Peregrine

AssetCenter
Database Administrator

Peregrine SYSTEMS

PART NUMBER ACT-4.0.0-ENG-01021-00064

This edition applies to version 4.0.0 of the licensed program

AssetCenter

Peregrine Systems, Inc.
Worldwide Corporate Campus and Executive Briefing Center
3611 Valley Centre Drive San Diego, CA 92130
Tel 800.638.5231 or 858.481.5000
Fax 858.481.1751
www.peregrine.com

Table of Contents

List of Figures

List of Tables

1 Introduction

CHAPTER

AssetCenter Database Administrator is an administration tool for the AssetCenter databases. It enables you to perform a large number of operations, for example the:

- Creation of a database.
- Customization of objects stored in the database (tables, fields, links, indexes, screens, etc;).
- Repair a damaged database.
- Update the structure of a database.
- Extract information contained in a database.

This tool is aimed at the administration and requires a certain number of precautions:

- Since this tools enables you to modify the structure of an AssetCenter database, we recommend limiting its use to users with the appropriate skills.
- AssetCenter also enables you, when you use the "Admin" login, to modify the database (configuration of objects, links, etc.). However,

you should never simultaneously modify the same database using AssetCenter and AssetCenter Database Administrator.

- AssetCenter Database Administrator requires a connection with the "Admin" login (Administrator) or a login with administrative rights to the database in question. We advise against giving more than one person this login. It can result in conflicts when updating the database or compromise the integrity of the structure of the database thus rendering it unusable.

2 Graphical interface

CHAPTER

AssetCenter Database Administrator is executed from the same program group as AssetCenter. Click its icon to launch it.

Note: At startup, the main AssetCenter Database Administrator screen appears entirely in gray as long as no file has been loaded. You can automatically load the last document used by setting the At startup, automatically load last document used option to Yes (in the Confirmations section accessible by the Edit/Options menu item).

Presentation of the graphical interface

AssetCenter Database Administrator's user interface is composed of three panes:

- A menu bar, containing a toolbar.

- A pan containing a list of AssetCenter database tables.
- A main pane, also called a Customization pane, which regroups the information about a table's objects.

This graphical interface respects the same ergonomic database rules as AssetCenter, such as for consultation and creation. For more information on these rules, refer to the **Introduction** guide of AssetCenter.

Selecting a table

To select a table, click its title in the left-hand pane of the user interface.

Select the displayed object type.

For any given table, you can select the object type displayed in the main pane using the **Display** menu. The available objects are:

- Fields
- Links
- Indexes
- Details
- Pages

Using the Customization pane

The Customization pane is divided into three parts:

- The first part (on top) displays the general information about the selected table.
- The second part (in the middle) lists all the objects of a given type appearing in the table. You can choose the type of object to appear using the **Display** menu.
- The third part (on the bottom) contains the information about the selected object.

Note: Only certain information can be customized. The non-editable values appear in grayed-out fields.

Manipulating files

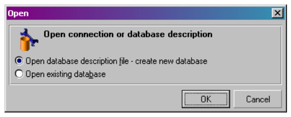
The **File** menu regroups all the functions relating to the loading and the saving of a file.

Opening a file

Select the **File/Open** menu item.

Selecting this menu item displays the following screen:

Figure 2.1. Database opening screen


Open
Open connection or database description
Open database description file - create new database
Open existing database
OK
Cancel

This screen enables you to choose one of two AssetCenter Database Administrator functions:

- Create a new database or modify a database description file by selecting the **Open database description file - create new database** option.
- Customize an existing database by selecting the **Open existing database** option.

You can open an AssetCenter Database Administrator session by selecting one of these two options.

Note: In the last part of the File menu, AssetCenter Database Administrator lists by default the last four opened documents. You can thus quickly open one of these documents by directly clicking it in the menu. To set the number of documents kept in memory, you can use the Maximum number of documents to store in memory option (in the Display section, available via the Edit/Options menu item.

Open database description file - create new database

To create a new database AssetCenter Database Administrator needs a description of that database. This description comes in the form of a file that contains the structural information relating to an AssetCenter database. It plays the role of the data model during the creation of the database.

By validating this choice, you must provide AssetCenter Database Administrator a description file, whose file extension is ".dbb". The "gbbase.dbb" provided with your AssetCenter application is located in the installation folder, and is used exactly for this purpose. We recommend that you make a copy of this file in order to always have in your possession a standard description file.

Note: We also advise you to systematically generate a database description file for your databases (using the File/Save as menu item) and to make a copy of this file: You will need it if ever you want to repair your database.

Open an existing database

You must connect to the AssetCenter database before being able to customize it. Selecting this option will display the connection that is also used for AssetCenter.

We will not list all the possibilities available to edit this connection, accessible via the ☰ button. If you want an exhaustive list of these possibilities, refer to the "Introduction reference guide".

Closing a file

The **File/Close** menu item enables you to close an AssetCenter Database Administrator session. If you made any modifications, AssetCenter Database Administrator asks to if you'd like to save them before closing the session.

To signal that a modification has been made to a database, and before validating these modifications, AssetCenter Database Administrator adds an asterisk to the name of the document in the application's title bar.

Saving a file

Two modes of saving a document are available:

- The **File/Save** menu item enables you to save the modifications made to the database description file or to the database.
- The **File/Save as** menu item enables you to perform the following tasks:

  1. If a database description file is open, this menu item enables you to save the structure of the database in a new database description file.
  2. If a database file is open, this menu item enables you to save the structure of the database in a database description file.

Quit the application

This menu item enables you to quit AssetCenter Database Administrator. If you made any modifications during the session, AssetCenter Database Administrator asks if you want to save them.

Edit functions

AssetCenter Database Administrator proposes all the classic edition functions:

Cut/Copy/Paste

The **Edit** menu regroups all the operations that you can perform using a selection:

- **Cut** (Ctrl+X) to cut a selection.
- **Copy** (Ctrl+C) to copy a selection.
- **Paste** (Ctrl+V) to paste a selection.

Note: These functions only work in editable field zones.

Perform a search

AssetCenter Database Administrator proposes an advanced text searching function, accessible via the **Edit/Find** menu item.

As you have already seen, the database structure is composed of objects (tables, fields, links, etc.), which are characterized by the information relating to their properties. For example, the **SQL name** of a table is a property of that table, just as the **Type** of a field is one of the properties of that field.

The global structure of a database is thus composed of hierarchical objects, each one possessing one or more properties.

The search function proposed by AssetCenter Database Administrator is performed on the integrality of the database structure. When you perform a search, the software looks through all the database objects and their properties. The list of results is then created in the software's memory, and you can browse this list by using the **Edit/Next** or **Edit/Previous** menu items (or their keyboard shortcuts: F3 and Shift+F3, respectively).

Note: You can specify the direction of the search by using the Up or Down options in the dialog box.

Application options

The **Edit/Options** menu item enables you to define your preferences of use for the AssetCenter Database Administrator. Each option is described in the **Description** zone of the dialog box.

