

MODULE 2: MASTER TABLES AND PAGES

Module Overview

Master tables contain key business information about subjects and objects of business processes. When you develop solutions, you typically first develop the master tables, because all transactional and detailed information in a application area depends on these tables. Master tables are very detailed and typically contain many fields. These fields describe all properties and characteristics of a subject, such as a customer or vendor, or an object, such as an item or a fixed asset, upon which the processes of a application area in Microsoft Dynamics NAV® 2013 depend.

Users cannot view or enter information directly in tables. This means that you must provide other objects that enable users to view and manage this data. In Microsoft Dynamics NAV 2013, *page* is the object type that provides this functionality. For every master table, you also have to provide at least two pages that are required for the basic process of managing the master data. These pages are a *card page* and a *list page*.

Master tables do not include only fields, but also some necessary C/AL logic that executes when certain system events occur. Some of this logic is mandated by Microsoft Dynamics NAV user experience standards, whereas some logic completely depends on the business requirements and processes that are relevant for the users. During development of the master tables and corresponding pages, you must make sure that you include all the necessary C/AL code in the relevant table or page triggers.

In this chapter, you develop the master tables and pages for the Seminar Management module, and make sure that they satisfy your customer's requirements.

Objectives

The objectives are:

- Explain the master table and page standards.
- Work with table event triggers.
- Work with the complex data types and their member functions.
- Explain multilanguage functionality.
- Define the strategy for implementing customers and participants,
- Create tables to manage the seminar rooms.
- Create instructor data management.
- Create seminar data management.

Prerequisite Knowledge

Before you design and develop the solution for managing master data, you must review Microsoft Dynamics NAV 2013 standard functionality. In each module, this "Prerequisite Knowledge" lesson presents Microsoft Dynamics NAV 2013 standards and principles that you can apply while customizing the solution for CRONUS International Ltd.

Triggers

Triggers are methods that execute when a system event occurs or when they are invoked by code. Following are two types of *active* triggers (triggers that contain executable code):

- Event Triggers have names that begin with "On..." and execute when specific events occur. For example, the OnInsert trigger executes when a user inserts a record in a table.
- Function Triggers are custom triggers that developers define. There are many function triggers that are defined as a part of the base application. You can add many more as part of your customization. These triggers execute when they are called by other C/AL code.

The *Documentation Trigger* is the third type of trigger. There is only one Documentation trigger per object, and anything that is written in it is not processed by the application, even if it contains C/AL code. It typically contains free-form documentation. This trigger lets you document the changes that you have made to objects.

This chapter demonstrates how to use these triggers in various scenarios.

Multi-Language Support

Microsoft Dynamics NAV 2013 is a multilanguage application. This means that a localized version of Microsoft Dynamics NAV 2013 can present itself in different languages. In other words, two or more RoleTailored clients that are connected to the same database can present their user interface in two or more languages at the same time. Users can change the language at any time, and the change is applied immediately.

Before you start to develop in a multilanguage-enabled database, set the working language to English (United States). To do this, follow this procedure:

1. On the **Tools** menu, click **Language**.
2. Select English (United States).
3. Click **OK**.

To enable multilanguage functionality for your custom solutions, use the following guidelines:

- Always define the Name property of an object, a control, or a table field in English (United States), or ENU. You must make sure that this is never visible to the user, by specifying the Caption property.
- Always provide language-specific names in the CaptionML property of the objects, controls, or table fields.



Note: A property that ends in ML means multilanguage. Make sure that you always set properties to take advantage of the multilanguage functionality of Microsoft Dynamics NAV 2013. Examples of the Multilanguage properties are CaptionML, OptionCaptionML and InstructionalTextML.

Participants

The core business of CRONUS International Ltd. is organizing and delivering seminars. The Seminar Management solution must enable users to schedule seminars and enroll participants in the seminars. Very frequently, companies send several employees to participate in seminars, and those companies pay for their employees' participation. Sometimes, participants are not related to a specific company, and they pay for seminar participation themselves.

Solution Design

CRONUS International Ltd. core functional requirements describe participants as persons who participate in seminars. For each participant, the solution must track the name, address, and contact details including at minimum, a telephone number and an email address. This means that the solution must give users a way to keep track of participants. This information is used for invoice processing.

One of the principles of solution design in Microsoft Dynamics NAV 2013 is to use standard functionality as much as you can. One of the customer's nonfunctional requirements also emphasizes this principle whenever possible. The solution must not duplicate functions that are already present in the application or cause redundant functionality.

C/SIDE Solution Development in Microsoft Dynamics® NAV 2013

Therefore, you must make sure that you determine whether there is existing functionality in Microsoft Dynamics NAV 2013 that CRONUS International Ltd. can use to represent participants. Following is the minimum information that you must track:

- Name
- Address
- Phone Number
- Email Address

There are many tables in Microsoft Dynamics NAV 2013 that include this kind of information about an entity, You must determine whether there is a table that you can use, or if you must develop one.

Following are two candidate tables in Microsoft Dynamics NAV 2013 that both relate to sales and already contain all required fields and functionality:

- Table 18, Customer
- Table 5050, Contact

You may be able to use either of these tables to satisfy the requirement, and may not have to develop new tables. To make sure that you select the correct table, you must consider all requirements that may help you decide. There is a requirement that participants receive invoices for seminar participation. If participants pay for participation themselves, they receive the invoice. If a participant's company pays for participation, the company receives the invoice. This means that there is a relationship between companies and participants. You have to make sure that the solution reflects that relationship.

The following figure represents the relationship between participants and companies:

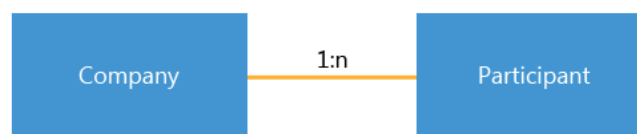


FIGURE 2.1: PARTICIPANTS AND COMPANIES RELATIONSHIP

Module 2: Master Tables and Pages

Because there are no specific statements that describe companies, you are free to select an existing relationship between tables that are already present in the application. The only guideline that limits you is the following two assumptions from the requirements:

- Participants are obviously persons.
- Companies are obviously business entities, and not persons.

This means that you must look for a set of two tables that meet the following criteria:

- There must be a table that represents a person.
- There must be a table that represents a business entity.
- There must be a clear relationship between the two tables, so that the person belongs to the business entity.

In Microsoft Dynamics NAV 2013 the table that represents business entities in Sales area, is table **18, Customer**. The table that represents persons is **5050, Contact**. There is an existing relationship that links contacts to customers.

This means that the best solution at this point is to map a customer's requirements to the Microsoft Dynamics NAV 2013 standard functionality in the following way.

Required Functionality	Microsoft Dynamics NAV 2013 Standard Functionality
Participant	Contact
Company	Customer

However, the requirements state that both participants and companies can receive invoices. In Microsoft Dynamics NAV 2013 you can send only invoices to customers, but not to contacts. This leaves you with two design options:

- Redesign invoicing functionality to enable contacts to receive invoices.
- For participants who pay for participation themselves, always use both the customer record for invoicing, and the contact record for participation registration.

The first design choice is not a good one because it introduces significant changes to standard functionality. This typically causes many regression issues and other types of bugs. It also makes the application more difficult to develop, maintain, and upgrade to a future version of Microsoft Dynamics NAV. Unless there are other options, this approach should always be your last choice.

The second design choice is not directly covered in requirements. However, it does not directly contradict them either. Even though users do not send invoices to participants directly, when participants pay for participation themselves, the participants are represented by customer records in addition to contact records. This does not introduce any changes into the standard functionality of Microsoft Dynamics NAV 2013. It also does not conflict or contradict the requirements.

It is best practice to use standard functionality as much as you can. However, customer requirements also explain that as long as there are no other requirements that contradict the proposed solution, any solution that proposes a standard functionality or the best practices of Microsoft Dynamics NAV 2013 is preferred.

Therefore, instead of customizing the sales area, you should opt for the second solution, and use the standard Microsoft Dynamics NAV 2013 functionality.



Note: *In this lesson, the design process and the decisions that you must make during that process are followed in detail. This lesson introduces you to the best practices of Microsoft Dynamics NAV 2013 solution design. In future lessons and chapters most of the decisions are already made.*

Development

After design decisions are made, you must develop the necessary tables and pages that represent the associated entities, and enable users to view and manage them.

Because you have decided to use the standard functionality of customers and contacts, and because both identified tables already have the necessary card and list pages that are typically required to maintain master data, you do not have to do any development.



