CHAPTER 8: DIMENSIONS

Objectives

The objectives are:

- Modify the DimensionsManagement codeunit.
- Configure master tables and forms for dimensions.
- Configure tables and forms for seminar registrations.
- Configure tables and forms for posted seminar registrations.
- Configure tables, forms, and codeunits for seminar posting.
- Create a Seminar Invoice Report for dimensions.

Introduction

Dimensions are data added to an entry to act as markers for the program, which allows entries with similar characteristics to be grouped for analysis purposes. Every entry in the program can have dimensions, including:

- Master files
- Transaction document headers and lines
- Journal lines
- Ledger entries
- Posted documents and their lines.

Dimensions are used to describe how analysis occurs. A two-dimensional analysis, for example, could be analysis of sales for an area. However, by using more than two dimensions when creating an entry, a more complex analysis can be performed. For example an analysis of sales for each sales campaign for each customer group in each area.

Each dimension can have an unlimited number of dimension values. For example, a dimension called Department can have dimension values of Sales, Administration, Purchasing, and so on. Users define and tailor these dimensions and values to their company's needs.

Dimension Types

There are three types of dimensions: Global, Shortcut, and Budget.

Global: When setting up dimensions in the G/L Setup, two of them can be global dimensions. These dimension types can be used throughout the program as a filter for G/L Entries and on reports, account schedules, and batch jobs.

Shortcut: Enter shortcut dimensions on journal and document lines. These lines have eight fields that are designated for dimensions. The first two are always the global dimensions, but the remaining six can be selected from those set up as shortcut dimensions in the G/L Setup. Dimensions that are not set up as shortcut dimensions can also be specified, but these must be set up in a separate Dimensions window for the header or line.

Budget: Four dimensions can be defined for each budget.

When a global, shortcut, or budget dimension us set up, the program automatically renames all fields that use the dimension type with the code caption specified in the dimension setup.

Where dimensions are stored depends on the type of entry. The following table shows different tables that contain dimensions with the types of entries with which they are associated.

Dimension Table	Type of Entry
352 Default Dimension	Master file records
355 Ledger Entry Dimension	Ledger entries
356 Journal Line Dimension	Journal lines
357 Document Dimension	Document headers and lines
358 Production Document Dimension	Production orders, lines, and components
359 Posted Document Dimension	Posted document headers and lines
361 G/L Budget Dimension	Budget entries

Code Walkthrough – Dimension Management Codeunit

Codeunit **DimensionManagement** (408) contains functions that help populate the Dimension tables.

The **TypeToTableID** functions in the **DimensionsManagement** codeunit take one option parameter and then return the table ID of the table related to that parameter. For example, if "G/L Account" is passed in, the function returns table ID 15. These functions are used, through other functions, to populate the dimension tables where the table ID is needed. Without them, it would be necessary to write a similar case statement every time the table ID of an option field was needed. The solution in this chapter will require a new **TypeToTableID** function to be added for the **Seminar** table for table 95000.

Notice the **SetupObjectNoList** function. This function sets a temporary record object with a list of tables that use dimensions. This function is used in table 352, in the **OnLookup** trigger of the **Table ID** field. Modify **SetupObjectNoList** to include the master tables.

Other functions of note in this codeunit are **ValidateDimValueCode** and **SaveDefaultDim**. **ValidateDimValueCode** takes in two parameters, the field number and the dimension value. The function checks to see that the dimension is valid, and returns an error if it is not. **SaveDefaultDim** saves the dimension in the default dimension table.

Many of the functions in the DimensionManagement codeunit will be required to add dimensions functionality to the Seminar module. In general, there are functions that correspond to each type of **Dimension** table for getting default dimensions, as well as updating, saving, deleting, inserting, and validating the consistency and combinations of dimensions.

Using Microsoft Dynamics NAV Developer's Toolkit

When developing in Microsoft Dynamics™ NAV, information such as where and how often a certain table or field is used is often required. This kind of analysis is not possible in the Object Designer, so the Microsoft Dynamics NAV Developer's Toolkit is available to aid in analyzing databases, as well as comparing and merging databases.

In the Developer's Toolkit, objects can be imported and analyzed by searching for certain words or phrases, revealing where a certain object, field, or key is used, or finding the relations between objects. Two or three versions of a database can also be complared to see any differences.

The Developer's Toolkit works with the text version of a Microsoft Dynamics NAV database or object by parsing the different elements and storing the results in its own database. The Developer's Toolkit makes no modifications to the database, although objects and databases can be exported as text files. The database that the Toolkit uses is a Microsoft Dynamics NAV C/SIDE database, which is not the same as a Microsoft Dynamics NAV database. To read from and write to the database, the Developer's Toolkit uses C/FRONT® , which is the toolkit that makes it possible to develop applications in the C programming language that access a C/SIDE database. There is more information about C/FRONT in Appendix C of this manual.

The Developer's Toolkit allows direct access to one or more Microsoft Dynamics NAV clients that are open on your computer. Through this access, it is possible to:

- Quickly import objects.
- Update from or to the client.
- Export objects using an export worksheet.

Some of the features of the Developer's Toolkit include:

- The ability to view object relationships in tree or diagram form
- The ability to search the database for a word or phrase using Source Finder
- The ability to compare and merge databases or simply to compare two versions of a database
- The ability to list all of a specific object's relations to or from other objects
- The ability to list all the places where a specific object (or an element of an object such as a field, key, trigger, or procedure) is used
- The ability to view the method flow in C/AL

Dimensions in Master Tables, Forms, and CodeUnits

The next step is to add dimension functionality to the seminar module. This requires making dimensions available from master files, seminar registrations, seminar posting, and invoicing.

Solution Analysis

The purpose of implementing dimensions on the master file level is to enable analysis of master data.

In accordance with Microsoft Dynamics NAV standards, the seminar master files should be associated with default dimensions. The master file dimensions then flow to the transactions in which the master data is used, and these document dimensions eventually flow to the ledger entries and posted document dimensions.

