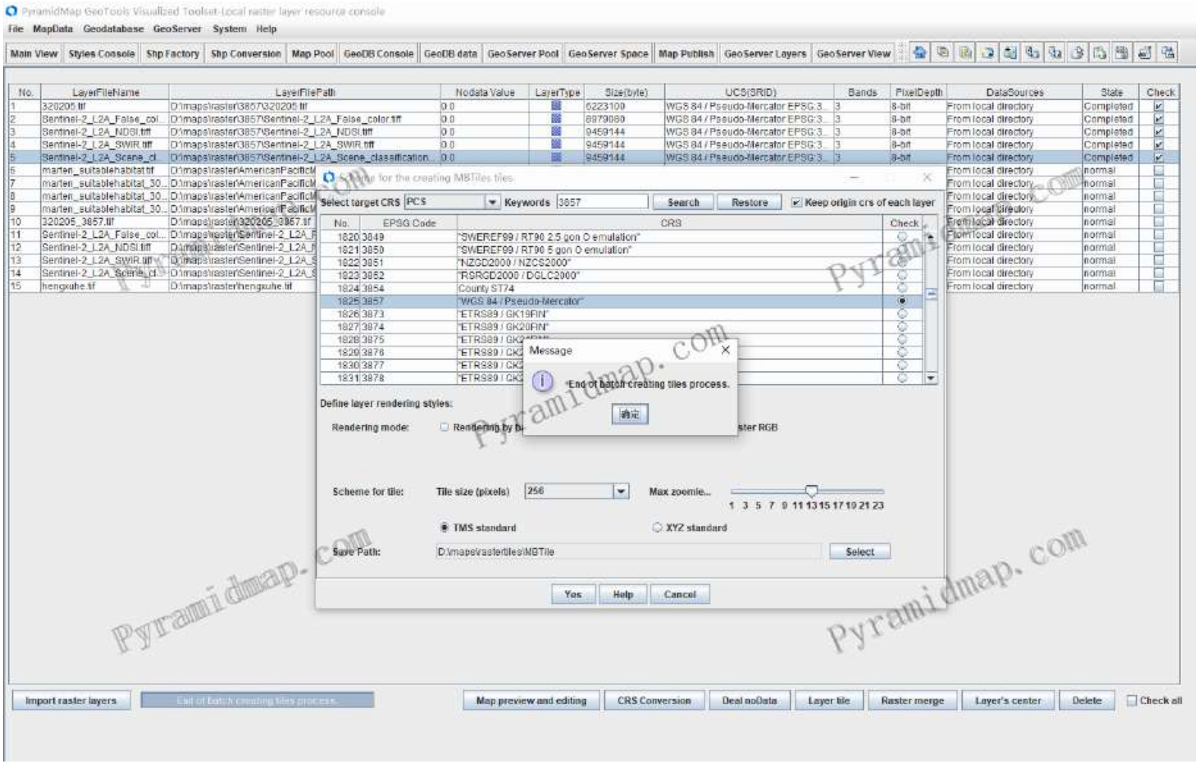


Figure 6-120: Raster MBtiles tiling completed

The generated target MBTtiles files is shown as Figure 6-121.

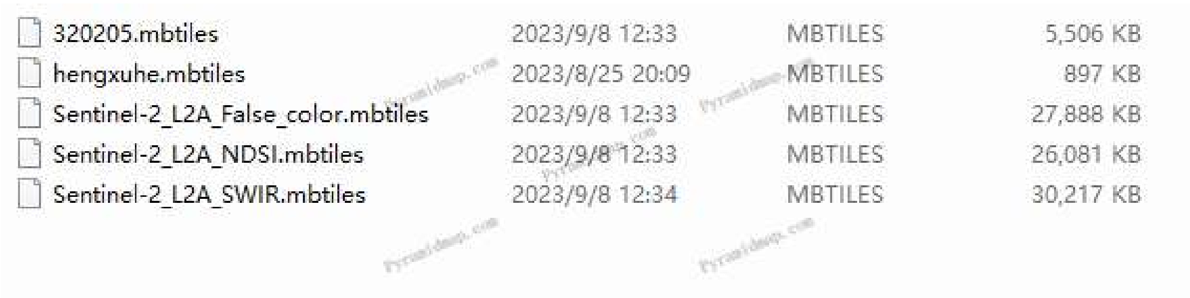


Figure 6-121: The target raster MbTiles files

The tile file corresponds to the original layer name, and each raster layer corresponds to its MBtiles file. MBtiles can be freely deployed to web servers, such as Tomcat, without needing professional map servers such as GeoServer, therefore greatly reducing the difficulty of map deployment and maintenance as well as project costs. You can deploy independently, access data through HTTP URLs, or integrate deployment with projects, and access data through relative paths. Taking Leaflet as an example to implement web side MBtiles loading, the complete code is as follows:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Leaflet raster XYZ tiles Example</title>
    <link rel="stylesheet"
href="https://unpkg.com/leaflet@1.7.1/dist/leaflet.css" />
</head>
<style type="text/css">
    body {
        margin: 0;
```

```
        padding: 0;
    }
    html, body, #map{
        width: 100%;
        height: 100%;
    }
</style>
<body>

<div id="map" ></div>

<script src="https://unpkg.com/leaflet@1.7.1/dist/leaflet.js"></script>
<script>
    var map = L.map('map').setView([31.562710059362658,120.29751401540051], 8);
// 初始化地图并设置中心和缩放级别
    // Loading ArcGIS online basemap

L.tileLayer('https://server.arcgisonline.com/ArcGIS/rest/services/World_Topo_Map/
MapServer/tile/{z}/{y}/{x}').addTo(map);
    // Load local MBTiles, please modify according to your file path.
    L.tileLayer('.data/tiles/mbtiles3857/Sentinel-
2_L2A_NDSI.mbtiles').addTo(map);
</script>

