

Uppgjord (även faktaansvarig om annan) - <i>Prepared (also subject responsible if other)</i>	Nr - <i>No.</i>		
ETX/DN/SP Joakim Hirsch			
Dokansv/Godkänd - <i>Doc respons/Approved</i>	Kontr - <i>Checked</i>	Datum - <i>Date</i>	Rev
ETX/DN/SP (Peter Högfeltdt)		1999-02-16	PA1
			File
			odbc_ref_man.fm5

© Ericsson Telecom AB, 1998

# ODBC Reference Manual

Uppgjord (även faktaansvarig om annan) - Prepared (also subject responsible if other)		Nr - No.		
Dokansv/Godkänd - Doc respons/Approved	Kontr - Checked	Datum - Date	Rev	File
		1999-02-16	PA1	

## Contents

<b>1</b>	<b>Introduction</b>	<b>3</b>
<b>1.1</b>	<b>Start and Stop</b>	<b>3</b>
1.1.1	start_link/[2, 3]	3
1.1.2	stop/[1, 2]	5
<b>1.2</b>	<b>Utility API</b>	<b>6</b>
1.2.1	init_env/[1, 2]	6
1.2.2	connect/[3, 4, 5, 6]	7
1.2.3	execute_stmt/[3, 4]	8
1.2.4	disconnect/[2, 3]	10
1.2.5	terminate_env/[2, 3]	11
<b>1.3</b>	<b>Basic API</b>	<b>12</b>
1.3.1	sql_alloc_handle/[3, 4]	12
1.3.2	sql_bind_col/[4, 5]	13
1.3.3	sql_bind_parameter/[8, 9]	14
1.3.4	sql_close_cursor/[2, 3]	16
1.3.5	sql_connect/[5, 6]	16
1.3.6	sql_describe_col/[4, 5]	17
1.3.7	sql_disconnect/[2, 3]	19
1.3.8	sql_driver_connect/[5, 6]	20
1.3.9	sql_end_tran/[4, 5]	21
1.3.10	sql_exec_direct/[3, 4]	22
1.3.11	sql_fetch/[2, 3]	23
1.3.12	sql_free_handle/[3, 4]	24
1.3.13	sql_get_connect_attr/[4, 5]	25
1.3.14	sql_get_diag_rec/[5, 6]	27
1.3.15	sql_num_result_cols/[2, 3]	28
1.3.16	sql_row_count/[2, 3]	29
1.3.17	sql_set_connect_attr/[5, 6]	30
1.3.18	sql_set_env_attr/[5, 6]	31
1.3.19	alloc_buffer/[3, 4]	32
1.3.20	dealloc_buffer/[2, 3]	33
1.3.21	read_buffer/[2, 3]	33
1.3.22	write_buffer/[3, 4]	34
<b>1.4</b>	<b>Error Messages and Exceptions</b>	<b>35</b>
<b>1.5</b>	<b>References</b>	<b>35</b>

Uppgjord (även faktaansvarig om annan) - <i>Prepared (also subject responsible if other)</i>	Nr - No.		
Dokansv/Godkänd - <i>Doc respons/Approved</i>	Kontr - <i>Checked</i>	Datum - <i>Date</i>	Rev File
		1999-02-16	PA1

# 1 Introduction

The ODBC API is divided into three parts:

Start and Stop	Starts and stops the server process.
Utility API	Consists of functions that are easier to use than the Basic API. These functions are on a higher level, do more of the job, but allow less control to the application programmer.
Basic API	Gives access to the IDL Interface functions, which are mapped on ODBC functions.

All functions described are synchronous.

The interface supports all ODBC defined SQL data types except binaries. They are all mapped on Erlang strings.

The type `string()` is a `list()` of integers representing ASCII codes. The type `boolean()` is either the macro `?SQL_TRUE` or the macro `?SQL_FALSE`.

The default Timeout for all functions is 5000 ms, unless otherwise stated.

## 1.1 Start and Stop

### 1.1.1 `start_link/[2, 3]`

#### Description:

Starts a new ODBC server process, registers it with the supervisor, and links it to the calling process. Opens a unique IDL connection to a new C node on the local host, using the same cookie as is used by the node of the calling process. Links to the process on the C node.

#### Arguments and Return Value:

```
start_link(Args, Options) ->
start_link(ServerName, Args, Options) ->
Result
```

Args ->	[Arg]	Initialisation arguments.
Arg ->	{buffer_size, integer()}	The initial size of the buffer through which

Uppgjord (även faktaansvarig om annan) - <i>Prepared (also subject responsible if other)</i>		Nr - No.	
Dokansv/Godkänd - <i>Doc respons/Approved</i>	Kontr - <i>Checked</i>	Datum - <i>Date</i> 1999-02-16	Rev PA1
		File	

communication with the C node is done. Doesn't limit the amount of data that can pass in either direction of a function call, since the buffer will grow dynamically. The default is 32 kb. The minimum is 4 kb.

{ max\_len\_data, integer() }

The maximum length of table data, including null-termination, returned from ODBC. This value must be chosen with the buffer size in mind. The default is 8 kb. Used only by the Utility API.

NOTE: The data source or driver may have a lower limit for the max size of returned data. This limit is the value of the optional statement attribute SQL\_ATTR\_MAX\_LENGTH (see [1]).

{ max\_len\_err\_msg, integer() }

The maximum length of the message part of ODBC error messages, including null-termination,. This value must be chosen with the buffer size in mind. The default is 1 kb. Used only by the Utility API.

{ max\_len\_str, integer() }

The maximum length of other strings, including null-termination, passed from ODBC to the ODBC server (e.g. column names). Does not limit the size of returned table values. This value must be chosen with the buffer size in mind. The default is 1 kb. Used only by the Utility API.

Options -> [Opt]

List of options.

Uppgjord (även faktaansvarig om annan) - <i>Prepared (also subject responsible if other)</i>	Nr - No.		
Dokansv/Godkänd - <i>Doc respons/Approved</i>	Kontr - <i>Checked</i>	Datum - <i>Date</i> 1999-02-16	Rev PA1 File

Opt ->	{timeout, integer()}    {debug, [Dbg]}	The time in ms allowed for initialisation, see gen_server. Debug options, see gen_server.
Dbg ->	trace   log   statistics   {log_to_file, FileName}   {install, {Func, FuncState}}	Debug options, see gen_server.
ServerName ->	{local, atom()}   {global, atom()}	When supplied, causes the server to be registered, locally or globally. If the server is started without a name it can only be called using the returned Pid.
Result ->	{ok, pid()}   {error, Reason}	The pid of the erl. server. Error tuple.
Reason ->	{already_started, pid()}   timeout   {no_c_node, Info}	Server already started. Timeout expired. Can't start C node. The program may not have been found or may not have been executable e.g.
Info ->	string()	More information.

NOTE: There is no default timeout value. Not using the timeout option is equivalent to having an infinite timeout value.

NOTE: Timeout is reported as an error here, not an exception.

NOTE: The debug options are described in the sys module documentation.

## 1.1.2 stop/[1, 2]

### Description:

Stops the ODBC server process as soon as all already submitted requests have been processed. The C node is also stopped.