Solution Design

Enabling dimensions for master files requires changes to the master tables and forms, as well as to the codeunits that manage dimensions.

User interface changes will include adding menu items for dimensions to the **Seminar Card** form (123456700), the **Seminar Room Card** (123456703), and the **Instructors** form (123456705), forms from which the master data is defined.

Entries for master files will be included in the **Default Dimension** table (352). This table and the **DimensionManagement** codeunit must be modified so that they accept entries from the **Seminar**, **Seminar Room**, and **Instructor** tables.

When the user creates a new record in a master table, the program inserts or updates the default dimensions. Likewise, when the user deletes a record, the program automatically deletes the associated dimensions.

ValidateShortcutDimCode: When the user enters a value in a dimension field, use this function to validate the entry. The **DimensionManagement** codeunit has a function called **ValidateDimValueCode** that helps with this.

Finally, these tables will require modification:

- Table 123456700 Seminar
- Table 123456702 Seminar Room
- Table 123456703 Instructor
- Modify the code in Table 352 Default Dimension so that the table accepts entries from the seminar master files.

Lab 8.1 - Modifying the DimensionManagement Codeunit

The first step in enabling dimensions for master files is to modify the **DimensionManagement** codeunit.

Open codeunit **DimensionManagement** (408) and look at the functions. Notice a number of generic functions that have parameters like **TableID** and **DocType**, so they can be used any table. There are also functions like **TypeToTableID2** that can be used for one specific table. Begin modifying codeunit 408 by adding a table-specific function. Follow these steps:

1. Create a function called **TypeToTableID123456700** that takes a parameter called Type with a data type of Option and the options Resource, G/L Account. The function returns an integer.
2. Enter code in the function trigger so that the function returns the ID of the Resource table when the Type is Resource, and returns the ID of the G/L Account table when the Type is G/L Account.
3. Now modify the **SetupObjectNoList** function so that it takes into account the three tables **Seminar**, **Seminar Room**, and **Instructor**. Change the **Dimensions** property of the **TableIDArray** variable from 20 to 23.
4. Comment out the existing code in the function trigger that sets the **NumberOfObjects** variable to 20. Add code so that the **NumberOfObjects** variable is set to 23. Set the values of **TableIDArray**[21]-[23] to correspond to the **Seminar**, **Seminar Room**, and **Instructor** tables.

Lab 8.2 - Modifying the Tables and Forms for Dimensions in Master Files

The next stop is to modify the master file tables and forms to enable dimensions:

1. Add two new fields to the **Seminar** table (123456700) as follows:

No.	Field Name	Type	Length	Comment
15	Global Dimension 1 Code	Code	20	Caption Class is 1,1,1. Relation to the **Code** field on the Dimension Value table where the **Global Dimension No.** is 1.
16	Global Dimension 2 Code	Code	20	Caption Class is 1,1,2. Relation to the **Code** field on the Dimension Value table where the **Global Dimension No.** is 2.

2. Create a new function for the table called **ValidateShortcutDimCode**. This function takes two parameters: an integer called **FieldNumber** and a code variable with a length of 20 called **ShortcutDimCode**, which is passed by reference.

3. Enter code in the function so that it runs the **ValidateDimValueCode** and **SaveDefaultDim** functions from the **DimensionManagement** codeunit and modifies the table.

4. Enter code in the appropriate triggers so that when the user enters or changes a value in the **Global Dimension Code 1** or **Global Dimension Code 2** fields, the program runs the **ValidateShortcutDimCode** function (the **FieldNumber** values are 1 and 2, respectively).

5. Enter code in the appropriate trigger so that when the user inserts a new record to the table, the program runs the **UpdateDefaultDim** function from the **DimensionManagement** codeunit.

6. Enter code in the appropriate trigger so that when the user deletes a record from the table, the program runs the **DeleteDefaultDim** function from the **DimensionManagement** codeunit.

7. Add the same global dimension fields to the **Seminar Room** table (123456702) added to the **Seminar** table. Add these fields as the last two fields of the table. Add the same functions and code modifications as for the **Seminar** table.

8. Add the same global dimension fields to the **Instructor** table (123456703) added to the **Seminar** and **Seminar Room** tables. Add these fields as the last two fields of the table. Add the same functions and code modifications as for the **Seminar** and **Seminar Room** tables.

9. When dimensions are added to any master file in Microsoft Dynamics NAV, they are stored in the **Default Dimension** table (352). Modify this table so that it can accept entries from the seminar master files. Enter code in the **UpdateGlobalDimCode** function trigger in the **Default Dimension** table for each of the three master tables, **Seminar**, **Seminar Room** and **Instructor**, so that, depending on the **TableID**, the program gets the appropriate record from the appropriate table, sets the appropriate **Global Dimension Code** field to the **NewDimValue**, and modifies the table.

10. Add the menu items for Dimensions to the **Seminar Card** and **Seminar Room Card** forms as follows:

Menu Button	Options	Comment
Seminar	Dimensions (SHIFT + CTRL + D)	Opens the **Default Dimensions** form (540) for the selected entry. The link should run whenever the form is updated.

11. Add the menu button and menu items for Dimensions to the **Instructors** form as follows:

Menu Button	Options	Comment
Instructor	Dimensions (SHIFT + CTRL + D)	Opens the **Default Dimensions** form (540) for the selected entry. The link should run whenever the form is updated.

Dimensions in Registration

The next step in enabling dimension functionality for the seminar module is to set up the seminar registrations and posted registrations to accept dimensions as well. This will allow users to perform analyses on the registration data using the dimensions to group the information properly.

Solution Analysis

The goal is allowing the user to define dimensions for seminar registration headers and lines, which then flow to the dimensions for the posted registration entries.

When creating a seminar registration, the seminar managers enter dimension information for the header and lines. After posting the registration, the seminar managers can enter or change dimension information on the posted registration header and lines.