Best Practice: *Do not try to change the names or captions of standard tables and pages to match your customer's terminology. Customers quickly adapt to Microsoft Dynamics NAV 2013 standards, and are not confused by a participant master record that is called Contact.*

Instructors and Rooms

The next step is to create tables to manage the instructors who teach the seminars, and tables for the seminar rooms in which the seminars are held. Your goal is to design the solution that meets the requirements, but also takes advantage of any standard Microsoft Dynamics NAV functionality.

Solution Design

The CRONUS International Ltd. requirements for the management of instructors explain that each seminar is taught by an instructor, who is either a CRONUS International Ltd. employee or a subcontractor. For subcontractors, the subcontracting rate must be tracked.

The requirements for the management of seminar rooms explain that each seminar is held in a seminar room. Some seminars are held in-house, and some are held off-site. If a seminar occurs in-house, a room must be assigned. For off-site rooms, the rental rate must be tracked. For all room types, the solution must track the maximum number of participants that the room can accommodate.

Another CRONUS International Ltd. requirement applies to both instructors and rooms. The solution must be aware of and provide the tools to manage and plan for the availability and the capacity of both rooms and instructors.

The instructor and room entities are obviously very similar and have many common requirements. Both share the following characteristics:

- They require the same level of information to describe them.
- They can be internal or external.
- If they are external, they require tracking of additional costs.
- They require a level of availability and capacity management.

The only difference between instructors and rooms is that rooms have to keep track of the maximum number of participants.

In Microsoft Dynamics NAV 2013, there is a resource management application area which closely resembles this functionality. It keeps track of people and equipment (machines), manages their capacity and availability, and keeps track of their costs or prices.

The following table describes the map between resources and instructors and rooms.

Seminar Management Requirement	Resource Management Functionality
Instructor/Room	Type field, with options Person and Machine
Maximum number of participants for rooms	
Availability and capacity management	Availability and capacity management
Internal/External	
Additional costs for external instructors or rooms	Resource costs/Resource prices

There are many functional similarities, and choosing to customize the resource management application area is a good design decision.



Note: Another design approach is to develop separate tables to keep track of instructors and rooms. However, that approach requires duplicating functionality between the standard solution and your custom solution. CRONUS International Ltd. functional requirements advise against this approach.

Development

The design specifies that the table **156, Resource** table represent the instructors and the rooms. Resources of type Person represent instructors, and resources of type Machine represent the rooms. You do not have to create any additional objects.

The following table summarizes the customizations that you must develop.

Object	Customization
Table 156, Resource	Add new fields: <ul style="list-style-type: none"> • Internal/External • Maximum Participants
Page 76, Resource Card	Add the Internal/External field to the General FastTab. After Personal Data , add the Room FastTab. In the Room FastTab, add the Maximum Participants field.

Module 2: Master Tables and Pages

Object	Customization
Page 77, Resource List	<p>After the Name field, add the Internal/External and Maximum Participant fields.</p> <p>Make sure that you hide the Type field if the page is filtered by Type.</p> <p>Make sure that you hide the Maximum Participant field if the page is filtered to show only instructors.</p>

The "Page 76, Resource Card" figure shows pages after development work is completed.

The following figure shows the page 76, **Resource Card**, after the customizations.

FIGURE 2.2: PAGE 76, RESOURCE CARD

The "Page 77, Resource List" figure shows the list after customization.

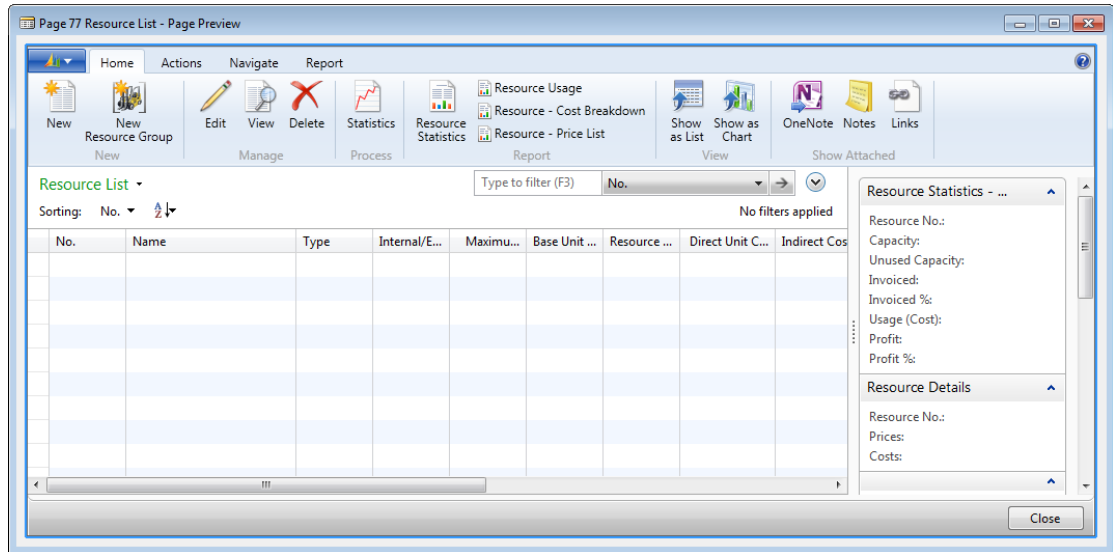


FIGURE 2.3: PAGE 77, RESOURCE LIST

Lab 2.1: Customize Resource Tables and Pages

Scenario

You are a developer who works on the implementation project of Microsoft Dynamics NAV 2013 at CRONUS International Ltd. You must customize the resource table to meet the requirements in the Functional Requirements Document for the project.

Exercise 1: Customize Resource Table

Exercise Scenario

Start by customizing the table to include fields that are specified in the design. These fields are **Internal/External** of type option, and **Maximum Participants** of type integer. You must make sure of the following:

- Your changes do not interfere with any future customizations your customer CRONUS International Ltd. or a third-party might make.
- You clearly document all changes that you make so that other developers can easily identify them.

Task 1: Add Fields to the Table

High Level Steps

1. Design the table **156, Resource**.
2. Add the fields **Internal/External**, of type option, and **Maximum Participants** of type integer.
3. Set the Caption, OptionString, and OptionCaption properties for the fields that you added in the previous step.
4. Save, and then compile the table.

Detailed Steps

1. Design the table **156, Resource**.
 - a. In **Object Designer**, click **Table**.
 - b. In the object list, find table **156, Resource**, and then click **Design** to open the **Table Designer**.
2. Add the fields **Internal/External**, of type option, and **Maximum Participants** of type integer.
 - a. At the end of the field list, add the following fields.

Field No.	Field Name	Data Type
123456701	Internal/External	Option
123456702	Maximum Participants	Integer

Field No.	Field Name	Data Type
123456703	Quantity Per Day	Decimal

3. Set the Caption, OptionString, and OptionCaption properties for the fields that you added in the previous step.
 - a. Select the **Internal/External** field.
 - b. Press SHIFT+F4 to open the **Properties** window.
 - c. In the Caption property, enter "Internal/External".
 - d. In the OptionString property, type "Internal,External". Make sure not to enter any spaces.
 - e. In the OptionCaption property, type "Internal,External".
 - f. Continue on the **Properties** window in **Table Designer**, and select the **Maximum Participants** field.
 - g. In the **Properties** window, in the Caption property, enter "Maximum Participants".
 - h. Close the **Properties** window.
4. Save, and then compile the table.
 - a. Click **File > Save**.
 - b. In the **Save** dialog box, make sure that **Compiled** is selected, and then click **OK**.

Task 2: Document the Changes

High Level Steps

1. Describe your changes in the **Documentation** trigger.
2. Set the Description property to indicate that the fields that you added are a part of the customization.
3. Append the version tag to the Version List property.

Detailed Steps

1. Describe your changes in the **Documentation** trigger.
 - a. Press **F9** to access **C/AL Editor**.
 - b. In the **Documentation** trigger, describe the changes that you made to the table.

For example, the description trigger may contain the following code:

Documentation Trigger

CSD1.00 - 2012-06-15 - D. E. Veloper

Chapter 2 - Lab 1

- Added new fields:

- Internal/External

- Maximum Participants

- c. Close the **C/AL Editor**.
2. Set the Description property to indicate that the fields that you added are a part of the customization.
 - a. In the **Table Designer**, in the **Description** column for the **Internal/External** and **Maximum Participants** fields, enter "CSD1.00".
 - b. Press CTRL+S to save the table.
 - c. In the **Save** dialog box, click **OK**.
 - d. Close **Table Designer**.
3. Append the version tag to the Version List property.
 - a. In **Object Designer**, make sure that table **156, Resource** is selected.
 - b. Select the Version List property, and press F2 to edit the value.
 - c. At the end of the version list, enter ",CSD1.00".

