

MODULE 6: UPGRADE REPORTS

Module Overview

In Microsoft Dynamics® NAV 2013, you can create reports or you can import reports from earlier versions of Microsoft Dynamics NAV. Typically, reports from an earlier version are migrated to Microsoft Dynamics NAV 2013. To avoid the loss of work, this module explains how to migrate reports to Microsoft Dynamics NAV 2013.

Objectives

The objectives are:

- Explain the report upgrade workflow for different report types.
- Implement report dataset optimization.
- Describe the upgrade process and flow of reports to the RoleTailored client.
- Explain how to use the dataset for the report dataset designer.
- Describe how to upgrade classic reports.
- Apply dataset optimization techniques in reports.
- Apply report design UX guidelines.
- Optimize performance.
- Build the request page.
- Import and upgrade a report that has only a Classic report layout to a Microsoft Dynamics NAV 2013 report that has an RDLC layout.
- Modify the RDLC layout.

How to Upgrade a Report to Microsoft Dynamics NAV 2013

This lesson explains the upgrade process for reports from an earlier version of Microsoft Dynamics NAV to Microsoft Dynamics NAV 2013.

Upgrading a Report

Upgrading reports is an automated process that extracts dataset information and deletes unnecessary classic report components from the report object.

In Microsoft Dynamics NAV 2009, a report object can contain the following combinations of components:

- Classic report layout (sections) and (optional) request form.
- Client report definition (RDLC) layout and (optional) request page.
- A hybrid report that contains both the classic report layout (sections) and the (optional) request form and an RDLC report layout and (optional) request page.
- A ProcessingOnly report without a layout.

In Microsoft Dynamics NAV 2013, a report object can contain only an RDLC layout and request page.

If you import a report from Microsoft Dynamics NAV 2009 that has both the classic report layout and the RDLC report layout, then you must upgrade that report. In Microsoft Dynamics NAV 2013, you can compile. However, you cannot run or design a report that has a classic report layout. When you upgrade a report, the classic report layout and the Request form are deleted. Then, the field information from the report sections is converted to a dataset definition that is valid for Microsoft Dynamics NAV 2013 reports. Dynamics NAV 2013 uses RDLC 2008 and this implies an adjustment and modification to convert and upgrade of the RDLC XML source file.

You can have reports that do not have RDLC layouts. For example, you could have deleted the RDLC layout from a report so that you could view the Classic layout only. You can also have a report from a version that is from an earlier version of Dynamics NAV. If this is the case, your report does not contain an RDLC layout or a request page. If you import a report that has only a Classic report layout and request form, then when you upgrade the report, an RDLC report layout is created automatically. You can create a request page for the report or you can use the form transformation tool to transform a request form to a request page.

After you upgrade a report, it is converted from an RDLC 2005 format to an RDLC 2008 format. To convert a report to RDLC 2008 format, open the report in either Visual Studio 2010 Report Designer or from the RoleTailored client that uses Report Viewer 2010.

Upgrade the Path for a Hybrid Report

The steps that are required to upgrade a hybrid report are outlined in the following flowchart.