Solution Design

Transaction documents and lines use shortcut dimension codes. Enabling dimensions in transaction documents means setting up the shortcut dimension codes for the header and lines, and writing code so that the dimensions from the seminar, seminar room, and instructor chosen for the seminar registration are also included. This also enables dimensions on the posted seminar registration tables and forms.

GUI Design

User interface changes will need to be made to these forms. The **Seminar Registration** form and subform should be modified by adding the eight shortcut dimension code fields to the subform as shown in Figure 8-1, starting with the **Department Code** field. These fields should not be visible until the user selects them using Show Column.

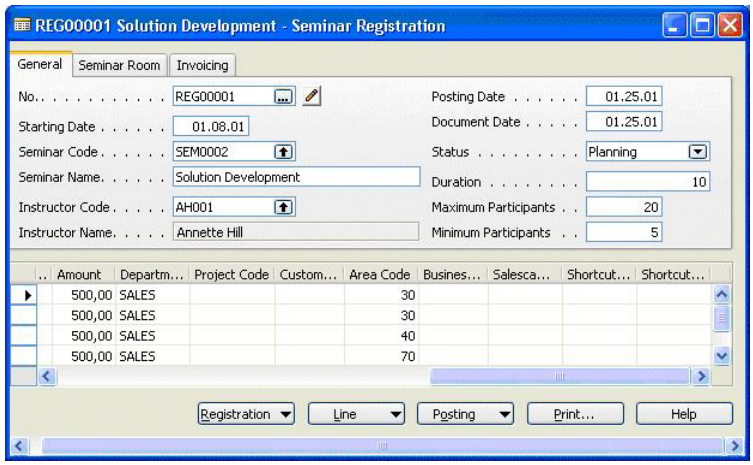

FIGURE 8–1: THE SEMINAR REGISTRATION FORM AND SUBFORM (123456710 AND 123456711)

The **Posted Seminar Registration** form and subform should be modified by adding two shortcut dimension code fields to the subform, which are shown in Figure 8-2 as the **Project Code** and **Department Code** fields. These fields should not be visible until the user selects them using Show Column.

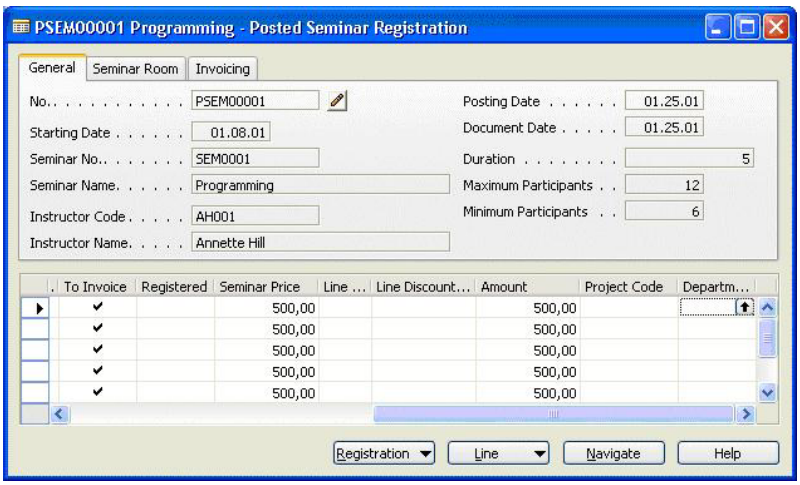

FIGURE 8–2: THE POSTED SEMINAR REGISTRATION FORM AND SUBFORM (123456734 AND 123456735)

Functional changes will need to be made to the **Seminar Registration Header** and **Seminar Registration Line** tables to get default dimensions, validate dimensions, insert dimensions, and delete dimensions. The standard functions from the **DimensionManagement** codeunit can also be used to do most of the real work, but functions are required to call these standard functions.

ValidateShortcutDimCode: As with the master files, create this function to validate and save the dimensions, using the **SaveDefaultDim** function from the **DimensionManagement** codeunit.

CreateDim: For each of the tables mentioned previously, this function should get the default dimensions for certain key dimension fields by running the **GetDefaultDim** function from the **DimensionManagement** codeunit. How many and for which fields the function retrieves default dimensions depends on the table. For the **Seminar Registration Header**, these fields will be **Seminar Code**, **Instructor Code**, **Room Code**, and **Job No.** For the **Seminar Registration Line**, there will be only one field: **Bill-To Customer No**.

Changes will be made to these tables:

- Table 123456710 Seminar Registration Header
- Table 123456711 Seminar Registration Line
- Table 123456718 Posted Seminar Reg. Header
- Table 123456719 Posted Seminar Reg. Line

Also, the code in the **DocumentDimension** table (357) will need to be changed so that the table can accept entries from the seminar registrations.

Lab 8.3 - Modifying the Tables for Dimensions in Seminar Registrations

Begin by adding the necessary fields and code to the seminar registration table:

1. Add the shortcut dimension fields to the **Seminar Registration Header** table (123456710) as follows:

No.	Field Name	Type	Length	Comments
35	Shortcut Dimension 1 Code	Code	20	Caption Class is 1,2,1. Relation to the **Code** field on the Dimension Value table, where the Global Dimension No. is 1.
36	Shortcut Dimension 2 Code	Code	20	Caption Class is 1,2,2. Relation to the **Code** field on the Dimension Value table, where the Global Dimension No. is 2.

2. Create a new function for the table called **ValidateShortcutDimCode**. This function takes two parameters: an integer called **FieldNumber** and a code variable with a length of 20 called **ShortcutDimCode**, which is passed by reference.
3. Enter code in the function trigger so that it runs the **ValidateDimValueCode** function from the **DimensionManagement** codeunit. If the **No.** is not blank, the function runs the **SaveDocDim** function from the **DimensionManagement** codeunit; otherwise, it runs the **SaveTempDim** function.
4. Enter code in the appropriate triggers so that when the user enters or changes a value in the new **Shortcut Dimension Code** fields, the program runs the **ValidateShortcutDimCode** function (the **FieldNumber** values are 1 and 2, respectively).

