Data Access

Data access

Current values

Basic current values are sorted out by a value type into the following five tables.

1. AnalogPoints - real values and relative time values.

COLUMN NAME	VALUE TYPE	SIZE
NAME	CHAR	64
DESCRIPT	CHAR	50
VALUE	DOUBLE	8
STATUS	INTEGER	4
WEAK	INTEGER	4
DEFAULT	INTEGER	4
LIMITS	INTEGER	4
FLAGS	INTEGER	4

2. IntegerPoints - integer values.

COLUMN NAME	VALUE TYPE	SIZE
NAME	CHAR	64
DESCRIPT	CHAR	50
VALUE	INTEGER	4
STATUS	INTEGER	4
WEAK	INTEGER	4
DEFAULT	INTEGER	4
LIMITS	INTEGER	4
FLAGS	INTEGER	4

3. EnumPoints - enumerated types including *Boolean* type.

COLUMN NAME	VALUE TYPE	SIZE
NAME	CHAR	64
DESCRIPT	CHAR	50
VALUE	INTEGER	4
STATUS	INTEGER	4
WEAK	INTEGER	4
DEFAULT	INTEGER	4
LIMITS	INTEGER	4
FLAGS	INTEGER	4

4. TimePoints - absolute time values.

COLUMN NAME	VALUE TYPE	SIZE
NAME	CHAR	64
DESCRIPT	CHAR	50
VALUE	TIMESTAMP	
STATUS	INTEGER	4
WEAK	INTEGER	4
DEFAULT	INTEGER	4
LIMITS	INTEGER	4
FLAGS	INTEGER	4

5. TextPoints

COLUMN NAME	VALUE TYPE	SIZE
NAME	CHAR	64
DESCRIPT	CHAR	50
VALUE	CHAR	254
STATUS	INTEGER	4
WEAK	INTEGER	4
DEFAULT	INTEGER	4
LIMITS	INTEGER	4
FLAGS	INTEGER	4

Variables of the types Value array and Structured variable are in separated tables. Table name is derived from a name of the variable so that the dot character "." is substituted by the underscore character "_" in the name.

Each object of the Value array type creates a table of the following format:

COLUMN NAME	VALUE TYPE	SIZE
TIME	TIMESTAMP	
VALUE	DOUBLE	8
STATUS	INTEGER	4
ROW_ID	INTEGER	4
WEAK	INTEGER	4
DEFAULT	INTEGER	4
LIMITS	INTEGER	4
FLAGS	INTEGER	4

Each object of Structured variable type creates a table of the format corresponding to a form of the structure variable. In addition, ROW_ID column, determining an order number - structured variable row, is added to this table.

Historical data

The variables of Historical value type create separated tables. Table name is derived from the name of the historical variable so that the dot character "." is substituted by the underscore character "_" in the name.

Each variable of Historical value type creates a table of the following format:

COLUMN NAME	VALUE TYPE	SIZE
TIME	TIMESTAMP	
VALUE	DOUBLE	8
STATUS	INTEGER	4
STEP	INTEGER	4
WEAK	INTEGER	4

DEFAULT	INTEGER	4
LIMITS	INTEGER	4
FLAGS	INTEGER	4
ARCH_FLAGS	INTEGER	4

There is implemented the table **ArchivValues** which enables to get values of several archive objects with the same time. By a selection from this table, it is possible to enter several names of archive objects within one command.

For example:

SELECT TIME,VALUE01,STATUS01,VALUE02,STATUS02 FROM ArchivValues WHERE TIME >{ts '2000-04-11 10:00:00'} and TIME <{ts '2000-04-11 12:00:00'} and NAME01="H.AAA1" and NAME02="H.AAA2" and STEP=60

The structure of the table **ArchivValues**:

COLUMN NAME	VALUE TYPE	SIZE
TIME	TIMESTAMP	
STEP	INTEGER	4
NAME01	CHAR	64
VALUE01	DOUBLE	8
STATUS01	INTEGER	4
WEAK01	INTEGER	4
DEFAULT01	INTEGER	4
LIMITS01	INTEGER	4
FLAGS01	INTEGER	4
ARCH_FLAGS01	INTEGER	4
NAME02	CHAR	64
VALUE02	DOUBLE	8
STATUS02	INTEGER	4
WEAK02	INTEGER	4
DEFAULT02	INTEGER	4
LIMITS02	INTEGER	4
FLAGS02	INTEGER	4
ARCH_FLAGS02	INTEGER	4
...		
NAMEn	CHAR	64
VALUEn	DOUBLE	8
STATUSn	INTEGER	4
WEAKn	INTEGER	4
DEFAULTn	INTEGER	4
LIMITSn	INTEGER	4
FLAGSn	INTEGER	4
ARCH_FLAGSn	INTEGER	4

Where *n* is at most 12.