</body>
</html>
```

The loading effect on the map is shown in Figures 6-122.

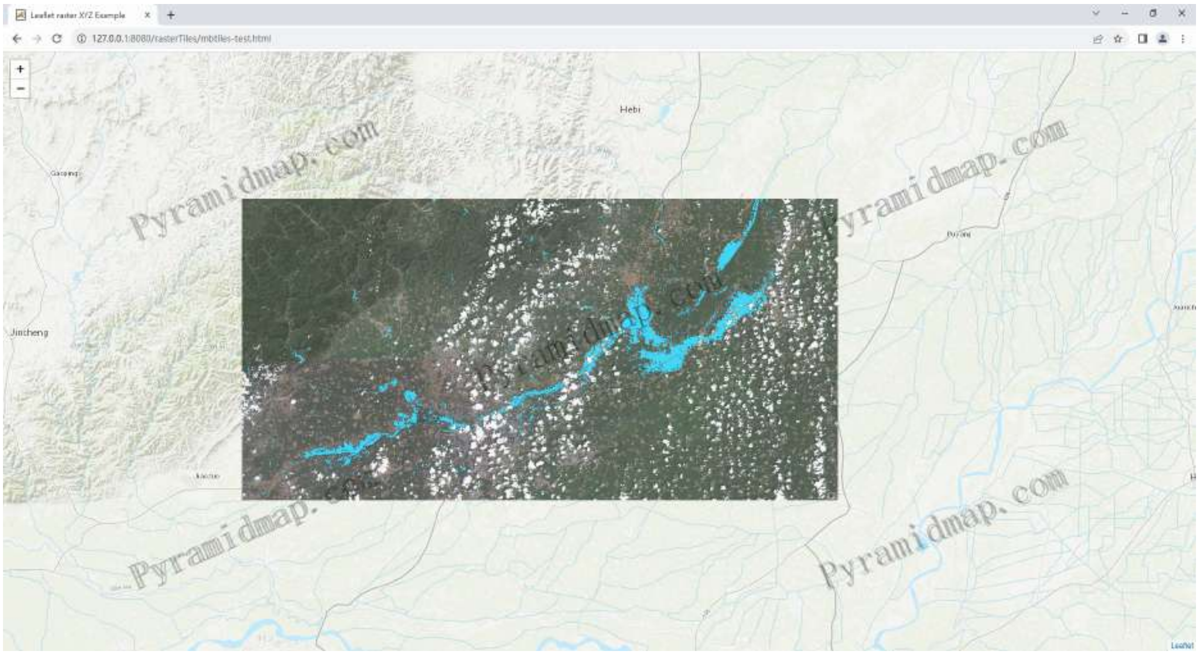


Figure 6-122: Loading raster MBTiles in Leaflet

It should be noted that although PyramidMap supports tiling operations in all standard coordinate systems, most WebGIS platforms currently use the web Mercator 3857 coordinate system. Therefore, in general, the Web Mercator 3857 coordinate system should be chosen for tile.

7 Map rendering and sld symbol

Before starting this chapter, we must understand that geospatial data does not have inherent visual components and must be styled in order to view the data. Style specifies the color, thickness, and other visible attributes used to present data on the map. For example, in ArcGIS, the display styles of point, line, and area map features are defined in mxd map documents. For the same layer, different rendering styles can be defined in different map documents to present different display effects. In GeoServer, styling is done using a markup language called Styled Layer Descriptor (SLD). SLD is an XML based markup language that is very powerful, and although it is somewhat complex, we can visually define it through the PyramidMap tool. This chapter focuses on the functions of SLD, its definition method in PyramidMap, management modes, and how it works in GeoServer.

SLD defines the symbolic features of map elements, which include: ① geometric types of points, lines, and surfaces; ② Edge width, color, and transparency; ③ Fill color and transparency; ④ Image size and icon; ⑤ Annotation fields, fonts, font colors, sizes, normal or bold, annotation positions, and fine tuning offsets describe the display of map features. On this basis, more advanced styles can be used. Especially for Point types, with well-known shapes such as circles, squares, stars, and even custom graphics or text can be used to specify points. You can use the dash style and hash value to set the line style. You can use custom tile shapes to fill polygons. Styles can be based on attributes in the data to style certain features in different ways. You can also annotate features with text. Styles can also be determined by the zoom level to fit their apparent size. In short, the possibility of using SLD to describe the display mode of spatial geometric shapes is enormous.

Clearly, defining and modifying SLD files is a complex process. PyramidMap provides a visualization approach that enables the creation, editing, and maintenance of SLD, and maintains bidirectional synchronization with the GeoServer server. It uploads data to the designated service space of GeoServer, forming a complete SLD production, management, and service process.

7.1 Define sld symbols at client

7.1.1 Create sld on visualizing layer nodes

At the visualization layer node on the left of the main map interface, create the sld symbol through the right-click shortcut menu, as shown in Figure 7-1.

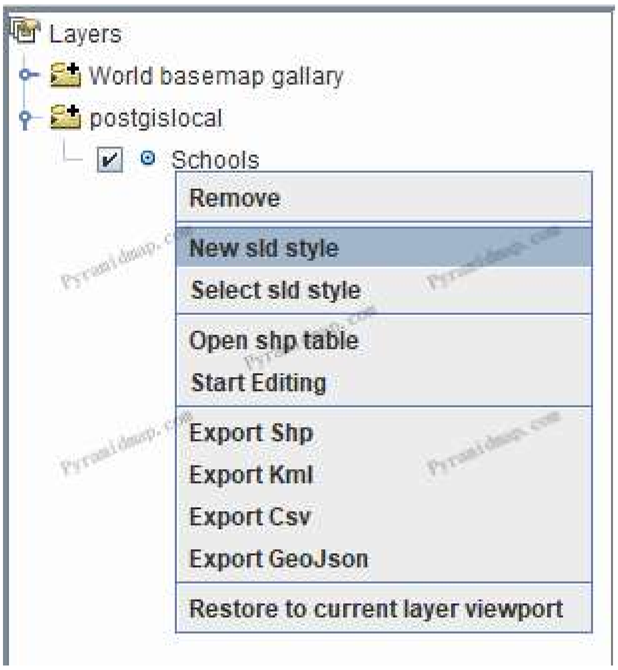


Figure 7-1： Creating sld symbols on visualization layer node

PyramidMap will implement the sld symbol definition method through the visual palette according to the geometric type (point, line, polygon) of the selected layer, as shown in Figure 7-2.

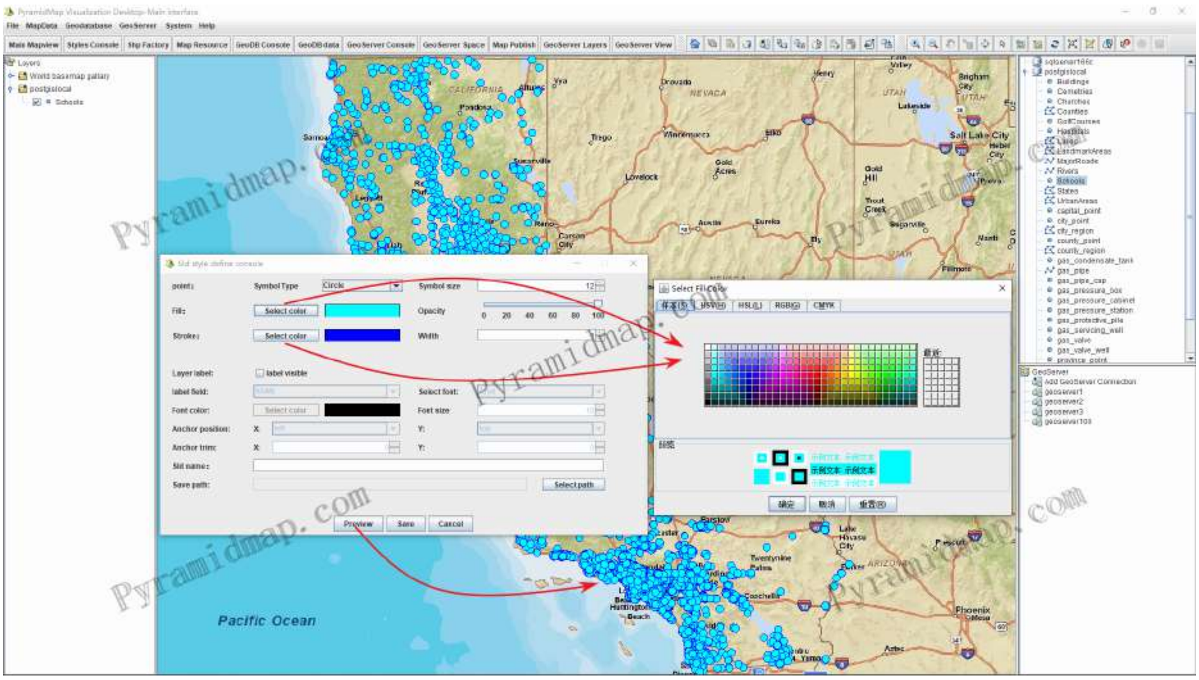


Figure 7-2： Create sld symbol definition on layer node through palette

In the sld definition module, create map symbols for different geometric types of points, lines, and surfaces through the color palette. The symbol features include: stroke color, stroke width, fill color, transparency, size, icon, annotation field, font, font color, size, normal or bold, annotation position, fine adjustment offset, etc. The display effect can be previewed in real time on the layer, saved as an sld file, and maintained in the sld resource pool at the same time, As shown in Figure 7-3.

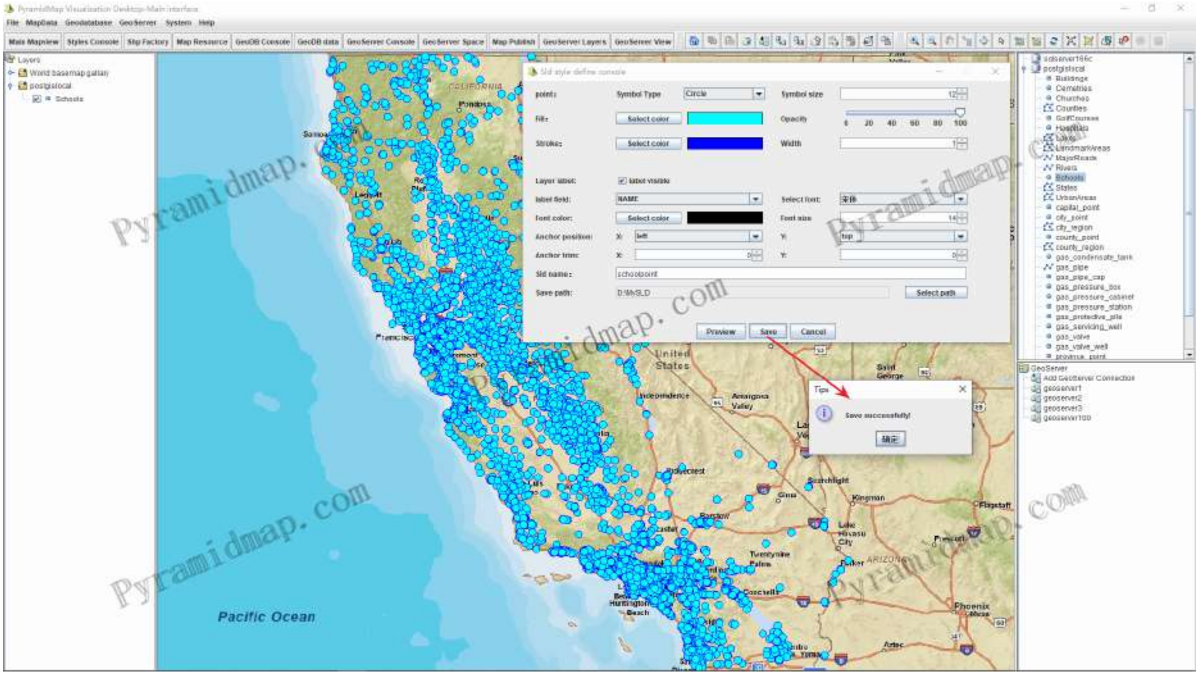


Figure 7-3: Create and save sld symbol definition on layer node through palette

The following is a simple SLD example we created that can be applied to layers containing points.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<StyledLayerDescriptor version="1.0.0"
    xsi:schemaLocation="http://www.opengis.net/sld StyledLayerDescriptor.xsd"
    xmlns="http://www.opengis.net/sld"
    xmlns:ogc="http://www.opengis.net/ogc"
    xmlns:xlink="http://www.w3.org/1999/xlink"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <NamedLayer>
    <Name>Simple point</Name>
    <UserStyle>
      <Title>Simple point sld</Title>
      <FeatureTypeStyle>
        <Rule>
          <PointSymbolizer>
            <Graphic>
              <Mark>
                <WellKnownName>circle</WellKnownName>
                <Fill>
                  <CssParameter name="fill">#FF0000</CssParameter>
                </Fill>
              </Mark>
              <Size>10</Size>
            </Graphic>
          </PointSymbolizer>
        </Rule>
      </FeatureTypeStyle>
    </UserStyle>
  </NamedLayer>
</StyledLayerDescriptor>
```

Although the example looks long, only a few lines are really important to understand. **Line 14** states that a “PointSymbolizer” is to be used to style data as points. **Lines 15-17** state that points are to be styled using a graphic shape specified by a “well known name”, in this case a circle. SLD provides names for many shapes such as “square”, “star”, “triangle”, etc. **Lines 18-20** specify the shape should be filled with a color of `#FF0000` (red). This is an RGB color code, written in hexadecimal, in the form of #RRGGBB. Finally, **line 22** specifies that the size of the shape is 10 pixels in width. The rest of the structure contains metadata about the style, such as a name identifying the style and a title for use in legends.

7.1.2 Create map symbols in the sld resource pool

PyramidMap performs centralized and unified management of client SLDs in the way of resource management pool. Enter through the "Styles console" entry of the main interface menu to open the SLD resource management pool module, as shown in Figure 7-4.

