Utilizing OGR to export spatial and tabular data via command line

The following information was provided by Greg Krakow, at Georgia Nongame Conservation Section. This document provides instructions regarding the installation and use of the free and open source GDAL utility, **ogr2ogr**, for exporting Biotics 5 data to ESRI shapefiles, geodatabases and tables for local use. The scripts can be placed in simple Windows batch (.bat) files to automatically run using Windows Scheduled Tasks.

**NOTE**: ogr2ogr does NOT support CLOB fields (i.e. eo\_data, gen\_desc,…) so in order to include a CLOB field, it must first be converted to a string. Following is an example utilizing EO\_DATA:

*dbms\_lob.substr(EO.EO\_DATA,255) as EO\_DATA*

And included in a select statement :

*select eo\_shape.shape, eo\_shape.eo\_id, dbms\_lob.substr(EO.EO\_DATA,255) as EO\_DATA from eo\_shape, eo where eo\_shape.eo\_id=eo.eo\_id*

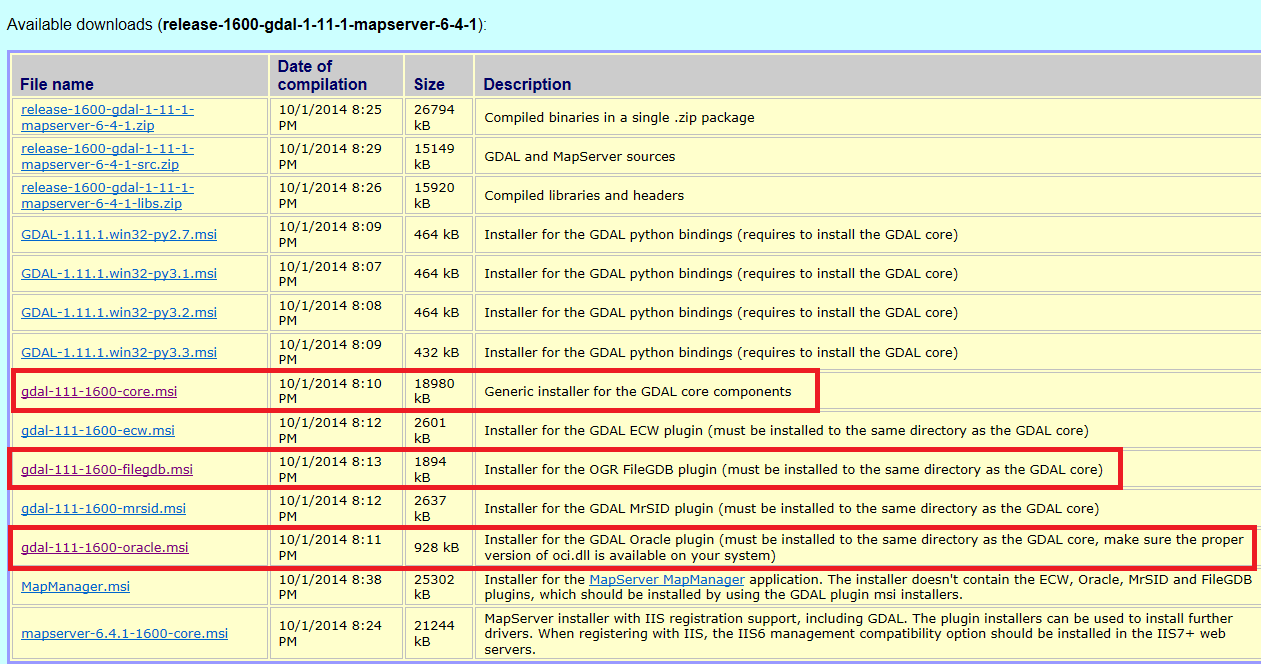
Software Requirements:

* Oracle client
* GDAL

# Installing OGR and other required software

1. Download and install Windows binaries for GDAL and extensions for Oracle and ESRI file geodatabases.
   1. Within an internet browser, go to <https://www.gisinternals.com/release.php> and select the **Download** of the latest release for your Windows version (32- or 64- bit)
   2. Download the following 3 files from that webpage , noting that versions (numbers) will differ:

* [gdal-xxxx-core.msi](http://www.gisinternals.com/sdk/Download.aspx?file=release-1600-gdal-1-11-1-mapserver-6-4-1\gdal-111-1600-core.msi)
* [gdal-xxxx-filegdb.msi](http://www.gisinternals.com/sdk/Download.aspx?file=release-1600-gdal-1-11-1-mapserver-6-4-1\gdal-111-1600-filegdb.msi)
* [gdal-xxxx-oracle.msi](http://www.gisinternals.com/sdk/Download.aspx?file=release-1600-gdal-1-11-1-mapserver-6-4-1\gdal-111-1600-oracle.msi)



1. Install the three 3 windows msi files:
2. In Windows Explorer, double-click the **gdal-xxxx-core.msi** file.
3. Check **Accept license**.
4. Click on **Typical Installation**
5. Click **Install**
6. Repeat step 3 for the **gdal-xxxx-filegdb.msi** & **gdal-xxxx-oracle.msi** files.
7. Add the GDAL path to the Windows PATH environmental variable and create a new environmental variable called GDAL\_DATA.
   1. From Windows Explorer right click on **Computer** and select P**roperties**.
   2. On the right side of the Computer properties window click on the **Change settings**.
   3. Within the *System Properties* dialog, navigate to the **Advanced** tab, and click **Environmental Variables**.
   4. Within the *System variables* section, select the **Path** variable and click the **Edit** button.
   5. On the keyboard, click the **End** key to go to the end of the PATH string.
   6. Add a semicolon at the end of the text string and then add the complete folder name where the org2ogr.exe is located. If you are using the 32 bit version It should be: **C:\Program Files (x86)\GDAL**. Click **OK**.

**TIP**: Navigate to GDAL in Windows Explorer and then right click on GDAL in the address bar at the top of the window and select **Copy address as text** to put the proper path into windows clipboard as text to paste into the Environmental Variable editing window.

* 1. Within the *System variables* section, click the **New** button.
     1. Enter *Variable name*: **GDAL\_DATA**
     2. Enter *Variable value*: **C:\Program Files (x86)\GDAL\gdal-data**
  2. Click **OK**, **OK**, **OK** to save the appended PATH variable and new GDAL\_DATA variable to your Windows system.

1. Test the setup:
   1. Open a Windows **Command Prompt** window.
   2. Within the *Command Prompt* window, type **ogr2ogr** and press **Enter**. You should see text displaying help for ogr2ogr displayed in the window.
   3. **I**f you get a message **'ogr2ogr' is not recognized as an internal or external command, operable program or batch file** then something went wrong when you added the PATH variable. Check the Environmental Variables:
      1. Within the *Command Prompt*, type: **PATH** and press **Enter**. You should see **C:\Program Files (x86)\GDAL** at the end of the path text string.
      2. Within the *Command Prompt*, type: **ECHO %GDAL\_DATA%** and press **Enter**. You should receive **C:\Program Files (x86)\GDAL\gdal-data** from the command prompt.
2. [Install the latest Oracle client](http://bioticssupport.natureserve.org/solution/categories/28022/folders/264999/articles/191986-installing-oracle-11g-client), if not already installed. Users of Biotics that view/update Biotics 5 data in Crystal Reports, Access, SQL, SQLDeveloper, TORA, Toad, Excel, or other programs should already have this installed.

# Utlizing OGR to export shapefiles and data from Biotics 5

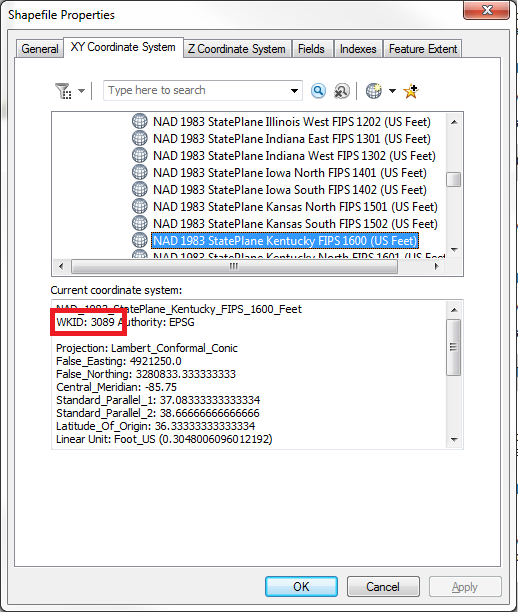
Following are some useful examples; further information and export options can be found at <http://www.gdal.org/ogr2ogr.html>.