CHAPTER 3

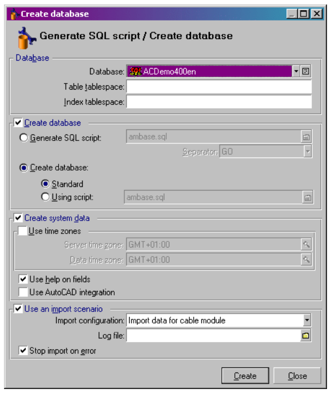
Creating a database

AssetCenter Database Administrator enables you to create an AssetCenter database based on the model of the open database's description file.

Note: To create an AssetCenter database, you need to have already created an empty shell using your DBMS. For more information on this subject, refer to the guide provided with your DBMS.

Selecting the **Action/Create database** menu item triggers the appearance of database creation screen.

Figure 3.1. Creating a database


Create database

Generate SQL script / Create database

Database
- Database: ACDemo400en
- Table tablespace:
- Index tablespace:

[✓] Create database
- ○ Generate SQL script: ambase.sql
- Separator: GO
- ● Create database:
  - ● Standard
  - ○ Using script: ambase.sql

[✓] Create system data
- [ ] Use time zones
  - Server time zone: GMT+01:00
  - Data time zone: GMT+01:00
- [✓] Use help on fields
- [ ] Use AutoCAD integration

[✓] Use an import scenario
- Import configuration: Import data for cable module
- Log file:
- [✓] Stop import on error

Options of the Database frame

These options enable you to select an existing AssetCenter connection and define table spaces for tables and indexes. Table spaces must be created beforehand with your DBMS.

Options of the Create database frame

The options in this frame are only available if the **Create database** option is checked. In this case, you can:

- Directly create the database, if the **Create database** and **Standard** option buttons are selected.
- Generate a SQL script to create the database later on if the **Generate SQL database creation script** option button is selected. Click 🗀 to name the script and define a SQL separator (classic separators include "/" for Oracle, and "GO", for other databases) from the drop down list.

Note: The drop-down list is editable. You can freely define any other separator as long as they are valid (for example ";").

Nothing stops you from defining a separator such as "<MySeparator>", but the database creation script will no longer work.

- Create the database using a SQL database creation script (which you can generate using the **Generate SQL database creation script** option in the same frame) if the **Create database** and **Using script** options are selected. Click to give the name of the SQL script.

Options of the Create system data frame

If you check the **Create system data** check box, AssetCenter Database Administrator creates the following system data:

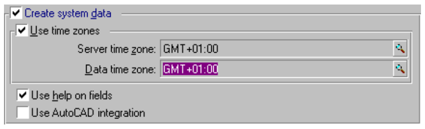
1. Database description file stored in the system table (**System tables** table (SQL name: SysBlob)).
2. Password (empty by default) of the "Admin" login encrypted and stored in the "sysblob" table.
3. "Admin" user.
4. Records with null identifiers (used to simulate outer joins) in each table.
5. Itemized lists and counters.

If you check the **Use time zones** check box, AssetCenter Database Administrator creates the following system data:

1. Information on time zones stored in the **Options of the application** table (SQL name: amOption).

Note: When the Use time zones option is selected, it enables you to define time zones for the server and data in GMT.

Figure 3.2. System data creation options



✓ Create system data
✓ Use time zones
Server time zone: GMT+01:00
Data time zone: GMT+01:00
✓ Use help on fields
☐ Use AutoCAD integration

If you check the **Use help on fields** box, AssetCenter Database Administrator prepares your database to accommodate the information concerning the extended help on fields.

If you think you will use the functions of integration of AssetCenter with AutoCAD, you should must select **Use AutoCAD integration without FacilityCenter** in order to prepare your database for this utilization.

Options of the Use an Import scenario frame

When you create your database, you can choose to automatically import certain information from the AssetCenter datakit. To do this:

1. Select the **Use an import scenario** option.
2. Select the information that you want to import using the **Import configuration** drop-down list. Depending on your needs, you can select all or part of the information in the data kit.

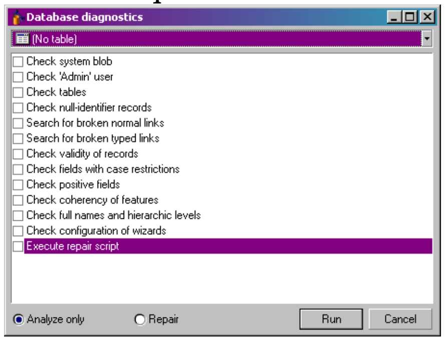
Note: You can also import the same information after the creation of the database using the AssetCenter import function.

3. You can then select the **Log file**. All the operations performed during this import, as well as any possible errors and warnings, will be written to this file.
4. In case of any problems, you can select the **Stop on error** option to stop the import of data when an any errors are encountered.

4 Diagnostic and repairs of a database

CHAPTER

The **Action/Diagnostics/Repair database** menu item enables you to test the integrity of an existing AssetCenter database. In order to do this (and for the menu to even be available), you have to already be connected to the database in question.



Database diagnostics

(No table)

- [ ] Check system blob
- [ ] Check 'Admin' user
- [ ] Check tables
- [ ] Check null-identifier records
- [ ] Search for broken normal links
- [ ] Search for broken typed links
- [ ] Check validity of records
- [ ] Check fields with case restrictions
- [ ] Check positive fields
- [ ] Check coherency of features
- [ ] Check full names and hierarchic levels
- [ ] Check configuration of wizards
- [ ] Execute repair script

[✓ Analyze only] Repair

You have two options for this:

- **Analyze only**: AssetCenter Database Administrator performs a simple diagnostic of the database, but does not perform any repairs in case of problems.

- **Repair**: AssetCenter Database Administrator performs a diagnostic of the database and makes any necessary reparations.

Warning: The reparation is conditioned by whether or not you activate the Execute repair script, in the same window. If this option is not activated, only the automatic reparation will take place. If you activated this option, however, the repair wizards will be launched after the automatic repairs.

AssetCenter Database Administrator checks and repairs the following items according to the options that you selected:

- Presence of the "sysblob" table (system data integrated with the database) in the database.

Note: Only the analysis of the "sysblob" table is possible.

- Existence of the "Admin" user (database administrator) in the table departments and employees.
- Presence of all tables in an AssetCenter database.
- Presence of records will null identifiers for all tables.
- Search for broken links (simple or complex)
- Checking validity of records

Note: The repair is done using a wizard that is launched automatically.

- Checking of broken fields (example: "user" instead of "User").

- Checking positive fields

Note: The repair is done using a wizard that is launched automatically.

- Checking coherence of features

Note: The repair is performed automatically or by using a wizard according to the value of the feature (text, number, date).

- Checking the full names and hierarchical levels: This mainly concerns the Employees table.
- Checking configuration of wizards

Note: Only the analysis is possible.

When you click **Run**, AssetCenter Database Administrator asks you to choose a log file to which will be written the operations and/or repairs performed during the diagnostic. During this diagnostic, the result of each test is indicated with an icon.

- ℹ indicates that the test has been successfully completed.
- ⚠ indicates that the test has failed but the database is useable.
- ⛔ indicates that the test has failed. The database may not be useable.