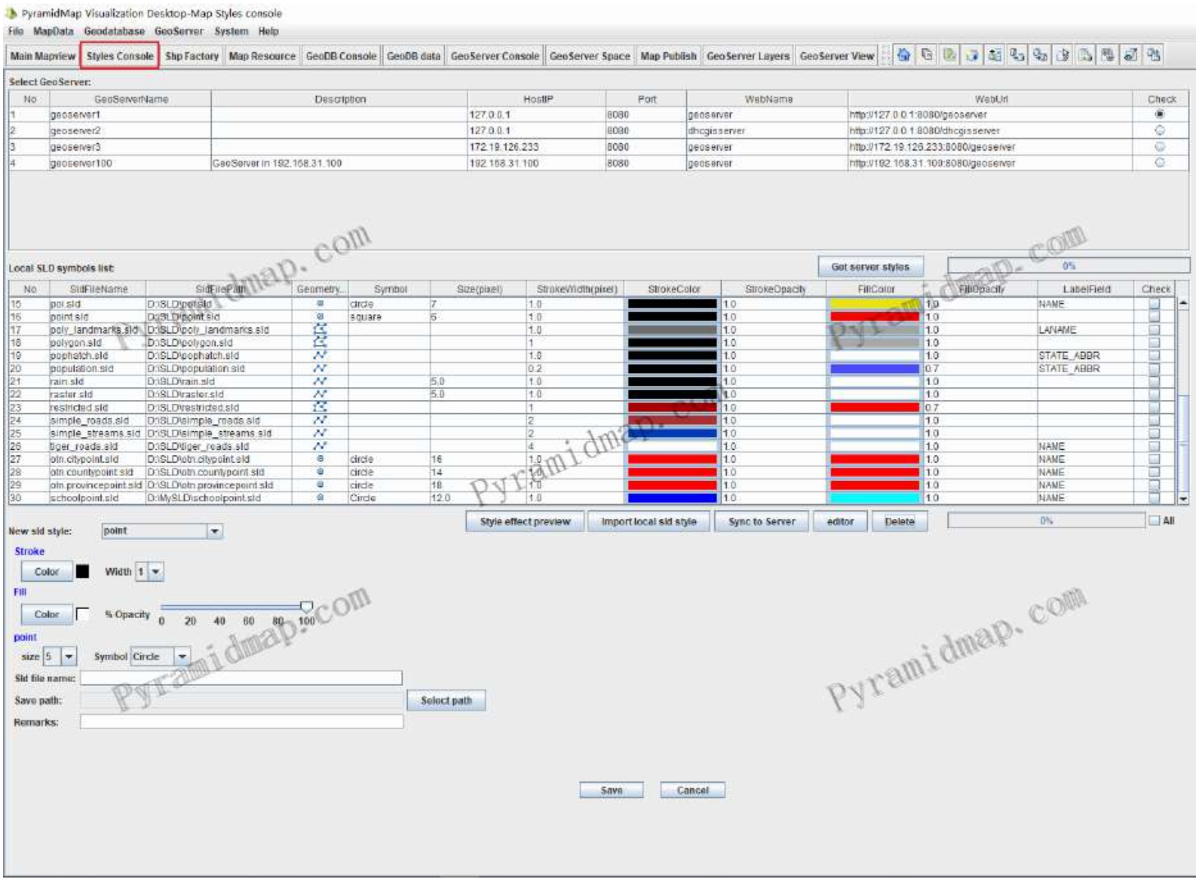


Figure 7-4： Sld symbol resource pool and creation module

PyramidMap maintains a local SLD resource pool itself, and uniformly and centrally manages the SLD files created by the client and obtained from the GeoServer. There are two ways to obtain the Sld data source, one is from the GeoServer, the other is created at the PyramidMap client. All work flows are completed in the SLD unified resource management pool module, as shown in Figure 7-5.

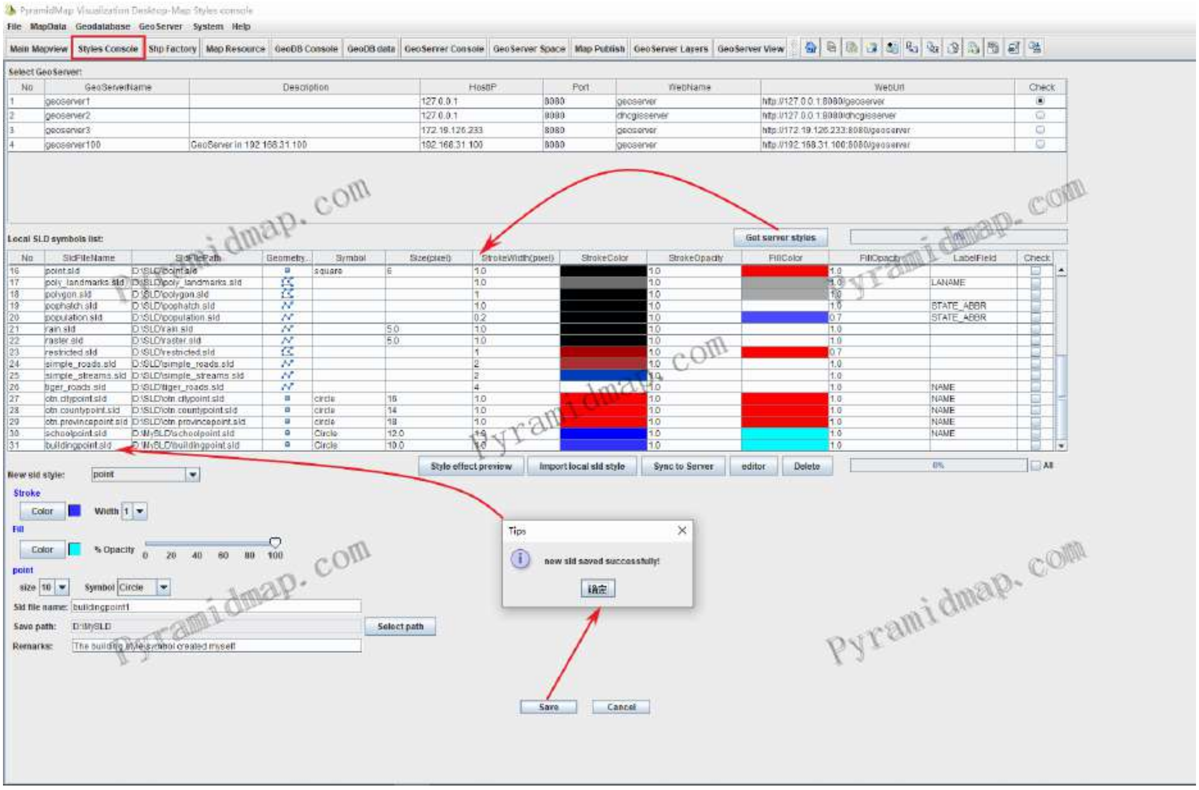


Figure 7-5：Complete the sld symbol creation and management workflow in the sld unified resource management pool

The defined sld file is included in the resource management list to realize resource sharing. You can select symbols in the sld resource pool to preview the effect, as shown in Figure 7-6.

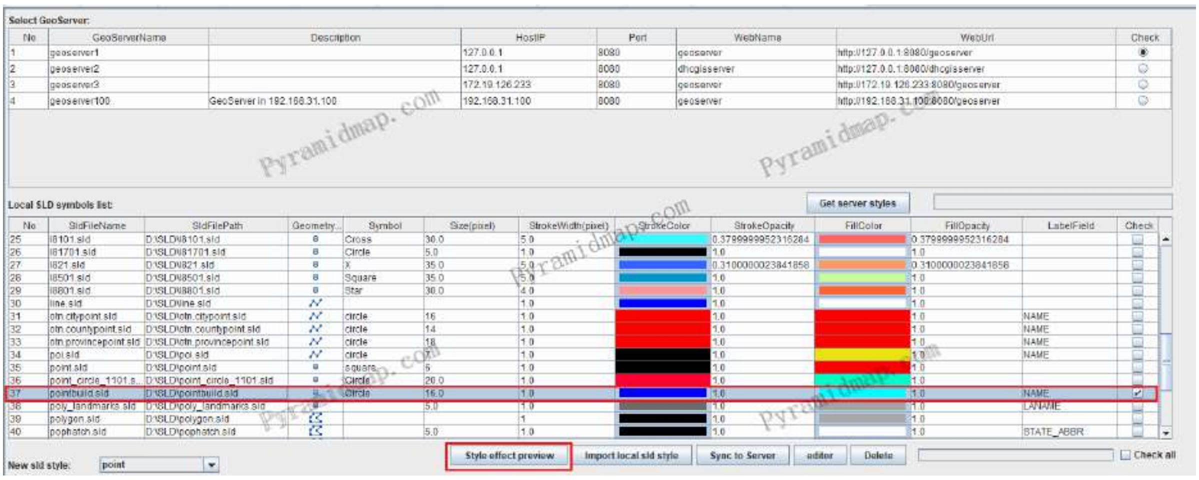
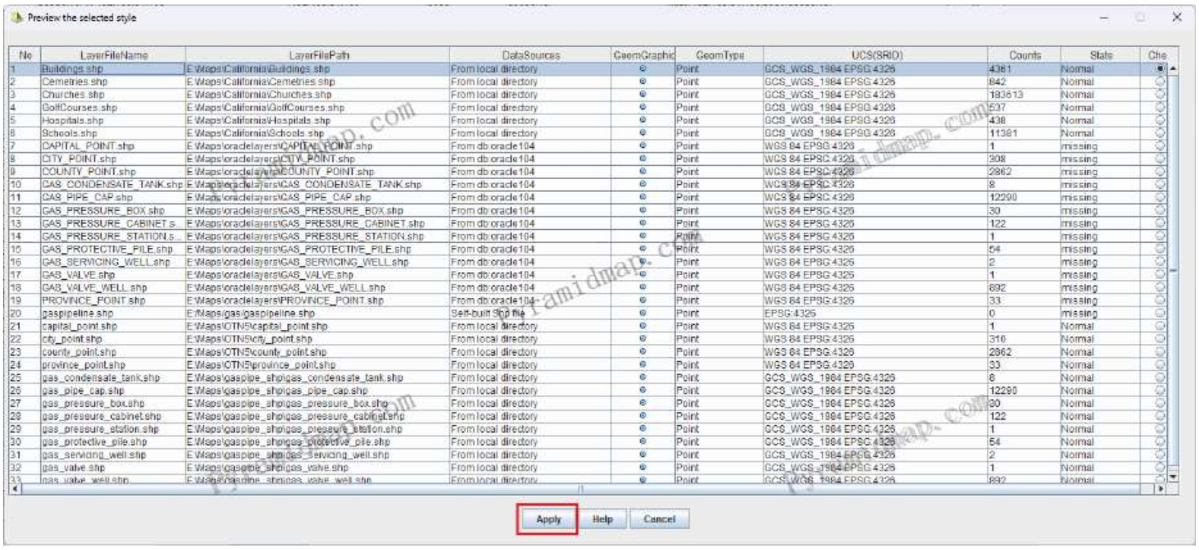