The following command line script exports the results of an Oracle Spatial SQL query of the entire EO\_SHAPE table to:

* an ESRI Shapefile:

*ogr2ogr -f "ESRI Shapefile" c:/temp/eo\_shape.shp -lco SHPT=POLYGON OCI:biotics\_report/password@biotics5:EO\_SHAPE -a\_srs EPSG:26918 –progress*

Explanation of command line switches for the above:

* **-f "ESRI Shapefile" c:/temp/eo\_shape.shp**: Create a file, c:/temp/eo\_shape.shp, of format ESRI ShapeFile
* **-lco SHPT=POLYGON**: Convert 3D polygonz (XYZ) to 2D polygon (XY). Required for editing in some older programs like ArcView 3.x
* **OCI:biotics\_report/PASSWORD@biotics5:EO\_SHAPE**: Use OCI (Oracle Call Interface) to connect to Oracle database and return view, EO\_SHAPE. **NOTE**: File name must be upper case. Several tables may be exported by separating the names with commas.**-**
* **-a\_srs EPSG:26918**: Assign the projection to create a prj.\*, projection file, by assigning it a spatial reference system. The spatial reference system should indicate that used within Biotics 5 as the state projection. To determine this, export a managed layer (EO, SF, MA, or Site) from Biotics 5 and view the shapefile Properties in ArcCatalog. Navigate to the **XY Coordinate System** tab in the *Shapefile Properties* dialog – the number is identified as the **WKID** (3089 in the screenshot below). **NOTE**: **–t\_srs srs\_def:** used to project to a different projection than the state projection used in Biotics 5.  
  
* **-progress**: show a progress bar.
* an ESRI ArcGIS 10.x Geodatabase:

*ogr2ogr -update -overwrite -nlt multipolygon -nln eo\_shape -f FileGDB c:/temp/test.gdb OCI:biotics\_report/password@biotics5:EO\_SHAPE -a\_srs EPSG:26918 -progress*

Explanation of command line switches for the above:

* **update:** Open existing output datasource in update mode rather than trying to create a new one
* **overwrite:** Delete the output layer and recreate it empty
* **nlt *type*:** Define the geometry type for the created layer. One of NONE, GEOMETRY, POINT, LINESTRING, POLYGON, GEOMETRYCOLLECTION, MULTIPOINT, MULTIPOLYGON or MULTILINESTRING. And CIRCULARSTRING, COMPOUNDCURVE, CURVEPOLYGON, MULTICURVE and MULTISURFACE for GDAL 2.0 non-linear geometry types. Add "25D" to the name to get 2.5D versions. Starting with GDAL 1.10, PROMOTE\_TO\_MULTI can be used to automatically promote layers that mix polygon or multipolygons to multipolygons, and layers that mix linestrings or multilinestrings to multilinestrings. Can be useful when converting shapefiles to PostGIS (and other target drivers) that implements strict checks for geometry type. Starting with GDAL 2.0, CONVERT\_TO\_LINEAR can be used to to convert non-linear geometries types into linear geometries by approximating them.  
  **NOTE**: To insure that spatial comparisons return the correct results, use the MULTIPOINT, MULTIPOLYGON, and MULTILINESTRING options rather than their singular counterparts or all features of a multi-part feature will not be selected unless using SELECT BY (i.e. versus INTERSECT). See BX-3353.
* **nln *name*:**Assign an alternate name to the new layer
* **-f FileGDB c:/temp/test.gdb:** Create a file, c:/temp/test.gdb, of format ESRI geodatabase
* **OCI:biotics\_report/PASSWORD@biotics5:EO\_SHAPE**: Use OCI (Oracle Call Interface) to connect to Oracle database and return view, EO\_SHAPE. **NOTE**: File name must be upper case. Several tables may be exported by separating the names with commas.**-**
* **-a\_srs EPSG:26918**: Assign the projection to create a prj.\*, projection file, by assigning it a spatial reference system. The spatial reference system should indicate that used within Biotics 5 as the state projection. If your system utilizes a custom projection, contact NatureServe for this number. **NOTE**: **–t\_srs srs\_def:** should be used to project to a different projection than the state projection used in Biotics 5.
* **-progress**: show a progress bar.