CHAPTER 5

Extracting information

AssetCenter Database Administrator enables you to extract information from the database by controlling the nature of the extracted information and the format of extraction.

There are several ways to obtain a description of the AssetCenter database structure:

- The **database.txt** et **tables.txt** files: They contain the complete structure of the database. These files are located in the Infos sub-folder of the AssetCenter installation folder.

Note: These files describe the default database structure. It does not include any customization you may have performed. In order for these description files to reflect the customization of your database, use the AssetCenter Database Administrator software with a connection to your database.

- AssetCenter Database Administrator program: It enables you to freely create description files of the AssetCenter database (tables, fields, links and indexes). It makes use of:
  - An AssetCenter database description file (**.dbb** file extension) or a connection to an AssetCenter database.
  - A model (**.tpl** file extension) that describes which information to generate. We provide you with the standard models, but you can create your own if you wish. Sophisticated models enable you to create RTF or HTML formatted files.

Note: Among the standard models provided with AssetCenter, one of them, the dbdict.tpl file, enables you to export all the customization information (including the information about features, calculated fields, configuration scripts, etc.) from your database to a standard text-formatted file. Used along with a "Source Control" tool, this description file can be very useful for keeping a trace of all customization modifications made to the database.

- The AssetCenter program.

This functionality is accessible via the **Action/Templates** menu item, which is then divided into several sub-items:

- **Select folder** allows you to specify the directory where AssetCenter Database Administrator will search for description templates. The software searches all the sub-directories in the selected directory.
- **Refresh list** restarts the search for description files from the directory specified the last time you changed directory.
- The remaining sub-menus consist of all the description templates found by AssetCenter Database Administrator in the folder. You can execute a description template by selecting its name from the list displayed in the menu.

Note: When executing a database description template, if AssetCenter Database Administrator encounters a variable whose value is not defined in the template, a screen is displayed for you to enter the value of that variable.

Introduction

The internal structure of a database can be viewed as a collection of hierarchical objects: a database that contains tables, which themselves contain fields, links, indexes, etc.

Describing a database means browsing its structure and extracting from it the information that you need. The way in which AssetCenter Database Administrator extracts information (as much the contents as the form under which the contents are extracted) is described in template files. These files are little programs whose syntax is easily understandable if you have a little programming experience. This syntax is described in the following sections of this chapter.

Database description parameters

The following parameters are used to describe the database:

```
Instance DATABASE
 Property P1-n
 Collection TABLES as TABLE

Instance TABLE
 Property P1-n
 Collection FIELDS as FIELD
 Collection LINKS as LINK
 Collection INDEXES as INDEX
 Object O1-n as <name of the instance>
```

```
Instance FIELD
 Property P1-n
 Object O1-n as <name of the instance>

Instance LINK
 Property P1-n
 Object O1-n as <name of the instance>

Instance INDEX
 Property P1-n
 Collection FIELDSINDEX as FIELD
 Object O1-n as <name of the instance>

Instance SCRIPT
 Property P1-n
 Collection REFERENCEDFIELD as SCRIPTFIELD
 Object O1-n as <name of the instancee>

Global Values
 Property P1-n
```

Describing the structure of an AssetCenter database is equivalent to describing the following instances:

- Database: the database itself.
- Table: database tables.
- Field: fields in the tables.
- Link: links in the tables.
- Index: indexes in the tables.
- Script: scripts for calculating the values of fields.

Each instance may be described with the following information:

- Property: a property of the instance.

  For example:

  ```
  Instance Table
   Property SqlName
  ```

The "SqlName" property gives the SQL name of the table.

- Collection: set of items that constitute one of the components of an instance.

  For example:

  ```
  Instance Index
   Collection FieldsIndex as Field
  ```

  An index (one of the components of the "Index" instance) is in particular defined by a set of fields ("FieldsIndex" collection). Each field is an item in the "Field" instance.

- Object: designated component of an instance.

  For example:

  ```
  Instance Link
   Object SrcField as Field
  ```

  A link (one of the components of the "Link" instance) is in particular defined by a source field ("SrcField" object). This field is a component of the "Field" instance.

Syntax for description templates

AssetCenter Database Administrator uses templates that define the information to extract, and how to process and present it.

These files must be in the following format:

- Type: text
- Character set: ANSI
- Extension: **.tpl**.

Their syntax is as follows:

- Fixed text
- Comments
- Including another template
- Browsing, sorting and filtering components
- $if...$else...$elseif...$endif conditions

- Functions available in description templates
- Processing the value of a property using a function defined in a template
- Deleting the end-of-paragraph mark
- Counting the number of browsed components
- Defining a global variable for the template

Fixed text

Any character string that does not start with the "$" character, and which is not part of a function, is generated as is by AssetCenter Database Administrator.

Note: To output a "$", the template must contain the following string: "$$".

For example:

The template:

```
List of tables.
SQL NAME
$$
```

generates the following output:

```
List of tables.
SQL NAME
$
```

Comments

Lines to be ignored by AssetCenter Database Administrator, and which are used to add comments to the template, must begin with the "$" character and be followed by a space.

For example:

```
$ This is a comment line.
```

Including another template

To include an external template in a template, use the following syntax:

```
$include "<full name of the template to include>"
```

For example:

```
$include "e:\modèles\dbscript.tpl"
```

Example of use: enables you to definitively define a reference template containing the functions that may be used by other templates, which include the reference template.

Browsing, sorting and filtering components

General syntax

```
$for [<name of the collection> | *] [alias <name
of the alias>] [sort (<name of the first property>
 (ASC|DESC) [, <name of the next property>
(ASC|DESC)])] [<filtering condition>]
...
$endfor
```

Browsing the components of a collection with "$for...$endfor"

To perform an iterative browse of the components of a collection, use the following syntax:

```
$for <name of the collection>
...
 $for <sub-collection>
 ...
 $endfor
$endfor
```

For example:

```
$for Tables
...
 $for Fields
...
 $endfor
$endfor
```

You must respect the hierarchy between collections. Examples:

1 The "Fields" collection is dependent on the "Tables" collection.
2 The "FieldsIndex" collection is dependent on the "Indexes" collection.

You may replace <name of the collection> with the "*" character. This calls all the collections in the current instance. Example:

```
$for Tables
...
 $for *
  $(SqlName)
  ...
 $endfor
$endfor
```

Allows you to obtain the SQL name of all the collections in the "Table" instance, i.e.: "Fields", "Links" and "Indexes".

Sorting the final result with "sort"

To sort the components in a collection, use the following syntax:

```
$for <collection> sort (<name of the first
property> (ASC|DESC) [, <name of the next property>
 (ASC|DESC)]))]
...
$endfor
```

Where:

1 ASC: ascending alphabetical order.
2 DESC: descending alphabetical order.

For example:

```
$for Tables sort (SqlName ASC)
...
 $for Fields sort (Usertype DESC, UserTypeFormat
ASC, SqlName ASC)
...
 $endfor
$endfor
```

Obtaining the properties of items in a collection or an object

To obtain the value of item properties of a collection or an object, use the following syntax:

```
$for <collection>
...
 $([<name or alias of the collection>.][<name of
the objet>.]<Property>
...
$endfor
```

Note: <name or alias of the collection> is not necessary if the property is called in a "$for... $endfor" loop in the collection.