For seminar registrations, get the default dimensions from the **Seminar**, **Job**, **Seminar Room**, and **Instructor** that the user assigns to the registration and assign these to the registration's Document Dimensions. Create a **CreateDim** function to do this.

5. Create a new function called **CreateDim** in the **Seminar Registration Header** table. This function takes eight parameters:
   - Four integers called Type1, Type2, Type3 and Type 4.
   - Four code variables of length 20 called No1, No2, No3 and No4. The order of the parameters is Type1, No1, Type2, No2, and so forth.

6. In C/AL Locals, define an integer array variable named **TableID** with 10 dimensions and a code array variable named **No** of length 20 with dimensions. Only four of the ten dimensions will be used, but ten must be specified due to the parameters of the **GetDefaultDim** function in the **Dimension Management** codeunit.

7. Enter code in the **CreateDim** function trigger so that the function assigns the Type and No parameters to corresponding dimensions of the TableID and No variables. The function then clears both **Shortcut Dimension Code** fields and runs the **GetDefaultDim** function from the **DimensionManagement** codeunit. If the **No.** is not blank, the function runs the **UpdateDocDefaultDim** function of the **DimensionManagement** codeunit.

Now use the **CreateDim** function to assign the default dimensions from the **Seminar**, **Job**, **Seminar Room**, and **Instructor** whenever default dimensions are entered or changed.

8. Enter code in the appropriate triggers so that when the user changes the **Seminar Code**, **Instructor Code**, **Room Code**, or **Job No**. fields, the program runs the GetDimensions function of the Document Dimension table, runs the CreateDim function for the **Seminar Code**, **Instructor Code**, **Room Code**, and **Job No**., and runs the UpdateAllLineDim function from the Document Dimension table.

Next, add code to ensure that when the user creates or deletes a seminar registration, the program automatically creates or deletes dimensions as appropriate.

9. Enter code in the appropriate trigger so that when the user creates a new **Seminar Registration Header**, the program runs the **InsertDocDim** function from the **DimensionManagement** codeunit.

10. Enter code in the appropriate trigger so that when the user deletes a **Seminar Registration Header**, the program deletes all associated dimensions by running the **DeleteDocDim** function from the DimensionManagement codeunit.

The next modifications are to the **Seminar Registration Line** table.

11. Add the shortcut dimension fields to the **Seminar Registration Line** table (123456711) as follows:

No.	Field Name	Type	Length	Comments
15	Shortcut Dimension 1 Code	Code	20	Caption Class is 1,2,1. Relation to the **Code** field on the Dimension Value table, where the Global Dimension No. is 1.
16	Shortcut Dimension 2 Code	Code	20	Caption Class is 1,2,2. Relation to the **Code** field on the Dimension Value table, where the Global Dimension No. is 2.

12. Create a new function for table 123456711 called **ValidateShortcutDimCode**. As for the **ValidateShortcutDimCode** function in **the Seminar Registration Header**, add parameters and code so that this function validates and saves the document dimensions for the dimension field passed to it.

13. Enter code in the appropriate triggers so that when the user enters or changes a value in the **Shortcut Dimension Code** fields, the program runs the **ValidateShortcutDim** function (the **FieldNumber** values are 1 and 2, respectively).

For seminar registration lines, get the default dimensions from the **Seminar Registration Header**. Create a **CreateDim** function to do this.

14. Create a new function called **CreateDim** in the **Seminar Registration Line** table. As mentioned in the functional design, the **Seminar Registration Line** table has one key dimension field from which the default dimensions **Bill-To Customer No** can be retrived**.**. The **CreateDim** function is similar to the **CreateDim** function in the **Seminar Registration Header**, except that it takes only two parameters:

    – An integer called **Type1**
    – A code variable of length 20 called **No1**

15. Enter code in the function trigger so that the function assigns the parameters to a corresponding local integer variable and local code variable. The function clears both **Shortcut Dimension Code** fields and runs the **GetPreviousDefaultDocDim** and **GetDefaultDim** functions from the **DimensionManagement** codeunit. If the **Line No.** is not 0, the function runs the **UpdateDocDefaultDim** function of the **DimensionManagement** codeunit.

16. Enter code in the appropriate trigger so that when the user enters or changes the **Bill-To Customer No**., the program runs the **CreateDim** function, passing the **Customer** table number as the first parameter.

17. Now create two functions, **LookupShortcutDimCode** and **ShowShortcutDimCode**. The **LookupShortcutDimCode** function takes two parameters: an integer called **FieldNumber** and a code variable of length 20 called **ShortcutDimCode**, which is passed by reference.

18. Enter code in the **LookupShortcutDimCode** function trigger so that the function runs the **LookupDimValueCode** function from the **DimensionManagement** codeunit. If the **Line No.** of the record is not 0, the function runs the **SaveDocDim** function from the **DimensionManagement** codeunit; otherwise, it runs the **SaveTempDim** function of the **DimensionManagement** codeunit.

19. Create a new function called **ShowShortcutDimCode** that takes one parameter: a code variable of length 20 that is passed by reference, called **ShortcutDimCode**. This parameter is an array of eight dimensions.

20. Enter code in the **ShowShortcutDimCode** function trigger so that if the **Line No.** of the record is not 0, the function runs the **ShowDocDim** function of the **DimensionManagement** codeunit; otherwise, it runs the **ShowTempDim** function of the **DimensionManagement** codeunit.

21. Next a function that displays the line dimensions requested by the user is needed. Create a new function called **ShowDimensions**.

22. Enter code in the **ShowDimensions** function trigger so that the function makes sure there is a **Document No.** and **Line No.**, and then shows the corresponding lines from the **Document Dimension** table in the **Document Dimensions** form.

