CALCSTATFUNC

CALCSTATFUNC action

Function

Evaluation of a statistical archive function on demand.

Declaration

```
 CALCSTATFUNC archIdent, timeFromIdent_TmA, timeToIdent_TmA,
statFuncIdent_Int, validPercIdent_Int, paramIdent, retValueIdent,
statusIdent_Int [,archivInstance_Int]
```

Parameters

archIdent	in	Reference to one value of historical value or reference to object, values of which have been archived. **Warning:** If the parameter is the reference to an object archived several times, there is not specified which one of the historical objects is to be used.
timeFromItemIdent_TmA	in	Identifier of *AbsTime* type - interval beginning.
timeToItemIdent_TmA	in	Identifier of *AbsTime* type - interval end.
statFunctIdent	in	Identifier of *Int* type - statistical function type.
validPercIdent	in	Identifier of *Int* type - validation criteria.
paramIdent	in	Identifier of *Int* or *Real* types - parameter for some types of functions.
retValueIdent	out	Identifier for calculation result.
statusIdent	out	Calculation (action) success.
archivInstance_Int	in	Optional identifier of *Int* type - identification of archive instance. If the parameter is not defined, the value 0 will replace it.

Description

The action executes calculation of the statistical function given by the parameter *statFuncIdent_Int* on the values represented by the historical value *archIdent* within the time interval given by the parameter *timeFromIdent_TmA* and *timeToIdent_TmA* . After successful execution of the action, the parameter *retValueIdent* contains a result value of the calculation. The parameter *statusIdent_Int*gets one of the following values:

- _ERR_TRANS_ABORT
- _ERR_TRANS_ERROR
- _ERR_TRANS_IGNORED
- _ERR_NO_ERROR

Set of implemented functions is equal with the functions, which may be configured for statistical archive (except the *FILTER* and *EACOAVG* functions). For the individual functions, there are predefined local constants in ESL.

The parameter *validPerctIdent_Int* represents Validation criteria used during the evaluation of the statistical function. It can get values within 1 .. 100 (the error ERR_RANGE_ERROR occurs if the condition is broken).

For some statistical function, it is required to enter the parameter *paramIdent* according to the table:

Statistical function	Parameter description

_STAT_F_INTEGRAL	Integral time units. For individual types, there are also established predefined local variables according to the table: **Constant** / **Description** _INTEGRAL_HOURUNIT — Hour integral _INTEGRAL_MINUNIT — Minute integral _INTEGRAL_SECUNIT — Second integral
_STAT_F_GE_TIME _STAT_F_GT_TIME _STAT_F_LE_TIME _STAT_F_LT_TIME _STAT_F_ADDITION_PARAM _STAT_F_INCREMENT_PARAM _STAT_F_DELTA_PARAM	Compare value

For the other statistical archive functions, the parameter *paramIdent* is not being evaluated.

Value of parameter *archivInstance_Int* defines the instance of archive which executes the request. If the parameter is not defined (or the value is 0), the active instance of archive will execute the request.

Note: When calculating **CALCSTATFUNC** action, a time limitation which is active in the computations of statistical archives and does not allow for calculation for future times, is no applied. With **CALCSTATFUNC** , it is therefore possible, for example, to calculate statistics over a script filled archive that has data in the future.

Example

```
 TIME _bt
 TIME _et
 REAL _retValue
 INT _retCode


 _bt := %StrToTime("8:01:00 16-10-2003")
 _et := %StrToTime("8:02:00 16-10-2003")

 CALCSTATFUNC H.ArchObj, _bt, _et,  _STAT_F_SUM, 100, 0, _retValue,
_retCode

 IF _retCode = _ERR_NO_ERROR THEN
 ; calculation done
 ELSE
   ; an error occurred
 ENDIF
```

Note 1

Using the action **CALCSTATFUNC** does not allow to evaluate the *FILTER* and *ECOAVG* statistical functions.

Note 2

If it is necessary to calculate statistics over a periodic archive (e.g. *H.Period*), it should be noted that the periodic archive has values only in multiples of the period. For example, if it were necessary to calculate the integral from the periodic archive and the beginning of the interval would not be multiples of the period, 0 would be integrated from the beginning of the interval to the first multiple of the period.

This feature can be circumvented by creating a on-change computed archive object (e.g. *H. PeriodOnRead*) that is computed on-read and that copies *H.Period*. Integral computed from *H. PeriodOnRead* archive will behave as expected.

Related pages:

Script actions