Using emissary RT: Files

Overview

Emissary RT: Files is an ODBC driver designed to allow a user to query and update Windows files from an existing or custom made ODBC-enabled application. File information (such as filename, size, directory, attributes, and specialized meta data) is presented to the application as rows across a collection of related tables, and most standard SQL commands can be used to read and/or write this data.

By making use of emissary RT's lightweight, yet powerful Real-Time SQL engine, the latest file information is always shown, and updates take place immediately. No need for interim or temporary tables, programming code, or synchronization to accomplish SQL controlled interoperability with Windows files. In addition to reading standard file information, *emissary RT: Files* can also recognize image (JPG, GIF, PNG, BMP) and audio (WAV, MP3, OGG) related meta-data, such as resolution, resizing, comments, codec, duration, bitrate, *etc*.

Application Usage Examples

Examples can be performed via an ODBC-enabled application such as MS Access or Excel, using the GUI or SQL commands. No programming required.

- Delete all files of a certain size, age, type, extension, partial name, etc for cleanup
- Replace one set of file extensions with another
- Copy and resize all images above a certain resolution size to create thumbnails
- List all audio files below a certain bitrate or quality to note for conversion
- Selectively hide or make read-only multiple files recursively across multiple directories
- Convert all images of one type to another

Programming Usage Examples

Examples can be performed in your custom web or desktop application via an ODBC connection.

- Link file and meta-data related information to foreign tables in a Document Management System
- Import audio information in a music related application or database
- Batch display and/or convert images in a webapp photo album

Shown are only some of the possibilities. Many file-related tasks can be made easier via leveraging the power and flexibility of ODBC and SQL queries.

Available Types

File information is presented to the application as rows across a collection of related tables. These tables are as follows (follow links for full field listings and additional usage notes):

Table	Description
Standard	Standard information common to all files, such as name, size, attributes, etc
Image	Information related to image files, such as horizontal resolution, vertical resolution, comments, etc
Audio	Information related to audio files, such as codec, duration, channels , etc
Settings	Current data source configuration options, allowing run-time reconfiguration
Errors	Log containing errors generated from any query operations

Table - Standard

Field	Type	Read-only	Description
ID	Integer	Yes	Primary Key (unique ID) for row
Path	Char	No	Full pathname. Must be specified for INSERT statements. Will normally rename/move file when used in UPDATE statements, or copy the file when copy mode is enabled
Parent	Char	No	Parent directory. Will normally move file when used in UPDATE statements, or copy the file when copy mode is enabled
Name	Char	No	Filename. Will normally rename file when used in UPDATE statements, or copy the file when copy mode is enabled
Base	Char	No	Filename without extension. Will normally rename file when used in UPDATE statements, or copy the file when copy mode is enabled
Extension	Char	No	File extension (include leading dot). Will normally rename file when used in UPDATE statements, or copy the file when copy mode is enabled
Size	Integer	Yes	Filesize in bytes
CreateDate	Timestamp	No	File creation date/time
ModifyDate	Timestamp	No	File last modified date/time
Directory	Integer	Yes*	0 = File, 1 = Directory. *Must be specified for INSERT statements to create directories
ReadOnly	Integer	No	File Attribute: Read-only (0 = No, 1 = Yes)
Hidden	Integer	No	File Attribute: Hidden (0 = No, 1 = Yes)
System	Integer	No	File Attribute: System (0 = No, 1 = Yes)
Archive	Integer	No	File Attribute: Archive (0 = No, 1 = Yes)
Compressed	Integer	Yes	File Attribute: Compressed (0 = No, 1 = Yes)

