
AIMMS Function Reference - Dataset Functions

This file contains only one chapter of the book. For a free download of the complete book in pdf format, please visit www.aimms.com

Dataset Functions

AIMMS supports the following functions for accessing the datasets in the **Data Manager**:

- DatasetCreate
- DatasetDelete
- DatasetFind
- DatasetGetCategory
- DatasetGetChangedStatus
- DatasetLoadCurrent
- DatasetLoadIntoCurrent
- DatasetMerge
- DatasetNew
- DatasetSave
- DatasetSaveAll
- DatasetSaveAs
- DatasetSelect
- DatasetSelectNew
- DatasetSetChangedStatus
- DatasetSetCurrent

DatasetCreate

The procedure `DatasetCreate` creates a new dataset node in the Data Management tree. The data category, the name of the dataset and the folder in which it is created is given as an argument to the procedure.

```
DatasetCreate(  
    data_category,    ! (input) element in AllDataCategories  
    dataset_path,    ! (input) scalar string expression  
    dataset          ! (output) element parameter into AllDataSets  
)
```

Arguments:

data_category

An element in `AllDataCategories`, specifying the data category for which a dataset must be created.

dataset_path

A string expression holding the path and name of the new dataset. The path is specified relative to the corresponding data category root node in the Data Management tree.

dataset

An element parameter into `AllDataSets`. On successful return this parameter will refer to the newly created element in `AllDataSets`.

Return value:

The procedure returns 1 if the dataset is created successfully. It returns 0 if the dataset could not be created or if the dataset already exists.

Remarks:

- This function is only applicable if the project option Data Management style is set to Single Data Manager file.
- If the specified path contains folders that do not exist, then these folders are created automatically. To check whether a specific dataset path already exists you can use the procedure `DatasetFind`.

See also:

The procedures [DatasetFind](#), [DatasetDelete](#).

DatasetDelete

The procedure `DatasetDelete` deletes a specific dataset node from the Data Management tree.

```
DatasetDelete(  
    data_category, ! (input) element in AllDataCategories  
    dataset       ! (input) element parameter into AllDataSets  
)
```

Arguments:

data_category

An element in `AllDataCategories`, specifying the data category for which a dataset is to be deleted.

dataset

An element parameter into `AllDataSets`, representing the dataset that you want to delete.

Return value:

The procedure returns 1 if the dataset is deleted successfully, or 0 otherwise.

Remarks:

- This function is only applicable if the project option Data Management style is set to Single Data Manager file.

See also:

The procedure [DatasetFind](#).

DatasetFind

The procedure `DatasetFind` searches the Data Management tree for a dataset with a specific name and belonging to a specific data category.

```
DatasetFind(  
    data_category,    ! (input) element in AllDataCategories  
    dataset_path,    ! (input) scalar string expression  
    dataset          ! (output) element parameter into AllDataSets  
)
```

Arguments:

data_category

An element in `AllDataCategories`, specifying the data category for which the datasets must be searched.

dataset_path

A string expression holding the path and name of a dataset. The path is specified relative to the corresponding data category root node in the Data Management tree.

dataset

An element parameter into `AllDataSets`. On successful return this parameter will refer to the dataset found.

Return value:

The procedure returns 1 if the dataset is found, and 0 otherwise.

Remarks:

- This function is only applicable if the project option `Data Management style` is set to `Single Data Manager file`.

See also:

The procedures [DatasetCreate](#), [DatasetDelete](#).

DatasetGetCategory

The procedure `DatasetGetCategory` retrieves the data category of a specific dataset.

```
DatasetGetCategory(  
    dataset,          ! (input) element of the set AllDataSets  
    data_category    ! (output) element parameter into AllDataCategories  
)
```

Arguments:

dataset

An element of the set `AllDataSets`, referring to the dataset for which you want to retrieve its data category.

data_category

An element parameter into `AllDataCategories`, on successful return this argument will contain the data category of the given dataset.

Return value:

The procedure returns 1 on success, 0 otherwise.

Remarks:

- This function is only applicable if the project option `Data Management style` is set to `Single Data Manager file`.

DatasetGetChangedStatus

The function `DatasetGetChangedStatus` returns whether the data associated with a specific data category has changed and thus needs to be saved.

```
DatasetGetChangedStatus(  
    data_category      ! (input) element in AllDataCategories  
)
```

Arguments:

data_category

An element in `AllDataCategories`, specifying the data category for which the changed status must be retrieved.