### Arguments and Return Value:

stop(Server) ->  
stop(Server, Timeout) ->  
ok

Uppgjord (även faktaansvarig om annan) - <i>Prepared (also subject responsible/lf other)</i>		Nr - No.		
Dokansv/Godkänd - <i>Doc respons/Approved</i>	Kontr - <i>Checked</i>	Datum - <i>Date</i>	Rev	File
		1999-02-16	PA1	

Server ->	pid()   Name   { global, Name }   { Name, Node }	The pid of the server process, a registered name, a globally registered name, or a registered name on a remote node.
Timeout ->	integer()   infinity	Max time (ms) for serving the request.

## 1.2 Utility API

The Utility API uses three maximum string length parameters: the maximum data string length (`max_len_data`), the maximum error message length (`max_len_err_msg`), and the maximum length of 'other strings' (e.g. column names) passed from ODBC (`max_len_str`). These can be set in the call to `start/[2, 3]` and `start_link/[2, 3]`, but there are default values.

Errors reported by the ODBC API are returned in lists. The relative order of these errors is the same as specified in [1]. Warnings are always ignored and execution proceeds. Should an error occur, execution stops.

### 1.2.1 `init_env/[1, 2]`

#### Description:

Initialises the ODBC environment on the C node. This can also be done through use of functions of the Basic API.

#### Arguments and Return Value:

```
init_env(Server) ->
init_env(Server, Timeout) ->
  {ok, RefEnvHandle} |
  {error, {Fcn, [Reason]}}
```

Server ->	pid()   Name   { global, Name }   { Name, Node }	The pid of the server process, a registered name, a globally registered name, or a registered name on a remote node.
Timeout ->	integer()   infinity	Max time (ms) for serving the

Uppgjord (även faktaansvarig om annan) - <i>Prepared (also subject responsible if other)</i>	Nr - No.		
Dokansv/Godkänd - <i>Doc respons/Approved</i>	Kontr - <i>Checked</i>	Datum - <i>Date</i> 1999-02-16	Rev PA1 File

RefEnvHandle ->	term()	request. Reference to the initialised environment.
Fcn ->	atom()	The originating function.
Reason ->	{SqlState, MoreInfo}	An ODBC error tuple.
SqlState ->	string()	The SQL state, see [1].
MoreInfo ->	{NativeCode, Msg, LenMsg}	More error info.
NativeCode ->	string()	Data source specific error code.
Msg ->	string()	Error message.
LenMsg ->	integer()	Length of Msg before truncation.

## 1.2.2 connect/[3, 4, 5, 6]

### Description:

Opens a connection to a data source. There can be only one open data source connection per server. connect/[3, 4] is used when the information that can be supplied through connect/[5, 6] does not suffice.

NOTE: The syntax to be used for ConnectStr is described under SQLDriverConnect in [1]. The ConnectStr must be complete.

### Arguments and Return Value:

```
connect(Server, RefEnvHandle, ConnectStr) ->
connect(Server, RefEnvHandle, ConnectStr, Timeout) ->
connect(Server, RefEnvHandle, DSN, UID, PWD) ->
connect(Server, RefEnvHandle, DSN, UID, PWD, Timeout) ->
{ok, RefConnHandle} |
{error, {Fcn, [Reason]}}
```

Server ->	pid()   Name   {global, Name}   {Name, Node}	The pid of the server process, a registered name, a globally registered name, or a registered name on a remote node.
RefEnvHandle ->	term()	Reference to the

Uppgjord (även faktaansvarig om annan) - <i>Prepared (also subject responsible if other)</i>		Nr - No.		
Dokansv/Godkänd - <i>Doc respons/Approved</i>	Kontr - <i>Checked</i>	Datum - <i>Date</i>	Rev	File
		1999-02-16	PA1	

ConnectStr ->	string()	environment. Returned by init_env/1. Connection string. For syntax see SQLDriverConnect in [1].
DSN ->	string()	Name of the data source.
UID ->	string()	User ID.
PWD ->	string()	Password.
Timeout ->	integer()   infinity	Max time (ms) for serving the request.
RefConnHandle ->	term()	Reference to the opened connection.
Fcn ->	atom()	The originating function.
Reason ->	{SqlState, MoreInfo}	An ODBC error tuple.
SqlState ->	string()	The SQL state, see [1].
MoreInfo ->	{NativeCode, Msg, LenMsg}	More error info.
NativeCode ->	string()	Data source specific error code.
Msg ->	string()	Error message.
LenMsg ->	integer()	Length of Msg before truncation.

### 1.2.3 execute\_stmt/[3, 4]

#### Description:

Executes a single SQL statement. All changes to the data source are automatically committed if successful. Data that is returned for SELECT statements is in string form. Use the Basic API to retrieve binaries and large data (that needs to be divided into smaller chunks).

#### Arguments and Return Value:

```
execute_stmt(Server, RefConnHandle, Stmt) ->
execute_stmt(Server, RefConnHandle, Stmt, Timeout) ->
  {updated, NRows} |
  {selected, [ColName], [Row]}
  {error, {Fcn, [Reason]}}
```

Uppgjord (även faktaansvarig om annan) - <i>Prepared (also subject responsible if other)</i>	Nr - No.		
Dokansv/Godkänd - <i>Doc respons/Approved</i>	Kontr - <i>Checked</i>	Datum - <i>Date</i> 1999-02-16	Rev PA1 File

Server ->	pid()   Name   {global, Name}   {Name, Node}	The pid of the server process, a registered name, a globally registered name, or a registered name on a remote node.
RefConnHandle ->	term()	Reference to an open connection. Returned by connect/[3, 5].
Stmt ->	string()	SQL statement to execute.
Timeout ->	integer()   infinity	Max time (ms) for serving the request.
NRows ->	integer()	The number of updated rows for update, insert, or delete statements, or -1 if the number is not available. For other statement types the value is driver defined, see [1].
ColName ->	string()	The name of a column in the resulting table.
Row ->	[Value]	One row of the resulting table.
Value ->	string()   null	One value in a row.
Fcn ->	atom()	The originating function.
Reason ->	{SqlState, MoreInfo}	An ODBC error tuple.
SqlState ->	string()	The SQL state, see [1].
MoreInfo ->	{NativeCode, Msg, LenMsg}	More error info.
NativeCode ->	string()	Data source specific error code.
Msg ->	string()	Error message.

Uppgjord (även faktaansvarig om annan) - <i>Prepared (also subject responsible if other)</i>		Nr - No.	
Dokansv/Godkänd - <i>Doc respons/Approved</i>	Kontr - <i>Checked</i>	Datum - <i>Date</i>	Rev PA1
		1999-02-16	File

LenMsg -> integer()

Length of Msg  
before truncation.

NOTE: {updated, 0} is returned when a statement that does not select or update any rows is successfully executed.

NOTE: The ColNames are ordered the same way as the Values in the Rows (the first ColName is associated with the first Value of each Row etc.). The Rows have no defined order since they represent a set.

NOTE: Column names will be truncated if they are longer than the maximum string length (see option to start\_link/[2, 3]). Table values will be truncated if they are longer than the maximum data length, or longer than the value of the statement attribute SQL\_ATTR\_MAX\_LENGTH. If the amount of memory needed to retrieve a table value from a data source can not be determined, the default maximum data length (see start\_link/[2, 3]) is used.