The following exports the same thing but utilizes an SQL statement – this allows the user to utilize a WHERE clause or more complex query:

*ogr2ogr -f "ESRI Shapefile" c:/temp/eo\_shape.shp -lco SHPT=POLYGON OCI:biotics\_report/password@biotics5 -sql "select \* from eo\_shape" -a\_srs EPSG:26918 -progress*

Explanation of command line switches for the above:

* **-sql "select \* from eo\_shape"**: Use an Oracle SQL select statement to create data table for export from database. For testing add “where rownum < 5” to sql statement.

You can also use ogr2ogr to export non spatial tables from Oracle (dbf, sqlite, csv and many more). The exported table can then be joined to the exported shapefile/geodatabase. The following exports a csv table that opens in Excel, Access and many other spreadsheets and databases – the first statement exports the entire table, the second does as well, but utilizes a SQL statement to define the outgoing data.

*ogr2ogr -f CSV c:\temp\tableName.csv OCI:biotics\_report/password@biotics5:EO\_VIEW\_ALL\_ATT -progress*

*ogr2ogr -f CSV c:\temp\sqlSelect.csv OCI:biotics\_report/password@biotics5 -sql "select \* from EO\_VIEW\_STD\_ATT" -PROGRESS*

ERRORs similar to the following can be ignored. The script seems to run correctly in spite of these.

*ERROR 1: ORA-04043: object AA\_EO does not exist  in OCIDescribeAny*

Alternatively, export to a shapefile utilizing a SQL select statement which queries the EO\_SHAPE table as well as the EO\_VIEW\_ALL\_ATT\_MAP view to get a shapefile containing the desired data already joined to the shapes selected. This example replicates the Mapper function “Export Spatial Data” with “Element Occurrences”.

*ogr2ogr -f "ESRI Shapefile" c:/temp/ eo\_shape\_join.shp -lco SHPT=POLYGON OCI:biotics\_report/password@biotics5 -sql "select eo\_shape.shape, eo\_view\_all\_att\_map.\* from eo\_shape, eo\_view\_all\_att\_map where eo\_shape.eo\_id=eo\_view\_all\_att\_map.eo\_id" -a\_srs EPSG:26918 -progress*

To create a Windows batch (\*.BAT) file just create a text file with the ogr2ogr command in the file. You may want to delete some files first before ogr2ogr runs. Place the following batch file text in a text file and save it with a \*.bat extension. You can then have the user double click on it to execute and extract data as needed from Biotics5. Alternaticely, set Windows Task Scheduler to execute the batch file in the middle of the night to help reduce bandwidth usage during working hours and save staff time for when they may need the output.

:::::::::::  START OF BATCH FILE ::::::::::::::::::::

REM batch file lines starting with REM or :: at the beginning are not executed

REM Delete all csv files from c:\tmp2

del c:\tmp2\del \*.csv

REM Extract AA\_ES and AA\_EO views from Bioitcs5 as comma delimited CSV files and place in C:\tmp2

ogr2ogr -update -overwrite -f CSV c:\tmp2 OCI:biotics\_report/PASSWORD@biotics5:AA\_ES,AA\_EO -nlt NONE -lco SEPARATOR=TAB –progress

::::::::::::: END OF BATCH FILE :::::::::::::::::::::

Sample batch file for exporting shapefiles:

:::::::::::  START OF BATCH FILE ::::::::::::::::::::

REM Delete eo\_del.shp shapefile and all of its parts from c:\tmp

del c:\tmp\eo\_del.\*

REM Export the AA\_EO\_DEL spatial view from Biotics5 into a 2D shapefile named eo\_del.shp that

REM has a projection file for UTM Zone 17N NAD83 (EPSG:26917) and that will load into ArcView 3.3.

ogr2ogr -f "ESRI Shapefile" c:/tmp/eo\_del.shp -lco SHPT=POLYGON OCI:biotics\_report/PASSWORD@biotics5 -sql "select \* from aa\_eo\_del where rownum < 10" -a\_srs EPSG:26917 -progress

::::::::::::: END OF BATCH FILE :::::::::::::::::::::

# Troubleshooting

Encountered a set of issues here after installing latest GDAL, among the errors I got:

ERROR 1: PROJ: proj\_as\_wkt: Cannot find proj.db

ERROR 1: PROJ: proj\_create\_from\_wkt: Cannot find proj.db

Fixed these by adding an additional system environment variable: PROJ\_LIB and setting it to C:\Program Files (x86)\GDAL\projlib, which is where GDAL's projection database is located.

Information provided by Chris Fontenot.