NAME

Sybtcl - Sybase SQL Server access commands for Tcl

INTRODUCTION

Sybtcl is a collection of Tcl commands and a Tcl global array that provides access to a Sybase Server. Each Sybtcl command generally invokes several Sybase Open Client (a.k.a. DB-Library) library functions. Programmers using Sybtcl should be familar with basic concepts of DB-Library programming.

SYBTCL COMMANDS

sybconnect *login-name password ?server? ?appname? ?ifile? ?charset?*

Connect to a Sybase server using *login-name* and *password*. If *server* is specified, then an attempt will be made to login to the named Sybase server. If *server* is not specified, then the environment variable *DSQUERY* is used to determine a server; if *DSQUERY* is not set, sybconnect will try to connect to the Sybase server named *SYBASE*. If an *appname* is specified, then that value is passed to the server to display during a process list. If *ifile* is specified, that file is used to resovle server addresses; if not specified, the normal $SYBASE/interfaces file is used. If *charset* is specified, the charset should reference a vaild character set available on your Sybase installation. See the $SYBASE/charsets directory for charsets supported on your installation.

Login-name and *password* are required parameters. All other parameters are optional postitional parameters. For example, to specify the *ifile* paramater, *server* and *appname* must also be specified. Optional parameters may be a null string, in which case the default value is used.

A handle is returned and should be used for all other Sybtcl commands using this connection. Multiple connections to the same or different servers are allowed, up to a maximum of 50 total connections. Sybconnect raises a Tcl error if the connection is not made for any reason (login or password incorrect, server not found in the Sybase interfaces file, network unavailable, etc.).

syberrhandler *?tclproc?*

Set a Tcl procedure to be called upon any DB-Lib error. The Tcl procedure should exist, and take the following arguments to the procedure:

handle
: The Sybtcl handle associated with the error condition. Handle may be a null string, such as in the case of an error occurring during sybconnect.

severity The Sybase severity level of the error.

dberr The Sybase DB-Lib error code.

oserr The operating system error code.

dberrstr
: The DB-Lib error string associated with the error code.

oserrstr
: The operating system error string associated with the error.

Example:

```
proc err_handler {handle severity dberr oserr dberrstr oserrstr} {
   puts "error: $handle $severity $dberr $oserr $dberrstr $oserrstr"
   return 1
}
syberrhandler err_handler
```

The procedure may also return an integer value to indicate an action to perform:

"0" print an error message and abort the program immediately.

"1" continue by returning a "FAIL" condition to the currently executing DB-Lib function.

"2" cancel the operation that caused the error if a result of a timeout condition. For any other case, this value is considered as "0", immediate program termination.

"3" continue to wait for one additional time out period if a result of a timeout condition. For any other case, this value is considered as "0", immediate program termination.

If no value is returned, or any other value is return, Sybtcl treats the return value as "1" (continue).

Syberrhandler called without an argument returns the current error handler. If called with a null string, the error handler is disabled.

sybmsghandler *?tclproc?*

Set a Tcl procedure to be called upon any server message. The Tcl procedure should exist, and take the following arguments to the procedure:

handle
The Sybtcl handle associated with the error condition.

msgno The Sybase server message number.

msgstate
The Sybase server message state.

severity The Sybase severity level of the error.

msgtext The server message text.

svrname
The name of the server generating the message.

procname
The name of the procedure generating the message, if any.

line The line numer of the procedure generating the message, if any.

Example:

```
proc msg_handler {handle msgno msgstate severity msgtext svrname procname line} {
   puts "msg: $handle $msgno $msgstate $severity $msgtext $svrname $procname $line"
}
syberrhandler msg_handler
```

Sybghandler called without an argument returns the current message handler. If called with a null string, the message handler is disabled.

sybuse *handle ?dbname?*

Return the database name currently in use. If *dbname* is specified, then attempt to use the named database. *Handle* must be a valid handle previously opened with sybconnect. If *dbname* is used successfully, the dbname is returned. Sybuse raises a Tcl error if the handle specified is not open or the database name specified could not be used.