Usage Notes

1. INSERT and DELETE statements run against the 'Standard' table will create and delete files/directories, respectively
2. As noted, the 'Path' and 'Directory' fields must be specified, at minimum, for INSERT statements. Any non read-only field may also be specified in INSERT statements to be applied to newly created files/directories
3. Changes made to fields will automatically reflect (upon the next query) to fields which contain aggregate information - *e.g.* if an UPDATE statement changes the 'Extension' column of a row, the 'Path' and 'Name' columns of that row will automatically reflect these changes in the next query run (and vice versa - changes to 'Name' would automatically reflect in 'Extension')

Table - Image

Field	Type	Read-only	Description
ID	Integer	Yes	Primary Key (unique ID) for row. Foreign Key to 'Standard' table
Format	Char	No	Image format type (JPG, GIF, PNG, BMP). Will convert file (and change extension) when used in UPDATE statements, or copy into new file when copy mode is enabled
HorizRes	Integer	No	Horizontal resolution in pixels. Will resize image (maintaining vertical resolution) when used in UPDATE statements
VertRes	Integer	No	Vertical resolution in pixels. Will resize image (maintaining horizontal resolution) when used in UPDATE statements
Size	Double	No	Current normalized size of image. Will grow/shrink an image when used in UPDATE statements. See notes below
Comment	Char	Yes	Embedded image comments (if supported)

Usage Notes

1. INSERT and DELETE statements may not be run against the 'Image' table. For creating and deleting files, please see the 'Standard' table
2. Changes made to the 'Size' field in UPDATE statements will grow or shrink the image, maintaining the aspect ratio, by a factor of the value specified. *E.g.* for an image originally 200x300, specifying 3 for 'Size' in an UPDATE statement would resize the image to 600x900. Specifying 0.5 for 'Size' in an UPDATE statement would resize the image to 100x150. A value of 1 will always be returned for 'Size' from a SELECT statement
3. Fields may not be populated if row corresponds to a directory, or non-image file type. The 'ID' field also serves as a foreign key into the 'Standard' table, and a JOIN may be used between the two tables

Table - Audio

Field	Type	Read-only	Description
ID	Integer	Yes	Primary Key (unique ID) for row. Foreign Key to 'Standard' table
Format	Char	Yes	Audio container format type (WAVE, MP3, OGG)
Codec	Char	Yes	Audio codec type (MPEG-1, Microsoft PCM, u-Law, Vorbis, etc)
Duration	Double	Yes	Length of audio in seconds
Channels	Integer	Yes	Channel count (1 = mono, 2 = stereo/dual mono)
SampleRate	Integer	Yes	Number of samples per second (Hz)
BitDepth	Integer	Yes	Number of bits per sample (where resolution applicable)
BitRate	Integer	Yes	Number of bits per second across all channels

Usage Notes

1. INSERT and DELETE statements may not be run against the 'Audio' table. For creating and deleting files, please see the 'Standard' table
2. In the case of variable bitrate audio, bitrate value will reflect the first sample found in the file
3. Fields may not be populated if row corresponds to a directory, or non-audio file type. The 'ID' field also serves as a foreign key into the 'Standard' table, and a JOIN may be used between the two tables

Table - Settings

Field	Type	Read-only	Description
Key	Char	Yes	Data source setting keyword. See Settings and Options
Setting	Char	No	Data source setting value. See notes
Description	Char	Yes	Full description of data source setting

Usage Notes

1. INSERT and DELETE statements may not be run against the 'Settings' table
2. Initial values of the 'Setting' field will reflect data source settings as configured from the ODBC manager
3. Changes made to 'Setting' field will immediately update data source settings for the duration of the connection (settings will revert to permanent values upon disconnect). Changes to Base Path or Recursion will cause *emissary RT: Files* to rebuild all caches

Table - Errors

Field	Type	Read-only	Description
ID	Integer	Yes	Primary Key (unique ID) for row
Date	Timestamp	Yes	Date/time the error occurred
Error	Char	Yes	Error description and details
Query	Char	Yes	SQL query that caused the error

