Parametrization of SQL commands and expression

Parameterization of SQL commands and expressions

Parameterization of SQL commands and expressions means symbolic specification of values in SQL commands and expressions and later filling-in the parameters by actual values.
An example of parameterized expression: *" Id= #PAR# AND Name LIKE #PAR# "*
An example of parameterized command: *"SELECT Name, Surname FROM Persons WHERE Id= #PAR# AND Name LIKE #PAR# "*

There are 3 ways of parameter specification usable by selected actions executed by the process D2000 DBManager:

- **By character *"?"***
  Convention used by ODBC interface (works for ODBC version of the process DBManager only). Every parameter is specified by a single question mark.
  Example: *" Id > ? AND Name = ? "*
  Note: This parameterized SQL expression won't work for OCI version of the process DBManager.
- **By expression *":name"***
  Convention used by OCI interface (works for OCI version of the process DBManager only). Every parameter is specified by a colon and a name.
  Example: *" Id > :par1 AND Name = :par2 "*
  Note 1: This parameterized SQL expression won't work for ODBC version of the process DBManager.
  Note 2: This way of parameterization supports multiple re-using of the same parameter (e.g. *:par1*) in SQL command or expression:
  "SELECT Name, Surname FROM Person WHERE Name= :par1 OR Surname = :par1 OR BirthDate > :par2 "
- **By expression *"#PAR#"***
  Internal notation for the process DbManager. ODBC version of the process DBManager replaces all expressions *#PAR#* by a character *?* (converts the notation to ODBC convention) before preparing the statement, OCI version of the process DBManager replaces all expressions *#PAR#* by expressions *:01*, *:02* etc. (converts the notation to OCI convention).
  This way of parameter specification is universal and SQL commands and expressions using it don't have to be changed when moving between ODBC and OCI versions of the process DBManager.

Parameterization can be used in SQL expression by a selected set of database-oriented actions, specifically the actions:

- DB(S)_DELETE
- DB(S)_READ
- DB(S)_UPDATE

If the SQL expression is parameterized, the action must use the keyword **BINDIN** followed by one of these choices:

- list of objects, constants or local variables,
- reference to a row of local variable of *Record* type or to a row of structured variable,

The values of parameters will be filled-in according to the values specified after the keyword **BINDIN**.

Parameterization is supported also by the action SQL_PREPARE, but in this case the list of parameters' values is not part of the action. Using the keyword **BINDOUT** (instead of **BIND**) specifies, that the SQL command is parameterized and consequently the values of the parameters must be filled-in by calling the action SQL_BINDIN. The advantage is that to obtain results corresponding to different values of the parameters the action SQL_PREPARE doesn't have to be called again - it is sufficient to repeat the action SQL_BINDIN to fill-in new values of the parameters and consequently the action SQL_FETCH can be used to obtain results.

The meaning of parameterization is to speed up and make easier the work of SQL engine. The processing of SQL command includes its parsing and compilation. If the same SQL command is reused, the SQL engine can recycle already parsed and compiled command. E.g. by the means of parameterization the commands
"SELECT Name FROM Person WHERE Id=1"
"SELECT Name FROM Person WHERE Id=5"
"SELECT Name FROM Person WHERE Id=100"
can be replaced by a single command
"SELECT Name FROM Person WHERE Id= #PAR#"
and instead of three SQL commands the database engine will parse and compile only one SQL command (and during second and further executions it can be recycled).

If the parameterization is used in the action SQL_PREPARE and parameters' values are set by multiple calls to the action SQL_BINDIN, the parsing and compilation of SQL command is guaranteed to be performed only once (during the call to SQL_PREPARE).
If the parameterization is used in the actions DB(S)_DELETE, DB(S)_READ or DB(S)_UPDATE, it is recommended to modify the database parameters (e. g. Oracle: *session_cached_cursors*), so that the SQL engine caches sufficient number of compiled commands (cursors), so that they stay in the cursor cache between calls to DB(S)_* actions.

Related topics:

DB_DELETE
DB_READ
DB_UPDATE

SQL_PREPARE
SQL_BINDIN
SQL_FETCH
SQL_FREE

All database actions