Using Query Database the LASP IDL-DB Interface software

The query_database IDL procedure

The purpose of the query_database procedure is to execute an SQL command that either retrieves (**"select**") data from a database or alters the contents of the database ("**insert**", "**update**", and "**delete**").

The query_database procedure has also been used with stored procedures. At present there is an outstanding issue QUERYDB-100 that will be fixed in the next release (version 2.3).

The IDL **query_database** procedure, based upon the IDL-Java Bridge, allows one query to be sent and processed. For "**select**" queries, the variable ***nrows*** contains the number of rows of data returned. If this is greater than 0, the results are returned in the variable ***data***, as an array of structures, each structure corresponding to one row. For "**insert**", "**update**", and "**delete**" operations, ***nrows*** returns the number of rows affected.

Login credentials

To do any of these operations requires login credentials to the database.

Note that the login process can use a text file found in any of several locations (which will be read by the *fjava_read_loginfile* procedure). The order of precedence for login text files is:

1. a file specified by the keyword "**dbloginfile**" in the query_database call
2. a file specified by the environment variable "**DB_LOGIN_FILE**"
3. a file in the current working directory named "**.qdbResources**"
4. a file in the current working directory named "**.dbLogin**"
5. a file in the user's home directory named "**.qdbResources**"
6. a file in the user's home directory named "**.dbLogin**"

The files named "**.qdbResources**" are assumed to be multi database resource specifications.

The files named "**.dbLogin**" are assumed to be single resource specifications and are what query_database users have worked with for all releases prior to 2.1.

The detailed specification for correctly writing these files is available at Query Database single and multi-database resource file formats

Specifying resources

When LASP used Sybase almost exclusively the standard was to specify a server and database (schema) name because these could be used to generate the URL that was necessary for the actual database connection. While **query_database** maintains this legacy compatibility, LASP's shift to Oracle has made this more complicated, and it is now preferable to specify a URL rather than a server / database pair.

query_database does maintain a set of tables that map servers, database, urls, and drivers. A **query_database** user going to a known LASP Sybase or Oracle server will NOT need to specify a driver because the software handles this automatically. This set of supported servers is at the bottom of this page.

query_database can be used with databases and drivers other than Sybase or Oracle, but the user must explicitly provide the URL and the name of the driver jar, as well as the jar itself using the dbDriver and dbUrl keywords.

Multiple queries

In general ***concatenating multiple queries is bad programming practice and should be avoided***. At the present time, however Sybase does allow concatenation of multiple queries separated by the unix newline character, which is represented in IDL as *string(10B)*. Oracle presently does not. The Sybase implementation with the jconn3.jar (not the most recent Sybase offering, but the most commonly used in LASP) produces strange results that do not lend themselves to mechanized processing. I.e., under certain circumstances multiple copies of the JDBC ResultSet are returned for the second query. For this reason, although multiple queries are legal under Sybase, Query Database does not support them, and will return only the results from the first in the sequence of queries.

Multiple newline delimited queries are permitted for calls to the Sybase databases. The analogous call (i.e. changing the database, instrument mode id, etc.) will NOT run on an Oracle database. Although the following call does not produce an error message, only the results from the first query are returned.

```
query = "select max(version) from CalibrationMetadata "$
                + "where calibrationTableName = 'HeaterNodeResistanceCal'
"$
                + "and instrumentModeId = 29" + string(10B)$
                + " select max(version) from CalibrationMetadata "$
                + "where calibrationTableName = 'ApertureAreaCalibration'
"$
                + "and instrumentModeId = 29"
query_database, query, data, nrows, database='SORCE_TIM_V16'
```

The following is far better form. Eg., the following will work and will produce the desired results from both queries, as will its Oracle counterpart.

```
query = "select max(version) from CalibrationMetadata "$
                + "where calibrationTableName = 'HeaterNodeResistanceCal'
"$
                + "and instrumentModeId = 29"
query_database, query, data, nrows, database='SORCE_TIM_V16'

query = "select max(version) from CalibrationMetadata "$
                + "where calibrationTableName = 'ApertureAreaCalibration'
"$
                + "and instrumentModeId = 29"
query_database, query, data, nrows, database='SORCE_TIM_V16'
```