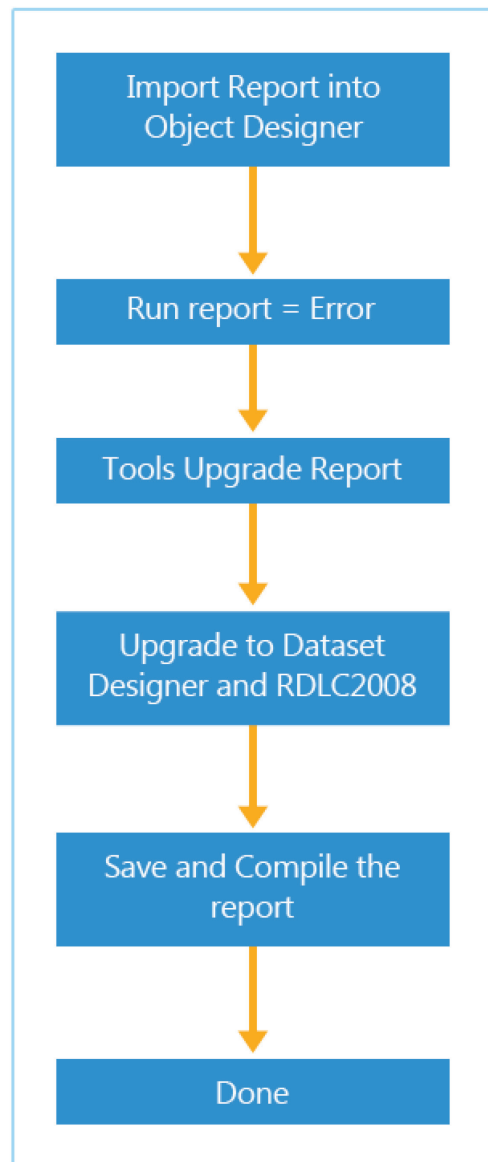


FIGURE 6.1: UPGRADE HYBRID REPORT WORKFLOW SCREENSHOT



Best Practice: For the Upgrade Report to work, you must first compile the report.

Upgrade the Path for a Processing Only Report

A processing only report is a report without a layout. The report does not produce any printed output. This type of report processes data. The steps that are required to upgrade a processing only report are outlined in the following flowchart.

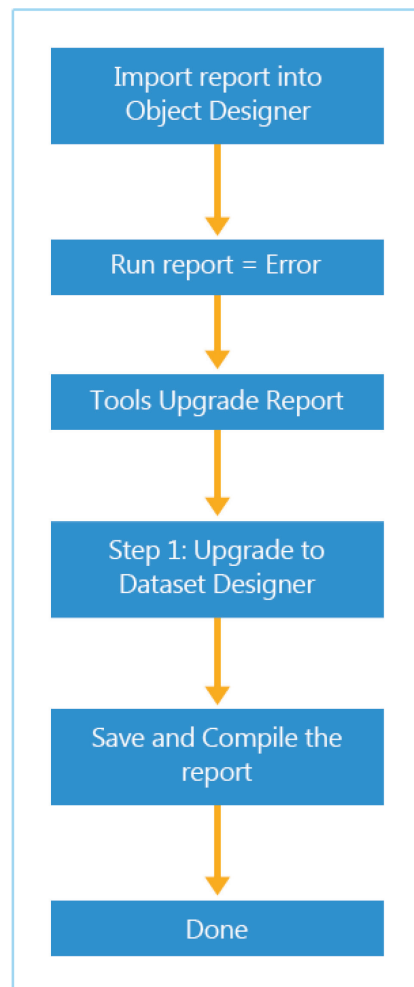
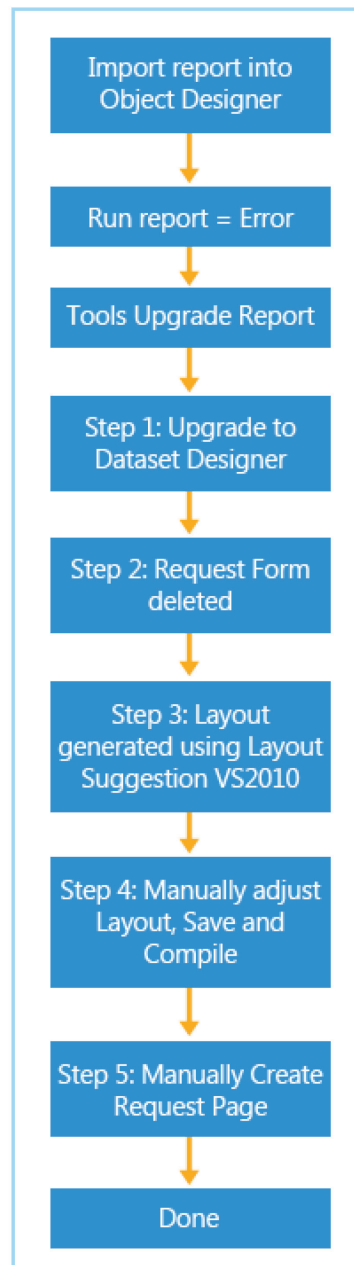


FIGURE 6.2: UPGRADE PROCESSING ONLY REPORT WORKFLOW SCREENSHOT

Upgrade the Path for a Classic Report

A Classic report is a report that contains a Classic layout but no RDLC layout.

The steps that are required to upgrade a Classic report are outlined in the following flowchart.



The Upgrade Process

Before you upgrade reports, it is recommended that you create a backup of the reports. You cannot undo the upgrade after it is finished. To back up reports, use one of the following methods:

- Export the reports to a file. (FOB, TXT or XML)
- Create a backup of the database.