Exercise 2: Customize Resource Card

Exercise Scenario

After you add the fields to the table, you are ready to customize the pages. Each master table has two important pages that you must consider for customization every time that you add fields to a table: the card page and the list page.

You first customize the card page and add relevant fields and a FastTab, as specified in the design.



Note: Even though the steps to document the changes in the object are not the part of this exercise, make sure that you always document the changes to the documentation that you made to the table **156, Resource** in a similar manner.

Task 1: Add Controls to the Page

High Level Steps

1. Design page **76, Resource Card**.
2. Add the Internal/External and Quantity Per Day fields to the General FastTab.
3. Add the **Room** FastTab as the last FastTab in the page.
4. Add the **Maximum Participants** field to the **Room** FastTab.
5. Save, compile, and then close the page.

Detailed Steps

1. Design page **76, Resource Card**.
 - a. In **Object Designer**, click **Page**.
 - b. In the object list, find page **76, Resource Card**, then click **Design** to open the **Page Designer**.
2. Add the Internal/External and Quantity Per Day fields to the General FastTab.
 - a. Under the **General** group, position the cursor in the first row under the **Type** field.
 - b. Press F3 to insert a new row.
 - c. Click **View > Field Menu**.
 - d. In the **Field Menu** window, select the **Internal/External** field, and then click **OK**.
 - e. Under the **General** group, position the cursor in the first row under the **Base Unit of Measure** field.
 - f. Press F3 to insert a new row.
 - g. Click **View > Field Menu**.
 - h. In the **Field Menu** window, select the **Quantity per Day** field, and then click **OK**.
3. Add the **Room** FastTab as the last FastTab in the page.
 - a. In **Page Designer**, select the row of type Container, and subtype FactBoxArea.
 - b. Press F3 to insert a new row.
 - c. In the **Type** column, select **Group**.
 - d. Press SHIFT+ALT+LEFT to remove one level of indentation from the row.
 - e. In the **Caption** column, enter "Room".

4. Add the **Maximum Participants** field to the **Room** FastTab.
 - a. In **Page Designer**, select the row of type Container, and subtype FactBoxArea.
 - b. Press F3 to insert a new row.
 - c. Click **View > Field Menu**.
 - d. In the **Field Menu** window, select **Maximum Participants**, and then click **OK**.
5. Save, compile, and then close the page.
 - a. Click **File > Save**.
 - b. In the **Save** dialog box, make sure that the **Compiled** check box is selected, and then click **OK**.
 - c. Close **Page Designer**.

Exercise 3: Customize Resource List

Exercise Scenario

After you customize the **Resource Card** page, you are ready to customize the **Resource List** page. As specified in the design, you must add all of the fields that you added to the **Resource** table to this page.

List pages never include all of the fields from the underlying table, but only include sufficient information for users to quickly locate a record. Therefore, you must make sure that you exclude any irrelevant fields from the list tables. Only include those fields that users need.

Use the **Resource** table for both instructors and rooms. However, users do not want to see the combined list of instructors and rooms. Instead, they either want to see the list of instructors or the list of rooms. This indicates that any fields that are irrelevant to instructors or rooms should be hidden when the users view the list of instructors or rooms. You must write the code that achieves this goal.

Task 1: Add Variables

High Level Steps

1. Design page **77, Resource List**.
2. Add the following two Boolean variables: *ShowType*, and *ShowMaxParticipants*.
3. Set the variable properties to make sure that the variables are included in the page dataset.

Detailed Steps

1. Design page **77, Resource List**.
 - a. In **Object Designer**, click **Page**.
 - b. In the object list, find page **77, Resource List**, then click **Design** to open the **Page Designer**.
2. Add the following two Boolean variables: *ShowType*, and *ShowMaxParticipants*.
 - a. Click **View > C/AL Globals**.
 - b. Add the following variables.

Name	Data Type
ShowType	Boolean
ShowMaxParticipants	Boolean

3. Set the variable properties to make sure that the variables are included in the page dataset.
 - a. Select the **ShowType** variable.
 - b. Press SHIFT+F4 to open the **Properties** window.
 - c. Set the IncludeInDataset property to **Yes**.
 - d. In the **C/AL Globals** window, select the *ShowMaxParticipants* variable.
 - e. In the **Properties** window, set the IncludeInDataset property to **Yes**.
 - f. Close the **Properties** window.
 - g. Close the **C/AL Globals** window.

Task 2: Add Fields to the Page

High Level Steps

1. Under the Type field, add the Internal/External and Maximum Participants fields.
2. Set the Visible property expression of the **Type** field to **ShowType**.
3. Set the Visible property expression of the **Maximum Participants** field to *ShowMaxParticipants*.

Detailed Steps

1. Under the Type field, add the Internal/External and Maximum Participants fields.
 - a. In **Page Designer**, select the first field under the **Type** field.
 - b. Press F3 to insert a new row.

- c. Click **View > Field Menu**.
 - d. In the **Field Menu** window, select the **Internal/External** and **Maximum Participants** fields, and then click **OK**.
 - e. In the confirmation dialog box, click **Yes**.
2. Set the Visible property expression of the **Type** field to **ShowType**.
 - a. In **Page Designer**, select the **Type** field.
 - b. Press SHIFT+F4 to open the **Properties** window.
 - c. In the Visible property, type "ShowType".
 3. Set the Visible property expression of the **Maximum Participants** field to *ShowMaxParticipants*.
 - a. In **Page Designer**, select the **Maximum Participants** field.
 - b. In the **Properties** window, in the Visible property, enter "ShowMaxParticipants".
 - c. Close the **Properties** window.

Task 3: Add Code to the Page

High Level Steps

1. In the OnOpenPage trigger, add the code that sets the value of the *ShowType* and *ShowMaxParticipants* variables depending on the state of the filters in the view filter group. The *ShowType* variable must be *TRUE* if the list is not filtered on the **Type** field. The *ShowMaxParticipants* variable must be *TRUE* if the list is filtered to show the resources of type Machine.
2. Save, compile, and then close the page.

Detailed Steps

1. In the OnOpenPage trigger, add the code that sets the value of the *ShowType* and *ShowMaxParticipants* variables depending on the state of the filters in the view filter group. The *ShowType* variable must be *TRUE* if the list is not filtered on the **Type** field. The *ShowMaxParticipants* variable must be *TRUE* if the list is filtered to show the resources of type Machine.
 - a. In **Page Designer**, press F9 to access the **C/AL Editor**.
 - b. In the OnOpenPage trigger, add the comments block to indicate the changes that you are about to introduce.

The comments block may resemble the “Comments Block” example.

Comments block.

```
//CSD1.00>  
  
//CSD1.00<
```

- c. Within the comments block, add the code which sets the *ShowType* variable to *TRUE* if the page is filtered on the **Type** field.

The code may resemble the “Setting ShowType” example.

Setting ShowType

```
ShowType := GETFILTER(Type) = '';
```

- d. Add another line of code that sets the *ShowMaxParticipants* variable to *TRUE* if the filter on the **Type** field is equal to Machine.

The code may resemble the following “Setting ShowMaxParticipants” example.

Setting ShowMaxParticipants

```
ShowMaxParticipants := GETFILTER(Type) = FORMAT(Type::Machine);
```

- e. Around the two lines that set the values of variables, add code that first switches the filter group to 3, and then back to 0. Make sure that only filters that are set through the *RunPageView* property are considered.

The resulting code in the *OnOpenPage* trigger may resemble the following “OnOpenPage Trigger Code” example.

OnOpenPage Trigger Code

```
//CSD1.00>  
  
FILTERGROUP(3);  
  
ShowType := GETFILTER(Type) = "";  
  
ShowMaxParticipants := GETFILTER(Type) = FORMAT(Type::Machine);  
  
FILTERGROUP(0);  
  
//CSD1.00<
```

2. Save, compile, and then close the page.
 - a. Click **File > Save**.
 - b. In the **Save** dialog box, make sure that the **Compiled** check box is selected, and then click **OK**.
 - c. Close **Page Designer**.

Seminars

There is no standard functionality in Microsoft Dynamics NAV 2013 that handles Seminar Management requirements. You must develop a completely new application area. A standard application area in Microsoft Dynamics NAV 2013 consists of a setup table, master tables and pages, and any necessary subsidiary or supplemental tables with their pages.

The core master record of the Seminar Management functionality is the seminar. All master tables share some common characteristics, and any new master tables that you develop must also have the same functionality.

All master pages are accompanied by a card page and a list page. Both pages provide a similar, intuitive, and consistent functionality that you must also provide in any master pages that you develop.