Arguments and keywords for query database

The following is taken from the IDL documentation

```
; CALLING SEQUENCE:
;
;    query_database, $
;       sqlString, $          ; the SQL command
;       data, $               ; the data to be returned (output)
;       nrows $               ; the number of rows in the data (output)
;       [, limit_rows=limit_rows] $
;       [, debug=debug] $
;       [, database=database] $
;       [, server=server] $
;       [, user=user] $
;       [, password=password] $
;       [, max_image_length=max_image_length] $
;       [, RESET_LOGIN=RESET_LOGIN] $
;       [, GET_LOGIN=GET_LOGIN] $
;       [, LOGIN_INFO=LOGIN_INFO] $
;       [, dbloginfile=dbloginfile] $
;       [, dbResourceId=dbResourceId] $
;       [, dbDriver=dbDriver] $
;       [, dbUrl=dbUrl] $
;       [, dbConnect=dbConnect] $
;       [, dbClose=dbClose]
;
; INPUTS:
;   sqlString - a scalar string containing the SQL query statement that
;               should be used to retrieve data from the database.
;
; KEYWORD INPUTS:
;   limit_rows - limits the number of rows of data returned by a query.
The
;      number of rows returned will less than or equal to this number.
;   debug - prints debugging information to standard output.
;   database - specifies the database or instance to use.  (Overrides any
;      entry in the login text file.)
;   server - specifies the database server to use.  (Overrides any
;      entry in the login text file.)
;   user - specifies the database user.  (Overrides any
;      entry in the login text file.)
;   password - specifies the user's password.  (Overrides any
;      entry in the login text file.)
;   max_image_length - specifies the maximum length that an image or text
;      datatype can be.  (This is mainly used for Sybase ASE servers.)
;   RESET_LOGIN - closes any existing connection and reconnects using
;      the current passed parameters.
;   GET_LOGIN - puts the current login information into the LOGIN_INFO
keyword.
;   dbloginfile - specifies the login text file to use.  Overrides the
;      default login text file.
;   dbResourceId - this specifies a group of items in a multi-database
resource file.
;   in_dbDriver - specifies the JDBC driver to use. (Overrides any entry in
the
;      login text file.)
;   in_dbUrl - specifies the database URL that the JDBC API uses to connect
;      to a database. (Overrides any entry in the login text file.)
;   dbConnect - indicates that a connection should be held open until the
;      dbClose keyword is set.  In standard operation (dbConnect is not
set),
;      a connection is closed after each SQL command.  Setting this keyword
;      keeps the connection open, which reduces connection overhead and
;      is useful when using one database.  Note that if dbConnect is set,
;      subsequent login information is ignored until the RESET_LOGIN or
;      dbClose keywords are used. Note that dbConnect is NOT compatible
;      with using a resource id for multi database connections.
;   dbClose - closes the connection and unsets the dbConnect status.  This
;      keywords precedes all other keywords, and no SQL commands will be
;      executed when this keyword is set.
;
; OUTPUTS:
;   data - The array of structures that contain the return data from
"select"
;          queries.  If no data are returned, either because no rows
matched
;          the SQL query or because an "insert", "update", or "delete"
;          operation rather than a "select" was performed, the content of
;          this variable is not specified.  For "select" queries, always
;          check the value of nrows to be sure that this variable is not
;          undefined.
;
;   nrows - the number of rows of data returned by the query.
;           For "select" queries this will equal the number of
;             elements of data in the data variable.
;           For "insert", "update", or "delete" this is the number
;             of rows affected.
;
; KEYWORD OUTPUTS:
;   LOGIN_INFO - if the GET_LOGIN keyword is set, returns the current login
;      information in a structure = {server, database, user, password}
;
; COMMON BLOCK:
;   query_database
;
; DEPENDENCIES:
;    idldb.jar - contains the Java JDBC code to interface with
;       database servers (e.g., /<full_path>/lasp_dbexchange.jar)
;    ojava_dbexchange - implements the IDL-Java Bridge
;    <JDBC driver class> - a JDBC driver to connect to a database server
(e.g.,
;       /<full_path>/jconn3.jar)
;    <Java Virtual Machine> - JVMs are usually part of a Java installation.
;       Also, the environmental variables or configuration files need to be
set
;       properly.  (See the IDL documentation on the IDL-Java Bridge.)
;
; EXAMPLE:
;    - Uses information from the .dbLogin file to supply user, password,
;    -   and server.
;    - Data is returned in the data parameter and nrows is set.
;    query_database, "select * from TestTable3", data, nrows, $
;       database="SORCE_TEST"
;
;    - Resets the connection to a held connection using keyword parameters
;    - No SQL command is issued
;    query_database, /dbConnect, /reset_login, user="rept_user", $
;       password="rept_password", database="REPT_CT", server="rept-db"
;
;    - Releases the current database connection and resets login
```

```
  information.
  ;    query_database, /dbClose
  ;
```

The dbConnect keyword

The connection can be closed after each call, retaining login information, or held open by using the **dbConnect** keyword.