sybsql *handle sql-command ?-async?*

Send the Sybase Transact SQL statements *sql-command* to the server. *Handle* must be a valid handle previously opened with sybconnect. If the optional argument **-async** is specified, then the SQL is sent to the server without waiting for a response and sybsql will return immediately with the value "PENDING". The sybpoll command can be used to check the status of the server results; sybnext must be called to determine if result rows were generated. If **-async** is omitted, then sybsql will wait until the server has responed. Sybsql will return "REG_ROW" if the SQL

statements generated return rows, "NO_MORE_ROWS" if the SQL commands executed, but no rows were returned in the first or only set of results. The **sybmsg** array index *retstatus* is set with the return code of a stored procedure, if any; the *nextow* index is also set to the value returned by sybsql.

Multiple SQL statements may be specified in *sql-command*. *Sybnext* allows retrieval of return rows generated; *Sybretval* allows retrieval of any return values from stored procedures. See notes regarding stored procedure output variables.

Sybsql performs an implicit *sybcancel* if any results are still pending from the last execution of sybsql. Sybsql raises a Tcl error if the handle specified is not open, or if the SQL commands are syntactically incorrect.

Table inserts made with sybsql should follow conversion rules in the Sybase Commands Reference manual (image or binary data is hexadecimal string preceded with "0x"; datetime should be a recognizable date, etc. The SQL Server CONVERT function may be used to force conversions.

sybpoll *handle ?timeout? ?-all?*

Return a list of Sybtcl handles that have results waiting. *Handle* must be a valid handle previously opened with sybconnect. If the last SQL results are not ready, a null string is returned. An optional *timeout* value in milliseconds may be specified, which is the amount of time the sybpoll will wait before returning a result. If the timeout value is -1, sybpoll will wait until results are available before returning. The default timeout value is 0, which polls and immediately returns.

The option *-all* may be specified, in which all handles that have been executed with -async are checked, and a Tcl list of all handles that have results waiting are returned. When *-all* is combined with a timeout of -1, then sybpoll waits until any async handle has results waiting, and then returns a list of all handles that have results waiting. If the last SQL statements executed with *handle* were not sent as -async with sybsql, sybpoll returns a null string.

sybevent *handle ?script?*

Arrange to run a script when server data is available. *Handle* must be a valid handle previously opened with sybconnect. *Sybsql* must have previously been executed, without or without *-async* mode. The event handler script should execute *sybnext* in order to process available data. The event handler may be executed during any phase of result processing including REG_ROW, compute rows, NO_MORE_ROWS, and NO_MORE_RESULTS.

If *?script?* is not specified, the existing event handler script is returned. If *?script?* is a null string, the event handler is removed. The event handler is also removed at end of all results for the current SQL statement.

For the event handler to actually run, the Tcl interpreter must be processing events. Events can be processed on demand by executing the Tcl **update** command, or until a variable is set with the **vwait** command. Events are also processed while a Tk (wish) program is waiting on user input.

Sybtcl's callback handler is greedy; it continues to invoke the sybevent script while data rows are available. To allow other Tcl events to be processed, set *sybmsg(bgevents)* to **idletasks** or **all**.

sybnext *handle ?commands? ?substitution_character? ?tclvar colnum ...?*

Return the next row from the last SQL statements executed with sybsql as a Tcl list. *Handle* must be a valid handle previously opened with sybconnect. Sybnext raises a Tcl error if the handle specified is not open. A null string is returned if there are no more rows in the current set of results. The Tcl list that is returned by sybnext contains the values of the selected columns in the

order specified by *select*.

If the SQL statements where executed with the -async option of sybsql, then sybnext will wait until results are available. Sybpoll may be used to check for results in a non-blocking manner. Any errors in the SQL statements will cause sybnext to fail.