Figure 7-6：Select the map symbol in the sld resource pool for effect preview

PyramidMap will match in the layer resource pool according to the geometry type of the selected sld symbol, and return the list of layers of the same type, as shown in Figure 7-7.



Preview the selected style

No	LayerFileName	LayerFilePath	DataSources	GeomGraphic	GeomType	UCS(SRID)	Counts	State	Che
1	Buildings.shp	E:\Maps\California\Buildings.shp	From local directory		Point	GCS_WGS_1984 EPSG:4326	4361	Normal	
2	Cemetries.shp	E:\Maps\California\Cemetries.shp	From local directory		Point	GCS_WGS_1984 EPSG:4326	842	Normal	
3	Churches.shp	E:\Maps\California\Churches.shp	From local directory		Point	GCS_WGS_1984 EPSG:4326	183613	Normal	
4	GolfCourses.shp	E:\Maps\California\GolfCourses.shp	From local directory		Point	GCS_WGS_1984 EPSG:4326	537	Normal	
5	Hospitals.shp	E:\Maps\California\Hospitals.shp	From local directory		Point	GCS_WGS_1984 EPSG:4326	438	Normal	
6	Schools.shp	E:\Maps\California\Schools.shp	From local directory		Point	GCS_WGS_1984 EPSG:4326	11381	Normal	
7	CAPITAL_POINT.shp	E:\Maps\oraclelayers\CAPITAL_POINT.shp	From db:oracle104		Point	WGS 84 EPSG:4326	1	missing	
8	CITY_POINT.shp	E:\Maps\oraclelayers\CITY_POINT.shp	From db:oracle104		Point	WGS 84 EPSG:4326	308	missing	
9	COUNTY_POINT.shp	E:\Maps\oraclelayers\COUNTY_POINT.shp	From db:oracle104		Point	WGS 84 EPSG:4326	2862	missing	
10	GAS_CONDENSATE_TANK.shp	E:\Maps\oraclelayers\GAS_CONDENSATE_TANK.shp	From db:oracle104		Point	WGS 84 EPSG:4326	8	missing	
11	GAS_PIPE_CAP.shp	E:\Maps\oraclelayers\GAS_PIPE_CAP.shp	From db:oracle104		Point	WGS 84 EPSG:4326	12290	missing	
12	GAS_PRESSURE_BOX.shp	E:\Maps\oraclelayers\GAS_PRESSURE_BOX.shp	From db:oracle104		Point	WGS 84 EPSG:4326	30	missing	
13	GAS_PRESSURE_CABINET.s...	E:\Maps\oraclelayers\GAS_PRESSURE_CABINET.shp	From db:oracle104		Point	WGS 84 EPSG:4326	122	missing	
14	GAS_PRESSURE_STATION.s...	E:\Maps\oraclelayers\GAS_PRESSURE_STATION.shp	From db:oracle104		Point	WGS 84 EPSG:4326	1	missing	
15	GAS_PROTECTIVE_PILE.shp	E:\Maps\oraclelayers\GAS_PROTECTIVE_PILE.shp	From db:oracle104		Point	WGS 84 EPSG:4326	54	missing	
16	GAS_SERVICING_WELL.shp	E:\Maps\oraclelayers\GAS_SERVICING_WELL.shp	From db:oracle104		Point	WGS 84 EPSG:4326	2	missing	
17	GAS_VALVE.shp	E:\Maps\oraclelayers\GAS_VALVE.shp	From db:oracle104		Point	WGS 84 EPSG:4326	1	missing	
18	GAS_VALVE_WELL.shp	E:\Maps\oraclelayers\GAS_VALVE_WELL.shp	From db:oracle104		Point	WGS 84 EPSG:4326	892	missing	
19	PROVINCE_POINT.shp	E:\Maps\oraclelayers\PROVINCE_POINT.shp	From db:oracle104		Point	WGS 84 EPSG:4326	33	missing	
20	gaspipeline.shp	E:/Maps/gas/gaspipeline.shp	Self-built Shp file		Point	EPSG:4326	0	missing	
21	capital_point.shp	E:\Maps\OTN5\capital_point.shp	From local directory		Point	WGS 84 EPSG:4326	1	Normal	
22	city_point.shp	E:\Maps\OTN5\city_point.shp	From local directory		Point	WGS 84 EPSG:4326	310	Normal	
23	county_point.shp	E:\Maps\OTN5\county_point.shp	From local directory		Point	WGS 84 EPSG:4326	2862	Normal	
24	province_point.shp	E:\Maps\OTN5\province_point.shp	From local directory		Point	WGS 84 EPSG:4326	33	Normal	
25	gas_condensate_tank.shp	E:\Maps\gaspipe_shp\gas_condensate_tank.shp	From local directory		Point	GCS_WGS_1984 EPSG:4326	8	Normal	
26	gas_pipe_cap.shp	E:\Maps\gaspipe_shp\gas_pipe_cap.shp	From local directory		Point	GCS_WGS_1984 EPSG:4326	12290	Normal	
27	gas_pressure_box.shp	E:\Maps\gaspipe_shp\gas_pressure_box.shp	From local directory		Point	GCS_WGS_1984 EPSG:4326	30	Normal	
28	gas_pressure_cabinet.shp	E:\Maps\gaspipe_shp\gas_pressure_cabinet.shp	From local directory		Point	GCS_WGS_1984 EPSG:4326	122	Normal	
29	gas_pressure_station.shp	E:\Maps\gaspipe_shp\gas_pressure_station.shp	From local directory		Point	GCS_WGS_1984 EPSG:4326	1	Normal	
30	gas_protective_pile.shp	E:\Maps\gaspipe_shp\gas_protective_pile.shp	From local directory		Point	GCS_WGS_1984 EPSG:4326	54	Normal	
31	gas_servicing_well.shp	E:\Maps\gaspipe_shp\gas_servicing_well.shp	From local directory		Point	GCS_WGS_1984 EPSG:4326	2	Normal	
32	gas_valve.shp	E:\Maps\gaspipe_shp\gas_valve.shp	From local directory		Point	GCS_WGS_1984 EPSG:4326	1	Normal	
33	gas_valve_well.shp	E:\Maps\gaspipe_shp\gas_valve_well.shp	From local directory		Point	GCS_WGS_1984 EPSG:4326	892	Normal	

Apply Help Cancel

Figure 7-7： Returns the list of layers with the same geometry type as the selected sld

Click "Apply" to preview the effect, as shown in Figure 7-8.