ESL RPC procedure call

D2000 ODBC Driver enables also ESL RPC procedure call which corresponds to CALL action. There are created the special tables, which names are derived from D2000 EventHandler name, on which the procedure is to be done, the object of event type and RPC procedure name. They are joined by the character "^" (e.g. SELF_EVH^E_event^proc). The dot characters "." in the name are replaced by an underscore character "_". The tables without Event handler process name in their name (e.g. E_event^proc) are intended for a broadcast RPC procedure call.
The columns are named according to the procedure parameters. RECORD type parameters are represented by columns which are composed by the parameter name and structure column name (parameter^column). These tables also contain "ROW_ID" column. The dot character "." is substituted by the underscore character "_" in the parameter name. The RPC procedure tables contain the columns "$async" and "$instanceId".

The procedure call types:

- synchronous - the column "$async" may not be used in SELECT part,
- asynchronous - the only column "$async" may be used in SELECT part,
- broadcast - RPC table without Event handler name may be used.

When the procedure call is addressed directly to some instance, WHERE part have to contain the column $instanceId".

A general syntax of RPC procedure call:

```
SELECT $async | param1, param2, param3, ...
        FROM [proces_EVH^]E_event^proc
        [WHERE param3 = x AND param4 = y AND ... [$instanceId = id]]
```

Examples of RPC procedure call via ESL and their ODBC equivalent:

synchronous call:	CALL E.event MyProc(_param1, 2) SYNC ON SELF.EVH SELECT _param1 FROM SELF_EVH^E_event^MyProc WHERE _param2 = 2
asynchronous call:	CALL E.event MyProc(1, 2) ASYNC ON SELF.EVH SELECT $async FROM SELF_EVH^E_event^MyProc WHERE _param1 =1 AND _param2 = 2

WHERE part may contain only a value assignment to the parameters.
SELECT output will contain as many rows as RECORD parameter with the most rows or 1 row, if the procedure has only the unstructured types of parameters. If some exception occurs while the procedure is done, SQL query runtime will fail. The result of the asynchronous procedure call is unknown, therefore this query does not return any data.

Limitations

The limits for the command **SELECT**:

1. A SELECT command can contain just one table.
2. Sorting is enabled only according to value of one column (ORDER BY).
3. Selecting an historical value must contain a time interval, e.g.

SELECT TIME,VALUE,STATUS FROM H_NAME WHERE (TIME >"10-04-2000" AND TIME <"12-04-2000 10:00").

It can contain a definition of required time step in seconds, e.g.

SELECT TIME,VALUE,STATUS FROM H_NAME WHERE (TIME >"10-04-2000" AND TIME <"12-04-2000 10:00") AND STEP=60.

The syntax for entering an absolute time is either "dd-mm-rrrr [hh[:mi[:ss[.mss]]]]" or {ts '2000-04-11 10:00:00'}.

The limits for RPC procedure call:

1. RPC procedures:
    - without any parameters,
    - containing ALIAS parameters,
    - containing IN parameters of RECORD type

    cannot be called.
2. The column $instanceId of RPC tables can be only in WHERE part of the SQL query.
3. The column $async of RPC tables can be only in SELECT part of the SQL query and cannot be combined with other column.
4. The columns of RPC tables, representing the columns of RECORD parameters, cannot be used in WHERE part of the SQL query.
5. The columns of RPC tables, representing IN parameters, cannot be used in SELECT part of the SQL query.
6. Neither ORDER BY clause nor the aggregate functions (count, avg, max, ...) are supported while RPC procedure is called.

Description of columns with status information

STATUS	- value is valid if STATUS = 1
WEAK	- value is Weak if WEAK = 1
DEFAULT	- value is in Default status if DEFAULT = 1
LIMITS	- the column can contain the following values: InLimit = 0 , VL_Limit = 1 , L_Limit = 2 , H_Limit = 3 , VH_Limit = 4 , LimitsProblem = 5

FLAGS	- the column gets values of Extended flags (in bitsets): BF_A = 1, BF_B = 2, BF_C = 4, BF_D = 8, BF_E = 16, BF_F = 32, BF_G = 64, BF_H = 128, BF_I = 256, BF_J = 512, BF_K = 1024, BF_L = 2048, BF_M = 4096, BF_N = 8192, BF_O = 16384, BF_P = 32768
ARCH_FLAGS	- the column gets archive flags (as a sum of these constants).

Related pages:

D2000 ODBC Driver
DSN configuration (Data Source Name)