

The premier hybrid cloud and AI event **Think 2021**
May 11 — Americas | May 12 — APAC & EMEA

IBM Developer Recipes Home All recipes My recipes

Manage Db2 spatial data with open source GDAL/OGR

David Adler

Published on November 7, 2016 / Updated on November 9, 2020

👁 2292 ☆ 0 0

Contents

Overview

Ingredients

Setup Db2 and sample data

Install GDAL/OGR

Get information about vector spatial tables in Db2


Import and export vector spatial data to/from DB2

Import/Export raster data to/from Db2

Develop applications using the GDAL Python API

Set up GDAL with Db2 on Ubuntu Linux

Conclusion

 Recipes are community-created content. They are neither monitored nor endorsed by IBM. If you find inappropriate content, please [Abuse](#) to let us know. For more information on community content, please refer to our [Terms of Use](#).

Overview

Skill Level: Any Skill Level

GDAL/OGR is an open source toolkit that provides many utilities to work with spatial data.

This recipe takes you through the steps to convert spatial data in raster and vector formats to and fr

Ingredients

- Db2 client and server([Db2 Developer client & server download](#))
- GDAL/OGR Windows install packages ([download](#))
- [GDAL/OGR web site and documentation](#)
- Sample data ([download](#))
- Db2 spatial documentation as appropriate for your environment:
 - [Db2 for z/OS Knowledge Center](#)
 - [Db2 LUW Knowledge Center](#)

With this recipe you will populate Db2 tables with spatial data, get information about the spatial data a in both vector and raster formats into and from DB2.

Step-by-step

1 Setup Db2 and sample data

1. Note: the GDAL/OGR installation for Windows includes the DB2 support. In order to use GI need to build this support from source following the instructions at the end of this recipe. GDAL/OGR connecting to a local DB2 database. Change the database connection values in database name and userid.
2. Download and unzip the sample data to a convenient directory.

3. Open a Db2 command window where you can execute SQL statements. If you are using a remote server, make sure it is cataloged locally. You can use the scripts **catalog-luw.sql** or **catalog-zos.sql** as a starting point for changes for the remote server location. (Use **catalog-luw.sql** for DashDB)

You can execute the script with a command like:

```
db2 -tvf catalog-luw.sql
```

4. Import sample spatial for banks, customers and San Jose streets.
 - Modify the scripts **import-luw.sql** or **import-zos.sql** as appropriate to change the userid and database. (The scripts contain descriptions of each of the steps and any modifications needed)
 - Execute the script with a command like:


```
db2 -tvf import-luw.sql
```
 - Check that there were no processing errors
5. Verify the imported data with the SQL statement (change “**dash5510**” to your connection name):


```
db2 select name, street, varchar(db2gse.st_astext(geom),32) from dash5510.banks
```

 which should return the name, street address and location of the **Meridian** and **San Carlos** streets.

2 Install GDAL/OGR

1. From the GDAL/OGR ([download](#)) location select the release that corresponds to your Windows architecture:


```
release-1911-gdal-2-3-3-mapserver-7-2-1
```

 or


```
release-1911-x64-gdal-2-3-3-mapserver-7-2-11
```
2. Select and download the “**Generic installer for the GDAL core components**” (currently named **gdal-core-components-2-3-3-mapserver-7-2-11.exe**)
3. Execute the downloaded file to have Windows install GDAL
4. If you want to use the Python support, also download and install the “**Installer for the GDAL Python bindings**” (currently named **gdal-python-bindings-2-3-3-mapserver-7-2-11.exe**) that correspond to the version of Python you have installed. It is important that the “bitness” of the GDAL Python bindings and the Python environment all be the same.

3 Get information about vector spatial tables in Db2

The **ogrinfo** utility is the main tool to get information about the vector spatial tables in DB2.

From the Windows start menu, navigate to the **GDAL** application and select the GDAL command prompt.