For example:

```
$for Tables
 $for Fields
  $(Tables.SqlName) $(SqlName)
 $endfor

 $for Links
  $(Tables.SqlName) $(SqlName)
$(Reverselink.SqlName)
```

```
 $endfor

$endfor
```

Assigning an alias with "alias"

For the moment, aliases have no particular use.

Filtering the contents of the collection with "filter"

To filter the components of a collection, use the following syntax:

```
$for <collection> filter <filter condition>
...
$endfor
```

The filter condition is written in Basic.

For example:

```
$for tables filter $Left($SqlName, 1) = "p"
...
$endfor
```

Keeps only tables whose SqlName starts with the letter "p".

$if...$else...$elseif...$endif conditions

You can define the scope of a condition to include a property so that a component is selected.

Syntax:

```
$if <test condition>
...
$elseif <test condition>
...
$else <test condition>
...
$endif
```

Test conditions may be expressed using Basic formulas, functions defined in "$script...$endscript" format, and properties of instances.

For example:

```
$for Links
$if $(typed) = 0
$(Tables.SqlName) $(SqlName) $(SrcField.SqlName)
$(DstTable.SqlName)
$else
$(Tables.SqlName) $(SqlName) $(SrcField.SqlName)
$endif
$endfor
```

Functions available in description templates

AssetCenter Database Administrator contains a few pre-defined functions that may be used in templates.

ValueOf(<strProperty> as String) as String

Alternative syntax for calling the value of the **Property** property.

Property must be in uppercase.

For example:

```
$ValueOf("PRIMARYKEY")
```

Produces the same result as:

```
$(PrimaryKey)
```

SetProperty(<strProperty> as String, <strValue> as String, <iValueType> as Integer) as String

Creates a global variable called **Property** and of type **ValueType** for the template.

Property must be in uppercase.

Examples:

```
I = SetProperty("NEWPROPERTY", "2", VarType(2))
```

Creates a global variable called **NEWPROPERTY** for the template, with a numeric value of **2**, and returns a return code **I** equal to "0" if the variable was created correctly.

```
I = SetProperty("NEWPROPERTY", "Test",
VarType("Test"))
```

Creates a global variable called **NEWPROPERTY** for the template, with a text value of **Test** and returns a return code **I** equal to "0" if the variable was created correctly.

Exist(<strProperty> as String) as Integer

Tests for the presence of the **Property** global variable in the template.

For example:

```
Exist("NEWPROPERTY")
```

Returns a numeric value of **1** if the property exists, otherwise returns **0**.

LogError(<strErrorCode> as String, <strMessage> as String) as String

Defines an **ErrorCode** and an error message **Message** to be returned.

For example:

```
LogError(1, "the property not found")
```

Produces an ASCII error message in the defined cases.

SetOutput(<strFile> as String) as String

Defines the output file for the results. Has priority over the output file defined in the command line.

Examples:

```
SetOutput("e:\exportdb\sortie.txt")
```

Stores the results in a file called **"e:\exportdb\output.txt"**.

```
SetOutput("")
```

Displays the results on the screen.

CollectionCreate(<strName> as String) as Integer

Declares a new collection of database items. The name of the collection created must be a valid database collection, such as "Fields" or "Tables". This function and the following functions are typically used to scan the components of a collection. They can thus be used as a substitute for the proprietary "$For....$Next" syntax

For example:

```
CollectionNext() as
IntegerCollectionCreate("Fields")
```

The function returns "0" if the collection is created. Other values correspond to error codes that are displayed explicitly.

CollectionNext() as Integer

Carries out an iteration on the collection previously defined using "CollectionCreate()".

For example:

```
CollectionNext()
```

The function returns "0" if the iteration is carried out successfully. All other return codes correspond to errors. The function also returns an error if the last element of a collection is reached.

CollectionName() as String

Returns the name of the collection previously declared using the "CollectionCreate()" function.

For example:

```
strName=CollectionName()
```

CollectionIsFirst() as Integer

Allows you to test if the element of the collection to which the program is pointing is first of the collection.

For example:

```
CollectionIsFirst()
```

This function returns "1" if the element is the first of the collection; else "0" in all other cases.

CollectionIsLast() as Integer

Allows you to test if the element of the collection to which the program is pointing is last of the collection.

For example:

```
CollectionIsLast()
```

This function returns "1" if the element is the last of the collection; else "0" in all other cases.

CollectionCurrentIndex() as Integer

Returns the index number of the element of the collection to which the program is pointing. The collection must be declared beforehand using the "CollectionCreate()" function.

For example:

```
Number=CollectionCurrentIndex()
```

CollectionCount() as Integer

Returns the number of elements contained in the current collection previously declared using the "CollectionCreate()" function.

For example:

```
iCollec=CollectionCount()
```

Processing the value of a property using a function defined in a template

Using a function with "<function>"

You can define functions and process property values using those functions.

Syntax for using the function:

```
$<function>($(<property 1>,...,<property n>))
```

Examples:

```
$StrType($(Type))
```

```
$Duplicates($(Duplicates), $(NullValues))
```

Defining functions with "$script...$endscript"

Functions are defined within a Basic block delimited by two markers: "$script" and "$endscript":

```
$script
...
  Function
...
  End Function
...
$endscript
```

Functions have the following syntax:

```
Function <name of the function>({ByVal|ByRef}
[<name of the input variable> as <data entry
format>]*) as <output format>
...
End Function
```

Functions may be expressed using Basic formulas and instance properties.

Examples:

```
Function ReturnYesNo(ByVal iValue as Integer) as
String
if iValue = 1 then
  ReturnYesNo = "Yes"
else
  ReturnYesNo = "No"
end if
End Function
```

```
Function StrType(ByVal iValue as Integer) as String
```

```
select case iValue
  case 1: StrType = "byte"
  case 2: StrType = "short"
  case 3: StrType = "long"
  case 4: StrType = "float"
  case 5: StrType = "double"
  case 6: StrType = "string"
  case 7: StrType = "date+time"
  case 9: StrType = "blob"
  case 10: StrType = "date"
  case 12: StrType = "memo"
  case else
    Dim strError as String
   strError = "Type" + CStr(iValue) + " undefined"

    strType = LogError(1, strError)
End select
End Function
```

Deleting the end-of-paragraph mark

In some cases, you may need to insert information within a line, but the function generating the information must start at the beginning of a line.

In these cases, you can add the following string:

```
$nocr
```

to the end of the line before the function.

