

DB_READ

DB_READ and DBS_READ actions

Function	Reading of one or several row of the table.		
Declaration	<pre>DB_READ handleIdent_Int, rowIdent, retCodeIdent_Int [WHERE strExpression_Str [BINDIN varIdent1, varIdent2, ...]] [ORAHINT hintIdent_Str] DB_READ handleIdent_Int, rowIdent, retCodeIdent_Int [WHERE strExpression_Str [BINDIN structRowIdent]] [ORAHINT hintIdent_Str] DBS_READ dbObjIdent, rowIdent, retCodeIdent_Int [WHERE strExpression_Str [BINDIN varIdent1, varIdent2, ...]] [TRANS transHandle_Int] [ORAHINT hintIdent_Str] DBS_READ dbObjIdent, rowIdent, retCodeIdent_Int [WHERE strExpression_Str [BINDIN structRowIdent]] [TRANS transHandle_Int] [ORAHINT hintIdent_Str]</pre> <p>or</p> <pre>DB_READ handleIdent_Int, localStructIdent, retCodeIdent_Int, maxRows_Int [WHERE strExpression_Str [BINDIN varIdent1, varIdent2, ...]] [ORAHINT hintIdent_Str] DB_READ handleIdent_Int, localStructIdent, retCodeIdent_Int, maxRows_Int [WHERE strExpression_Str [BINDIN structRowIdent]] [ORAHINT hintIdent_Str] DBS_READ dbObjIdent, localStructIdent,retCodeIdent_Int, maxRows_Int [WHERE strExpression_Str [BINDIN varIdent1, varIdent2, ...]] [TRANStransHandle_Int] [ORAHINT hintIdent_Str] DBS_READ dbObjIdent, localStructIdent,retCodeIdent_Int, maxRows_Int [WHERE strExpression_Str [BINDIN structRowIdent]] [TRANStransHandle_Int] [ORAHINT hintIdent_Str]</pre>		
Parameters	han dle Ident _Int	in	Identifier (handle) of <i>Int</i> type of the connection with a table (DB_CONNECT).
	dbO bjId ent	in	A reference to an object of <i>Table</i> type
	rowI dent	in /o ut	Whole local structure identifier.
	retC odeI dent _Int	o ut	Return code identifier.
	loca lStr uctI dent	o ut	Whole local structure identifier.
	max Row s_Int	in	Identifier of <i>Int</i> type. Maximal number of read rows.
	strE xpre ssio n_S tr	in	Expression of <i>String</i> type, that defines a table row (rows) which is to be read. The parameter can also contains a sorting condition (e.g. " <i>ID<100 ORDER BY Name</i> "). If the expression is parameterized , the keyword BINDIN and the values of parameters (<i>structRowIdent</i> or <i>varIdent1</i> , <i>varIdent2</i> , ...) are mandatory.

varl dent 1, varl dent 2, ...	in	List of objects, constants or local variables , which will specify the values of parameters of parameterized SQL expression <i>strExpression_Str</i> .
stru ctR owl dent	in	Reference to a row of local variable of <i>Record</i> type or to a row of structured variable . The row's values will specify the values of parameters of parameterized SQL expression <i>strExpression_Str</i> .
tran sHa ndle _Int	in	Identifier of the Connection to the database.
hintl dent _Str	in	Expression of <i>String</i> type that defines Oracle SQL hint. It is used as an instruction for the performance optimizer of SQL command. The value is used without the opening and terminating characters <i>/*+ <orah int> */</i> . The example is mentioned here .

Return code

The value of the parameter *transHandle_Int*. See the table of [error codes](#). It is possible to get [extended error information](#).

Description

The **WHERE** clause for SQL command **SELECT**, which executes the selection from a table, is defined by either the value of *strExpression_Str*, or, if this value is not defined, the values from the defined (valid) key items (those items which were set as [key](#) when configuring the object [Table](#)) in the first row of structure (*rowIdent* or *localStructIdent[1]*). These values must be set before executing **DB_READ (DBS_READ)**.

