GETARCHARR

GETARCHARR action

Function

Reading a value block of specified historical values within a given time interval.

Declaration

```
 GETARCHARR archIdent, locVarColValueIdent_Rec, [locVarColFlagsIdent_Rec],
timeFromIdent_TmA, timeToIdent_TmA, stepIdent_Int, maxValsIdent_Int,
statusIdent_Int[, archivInstance_Int]
```

Parameters

archIdent	in	Reference to one value of historical value, reference to object or item identifier of Structured variable type object (*note:* values of object or item must be 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.
locVarColValueIdent_Rec	out	Reference to column of a RECORD type local variable for result values
locVarColFlagsIdent_Rec	out	Optional parameter. Reference to column of a *RECORD* type local variable for archive flags (*Integer* type).
timeFromItemIdent_TmA	in	Identifier *AbsTime* type for the interval beginning.
timeToItemIdent_TmA	in	Identifier of *AbsTime* type for the interval end.
stepIdent	in	Identifier of *Int* type - time step for the resampling of values in the archive. For resampling details, see Resampling note.
maxValsIdent_Int	in	Maximal number of values. If more values is in the required interval, they will be trimmed and the action returns the warning _ERR_MORE_DATA in the identifier *statusIdent_Int.*
statusIdent_Int	out	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 reads values of the historical value *archIdent* within the time interval from `timeFromIdent_TmA` to `timeToIdent_TmA` with the steps `stepIdent_Int` (given in seconds). Maximal number of values is given by the identifier `maxValsIdent_Int`. See reading the values.

The parameter *stepIdent_Int* defines the resampling (in seconds) of read values. If it is equal to 0, reading is not to be resampled.
For resampling details, see Resampling note.

If the parameter *archIdent* is the reference to an object of Historical value type, the action performance is described above. If the parameter is the reference to an object (not of Historical value type) or a structured variable item that is not of ***Object*** type, the system is attempting to find an object of Historical value type that archives values of the object (item).
If the parameter *archIdent* is the reference to a structured variable item that is of ***Object*** type, the item "points" to an object in the system. If the object is of Historical value type, the action will read data from it. If it is not, the system is attempting to find an object of Historical value type that archives values of the object.

The return code `statusIdent_Int` can get one of the following values:

- _ERR_TRANS_ABORT
- _ERR_TRANS_ERROR
- _ERR_TRANS_IGNORED
- _ERR_NO_ERROR
- _ERR_NO_DATA - no data within given interval
- _ERR_MORE_DATA - more data than `maxValsIdent_Int` within given interval
- *_ERR_OBJECT_IS_NOT_IN_ARCHIVE*

The error _ERR_MORE_DATA has only informative character and the required number of data is available. If the value of the identifier `stepIdent_Int` = 0, then values from the given interval are not be oversampled.

`locVarColValueIdent``_Rec` - is the reference to an item of a RECORD type local variable. The action, after successful reading of values, resizes the array (internally the action REDIM) to the required number of rows and in sequence (from 1...) assign a value from the archive to the given item. Likewise *`locVarColFlagsIdent`*`_Rec` - is the reference to an item in a local variable of RECORD type. The item must be *Int* type. Analogous to the previous parameter, the action will resize the array (the sizes will be equal) and assign flags from the archive to the given item in each row (see the action GETARCHVAL). The parameter is optional and it can be omitted. An item for data from the archive and an item for archive flags may be from the same local variable.

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.