#### 1.2.4 disconnect/[2, 3]

##### Description:

Closes the connection to a data source.

##### Arguments and Return Value:

disconnect(Server, RefConnHandle) ->  
disconnect(Server, RefConnHandle, Timeout) ->  
ok |  
{error, {Fcn, [Reason]}}

Server -> pid() |  
Name |  
{global, Name} |  
{Name, Node}

The pid of the server process, a registered name, a globally registered name, or a registered name on a remote node.

RefConnHandle -> term()

Reference to an open connection. Returned by connect/[3, 5].

Timeout -> integer() |  
infinity

Max time (ms) for serving the request.

Uppgjord (även faktaansvarig om annan) - <i>Prepared (also subject responsible if other)</i>	Nr - No.		
Dokansv/Godkänd - <i>Doc respons/Approved</i>	Kontr - <i>Checked</i>	Datum - <i>Date</i> 1999-02-16	Rev PA1 File

Fcn ->	atom()	The originating function.
Reason ->	{SqlState, MoreInfo}	An ODBC error tuple.
SqlState ->	string()	The SQL state, see [1].
MoreInfo ->	{NativeCode, Msg, LenMsg}	More error info.
NativeCode ->	string()	Data source specific error code.
Msg ->	string()	Error message.
LenMsg ->	integer()	Length of Msg before truncation.

### 1.2.5 terminate\_env/[2, 3]

#### Description:

Cleans up the ODBC environment on the C node. This can also be done through use of functions of the Basic API.

#### Arguments and Return Value:

terminate\_env(Server, RefEnvHandle) ->  
 terminate\_env(Server, RefEnvHandle, Timeout) ->  
 ok |  
 {error, {Fcn, [Reason]}}

Server ->	pid()   Name   {global, Name}   {Name, Node}	The pid of the server process, a registered name, a globally registered name, or a registered name on a remote node.
RefEnvHandle ->	term()	Reference to the environment. Returned by init_env/1.
Timeout ->	integer()   infinity	Max time (ms) for serving the request.
Fcn ->	atom()	The originating

Uppgjord (även faktaansvarig om annan) - Prepared (also subject responsible if other)		Nr - No.		
Dokansv/Godkänd - Doc respons/Approved	Kontr - Checked	Datum - Date	Rev	File
		1999-02-16	PA1	

Reason ->	{SqlState, MoreInfo}	function. An ODBC error tuple.
SqlState ->	string()	The SQL state, see [1].
MoreInfo ->	{NativeCode, Msg, LenMsg}	More error info.
NativeCode ->	string()	Data source specific error code.
Msg ->	string()	Error message.
LenMsg ->	integer()	Length of Msg before truncation.

## 1.3 Basic API

To use the Basic API it is necessary to gain a comprehensive understanding of ODBC by studying [1].

ODBC defines the concept of *deferred buffers*. A deferred buffer is one that exists longer than one function call, so it can be used in several calls. Deferred buffers come in pairs: one data buffer and one length/indicator buffer. The length/indicator buffer is used for communicating the length of data in the data buffer, or to indicate something about the data (e.g. that it is a null-value). The Basic API handles these buffers accordingly: they are allocated, deallocated, read, and written pair-wise.

### 1.3.1 sql\_alloc\_handle/[3, 4]

#### Description:

Allocates memory for an environment, connection, or statement handle. See SQLAllocHandle in [1].

#### Differences from the ODBC Function:

Allocation of descriptor handles is not supported. The parameters Server and Timeout have been added. The ODBC output parameter OutputHandlePtr has been changed into the returned value RefOutputHandle. Connection pooling is not supported.

#### Arguments and Return Value:

Uppgjord (även faktaansvarig om annan) - <i>Prepared (also subject responsible if other)</i>	Nr - No.		
Dokansv/Godkänd - <i>Doc respons/Approved</i>	Kontr - <i>Checked</i>	Datum - <i>Date</i>	Rev File
		1999-02-16	PA1

sql\_alloc\_handle(Server, HandleType, RefInputHandle) ->  
 sql\_alloc\_handle(Server, HandleType, RefInputHandle, Timeout) ->  
 {Result, RefOutputHandle}

Server ->	pid()   Name   {global, Name}   {Name, Node}	The pid of the server, a registered name, a globally registered name, or a registered name on a remote node.
HandleType ->	?SQL_HANDLE_ENV   ?SQL_HANDLE_DBC   ?SQL_HANDLE_STMT	Macros which determine which type of handle to allocate.
RefInputHandle ->	term()   ?SQL_NULL_HANDLE	The context in which the new handle is to be allocated. When allocating an environment handle, use ?SQL_NULL_HANDLE.
Timeout ->	integer()   infinity	Max time (ms) for serving the request.
Result ->	?SQL_SUCCESS   ?SQL_SUCCESS_WITH_INFO   ?SQL_INVALID_HANDLE   ?SQL_ERROR	Result macro.
RefOutputHandle ->	term()   ?SQL_NULL_HENV   ?SQL_NULL_HDBC   ?SQL_NULL_HSTMT	Reference to the allocated handle, or a value representing an error.

### 1.3.2 sql\_bind\_col/[4, 5]

#### Description:

Assigns storage and data type for a column in a result set (binds a buffer to a column). See SQLBindCol in [1]. Buffers/columns can also be unbound.

NOTE: The memory associated with RefBuf has to be allocated already.

#### Differences from the ODBC Function:

Neither binding of arrays nor the use of binding offsets is supported. It is not possible to unbind the data buffer without also unbinding the data length buffer. The parameters Server and Timeout have been added. The input parameters TargetType, TargetValuePtr, BufferLength, and StrLen\_or\_IndPtr of the ODBC function have been replaced with the RefBuf parameter (which represents the same data).

Uppgjord (även faktaansvarig om annan) - <i>Prepared (also subject responsible if other)</i>		Nr - No.		
Dokansv/Godkänd - <i>Doc respons/Approved</i>	Kontr - <i>Checked</i>	Datum - <i>Date</i>	Rev	File
		1999-02-16	PA1	

### Arguments and Return Value:

sql\_bind\_col(Server, RefStmtHandle, ColNum, RefBuf) ->  
 sql\_bind\_col(Server, RefStmtHandle, ColNum, RefBuf, Timeout) ->  
 Result

Server ->	pid()   Name   {global, Name}   {Name, Node}	The pid of the server, a registered name, a globally registered name, or a registered name on a remote node.
RefStmtHandle ->	term()	Reference to the statement handle.
ColNum ->	integer()	Column number from left to right starting at 1.
RefBuf ->	integer()   ?NULL_REF	Reference to the buffer where the column data is placed (and to the associated length/indicator buffer). ?NULL_REF removes the binding between a buffer and a column.
Timeout ->	integer()   infinity	Max time (ms) for serving the request.
Result ->	?SQL_SUCCESS   ?SQL_SUCCESS_WITH_INFO   ?SQL_ERROR   ?SQL_INVALID_HANDLE	Result macro.