***HINT**: After you have filtered the Document Dimension table to the appropriate records, use the SETTABLEVIEW function for the Document Dimensions form before running it.*

As in the **Seminar Registration Header** table, ensure that when the user creates or deletes a **Seminar Registration Line**, the program automatically handles the associated dimensions appropriately.

23. Enter code in the appropriate triggers so that when the user creates a **new Seminar Registration Line**, the program runs the **InsertDocDim** function from the **DimensionManagement** codeunit, and when the user deletes a record, the program runs the **DeleteDocDim** function from the **DimensionManagement** codeunit.

24. Lastly, before finishing with the tables, modify the **Document Dimension** table so that it can accept dimensions from the seminar registrations. Define local record variables for the **UpdateGlobalDim** function of the Document Dimension table (357), for the **Seminar Registration Header** and **Seminar Registration Line** tables. Set the IDs for these variables to 123456700 and 123456701, respectively.

25. Add to the existing code in the **UpdateGlobalDim** function trigger. Do this so that for each of the three record variables just added (if the TableID corresponds to the table ID for that table), the program gets the corresponding record. This sets either the Shortcut Dimension 1 Code or the Shortcut Dimension 2 Code (depending on the **GlobalDimCodeNo** value passed to the function) to the **NewDimValue**, and modifies the record.

Modify the **UpdateLineDim** function so that if the user changes a dimension on the **Seminar Registration Header**, the function changes the dimensions on any associated **Seminar Registration Line**.

26. In the **Document Dimension** table (357), modify the **UpdateLineDim** function so that if the Table ID is for the **Seminar Registration Header**, the function filters the document dimensions to those that have a Table ID for the **Seminar Registration Line** table. The function then looks for any associated **Seminar Registration Line** records. If it finds any, and the user wants to update those lines, the function deletes the old dimensions and inserts the new updated **Document Dimension** record passed to it using the **InsertNew** function.

***HINT**: Look at how the UpdateLineDim function handles the case where the **Table ID** is Sales Header or Purchase Header.*

---

27. Modify the **UpdateAllLineDim** function in a similar way so that if the **TableNo** passed to it is for the **Seminar Registration Header**, the function updates the dimensions for associated **Seminar Registration Line** records.

Now that the seminar registration tables are ready for dimensions, dimensions must be enabled on the forms.

Lab 8.4 - Modifying the Forms for Dimensions in Seminar Registrations

Define functions in the forms to make it possible to show all the dimensions for a registration header or line, rather than just those defined as the global or shortcut dimensions on the tables:

1. In the **Seminar Registration** subform (123456711), create a global code variable **ShortcutDimCode** with a length of 20. Define this variable as an array of eight dimensions.
2. Define a new function for the form called **ShowDimensions** for the form. Enter code in the function trigger so that this function simply runs the **ShowDimensions** function for the current record.
3. Define a new function called **UpdateForm** that takes a parameter of a Boolean variable called **SetSaveRecord**. Enter code in the function trigger so that this function updates the current form, running the form update trigger code if **SetSaveRecord** is TRUE.
4. Enter code in the appropriate triggers so that the program performs the following:
   - When the program gets a record for the form, or the user enters or changes the **Bill-To Customer No.**, the program updates the **ShortcutDimCode** variable with the document dimensions by calling the **ShowShortcutDimCode** function.
   - After the program gets the record, it updates the controls on the form.
   - When the user goes to a new record, it clears the **ShortcutDimCode** variable.
5. Add the eight **Shortcut Dimension Code** fields to the form as shown in the GUI Design section. First, add the shortcut **Dimension 1 Code** and **Shortcut Dimension 2 Code** fields to the form and then add six text boxes. The caption classes for these six text boxes are '1,2,x', where x is the respective number of the field (three through eight). The sources for these fields are the respective dimensions of the **ShortcutDimCode** variable (also three through eight). For each of these fields, enter code so that when the user enters or changes a value, the program runs the **ValidateShortcutDimCode** function for the corresponding **ShortcutDimCode** dimension value. Also, enter code so that when the user performs a lookup on the field, the program runs the **LookupShortcutDimCode** function for the corresponding **ShortcutDimCode** dimension value.

6. In the **Seminar Registration** form (123456710), set the property so that the form is updated when it is activated.

7. Enter code in the appropriate trigger so that the form is updated when the user enters or changes a value in the **Seminar Code**, **Instructor Code**, **Room Code**, or **Job No**. fields.

8. Add the menu button and menu items to the **Seminar Registration** form (123456710) as shown in the following table. Add code to the appropriate trigger so that when the user selects the **Dimensions** menu item, the program runs the **ShowDimensions** function of the subform.

Menu Button	Options	Comments
Registration	Dimensions	Opens the **Document Dimensions** form 546 for selected No., where the Line No. is 0. The link should run whenever the form is updated.
Line	Dimensions (CTRL + SHIFT + D)	Runs code in the **OnPush** trigger to show the line.

Lab 8.5 - Modifying the Tables and Forms for Dimensions in Posted Seminar Registrations

Now modify the posted registration tables and forms to enable dimensions. The user cannot enter or change dimensions for posted documents, so simply make the posted dimensions available for the user to view. Follow these steps:

1. Add the shortcut dimension fields to the **Posted Seminar Reg. Header** table (123456718) as follows:

No.	Field Name	Type	Length	Comments
35	Shortcut Dimension 1 Code	Code	20	Caption Class is 1,2,1. Relation to the **Code** field on the Dimension Value table, where the Global Dimension No. is 1.
36	Shortcut Dimension 2 Code	Code	20	Caption Class is 1,2,2. Relation to the **Code** field on the Dimension Value table, where the Global Dimension No. is 2.

2. Enter code in the appropriate trigger so that when the user deletes a **Posted Seminar Reg. Header**, the program deletes the associated posted document dimensions by calling the **DeletePostedDocDim** function from the **DimensionManagement** codeunit.