For example:

```
...
$for Indexes
 $(Tables.Sqlname) $(Sqlname) $nocr
 for FieldsIndex
  $if $(Islast) = 1
   $(Sqlname)
  $else
   $(Sqlname)$nocr
    $nocr
  $endif
$endfor
...
```

generates the following output:

```
...
amProduct Prod_BrandModel Brand, Model
amProduct Prod_CatalogRef CatalogRef
amProduct Prod_lCategIdBrand lCategId, Brand, Model
```

Counting the number of browsed components

To count the number of components that were browsed in a collection, taking a filter into account if appropriate, use the following syntax:

```
$for <collection> filter <filter condition>
 $(count)
...
$endfor
```

Defining a global variable for the template

To define a global variable, use the following syntax:

```
$<variable name> = <Basic formula>
```

Examples:

```
$A = 1
```

```
$Var = "text"
```

```
$A = $(A) + 1
```

```
$Form = Left($(Var), 2)
```

Information on certain database description parameters

This section includes information on the following description parameters:

- Database instance
- Table instance
- Field instance
- Link instance
- Index instance
- Script instance
- Global variables

Database instance

Properties

Table 5.1. Database instance properties

Property name	Description	Connection required
LoginName	Name of the Login that you use to access the database.	X
TableCount	Total number of tables in the database.	
Connected	This field may have two values: • 1: AssetCenter Database Administrator was executed in reference to a connection. • 0: AssetCenter Database Administrator was executed in reference to a database description file.	
Connection	Name of the AssetCenter connection used to access the database.	X

Table instance

Properties

Table 5.2. Table instance properties

Property name	Description	Connection required
Rights.Create	This property may have two values: • 1: the login has create access rights for the table. • 0: the login does not have create access rights for the table.	X
Rights.Delete	This property may have two values: • 1: the login has delete rights for the table. • 0: the login does not have delete rights for the table.	X
ComputeString	Table description string.	
InternalName	Internal name. This information has no particular use.	
Label	Label.	
Desc	Description.	
SqlName	SQL name.	
FieldCount	Total number of fields in the table.	
LinkCount	Total number of links in the table.	
IndexCount	Total number of indexes in the table.	
IsFirst	Indicates whether the item is located at the start of the collection, given the filter and the sort order: • 0: no • 1: yes	
IsLast	Indicates whether the item is located at the end of the collection, given the filter and the sort order: • 0: no • 1: yes	
Count	Count of the browsed items in the collection, given the filter.	
CurrentIndex	Position of the item in the collection, given the filter and the sort order.	

Objects

Table 5.3. Table instance objects

Object name	Description
MainIndex as Index	Main index.
PrimaryKey as Field	Primary key.
FeatureValueTable as Table	Table in which the feature values are stored.
Base as Database	Described database.

Field instance

Properties

Table 5.4. Field instance properties

Property name	Description	Connection required
Rights.Update	This property may have two values: • 1: the login has update rights for the field. • 0: the login does not have update rights for the field.	X
Rights.Write	This property may have two values: • 1: the login has create access rights for the field. • 0: the login does not have create access rights for the field.	X
Rights.Read	This property may have two values: • 1: the login has read rights for the field. • 0: the login does not have read rights for the field.	X
UserType	By default, the data entry and display format is the format for the "Type" property. The "UserType" property enables you to specify the data entry and display format when it is verified.	
Type	Storage format.	
UserTypeFormat	Additional information on the "UserType" parameter.	
Size	Maximum size of the values in the field, in number of characters.	
ReadOnly	Defines whether the field may be modified, whatever the access rights for the person connected to the database. This field may have two values: • 1: the field can never be modified by users. • 0: the field may be modified by a user, if the user has the required access rights.	
Historized	This field may have two values: • 1: history is recorded for the field. • 0: history is not recorded for the field.	
ForeignKey	This field may have two values: • 1: the field is a foreign key. • 0: the field is not a foreign key.	
PrimaryKey	This field may have two values: • 1: the field is a primary key. • 0: the field is not a primary key.	
InternalName	Internal name. This information has no particular use.	

Property name	Description	Connection required
Label	Label of the field, as it appears in detail screens, for example.	
Desc	Description.	
SqlName	SQL name.	
Comment	Comments on the use of the field.	
Sample	Examples of values that may be assigned to the field.	
Warning	Important information for the field.	

Objects

Table 5.5. Field instance objects

Object name	Description
Base as Database	Described database.
Table as Table	Table containing the field.
MandatoryScript as Script	Mandatory calculation script for the field.
DefaultScript as Script	Default script for calculating the field's value.

Possible values for the "Type" property

Table 5.6. Possible values for the "Type" property

Stored value	Literal value	Description
1	byte	Integer from -128 to +127.
2	short	Integer from -32,768 to +32,767.
3	long	Integer from -2,147,483,647 to +2,147,483,646.
4	float	4-byte floating point number.
5	double	8 byte floating-point number.
6	string	Text in which all characters are allowed.
7	date+time	Date and time.
9	blob	Variable-length binary field (or BLOB Binary large object) can stores images and forms, for example, without size restriction.
10	date	Date only (no time).
12	memo	Variable-length text field.

Possible values for the "UserType" property

Table 5.7. Possible values for the "UserType" property

Stored value	Literal value
0	Default
1	Number
2	Yes/No
3	Money
4	Date
5	Date+Time
7	System itemized list
8	Custom itemized list
10	Percentage
11	Time span
12	Table or field SQL name

Possible values for the "UserTypeFormat" property

This property is used when the "UserType" property is set to:

- "Custom Itemized list": indicates the name of the itemized list associated with the field.
- "System Itemized list": provides the list of itemized entries.
- "Time span": indicates the display format.
- "Table or field SQL name": the property contains the SQL name of the field which stores the SQL name of the table containing the field that specifies the described field.

Link instance

Properties

Table 5.8. Link instance properties

Property name	Description	Connection required
Rights.Update	This property may have two values: • 1: the login has update rights for the link. • 0: the login does not have update rights for the link.	X
Rights.Write	This property may have two values: • 1: the login has create access rights for the link. • 0: the login does not have create access rights for the link.	X
Rights.Read	This property may have two values: • 1: the login has read rights for the link. • 0: the login does not have read rights for the link.	X
Type	Type of the link.	
UserType	Type of information managed by the link.	
Typed	Indicates whether or not the link's target table is pre-defined. When this is not the case, the table's SQL name is stored in one of the fields of the record. • 1: The target table is pre-defined. • 0: The target table is pre-defined.	
Historized	This field may have two values: • 1: history is recorded for the field. • 0: history is not recorded for the field.	
Cardinality	Cardinality of the link.	
InternalName	Internal name. This information has no particular use.	
Label	Label.	
Desc	Description	
SqlName	SQL name.	

Objects

Table 5.9. Link instance objects

Object name	Description
Base as Database	Described database.
SrcField as Field	Source field.
SrcTable as Table	Source table.
DstTable as Table	Target table.
DstField as Field	Target field.
RelTable as Table	Relation table.
RelSrcField as Field	Source field of the relation table.
RelDstField as Field	Target field of the relation table.
TypeField as Field	When a link's target table is not pre-defined, this property indicates the field containing the SQL name of the target table.
ReverseLink as Link	Reverse link.