### 1.3.3 sql\_bind\_parameter/[8, 9]

#### Description:

Binds a buffer to a parameter marker in an SQL statement. See SQLBindParameter in [1].

NOTE: The memory associated with RefBuf has to be allocated already.

#### Differences from the ODBC Function:

Only input parameters is supported. Neither binding of arrays nor the use of binding offsets is supported. Multiple values for parameters is not supported. The use of arbitrary 32-bit integers for the ParameterValuePtr argument in the ODBC function (RefBuf here) is not supported. The parameters Server and Timeout have been added. The input parameters ValueType,

Uppgjord (även faktaansvarig om annan) - <i>Prepared (also subject responsible if other)</i>	Nr - No.		
Dokansv/Godkänd - <i>Doc respons/Approved</i>	Kontr - <i>Checked</i>	Datum - <i>Date</i> 1999-02-16	Rev PA1 File

ParameterValuePtr, BufferLength, and StrLen\_or\_IndPtr of the ODBC function have been replaced with the RefBuf parameter (which represents the same data).

### Arguments and Return Value:

sql\_bind\_parameter(Server, RefStmtHandle, ParNum, InOutType, ParSqlType, ColSize, DecDigs, RefBuf) ->

sql\_bind\_parameter(Server, RefStmtHandle, ParNum, InOutType, ParSqlType, ColSize, DecDigs, RefBuf, Timeout) ->

#### Result

Server ->	pid()   Name   {global, Name}   {Name, Node}	The pid of the server, a registered name, a globally registered name, or a registered name on a remote node.
RefStmtHandle ->	term()	Reference to the statement handle.
ParNum ->	integer()	Parameter number from left to right starting at 1.
InOutType ->	?SQL_PARAM_INPUT	Input parameter.
ParSqlType ->	integer()	A macro representing a supported SQL data type.
ColSize ->	integer()	The maximum size of the parameter. Ignored for certain values of ParSqlType. See also appendix D in [1].
DecDigs ->	integer()	The scale of the parameter. Ignored for certain values of ParSqlType. See also appendix D in [1].
RefBuf ->	term()	Reference to the buffer where parameters are placed before execution of the statement (and to the associated length/indicator buffer, which must contain one of the following values: The length of the data in the data buffer referred to by RefBuf, ?SQL_NTS, ?SQL_NULL_DATA, ?SQL_DATA_AT_EXEC, or the result of the macro ?SQL_LEN_DATA_AT_EXEC(L), where L is the length of the data to be sent for the parameter,

Uppgjord (även faktaansvarig om annan) - <i>Prepared (also subject responsible if other)</i>		Nr - No.		
Dokansv/Godkänd - <i>Doc respons/Approved</i>	Kontr - <i>Checked</i>	Datum - <i>Date</i>	Rev	File
		1999-02-16	PA1	

Timeout ->	integer()   infinity	for the length/indicator buffer). Max time (ms) for serving the request.
Result ->	?SQL_SUCCESS   ?SQL_SUCCESS_WITH_INFO   ?SQL_ERROR   ?SQL_INVALID_HANDLE	Result macro.

### 1.3.4 sql\_close\_cursor/[2, 3]

#### Description:

Closes a cursor that has been opened on a statement and discards pending results. See SQLCloseCursor in [1].

#### Differences from the ODBC Function:

The parameters Server and Timeout have been added.

#### Arguments and Return Value:

sql\_close\_cursor(Server, RefStmtHandle) ->  
sql\_close\_cursor(Server, RefStmtHandle, Timeout) ->  
Result

Server ->	pid()   Name   {global, Name}   {Name, Node}	The pid of the server, a registered name, a globally registered name, or a registered name on a remote node.
RefStmtHandle ->	term()	Reference to the statement handle.
Timeout ->	integer()   infinity	Max time (ms) for serving the request.
Result ->	?SQL_SUCCESS   ?SQL_SUCCESS_WITH_INFO   ?SQL_ERROR   ?SQL_INVALID_HANDLE	Result macro.

### 1.3.5 sql\_connect/[5, 6]

#### Description:

Uppgjord (även faktaansvarig om annan) - <i>Prepared (also subject responsible if other)</i>	Nr - No.		
Dokansv/Godkänd - <i>Doc respons/Approved</i>	Kontr - <i>Checked</i>	Datum - <i>Date</i> 1999-02-16	Rev PA1 File

Establishes a connection to a driver and a data source. See SQLConnect in [1].

#### Differences from the ODBC Function:

Connection pooling is not supported.

The parameters Server and Timeout have been added. The input parameters NameLength1, NameLength2, and NameLength3 of the ODBC function have been excluded.

#### Arguments and Return Value:

sql\_connect(Server, RefConnHandle, DSN, UID, Auth) ->  
 sql\_connect(Server, RefConnHandle, DSN, UID, Auth, Timeout) ->  
 Result

Server ->	pid()   Name   {global, Name}   {Name, Node}	The pid of the server, a registered name, a globally registered name, or a registered name on a remote node.
RefConnHandle ->	term()	Reference to the connection handle.
DSN ->	string()	The name of the data source.
UID ->	string()	The user ID.
Auth ->	string()	The user's password for the data source.
Timeout ->	integer()   infinity	Max time (ms) for serving the request.
Result ->	?SQL_SUCCESS   ?SQL_SUCCESS_WITH_INFO   ?SQL_ERROR   ?SQL_INVALID_HANDLE	Result macro.

### 1.3.6 sql\_describe\_col/[4, 5]

#### Description:

Returns the result descriptor – column name, type, column size, decimal digits, and nullability – for one column in the result set. See SQLDescribeCol in [1].

Uppgjord (även faktaansvarig om annan) - Prepared (also subject responsible if other)		Nr - No.		
Dokansv/Godkänd - Doc respons/Approved	Kontr - Checked	Datum - Date	Rev	File
		1999-02-16	PA1	

To decide the buffer size needed (how many characters or bytes) to retrieve data for the column it is necessary to calculate the display size (see also appendix D in [1]). The function `display_size(SqlType, ColSize) -> integer()` does the calculation. The input parameters are returned by `sql_describe_col`.

#### Differences from the ODBC Function:

The function does not support retrieval of bookmark column data.

The parameters `Server` and `Timeout` have been added. The output parameters `ColumnName`, `NameLengthPtr`, `DataTypePtr`, `ColumnSizePtr`, `DecimalDigitsPtr`, and `NullablePtr` of the ODBC function have been changed into the returned values `ColName`, `LenColName`, `SqlType`, `ColSize`, `DecDigs`, and `Nullable`. `BufLenColName` must be  $> 0$ .

#### Arguments and Return Value:

`sql_describe_col(Server, RefStmtHandle, ColNum, BufLenColName) ->`  
`sql_describe_col(Server, RefStmtHandle, ColNum, BufLenColName,`  
`Timeout) ->`

{Result,  
{ColName, LenColName}, SqlType, ColSize, DecDigs, Nullable}

Server ->	pid()   Name   {global, Name}   {Name, Node}	The pid of the server, a registered name, a globally registered name, or a registered name on a remote node.
RefStmtHandle ->	term()	Reference to the statement handle.
ColNum ->	integer()	The column number from left to right, starting at 1.
BufLenColName ->	integer()	Length ( $>0$ ) of ColName buffer. Allow room for null-termination
Timeout ->	integer()   infinity	Max time (ms) for serving the request.
Result ->	?SQL_SUCCESS   ?SQL_SUCCESS_WITH_INFO	Result macro.