Note: Verify that you have Visual Studio 2010 installed. This includes the Visual Web Developer feature. You can use any of the following editions of Visual Studio 2010:

- Premium
- Ultimate

To upgrade a report, follow these steps.

1. In the development environment, on the **Tools** menu, click **Object Designer**.
2. Import the report that you want to upgrade into Microsoft Dynamics NAV 2013.
3. In Object Designer, click **Report**, and then select the report that you want to upgrade.
4. To compile the report, click **Tools, Compile**.
5. On the **Tools** menu, click **Upgrade Report**.
6. Click the Yes button to confirm that you want to upgrade the reports.



FIGURE 6.4: CONFIRM UPGRADE OF REPORT SCREEN SHOT

7. The Classic report layout and request form for each report is deleted during the upgrade.
(The Transformation Tool could be used to transform it to a Request Page.)
8. Select the report that you upgraded, and then click **Design**.

9. On the **View** menu, click **Layout** to open Visual Studio Report Designer.
10. Click the **OK** button to convert the report to RDLC 2008 format.
11. On the **File** menu, click **Save Report.rdlc**.
12. In the Microsoft Dynamics NAV development environment, select an empty line in Report Dataset Designer. A dialog box appears with the following message:

"The layout of report id: '<report ID>' is changed by another application.

Filename of the changed RDLC: <filename>

Do you want to load the changes?"

13. Click **Yes** to load the changes.
14. In the development environment, on the **File** menu, click **Save**.
15. In the **Save** dialog box, click **Compile**, and then click the **OK** button.

Optimize the Dataset for the New Report Dataset Designer

This lesson explains how to prepare to upgrade a report to Microsoft Dynamics NAV 2013.

Describe How the Dataset is Generated

Before you manually update the new RDLC report layout, you should analyze and understand the report. Sometimes you can achieve the same result by using different functionality in the RDLC layout than in the Classic report layout. By understanding the report and the features that are available in Visual Studio Report Designer, you can determine the manual modifications that you must make to the new RDLC report layout. For example, the filtering can be managed in the development environment or in the Visual Studio Report Designer.

Additionally, you should understand how the dataset that contains the data that is displayed in the report is generated. For a Classic report layout, the data for each data item is retrieved from the data item and it is formatted in the report sequentially. For example, if you run an **Order Confirmation** report for multiple sales orders and the data for the first order is retrieved and formatted for the Classic report layout, then the data for the second order is retrieved and formatted, and so on.

For an RDLC report layout, all data for the report is retrieved from the data item into one dataset. Then the data is included in the RDLC layout. For example, if you run an **Order Confirmation** report for multiple orders, then the data from all orders is merged into one dataset. The layout specifies that the data is grouped on pages according to the sales order number.

More information about the generation of the dataset is available in the “Creating Reports” module.

Optimize the Dataset

You must understand the dataset to understand the report and know how to debug it. Report performance is an important concept of report design, especially when you upgrade a report. Optimizing the dataset of a report usually means minimizing the dataset. The fewer rows that are located in the dataset at run time, the quicker the report will execute and fewer memory resources are consumed. When a report dataset grows, the application will consume increased memory and then when memory is no longer available, the report execution fails. To avoid this scenario, you can optimize the dataset by following these guidelines.

Captions and Labels

In reports that are upgraded from earlier versions of Microsoft Dynamics NAV 2013, the field captions are columns of the dataset. To have the caption of a field, title for a chart, or a title for the report, translated into the user’s language at run time, the field is added as a column in the dataset. In the Microsoft Dynamics NAV 2013 Report Dataset Designer, you can use labels instead. You can add labels that are related to another data item, such as the caption for a table field, or labels that are not related to another data item, such as the title of the report. Labels are passed as parameters to the report. The Label Report Parameter figure shows an example.