Subsequent calls use the retained login information to make a connection; any login information (user, password, server, database) as a keyword will override retained login information (unless **dbConnect** is used.)

The **dbConnect** keyword will retain a connection from call to call. Hence, any login information provided in subsequent calls after the first call will be ignored; such a connection can only be closed by using the **dbClose** keyword or the **RESET_LOGIN** keyword.

Usage examples

1. - Uses information from the .dbLogin file or from a previous call to supply connection information

Note that this example assumes the database connection has already been established, either through the presence of a .dbLogin file, such as the sample below, or through a previous query_database call. In this example, the SQL query "select * from Columns" returns thirty-five (35) rows of data.

```
[.dbLogin file]
   user     = testuser
   password = <password>
   url = jdbc:sybase:Tds:sorce-db:4100/SORCE
   driver = com.sybase.jdbc3.jdbc.SybDriver

IDL> query_database, "select * from Columns", data, nrows
   DATA            STRUCT    = -> <Anonymous> Array[35]
   NROWS           LONG      =          35
```

2. - Establish the connection for subsequent calls

Because the SQL command, data, and nrows are not used in this call to query_database, they will retain their values from the previous query_database calls or remain undefined if there were no previous query_database calls.

```
IDL> query_database, /dbConnect, user="rept_user", password="rept_password",
database="REPT_CT", server="rept-db"
```

3. - Releases the current database connection and resets login information.

Note that the sql command is an empty string because it will not be executed.

```
IDL> sql = ""
IDL> query_database, sql, data, nrows, /dbClose
   DATA            UNDEFINED = <Undefined>
   NROWS           INT       =        0
```

4. - Uses the dbUrl and dbDriver keywords

Note that the following code snippets are intended as examples. Their actual behavior in a program will depend on whether resource files (.dbLogin, .qdbResources, etc.) are present, and on what, if any, settings may have been changed by previous query_database calls.

For Sybase connections, port 4100 is standard. Note that it is not necessary to specify the driver for Sybase connections because query_database automatically uses the correct Sybase driver.

```
IDL> user = "testuser"
 IDL> password = "testpassword"
 IDL> host = "sorce-db"
 IDL> database = "SORCE"
 IDL> dbdriver = 'com.sybase.jdbc3.jdbc.SybDriver'
 IDL> dburl = 'jdbc:sybase:Tds:'+host+':4100/'+database
 IDL> ; expanded, dburl = 'jdbc:sybase:Tds:sorce-db:4100/SORCE'
 IDL> qry = 'select getdate()'
 IDL> query_database, qry, data, nrows, user=user, $
 IDL>   password=password, dbdriver=dbdriver, dburl=dburl
```

For Oracle connections, port 1521 is the default. Note that it is not necessary to specify the driver for Oracle connections because query_database automatically uses the correct Oracle driver. Note also that there are slight differences in the url syntax between Oracle and Sybase.

```
IDL> user = "testuser"
 IDL> password = "testpassword"
 IDL> host = 'lasp-db-dev'
 IDL> service = 'lasp-db-dev'
 IDL> url = 'jdbc:oracle:thin:@'+host+':1521/'+service
 IDL> ; expanded, url = 'jdbc:oracle:thin:@lasp-db-dev:1521/lasp-db-dev'
 IDL> driver = 'oracle.jdbc.driver.OracleDriver'
 IDL> qry = 'select * from DUAL'
 IDL> query_database, qry, data, nrows, user=user, $
 IDL>   password=password, dburl=url, dbdriver=driver
```

LASP Servers currently recognized by query_database

query_database currently recognizes all of the following servers and associates either the Sybase or the Oracle driver with them as follows.

Sybase servers

1. aim-db
2. cassini-db
3. cassini-db-dev
4. eve-db
5. eve-db-dev
6. glory-db
7. glory-db-dev
8. icesat-db
9. lasp-db-dev
10. kepler-db
11. kepler-db-dev
12. lapis-db
13. lisird-db
14. moc-db
15. moc-db-dev
16. ps-db
17. ps-db-dev
18. quikscat-db
19. rept-db
20. sdc-db
21. snoe-db
22. sorce-db

23. sorce-db-dev
24. tqsm-db
25. uars-db

Oracle servers

1. goesr-db
2. goesr-db-dev
3. ldex-db
4. ldex-db-dev
5. maven-db
6. maven-db-dev
7. tcte-db
8. tcte-db-dev
9. tsis-db
10. tsis-db-dev