Return value:

The function returns 1 if the data has changed, 0 otherwise.

Remarks:

- This function is only applicable if the project option Data Management style is set to Single Data Manager file.

See also:

The functions [DatasetSetChangedStatus](#), [DatasetSave](#).

DatasetLoadCurrent

The procedure `DatasetLoadCurrent` loads an existing dataset as the new current dataset for a specific data category. You can use it to load either a dataset that is passed as argument to the procedure, or a dataset that the user can select via a dialog box. If the data of the corresponding data category has changed, then the user is asked to save this data first.

```
DatasetLoadCurrent(
    data_category, ! (input) element in AllDataCategories
    dataset,       ! (input/output) an element parameter into AllDataSets
    [dialog]      ! (optional) 0 or 1
)
```

Arguments:

data_category

An element in `AllDataCategories`, specifying the data category for which a dataset is loaded.

dataset

An element parameter in the set `AllDataSets`. If the argument *dialog* is set to 0, then no dialog box is shown and the dataset to which the element parameter currently refers is loaded. If the argument *dialog* is set to 1, then the value of the element parameter is used to initialize the dialog box. The dataset that the user has selected and is loaded successfully is returned through this argument.

dialog (optional)

An integer value indicating whether or not the user gets a dialog box in which he can select the dataset to load. The default value is 1, thus if this argument is omitted the dialog box will be shown.

Return value:

The procedure returns 1 on success. If the user canceled the operation, then the procedure returns 0. If any other error occurs then the procedure returns -1 and `CurrentErrorMessage` will contain a proper error message.

Remarks:

- This function is only applicable if the project option Data Management style is set to Single Data Manager file.
- If you want to suppress the dialog box for the unsaved data, then you may call `DatasetSetChangedStatus(category,0)` prior to `DatasetLoadCurrent`.

See also:

The procedures [DatasetLoadIntoCurrent](#), [DatasetMerge](#), [DatasetSave](#), [DatasetSetChangedStatus](#).

DatasetLoadIntoCurrent

The procedure `DatasetLoadIntoCurrent` loads the data of an existing dataset as the new current dataset for a specific data category. You can use it to load either a dataset that is passed as argument to the procedure, or a dataset that the user can select via a dialog box. The data that is stored in the dataset will overwrite any data of the currently active dataset, and thus this current dataset is set to have changed data.

```
DatasetLoadIntoCurrent(  
    data_category, ! (input) element in AllDataCategories  
    dataset,       ! (input/output) an element parameter  
                  ! into AllDataSets  
    [dialog]       ! (optional) 0 or 1  
)
```

Arguments:

category

An element in `AllDataCategories`, specifying the data category for which a dataset is loaded.

dataset

An element parameter in the set `AllDataSets`. If the argument *dialog* is set to 0, then no dialog box is shown and the dataset to which the element parameter currently refers is loaded. If the argument *dialog* is set to 1, then the value of the element parameter is used to initialize the dialog box. The dataset that the user has selected and is loaded successfully is returned through this argument.

dialog (optional)

An integer value indicating whether or not the user gets a dialog box in which he can select the dataset to load. The default value is 1, thus if this argument is omitted the dialog box will be shown.

Return value:

The procedure returns 1 on success. If the user canceled the operation, then the procedure returns 0. If any other error occurs then the procedure returns -1 and `CurrentErrorMessage` will contain a proper error message.

Remarks:

- This function is only applicable if the project option Data Management style is set to Single Data Manager file.

See also:

The procedures [DatasetLoadCurrent](#), [DatasetMerge](#), [DatasetSave](#), [DatasetSetChangedStatus](#).

DatasetMerge

The procedure `DatasetMerge` merges the data of an existing dataset with the current data. You can use it to merge either a dataset that is passed as argument to the procedure, or a dataset that the user can select via a dialog box. Only the non-default data that is stored in the dataset will be merged with the current data.

```
DatasetMerge(  
    data_category, ! (input) element in AllDataCategories  
    dataset,       ! (input/output) an element parameter into AllDataSets  
    [dialog]      ! (optional) 0 or 1  
)
```

Arguments:

data_category

An element in `AllDataCategories`, specifying the data category for which a dataset is loaded.

dataset

An element parameter in the set `AllDataSets`. If the argument *dialog* is set to 0, then no dialog box is shown and the dataset to which the element parameter currently refers is loaded. If the argument *dialog* is set to 1, then the value of the element parameter is used to initialize the dialog box. The dataset that the user has selected and is loaded successfully is returned through this argument.

dialog (optional)

An integer value indicating whether or not the user gets a dialog box in which he can select the dataset to load. The default value is 1, thus if this argument is omitted the dialog box will be shown.