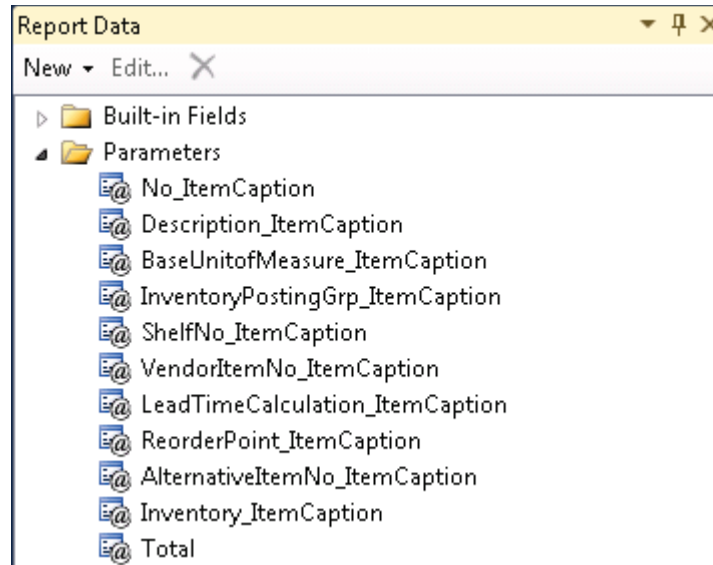


FIGURE 6.5: LABEL REPORT PARAMETER



Note: In Visual Studio Report Designer, if you want to use the **Caption** or **CaptionML** values of a field as a label on a report layout, then you must include the caption in the dataset. You design the dataset by using Report Dataset Designer. When you add table fields to the dataset with Report Dataset Designer, you can set the **IncludeCaption** property to **Yes** to specify that the caption for the field is also added to the dataset. If you do not include the caption in the dataset, then in a multilanguage application, you cannot change the language of the labels in the report.

You can set the **IncludeCaption** property in the **Properties** window of a data item or in the **IncludeCaption** column of Report Dataset Designer for a data item or a column. If you set the **IncludeCaption** column of a data item, then the **IncludeCaption** property is set to **Yes** for every child column of the data item.

The following is a summary of captions and labels:

- If there is a caption in the source expression, then consider using **IncludeCaption** instead.
- If there is a text constant in the source expression, consider using a **Label**.
- There is a significant reduction in memory consumption when values are passed as report parameters instead of fields on each row.

Remove unused columns from the dataset

A column in the report dataset designer will become a column in the result set that is generated at run time. If the field is not used or not required to be available in the resultset, it should be removed from the report dataset designer. This reduces memory consumption and provides improved report execution.

Avoid the usage of binary large object (BLOB) fields in the dataset

A BLOB field can potentially contain 2gb of data. Adding it to the report dataset can potentially increase the size of the dataset significantly. When a BLOB field, for example the company logo, is required in the dataset it is recommended to use a separate integer data item to hold the BLOB field(s). You can filter the integer dataitem in its **DataItemTableView** property by using this filter:

```
WHERE(Number=CONST(1))
```

Avoid unnecessary rows in the result set

If a data item is in the Report Dataset Designer for calculations only, and its fields are not required for the dataset, consider using the DataItemTable view property of the data item to filter the data item.

Apply the correct filters in the dataset

Filters can be applied in the RDLC layout of a report. All report data regions in Visual Studio Report Designer can be used for filtering at run time. However, if the same result can be achieved by applying filters in the Report Dataset Designer, by using properties or C/AL code, it will produce a smaller dataset and improve report execution performance. Filter, sort, and aggregation is more efficient on the data source than during report processing.

A good example of this technique can be found here:

<http://code.msdn.microsoft.com/windowsdesktop/How-to-boost-performance-ffb6c384> "How to boost performance on Dynamics NAV RTC reports: Report 1001, Inventory Valuation"


The Windows Page File

The following demonstration will show how to set the size of the page file.

Demonstration: Change the Size of the Windows Page File

When a report is executed, the dataset is sent from the Microsoft Dynamics NAV server into the memory on the server. If the dataset exceeds the memory Windows makes available, then Windows will stop allocating additional memory and the client might stop responding. If you cannot reduce the dataset, then the only alternative is to increase available memory on the server.

You can do this by physically adding more memory or by increasing the page file in Windows. Keep in mind that the Client is still a 32-bit application.

 **Best Practice:** Windows can handle the size of the page file for general Windows performance. So interfering with this can cause problems for any other tasks that are running on that server. So if you do decide to increase the page file, then it is recommended that you do this on a dedicated server that is used to print only large reports.

Demonstration Steps

1. Change the page file.
 - a. Click **Control Panel > System and Security > System > Advanced System Settings > Advanced**.
 - b. Click **"Settings..."** in the **"Performance"** section.
 - c. Select the **Advanced** tab.
 - d. Click **Change** on the **Virtual Memory** section.
 - e. The Virtual Memory window appears.