Setup Table and Page

Setup table is one of the core tables of a application area in Microsoft Dynamics NAV 2013. The setup table provides the settings that define the behavior of the business logic in a specific application area of the application. The business logic frequently depends on various configuration settings. Every time that the code makes a decision about how to handle a specific situation, it must check the appropriate setting in the setup table.

For example, when users post invoices, the business decision may be that the posting is only allowed in a specific period. Instead of hard coding the starting and ending dates in the code, you should add the **Allow Posting From** and **Allow Posting To** dates to a setup table. Then you can change the logic of the code to check whether the **Posting Date** of the invoice falls between the dates that are specified in the table. This makes it easy for users to change the periods of allowed posting. This is exactly how the **General Ledger Setup** table controls the posting behavior of any kinds of documents that affect the **General Ledger** application area.

At a minimum, setup tables contain the fields for configuring the number series that control the assignment of numbers to master records, documents, and posted documents of the application area. The more master record and document types, the more fields the setup table contains. There can only be one setup record for a application area. That is why the table has a single primary key field which is called **Primary Key**.

A setup table always has only one page of type card that shows the information from the table. The setup card must make sure that there can be only one record in the table. Therefore, this page always has the `InsertAllowed` and `DeleteAllowed` properties set to `No`. When the user opens the page and there is no record in the setup table, the page typically inserts the setup record.

Master Tables

Master tables contain key information about objects or subjects of business transactions. Examples of master tables are **Customer, Vendor, Item, G/L Account**, or for this solution, **Seminar**. Users must be able to uniquely identify the records in these tables. For that reason, users typically use consecutively assigned numbers. Therefore, master tables always have the primary key that consists of a single field named **No.**, of type code, and length 20.

Number Series Functionality

When a record is inserted, the application assigns a value to this field from a number series that is defined in the setup table for the application area. All master tables in Microsoft Dynamics NAV 2013 follow the same code pattern to implement this functionality. You must follow the same pattern in any master tables that you develop.

The following "Assigning a number from number series" code example from the **Resource** table shows how standard master tables perform this task.

Assigning a number from number series

```
IF "No." = " THEN BEGIN

    ResSetup.GET;

    ResSetup.TESTFIELD("Resource Nos.");

    NoSeriesMgt.InitSeries(ResSetup."Resource Nos.",xRec."No. Series",0D,"No.,"No.
Series");

END;
```

This code first checks if the **No.** field is empty. This means that the user has not provided an explicit value in it. If there is no value, the code first retrieves the setup record for the application area and makes sure that the appropriate number series field is defined. Then, the code calls the standard number series functionality which assigns the next number that is based on the configuration in the **Number Series Line** table.

Every master table has a description field that is called either **Name** or **Description**, and is always of type text and length 50.

To support the number series functionality, you must provide a function that lets users select alternative number series. This function is called from the card page when users click the **AssistEdit** button in the **No.** field.

The following code example demonstrates the C/AL logic of the AssistEdit function of the **Customer** table.

Standard AssistEdit Code

```
WITH Cust DO BEGIN

    Cust := Rec;

    SalesSetup.GET;

    SalesSetup.TESTFIELD("Customer Nos.");

    IF NoSeriesMgt.SelectSeries(SalesSetup."Customer Nos.",OldCust."No.
Series","No. Series") THEN BEGIN

        NoSeriesMgt.SetSeries("No.");

        Rec := Cust;

        EXIT(TRUE);

    END;

END;
```

Finally, you must make sure that you observe the number series rules if users try to change the **No.** field. When numbers are assigned from a number series, the number series may dictate whether users can manually change the value which was assigned from the number series. There is the **No. Series** field in each master table that keeps track of the number series from which the **No.** field was assigned. To check whether the users can change the assigned number, you must write the code in the OnValidate trigger of the **No.** field.

The following code example demonstrates how the **Resource** table performs this check in the OnValidate trigger of the **No.** field.

No. – OnValidate Trigger in the Resource Table

```
IF "No." <> xRec."No." THEN BEGIN

    ResSetup.GET;

    NoSeriesMgt.TestManual(ResSetup."Resource Nos.");

    "No. Series" := "";

END;
```

Blocked Field

Master records are used in transactions. This means that after users have worked with the application for a while, many transactional records will relate to any master record. Many master records become obsolete at a specific time. For example, a customer may stop being your customer, or you may stop purchasing goods from a vendor, or an item may be discontinued. In these situations, you typically do not delete master records, but block them so that they remain available for any analysis or comparison purposes. However, you cannot use them in transactions any longer. The field which controls this functionality is called **Blocked**, it is of type Boolean, and it is present in all master tables.



Note: Occasionally, the **Blocked** field is an Option field that enables several types of blocks, letting the master record be used in one set of transactions, and preventing it from being used in other transaction types. For example, the **Blocked** field in the **Customer** and **Vendor** tables is of type Option.



Note: Not every table that contains the **Blocked** field is a master table. However, most of the tables that use this field use it in a similar manner.

Never write any logic that handles the **Blocked** field directly in the master table. But you must make sure that you check the **Blocked** field in the code of any tables that refer to or use the master table. For example, you check whether an item is blocked from the OnValidate trigger of the **No.** field in the **Sales Line** table, to make sure that users cannot use a blocked item in a sales transaction.

Master records are rarely changed, and users typically want to know when a specific master record was changed. Therefore, all master tables contain a noneditable field named **Last Date Modified** of type Date. Master tables also provide the code that sets the value of this field to the current system date when users change the record.

The following code example shows the code in the **OnModify** and **OnRename** triggers that sets the **Last Date Modified** field.

Setting Last Date Modified

```
"Last Date Modified" := TODAY;
```

Legacy Fields

Following are two fields that are typically present in all master tables:

- Comment
- Search Name or Search Description

Both of these fields are legacy fields and do not provide any useful functionality in Microsoft Dynamics NAV 2013. You should add these fields to any master table that you create to stay consistent with the standard layout of the master tables. They support any possible future use of these fields.

Posting Group Fields

Master records participate in business transactions, and business transactions typically have at least some financial aspect to them. The general ledger tracks all financial value of a company and can contain many accounts. In Microsoft Dynamics NAV 2013, the posting groups control the accounts that are used during different kinds of transactions. Each master record has relationships to the posting groups to facilitate selection of the appropriate accounts during the posting of master data transactions.

At the very minimum, all master data uses at least the general posting groups. General posting groups are one of several types of posting groups in Microsoft Dynamics NAV 2013. There are two types of general posting groups:

- *General business posting groups* that define the posting rules for the subjects of the transactions
- *General product posting groups* that define the posting rules for the objects of the transactions

When you design the master table, you must include either the **Gen. Bus. Posting Group** or the **Gen. Prod. Posting Group** field. You decide which one to include by first understanding how the master record is used. If the master record is the subject of your transactions (it represents who you are doing business with, such as customers or vendors), then select the **Gen. Bus. Posting Group** field. If the master record is the object of your transactions (it represents what you are operating with, such as items or resources), then select the **Gen. Prod. Posting Group** field.

Finally, if the master record has any tax relevance for your business, you must include either the **VAT Bus. Posting Group** or the **VAT Prod. Posting Group** field in the table. The rules for these two fields are the same as with the general posting groups.

Master Pages

All master tables have at least the following two pages that enable users to view and enter data in the table:

- List page is used in the list place of the RoleTailored client. It provides an overview of all records in the table.
- Card page is shown when users double-click a row in the list, or click **New**, **Edit**, **View**, or **Delete** actions in the list.

To enable standard navigation for master data pages in the RoleTailored client, you must set the following properties.

Object	Property	Remarks
Master Table	LookupPageID	Specifies which page provides the standard look-up behavior when users are looking up information from another table. For example, users look up information from the Item table when they enter lines in the Item Journal .
List Page	CardPageID	Specifies which page shows when users double-click lines in the list page, or when users click the New , Edit , View , or Delete actions.
List Page	Editable	Must be set to No , to prevent changes in the table directly from the list.

Solution Design

The CRONUS International Ltd. functional requirements are as follows:

- Seminars have a fixed duration, and a minimum and maximum number of participants.
- On occasion, seminars can be overbooked, depending on the capacity of the assigned room.
- Each seminar can be canceled if there are not enough participants.
- The price of each seminar is fixed.

C/SIDE Solution Development in Microsoft Dynamics® NAV 2013

It is obvious from the requirements description that seminars are a key business object for CRONUS International Ltd. Seminars are the master data for the Seminar Management application area that you are developing. Therefore, you must provide the following two tables:

- A setup table to configure the entire Seminar Management application area
- The master table to store information about all seminars

You also must provide all relevant pages for these tables so that users can move through the application and manage seminar information.