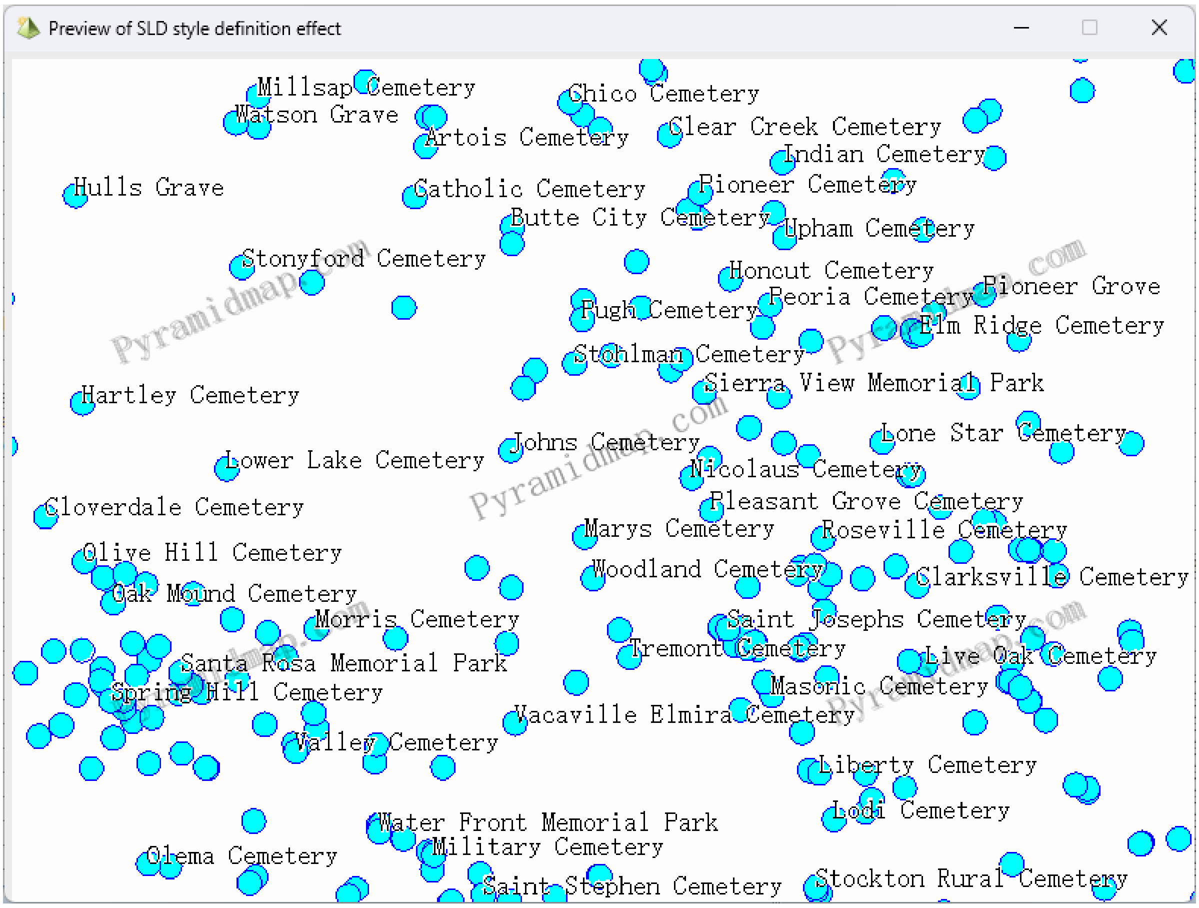


Figure 7-8： Sld symbol map effect preview

7.1.3 Get GeoServer sld symbols

Obtain sld symbols from the GeoServer through connection pool and add to the local SLD resource pool, as shown in Figure 7-9.

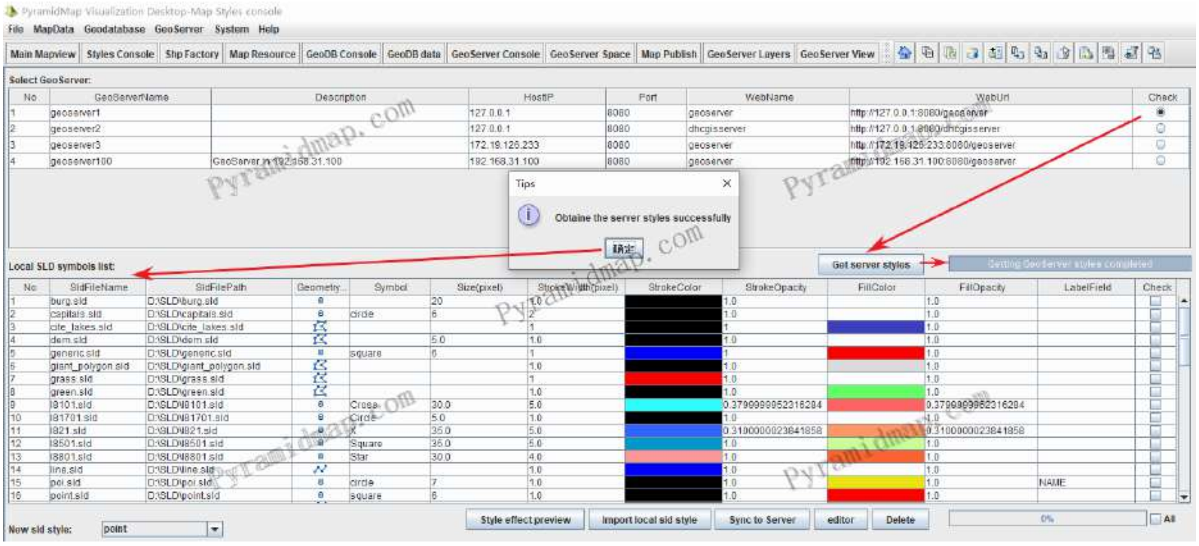


Figure 7-9：Obtain sld symbols from the GeoServer through connection pool

Select the GeoServer connection, click "Get server styles", PyramidMap gets the global SLDs in GeoServer and the SLDs in each workspace, and downloads them to the client resource pool for sharing.

7.2 Map rendering effect with sld

At the visualization layer node on the left side of the main interface, right-click the shortcut menu to Open the sld selection list. In this module, the sld resource files of the same type will be automatically filtered from the sld symbol resource pool maintained by the system according to the geometric type (point, line, and face) of the selected layer to form a selectable list. You can select the corresponding sld symbol definition to achieve the desired map rendering effect ,all of the workflow as shown in Figure 7-10.

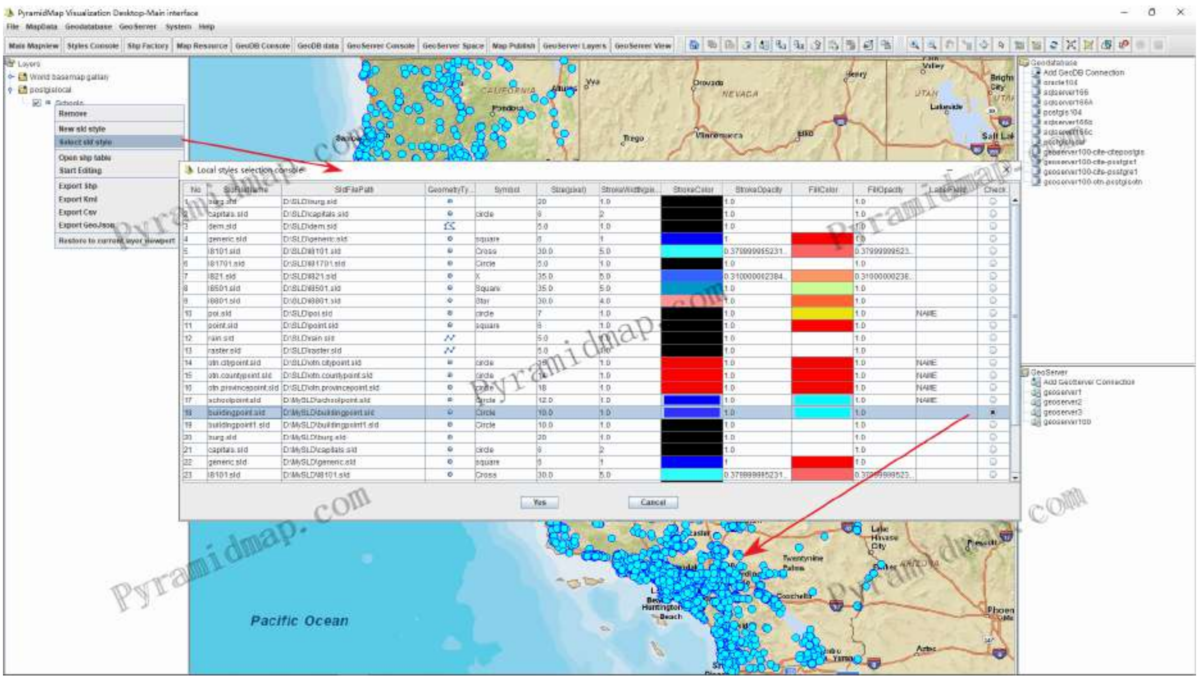


Figure 7-10：Select sld for visualization layer node to achieve map rendering effect

7.3 Client sld symbols submitted to GeoServer

PyramidMap submits the client sld symbols to the GeoServer through the connection pool, as shown in Figure 7-11.

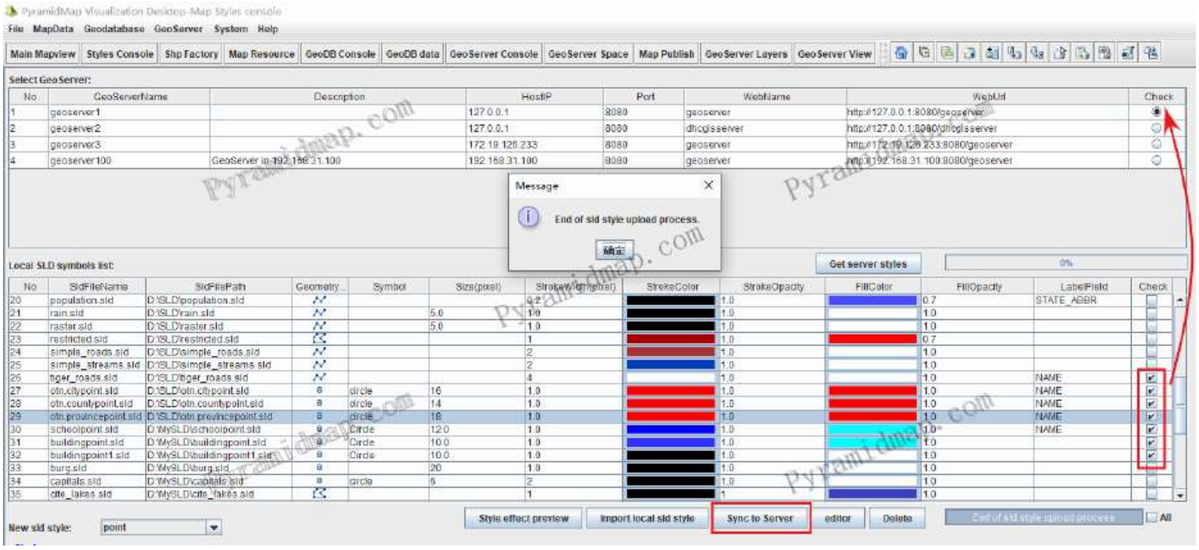


Figure 7-11： Client sld resources submitted to the GeoServer

Select all or part of the sld files and click “Sync to Server” to synchronously submit the selected sld symbols to the global space inside the GeoServer.

8 Publishing map service

WebGIS is the trend of map application development and the most extensive map application mode at present, including browser end, mobile end and embedded terminal application. Vector layer, raster layer and Geodatabase layer must be published as map service url to be applied in WebGIS terminal which hosted in map server such as GeoServer that provides web map service. GeoServer is based on the OpenGIS Web server specification, follows the OGC open standard, and can run in any J2EE/servlet based container, such as tomcat, webLogic, and webSphere, etc. GeoServer has complete functions and supports multiple map service functions, such as WMS/WFS/WCS/WMTS/KML. In the traditional way, using the GeoServer console to publish map services is very complicated and requires highly professional staff.