Uppgjord (även faktaansvarig om annan) - <i>Prepared (also subject responsible if other)</i>	Nr - No.		
Dokansv/Godkänd - <i>Doc respons/Approved</i>	Kontr - <i>Checked</i>	Datum - <i>Date</i> 1999-02-16	Rev PA1 File

ColName ->	?SQL_ERROR   ?SQL_INVALID_HANDLE string()	The column name.
LenColName ->	integer()	Actual length of ColName.
SqlType ->	integer()	An ODBC SQL data type (ODBC supported data types are supplied through macros, or a driver-specific type (not supplied through macros).
ColSize ->	integer()	The precision of the column (see appendix D in [1]). If the precision can not be determined, 0 is returned.
DecDigs ->	integer()	The scale of the column (see appendix D in [1]). If the scale can not be determined, or is not applicable, 0 is returned.
Nullable ->	?SQL_NO_NULLS   ?SQL_NULLABLE   ?SQL_NULLABLE_UNKNOWN	Indicates whether the column allows NULL values or not.

### 1.3.7 sql\_disconnect/[2, 3]

#### Description:

Closes the connection associated with a specific connection handle. See SQLDisconnect in [1].

#### Differences from the ODBC Function:

Uppgjord (även faktaansvarig om annan) - Prepared (also subject responsible if other)		Nr - No.		
Dokansv/Godkänd - Doc respons/Approved	Kontr - Checked	Datum - Date	Rev	File
		1999-02-16	PA1	

Connection pooling is not supported.

The parameters Server and Timeout have been added.

Arguments and Return Value:

sql\_disconnect(Server, RefConnHandle) ->  
 sql\_disconnect(Server, RefConnHandle, Timeout) ->  
 Result

Server ->	pid()   Name   { global, Name }   { Name, Node }	The pid of the server, a registered name, a globally registered name, or a registered name on a remote node.
RefConnHandle ->	term()	Reference to the connection handle.
Timeout ->	integer()   infinity	Max time (ms) for serving the request.
Result ->	?SQL_SUCCESS   ?SQL_SUCCESS_WITH_INFO   ?SQL_ERROR   ?SQL_INVALID_HANDLE	Result macro.

### 1.3.8 sql\_driver\_connect/[5, 6]

Description:

Establishes a connection to a driver and a data source which needs more connection information than SQLConnect offers. See SQLDriverConnect in [1].

Differences from the ODBC Function:

The function does not support prompting with pop-ups, so the connection string supplied must be complete.

The parameters Server and Timeout have been added. The input parameters WindowHandle and StringLength1 of the ODBC function have been excluded. The output parameters OutConnectionString and StringLength2Ptr have been changed into the returned values OutConnStr and LenOutConnStr. BufLenOutConnStr must be > 0.

Arguments and Return Value:

Uppgjord (även faktaansvarig om annan) - <i>Prepared (also subject responsible if other)</i>	Nr - No.		
Dokansv/Godkänd - <i>Doc respons/Approved</i>	Kontr - <i>Checked</i>	Datum - <i>Date</i>	Rev PA1
		1999-02-16	File

```

sql_driver_connect(Server, RefConnHandle, InConnStr,
                  BufLenOutConnStr, DrvCompletion) ->
sql_driver_connect(Server, RefConnHandle, InConnStr,
                  BufLenOutConnStr, DrvCompletion, Timeout) ->
  {Result, {OutConnStr, LenOutConnStr}}

```

Server ->	pid()   Name   {global, Name}   {Name, Node}	The pid of the server, a registered name, a globally registered name, or a registered name on a remote node.
RefConnHandle ->	term()	Reference to the connection handle.
InConnStr ->	string()	A complete connection string.
BufLenOutConnStr ->	integer()	Length (>0) of OutConnStr buffer. Allow room for null-termination
DrvCompletion ->	?SQL_DRIVER_NOPROMPT	No prompting with pop-ups.
Timeout ->	integer()   infinity	Max time (ms) for serving the request.
Result ->	?SQL_SUCCESS   ?SQL_SUCCESS_WITH_INFO   ?SQL_NO_DATA   ?SQL_ERROR   ?SQL_INVALID_HANDLE	Result macro.
OutConnStr ->	string()	A complete connection string.
LenOutConnStr ->	integer()	The length of OutConnStr before truncation.

### 1.3.9 sql\_end\_tran/[4, 5]

#### Description:

Requests a commit or rollback operation for all active operations on all statement handles associated with a connection. It can also request that a commit or rollback operation be performed for all connections associated with the environment handle. See `SQLEndTran` in [1].

Uppgjord (även faktaansvarig om annan) - Prepared (also subject responsible if other)		Nr - No.		
Dokansv/Godkänd - Doc respons/Approved	Kontr - Checked	Datum - Date	Rev	File
		1999-02-16	PA1	

NOTE: Rollback of transactions may be unsupported by core level drivers.

#### Differences from the ODBC Function:

The parameters Server and Timeout have been added.

#### Arguments and Return Value:

sql\_end\_tran(Server, HandleType, RefHandle, ComplType) ->  
 sql\_end\_tran(Server, HandleType, RefHandle, ComplType, Timeout) ->  
 Result

Server ->	pid()   Name   {global, Name}   {Name, Node}	The pid of the server, a registered name, a globally registered name, or a registered name on a remote node.
HandleType ->	?SQL_HANDLE_ENV   ?SQL_HANDLE_DBC	The type of handle for which to perform the transaction (all connections associated with an environment or a specific connection).
RefHandle ->	term()	Reference to the handle
ComplType ->	?SQL_COMMIT ?SQL_ROLLBACK	Commit operation. Rollback operation.
Timeout ->	integer()   infinity	Max time (ms) for serving the request.
Result ->	?SQL_SUCCESS   ?SQL_SUCCESS_WITH_INFO   ?SQL_ERROR   ?SQL_INVALID_HANDLE	Result macro.

### 1.3.10 sql\_exec\_direct/[3, 4]

#### Description:

Executes a preparable statement using the current values of the parameter marker buffers (bound with sql\_bind\_parameter/[8, 9]) if any parameter markers exist in the statement. See SQLExecDirect in [1].

#### Differences from the ODBC Function:

?SQL\_NO\_DATA is returned only in connection with positioned updates, which are not supported.

Uppgjord (även faktaansvarig om annan) - <i>Prepared (also subject responsible if other)</i>	Nr - No.		
Dokansv/Godkänd - <i>Doc respons/Approved</i>	Kontr - <i>Checked</i>	Datum - <i>Date</i>	Rev File
		1999-02-16	PA1

The parameters Server and Timeout have been added. The input parameter TextLength of the ODBC function has been excluded.