The following “Seminar Table Relationships” diagram shows the Seminar Setup and Seminar tables and their relationships to the other standard tables.

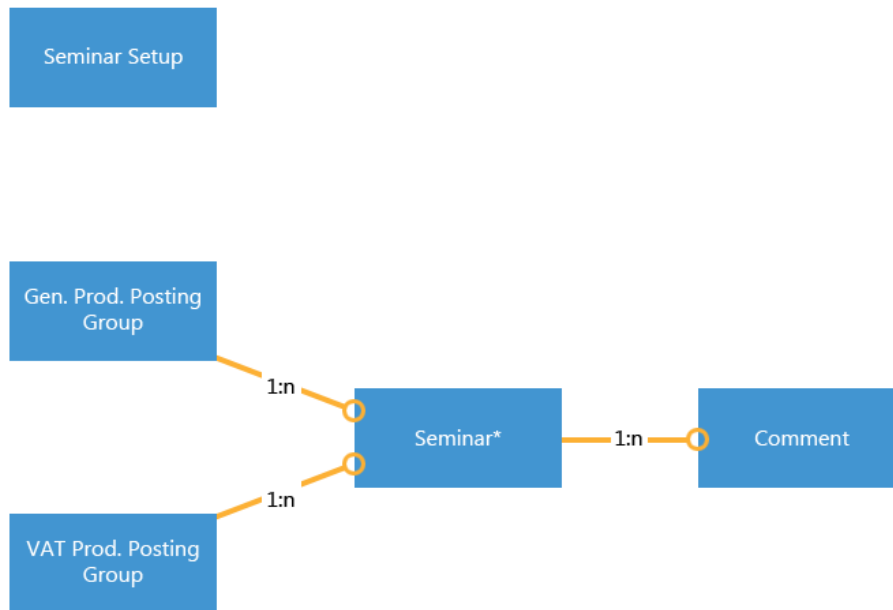


FIGURE 2.4: SEMINAR TABLE RELATIONSHIPS



Note: In this, and all additional entity relationship diagrams, the tables that are marked with an asterisk (*) are new tables that you must develop. All other tables are standard Microsoft Dynamics NAV 2013 tables.

Development

Your first development goal is to develop the tables and then the pages for your new Seminar Management application area. These tables and pages must follow standard Microsoft Dynamics NAV 2013 principles for the setup, master tables, and pages. These master tables and pages must provide the same functionality that users experience with other master tables and pages elsewhere in the application.

When you develop pages for Microsoft Dynamics NAV 2013, consider adding FactBoxes to the pages. FactBoxes enable users to see relevant information about data that is shown in the page. There are several system-provided FactBoxes that manage system-wide data, such as notes and links. Many application-provided FactBoxes display the application data.

At the very minimum, all card and list pages for master tables must have **Links** and **Notes** FactBoxes.

The **Seminar Setup** page as shown in the “Seminar Setup” figure, allows users to configure the Seminar Management application area.

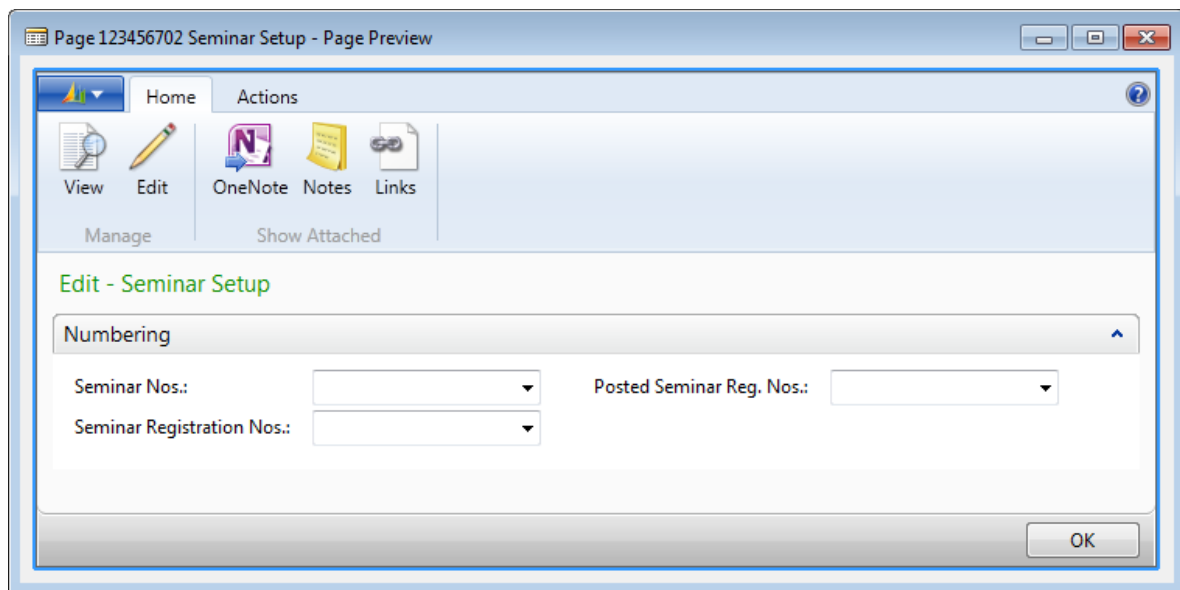


FIGURE 2.5: SEMINAR SETUP (PAGE 123456702)

The **Seminar Card** page as shown in the “Seminar Card” figure, allows users to enter and change the data in the **Seminar** table.

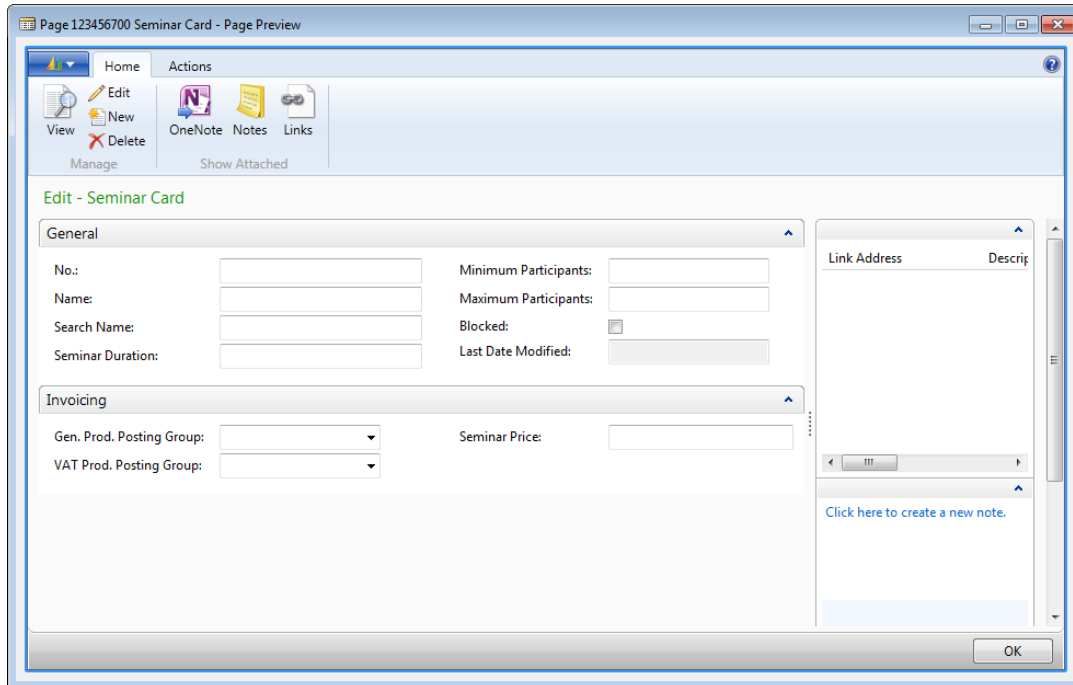


FIGURE 2.6: SEMINAR CARD (PAGE 123456700)

The **Seminar List** page, as shown in the "Seminar List" figure, provides an overview of the seminars, but does not let users change the data in the **Seminar** table.