The significance of PyramidMap is to provide a visual guidance process to publish the layers in the local resource pool to the GeoServer server, which is simpler and easier to use than the traditional web console mode of GeoServer. The human-computer interaction is more friendly and intuitive, and the operation is convenient, which greatly reduces the professional skill requirements for operators. Even companies and individuals without GIS capabilities can easily use it, which is the biggest feature of the software. PyramidMap supports three types of layer publishing: Shp file type layer, image file type layer, and geographic database type layer. The main interface provides shortcut menu entries, as shown in Figure 8-1.

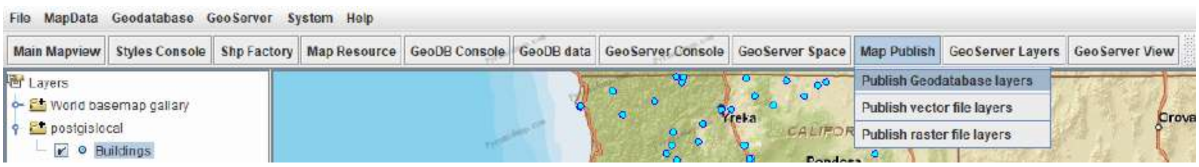


Figure 8-1： Main interface map service publishing shortcut menu entry

GeoServer supports the following three data storage modes of layer services: Geodatabase layer type, Shp vector file type, and raster image file type.

8.1 Publish Shp vector layers

Publish the local Shp vector layer to the shapefile directory on the GeoServer and use it as a data storage. GeoServer allows to store shp in file mode and output map services to web client. This is a complex process to package and submit client Shp files to the data cache path preset by GeoServer, and output them as rest map services for WebGIS access. PyramidMap simplifies this process through high integration, and makes it easier through visual guidance,the operation flows is shown in Figure 8-2.

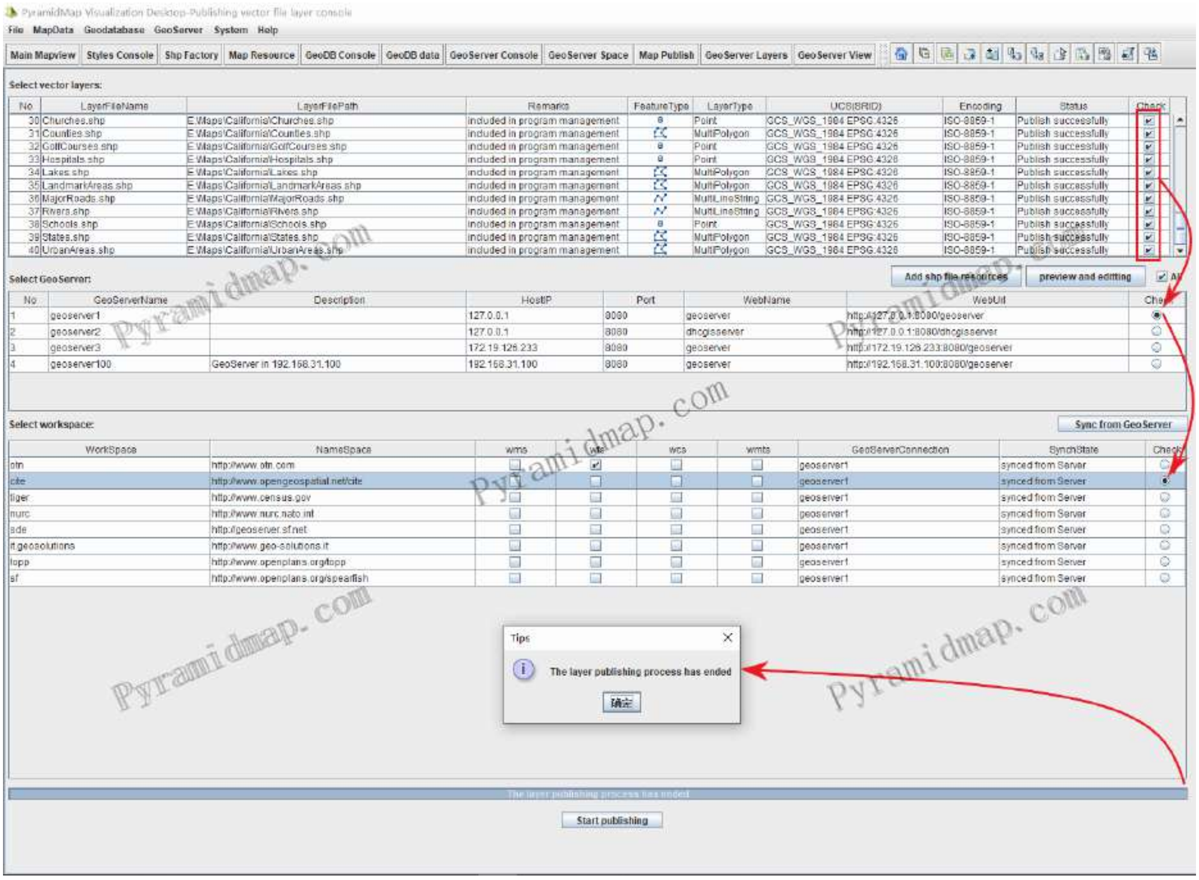


Figure 8-2：Workflow of publishing Shp vector layer to shapefile directory on GeoServer

The Shp file is uploaded to the shapefile directory preset by GeoServer, stored as a file, and exported to the map rest service.

8.2 Publish raster layers

Like the Shp vector layer publishing process, GeoServer allows raster layers being stored in file directory mode and output services to web client. Select the layers to publish in the local raster files resource pool, and then select the target GeoServer and workspace to publish. As shown in Figure 8-3.

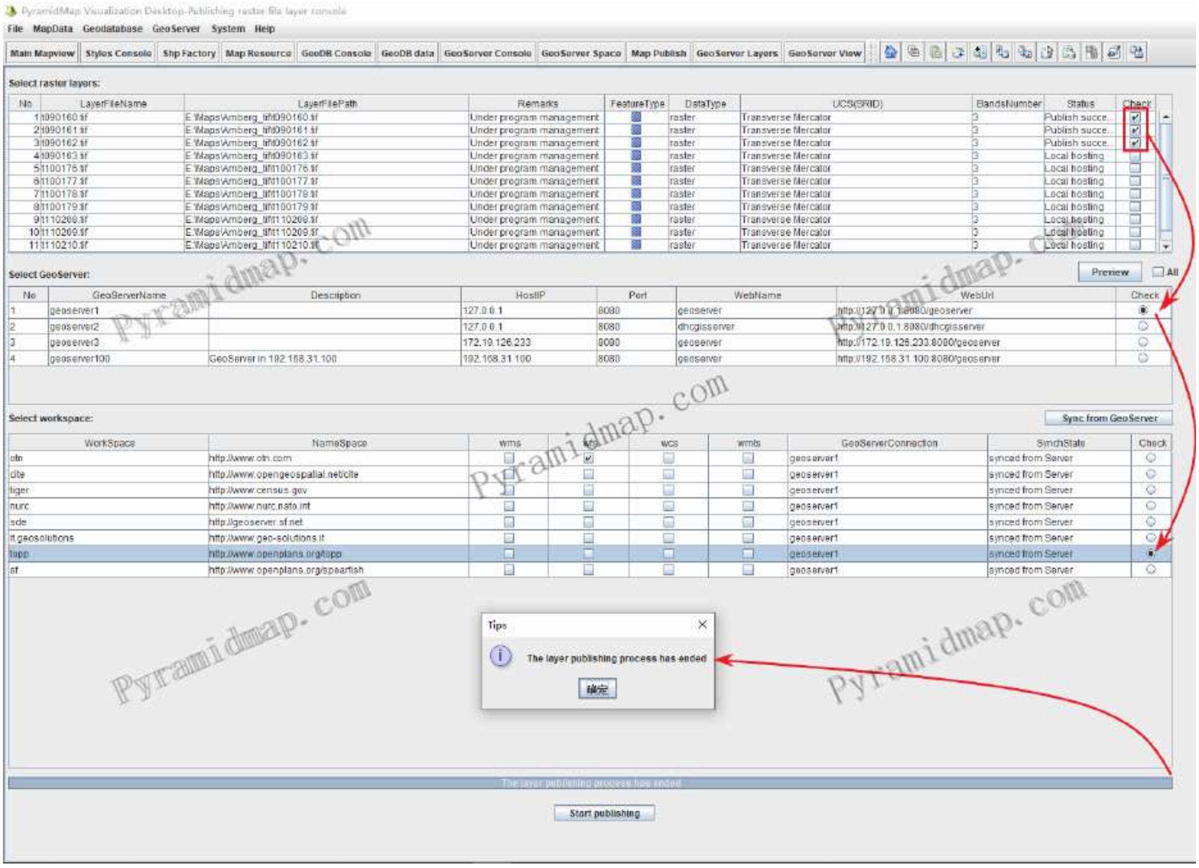


Figure 8-3： Workflow of publishing raster layer to raster file directory on GeoServer

8.3 Publish Geodatabase layers

The more powerful service function of GeoServer is that it can adapt to most DBMS geographic database systems, publish the map data as a service, and provide flexible web map services in a more extensive way to meet large-scale industrial map applications. Based on this, PyramidMap integrates the adaptation interface to the geographic database supported by GeoServer, supports but is not limited to publishing the layers in Oracle, PostGIS, MySQL and other geographic databases to GeoServer, and outputs WMS/WFS and other types of map services, as shown in Figure 8-4.