3. Add the shortcut dimension fields to the **Posted Seminar Reg. Line** table (123456719) as follows:

No.	Field Name	Type	Length	Comments
15	Shortcut Dimension 1 Code	Code	20	Caption Class is 1,2,1. Relation to the **Code** field on the Dimension Value table, where the Global Dimension No. is 1.
16	Shortcut Dimension 2 Code	Code	20	Caption Class is 1,2,2. Relation to the **Code** field on the **Dimension Value** table, where the **Global Dimension No.** is 2.

4. Create a new function called **ShowDimensions**. Enter code in the function trigger so that the function tests that a **Document No**. and **Line No**. exist. If so, the function filters the **Posted Document Dimension** table to the records corresponding to the **Posted Seminar Reg. Line** and shows those records in the **Posted Document Dimension** form.

5. As for the **Posted Seminar Reg. Header** table, enter code in the appropriate trigger so that when the user deletes a **Posted Seminar Reg. Line** record, the program deletes the associated **Posted Document Dimension** records.
6. In the **Posted Seminar Reg.** subform (123456735), create a function called **ShowDimensions**. Enter code in the function trigger so that it runs the **ShowDimensions** function of the current record.
7. Add the text boxes and labels for the new **Shortcut Dimension Code** fields as shown in the GUI Design section.
8. In the **Posted Seminar Registration** form 123456734, set the property to specify that the form should be updated when activated.
9. Add the new **Dimensions** menu item to the **Registration** menu button and the new **Line** menu button as follows:

Menu Button	Options	Comments
Registration	Dimensions	Opens the **Posted Document Dimensions** form (547) for the selected **No**., where the **Line No**. is 0. The link should run whenever the form is updated.
Line	Dimensions (CTRL + SHIFT + D)	Runs code in the **OnPush** trigger to show the line dimensions (runs the **ShowDimensions** function of the **Posted Seminar Reg.** subform).

Dimensions in Seminar Posting

In this section dimensions functionality is added to the seminar posting portion of the Seminar module.

Solution Analysis

Now that dimension functionality has been added to non-posted and posted seminar registration tables and forms, the next step is adding a means of bringing dimensions from seminar registrations to posted seminar registrations. Ledger entries also need dimensions.

Adding this functionality will mean that when seminar managers post seminar registrations, the dimensions defined for those registrations are automatically transferred to the posted seminar registrations by the posting process. Furthermore, the global dimensions are transferred to the seminar ledger entries.

Solution Design

To enable the transfer of dimensions in the posting process, add dimension fields to the journal and ledger entry tables. Then add the code to transfer the dimensions from seminar registration headers and lines to the posted registration headers and lines.

GUI Design

The only modifications necessary to the user interface are to the **Seminar Ledger Entries** form. To accommodate dimensions, global dimension code fields are needed. These fields appear in Figure 8-3 as the **Department Code** and **Project Code** fields.

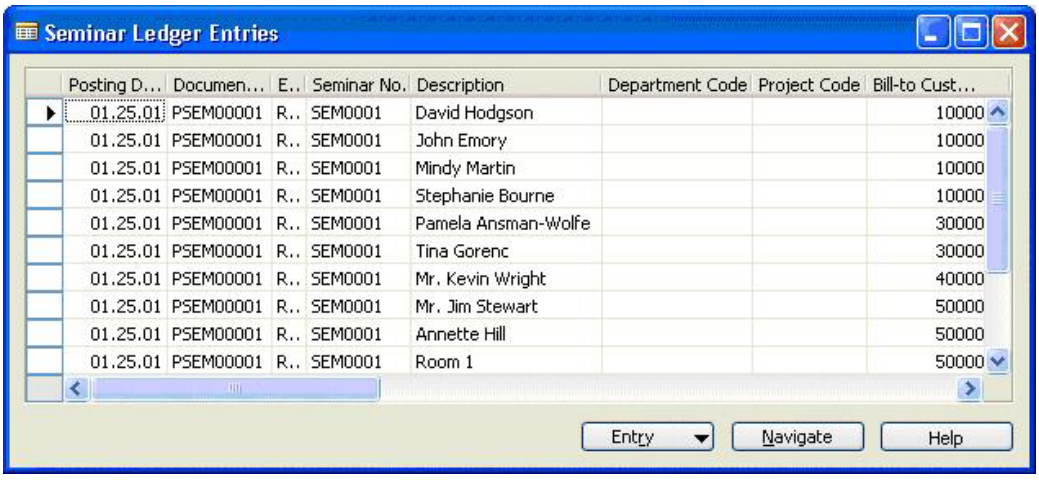

FIGURE 8–3: THE SEMINAR LEDGER ENTRIES FORM (123456721)

Functional Design

In the **Seminar Jnl.-Check Line** codeunit, add code to validate the dimensions in the **Seminar Journal Line**.

In the **Seminar Jnl.-Post Line** codeunit, add code to transfer the journal line dimensions to the ledger entry dimensions.

In the **Seminar-Post** codeunit, add code to transfer the dimensions from the **Document Dimension** records to the **Posted Document Dimension**. For the **Seminar Registration Header**, the program should move the **Seminar Code**, **Job No**., **Room Code**, and **Instructor Code** dimensions. For the **Seminar Registration Line**, the program should move the **Bill-To Customer No**. dimension.

Table Design

Two changes need to be made to the tables. A **Shortcut Dimensions** field needs to be added to both the **Seminar Journal Line** table (123456731) and the **Seminar Ledger Entry** table (123456732).

Lab 8.6 – Modifying the Tables, Codeunits and Forms for Dimensions in Seminar Posting

Follow these steps to add dimensions support to the Seminar module objects:

1. Add the following fields to the **Seminar Journal Line** table (123456731):

No.	Field Name	Type	Length	Comments
25	Shortcut Dimension 1 Code	Code	20	Caption Class is 1,2,1. Relation to the **Code** field on the Dimension Value table, where the Global Dimension No. is 1.
26	Shortcut Dimension 2 Code	Code	20	Caption Class is 1,2,2. Relation to the **Code** field on the Dimension Value table, where the Global Dimension No. is 2.