Usage Notes

1. INSERT and DELETE statements may not be run against the 'Errors' table
2. The 'Errors' table is automatically purged prior to executing UPDATE, INSERT, or DELETE statements. The table should be checked for error details immediately after an unsuccessful query fails to execute

Supported SQL Syntax

SELECT Statement

```
SELECT select_expression [, select_expression . . .]
  [FROM table_expression
    [WHERE general_expression]
    [ORDER BY general_expression [ASC | DESC], . . .]
    [LIMIT [row_offset,] row_count ] ]
```

Note: SELECT statements used with a FROM command will retrieve data from the table(s) specified in the table_expression (see below). Usage without a FROM command will return a single row, executing any specified expressions in the select_expression (see below). At least one select_expression is required.

UPDATE Statement

```
UPDATE table_expression
  SET column1_name=general_expression [, column2_name=general_expression . . .]
  [WHERE where_expression]
  [ORDER BY order_expression [ASC | DESC], . . .]
  [LIMIT [row_offset,] row_count ]
```

Note: UPDATE statements used with an ORDER BY command will control the order in which file operations are performed. This can be useful if the order of updating filenames may otherwise cause a name collision with pre-existing files. If the SET command includes any expressions with column names, the value of the field in the currently updating row will be used. LIMIT will constraint which files are updated from the total UPDATE rowset.

INSERT Statement

```
INSERT INTO table_name
  [(column1_name, ...)]
  {VALUES | VALUE} (general_expression, . . .)
```

```
INSERT INTO table_name
  SET column1_name=general_expression [, column2_name=general_expression . . .]
```

Note: INSERT statements may use either syntax shown above. If the first syntax is used without specifying column names, the number of VALUES/VALUE expressions specified must equal the number of columns in the table. For read only fields, the value specified is ignored.

DELETE Statement

```
DELETE {table_name[.*] | *}
  FROM table_expression
```

```
    [WHERE general_expression]
    [ORDER BY general_expression [ASC | DESC], . . .]
    [LIMIT [row_offset,] row_count ]
```

Note: DELETE statements used with an ORDER BY command will control the order in which file operations are performed. LIMIT will constraint which files are deleted from the total DELETE rowset.

Select Expressions

```
{general_expression | [table_name.] { * | column_name} } [[AS] alias]
```

Note: Table and column names may be delineated using the ` character.

Table Expressions

```
{table1_name} [[AS] alias] [, {table2_name} [ [AS] alias] . . .]
  [[INNER | LEFT [OUTER] | CROSS] JOIN table_name
    [ON general_expression] . . .]
```

Note: Table and column names may be delineated using the ` character. Comma separated tables specified after the first table in a table_expression before JOIN commands will be treated as CROSS JOINed tables.

General Expression Operators and Functions

Literal	Operands/Arguments	Precedence	Description	
=	binary	1	Assign. Recognized in UPDATE and INSERT statements	
=	binary	7	Equal. Case insensitive for strings, case sensitive for BINARY type. Returns boolean value (0 = false, 1 = true)	
		binary	7	Not equal. Case insensitive for strings, case sensitive for BINARY type. Returns boolean value (0 = false, 1 = true)
>	binary	7	Greater than. Case insensitive for strings, case sensitive for BINARY type. Returns boolean value (0 = false, 1 = true)	
			Greater than or equal. Case	

