MapInfo EFAL

This driver supports the MapInfo TAB file format including the MapInfo (Native) and the new MapInfo Extended (NATIVEX) formats.

Contents

1. Overview
2. Options
3. Limitations
4. Installation
5. Examples
6. Notes

Overview

This Driver plugin is built for GDAL version 3.0.X and supports the MapInfo Native and NativeX formats using the EFAL library from Precisely The driver supports reading, writing, and creating data in these formats. The MITAB library does not support the newer MapInfo Extended (NativeX) format which allows for larger file sizes, unicode character encodings, and other internal enhancements.

EFAL is a new thread-safe SDK provided by Precisely. You can download the SDK from "https://support.precisely.com/product-downloads/item/mapinfo-efal-sdk-download/". The EFAL SDK enables access to the MapInfo SQL data access engine. It also provides the ability to open, create, query, and modify data in MapInfo TAB and MapInfo Enhanced TAB file formats including the MapInfo Seamless tables. The SDK includes detailed documentation on the API. The SDK DLLs and related files must be available on your machine for the driver to execute.

The EFAL driver for GDAL is supported on 64-bit Windows, Ubuntu, AmazonLinux, OracleLinux and CentOS.

This driver supports the following capabilities:

Opening and querying of MapInfo Native and NativeX TAB files

- The EFAL API uses WKB as the interchange format for geometries.
  - Arc and Ellipse types are also not currently supported.
  - RECT and Rounded RECT types are returned as polygons.

  - Legacy Text type is returned as custom WKB geometry.
- Coordinate systems are supported using the MITAB capabilities through the importFromMICoordSys and exportToMICoordSys methods on the OGRSpatialReference class. Due to this, the MITAB driver must be included in any build of GDAL that intends to include the EFAL driver.
- Styles in TAB files are converted to/from OGR style strings; currently to a very similar level of interoperability as MITAB.
- Spatial filtering - MBR only
- Attribute filtering using SQL expressions
- Read by Feature ID (random read)
- Opening a folder as a datasource will open and expose all TAB files within the folder (like MITAB)
- File locking option when opening to improve random read performance (details below)

Editing of features within MapInfo tables

- Insert, Update, and Delete supported
- File locking option when opening to improve insert, update, delete performance (details below)

Creation of new MapInfo Native and NativeX TAB tables (layers)

- Can specify Native or NativeX
- Character encoding options including UTF-8 and UTF-16 for NativeX
- Feature property types: Integer Integer64 Real String Date DateTime Time
- Blocksize options

Differences with MITAB

- No MID/MIF support.

- Table (layer) schema lists geometry column as having name “OBJ” whereas MITAB does not.
- EFAL reports geometry type for table level whereas MITAB reports it as Unknown.
- Style strings - order of PEN and BRUSH clauses are usually reversed although results are the same.

Options

Open Options

These options apply to either the input dataset open option (-oo) or the destination dataset open option (-doo)

MODE={READ-ONLY|LOCK-READ|READ-WRITE|LOCK-WRITE} - Controls how the table is opened and files are locked.

- READ-ONLY - table is opened in read-only mode and files are not locked
- LOCK-READ - table is opened in read-only mode and files are locked for shareable read
- READ-WRITE - table is opened in write mode (supports insert, update, and delete) but files are only locked during editing operations
- LOCK-WRITE - table is opened in write mode and files are locked for write for the lifetime of the table

Layer Creation options

These options apply to the layer creation options (-lco)

- BOUNDS=[xmin],[ymin],[xmax],[ymax]

Dataset Creation options

These options apply to the dataset creation options (-dcco)

- FORMAT={NATIVE|NATIVEX}, Default is NATIVE
- CHARSET - Default is WLATIN1 for NATIVE, UTF-8 for NATIVEX. The list of allowed values is:
  - WLATIN1

  - WLATIN2
  - WARABIC
  - WCYRILLIC
  - WGREEK
  - WHEBREW
  - WTURKISH
  - WTCHINESE
  - WSCHINESE
  - WJAPANESE
  - WKOREAN
  - WTHAI
  - WBALTICRIM
  - WVIETNAMESE
  - UTF8
  - UTF16
  - ISO8859_1
  - ISO8859_2
  - ISO8859_3
  - ISO8859_4
  - ISO8859_5
  - ISO8859_6
  - ISO8859_7
  - ISO8859_8
  - ISO8859_9
  - CP437
  - CP850
  - CP852
  - CP857
  - CP860
  - CP861
  - CP863
  - CP865
  - CP855
  - CP864
  - CP869
  - NATIVE
- BLOCKSIZE=[number], Default is 16384, max is 32768.