The optional *commands* argument allows sybnext to repeatedly fetch rows and execute *commands* for each row. Substitutions are made on *commands* before passing it to Tcl_Eval() for each row. An optional argument consisting of a single character can be specified for a column number substitution character. If none is specified, the character '@' will be used to denote the substitution character. If the substitution character is a null string, no column substitutions will be performed on the *commands* string. Sybnext interprets the substitution character followed by a number (**@n**) in *commands* as a result column specification. For example, **@1, @2, @3** refer to the first, second, and third columns in the result. **@0** refers to the entire result row, as a Tcl list. Substitution columns may appear in any order, or more than once in the same command. Substituted columns are inserted into the *commands* string as proper list elements, i.e., one space will be added before and after the substitution and column values with embedded spaces are enclosed by {} if needed.

Tcl variables may also be set for *commands* on each row that is processed. Tcl variables are specified after the *substitution_character*, consisting of matching pairs of Tcl variable names and a column numbers. Column number may be "0", in which the Tcl variable is set to the entire result row as a Tcl list. Column numbers must be less than or equal to the number of columns in the SQL result set.

Tcl variable column pairs may also be specified as a list argument.

Sybnext will execute *commands* until NO_MORE_ROWS. If additional results are pending, subsequent sybnext commands will retrieve the next set of results.

A Tcl error is raised if a column substitution number is greater than the number of columns in the results. Note that Transact-SQL "compute" statements are considered to be part of the current select's result set, and thus, a different number of columns may be returned, causing the sybnext column substitution to fail when the compute row is returned. If the commands execute **break**, sybnext execution is interrupted and returns with TCL_OK. Remaining rows may be fetched with a subsequent sybnext command. If the commands execute **return** or **continue**, the remaining commands are skipped and sybnext execution continues with the next row. Sybnext will raise a Tcl error if the *commands* return an error. Commands should be enclosed in "" or {}.

The **sybmsg** array index *retstatus* is set with the return code of a stored procedure, if one was executed in the last SQL command to sybsql; the index *nextrow* is set to one of several values, depending on the results of sybnext. Refer to the section "SERVER MESSAGE AND ERROR INFORMATION" for information about how the *nextrow* value is set.

When *sybmsg* array element *binaryashex* is set to '1', 'yes', or 'true', sybnext performs conversions for image and binary data. Data is returned as a hexadecimal string, without a leading "0x". Use the SQL Server function CONVERT to force a specific conversion.

The **sybmsg** array index *maxtext* limits the amount of text or image data returned for each column returned. The default is 32768 bytes.

The **sybmsg** array index *nullvalue* can be set to specify the value returned when a column is null. The default is "0" for numeric data, and "" for other datatypes.

sybcols *handle*

Return the names of the columns from the last sybnext or sybretval command as a Tcl list. Sybcols returns the column name used in the SQL select command; a null string for any column that is an aggregate function (count, sum, avg, min, max) in a regular row. A compute row column name is returned as *function(column-name)*. Sybcols may be used after *sybretval*, in which the output variable names are returned (see notes).

The **sybmsg** array index *collengths* is set to a Tcl list corresponding to the lengths of the columns; index *coltypes* is set to a Tcl list corresponding to the types of the columns. Sybcols raises a Tcl error if the handle specified is not open.

sybcancel *handle*

Cancel any pending results from the last sybsql command. *Handle* must be a valid handle previously opened with sybconnect. Sybcancel may be used before sybnext exhausts all results. Sybcancel raises a Tcl error if the handle specified is not open.

sybretval *handle*

Return a Tcl list of the return values from a stored procedure. *Handle* must be a valid handle previously opened with sybconnect. If a stored procedure also returns rows, sybnext must have previously been called until NO_MORE_ROWS was encountered before sybretval can access the return values. The **sybmsg** array index *retstatus* contains the return code from the stored procedure. Sybretval raises a Tcl error if the handle specified is not open. See notes regarding stored procedure output variables.

sybwritetext *handle object colnum [-variable varname | -file filename | filename] ?-nolog?*

Write the contents of a variable or file to a TEXT or IMAGE column. *Handle* must be a valid handle previously opened with sybconnect. *Object* is the table and column name in the format **table.column**. *Colnum* is the relative position of the column from the last **select**. *varname* is the name of a Tcl variable containing data, or *filename* is the name of the file that contains the text or image data to write into the column. Text and image writes are logged by default, **-nolog** may be specified to disable logging (the database must have previously been set with a no log option.) If neither **-variable** or **-file** is specified, the argument is interpreted as a filename.