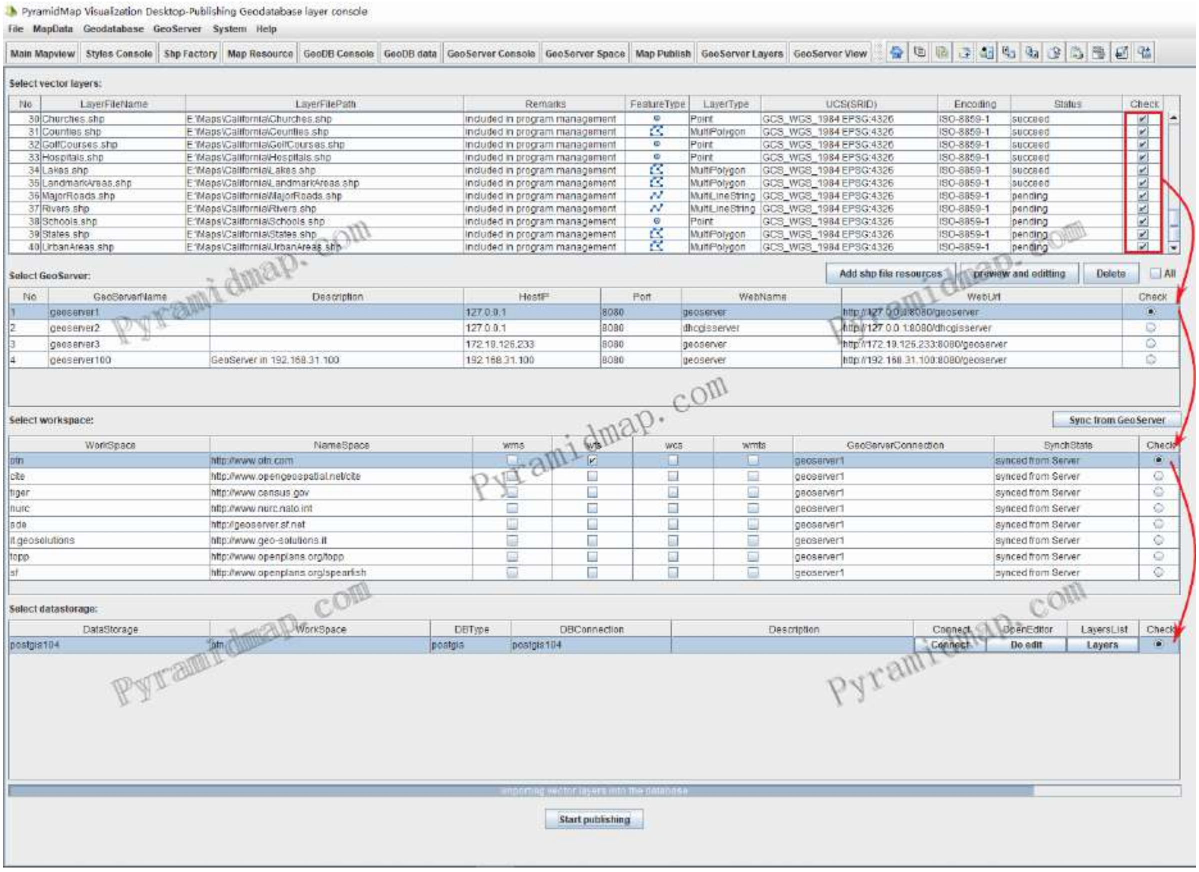


Figure 8-4： Workflow of publishing geodatabase layer to GeoServe

Select the vector layer to publish in the local vector layer resource pool, and then select the GeoServer target server to publish, the workspace in the server, and the data store. The map will first be imported into the geographic database corresponding to the data store, and then published as a map service in zaiGeoServer. PyramidMap realizes the traditional complicated publishing process through an integrated visual process, greatly reducing the difficulty of work and improving the efficiency.

8.4 Manage GeoServer layers

As the visualization client of GeoServer map server, PyramidMap implements unified management of layers in GeoServer, including layer preview and query. PyramidMap's management of GeoServer layers includes three main functions: GeoServer data source node in the main interface, as shown in Figure 8-5.

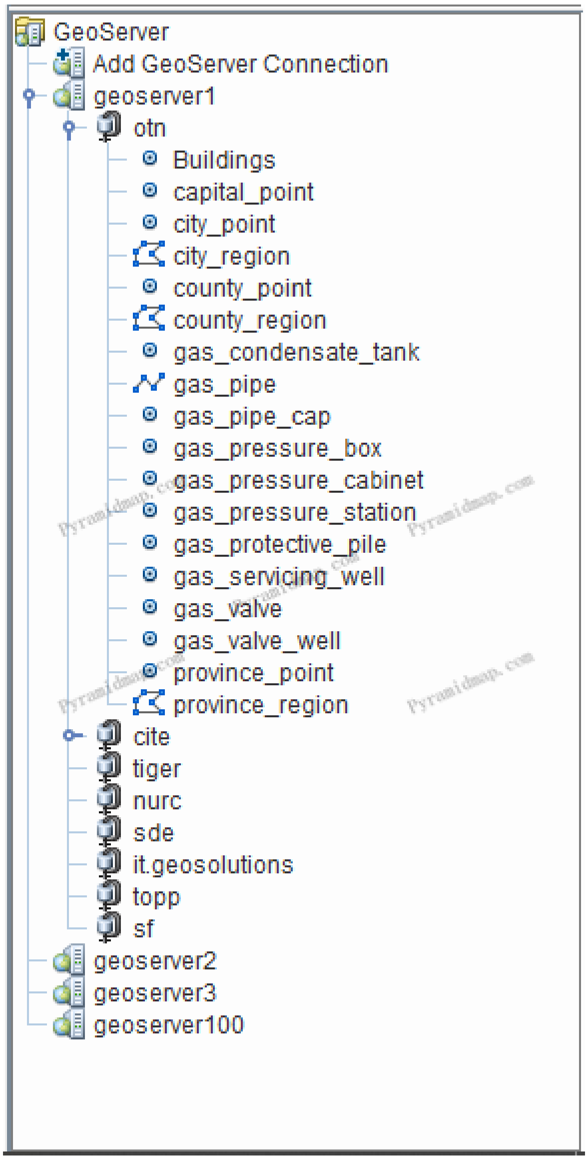


Figure 8-5： GeoServer data source node in the main interface

Through the GeoServer data source node, you can complete various hierarchical operations on the GeoServer workspace, data storage, and layers. PyramidMap provides access to GeoServer layer preview list and GeoServer layer management list through toolbar menu items, as shown in Figure 8-6.

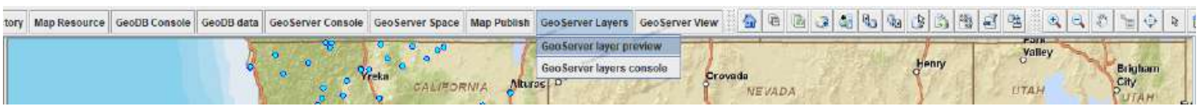


Figure 8-6： Main interface GeoServer layers operation entrance

Through the above menu items, the layer preview and management of GeoServer can be realized.

8.4.1 GeoServer layers preview

PyramidMap previews the layers published by the server through the GeoServer resource connection pool. The list of layers is shown in Figure 8-7.

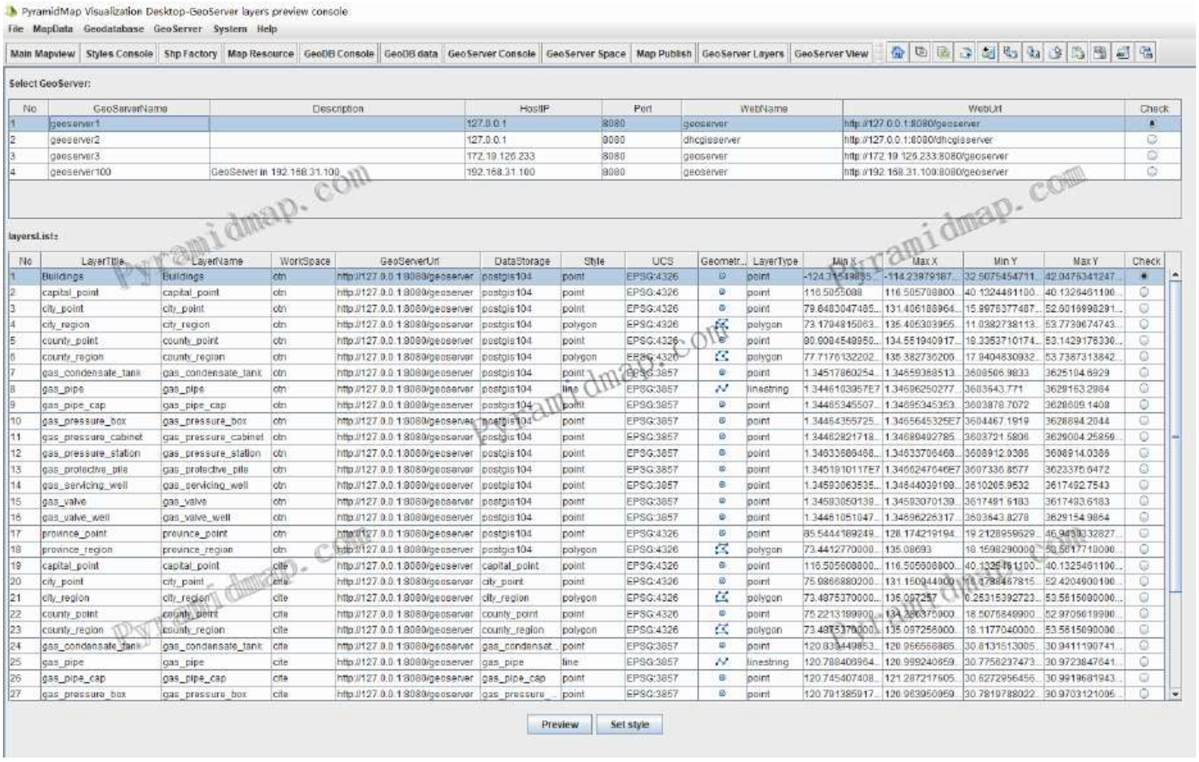


Figure 8-7： GeoServer layer list

The GeoServer layer list includes two types: vector and image. The vector layer is divided into point, line and face types. Select the vector layer and click Preview. PyramidMap will load and display the selected vector layer through WMS, as shown in Figure 8-8.