Limitations

- Driver does not support GDAL virtual filesystem.

Installation

EFAL SDK Runtime

The EFAL driver needs the EFAL SDK to be installed on the machine to work with GDAL.

Download the EFAL SDK by navigating to "https://support.precisely.com/product-downloads/item/mapinfo-efal-sdk-download/" and fill up the SDK request form to receive the download link on your e-mail. Once the SDK package is downloaded, unzip the package on your machine at the desired location.

Structure of EFAL SDK package includes three main folders:

- data –> Sample data folder.
- export –> Binaries folder.
  - Common –> Common files used across platforms.
  - ua64 –> 64 bit Binaries for AmazonLinux.
  - uc64 –> 64 bit Binaries for CentOS.
  - uo64 –> 64 bit Binaries for OracleLinux.
  - uu64 –> 64 bit Binaries for Ubuntu.
  - ux64 –> 64 bit Binaries for Windows.

  - uw32 –> 32 bit Binaries for Windows.
- Solution –> Samples folder.

Choose the binaries for the desired platform from the "export" folder and copy all files from the "export/Common" folder into the binaries folder. For example:

:: To use binaries for Ubuntu, copy all the files from the "export/Common" folder to the "export/uu64". Create the system environment variable EFAL_SDK_DIR pointing to the "export/uu64" directory.

When using GDAL with this driver, the location of the EFAL runtime must be available on the system path. For example

```
SET PATH=%PATH%;%EFAL_SDK_DIR% --> On Windows
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH;$EFAL_SDK_DIR --> On Linux
```

NOTE: Please refer to "https://support.precisely.com/product-downloads/item/mapinfo-efal-sdk-download/" for terms of usage of EFAL SDK.

EFAL Driver Plugin

The EFAL driver is built as a gdal plugin. In order to use the EFAL driver Library "ogr_EFAL.so or ogr_EFAL.dll" for the plugin needs to be copied into the GDAL_DRIVER_PATH(path to folder as per environment variable.) folder(you will need to explicitly set this environment variable to allow GDAL to find its plugin drivers). For example

:: Linux: export GDAL_DRIVER_PATH=/root/efal/drv cp ogr_EFAL.so $GDAL_DRIVER_PATH

Windows: SET GDAL_DRIVER_PATH=C:efaldrv cp ogr_EFAL.dll %GDAL_DRIVER_PATH%

Once EFAL driver library is copied into GDAL_DRIVER_PATH folder and EFAL SDK libraries are installed on your system, next call to "ogrinfo –formats" should list driver "MapInfo EFAL -vector- (rw+): MapInfo EFAL" when it goes through the driver registration process.

Examples

ogr2ogr -f "MapInfo EFAL" c:\data\new\usa_caps.TAB c:\data\samples\usa_caps.tab

ogr2ogr -dsco CHARSET=ISO8859_1 -f "MapInfo EFAL" c:\data\new\usa_caps.TAB c:\data\samples\usa_caps.tab

ogr2ogr -dsco FORMAT=NATIVEX -f "MapInfo EFAL" -f "MapInfo EFAL" c:\data\new\usa_caps.TAB c:\data\samples\usa_caps.tab

ogr2ogr -dsco CHARSET=ISO8859_1 -dsco FORMAT=NATIVEX -f "MapInfo EFAL" c:\data\new\usa_caps.TAB c:\data\samples\usa_caps.tab

ogr2ogr -oo MODE=LOCK-WRITE -f "MapInfo EFAL" c:\data\new\usa_caps.TAB c:\data\samples\usa_caps.tab

ogr2ogr -lco BOUNDS=-180,15,-60,75 -f "MapInfo EFAL" c:\data\new\usa_caps.TAB c:\data\samples\usa_caps.tab

---

Notes

Feb-26th-2021 : Added support for smallint and boolean type when converting from other formats to MapInfo EFAL supported formats.