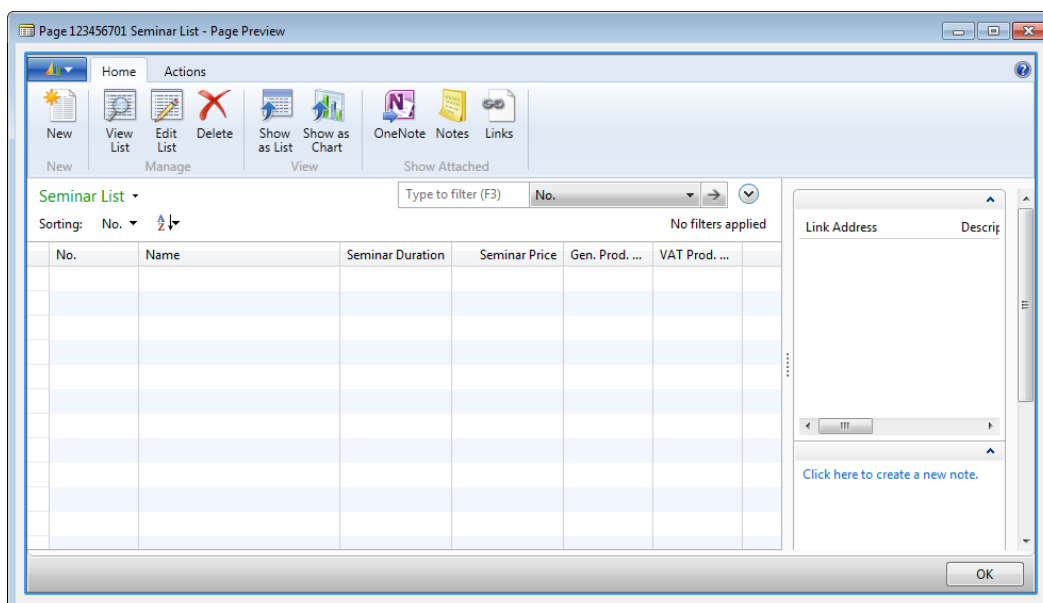


FIGURE 2.7: SEMINAR LIST (PAGE 123456701)

Lab 2.2: Creating Seminar Tables and Pages

Scenario

You must create tables for managing the seminar application area configuration and master data. Also, you must create pages that enable users to enter and maintain the data in these tables. These tables and pages must follow all the principles and Microsoft Dynamics NAV 2013 standards for setup tables, master tables, and card and list pages.

Exercise 1: Append the Table Name Option in the Comment Line table

Exercise Scenario

The *Comments* feature lets users enter arbitrary information about master records and documents. For master data, this information is kept in table **97, Comment Line**. This table has a composite primary key that consists of the following fields:

- **Table Name** that is an option field that specifies the master table to which a comment line relates
- **No.** that specifies the **No.** of the related record in the master table
- **Line No.** that specifies the unique line number of the comment line

To enable users to add comments to seminars, you must customize the **Table Name** option field to include the Seminar option.

Task 1: Add the Seminar Option

High Level Steps

1. Design the table **97, Comment line**.
2. Add the Seminar option to the list of options on the **Table Name** field.
3. Compile, save, and then close the table.

Detailed Steps

1. Design the table **97, Comment line**.
 - a. In **Object Designer**, click Table.
 - b. Select table **97, Comment Line**.
 - c. Click **Design** to open the **Table Designer** window.

2. Add the Seminar option to the list of options on the **Table Name** field.
 - a. Select the **Table Name** field.
 - b. Press SHIFT+F4 to open the **Properties** window.
 - c. In the OptionString property value, append six commas, and then enter "Seminar". Make sure that there are no spaces between the commas or before the word "Seminar".



Best Practice: When you append existing option fields with new options, you have to add several commas before the option that you add. This accommodates any options Microsoft may add to the field in a later version. If you do not add commas, any options that are added by Microsoft later will cause upgrade conflicts and result in possible bugs or issues.

- d. Repeat Step c for the OptionCaption property value.



Note: It is important not to add any spaces between the commas. If you add a space, then the option is not considered empty, and it is displayed as a blank option in the drop-down list. If you do not add a space, then the empty option is not displayed in the drop-down list.

- e. Close the **Properties** window.
3. Compile, save, and then close the table.
 - a. Click **File > Save**.
 - b. In the **Save** dialog box, make sure that the **Compiled** check box is selected.
 - c. Click **OK**.
 - d. Close **Table Designer**.

Exercise 2: Create the Seminar Tables

Exercise Scenario

You must create tables to support the Seminar Management functionality. You first create the **Seminar Setup** table, then the **Seminar** table, and finally you add the required C/AL code to the **Seminar** table to support standard master table behavior.

Task 1: Create the Seminar Setup Table

High Level Steps

1. Create the **Seminar Setup** table, and assign the ID of 123456701 to it.
2. Add fields to the **Seminar Setup** table.
3. Compile, save, and then close the table.

Detailed Steps

1. Create the **Seminar Setup** table, and assign the ID of 123456701 to it.
 - a. Make sure that the **Object Designer** shows tables.
 - b. Click **New** to open the **Table Designer** window.
 - c. Press SHIFT+F4 to access table properties.
 - d. In the **Properties** window, set the following properties.

Property	Value
ID	123456701
Name	Seminar Setup
Caption	Seminar Setup



Note: Notice that when you set the **Caption** property, the **CaptionML** property is automatically set as well. You must always set the **Caption** property of all objects and fields to take advantage of the Microsoft Dynamics NAV 2013 multilanguage functionality.

- e. Close the **Properties** window.

2. Add fields to the **Seminar Setup** table.
 - a. Continue in the **Table Designer** window for the **Seminar Setup** table. Add the following fields to the **Seminar Setup** table.

No.	Field Name	Type	Length	Remarks
1	Primary Key	Code	10	
2	Seminar Nos.	Code	10	Set the TableRelation property to 308 (the No. Series table).
3	Seminar Registration Nos.	Code	10	Set the TableRelation property to 308 (the No. Series table).
4	Posted Seminar Reg. Nos.	Code	10	Set the TableRelation property to 308 (the No. Series table).

3. Compile, save, and then close the table.
 - a. Click **File > Save**.
 - b. In the **Save** dialog box, make sure that the **Compiled** check box is selected, and then click **OK**.
 - c. Close **Table Designer**.

Task 2: Create the Seminar Table

High Level Steps

1. Create the **Seminar** table, and assign the ID of 123456700.
2. Add fields to the **Seminar** table.
3. Configure the **Comment** field as a FlowField which shows if related records exist in the **Comment Line** table.
4. Add keys to the **Seminar** table. Set the primary key to the **No.** field, and add a secondary key for the **Search Name** field.
5. Compile, save, and then close the table.


Detailed Steps

1. Create the **Seminar** table, and assign the ID of 123456700.
 - a. Make sure that the **Object Designer** shows tables.
 - b. Click **New** to open the **Table Designer** window.
 - c. Press SHIFT+F4 to access table properties.
 - d. In the **Properties** window, set the following properties.

Property	Value
ID	123456700
Name	Seminar
Caption	Seminar
LookupPagelD	123456700

- e. Close the **Properties** window.
2. Add fields to the **Seminar** table.
 - a. Continue in the **Table Designer** window for the **Seminar** table. Add the following fields to the **Seminar** table.

No.	Field Name	Type	Length	Remarks
1	No.	Code	20	
2	Name	Text	50	
3	Seminar Duration	Decimal		Set the DecimalPlaces property to 0:1
4	Minimum Participants	Integer		
5	Maximum Participants	Integer		
6	Search Name	Code	50	
7	Blocked	Boolean		
8	Last Date Modified	Date		Set the Editable property to No.
9	Comment	Boolean		Set the Editable property to No.

No.	Field Name	Type	Length	Remarks
10	Seminar Price	Decimal		<p>Set the AutoFormatType property to 1. This makes sure that the value is always formatted as an amount.</p> <hr/> <p> Note: If you want to learn more about the AutoFormatType property, see Developer and IT Pro Help.</p> <hr/>
11	Gen. Prod. Posting Group	Code	10	Set the TableRelation property to 251 (the Gen. Product Posting Group table).
12	VAT Prod. Posting Group	Code	10	Set the TableRelation property to 324 (the VAT Product Posting Group table).
13	No. Series	Code	10	Set the Editable property to No and the TableRelation property to 308 (the No. Series Table).

3. Configure the **Comment** field as a FlowField which shows if related records exist in the **Comment Line** table.
 - a. Select the **Comment** field.
 - b. Press SHIFT+F4 to open the **Properties** window.
 - c. Set the FieldClass property to FlowField. This also displays the CalcFormula property that was not previously visible.
 - d. In the CalcFormula property, click the **AssistEdit** button to open the **Calculation Formula** window.
 - e. In the **Method** field, select "Exist".
 - f. In the **Table Field**, select "Comment Line".
 - g. In the **Table Filter** field, click the **AssistEdit** button to open the **Table Filter** window.

h. Enter the following information in the **Table Filter** window.

Field	Type	Value
Table Name	CONST	Seminar
No.	FIELD	No.