The various options can be displayed with the command

ogrinfo -help

but you are probably better off looking at the detailed description at the web site <http://www.gdal.org/>

You can get a list of all the tables that have spatial columns with a command like:

**ogrinfo DB2ODBC:"DRIVER={IBM DB2 ODBC
DRIVER}";DATABASE=OSTEST;HOSTNAME=localhost;PROTOCOL=TCPIP;PORT=50000;UID=**

The parameters should be modified to match your database connection information. For a remote connection, replace localhost with the IP address of the database server. This example is using DB2 LUW; for DB2 for z/OS the parameters would be different. Note that the **DRIVER** specification needs to be surrounded by double-quotes as the driver name contains spaces.

This should result in a simple list of the sample spatial tables:

You can list information about a specific table and each of the features by specifying the **-al** option. The **tables** specification (this can result in a lot of output):

**ogrinfo -al DB2ODBC:"DRIVER={IBM DB2 ODBC
DRIVER}";database=OSTEST;Hostname=localhost;PROTOCOL=TCPIP;port=50000;UID=**

You can list just summary information about a specific table the **-al** and **-so** options and append the specification. If you define an Windows ODBC datasource, you can use a simpler syntax by just the datasource name (DSN):

```
ogrinfo -al -so DB2ODBC:DATABASE=ostest;DSN=ostest64;tables=banks;
```

4 Import and export vector spatial data to/from DB2

The **ogr2ogr** utility is the main tool to move or convert vector data from one source or format to another.

From the Windows start menu, navigate to the GDAL application and select the GDAL command line interface.

The various options can be displayed with the command

```
ogr2ogr -help
```

but you are probably better off looking at the detailed description at the web site <http://www.gdal.org/ogr2ogr.html>

To export a DB2 spatial table to the Esri shapefile format, you can use a command like the following. Specify the output name, **banks2.shp**, and then the source table in DB2.

```
ogr2ogr banks2.shp DB2ODBC:"DRIVER={IBM DB2 ODBC  
DRIVER}";DATABASE=OSTEST;HOSTNAME=localhost;PROTOCOL=TCPIP;PORT=50000;UID=
```

The Windows directory listing shows the 4 files that constitute a shapefile:

Conversely, to import a shapefile into DB2, you can use a command like the following:

```
ogr2ogr -f DB2ODBC DB2ODBC:"DRIVER={IBM DB2 ODBC  
DRIVER}";DATABASE=OSTEST;HOSTNAME=localhost;PROTOCOL=TCPIP;PORT=50000;UID=
```

banks2.shp

It is necessary to specify the option **-f DB2ODBC** to let **ogr2ogr** know that the destination is DB2.

The result can be checked with:

ogrinfo -al -so DB2ODBC:"DRIVER={IBM DB2 ODBC

DRIVER}";DATABASE=OSTEST;HOSTNAME=localhost;PROTOCOL=TCPIP;PORT=50000;UID=

which should return:

5 Import/Export raster data to/from Db2

The Db2 support for raster data through GDAL uses the [specification](#) defined by the Open Geospatial Consortium (OGC).

The **gdal_translate** utility is the main tool to move or convert raster data from one source or format to another.

The various options can be displayed with the command

gdal_translate -help

but you are probably better off looking at the detailed description at the web site

http://www.gdal.org/gdal_translate.html.

To export a DB2 spatial table to the Esri shapefile format, you can use a command like the following, specifying the output name, banks2.shp, and then the source table in DB2.

gdal_translate -of DB2ODBC -co RASTER_TABLE=small_world small_world.tif DB2ODBC:"DRIVER={IBM DB2 ODBC

DRIVER}";database=OSTEST;Hostname=localhost;PROTOCOL=TCPIP;port=50000;UID=ostest

The option **-of DB2ODBC** specifies that the output is to DB2. The option **-co RASTER_TABLE=small_world** specifies the name of the raster data in DB2. This is followed by the name of the input file and the DB2 connection string.