#### Arguments and Return Value:

sql\_exec\_direct(Server, RefStmtHandle, Stmt) ->  
 sql\_exec\_direct(Server, RefStmtHandle, Stmt, Timeout) ->  
 Result

Server ->	pid()   Name   {global, Name}   {Name, Node}	The pid of the server, a registered name, a globally registered name, or a registered name on a remote node.
RefStmtHandle ->	term()	Reference to the statement handle.
Stmt ->	string()	An SQL statement.
Timeout ->	integer()   infinity	Max time (ms) for serving the request.
Result ->	?SQL_SUCCESS   ?SQL_SUCCESS_WITH_INFO   ?SQL_NEED_DATA   ?SQL_ERROR   ?SQL_NO_DATA   ?SQL_INVALID_HANDLE	Result macro.

### 1.3.11 sql\_fetch/[2, 3]

#### Description:

Fetches a row of data from a result set. The driver returns data for all columns that were bound to storage locations with sql\_bind\_col/[4, 5]. See SQLFetch in [1].

#### Differences from the ODBC Function:

The parameters Server and Timeout have been added.

#### Arguments and Return Value:

sql\_fetch(Server, RefStmtHandle) ->  
 sql\_fetch(Server, RefStmtHandle, Timeout) ->  
 Result

Uppgjord (även faktaansvarig om annan) - Prepared (also subject responsible if other)		Nr - No.		
Dokansv/Godkänd - Doc respons/Approved	Kontr - Checked	Datum - Date	Rev	File
		1999-02-16	PA1	

Server ->	pid()   Name   {global, Name}   {Name, Node}	The pid of the server, a registered name, a globally registered name, or a registered name on a remote node.
RefStmtHandle ->	term()	Reference to the statement handle.
Timeout ->	integer()   infinity	Max time (ms) for serving the request.
Result ->	?SQL_SUCCESS   ?SQL_SUCCESS_WITH_INFO   ?SQL_NO_DATA   ?SQL_ERROR   ?SQL_INVALID_HANDLE	Result macro.

### 1.3.12 sql\_free\_handle/[3, 4]

#### Description:

Releases a handle and frees all resources associated with it. See SQLFreeHandle in [1].

#### Differences from the ODBC Function:

The function does not support deallocation of descriptor handles.

The parameters Server and Timeout have been added.

#### Arguments and Return Value:

sql\_free\_handle(Server, HandleType, RefHandle) ->  
sql\_free\_handle(Server, HandleType, RefHandle, Timeout) ->  
Result

Server ->	pid()   Name   {global, Name}   {Name, Node}	The pid of the server, a registered name, a globally registered name, or a registered name on a remote node.
HandleType ->	?SQL_HANDLE_ENV   ?SQL_HANDLE_DBC   ?SQL_HANDLE_STMT	Macros which define the type of handle to free.
RefHandle ->	term()	Reference to the handle.
Timeout ->	integer()   infinity	Max time (ms) for serving the request.

Uppgjord (även faktaansvarig om annan) - <i>Prepared (also subject responsible if other)</i>	Nr - No.		
Dokansv/Godkänd - <i>Doc respons/Approved</i>	Kontr - <i>Checked</i>	Datum - <i>Date</i>	Rev PA1
		1999-02-16	File

Result ->        ?SQL\_SUCCESS |                    Result macro.  
                   ?SQL\_ERROR |  
                   ?SQL\_INVALID\_HANDLE

### 1.3.13        sql\_get\_connect\_attr/[4, 5]

#### Description:

Returns the current setting of a connection attribute. See SQLGetConnectAttr in [1].

#### Differences from the ODBC Function:

Only the following attributes, and their possible values, are supported (through macros). More information can be found under SQLSetConnectAttr in [1]. Driver-specific attributes are not supported through macros, but can be retrieved, if they are of character or signed/unsigned long integer types.

- ?SQL\_ATTR\_ACCESS\_MODE
- ?SQL\_ATTR\_AUTOCOMMIT
- ?SQL\_ATTR\_ODBC\_CURSORS
- ?SQL\_ATTR\_TRACE
- ?SQL\_ATTR\_TRACEFILE
- ?SQL\_ATTR\_TRANSLATE\_LIB
- ?SQL\_ATTR\_TRANSLATE\_OPTION

According to [1], BufLen (BufferLength) can be set to SQL\_NTS. This is probably not correct, since it would make it impossible for the driver to detect that data needs to be truncated. Hence, the SQL\_NTS value has been disallowed.

The function takes a BufType parameter to distinguish between character type attributes and numeric type attributes. For character data the maximum string length must be supplied (allow room for null-termination). For driver-specific numeric type attributes, a subtype must be supplied. The returned value is either a tuple containing the attribute string and its length, or an integer, depending on the specified buffer type.

The parameters Server and Timeout have been added. The output parameters ValuePtr and StringLengthPtr of the ODBC function have been changed into the returned values CharValue and LenCharValue for character type attributes and NumValue for integer types. The input parameter BufferLength has been included in the BufType parameter. BufLen must be > 0.

#### Arguments and Return Value:

Uppgjord (även faktaansvarig om annan) - Prepared (also subject responsible if other)		Nr - No.		
Dokansv/Godkänd - Doc respons/Approved	Kontr - Checked	Datum - Date	Rev	File
		1999-02-16	PA1	

sql\_get\_connect\_attr(Server, RefConnHandle, Attr, BufType) ->  
 sql\_get\_connect\_attr(Server, RefConnHandle, Attr, BufType, Timeout) ->  
 {Result, Value}

Server ->	pid()   Name   {global, Name}   {Name, Node}	The pid of the server, a registered name, a globally registered name, or a registered name on a remote node.
RefConnHandle ->	term()	Reference to the connection handle.
Attr ->	integer()	One of the attributes described above or a driver-specific attribute.
BufType ->	{?SQL_C_CHAR, BufLen}   ?SQL_C_ULONG   {?SQL_C_ULONG, IntType}	The buffer type used for retrieving the data. For character type data also the buffer size. For integer type data that is driver-specific, also a subtype.
BufLen ->	integer()	Buffer size (>0) for character type data. Allow room for null-termination
IntType ->	?SQL_IS_UIINTEGER   ?SQL_IS_INTEGER	Used only for driver- specific attributes. See SQLGetConnectAttr in [1].
Timeout ->	integer()   infinity	Max time (ms) for serving the request.
Result ->	?SQL_SUCCESS   ?SQL_SUCCESS_WITH_INFO   ?SQL_NO_DATA   ?SQL_ERROR   ?SQL_INVALID_HANDLE	Result macro.
Value ->	{CharValue, LenCharValue}   NumValue	Attribute data.
CharValue ->	string()	The value of the attribute when of character type.
LenCharValue ->	integer()	The length of

Uppgjord (även faktaansvarig om annan) - <i>Prepared (also subject responsible if other)</i>	Nr - No.		
Dokansv/Godkänd - <i>Doc respons/Approved</i>	Kontr - <i>Checked</i>	Datum - <i>Date</i> 1999-02-16	Rev PA1 File

NumValue -> integer()

CharValue before truncation.  
The value of the attribute when of numeric type.

### 1.3.14 sql\_get\_diag\_rec/[5, 6]

#### Description:

Retrieves the current values of multiple fields of a diagnostic record that contains error, warning, and status information. See SQLGetDiagRec in [1].