Sybwritetext can only be used in a specific sequence with other sybsql commands. Refer to the Sybase DB-Library documentation for dbwritetext() and the DB-Library Reference Supplement discussion on text/image handling.

For example (assume $hand is an open handle, using the "pubs" database):

```
sybsql $hand "insert into au_pix (au_id) values ('111-22-3333')"
sybsql $hand "update au_pix set pic = null where au_id = '111-22-3333'"
sybsql $hand "select pic from au_pix where au_id = '111-22-3333'"
sybwritetext $hand au_pix.pic 1 image.file -nolog
```

An update to an existing text or image column can be made using the last two commands from the above example. Sybwritetext returns a integer number upon successful completion of the number of bytes written to the text/image column.

Sybwritetext raises a Tcl error for a variety of reasons: filename could not be opened or a failure in internal DB-Library routines. Common failures are specifing **-nolog** when the database does not support nolog; unable to access a valid text pointer due to invalid **object** or **colnum**; sybwritetext used out of sequence. Consult **sybmsg(msgtext)** or **sybmsg(dberrstr)** for information after a failure.

sybreadtext *handle [-variable varname | -file filename | filename]*

Read the contents of a TEXT or IMAGE column and write results into a variable or a file. *Handle* must be a valid handle previously opened with sybconnect. *varname* is the name of a Tcl variable, or *Filename* is the name of a file in which to write the text or image data. Sybreadtext can only be used after the successful **select** of a single text or image column. If neither **-variable** or **-file** is specified, the argument is interpreted as a filename. For example (assume $hand is an open handle, using the "pubs" database):

sybsql $hand "select copy from blurbs where au_id = ’486-29-1786’"
sybreadtext $hand blurb.txt

Sybreadtext returns a decimal number upon successful completion of the number of bytes read from the text/image column. Sybreadtext returns "0" if the last select returned more than one column or no row was returned.

The **sybmsg** array index *maxtext* limits the amount of text or image data that can be written to a file by sybreadtext. The default is 32768 bytes.

Sybreadtext raises a Tcl error for a variety of reasons: filename could not be opened, sybreadtext used out of sequence, etc.

sybclose *handle*

Closes the server connection associated with *handle*. *Handle* must be a valid handle previously opened with sybconnect. Sybclose returns a null string. Sybclose raises a Tcl error if the handle specified is not open.

SERVER MESSAGE AND ERROR INFORMATION

Sybtcl creates and maintains a Tcl global array to provide feedback of Sybase server messages, named **sybmsg**. Sybmsg is also used to communicate with the sybtcl interface routines to specify null return values and text/image limits. In all cases except for *nullvalue*, *fixedchar*, *floatprec*, *dateformat*, *bgevents*, *bgpollinterval* and *maxtext*, each element is reset to null upon invocation of any sybtcl command, and any element affected by the command is set. The sybmsg array is shared among all open sybtcl handles. Sybmsg should be defined with the global statement in any Tcl procedure needing access to sybmsg.

Sybmsg elements:

version

is set to the version of Sybtcl.

nullvalue

can be set by the programmer to indicate the string value returned for any null result. Setting sybmsg(nullvalue) to "default" will return "0" for numeric null data types (integer, float, and money) and a null string for all other data types. Nullvalue is initially set to "default".

fixedchar

can be set by the programmer to indicate that character datatypes returned by sybnext should not have trailing spaces trimmed from the value. Setting sybmsg(fixedchar) to "1", "true", or "yes" will ensure that all trailing spaces are returned. The default value ("") will cause trailing spaces to be trimmed.

binaryashex

can be set by the programmer to indicate that binary data types (binary, image) should be converted to hexstrings. Setting sybmsg(binaryashex) to "1", "true", or "yes" will convert binary

types to hex strings. The default value ("") will cause binary data to be stored bit-for-bit.

dateformat

can be set by the programmer to indicate formatting for date data types. The dateformat string can contain substitution values or literals. Substitions are made from the list below; other literals are copied verbatim. The default value is null, which will format dates a default format.