It follows that **WHERE** clause is not a part of SQL command **SELECT**, if the value of *strExpression_Str* is not defined and

- there are not the key items (the items specified as [key](#) ones when configuring the object *Tabuka*),
- or there is not defined (valid) any value of the key items,
- or the structure has a null size (for *localStructIdent*).

The first form of the action declaration allows to read just one row of a table to one line of a structure.

If more rows than one meet the selection condition during the selection, the action **DB_READ (DBS_READ)** is to be read the first row. If no row meet the condition, the action will send back the error by means of *retCodeIdent_Int*. Structure type (*rowIdent*) must be equal to the table type.

The second form of the declaration allows to read one or more rows from a table. Value of the parameter *maxRows* determines the maximal number of rows. If this value is set to -1, then the system will attempt to read all rows matching the specified condition. Maximum number of rows is limited by a Database configuration parameter [Maximum returned rows](#). Exceeding this number, the first [Maximum returned rows](#) will be stored in local structure *localStructIdent* and *retCodeIdent_Int* will be set to [_ERR_DATABASE_ROWS_LIMIT](#).

If the value of *maxRows* is greater than [Maximum returned rows](#), at the most [Maximum returned rows](#) will be read. If more rows are available, *retCodeIdent_Int* will be set to error code [_ERR_DATABASE_ROWS_LIMIT](#).

If the value of the parameter *maxRows* is set to -2, the system at first detects how many rows meet given condition. If the number of rows is 0 or it is more than [Maximum returned rows](#), the system will not read any rows, or reads all detected rows otherwise. For optimization reasons, this method may be faster than the method for *maxRows=-1*. Number of rows is detected by the SQL command "**SELECT COUNT(*) FROM table_name [WHERE where_condition]**". If the condition *StrExpression_Str* is defined, it is also tested for the presence of an **ORDER BY** clause. If the result is positive, the clause is removed along with all following text. It is removed because when detecting the number of rows e.g. Sybase doesn't accept "**SELECT COUNT(*) FROM aaaa WHERE bbbb = cccc ORDER BY dddd**" and returns the error **F unction or column reference to 'dddd' in the select list must also appear in a GROUP BY**.

The parameter *localStructIdent* must be the name of a local structure of the corresponding type (i.e. with the same Structure type that is used in the configuration of the object *dbObjIdent* of [Table](#) type). If you want to be sure that the result will not be controlled by **WHERE** clause, which consists of the key items, you should change the size of local structure to 0 (zero) before calling **DB_READ (DBS_READ)**.

After reading rows from the database, they are assigned to the local variable *localStructIdent*. Its size may be changed in case of need (**REDIM**). If no row meet the selection condition, the action is to be terminated successfully (*retCodeIdent_Int* = [_ERR_NO_ERROR](#)) and the structure size *localStructIdent* is to be changed to 0.

The advantage of using the action **DBS_READ** is omitting the table opening and closing (shorter code).

For D2000 v5.00: an disadvantage of the action **DBS_READ** is in speed. Each **DBS_READ** call results in necessity to open and close the database in DBManager - it can be a time-consuming operation and it is a comparatively nonstandard method in term of databases.

The need to open and close the database may be avoided in the scope of transaction processing so that the command is followed by the parameter

TRANS

.
For D2000 v6.00 and higher: DBManager [optimization](#) (connection recycling, predefined connections) causes the action **DBS_READ** to execute as fast as the action **DB_READ** and moreover time is saved omitting the execution of the action **DB_CONNECT**.

Example

[Work with a database table \(actions DB_ ...\).](#)

Related topics

[DB_CONNECT](#)
[DB_DELETE](#)
[DB_DISCONNECT](#)
[DB_INSERT](#)
[DB_INSUPD](#)
[DB_READ_BLOB](#)
[DB_UPDATE](#)
[DB_UPDATE_BLOB](#)

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

[All database related actions](#)



Related pages:

[Script actions](#)