2. Add the following fields to the **Seminar Ledger Entry** table (123456732):

No.	Field Name	Type	Length	Comments
31	Global Dimension 1 Code	Code	20	Caption Class is 1,2,1. Relation to the **Code** field on the Dimension Value table, where the Global Dimension No. is 1.
32	Global Dimension 2 Code	Code	20	Caption Class is 1,2,2. Relation to the **Code** field on the Dimension Value table, where the Global Dimension No. is 2.

With the necessary fields in place, make the changes to the three main seminar posting codeunits. Make the first changes to the **Seminar Jnl.-Check Line** codeunit.

Before checking the lines, copy the shortcut dimension values from the **Seminar Journal Line** to a **Journal Line Dimension** record.

3. Delete the code in the **OnRun** trigger of the **Seminar Jnl.-Check Line** codeunit 123456731 .

4. Enter code in the **OnRun** trigger so that if the Shortcut Dimension 1 Code in the **Seminar Journal Line** record is not empty, the program copies the **Seminar Journal Line table ID** and **Line No.** to a temporary **Journal Line Dimension** record variable. The program copies the **Global Dimension 1 Code** value from the G/L Setup to the **Dimension Code** field and the **Shortcut Dimension 1 Code** value from the **Seminar Journal Line** to the **Dimension Value Code** on the **Journal Line Dimension** record. The program then inserts the **Journal Line Dimension** record. Enter code to do the same with a new **Journal Line Dimension** record for the **Shortcut Dimension 2 Code** field on the **Seminar Journal Line**.

5. Add code in the **OnRun** trigger to run the **RunCheck** function with two parameters: the current record and the **Journal Line Dimension** record just created.

6. Add a parameter (passed by reference) of a record variable for the **Journal Line Dimension** table.

7. Enter code in the function trigger so that after the function checks the **Document Date**, it loads the table IDs of the **Seminar**, **Job**, **Seminar Room**, **Instructor**, and **Customer** tables into a local integer array variable (with ten dimensions). The function then loads the **Seminar Code**, **Job No**., **Room Code**, **Instructor Code**, and **Bill-To Customer No**. values into the corresponding dimensions of a local code array variable (with 10 dimensions). Using these two array variables, the function checks the dimensions by running the **CheckJnlLineDimValuePosting** function from the **DimensionManagement** codeunit. If this function returns false, the function shows an error. If the **Line No**. is not 0, this can be a specific error to help the user see exactly where the error occurred.

HINT: *You can use the GetDimValuePostingErr function from the DimensionManagement codeunit to help create the error message.*

Now modify the **Seminar Jnl.-Post Line** codeunit (123456732) to transfer the dimensions to the ledger entries.

8. In the **OnRun** trigger, enter the code entered into the **OnRun** trigger of the **Seminar Jnl.-Check Line** codeunit. This code loads and inserts **two Journal Line Dimension** records with information from the **Shortcut Dimension Code** fields into a temporary global **Journal Line Dimension** record variable. At the end of the code, call the **RunWithCheck** function with two parameters: the current record and the temporary **Journal Line Dimension** record variable.

9. As with the **Seminar Jnl.-Check Line** codeunit, add a parameter (passed by reference) to the **RunWithCheck** function. The parameter is a record variable of the **Journal Line Dimension** table.

10. Enter code in the **RunWithCheck** function trigger so that after the function performs the first copy of the **Seminar Journal Line** record, the function resets and deletes all records in the original global **Journal Line Dimension** variable. The function then copies the records from the **Journal Line Dimension** parameter variable to the global **Journal Line Dimension** variable. It does so by calling the **CopyJnlLineDimToJnlLineDim** function of the **DimensionManagement** codeunit.

11. In the **Code** function trigger, change the call of the **RunCheck** function of the **Seminar Jnl.-Check Line** codeunit so that it calls the function with the two parameters instead of only one.

12. In the **Code** function trigger, be sure that the journal line dimensions are moved to the new ledger entries' dimensions. The **DimensionManagement** codeunit makes this easy with a function called **MoveJnlLineDimToLedgEntryDim**. Enter code to call this function after the **Seminar Ledger Entry** record is inserted.

Now add code to the **Seminar-Post** codeunit (123456700) to move the document dimensions to the posted document dimensions.

13. Create a new local function called **CheckDimValuePosting** that takes four parameters: an integer variable called **TableID** and two record variables that are passed by reference for **the Seminar Registration Header** and **Seminar Registration Line**.

14. Enter code in the function trigger so that depending on the table ID sent to the function as the **TableID** parameter, the function loads an integer array variable (with ten dimensions) and a code array variable (of length 20 with 10 dimensions) with the table IDs and field values of the key dimension fields for the table. In each case, the function calls the **CheckDocDimValuePosting** function of the **DimensionManagement** codeunit, and shows an error if it returns FALSE.

15. Create a new local function called **CheckDimComb** that takes two parameters: an integer called **TableID** and an integer called **LineNo**.

16. Enter code in the function trigger so that if the **CheckDocDimComb** function of the **DimensionManagement** codeunit returns FALSE when run with the document dimensions, the function shows one of three errors: one error if the **Line No.** is 0, or one of the other two errors depending on whether the **TableID** is for the **Seminar Registration Header** or **Seminar Registration Line**.

Now create a function that checks the document dimensions and passes them to a temporary **TempDocDim** variable.

17. Create a new local function called **CopyAndCheckDocDimToTempDocDim**.

18. Enter code in the function trigger so that the function copies the appropriate records from the **Document Dimension** table to the **TempDocDim** variable. The function runs the **CheckDimComb** function for the **Seminar Registration Header** table ID and **Line No**. 0. The function then runs the **CheckDimValuePosting** function. If there are associated **Seminar Registration Line** records, the function runs the **CheckDimComb** and **CheckDimValuePosting** functions for each of the records.