Return value:

The procedure returns 1 on success. If the user cancelled the operation, then the procedure returns 0. If any other error occurs then the procedure returns -1 and `CurrentErrorMessage` will contain a proper error message.

Remarks:

- This function is only applicable if the project option `Data Management style` is set to `Single Data Manager file`.

See also:

The procedures [DatasetLoadCurrent](#), [DatasetLoadIntoCurrent](#), [DatasetSave](#), [DatasetGetChangedStatus](#).

DatasetNew

The procedure `DatasetNew` starts a new unnamed dataset for a specific data category. The procedure is similar to the command **Dataset New** from the **Data** menu. The procedure does not change any of the current data, it only sets the current dataset to unnamed. If you did have a currently named dataset and the data of this dataset has been changed, then AIMMS will ask whether or not this dataset should be saved first.

```
DatasetNew(
    data_category      ! (input) an element of AllDataCategories
)
```

Arguments:

data_category

An element in `AllDataCategories`, specifying the data category for which you want to start a new unnamed dataset.

Return value:

The procedure returns 1 on success. If the user cancelled the operation, then the procedure returns 0. If any other error occurs then the procedure returns -1 and `CurrentErrorMessage` will contain a proper error message.

Remarks:

- This function is only applicable if the project option Data Management style is set to Single Data Manager file.
- If you use `CaseNew`, then the name of this new case is not specified until you save the case. If you want to start a new named case, then you can use the following piece of code:

```
if ( CaseGetChangedStatus ) then
    if ( CaseSave = 0 ) then
        return ;
    endif ;
endif ;
if ( CaseSelectNew( a_case ) ) then
    CaseSetCurrent( a_case );
    CaseSetChangedStatus( a_case, 1 );
endif ;
```

See also:

The procedures [DatasetLoadCurrent](#), [DatasetSave](#), [DatasetSelectNew](#), [DatasetSetCurrent](#).

DatasetSave

The procedure `DatasetSave` saves the data of a data category to the active dataset. If there is no named active dataset, then the procedure behaves exactly as the `DatasetSaveAs` procedure.

```
DatasetSave(  
    data_category, ! (input) element in AllDataCategories  
    [confirm]      ! (optional) 0, 1 or 2  
)
```

Arguments:

data_category

An element in `AllDataCategories`, specifying the data category for which you want to save the data.

confirm (optional)

An integer to indicate whether or not the procedure should ask for confirmation before overwriting the existing dataset. If 0, then no confirmation dialog box is shown. If 1 (default), then whether or not the confirmation dialog box is shown depends on the case type properties. If 2, then always a confirmation dialog box is shown.

Return value:

The procedure returns 1 if the dataset is saved successfully. It returns 0 if the user canceled the save operation. If any other error occurs, then the procedure returns -1 and `CurrentErrorMessage` will contain an error message.

Remarks:

- This function is only applicable if the project option `Data Management style` is set to `Single Data Manager file`.

See also:

The procedures `DatasetSaveAs`, `DatasetSaveAll`, `DatasetLoadCurrent` and function `DatasetGetChangedStatus`.

DatasetSaveAll

The procedure `DatasetSaveAll` saves the data of all data category to the active datasets. If there are no named active datasets, then the procedure behaves according to the `DatasetSaveAs` procedure.

```
DatasetSaveAll(  
    [confirm]      ! (optional) 0, 1 or 2  
)
```

Arguments:

confirm (optional)

An integer to indicate whether or not the procedure should ask for confirmation before overwriting the existing datasets. If 0, then no confirmation dialog box is shown. If 1 (default), then whether or not the confirmation dialog box is shown depends on the case type properties. If 2, then always a confirmation dialog box is shown.

Return value:

The procedure returns 1 if the datasets are saved successfully. It returns 0 if the user canceled the save operation. If any other error occurs, then the procedure returns -1 and `CurrentErrorMessage` will contain an error message.

Remarks:

- This function is only applicable if the project option `Data Management style` is set to `Single Data Manager file`.

See also:

The procedures [DatasetSaveAs](#), [DatasetSave](#), [DatasetLoadCurrent](#), [DatasetGetChangedStatus](#).