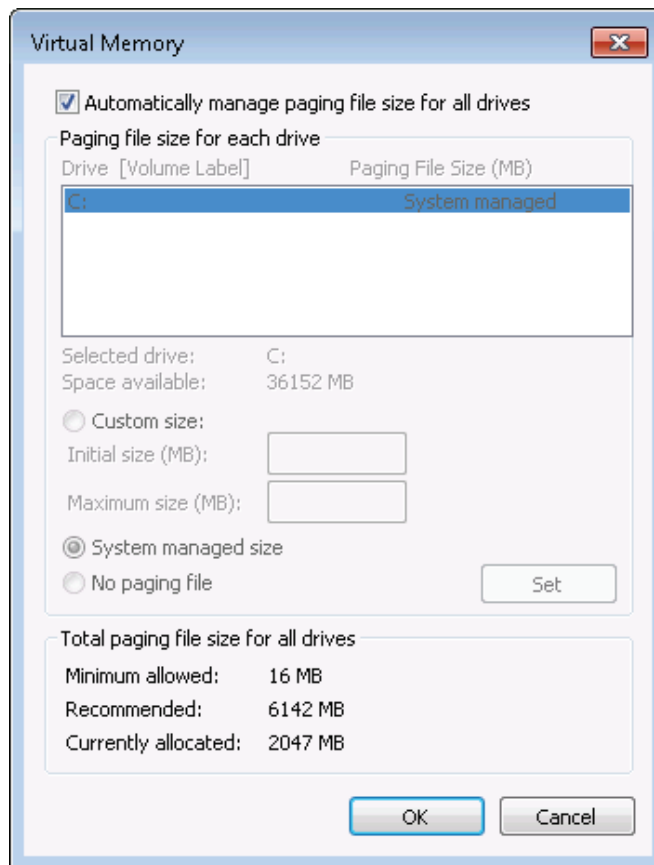


FIGURE 6.6: CHANGE PAGE FILE

- f. Deselect the option: **Automatically manage paging file size for all drives**.
- g. Select **custom size**.
- h. Enter an initial size and a maximum size in the corresponding fields.
- i. Restart the server.



Note: Finding the correct values for the initial size and maximum size could require several tries.

Changing the page file settings does not affect the performance of reports. However, by doing this, the client can receive more data before it runs out of memory.

Optimize the Layout for RDLC 2008

This lesson explains how to redesign the layout to prepare for an upgrade to Microsoft Dynamics NAV 2013.

Report Design Guidelines

In Microsoft Dynamics NAV 2013, for reports, the standard application uses new Report Design Guidelines. This provides consistency to reports. Now when you design a report, other developers will be familiar with the appearance of the report, and they will no longer have to invent a new appearance for each report. Additionally, the Report Design Guidelines will help you save time when you design reports.

More information about the Report Design Guidelines is available in the “Creating Reports” module of this course.

Report Performance Optimization Tips

Conditional visibility on a big dataset

For report items with conditional visibility, the report processor must apply grouping, sorting, and filtering expressions even if only top level data is visible. Although on-demand processing optimizes data evaluation by processing only visible data, all data is part of the report. If the user only wants to have detail data visible sometimes, a drill down report is a better choice.

Best Practices when Displaying Data on a Chart

Consider how much data is needed for a chart or gauge. Typically, having more pixels on a monitor will decrease performance and the visual display of graphics. It is recommended not to have more than seven or eight sections in a pie chart.

Some charts, such as, scatter graphs, benefit from many data points, whereas other charts, such as pie charts, are more effective with fewer data points. Therefore, carefully select a chart type based on the values in your dataset and how you want this information to be shown.

Some ways to combine data on a chart, include the following:

- When you use a pie chart, merge small sections into one section called "Other." This reduces the number of sections on your pie chart.
- When there are several data points, do not use data point labels. Data point labels are most effective when there are only a few points on the chart.
- Filter unwanted or irrelevant data. This helps highlight the key data that you are trying to show on the chart.
- If you want to add a bar graph to show ratio data in a table or matrix template, consider using a linear gauge instead. Gauges are better suited in displaying a single value inside a cell.

Expressions in the Page Header or Footer Force All Pages To Be Processed

When you include a reference to the built-in field [&TotalPages], the report processor must paginate the whole report before it can render the first page. If no reference to [&TotalPages] exists, the first page can be rendered and returned to the user immediately, without processing the rest of the report. Additionally, the report processor assumes that any complex expression in the page header or footer might contain a direct or indirect reference to [&TotalPages].

