

# SQL\_BINDIN

## SQL\_BINDIN action

Function	The action specifies values of the parameters and executes a SQL command <b>SELECT</b> prepared by action <a href="#">SQL_PREPARE</a> , if the latter action used <a href="#">parameterization</a> and a keyword <a href="#">BINDOUT</a> .														
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><a href="#">Identifier</a> - the unique number (handle) of a connection.</td></tr><tr><td>retCodeIdent_Int</td><td>out</td><td>Return code <a href="#">identifier</a>.</td></tr><tr><td>_Par1, _Par2, ...</td><td>in</td><td>List of objects, constants or <a href="#">local variables</a>, which will specify the values of parameters of <a href="#">parameterized</a> SQL command <b>SELECT</b>.</td></tr><tr><td>_VarRowIdent</td><td>in</td><td><a href="#">Reference to a row</a> of <a href="#">local variable</a> of the <i>Record</i> type or to a row of <a href="#">structured variable</a>. The row's values will specify the values of parameters of <a href="#">parameterized</a> SQL command <b>SELECT</b>.</td></tr></table>			handleIdent_Int	in	<a href="#">Identifier</a> - the unique number (handle) of a connection.	retCodeIdent_Int	out	Return code <a href="#">identifier</a> .	_Par1, _Par2, ...	in	List of objects, constants or <a href="#">local variables</a> , which will specify the values of parameters of <a href="#">parameterized</a> SQL command <b>SELECT</b> .	_VarRowIdent	in	<a href="#">Reference to a row</a> of <a href="#">local variable</a> of the <i>Record</i> type or to a row of <a href="#">structured variable</a> . The row's values will specify the values of parameters of <a href="#">parameterized</a> SQL command <b>SELECT</b> .
handleIdent_Int	in	<a href="#">Identifier</a> - the unique number (handle) of a connection.													
retCodeIdent_Int	out	Return code <a href="#">identifier</a> .													
_Par1, _Par2, ...	in	List of objects, constants or <a href="#">local variables</a> , which will specify the values of parameters of <a href="#">parameterized</a> SQL command <b>SELECT</b> .													
_VarRowIdent	in	<a href="#">Reference to a row</a> of <a href="#">local variable</a> of the <i>Record</i> type or to a row of <a href="#">structured variable</a> . The row's values will specify the values of parameters of <a href="#">parameterized</a> SQL command <b>SELECT</b> .													
Return code	The value of the parameter <i>transHandle_Int</i> . See the table of <a href="#">error codes</a> . It is possible to get <a href="#">extended error information</a> .														
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 <a href="#">SQL_PREPARE</a>. The command SELECT, defined by a value of the expression <i>selectStringExpr</i>, is prepared (and if the keyword <b>BINDOUT</b> is not used, then also executed) in the database.</p> <p>If the keyword <b>BINDOUT</b> was used, it means that the SQL SELECT command was <a href="#">parameterized</a>, and the second phase is needed. The command <b>SQL_BINDIN</b> 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 <a href="#">SQL_FETCH</a>.</p> <p><b>Note:</b> By using <a href="#">parameterization</a> 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 <a href="#">SQL_PREPARE</a>). Consequently the values of parameters must be specified by the action <b>SQL_BINDIN</b> (which will also execute the SQL command) and then the action <a href="#">SQL_FETCH</a> 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 <b>SQL_BINDIN</b> and obtain the new results by one or more calls of the action <a href="#">SQL_FETCH</a>.</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 <b>SQL_PREPARE</b>.</p>														
Example	<a href="#">Work with a database (actions SQL_ ...)</a>														

```

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](#)