

Database
Management
System

LINTER®

Version 5.9

Perl Extensions

Relational Expert Systems



Table of Contents

Introduction	3
Installation	4
Exported Functions	5
Open Connection.....	5
Close Connection	5
Open Cursor	5
Close Cursor.....	6
Execute Direct	6
Prepare Statement	7
Bind Variables to Query	7
Bind Parameter Array.....	7
Execute a Prepared Statement.....	8
Fetch Query Result.....	8
Bind Variables to Result.....	9
Unbind Variables	9
Get Data from One Column	10
Test for Null Data.....	10
Append Blob Data	10
Delete BLOB Column Value	11
Get BLOB Size	11
Get BLOB Segment.....	12
Commit.....	12
Rollback	12
Get Data from Single Column	13
Get Error Information.....	13
Get Error Message	14
Exported Constants	15
Available data types	15
Directions for Fetch	15
Transaction Modes.....	15
Supported Code Pages.....	16
Cursor and Connection Options.....	16
Extended Internal Error Codes	16

Introduction

This document describes the Linter module that provides an interface from Perl to Linter. It is implemented for Microsoft Windows and various UNX platforms.

Installation

1. Create subdirectory LinPerl in \$PERL5LIB/auto. For Win32, use: %PERL5LIB%\auto.
2. Copy file LinPerl.so (for Microsoft, LinPerl.dll) into the LinPerl directory.
3. Copy the file LinPerl.pm into \$PERL5LIB (%PERL5LIB% for Microsoft).
4. Enjoy using Perl with Linter.

Exported Functions

All functions return a completion code, an integer value, which contains a code from either the module or Linter.

The result(s) of any function are always returned via parameter(s). Module internal error codes are always negative. A zero code means successful completion.

Open Connection

OpenConnect opens a new connection to a Linter server.

Syntax

```
OpenConnect(username, password, server, mode, ConnectID);
```

<u>Parameter</u>	<u>Description</u>
user name	User's name. 18 char maximum. Case sensitive.
password	User's password. 18 char max. Case sensitive.
server	Node name of Linter server. Default is local host. 8 character maximum.
mode	Zero or an OR of codepage + transaction modeconstants.
ConnectID	ID of the new connection.

Example

```
$err = OpenConnect("KENT", "ALEX", "LINTER", CP_866 |
MODE_EXCLUSIVE, $con);
$err && [code for handling error]
```

Close Connection

The CloseConnect function closes an open connection.

Syntax

```
CloseConnect(ConnectID);
```

<u>Parameter</u>	<u>Description</u>
ConnectID	Connection identifier.

Example

```
$err = CloseConnect($con);
$err && [code for handling error]
```

Open Cursor

OpenCursor opens a new cursor.

Syntax

```
OpenCursor(connectID, CursNname, CursorID);
```

<u>Parameter</u>	<u>Description</u>
Connect	ID of parent connection.
CursName	Cursor name. 18 char max. Not currently implemented.
CursorID	Identifier of the new cursor.



The cursor's transaction mode and code page are inherited from the parent connection.

Example

```
$err = OpenCursor($con, "MY_CURSOR", $cur);
$err && [code for handling error]
```

See Also

Function OpenConnect.

Close Cursor

The CloseCursor function closes an open cursor.

Syntax

```
CloseCursor(CursorID);
```

<u>Parameter</u>	<u>Description</u>
CursorID	Cursor identifier.

Example

```
$err = CloseCursor($cur);
$err && [code for handling error]
```

Execute Direct

The ExecDirect function executes a query without using a prepared statement.

Syntax

```
ExecDirect(CursorID, query);
```

<u>Parameter</u>	<u>Description</u>
CursorID	Cursor identifier.
query	Text of query. Must be terminated with semi-colon (;).

Example

```
$err = ExecDirect($cur, "select * from auto;");
$err && [code for handling error]
```

See Also

Functions Fetch, Prepare, BindParameter, Execute.

Prepare Statement

The Prepare function prepares query for execution.

Syntax

```
Prepare(CursorID, query);
```

<u>Parameter</u>	<u>Description</u>
CursorID	Cursor identifier.
query	Text of query. Must be terminated with semi-colon (;).

Example

```
$err = Prepare($cur, "select * from auto where personid = ?;");  
$err && [code for handling error]
```

See Also

Functions BindParameter, Execute.

Bind Variables to Query

The BindParameter function binds variables to query parameters.

Syntax

```
BindParameter(CursorID, number, param);
```

<u>Parameter</u>	<u>Description</u>
CursorID	Cursor identifier.
number	Parameter number. Numbering starts with 1.
param	Perl variable.



If a parameter has been already bound, it will be unbound automatically.

Example

```
$err = BindParameter($cur, 1, $my_param);  
$err && [code for handling error]
```

See Also

Functions Prepare, Execute, BindParamArray.

Bind Parameter Array

The BindParamArray function binds a Perl array to query parameters.

Syntax

```
BindParamArray(CursorID, begin, end, param);
```

<u>Parameter</u>	<u>Description</u>
CursorID	Cursor identifier.
begin	First parameter number. Numbering begins with 1.
end	Last parameter number.
param	Perl array.