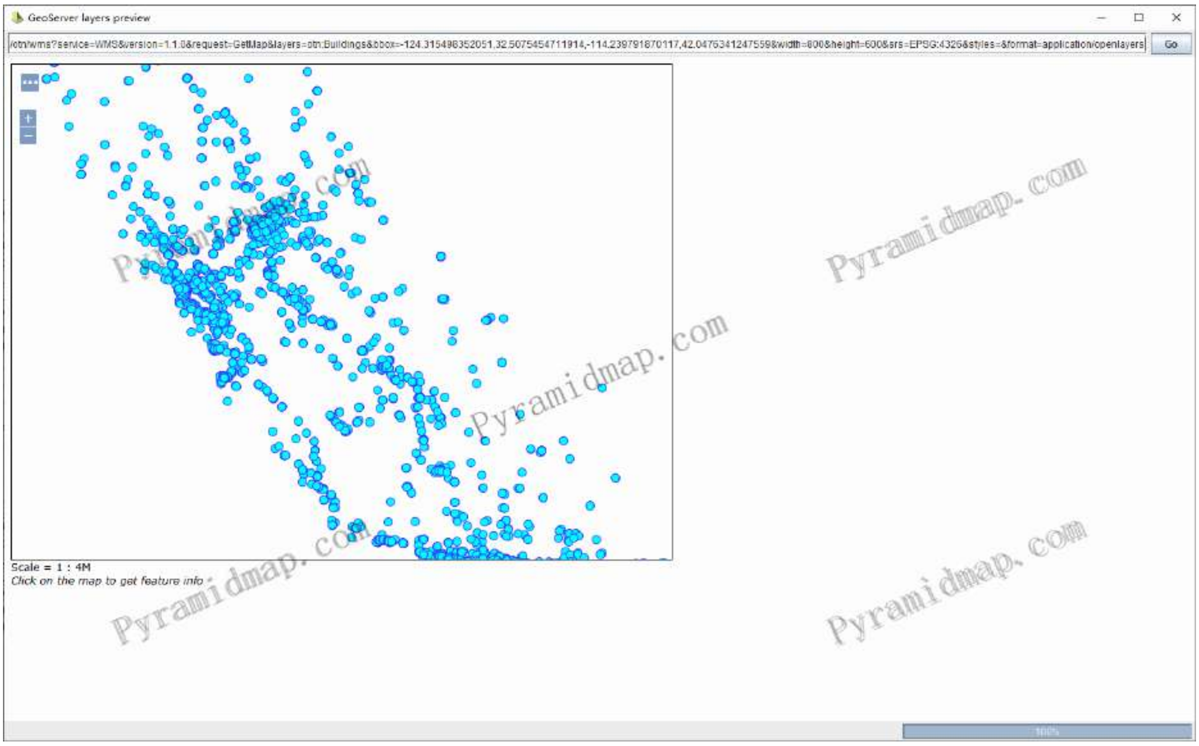


Figure 8-8： GeoServer vector layer preview

Select an raster layer and click Preview. PyramidMap will load and display the selected raster layer in WMS mode, as shown in Figure 8-9.

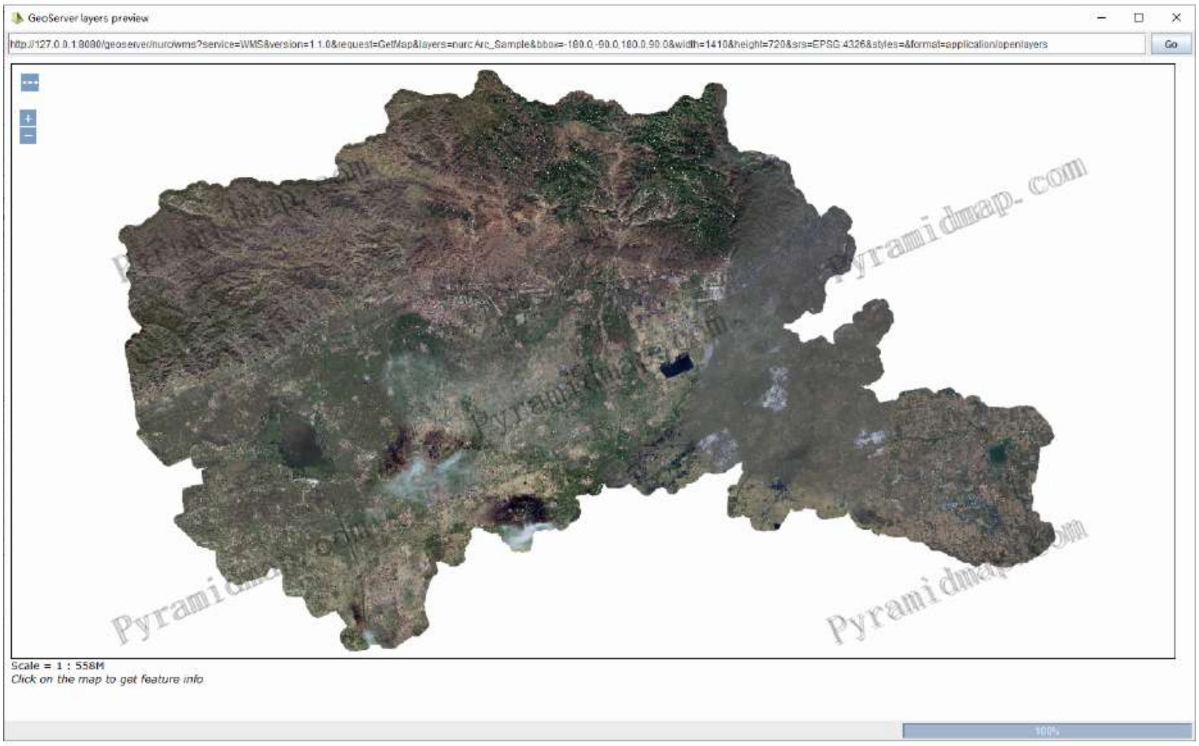


Figure 8-9：GeoServer raster layer preview

8.4.2 GeoServer layers export and conversion

PyramidMap provides some column export and data conversion operations for the layers in GeoServer, which is not available in regular GeoServer, giving more flexibility and operable space to map data. As shown in Figure 8-10.

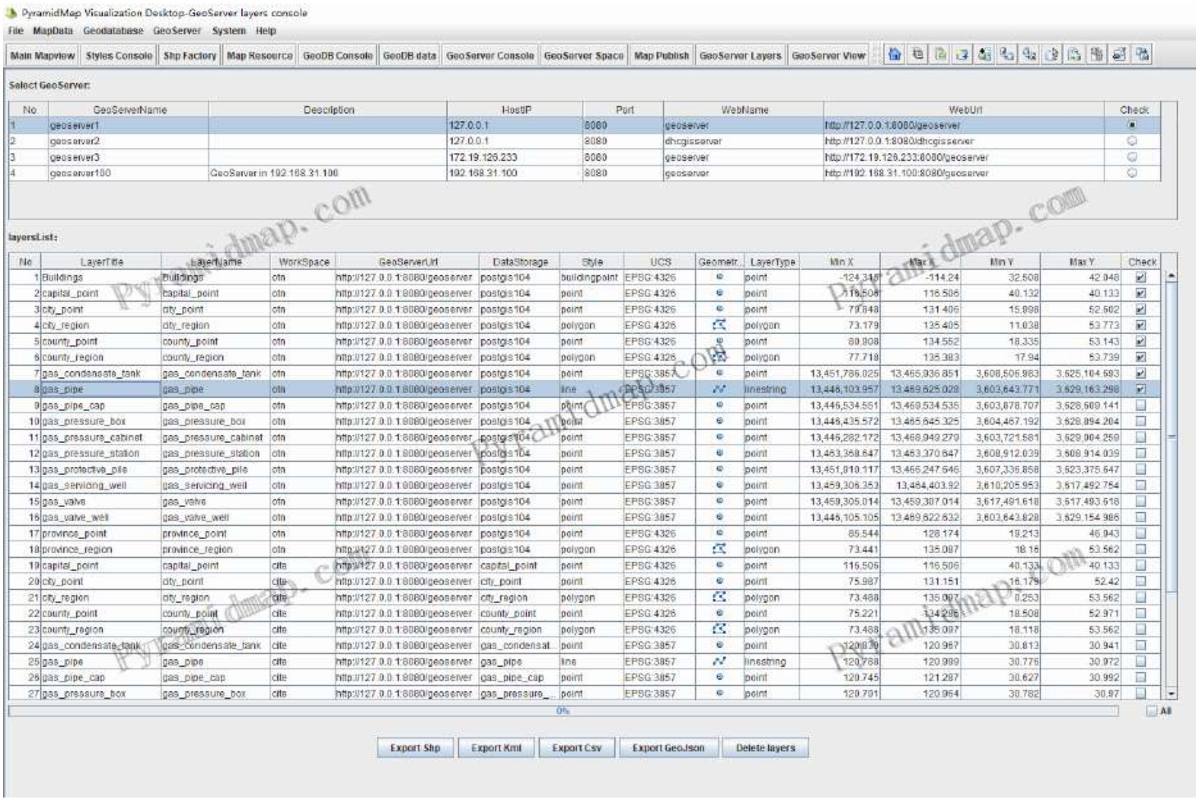


Figure 8-10：GeoServer layers export and conversion

PyramidMap can export the layers in GeoServer to Shp, Kml, Csv, GeoJson and other formats, realizing the visual transformation of map data and expanding the utilization space of map data.