Possible values for the "Type" property

Table 5.10. Possible values for the "Type" property

Stored value	Literal value
1	Normal
2	Own
4	Define
8	Neutral
16	Copy
18	Owncopy

Possible values for the "UserType" property

Table 5.11. Possible values for the "UserType" property

Stored value	Literal value
0	Normal
1	Comment
2	Image
3	History
4	Feature value

Index instance

Properties

Table 5.12. Index instance properties

Property name	Description
Duplicates	Indicates whether or not the index can have more than one non-NULL value. • 1: you can create several records where the group of fields in the index can have exactly the same value. • 0: you cannot create more than one record where the group of fields in the index has a given value.
NullValues	This property is only significant if the "Duplicates" property is set to "No". It indicates whether or not the index can have more than one "NULL" value (the index has a value of "NULL" if all the fields constituting the index have a value of "NULL".) • 1: you can create several records having an index of NULL. • 0: you can only create one record having an index of NULL.
InternalName	Internal name. This information has no particular use.
Label	Label as it appears in the detail screens.
Desc	Description.
SqlName	SQL name.

Objects

Table 5.13. Index instance objects

Object name	Description
Base as Database	Described database.
Table as Table	Table containing the index.

Script instance

Properties

Table 5.14. Script instance properties

Property name	Description
CalcMode	Indicates whether the value of the field is "yes", "no", or if a script calculates one of these two values. This property can have one of the following values: • 0: no • 1: yes • 2: script
ScriptType	Type of information managed by the script. This property can have one of the following values: • 1: data entry for a field is mandatory • 2: by default, display a feature in a table • 3: available character for a feature in a table • 4: record history for a field • 5: default value of a field • 6: **For inheritance purposes** (SQL name: bForInheritance) field of a feature in a table
Source	Script that calculates the value of a field as it is displayed in the interface.
RawSource	Script that calculates the value of a field as it is stored in the database.
VbReturnType	Type of string calculated by the script: • Integer: Integer from -32,768 to +32,767 • Long: Integer from -2,147,483,647 to +2,147,483,646 • Double: 8 byte floating point number • String: Text for which all characters are accepted • Date: Date (not time).

Objects

Table 5.15. Script instance objects

Object name	Description
Table as Table	Table containing the field whose value is calculated by the script.
Field as Field	Field containing the value calculated by the script.

Global variables

Table 5.16. Global variables

Property name	Description
Userlogin	Login you used when you connected to the database.
Time	Time at which AssetCenter Database Administrator was executed.
Date	Date on which AssetCenter Database Administrator was executed
Dbb.Fullname	Full pathname of the database description file used.
Dbb.Shortname	Name (without extension) of the database description file used.
Dbb.Path	Pathname for the database description file used.
Dbb.Name	Name (with extension) of the database description file used.
Dbb.Ext	Extension of the database description file used.
Template.Fullname	Full pathname of the database description template used.
Template.Shortname	Pathname of the database description template used (without the extension).
Template.Path	Path of the database description template used.
Template.Name	Name of the database description template used (with the extension).
Template.Ext	Extension of the database description template used.

CHAPTER

6 Customizing the database

AssetCenter Database Administrator enables you to perform different types of customizations.

- A customization of existing objects (tables, fields, links, screens, etc.).
- A customization by creating new objects.

Customizing existing objects

AssetCenter Database Administrator authorizing a limited customization on the existing database objects. In order to avoid any problems while using AssetCenter, certain object values and certain objects are in read-only. Typically, you cannot modify the pages of an existing detail.

Two distinct forms of database customization are handled by AssetCenter Database Administrator:

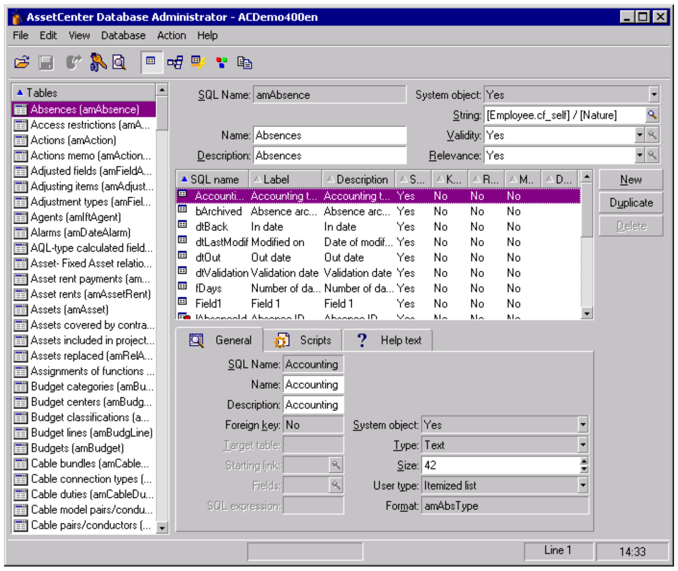
- Customizing before creating the database.
- Customizing after creating the database.

Note: The difference between these two cases is that you can only modify the size of "Text" fields before creating the database.

For each of these two cases, two levels of customization are possible:

- Customizing a table.
- Customizing objects (fields, links, indexes, screens, pages) of a table.

Figure 6.1. Database customization screen



Tables

- Absences (amAbsence)
- Access restrictions (amA...
- Actions (amAction)
- Actions memo (amAction...
- Adjusted fields (amFieldA...
- Adjusting items (amAdjust...
- Adjustment types (amFiel...
- Agents (amIftAgent)
- Alarms (amDateAlarm)
- AQL-type calculated field...
- Asset- Fixed Asset relatio...
- Asset rent payments (am...
- Asset rents (amAssetRent)
- Assets (amAsset)
- Assets covered by contra...
- Assets included in project...
- Assets replaced (amRelA...
- Assignments of functions ...
- Budget categories (amBu...
- Budget centers (amBudg...
- Budget classifications (a...
- Budget lines (amBudgLine)
- Budgets (amBudget)
- Cable bundles (amCable...
- Cable connection types (...
- Cable duties (amCableDu...
- Cable model pairs/condu...
- Cable pairs/conductors (...

SQL Name: amAbsence
System object: Yes
String: [Employee.cf_self] / [Nature]
Name: Absences
Validity: Yes
Description: Absences
Relevance: Yes

SQL name	Label	Description	S...	K...	R...	M...	D...
Accounti...	Accounting t...	Accounting t...	Yes	No	No	No	
bArchived	Absence arc...	Absence arc...	Yes	No	No	No	
dtBack	In date	In date	Yes	No	No	No	
dtLastModif	Modified on	Date of modif...	Yes	No	No	No	
dtOut	Out date	Out date	Yes	No	No	No	
dtValidation	Validation date	Validation date	Yes	No	No	No	
fDays	Number of da...	Number of da...	Yes	No	No	No	
Field1	Field 1	Field 1	Yes	No	No	No	

SQL Name: Accounting
Name: Accounting
Description: Accounting
Foreign key: No
System object: Yes
Target table:
Type: Text
Starting link:
Size: 42
Fields:
User type: Itemized list
SQL expression:
Format: amAbsType

Customizing tables

To customize the table, you can modify:

- The **Description** field.
- The **Label** field, which contains the name of the table as it is displayed under AssetCenter.
- The **String** field, which enables you to build the string which represents a record from this table under AssetCenter.