YYYY
four digit year, 1900-

YY
two digit year, 00-99

MM
two digit month, 1-12

MONTH
name of month, January-December

MON
month abbreviation, Jan-Dec

DD
two digit day, 1-31

hh
two digit hour, 0-23

mm
two digit minute, 0-59

ss
two digit second, 0-59

ms
three digit millisecond, 0-999

dy
three digit day of year, 0-365

dw
one digit day of week, 1-7 (Mon-Sun)

bgevents

can be set by the programmer to allow or disallow the processing of Tcl events while Sybtcl is waiting for server response. Events are processed during Sybtcl commands that may wait on server responses: sybsql (without -async option), sybnext (with commands option), sybwritetest, and sybreadtext. The default value is "idletasks". Possible values are:

idletasks
Process only events that have been deferred, such as display updates. Similar to the Tcl **update idletasks** command.

all
Process all events. Similar to the Tcl **update** command.

none
Do not process events during Sybtcl commands.

bgpollinterval

can be set by the programmer to specify the polling interval in milliseconds while processing background events. The value specified must be an integer between 1 and 1000. The default

bgpollinterval value is 200. Smaller values causes the polling loop to execute more frequently, which may cause higer CPU usage.

maxtext

can be set by the programmer to limit the amount of text or image data returned by sybnext and sybreadtext. The default is 32768 bytes. The maximum is 2147483647 bytes. Any value less than or equal to zero is ignored. Any change to maxtext becomes effective on the next call to sybsql. See notes on maxtext usage with sybnext.

handle

indicates the handle of the last sybtcl command. Handle is set on every sybtcl command (execpt where an invalid handle is used.)

isnull

is a list of binary elements corresponding to each column element returned by sybnext. Each element is set to "1" if the value is null, "0" if the value is not null.

nextrow

indicates the results of the last SQL command and subsequent next row processing. Nextrow is set by sybsql and sybnext. Possible values are:

REG_ROW

at least one row is available after execution of sybsql, or the results of sybnext returned a row as a Tcl list.

n

an integer number, which indicates that last row retrieved by sybnext returned a compute row. The value is the *computeid*, which is the relative compute statement in the last SQL command executed with sybsql.

NO_MORE_ROWS

indicates that sybsql executed successfully but no rows are available, or the results of sybnext did not return a row. Sybnext will return a null string. Return values from a stored procedure, if any, are available at this time. If more results are expected, a subsequent execution of sybnext will return the first row, if any, from the next set of results.

PENDING

indicates the last execution of sybsql was made with the "-async" flag. Sybpoll may be used to check the status of results. Sybnext will block until results are available. When sybsql is executed with -async, any errors will not be available until the first execution of sybnext.

FAIL

indicates that a server error has occurred. Appropriate error codes and messages are set in the sybmsg indices *dberr* and *dberrstr*. Sybnext will return a null string. If more results are expected, a subsequent execution of sybnext will return the first row, if any, from the next set of results.

NO_MORE_RESULTS

indicates that the final set of results from the last execution of sybsql have been processed by sybnext.

retstatus

indicates the return code after a stored procedure has executed. Retstatus is set by sybsql or sybnext, whenever a the results of a stored procedure are available.

collengths
is a Tcl list of the lengths of the columns returned by sybcols. Numeric columns (Sybase datatypes int, float, etc.) are given by the internal data lengths (e.g., int has a length of 4), character columns lengths are the maximum of any row returned. Collengths is only set by sybcols.

coltypes
is a Tcl list of the types of the columns returned by sybcols. Coltypes is only set by sybcols. Possible types returned are: char, text, binary, image, tinyint, smallint, int, float, real, numeric, decimal, bit, money, smallmoney, datetime, smalldatetime. Varchar and varbinary data types are reported as char and binary.

msgno
indicates the message number from a Sybase Server message. Can be set by any sybtcl command. Refer to Sybase documentation for interpretation. Since each sybtcl command may invoke several DB-Lib routines, there is a possiblity that several messages may be received from the server. Sybtcl will concatenate all server message numbers received during one sybtcl command, separating individual numbers by newlines.

