Customizable ODBC Driver Connections

To support new ODBC drivers that do not work out of the box with Alteryx, you can now create a custom Lua script which will map driver and database data types to Alteryx types.

Lua is a lightweight scripting language. For information, visit www.lua.org.

Script Location and Naming

- Lua scripts must be placed in the Alteryx **RuntimeData** folder:
  <installation_directory>\Alteryx\bin\RuntimeData\ODBC
- Custom scripts must be named the same as the driver. For example, the SQL Server Native Client 11.0 driver, **sqlncli11.dll** must be named **sqlncli11.lua**.
  Note: *You can determine a driver's file name via the* ***ODBC Data Source Administrator*** *application > **Drivers** tab > **File** column.*
- Script names are case insensitive.

The Common Lua Script

Your custom script must call the **common.lua** script, which contains code enum mappings and lengths for Alteryx data types, ODBC C typedefs, and C type identifiers.

- All scripts must use these common enums to map properly in the Alteryx code base.
- To use the **common.lua** script, the first line in a custom script must be:
  common = require('./RuntimeData/ODBC/common')
- Use the following syntax to reference a specific enum from the **common.lua** script in a custom Lua script:
  - common.types: *common.types.{Bool…}*
  - common.length: *common.length.{Bool…}*
  - common.ODBCTypes: *common.ODBCTypes.{SQL_BIT…}*
  - common.CTypes: *common.CTypes.{SQL_C_BIT…}*
- The **common.CTypes** enums specify which C data types will be used to store data in the application. For more information, see:
  https://msdn.microsoft.com/en-us/library/ms714556(v=vs.85).aspx
- The **common.ODBCTypes** enums are SQL data types, which map database-specific types to driver-specific SQL types. These types are returned through the ODBC API calls **SQLColAttribute**, **SQLColumns**, and **SQLDescribeCol** to understand the table schema in the database. For more information, see:
  https://msdn.microsoft.com/en-us/library/ms710150(v=vs.85).aspx
- The **common.ODBCTypes.SRC_SQLSERVER_SPATIALOBJ** and **common.ODBCTypes.SRC_DB2_BLOB** enums are internal Alteryx identifiers and are to be ignored.

Custom Lua Script Requirements

Each custom Lua script must include the following components. Refer to **Sample.lua** for guidance.

- First line: **common** = **require('./RuntimeData/ODBC/common')**
- An **AlteryxToC** table, which specifies mapping from an Alteryx data type to a C type.
    - **common.types**: specifies an Alteryx data type from which to map.
    - **c_type**: specifies the equivalent C type.
    - **c_length**: specifies the length of a C type.
    - **create_type**: specifies a database type.
    - **c_type** and **c_length** values are used for the ODBC API function **SQLBindCol**, which binds the data from Alteryx to columns in the database. For more information, see: https://msdn.microsoft.com/en-us/library/ms711010(v=vs.85).aspx
    - The create types are the actual database-specific identifiers used to build out queries, such as the create table statements.
- An **ODBCToAlteryx** table, which maps an ODBC SQL type to an Alteryx data type.
- The following four functions, which are used by Alteryx to return values from the **AlteryxToC** and **ODBCToAlteryx** tables. See Function Definitions.
    - getFieldInfo
    - getCType
    - getLength
    - getCreateType

Function Definitions

getFieldInfo(odbcInfo)

This function is used when Alteryx pulls data from a database. It maps the ODBC SQL type returned from a database to an Alteryx data type.

odbcInfo contains four parameters of information about the column returned from an ODBC API call. These parameters are optional and you should only use those that are needed to determine an appropriate Alteryx type. By default, the Alteryx type comes from the **ODBCToAlteryx** table defined earlier in the script.

- ODBC SQL type
- Length
- Precision – Used with numeric types such as Float and Double
- Unsigned

The return variable of this function must be named **alteryxType** and be a valid **common.types** enum.

getCType(altType)

This function is used by Alteryx to determine to which C type identifier a column should be bound when data is written from Alteryx to a database. The only parameter sent is an Alteryx data type.

The return variable of this function must be named **cType** and be a valid **common.CTypes** enum.

getLength(altType)

This function is used by Alteryx to determine the length of a bound column when data is written from Alteryx to a database. The only parameter sent is an Alteryx data type.

The return variable of this function must be named **cLength**.

getCreateType(colInfo)

This function is used by Alteryx to determine a database-specific data type for a column. It is the identifier that is used in such statements as create table.

colInfo contains four parameters of information that may be needed to determine the appropriate database type:

- Alteryx data type
- The column size
- The version of the database
- The version of the ODBC driver

Some of the create types contain words such as **size** and **scale** – for example, **varchar(size)**. Do not remove or replace with a value unless necessary, as Alteryx will replace them with appropriate values.

The return variable must be named **createType**.