- The **Validity** field applies to all records of a table in the database. It enables you to define the conditions of validity of creation or modification of a record in the table.
  - If this field is set to **Yes**, a record in the table can always be created or modified.
  - If this field is set to **No**, a record in the table can never be created or modified.
  - If this field is set to **Script**, you can define a script conditioning the validity of creation or modification of a record in the table.

    For example, for **Numerical** type features, if is possible to restrict the creation of a feature if its value is outside certain limits. The following script concerns the "Validity" field in the "amFeature" table:

```
if [seDataType] = 1 and [fMin] > [fMax] Then
  Err.Raise(-1, "The value of the 'minimum'
field must be inferior to the value of the
'maximum' field.")
  RetVal = FALSE
Else
  RetVal = TRUE
End If
```

Note: When creation or modification of a record is invalidated by the value of the "Validity" field for the table concerned, it is good practice to display an explicit error message via the standard Basic function "Err.Raise" in order to warn the user. If you do not do this, the user will not necessarily be able to understand why it is not possible to modify or create the record.

- The Relevance field applies to all the records of a database table. It enables you to delete records that are no longer pertinent (according to the criteria that you defined) and thus to rationalize the data in your database.

- If this field is set to Yes, a table record is always relevant, whatever the value of its fields. This record will never be deleted.
- If the field is set to No, a table record will never be relevant. It will systematically be deleted.
- If this field is set to Script, you can define a script conditioning the validity of creation or modification of a record in the table.

  For example, there is little interest in conserving the records of the Comments table (amComment) that are empty (memComment field empty). This type of records is not relevant. For systematize their deletion, you can write the following Relevance script for the Comments table:

  ```
  If [memComment]="" Then
    RetVal= FALSE
  Else
    RetVal = TRUE
  End If
  ```

Customizing objects

The second part of this customization pane enables you to read all the objects of a given type.

When you select an object in the list, AssetCenter Database Administrator displays the description of this object in the third part of the customization pane.

Note: The red icon identifies the primary key of the table.

Customizing a field, a link or an index

For this type of object, you can modify:

In the **Detail** tab

- The **Label** field.
- The **Description** field.
- The **Size** field enables you to specify the size of "Text" type fields. It can only be accessed when customizing a database before creating it or when modifying a database description file. The maximum size is 255 characters.

And in the **Scripts** tab:

Note: Except where specified, the following attributes are taken into account both by the user interface and external tools when accessing the database.

- The **History** field: indicates whether history is kept for modifications made to the field and possible conditions (via scripts).
- The **Read only** field: indicates whether it is possible to modify the field via the user interface and under what possible conditions (via scripts).

Note: This attribute is not taken into account when importing data using external tools. The import module does not take these into account either, as long as the fields are not defined as "Read only" at the level of mappings. On the other hand, if an import script maps a source field to a read-only target field, the read-only field can be modified all the same.

- The **Filter** field is not used in this version of AssetCenter.
- The **Formatting** field: enables you to format the value of the field automatically before storing it in the database.

For text fields:

- **Standard**: stores the value as entered.
- **Uppercase**: converts the value to uppercase before storing it.
- **Lowercase**: converts the value to lowercase before storing it.
- **Automatic**: converts the first letter to uppercase before storing the value.

For numeric fields:

- **Standard**: accepts all numbers, both positive and negative.
- **Positive**: rejects negative values. The warning message is displayed.

Note: If you modify this attribute, existing values in the database will not be converted.

- The "Mandatory" field enables you to define conditions making a field mandatory.

Note: Making a field mandatory can pose problems if it is not always visible (for example, if its being displayed is determined by the value of a field). Always bear this in mind when configuring a field or writing a script.

- The "Default" field specifies the field's default value. This value is automatically proposed by AssetCenter during the creation of a new record. The default values are defined using a Basic script.

Note: Calculated fields can only be used in the calculation of the default value of a standard field if their type is Calculated string or Basic script.

Note: It is not possible to attribute a default value to links to the amComment table.

In the **Help text** tab, you can customize the extended help concerning a database object. This help is available in AssetCenter via "Shift+F1" (or the **Help/ Help on this field** pop-up menu item) and can include up to three sections. By default, these sections are entitled "Description", "Example" and "Precautions". The titles of these sections can be customized by modifying the labels of the links with SQL names "Comment", "Sample" and "Warning" in the **Help on fields** table (SQL name: amHelp).

Note: The other fields in this section of the database customization screen are shown for informational purposes only and cannot be customized. Configuring the objects in this screen works in the same way as is available via the Configure object pop-up menu command.

Customizing a detail

For this type of object, you can modify:

In the **General** tab

- The **Label** field.
- The **SQL expressions** field. This field enables you to enter an SQL query that will be executed when you open the detail.

The **Detail** tab

- Proportions of the detail field
- Title
- Title of the detail
- System filter field
- Configuration of ADBs field
- (Functional) Domain

  Column names and widths

Creating action buttons

The **Buttons** tab enables you to create action buttons that will be displayed in the detail. To create a button:

- Click .
- A line is added to the list of buttons exposed in the tab. Click on each of the line's cells to define the following button properties:
  - Name: internal name of the button. It enables you to identify the button in a unique manner.
  - Description: label of the button as it is displayed in the graphical interface of AssetCenter.
  - On the fly: Enables you to specify if the actions taken when you click the button authorize the on-the-fly creation of records.
  - Multiple selection: Enables you to specify if the actions taken when you click the button can apply to several records.
  - No selection: Enables you to specify if the actions taken when you click the button can be launched without any record having been selected.
  - **Associated action**: Enables you to define the action executed when you click the button. Enter the SQL name of the action to execute.

```
<Type of action>:<SQL name of the action, of
the view, etc. ...>
```

In this syntax, the type of action can take the following values:

- **A** for an action.
- **S** for a screen.
- **V** for a view.
- **F** for a form.
- **R** for a report.

To delete an action button:

1. Select the button in the list displayed in the **Buttons** tab.
2. Click ▬.
3. Click **Modify**.

To reorganize the order in which the buttons appear in the detail, use the ▲ and ▼ buttons.

Creating new objects

AssetCenter Database Administrator enables you to create new objects for the database.

Methods of creation

The following steps propose a method of creating new objects. These steps assume that you are creating the largest object: a table. Each step corresponds to a particular selection in this chapter.

- Create your table.
- Create the fields, links and index of your table.
- Create the details for your table.
- Create, if you want, action buttons present in the detail.
- Create the pages of your details.
- Add the pages to your details.

- Save your modifications.
- If this fails, propagate your modifications.

Creating a table

To create a new table:

- Select the **Database/Add a table** menu item.
- AssetCenter Database Administrator displays the following creation window:
- In this window, populate the standard fields associated with a table:
  - The SQL name field enables you to identify the new table in a unique manner and, notably, to reference it in a Basic script.
  - The Label field contains the name of the table as it is shown in AssetCenter.
  - The Description field.
  - The primary key field contains the SQL name of the field used as the table's primary key.
  - If you want to associate features to your new table, select the Features option. AssetCenter Database Administrator will automatically create the additional tables necessary to deal with the features.
- Click **Create**. AssetCenter creates the table as well as the field defines as the primary key for this table, and it puts you in position to edit the fields for this table.