>=	binary	7	insensitive for strings, case sensitive for BINARY type. Returns boolean value (0 = false, 1 = true)
<	binary	7	Less than. Case insensitive for strings, case sensitive for BINARY type. Returns boolean value (0 = false, 1 = true)
<=	binary	7	Less than or equal. Case insensitive for strings, case sensitive for BINARY type. Returns boolean value (0 = false, 1 = true)
LIKE	binary	7	String comparison with wildcard matching. '%' matches 0 or more characters. '_' matches 1 character. Case insensitive for strings, case sensitive for BINARY type. Returns boolean value (0 = false, 1 = true)
+	binary	11	Add. Parses strings to numeric equivalent.
-	binary	11	Subtract. Parses strings to numeric equivalent.
*	binary	12	Multiply. Parses strings to numeric equivalent.
/	binary	12	Division. Parses strings to numeric equivalent.
%	binary	12	Modulo. Parses strings to numeric equivalent.
IS	binary	7	Equal (NULL safe). Case insensitive for strings, case sensitive for BINARY type. Returns boolean value (0 = false, 1 = true)
IS NOT	binary	7	Not equal (NULL safe). Case insensitive for strings, case sensitive for BINARY type. Returns boolean value (0 = false, 1 = true)
AND	binary	2	Logical AND. Returns boolean value (0 = false, 1 = true)
OR	binary	4	Logical OR. Returns boolean value (0 = false, 1 = true)

NOT	unary	5	Logical NOT. Returns boolean value (0 = false, 1 = true)
CONCAT	Variable	Function	String concatenation. CONCAT(string1, ...)
CONVERT	2	Function	Type conversion. CONVERT(value, type)
LOCATE	2/3	Function	Return starting position of substring. LOCATE(substring, full string, [start index])
SUBSTR	2/3	Function	Return substring. SUBSTR(string, [start index,] num of chars)

SQL Types

CHAR, VARCHAR, LONG VARCHAR, BINARY, SMALLINT, INTEGER, FLOAT, DOUBLE, TIMESTAMP

Miscellaneous ODBC Support

"{d '1995-01-15'}" style date literals, unnamed parameters via '?' literals, single prepare/multiple execution model with parameter updating, thread-safety. Contact Synthetic Dreams regarding any further ODBC support questions.

Performance and Considerations

To increase efficiency of processing SQL queries against the file system, *emissary RT: Files* makes use of an in-memory caching system. This cache (if enabled in the Data Source options) monitors the file system and is updated in real-time should files be changed. This cache is initially built when the ODBC connection is established, and is maintained for the lifetime of the connection. If real-time updates are disabled, *emissary RT: Files* will rescan and update its cache (if necessary) when it executes a SQL query. Due to being more resource intensive, image and audio information is only cached/updated if the SQL query involves those tables, respectively.

When making use of *emissary RT: Files* in custom applications, as the cache is built at connection time and is maintained for the lifetime of the connection, it is important to reuse the ODBC connection when possible. This can be more challenging in a web application environment, and may require changes to both the web server configuration and API used. An example includes PHP's odbc_pconnect function and a compatible Apache configuration (non-CGI mode), which creates a persistent connection across each request (for the session lifetime).

When executing queries containing related tables (either via JOIN or appropriate WHERE clauses), *emissary RT: Files* is optimized for predicates comparing the equality of Image.ID and/or Audio.ID against Standard.ID. *E.g.* "SELECT *FROM Standard INNER JOIN Audio ON Standard.ID = Audio.ID", "SELECT* FROM Image, Standard WHERE Image.ID = Standard.ID", "SELECT * FROM Standard, Image, Audio WHERE Standard.ID = Image.ID AND Standard.ID = Audio.ID", *etc*. Comparisons of Image.ID directly to Audio.ID are not optimized. Predicates may contain additional expressions, as long as OR operators do not allow for potential additional matches in the join. Any non-optimized predicate with valid syntax may be used, but performance will degrade significantly, as the system must internally perform a full cross join.

Additionally, all tables are indexed against their ID column, and will perform significantly faster with WHERE clauses that select for specific IDs, via inline values and/or parameters. As with related table optimization above, WHERE clauses optimized for ID indices may contain additional expressions, as long as OR operators do not allow for potenital additional matches.