- i. Click **OK** to close the **Table Filter** window. Make sure that you click **OK** before closing the window; otherwise, your changes are not saved.
 - j. Click **OK** to close the **Calculation Formula** window. Make sure not to close this window without clicking **OK**, because you will lose any changes that you made.
 - k. Close the **Properties** window.
4. Add keys to the **Seminar** table. Set the primary key to the **No.** field, and add a secondary key for the **Search Name** field.
- a. Click **View > Keys** to open the **Keys** window.
 - b. On the first line, enter "No."
 - c. On the second line, enter "Search Name".
 - d. Close the **Keys** window.
5. Compile, save, and then close the table.
- a. Click **File > Save**.
 - b. In the **Save** dialog box, make sure that the **Compiled** check box is selected, and then click **OK**.
 - c. Close **Table Designer**.

Task 3: Add Code to the Seminar Table

High Level Steps

1. Add the variables for the **Seminar Setup**, **Comment Line**, **Seminar** and **Gen. Product Posting Group** tables, and the `NoSeriesManagement` codeunit.
2. Add the code to the `OnInsert` trigger, to perform the following logic: if there is no value in the **No.** field, assign the next value from the number series that is specified in the **Seminar Nos.** number series in the **Seminar Setup** table.
3. Add the code to the `OnModify` and `OnRename` triggers to set the **Last Date Modified** field to the system date.
4. Add the code to the `OnDelete` trigger to delete any comment lines for the seminar record being deleted.

5. In the OnValidate trigger of the **No.** field, enter the code to perform the following logic: when the user changes the **No.** value, validate that the number series that is used to assign the number allows manual numbers. Then set the **No. Series** field to blank.
6. In the OnValidate trigger of the **Name** field, enter the code that sets the **Search Name** field if it was equal to the uppercase of the previous value of the **Name** field.
7. In the OnValidate trigger of the **Gen. Prod. Posting Group** field, enter the code that performs the following logic: if the **ValidateVatProdPostingGroup** function of the **Gen. Product Posting Group** table returns *TRUE*, set the **VAT Prod. Posting Group** to the value of the **Def. VAT Prod. Posting Group** field from the **Gen. Product Posting Group** table.
8. In the **Seminar** table, create a new function named **AssistEdit** with a return type of Boolean. In this function, enter the code that makes sure there is a value in the **Seminar Nos.** field in the **Seminar Setup** table, and then calls the **SelectSeries** function in the NoSeriesManagement codeunit to check the series number. If this function returns *TRUE*, call the **SetSeries** function in the NoSeriesManagement codeunit to set the **No.** field, and then return *TRUE*.
9. Compile, save, and then close the table.

Detailed Steps

1. Add the variables for the **Seminar Setup**, **Comment Line**, **Seminar** and **Gen. Product Posting Group** tables, and the NoSeriesManagement codeunit.
 - a. Design table **123456700** Seminar.
 - b. Click **View > C/AL Globals**.
 - c. Create the following variables.

Name	Data Type	Subtype
SeminarSetup	Record	Seminar Setup
CommentLine	Record	Comment Line
Seminar	Record	Seminar
GenProdPostingGroup	Record	Gen. Product Posting Group
NoSeriesMgt	Codeunit	NoSeriesManagement

Module 2: Master Tables and Pages

2. Add the code to the OnInsert trigger, to perform the following logic: if there is no value in the **No.** field, assign the next value from the number series that is specified in the **Seminar Nos.** number series in the **Seminar Setup** table.
 - a. Press F9 to open the **C/AL Editor** window.
 - b. In the OnInsert trigger, enter the following code:

```
IF "No." = " THEN BEGIN

SeminarSetup.GET;

SeminarSetup.TESTFIELD("Seminar Nos.");

NoSeriesMgt.InitSeries(SeminarSetup."Seminar Nos.",xRec."No.
Series",0D,"No.", "No. Series");

END;
```

3. Add the code to the OnModify and OnRename triggers to set the **Last Date Modified** field to the system date.
 - a. In the OnModify trigger, enter the following code:

```
"Last Date Modified" := TODAY;
```

- b. In the OnRename trigger, enter the same code.

4. Add the code to the OnDelete trigger to delete any comment lines for the seminar record being deleted.
 - a. In the OnDelete trigger, enter the following code:

```
CommentLine.RESET;

CommentLine.SETRANGE("Table Name",CommentLine."Table Name"::Seminar);

CommentLine.SETRANGE("No.", "No.");

CommentLine.DELETEALL;
```

5. In the OnValidate trigger of the **No.** field, enter the code to perform the following logic: when the user changes the **No.** value, validate that the number series that is used to assign the number allows manual numbers. Then set the **No. Series** field to blank.

- a. In the OnValidate trigger of the **No.** field, enter the following code:

```
IF "No." <> xRec."No." THEN BEGIN

    SeminarSetup.GET;

    NoSeriesMgt.TestManual(SeminarSetup."Seminar Nos.");

    "No. Series" := "";

END;
```

6. In the OnValidate trigger of the **Name** field, enter the code that sets the **Search Name** field if it was equal to the uppercase of the previous value of the **Name** field.

- a. In the OnValidate trigger of the **Name** field, enter the following code:

```
IF ("Search Name" = UPPERCASE(xRec.Name)) OR ("Search Name" = "") THEN BEGIN

    "Search Name" := Name;

END;
```

7. In the OnValidate trigger of the **Gen. Prod. Posting Group** field, enter the code that performs the following logic: if the **ValidateVatProdPostingGroup** function of the **Gen. Product Posting Group** table returns *TRUE*, set the **VAT Prod. Posting Group** to the value of the **Def. VAT Prod. Posting Group** field from the **Gen. Product Posting Group** table.

- a. In the OnValidate trigger of the **Gen. Prod. Posting Group** field, enter the following code:

```
IF xRec."Gen. Prod. Posting Group" <> "Gen. Prod. Posting Group" THEN BEGIN

    IF
    GenProdPostingGroup.ValidateVatProdPostingGroup(GenProdPostingGroup,"VAT
    Prod. Posting Group") THEN BEGIN

        VALIDATE("VAT Prod. Posting Group",GenProdPostingGroup."Def. VAT Prod.
        Posting Group");

    END;

END;
```

8. In the **Seminar** table, create a new function named **AssistEdit** with a return type of Boolean. In this function, enter the code that makes sure there is a value in the **Seminar Nos.** field in the **Seminar Setup** table, and then calls the **SelectSeries** function in the NoSeriesManagement codeunit to check the series number. If this function returns *TRUE*, call the **SetSeries** function in the NoSeriesManagement codeunit to set the **No.** field, and then return *TRUE*.
 - a. Click **View > C/AL Globals**.
 - b. In the **C/AL Globals** window, click the **Functions** tab.
 - c. In the first empty line, enter "AssistEdit".
 - d. Click **Locals**.
 - e. In the **C/AL Locals** window, click the **Return Value** tab.
 - f. In the **Return Type** field, select Boolean.
 - g. Close the **C/AL Locals** window.
 - h. Close the **C/AL Globals** window.
 - i. In the AssistEdit function trigger, enter the following code:

```
WITH Seminar DO BEGIN

    Seminar := Rec;

    SeminarSetup.GET;

    SeminarSetup.TESTFIELD("Seminar Nos.");

    IF NoSeriesMgt.SelectSeries(SeminarSetup."Seminar Nos.",xRec."No. Series","No.
Series") THEN BEGIN

        NoSeriesMgt.SetSeries("No.");

        Rec := Seminar;

        EXIT(TRUE);

    END;

END;
```

9. Compile, save, and then close the table.
 - a. Click **File > Save**.
 - b. In the **Save** dialog box, make sure that the **Compiled** check box is selected, and then click **OK**.
 - c. Close **Table Designer**.

Exercise 3: Create the Seminar Pages

Exercise Scenario

To fully support the master data functionality, you must define the **Seminar Setup** page to configure the Seminar Management application area, the **Seminar Card** page, and the **Seminar List** page. Then you must establish standard Microsoft Dynamics NAV 2013 master page functionality. This includes the RoleTailored client navigation and number series selection functionality.

Task 1: Create the Seminar Setup Page

High Level Steps

1. Create the **Seminar Setup** page as shown in the "Seminar Setup (Page 123456702)" figure, and assign the **ID** of 123456702 to it.
2. Add code to the OnOpenPage trigger to insert a new record, if there is not a record in the Seminar Setup table.
3. Compile, save, and then close the page.