Example

The example assumes the existence of the object `SD.ArchDemo` of *Structure definition* type, that contains the following items:

Structure definition items

Item name	Item type
Text	Text
Value	Real number
Flags	Integer

Reading values from the archive. Values and archive flags are in two various arrays.

```
 TIME _timeFrom
 TIME _timeTo
 INT _maxVals
 INT _step
 INT _status
```

```
 RECORD (SD.ArchDemo) _locAValArr
 RECORD (SD.ArchDemo) _locAFlagsArr
```

```
_timeTo := %StrToTime("10:00:00 31-12-1999")
 _timeFrom := %StrToTime("09:00:00 31-12-1999")
 _maxVals := 100
 _step := 0
 GETARCHARR H.ArchObj, _locAValArr^Value, _locAFlagsArr ^Flags, _timeFrom,
_timeTo, _step, _maxVals, _status
```

```
IF (_status = _ERR_NO_ERROR) | (_status = _ERR_MORE_DATA) THEN
   ; data are loaded
 ELSE
   ; an error occurred
 ENDIF
```

Seeing that the number of values and archive flags is always equal, the previous example should be implemented with the reading of values and flags into one array.

```
 TIME _timeFrom
 TIME _timeTo
 INT _maxVals
 INT _step
 INT _status
```

```
 RECORD (SD.ArchDemo) _locAValArr
```

```
_timeTo := %StrToTime("10:00:00 31-12-1999")
 _timeFrom := %StrToTime("09:00:00 31-12-1999")
 _maxVals := 100
 _step := 0
```

```
GETARCHARR H.ArchObj, _locAValArr^Value, _locAValArr^Flags, _timeFrom,
_timeTo, _step, _maxVals, _status
```

```
IF (_status = _ERR_NO_ERROR) | (_status = _ERR_MORE_DATA) THEN
   ; data are read
 ELSE
   ; an error occurred
 ENDIF
```

Reading data from structured historical value.

```
 TIME _timeFrom
 TIME _timeTo
 INT _maxVals
 INT _step
 INT _status
 INT _row

 RECORD (SD.ArchDemo) _locAValArr

 _timeTo := %StrToTime("10:00:00 31-12-1999")
 _timeFrom := %StrToTime("09:00:00 31-12-1999")
 _maxVals := 100
 _step := 0
 _row := 4   ; row 4

 GETARCHARR (H.StructArchObj\HBJ, _row, 5) _locAValArr^Value,
_locAValArr^Value, _timeFrom, _timeTo, _step, _maxVals, _status
 IF (_status = _ERR_NO_ERROR) | (_status = _ERR_MORE_DATA) THEN
   ; data are read
 ELSE
   ; an error occurred
 ENDIF
```

Resampling note

If the value of the *`stepIdent_Int`* parameter is nonzero, the read data is resampled with the specified period (in seconds). Resampling can also be with a period greater than 1 day (86400 seconds), e.g. 3 or 7 days.

For periodic archives (primary periodic, calculated periodic, statistical, script-filled periodic), resampling takes into account the transition between summer and winter time depending on the "*Use monotonic time*" setting in the configuration of archive being read. If e.g. resampling has a period of 86400 seconds and an initial time of "10:00:00 29-10-2021", so the time stamps for Central European (CET) time (transition from summer time to winter time is 31-10-2021 at 02:00) will be as follows :

Use monotonic time	Use local time
10:00:00 29-10-2021	10:00:00 29-10-2021
10:00:00 30-10-2021	10:00:00 30-10-2021
09:00:00 31-10-2021	10:00:00 31-10-2021
09:00:00 01-11-2021	10:00:00 01-11-2021

When using monotonic time, the resampling shifted the time from 10:00 to 09:00, as another hour was inserted at 31-10-2021 at 02:00 (so the day 31-10-2021 had 25 * 3600 seconds).
When using local time, the offset within the day is preserved (10:00).

For non-periodic archives (primary on-change, calculated on-change, script-filled on-change), resampling always takes into account the transition between summer and winter time (i.e. the offset within the day is preserved). Should a "monotonic" resampling be required, it is possible to configure a statistics archive (*Time Slice* statistical function) that is calculated *On Read* (so it does not burden the archive database) and has a "*Use monotonic time*" set. Subsequently, it is necessary to read this *Time Slice* statistics.

It is also possible to configure the *Time Slice* statistics for periodic archives, if it is necessary to perform resampling with the inverse setting of Monotonic/Local time as is in the configuration of the periodic archive.

Related pages:

Script actions
GETARCHARR_TO_CNT action