msgtext
the message text associated with msgno. Since each sybtcl command may invoke several DB-Lib routines, there is a possiblity that several messages may be received from the server. Sybtcl will concatenate all server messages received during one sybtcl command, separating individual messages by newlines. Output from Transact-SQL PRINT statments are collected in msgtext.

severity
indicates the severity level from a Sybase Server message or DB-Library routine. Can be set by any sybtcl command. Refer to Sybase documentation for interpretation.

svrname
indicates the name of the Sybase Server that generated a message. Can be set by any sybtcl command.

procname
indicates the name of the stored procedure that generated a message. Only set when a stored procedure was executed. Set by sybsql.

line
indicates the line number of the SQL command or stored procedure that generated a message. Set by sybsql.

dberr
indicates the error number generated by a DB-Library routine. Can be set by any sybtcl command. Refer to Sybase documentation for interpretation.

dberrstr
the error text associated with dberr.

oserr
indicates an operating system specific error number associated with a DB-Library error. Can be set by any sybtcl command.

oserrstr
the error text associated with oserr.

dblibinfo
is set to options that were in effect with Sybtcl was compiled. Possible values are: "system10" if linked with Sybase System 10 DB-Libs or higher, "ctcompt" if compiled with the CT-Lib compatiblity library.

NOTES

Tcl errors can also be raised by any sybtcl command if a command's internal calls to DB-Library routines fail. Sybtcl will return the name of the DB-Lib routine that caused an error.

When executing a stored procedure with sybsql, be sure to include in the SQL commands a "DECLARE" statement for local variables, and specify the local variables as "OUTPUT" on the "EXEC" statement. Otherwise, sybretval will not be able to access the return values. If the return variable names are to be accessed by sybcols, use the assignment form of "EXEC", e.g.:

```
declare @local-var datatype
exec stored-proc @proc-var-name = @local-var output
```

If a stored procedure is executed with sybsql, and the procedure uses Transact-SQL "PRINT" statements, check $sybmsg(msgtext) before executing any other Sybtcl commands. Otherwise, the PRINT output will be lost on the next command. Multiple PRINT statements are separated by newlines in $sybmsg(msgtext).

Sybtcl error and message handlers simply populate values in the sybmsg array. The error handler always returns INT_CANCEL.

To use the **-nolog** feature of sybwritetext, the following option must have been set on the database:

sp_dboption 'yourdbname', 'select into/bulkcopy', 'true'

The limit of the number of simultaneous connections is artificial, based on a fixed table in sybtcl. Change the source *#define SYBTCLPROCS* if more are needed.

The maximum amount of TEXT or IMAGE data returned by sybnext is ultimately dependent on sybtcl's ability to malloc() *maxtext* bytes of memory for each TEXT or IMAGE column retrieved. Setting sybmsg(maxtext) to too high a value may cause core dumps or memory shortages. Sybreadtext does not malloc() an area to hold the entire value; instead it retrieves TEXT and IMAGE in chunks and writes to a file. While maxtext limits the amount of data retrieved by sybreadtext, it shouldn't cause memory shortages as sybnext might.

When using sybnext with the optional tcl-commands argument, a substantial performance improvement can be realized by not using the substitution values (@1, @2, etc.) Instead, specify the substitution character as a null string {}, and use the tclvar-columnum argument pairs.

ENVIRONMENT VARIABLES

DSQUERY
: The default Sybase server.

SYBASE
: The home directory for Sybase files.

FILES

$SYBASE/interfaces - definitions for Sybase Servers.

AUTHOR

Tom Poindexter, Denver Colorado <tpoindex@nyx.net>, Currently maintained by D. J. Hagberg <dhagberg@millibits.com>. Version 3.0 released December, 2000. The ?commands? option to sybnext was borrowed from my work with Oratcl, a Tcl interface to the Oracle database product.
http://sybtcl.sourceforge.net/
http://www.nyx.net/~tpoindex