Detailed Steps

1. Create the **Seminar Setup** page as shown in the "Seminar Setup (Page 123456702)" figure, and assign the **ID** of 123456702 to it.
 - a. In **Object Designer**, click **Page** to access the list of pages.
 - b. Click **New** to create a new page.
 - c. In the **New Page** window, in the **Table** field, enter "123456701", and then select the **Create a page using a wizard** option.
 - d. Click **OK** to start the **Card Page Wizard**.
 - e. In the **FastTab Name** column, replace the existing value with "Numbering", and then click **Next**.
 - f. In **Available Fields**, select and then click > for the following fields:
 - Seminar Nos.
 - Seminar Registration Nos.
 - Posted Seminar Registration Nos.
 - g. Click **Next**, and then click **Finish**.
 - h. In the **Page Designer** window, select the first empty line, and then press SHIFT+F4 to access the **Page - Properties** window.

- i. Set the properties as follows.

Property	Value
ID	123456702
Name	Seminar Setup
Caption	Seminar Setup

- j. Close the **Page – Properties** window.

2. Add code to the OnOpenPage trigger to insert a new record, if there is not a record in the Seminar Setup table.
 - a. Click **View > C/AL Code**, or press F9 to open the **C/AL Editor** window.
 - b. In the OnOpenPage trigger, enter the following code:

```
IF NOT FINDFIRST THEN  
  
INSERT;
```

3. Compile, save, and then close the page.
 - a. Click **File > Save**.
 - b. In the **Save** dialog box, make sure that the **Compiled** check box is selected, and then click **OK**.
 - c. Close **Page Designer**.

Task 2: Create the Seminar Card Page

High Level Steps

1. Create the **Seminar Card** page as shown in the “Seminar Card (Page 123456700)” figure, and assign the **ID** of 123456700 to it.
2. Add the code to the OnAssistEdit trigger of the **No.** field control to call the **AssistEdit** function of the underlying table, and to update the page if the function returns *TRUE*.
3. Add an action to the page, to show the **Comment Sheet** page for the seminar that is currently displayed in the page.
4. Compile, save, and then close the page.

Detailed Steps

1. Create the **Seminar Card** page as shown in the “Seminar Card (Page 123456700)” figure, and assign the **ID** of 123456700 to it.
 - a. In **Object Designer**, click **New** to create a new page.
 - b. In the **New Page** window, in the **Table** field, enter “123456700”, and then select the **Create a page using a wizard** option.
 - c. Click **OK** to start the **Card Page Wizard**.

- d. In the **FastTab Name** list, in the first empty line enter "Invoicing", and then click **Next**.
- e. Add the following fields to the **General** FastTab:
 - No.
 - Name
 - Minimum Participants
 - Maximum Participants
 - Search Name
 - Seminar Duration
 - Blocked
 - Last Date Modified
- f. Add the following fields to the **Invoicing** FastTab:
 - Gen. Prod. Posting Group
 - VAT Prod. Posting Group
 - Seminar Price
- g. Click **Next**.
- h. Click the **System** tab, and then add the **RecordLinks** and **Notes** FactBoxes in that order.
- i. Click **Finish**.
- j. In the **Page Designer** window, select the first empty line, and then press SHIFT+F4 to access the **Page - Properties** window.
- k. Set the properties as follows.

Property	Value
ID	123456700
Name	Seminar Card
Caption	Seminar Card

- l. Close the **Page – Properties** window.
2. Add the code to the OnAssistEdit trigger of the **No.** field control to call the **AssistEdit** function of the underlying table, and to update the page if the function returns *TRUE*.
 - a. In the OnAssistEdit trigger of the **No.** field control, enter the following code:

```
IF AssistEdit THEN  
    CurrPage.UPDATE;
```

Module 2: Master Tables and Pages

3. Add an action to the page, to show the **Comment Sheet** page for the seminar that is currently displayed in the page.
 - a. Click **View > Page Actions** to open the Action Designer.
 - b. Enter the following information.

Type	SubType	Caption
ActionContainer	RelatedInformation	
ActionGroup		&Seminar
Action		Co&mmments

- c. Select the **Comments** action, and then click SHIFT+F4 to open the **Properties** window.
- d. In the Image property, enter "Comment".
- e. In the RunObject property, enter "Page Comment Sheet".
- f. In the RunPageLink property, click the **AssistEdit** button to open the **Table Filter** window.
- g. Enter the following information in the **Table Filter** window.

Field	Type	Value
Table Name	CONST	Seminar
No.	FIELD	No.

- h. Click **OK** to accept the changes and close the **Table Filter** window.
 - i. Close the **Properties** window.
4. Compile, save, and then close the page.
 - a. Click **File > Save**.
 - b. In the **Save** dialog box, make sure that the **Compiled** check box is selected, and then click **OK**.
 - c. Close **Page Designer**.

Task 3: Create the Seminar List Page

High Level Steps

1. Create the **Seminar List** page as shown in the "Seminar List (Page 123456701)" figure, and assign the **ID** of 123456701 to it.
2. Add an action to the page to show the **Comment Sheet** page for the seminar that is selected in the list.
3. Compile, save, and then close the page.

Detailed Steps

1. Create the **Seminar List** page as shown in the “Seminar List (Page 123456701)” figure, and assign the **ID** of 123456701 to it.
 - a. In **Object Designer**, click **New** to create a new page.
 - b. In the **New Page** window, in the **Table** field, enter “123456700”.
 - c. Select the **Create a page using a wizard** option, and then select the **List** option.
 - d. Click **OK** to start the **List Page Wizard**.
 - e. Add the following fields:
 - **No.**
 - **Name**
 - **Seminar Duration**
 - **Seminar Price**
 - **Gen. Prod. Posting Group**
 - **VAT Prod. Posting Group**
 - f. Click **Next**.
 - g. Click the **System** tab, and then add the **RecordLinks** and **Notes** FactBoxes in that order, and then click **Finish**.
 - h. In the **Page Designer** window, select the first empty line, and then press SHIFT+F4 to access the **Page - Properties** window.
 - i. Set the properties as follows.

Property	Value
ID	123456701
Name	Seminar List
Caption	Seminar List
Editable	No
CardPageID	Seminar Card

- j. Close the **Page – Properties** window.
2. Add an action to the page to show the **Comment Sheet** page for the seminar that is selected in the list.
 - a. Do not close the **Page Designer** for the **Seminar List** page.
 - b. Press SHIFT+F12 to go to the **Object Designer**.
 - c. Select page **123456700, Seminar Card** and click **Design**.
 - d. Click **View > Page Actions**.
 - e. In the **Action Designer** window, press CTRL+A, and then CTRL+C. This copies all actions from the **Seminar Card** page to clipboard.

Module 2: Master Tables and Pages

- f. Close the Action Designer and Page Designer windows for the **Seminar Card** page.
- g. On the **Window** menu, click the option for the **Page Designer** window for the **Seminar List** page.
- h. Click **View > Page Actions** to open the **Action Designer**.
- i. Press CTRL+V to past the actions from the clipboard.



Note: You can copy actions from one page to another using the standard CTRL+C and CTRL+V keyboard shortcuts. This copies actions with all their properties.

- j. Close the **Action Designer**.
3. Compile, save, and then close the page.
 - a. Click **File > Save**.
 - b. In the **Save** dialog box, make sure that the **Compiled** check box is selected, and then click **OK**.
 - c. Close **Page Designer**.

Module Review

Module Review and Takeaways

In this chapter, you used two of the most fundamental object types, tables and pages, to establish the foundation for the Seminar Management solution. These tables define and store data for the solution. The pages also provide an intuitive interface where the user can interact with the table data.

You have seen and applied the master table principles of Microsoft Dynamics NAV 2013 standards to the master tables and pages to make sure that users have a consistent experience across the application.

Test Your Knowledge

Test your knowledge with the following questions.

1. What is the main difference between Documentation triggers and event triggers?

2. What is the main difference between event triggers and function triggers?

3. Which property do you set on a page to display a specific caption in Swedish language?

- () Name
- () NameML
- () Caption
- () CaptionML
- () CaptionSwedish

4. When defining Multilanguage properties, you must always define a value in English (United States) language. Which language identifier represents English (United States)?
- ENU
 - ENG
 - EN
 - ENML

Test Your Knowledge Solutions

Module Review and Takeaways

1. What is the main difference between Documentation triggers and event triggers?

MODEL ANSWER:

Documentation triggers contain free-text documentation, and their contents are not compiled or run. Event triggers contain C/AL code that must compile and that runs when the event occurs.

2. What is the main difference between event triggers and function triggers?

MODEL ANSWER:

Event triggers are defined by the system. They run when a predefined event occurs. Function triggers are defined by the developer. They run when a C/AL code line calls them.

3. Which property do you set on a page to display a specific caption in Swedish language?

Name

NameML

Caption

CaptionML

CaptionSwedish

4. When defining Multilanguage properties, you must always define a value in English (United States) language. Which language identifier represents English (United States)?

ENU

ENG

EN

ENML