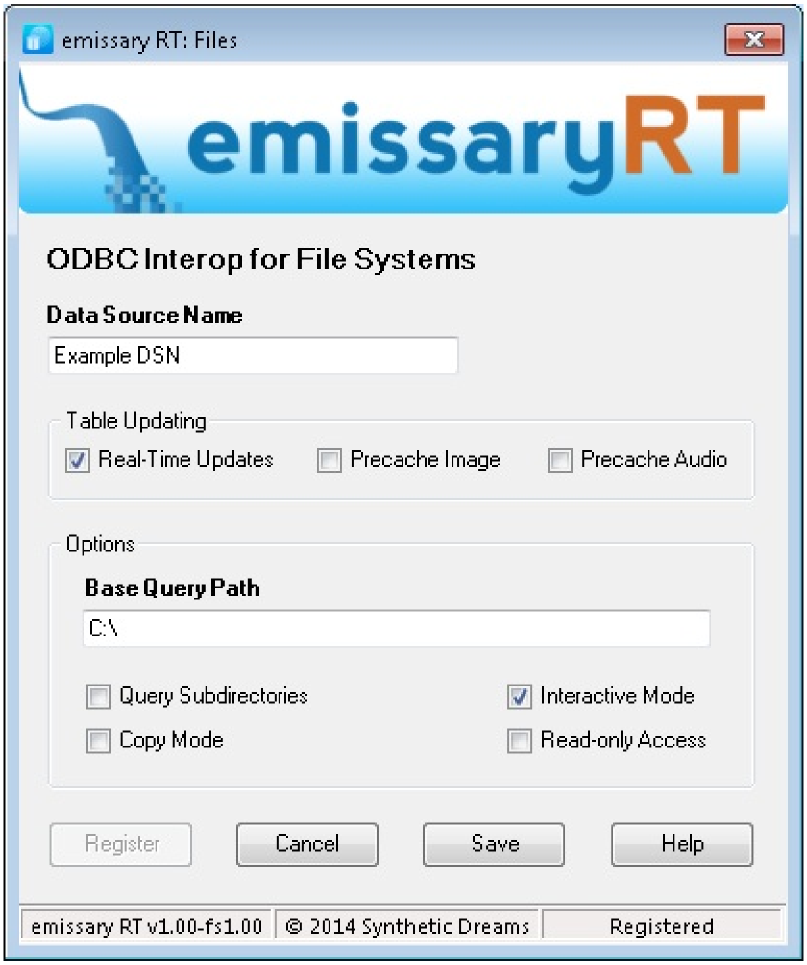
The execution time involved for a SQL query is dependent on the size of the file system (especially if recursion is enabled), the speed of the underlying hardware in querying the file system, if image or audio data is being queried, *etc*. Because it may be desired to execute a query that may take a significant time to process, *emissary RT: Files* provides an interactive mode that shows both a progress meter, and allows the cancelation of a query. Note - canceling an INSERT, UPDATE or DELETE query is not ACID compliant - changes are made to the file system in real-time, and are not automatically rolled back. Please construct a corresponding SELECT query for testing before executing any potentially destructive INSERT, UPDATE or DELETE queries.

Configuring a Data Source

Before making use of *emissary RT: Files*, it is necessary to configure one or more desired data sources. A data source, most importantly, indicates which Base Query Path *emissary RT: Files* will query, if it will recursively query all subfolders, if the access is read-only, *etc*. A full list of all options and their effect can be found in the Settings and Options guide.

DSNs can be created, configured, or deleted from the Microsoft ODBC Manager, typically found in the Administrative Tools menu. It is also important to use the correct version of the ODBC Manager depending on the architecture (32/64bit) of the ODBC-enabled application. Please refer to Microsoft's documention for further details.

Minimally, an *emissary RT: Files* data source must be configured with a data source name and base directory defined, as shown below (with the default options enabled).