If parameters have already been bound, they will be unbound automatically.

Example

```
$err = BindParamArray($cur, 1, 3, @my_param);
$err && [code for handling error]
```

See Also

Functions Execute, BindParameter.

Execute a Prepared Statement

The Execute function executes a prepared query.

Syntax

```
Execute(CursorID);
```

<u>Parameter</u>	<u>Description</u>
CursorID	Cursor identifier.

Example

```
$err = Execute($cur);
$err && [code for handling error]
```

See Also

Functions Prepare, BindParameter, Fetch.

Fetch Query Result

The Fetch function fetches query result row by row.

Syntax

```
Fetch(CursorID, direction, number);
```

<u>Parameter</u>	<u>Description</u>
CursorID	Cursor identifier.
direction	Direction for fetch, see function Fetch.
number	Absolute row number. Use only with FETCH_ABSNUM. Row numbering begins with 1.

Example

```
$err = Fetch($cur, FETCH_NEXT, 0);
$err = Fetch($cur, FETCH_ABSNUM, 100);
$err && [code for handling error]
```

See Also

Functions ExecDirect, Execute.

Bind Variables to Result

The BindColumn function binds variables to query result columns.

Syntax

```
BindColumn(CursorID, number, param);
```

<u>Parameter</u>	<u>Description</u>
CursorID	Cursor identifier.
number	Column number. Starts from 1.
param	Perl variable.

Example

```
$err = BindColumn($cur, 1, $make);
$err && [code for handling error]
```

See Also

Functions GetDataColumn, Fetch, UnbindColumn.

Unbind Variables

The UnbindColumn function unbinds variables from the query result columns.

Syntax

```
UnbindColumn(CursorID, number);
```

<u>Parameter</u>	<u>Description</u>
CursorID	Cursor identifier.
number	Column number. Starts from 1.

Example

```
$err = UnbindColumn($cur, 1);
$err && [code for handling error]
```

See Also

Function BindColumn.

Get Data from One Column

The GetDataColumn function gets data for one column.

Syntax

```
GetDataColumn(CursorID, number, var);
```

<u>Parameter</u>	<u>Description</u>
CursorID	Cursor identifier.
number	Column number. Starts from 1.
var	Perl variable for output. If result is NULL, value is "undefined". For BLOB columns, use BLOBGetData.

Example

```
$err = GetDataColumn($cur, 1, $make);  
$err && [code for handling error]
```

See Also

Functions ExecDirect, Execute, Fetch, BLOBGetData.

Test for Null Data

The TestNull function tests data for NULL value.

Syntax

```
TestNull(CursorID, number, var);
```

<u>Parameter</u>	<u>Description</u>
CursorID	Cursor identifier.
number	Column number. Starts from 1.
var	Perl variable for output.

Example

```
$err = TestNull($cur, 1, $is null);  
$err && [code for handling error] if ($is null != 0) ...
```

See Also

Functions GetDataColumn, BLOBGetData.

Append Blob Data

BLOBAppend appends a segment of data to the BLOB value of the current answer row.

Syntax

```
BLOBAppend (CursorID, var);
```

<u>Parameter</u>	<u>Description</u>
CursorID	Cursor identifier.
var	Perl variable which contains BLOB data.

Example

```
$err = BLOBAppend($cur, $blob);
$err && [code for handling error]
```

See Also

Functions BLOBClear, BLOBGetSize, BLOBGetData.

Delete BLOB Column Value

The BLOBClear function clears BLOB value of the current answer row.

Syntax

```
BLOBClear(CursorID);
```

<u>Parameter</u>	<u>Description</u>
CursorID	Cursor identifier.

Example

```
$err = BLOBClear($cur);
$err && [code for handling error]
```

See Also

Functions BLOBAppend, BLOBGetSize, BLOBGetData.

Get BLOB Size

The BLOBGetSize function gets size of the current row's BLOB data.

Syntax

```
BLOBGetSize(CursorID, var);
```

<u>Parameter</u>	<u>Description</u>
CursorID	Cursor identifier.
var	Perl variable for output.

Example

```
$err = BLOBGetSize($cur, $blob_size);
$err && [code for handling error]
```

See Also

Functions BLOBAppend, BLOBClear, BLOBGetData.

Get BLOB Segment

The BLOBGetData function gets a segment of BLOB data from the current answer row.

Syntax

```
BLOBGetData(CursorID, pos, len, var);
```

<u>Parameter</u>	<u>Description</u>
CursorID	Cursor identifier.
pos	Starting position, begins with 1.
len	Perl variable which sets size of BLOB segment. On completion, this variable contains the number of bytes actually read.
var	Perl variable for output.

Example

```
$err = BLOBGetData($cur, 1, 1000, $blob);  
$err && [code for handling error]
```

See Also

Functions BLOBAppend, BLOBClear, BLOBGetSize, GetDataColumn.

Commit

The Commit function writes the results of the current transaction permanently to the database.