#### Differences from the ODBC Function:

Retrieving information associated with descriptor handles is not supported.

The parameters Server and Timeout have been added. The output parameters SqlState, NativeErrorPtr, MessageText, and TextLengthPtr of the ODBC function have been changed into the returned values SqlState, NativeErr, ErrMsg, and LenErrMsg. BufLenErrMsg must be > 0.

#### Arguments and Return Value:

sql\_get\_diag\_rec(Server, HandleType, RefHandle, RecNum,  
BufLenErrMsg) ->

sql\_get\_diag\_rec(Server, HandleType, RefHandle, RecNum,  
BufLenErrMsg, Timeout) ->

{Result, SqlState, NativeErr, {ErrMsg, LenErrMsg}}

Server ->	pid()   Name   {global, Name}   {Name, Node}	The pid of the server, a registered name, a globally registered name, or a registered name on a remote node.
HandleType ->	?SQL_HANDLE_ENV   ?SQL_HANDLE_DBC   ?SQL_HANDLE_STMT	The type of handle for which to retrieve information.
RefHandle ->	term()	Reference to the handle.
RecNum ->	integer()	Indicates the status record from which to retrieve information (> 0).
BufLenErrMsg ->	integer()	Length of ErrMsg buffer (>0). Allow room for null- termination

Uppgjord (även faktaansvarig om annan) - Prepared (also subject responsible if other)		Nr - No.		
Dokansv/Godkänd - Doc respons/Approved	Kontr - Checked	Datum - Date	Rev	File
		1999-02-16	PA1	

Timeout ->	integer()   infinity	Max time (ms) for serving the request.
Result ->	?SQL_SUCCESS   ?SQL_SUCCESS_WITH_INFO   ?SQL_NO_DATA   ?SQL_ERROR   ?SQL_INVALID_HANDLE	Result macro.
SqlState ->	string()	The SQL state pertaining to the diagnostic record.
NativeErr ->	integer()	Data-source specific error code.
ErrMsg ->	string()	Error message.
LenErrMsg ->	integer()	The length of ErrMsg before truncation.

### 1.3.15 sql\_num\_result\_cols/[2, 3]

#### Description:

Returns the number of columns in a result set. See SQLNumResultCols in [1].

#### Differences from the ODBC Function:

The parameters Server and Timeout have been added. The output parameter ColumnCountPtr of SQLNumResultCols has been changed into the returned value ColCount.

#### Arguments and Return Value:

sql\_num\_result\_cols(Server, RefStmtHandle) ->  
sql\_num\_result\_cols(Server, RefStmtHandle, Timeout) ->  
{Result, ColCount}

Server ->	pid()   Name   {global, Name}   {Name, Node}	The pid of the server, a registered name, a globally registered name, or a registered name on a remote node.
RefStmtHandle ->	term()	Reference to the statement handle.
Timeout ->	integer()   infinity	Max time (ms) for serving the request.
Result ->	?SQL_SUCCESS   ?SQL_SUCCESS_WITH_INFO	Result macro.

Uppgjord (även faktaansvarig om annan) - <i>Prepared (also subject responsible if other)</i>	Nr - No.		
Dokansv/Godkänd - <i>Doc respons/Approved</i>	Kontr - <i>Checked</i>	Datum - <i>Date</i> 1999-02-16	Rev PA1 File

ColCount -> ?SQL\_ERROR |  
?SQL\_INVALID\_HANDLE  
integer() The number of columns in the result set.

### 1.3.16 sql\_row\_count/[2, 3]

#### Description:

Returns the number of rows affected by an UPDATE, INSERT, or DELETE statement. See SQLRowCount in [1].

#### Differences from the ODBC Function:

The parameters Server and Timeout have been added. The output parameter RowCountPtr of the ODBC function has been changed into the returned value RowCount.

#### Arguments and Return Value:

sql\_row\_count(Server, RefStmtHandle) ->  
sql\_row\_count(Server, RefStmtHandle, Timeout) ->  
{Result, RowCount}

Server ->	pid()   Name   {global, Name}   {Name, Node}	The pid of the server, a registered name, a globally registered name, or a registered name on a remote node.
RefStmtHandle ->	term()	Reference to the statement handle.
Timeout ->	integer()   infinity	Max time (ms) for serving the request.
Result ->	?SQL_SUCCESS   ?SQL_SUCCESS_WITH_INFO   ?SQL_ERROR   ?SQL_INVALID_HANDLE	Result macro.
RowCount ->	integer()	The number of affected rows. If the number of affected rows is not available -1 is returned. For exceptions, see SQLRowCount in [1].

Uppgjord (även faktaansvarig om annan) - Prepared (also subject responsible if other)		Nr - No.		
Dokansv/Godkänd - Doc respons/Approved	Kontr - Checked	Datum - Date	Rev	File
		1999-02-16	PA1	

### 1.3.17 sql\_set\_connect\_attr/[5, 6]

#### Description:

Sets attributes that govern aspects of connections. See `SQLSetConnectAttr` in [1]. The supported attributes are listed under `sql_get_connect_attr/[4, 5]`.

Driver-specific attributes are not supported through macros, but can be set, if they are strings or signed/unsigned long integers.

#### Differences from the ODBC Function:

Only character and signed/unsigned long integer attribute types are supported.

The parameters `Server` and `Timeout` have been added. The input parameter `StringLength` of the ODBC function has been replaced with the input parameter `BufType`.

#### Arguments and Return Value:

`sql_set_connect_attr(Server, RefConnHandle, Attr, Value, BufType) ->`  
`sql_set_connect_attr(Server, RefConnHandle, Attr, Value, BufType,`  
`Timeout) ->`

#### Result

Server ->	pid()   Name   {global, Name}   {Name, Node}	The pid of the server, a registered name, a globally registered name, or a registered name on a remote node.
RefConnHandle ->	term()	Reference to the connection handle.
Attr ->	integer()	One of the attributes described under <code>sql_get_connect_attr/[4, 5]</code> or a driver-specific attribute. The attributes defined by ODBC are supplied through macros, but driver- specific attributes are not.
Value ->	string()   integer()	The new attribute value.
BufType ->	?SQL_C_CHAR	The buffer type. Either a

Uppgjord (även faktaansvarig om annan) - <i>Prepared (also subject responsible if other)</i>	Nr - No.		
Dokansv/Godkänd - <i>Doc respons/Approved</i>	Kontr - <i>Checked</i>	Datum - <i>Date</i> 1999-02-16	Rev PA1 File

	?SQL_C_ULONG	(null-terminated) string, an {?SQL_C_ULONG, IntType} ODBC defined attribute of integer type, or a driver-specific attribute of integer type (which also has a subtype).
IntType ->	?SQL_IS_UIINTEGER   ?SQL_IS_INTEGER	Subtype for driver-specific integer attributes.
Timeout ->	integer()   infinity	Max time (ms) for serving the request.
Result ->	?SQL_SUCCESS   ?SQL_SUCCESS_WITH_INFO   ?SQL_ERROR   ?SQL_INVALID_HANDLE	Result macro.