19. Enter code in the **OnRun** trigger so that after testing the **Resource No.** field of the Instructor record, the program runs the **CopyAndCheckDocDimToTempDocDim** function.

20. Enter code in the **OnRun** trigger so that after **the Posted Seminar Reg. Header** is inserted, the program copies the document dimensions to the posted document dimensions by calling the **MoveOneDocDimToPostedDocDim** function from the **DimensionManagement** codeunit for the **Seminar Registration Header** dimensions in the **DocDim** variable. Do the same for the **Seminar Registration Line** dimensions after the **Posted Seminar Reg. Line** record is inserted.

In addition to posting the document dimensions to the **Seminar Ledger Entry** records, post the dimensions to the **Job Ledger Entry** records.

21. Enter code in the **PostJobJnlLine** function trigger so that the function performs the following tasks:

    – After copying the relevant fields to the **Job Journal Line** record when the **ChargeType** is **Participant**, the function copies the records for the corresponding **Seminar Registration Line** from the **TempDocDim** to a **TempJnlLineDim** variable. It does so by calling the **CopyDocDimToJnlLineDim** function from the **DimensionManagement** codeunit.
    – Further on in the function, instead of running the **Job Jnl.-Post Line** codeunit, the function runs the **RunWithCheck** function of the **Job Jnl.-Post Line** codeunit.

Now add similar code to the **PostSeminarJnlLine** function.

22. As done previously in the **PostJobJnlLine** function, enter code in the **PostSeminarJnlLine** function trigger so that depending on the **ChargeType**, the function copies the document dimensions for a corresponding record to a **TempJnlLineDim** variable, using the following guidelines:
    - For a **ChargeType** of **Instructor** or **Room**, the corresponding record is the **Seminar Registration Header**.
    - For a **ChargeType** of **Participant**, the corresponding record is the **Seminar Registration Line**.
    - Instead of running the **Seminar Jnl.-Post Line** codeunit, the function runs the **RunWithCheck** function of the **Seminar Jnl.-Post Line** codeunit.

The codeunit modifications are complete. All that remains is to make the dimensions visible in the **Seminar Ledger Entries** form.

23. Add the new **Global Dimension Code** fields to the **Seminar Ledger Entries** form 123456721 as shown in the GUI Design section.

24. Add the new **Entry** menu button and **Dimensions** menu item as follows:

Menu Button	Options	Comments
Entry	Dimensions (SHIFT+CTRL+D)	Opens the **Ledger Entry Dimensions** form (544) for the selected entry. The link should run whenever the form is updated.

Dimensions in Invoicing

In this section the invoicing feature of the Seminar module is enhanced to support dimensions.

Solution Analysis

When seminar managers create and post the sales invoices for seminar registrations, the invoices should automatically be created, and the dimensions defined for those registrations should be transferred. This can be accomplished by carrying the associated dimensions to the invoices when invoices are created, allowing the invoice lines to be analyzed by dimension.

Solution Design

To implement dimensions for invoicing, the **Create Seminar Invoices** report will be modified to enable the transfer of dimensions when invoices are created. Every **Seminar Ledger Entry** record used to create a sales invoice should be transferred from the corresponding **Ledger Entry Dimension** records to new **Document** Dimension records.

Lab 8.7 - Modifying the Create Seminar Invoices Report for Dimensions

Follow these steps to implement dimensions support for seminar invoices:

1. In the **Create Seminar Invoices** report (123456700), create two new local record variables: one for the **Document Dimension** table and one for the **Ledger Entry Dimension** table.
2. Enter code in the appropriate trigger so that after the program gets the **Seminar Ledger Entry** record and inserts the **Sales Line**, it deletes all **Document Dimension** records that correspond to the **Sales Line**. Then, for every **Ledger Entry Dimension** record that corresponds to the **Job Ledger Entry**, the program should create a new **Document Dimension** record for the **Sales Line** with the **Dimension Code** and **Dimension Value Code** of the **Ledger Entry Dimension** record.

Testing

Use this test script to verify that the implementations in this chapter are complete and correct:

1. Open the **Seminar Card** and view a seminar. Select **Dimensions** from the **Seminar** menu button and enter at least one dimension for this seminar.
2. Open the **Seminar Room Card** and view a room. Select **Dimensions** from the **Seminar** menu button and enter at least one dimension for this seminar room.
3. Open the **Instructors List** and select an instructor. Select **Dimensions** from the **Instructor** menu button and enter at least one dimension for this instructor.
4. Open the **Seminar Registration** form and create a new registration using the seminar, room, and instructor defined for dimensions. Because you set the line dimensions invisible, you have to use the **Show Column** feature to see the dimension fields in the subform.
5. View the header dimensions by selecting the **Dimensions** menu item from the **Registration** menu button. Verify that the values are as expected based on the values from the master records. View the line dimensions by selecting a line and then selecting **Dimensions** from the **Line** menu button. Verify that the values as well.
6. Post the registration and open the **Posted Seminar Registration** form. Verify that the dimensions are the same as those specified before posting.
7. Open the **Seminar Ledger Entries** form and view the entries for the registration just posted. Use the **Show Column** feature to see the dimension columns and verify that they are what was specified.

Conclusion

In this chapter, the existing seminar module objects were modified so that dimensions can be defined and carried throughout the module.

Test Your Knowledge – Dimensions

Review Questions

1. What is a dimension and what is its purpose in Microsoft Dynamics NAV?

2. In what tables are the dimensions for the following entry types stored:
   - "Master" data entries (like Customer or Seminar)
   - Transaction document header and lines (like Sales Line or Seminar Registration Header)
   - Journal lines
   - Ledger entries
   - Posted documents

3. Describe the flow of information and how dimensions are transferred from the master data through transaction documents to ledger entries and posted documents.

Quick Interaction: Lessons Learned

Take a moment to write down three Key Points you have learned from this chapter:

1.

2.

3.