Syntax

```
Commit(CursorID);
```

<u>Parameter</u>	<u>Description</u>
CursorID	Cursor identifier.

Example

```
$err = Commit($cur);  
$err && [code for handling error]
```

See Also

Functions ExecDirect, Execute, Rollback.

Rollback

The Rollback function rolls back results of the current transaction.

Syntax

```
Rollback(CursorID);
```

<u>Parameter</u>	<u>Description</u>
CursorID	Cursor identifier.

Example

```
$err = Rollback($cur);
$err && [code for handling error]
```

See Also

Functions ExecDirect, Execute, Commit.

Get Data from Single Column

The GetColInfo function returns information about the specified column.

Syntax

```
GetColInfo(CursorID, number, type, len, name);
```

<u>Parameter</u>	<u>Description</u>
CursorID	Cursor identifier.
number	Column number, starts from 1.
type	Perl variable for column type.
len	Perl variable for column length.
name	Perl variable for column name.



See section “Available Data Types” for more information on type and length.

Example

```
$err = GetColInfo($cur, 1, $type, $len, $name);
$err && [code for handling error]
```

See Also

Functions ExecDirect, Execute.

Get Error Information

The GetError functions return the Linter SQL Server error code and the OS error code.

Syntax

```
GetError(CursorID, linerr, syserr);
```

<u>Parameter</u>	<u>Description</u>
CursorID	Cursor identifier.
linerr	Perl variable for Linter error code.
syserr	Perl variable for OS error code.

Example

```
$err = GetError($cur, $linerr, $syserr);
$err && [code for handling error]
```

See Also

Function `GetErrorMsg`.

Get Error Message

The `GetErrorMsg` function returns a text message for a Linter error.

Syntax

```
GetErrorMsg(CursorID, linerr, errmsg);
```

<u>Parameter</u>	<u>Description</u>
CursorID	Cursor identifier.
linerr	Linter error code.
errmsg	Perl variable for message text.



The table `ERRORS` must exist in database, otherwise `GetErrorMsg` returns an empty string.

Example

```
$err = GetErrorMsg($cur, $linerr, $errmsg);  
$err && [code for handling error]
```

See Also

Function `GetError`.

Exported Constants

Available data types

<u>Macro Name</u>	<u>Type</u>	<u>Max Length</u>
LINCHAR	char	4000*
LIN_INTEGER	integer	4
LINREAL	real	4
LINDATE	timestamp	16
LINNUMERIC	numeric	16
LINBYTE	byte	4000*
LINBLOB	BLOB	2GB
LINSMALLINT	smallint	2
LINDOUBLE	Double	8

*240 for Linter ver.4.xx

Directions for Fetch

<u>Macro Name</u>	<u>Description</u>
FETCH_FIRST	Get first row.
FETCH_LAST	Get last row.
FETCH_NEXT	Get next row.
FETCH_PREV	Get previous row.
FETCH_ABSNUM	Get row by exact row number.

Transaction Modes

<u>Macro Name</u>	<u>Description</u>
TMAUTOCOMMIT	Set auto commit mode for transaction.
TMOPTIMISTIC	Set optimistic mode for transaction.
TMEXCLUSIVE	Set exclusive mode for transaction.

Supported Code Pages

<u>Macro Name</u>	<u>Description</u>
CP_866	MS-DOS 866.
CP_1251	Windows 1251.
CP_KOI8	Unix Cyrillic.

Cursor and Connection Options

<u>Macro Name</u>	<u>Set</u>	<u>Get</u>	<u>Description</u>
CO_CURNAME	+	+	Cursor name.
CO_ASYNCMODE	+	-	Asynchronous connection mode. Not implemented in current version.
CO_SYNCMODE	+	-	Synchronous connection mode.
CO_DTFORMAT	+	+	Format for DATE data type.
CO_COLCOUNT	-	+	No. of columns in answer.
CO_ROWCOUNT	-	+	No. of rows in answer.
CO_ERRSTR	-	+	No. of query line with error.
CO_ERRPOS	-	+	Number of error position in line.
CO_TRANSMODE	-	+	Transaction mode.
CO_CURROW	-	+	Current row in cursor.
CO_CURROWID	-	+	ROWID of current row.
CO_CONNECTID	-	+	Connection ID.
CO_NODENAME	-	+	Name of Linter server.

Extended Internal Error Codes

<u>Macro Name</u>	<u>Description</u>
LPERR_NOMEMORY	Not enough memory.
LPERR_INVCURSOR	Invalid cursor or connection.
LPERR_INVCOLNUM	Invalid column number.
LPERR_INVPARAMNUM	Invalid parameter number.
LPERR_INVSTATE	Invalid cursor/connection state.
LPERR_INVDIRECT	Invalid fetch direction.
LPERR_UNKNOWNOPT	Unknown cursor option.
LPERR_BADDTFORMAT	Bad format for data type DATE.

Macro Name**Description**

LPERR_NOBLOBCOL

BLOB column is absent in answer.

LPERR_NOPARENT

Cursor doesn't have parent.

LPERR_NOTIMPLEMENTED

Feature not implemented.