ODBC Interop for File Systems

Data Source Name
Example DSN

Table Updating
☑ Real-Time Updates ☐ Precache Image ☐ Precache Audio

Options
Base Query Path
C:\

☐ Query Subdirectories ☑ Interactive Mode
☐ Copy Mode ☐ Read-only Access

Settings and Options

The following attributes may be configured for each data source (and may be reconfigured at run-time):

Option	Settings Key	Description
RealTime Updates	RealTime	"True" = The Base Query Path (and subdirectories if Query Subdirectories is enabled) will be monitored for file updates, resulting in faster queries. "False" = The file system cache will only be updated at query time. A value of "False" should be used if the path being searched will experience a large number of changes while *emissary RT: Files* is being used (e.g. many file updates/creations/deletions/etc), and only a small amount of queries will be executed.
Precache Image	PrecacheImage	"True" = Upon connecting to the data source (or when updating the Base Serach Path at run-time), the Image table (contaning image-related data) will immediately be cached for all image files, resulting in slower connect time, but a faster initial query involving images. "False" = the Image table will not be cached until a query involving it is executed, resulting in faster connection time, but a slower initial query involving images.
Precache Audio	PrecacheAudio	"True" = Upon connecting to the data source (or when updating the Base Serach Path at run-time), the Audio table (contaning audio-related data) will immediately be cached for all audio files, resulting in slower connect time, but a faster initial query involving audio. "False" = the Audio table will not be cached until a query involving it is executed, resulting in faster connection time, but a slower initial query involving audio.
Base Query Path	BasePath	The directory to execute queries against. For example, after specifying "C:\temp" as a Base Query Path, running a SELECT query would return folders and files within "C:\temp" (and subfolders if Query Subdirectories is enabled), and running a DELETE query would delete folders and files within "C:\temp". This field is required, and must be a valid directory.
Query Subdirectories	Recurse	"True" = Subdirectories under the Base Query Path will also be included when executing a query. "False" = Only folders and files within the Base Query Path will be included when executing a query.
Copy Mode	Copy	"True" = When performing an UPDATE query on the "Standard" table that changes the value of "Path", "Name", "Parent", "Base", or "Extension", or on the "Image" table that changes the value of "Format", the corresponding file will be copied with the new pathname/extension, leaving the original file. "False" = When performing an UPDATE query on the "Standard" table that changes the value of "Path", "Name", "Parent", "Base", or "Extension", or on the "Image" table that changes the value of "Format", the corresponding file will be renamed with the new pathname information.

Interactive Mode	Interactive	"True" = When updating the cache, or executing a INSERT, UPDATE, or DELETE query, a progress meter will be shown to the user. The dialog will allow users to cancel queries. "False" = no dialog boxes will be displayed to the user. "False" must be used when making use of *emissary RT: Files* in custom applications that cannot interact with the desktop, such as PHP or ASP.NET web apps.
Read-only Access	ReadOnly	"True" = INSERT, UPDATE, and DELETE queries are disabled. "False" = INSERT, UPDATE, and DELETE queries are enabled.

Reconfiguration at Run-Time

In addition to specifying configuration options for an *emissary RT: Files* data source, these settings may also be changed during run-time if desired. Any run-time changes made will immediately affect the ODBC connection, but are temporary for that connection only. Future connections will use the settings as defined by the DSN.

To reconfigure a data source at run-time, the "Settings" table may be queried and updated. Each row of the table contains a Key and Value corresponding to a data source configuration option. A full list of all options and their effect can be found in the Settings and Options guide. Note: changing the RealTime, BasePath, or Recursion options may cause a rebuild of the internal cache.

Registering emissary RT: Files

For trialing purposes, *emissary RT: Files* may be freely downloaded and used. When unregistered, the system is fully functional with the exception of imposing a limit of returning and/or affecting 50 records for any query. Once purchased and registered, this restriction is lifted.

Registering *emissary RT: Files* may be performed from any DSN configuration dialog, using the "Register" button. The system will prompt for the license key received when purchasing the product, and can register automatically via the Internet, or manually via email by following the provided instructions. Manual registration may be performed on a different machine than where *emissary RT: Files* is installed.