### 1.3.18 sql\_set\_env\_attr/[5, 6]

#### Description:

Sets attributes that govern aspects of environments. See SQLSetEnvAttr in [1].

The following attributes, and their possible values, are supported (through macros). More information can be found under SQLSetEnvAttr in [1]. Other data types than character or unsigned long integer are not supported.

- ?SQL\_ATTR\_ODBC\_VERSION

#### Differences from the ODBC Function:

Only character and unsigned long integer attribute types are supported.

The parameters Server and Timeout have been added. The input parameter StringLength of the ODBC function has been replaced with the input parameter BufType.

#### Arguments and Return Value:

sql\_set\_env\_attr(Server, RefEnvHandle, Attr, Value, BufType) ->  
sql\_set\_env\_attr(Server, RefEnvHandle, Attr, Value, BufType, Timeout) ->  
Result

Server ->	pid()   Name   {global, Name}	The pid of the server, a registered name, a globally registered
-----------	-------------------------------------	---

Uppgjord (även faktaansvarig om annan) - Prepared (also subject responsible if other)		Nr - No.		
Dokansv/Godkänd - Doc respons/Approved	Kontr - Checked	Datum - Date	Rev	File
		1999-02-16	PA1	

	{Name, Node}	name, or a registered name on a remote node.
RefEnvHandle ->	term()	Reference to the environment handle.
Attr ->	integer()	One of the supported attributes.
Value ->	string()   integer()	The new attribute value.
BufType ->	?SQL_C_CHAR   ?SQL_C_ULONG	The buffer type. Either a (null-terminated) string or an ODBC defined attribute of integer type.
Timeout ->	integer()   infinity	Max time (ms) for serving the request.
Result ->	?SQL_SUCCESS   ?SQL_SUCCESS_WITH_INFO   ?SQL_ERROR   ?SQL_INVALID_HANDLE	Result macro.

### 1.3.19 alloc\_buffer/[3, 4]

#### Description:

Allocates a deferred data buffer and an associated length/indicator buffer.

#### Arguments and Return Value:

alloc\_buffer(Server, BufCType, Size) ->  
 alloc\_buffer(Server, BufCType, Size, Timeout) ->  
 {ok, RefBuf}

Server ->	pid()   Name   {global, Name}   {Name, Node}	The pid of the server process, a registered name, a globally registered name, or a registered name on a remote node.
BufCType ->	?SQL_C_CHAR   ?SQL_C_BINARY	The C data type of the buffer.
Size ->	integer()	The buffer size (>0). For character data, allow room for null-termination
Timeout ->	integer()   infinity	Max time (ms) for serving the request.
RefBuf ->	term()	A handle to the buffer.

Uppgjord (även faktaansvarig om annan) - <i>Prepared (also subject responsible if other)</i>	Nr - No.		
Dokansv/Godkänd - <i>Doc respons/Approved</i>	Kontr - <i>Checked</i>	Datum - <i>Date</i>	Rev   File
		1999-02-16	PA1

### 1.3.20 dealloc\_buffer/[2, 3]

#### Description:

Deallocates a deferred data buffer and the associated length/indicator buffer.

#### Arguments and Return Value:

dealloc\_buffer(Server, RefBuf) ->  
 dealloc\_buffer(Server, RefBuf, Timeout) ->  
 ok

Server ->	pid()   Name   {global, Name}   {Name, Node}	The pid of the server process, a registered name, a globally registered name, or a registered name on a remote node.
RefBuf ->	term()	A handle to the buffer.
Timeout ->	integer()   infinity	Max time (ms) for serving the request.

### 1.3.21 read\_buffer/[2, 3]

#### Description:

Returns the contents of a deferred data buffer and its associated length/indicator buffer. Used in connection with sql\_fetch/[2, 3].

NOTE: When the returned Value is a binary it is not given that it can be converted into an Erlang term with binary\_to\_term/1. It can be done if Value was created using term\_to\_binary/1 and if it was stored in the database through this interface. Value can always be converted into a list with binary\_to\_list/1.

#### Arguments and Return Value:

read\_buffer(Server, RefBuf) ->  
 read\_buffer(Server, RefBuf, Timeout) ->  
 {ok, {Value, LenInd}}

Server ->	pid()   Name   {global, Name}   {Name, Node}	The pid of the server process, a registered name, a globally registered name, or a registered name on a remote node.
RefBuf ->	term()	A handle to the buffer.
Timeout ->	integer()	Max time (ms) for serving the request.

Uppgjord (även faktaansvarig om annan) - <i>Prepared (also subject responsible/lf other)</i>		Nr - No.		
Dokansv/Godkänd - <i>Doc respons/Approved</i>	Kontr - <i>Checked</i>	Datum - <i>Date</i>	Rev	File
		1999-02-16	PA1	

Value ->	infinity string()   binary()	Contents of the buffer associated with RefBuf. The type is determined by the type chosen for RefBuf when it was allocated.
LenInd ->	integer()   ?SQL_NULL_DATA   ?SQL_NO_TOTAL	Length/indicator value associated with RefBuf.

### 1.3.22 write\_buffer/[3, 4]

#### Description:

Writes a value to a deferred data buffer.

NOTE: If the function fails, the contents of the deferred data buffer referred to by RefBuf, and the associated length/indicator buffer, is undefined.

#### Arguments and Return Value:

write\_buffer(Server, RefBuf, { Value, LenInd }) ->  
write\_buffer(Server, RefBuf, { Value, LenInd }, Timeout) ->  
ok

Server ->	pid()   Name   { global, Name }   { Name, Node }	The pid of the server, a registered name, a globally registered name, or a registered name on a remote node.
RefBuf ->	term()	A handle to the buffer.
Value ->	string()   binary()	Value to write to the buffer associated with RefBuf.
LenInd ->	integer()   ?SQL_NULL_DATA   ?SQL_NTS   ?SQL_DATA_AT_EXEC   no_len_ind	Length/indicator value associated with RefBuf. See SQLBindParameter in [1] also. If the associated length/indicator buffer should not be written, specify no_len_ind.
Timeout ->	integer()   infinity	Max time (ms) for serving the request.

Uppgjord (även faktaansvarig om annan) - <i>Prepared (also subject responsible if other)</i>	Nr - No.		
Dokansv/Godkänd - <i>Doc respons/Approved</i>	Kontr - <i>Checked</i>	Datum - <i>Date</i> 1999-02-16	Rev PA1 File

## 1.4 Error Messages and Exceptions

Errors caused by inability to contact the C node, allocate memory, or otherwise call ODBC functions cause exceptions. Exceptions are common to all functions. Errors caused by ODBC not being able to execute calls are reported through returned errors.

These exceptions terminate the client only.

{'EXIT', {badarg, M, F, A, ArgNo, Info}}	Argument is of wrong type or out of range.
{'EXIT', {internal_error, Info}}	Internal error.
{'EXIT', GenServerSpecificInfo}	Error detected by gen_server.

These cause the ODBC server, and the C node, to terminate as well:

{'EXIT', {timeout, Info}}	Timeout expired.
{'EXIT', {stopped, Reason}}	The ODBC server died.

## 1.5 References

- [1] Microsoft ODBC 3.0, Programmer's Reference and SDK Guide