

SQL_BINDIN

SQL_BINDIN action

Function	The action specifies values of the parameters and executes a SQL command SELECT prepared by action <code>SQL_PREPARE</code> , if the latter action used parameterization and a keyword <code>BINDOUT</code> .														
Declaration	<pre>SQL_BINDIN handleIdent_Int, retCodeIdent_Int, _Par1, _Par2, ... SQL_BINDIN handleIdent_Int, retCodeIdent_Int, _VarRowIdent</pre>														
Parameters	<table><tr><td>handleIdent_Int</td><td>in</td><td>Identifier - the unique number (handle) of a connection.</td></tr><tr><td>retCodeIdent_Int</td><td>out</td><td>Return code identifier.</td></tr><tr><td>_Par1, _Par2, ...</td><td>in</td><td>List of objects, constants or local variables, which will specify the values of parameters of parameterized SQL command SELECT.</td></tr><tr><td>_VarRowIdent</td><td>in</td><td>Reference to a row of local variable of the <i>Record</i> type or to a row of structured variable. The row's values will specify the values of parameters of parameterized SQL command SELECT.</td></tr></table>			handleIdent_Int	in	Identifier - the unique number (handle) of a connection.	retCodeIdent_Int	out	Return code identifier .	_Par1, _Par2, ...	in	List of objects, constants or local variables , which will specify the values of parameters of parameterized SQL command SELECT .	_VarRowIdent	in	Reference to a row of local variable of the <i>Record</i> type or to a row of structured variable . The row's values will specify the values of parameters of parameterized SQL command SELECT .
handleIdent_Int	in	Identifier - the unique number (handle) of a connection.													
retCodeIdent_Int	out	Return code identifier .													
_Par1, _Par2, ...	in	List of objects, constants or local variables , which will specify the values of parameters of parameterized SQL command SELECT .													
_VarRowIdent	in	Reference to a row of local variable of the <i>Record</i> type or to a row of structured variable . The row's values will specify the values of parameters of parameterized SQL command SELECT .													
Return code	The value of the parameter <i>transHandle_Int</i> . See the table of error codes . It is possible to get extended error information .														
Description	<p>Reading a database by the command SELECT is implemented in two or three phases. The first (preparatory) phase is executed by the action <code>SQL_PREPARE</code>. The command SELECT, defined by a value of the expression <i>selectStringExpr</i>, is prepared (and if the keyword BINDOUT is not used, then also executed) in the database.</p> <p>If the keyword BINDOUT was used, it means that the SQL SELECT command was parameterized, and the second phase is needed. The command <code>SQL_BINDIN</code> must be used to specify the values of the input parameters and execute the SQL statement.</p> <p>The last phase is the sequential reading of the rows, prepared by the command SELECT, using the action <code>SQL_FETCH</code>.</p> <p>Note: By using parameterization it is possible to make the work of SQL database easier, because the preparation (compilation) of parameterized SQL query will be performed only once (by the action <code>SQL_PREPARE</code>). Consequently the values of parameters must be specified by the action <code>SQL_BINDIN</code> (which will also execute the SQL command) and then the action <code>SQL_FETCH</code> may be called once or more times to obtain the results. Then it is possible to set new values of the parameters and re-execute the SQL command by repeating the action <code>SQL_BINDIN</code> and obtain the new results by one or more calls of the action <code>SQL_FETCH</code>.</p> <p>By proper setting of the database parameters (e.g. Oracle: <i>session_cached_cursors</i>) it is possible to ensure recycling of cursors (compiled statements) between the calls of <code>SQL_PREPARE</code>.</p>														
Example	Work with a database (actions SQL_ ...)														

```

INT  _handle      ; handle to database
INT  _retCode     ; return code
TEXT _name        ; product name
TEXT _type        ; product type
                        ; parameterized SQL command
TEXT _sql = "SELECT Name, Type FROM Products WHERE ID>= #PAR# AND ID<=
#PAR#"

SQL_CONNECT MyDatabase, _handle, _retCode
SQL_PREPARE _handle, _retCode, _sql BINDOUT _name, _type
SQL_BINDIN  _handle, _retCode, 1, 100 ; read all products between 1 and
100

DO_LOOP
    SQL_FETCH _handle, _retCode
    EXIT_LOOP _retCode # _ERR_NO_ERROR
    ; data processing goes here
END_LOOP

SQL_FREE _handle
SQL_DISCONNECT _handle

```

Related topics

[DB_TRANS_OPEN](#)
[DB_TRANS_COMMIT](#)
[DB_TRANS_ROLLBACK](#)
[DB_TRANS_CLOSE](#)

[SQL_CONNECT](#)
[SQL_DISCONNECT](#)
[SQL_EXEC_DIRECT](#)
[SQL_EXEC_PROC](#)

[SQL_PREPARE](#)
[SQL_FETCH](#)
[SQL_FREE](#)

[SQL_SELECT](#)

All database related actions.



Related pages:

[Script actions](#)