DatasetSaveAs

The procedure `DatasetSaveAs` shows a dialog box in which the user can specify a (new) dataset to which the data is saved.

```
DatasetSaveAs(  
    data_category, ! (input) element in AllDataCategories  
    dataset        ! (output) element parameter in AllDataSets  
)
```

Arguments:

data_category

An element in `AllDataCategories`, specifying the data category for which you want to save the data.

dataset

An element parameter in `AllDataSets`. On return this parameter will refer to the dataset that the user selected.

Return value:

The procedure returns 1 if the dataset is saved successfully. It returns 0 if the user cancelled the save operation. If any other error occurs, then the procedure returns -1 and `CurrentErrorMessage` will contain an error message.

Remarks:

- This function is only applicable if the project option Data Management style is set to Single Data Manager file.

See also:

The procedures [DatasetSave](#), [DatasetSaveAll](#), [DatasetLoadCurrent](#), [DatasetGetChangedStatus](#).

DatasetSelect

The procedure `DatasetSelect` shows a dialog box in which the user can select an existing dataset for a given data category.

```
DatasetSelect(  
    data_category, ! (input) element in AllDataCategories  
    dataset,      ! (output) element parameter in AllDataSets  
    [title]      ! (optional) string expression  
)
```

Arguments:

data_category

An element in `AllDataCategories`, specifying the data category for which you want to the user to select a dataset.

dataset

An element parameter in `AllDataSets`. On return the dataset will refer to the selected dataset.

title (optional)

A string expression that is used as the title for the dialog box. If this argument is omitted, then a default title is used.

Return value:

The procedure returns 1 if the user did select a dataset. If the user pressed **Cancel**, then the procedure returns 0.

Remarks:

- This function is only applicable if the project option `Data Management style` is set to `Single Data Manager file`.

See also:

The procedure [DatasetSelectNew](#).

DatasetSelectNew

The procedure `DatasetSelectNew` shows a dialog box in which the user can select a new dataset for a given data category.

```
DatasetSelectNew(  
    data_category, ! (input) element in AllDataCategories  
    dataset,       ! (output) element parameter in AllDataSets  
    [title]        ! (optional) string expression  
)
```

Arguments:

data_category

An element in `AllDataCategories`, specifying the data category for which you want to the user to select a new dataset.

dataset

An element parameter in `AllDataSets`. On return the dataset will refer to the selected dataset.

title (optional)

A string expression that is used as the title for the dialog box. If this argument is omitted, then a default title is used.

Return value:

The procedure returns 1 if the user did select a dataset. If the user pressed **Cancel**, then the procedure returns 0.

Remarks:

- This function is only applicable if the project option Data Management style is set to Single Data Manager file.
- If via this procedure the user creates a new dataset (i.e. a new dataset node in the data management tree), then this case dataset does not yet contain any data. The dataset will only contain data after you explicitly save data to it.

See also:

The procedures [DatasetSelect](#), [DatasetSetCurrent](#), [DatasetSave](#).

DatasetSetChangedStatus

The procedure `DatasetSetChangedStatus` can set the status of a data category to either changed or unchanged.

```
DatasetSetChangedStatus(  
    data_category,    ! (input) element in AllDataCategories  
    status            ! (input) 0 or 1  
)
```

Arguments:

data_category

An element in `AllDataCategories`, specifying the data category for which you want to set the changed status.

status

An integer value holding the new dataset status: 0 for unchanged, 1 for changed.

Return value:

The procedure returns 1.

Remarks:

- This function is only applicable if the project option Data Management style is set to Single Data Manager file.

See also:

The function [DatasetGetChangedStatus](#).

DatasetSetCurrent

The procedure `DatasetSetCurrent` sets the dataset that is regarded as the current dataset for a given data category. It does not load or save any data or checks whether data needs to be saved. You can, for example, use it to make a newly created dataset the current dataset, so that during a `DatasetSave` the data is written to this dataset.

```
DatasetSetCurrent(  
    data_category,    ! (input) element in AllDataCategories  
    dataset           ! (input) element of the set AllDataSets  
)
```

Arguments:

data_category

An element in `AllDataCategories`, specifying the data category for which you want to set the current dataset.

dataset

An element of the set `AllDataSets`, referring to the dataset that should become the current dataset.

Return value:

The procedure returns 1 on success, or 0 otherwise.

Remarks:

- This function is only applicable if the project option `Data Management style` is set to `Single Data Manager file`.

See also:

The procedures [DatasetNew](#), [DatasetCreate](#), [DatasetSelectNew](#), [DatasetSave](#).