Creating a field, link or index

To create a field, link or index for a table:

1. Select one of the following menu items: **Database/Add a field**, **Database/Add a link**, **Database/Add an index**.
2. Populate the different properties of each object. These properties are described in an exhaustive manner in the section on Customizing objects, and are thus not detailed here.

Creating a detail

A detail is a graphical view of the information stored in the objects of a table. The screens displayed in AssetCenter when you select a menu, click a toolbar icon or choose a view are all examples of a detail. Details are composed of several pages that are regrouped by tabs in the graphical interface of AssetCenter.

Note: AssetCenter is provided with a set of standard details that are used to support the functions of the application. These details cannot be edited or modified. If you want to add a page to an existing detail, for example, you need to duplicate the detail, name it, then add the desired items.

To create a detail:

1. Select in the left-hand pane the table for which you want to create a new detail.
2. Select the Display/Details or click the button on the toolbar.
3. Select the **Database/Add a detail** menu or click **New**.
4. Click **Create**.
5. Populate the following fields in the **General** tab:
   - SQL name: SQL name of the detail. This name enables you to identify the detail in a unique manner and to reference it in scripts or queries.
   - System name: internal name of the detail. This name is never exposed in the AssetCenter interface.
   - Label: label of the detail.
6. Populate the following fields in the **Detail** tab:
   - Title: defines the title of the detail as it appears in the list of screens (**Tools/List of screens**, in AssetCenter).

- Title of detail: defines the title of the window displayed in AssetCenter. The **String** of the description is added to this title.
- Domain: functional domain of this detail. If the functional domain that you enter already exists in the database (created using the **Administration/Functional domains** menu item), you will automatically have access to your screen in the functions pane under the corresponding functional domain.
- Proportions of the detail: ratio between the detail and the list
- Column names and widths: This field enables you to define the columns that will be displayed in a list. The syntax of this field is the following:

```
<SQL name of the field for the column
1>,<Proportional size of the column>,...
```

7 Click **Modify**.

You just created the empty shell of your new screen. Now all you need to do is populate this screen by adding pages that contain fields or links, and possibly action buttons.

Creating action buttons

AssetCenter Database Administrator offers you to possibility of creating buttons in your details. These buttons enable you to trigger the execution of an action, the display of a screen, the printing of a report or form, or to open a view.

To create this button:

1. Select the detail to which you want to add a button.
2. Select the **Buttons** tab of the detail.
3. Click ➕. A new item is created in the list of buttons for this tab.
4. You can directly edit the values for each property by clicking its value.

   - **Name:** SQL name of the button. This name enables you to identify the detail in a unique manner and to reference it in Basic scripts or queries.

- **Description:** label of the button as it is displayed in the graphical interface of AssetCenter.
- **On the fly:** Enables you to specify if the actions taken when you click the button authorize the on-the-fly creation of records.
- **Multiple selection:** Enables you to specify if the actions taken when you click the button can apply to several records.
- **No selection:** Enables you to specify if the actions taken when you click the button can be launched without any record having been selected.
- **Associated action:** Enables you to define the action that is executed when you click the button. The syntax of this field is the following:

  ```
  <Type of action>:<SQL name of the action, of
  the view, etc.>
  ```

  In this syntax, the type of action can take the following values:

  - **A** for an action.
  - **S** for a screen.
  - **V** for a view.
  - **F** for a form.
  - **R** for a report.

5 Click **Modify** to validate your changes.

Creating a page

In AssetCenter, a page is graphically represented by a tab containing fields. To create this page:

1 Select in the left-hand pane the table for which you want to create a new page.

2 Select the Display/Pages or click the button on the toolbar.

3 Select the **Database/Add a page** menu or click **New**.

4 Click **Create**.

5 Populate the following fields in the **General** tab:

- SQL name: SQL name of the page. This name enables you to identify the page in a unique manner and to reference it in scripts or queries.
- System name: internal name of the page. This name is never exposed in the AssetCenter interface.
- Title: title of the page. This title appears as the name of the tab in the AssetCenter interface.

6 Click the **Contents** tab of the page's detail. You can now select the fields that will be available in the page that you are created. To do this:

- Select in the list to the right (**Fields and links**) the field or the link that you want to add to the page, and click the ◀ button to transfer it to the list on the left (**List of fields**). You can also double-click the field or link on the right to transfer it to the left.
- If you want to delete a field from your page, select it in the list on the left and click the ▶ button. You can also double-click the field on the left to perform this same operation.
- The layout of this page and graphical organization of the fields on the page is automatically calculated by AssetCenter. You can, however, define the order of the fields' appearance on the page by ordering them in the left-hand pane using the buttons: ▲, ▼, ⤓ and ⤒.

7 Click **Modify**.

Adding a page to a detail

To add a page to a detail:

1 Select the detail to which you want to add a page.

2 Select the **Pages** tab of the detail. You can now select the page(s) that you want to add to the detail. To do this:

- Select in the list to the right (Available pages) the page(s) that you want to add to the detail, and click the ◀ button to transfer them to the list on the left (**List of fields**). You can also double-click the field or link on the right to transfer it to the left.

- If you want to delete a page from your detail, select it in the list on the left and click the button. You can also double-click the field on the left to perform this same operation.
- To define the order of the tabs' appearance you need to order them in the left-hand pane using the buttons: , , and . The first page in the list will be the first tab displayed; the second page will be the second tab appearing, etc.

Saving modifications

AssetCenter Database Administrator uses the database's Smart update engine to save your customizations. To save these modifications:

1. Select the File/Save or click the button on the toolbar.
2. AssetCenter Database Administrator displays a save window. You must define a **Log folder** in which the following information will be stored:
   - The operations performed while saving the modifications are stored in the **sdu.log**.
   - The SQL queries used to modify the database are stored in the **sdu.sql** file.
   - An XML file (**sdu.xml**) contains the structural differences between the initial database and the customized database.
3. Click **Update**. AssetCenter Database Administrator then updates your database by inserting the objects that you created.

Verifying your modifications

To check your modifications, you need to launch AssetCenter and connect to the customized database. Using the method previously defined, we created a new table. To access this new table:

1. Select the **Tools/List of screens** menu item.
2. Search for your detail in the list that appears on screen and click **OK**. Your new detail will appear.
3. You can create a view, if you want, for more facility. Select the **Tools/Views/Create view from current window** menu item.
4. Enter a **Name** for your view. This name will appear in the Functions and favorites panel.
5. Select a functional **Domain**. The **Name** of your view will appear under this functional domain the Functions and favorites panel.
6. You can now directly access your new detail.

Important note

In AssetCenter, all objects are defined by SQL names, in particular, the actions, view and screens. To open one of these objects from a menu or a function panel, AssetCenter uses the SQL name. If several objects, such as a view and a screen, have the same SQL name, AssetCenter tries to open the objects in the following order:

- View
- Screen

If, for example, you crate a new detail for the Departments and Employees table (amEmplDept), and you associate a view names amEmplDept to this detail, AssetCenter will open your detail when you select the **Portfolio/Departments and Employees** menu item, or when you click the **List of Employees** in the Functions and favorites panel.

December 19, 2001