Running the command shows the size of the image in pixels and a progress message:

The raster data is managed in 5 tables with the DB2 schema GPKG which can be shown with tl

You can use the **gdalinfo** utility to get information about raster data.

The various options can be displayed with the command

gdalinfo -help

but you are probably better off looking at the detailed description at the web site [http://www.g](http://www.gdal.org/)

The information about the raster data just loaded can be shown with a command like:

**gdalinfo -oo TABLE=small_world DB2ODBC:"DRIVER={IBM DB2 ODBC
DRIVER}";database=OSTEST;Hostname=localhost;PROTOCOL=TCPIP;port=50000;UID=os**

Which should show the following:

You can also use the `gdal_translate` utility to export raster data from DB2 to a file (or other format like:

```
gdal_translate -of GTIFF -oo TABLE=small_world DB2ODBC:"DRIVER={IBM DB2 ODBC DRIVER}";database=OSTEST;Hostname=localhost;PROTOCOL=TCPIP;port=50000;UID=os small_world2.tif
```

The option **-of GTIFF** specifies the output format. The option **-oo TABLE=small_world** specifies raster data in DB2. This is followed by the DB2 connection information and the output file name.

6 Develop applications using the GDAL Python API

GDAL provides bindings to write applications in C, C++, Python and Java. The easiest way to find these APIs is to Google “GDAL API” and the language.

Python is a very convenient language for developing applications as it doesn’t require compilation. Information to get started with can be found in the [OSGEO GDAL wiki](#) and the [GDAL tutorial](#).

It isn’t practical to go into an in-depth Python tutorial here of all the GDAL capabilities but a simple example is provided. (Make sure you add Python to your path before running). The file **db2test.py** is in the sample data. You can run it with a command like:

```
python db2test.py
```

This is the source code for the sample:

```
# include OGR functionalityimport ogr# attempt to open a DB2 ODBC connection
```

This should produce output like the following (the specific customer may be different):

7 Set up GDAL with Db2 on Ubuntu Linux

While the ODBC and Db2 support is not included in the GDAL distributions for Linux, it is fairly easy to build and configure this using the steps below. This assumes that you already have DB2 installed in a development environment with git and gcc.

1. Install the unixodbc support:

- sudo apt install unixodbc
- sudo apt install unixodbc-dev

2. Get the GDAL source

- git clone <https://github.com/OSGeo/gdal> gdal

3. Configure and build GDAL with ODBC and DB2 support

- cd gdal
- sed -i 's/walk/walk\ db2/' ogr/ogrsf_frmts/GNUMakefile
- sed -i 's/\-DWALK_ENABLED/\-DWALK_ENABLED\ \-DDB2_ENABLED/' ogr/ogrsf_frmts/GNUMakefile
- ./configure --with-odbc=/usr/include
- make
- sudo make install
- note: sudo ldconfig may be needed to resolve missing libgdal.so.20

4. Configure /etc/odbc.ini – change 'OSTEST' to your database name and DMEvAttr to refer

5. Configure /etc/odbcinst.ini – point to the appropriate DB2 library

6. Configure /home/db2inst1/sqllib/cfg/db2cli.ini – provide connection details for your datab

7. If you loaded the sample data in Linux as for Windows above, you should be able to issue a command like:
ogrinfo -al -so "DB2ODBC:database:OSTEST;dsn=OSTEST;tables=OSUSER.BANKS"

should return:

8 Conclusion

GDAL/OGR is one of the oldest and most widely used GIS toolkit.

With GDAL/OGR you can import raster and vector data to/from Db2 as well as converting between formats.

The Python API is a powerful environment for building GIS applications exploiting spatial data database.

by David Adler

Join The Discussion

Enter your comments...

[Contact](#) [Privacy](#) [Terms of use](#) [Accessibility](#) [Feedback](#) [Report Abuse](#) [Cookie Preferenc](#)