To avoid having the report processor paginate a lengthy report, do not include a reference to [&TotalPages] or any complex expressions in the page header and page footer.

Complex Tablix Data Region Grouping and Aggregate Functions

Many levels of nested and adjacent groups in a Tablix data region can affect report processing performance. Consider the level of grouping, the number of group instances, and the aggregate functions that require being evaluated after group, filter, and sort expressions are applied. For example, **Previous** is an 'expensive' aggregate function because its value depends on the sorted elements of a data region; **Sum** is not order-dependent and requires fewer resources. Other post-sort aggregates include **First** and **Last**.

Reducing the data in the report might be sufficient to provide acceptable performance without changing any aggregate function calls.

Report is Not Optimized for the Chosen Rendering Format

Some features are not supported in all renderers. If the primary format for viewing a report is a specific file format, you might have to modify the report design for a better viewing experience for the user.

For example, each page break defines a new sheet in Excel. Each sheet can handle no more than 65000 rows. Consider these limits when you set the page breaks in a report.

For exporting to Excel, do not merge cells in a Tablix data region. In free form reports, align report items vertically. Merged cells and unaligned report items interfere with Excel functionality in the exported report.

Design Tips for Better Report Processing

For better report performance, use the following information to help increase the time that is required to process your report:

- For reports that have many instances of text boxes, set **CanGrow** and **CanShrink** on text boxes to **FALSE**. By default, each cell in a Tablix data region contains a text box so that the total number of text boxes that must be rendered can grow quickly.
- For reports that have many images, set **AutoSize** on images to a different value such as **Fit**.
- For text boxes, do not set the property **TextAlign** to **General**. This value requires conditional processing, depending on the text box content.
- Do not use horizontal page breaks when they are not required. Review the margins, column widths, and white space in a report. For example, render the report to a .TIFF file and view it in the Microsoft Windows Picture and Fax viewer to determine whether additional pages are being rendered.
- Set the **KeepTogether** property on **Tablix** members only when you must control the specific rendering behavior for a Tablix data region. The **KeepTogether** feature requires additional processing when page breaks are calculated.

Add and Identify Hidden Fields

If you have hidden fields on a report, it is recommended that you mark them so that they are easy to identify. Additionally, you might want to move hidden fields to a different location in the layout to avoid problems with formatting.

To identify hidden fields, follow these steps.

1. In the development environment, on the **Tools** menu, click **Object Designer**.
2. In Object Designer, click **Report**, select the report to which you want to add a hidden field, and then click the **Design** button.
3. On the **View** menu, click Layout.
4. In Visual Studio Report Designer, on the **View** menu, click **Toolbox**.
5. From the toolbox, under Report Items, drag a text box to the report layout.



Note: If the hidden text box is in a table data region, it is recommended that you insert a new row in the table and add the hidden text box to this new row. In the **Properties** window of the row, under Layout, you change the value of the Height property to a larger value so that you can view the row when you design the report. However, make sure that the value is not so large that it will cause formatting problems when you print the report, such as 0.2cm.

6. Select the text box that you created. On the **View** menu, click the **Properties** window.
7. In the **Properties** window, under Font, set the Color property to Yellow, or select the yellow square from the drop-down list.
8. In the **Properties** window, under Position, expand Size, and then set the Width property to a small width, such as 0.15cm.
9. In the **Properties** window, under Visibility, set the Hidden property to True.
10. On the **File** menu, click Save Report.rdlc.

Hide Nonfiltered Headers

If you apply a filter to the data in a report, then it is recommended that you create an additional header row in the report to display the filter. You display this header row only if the user specifies a filter. If no filter is specified, then the filter header is not displayed.

To conditionally display the filter header, you must add a filter variable to the report dataset. Next, in the RDLC report layout, create a conditional expression on the visibility of the filter header that uses the filter variable that you added to the dataset.

An example of a report that uses additional header rows to display filter information is report 113, **Customer/Item Sales**.

To hide non-filtered headers, follow these steps.

1. In the development environment, on the **Tools** menu, click Object Designer.
2. In Object Designer, click **Report**, select the report that you want to modify, and then click **Design**.
3. On the **View** menu, click C/AL Globals.
4. In the **C/AL Globals** window, on the **Variables** tab, add a new variable for the filter. The DataType of the variable is Text.
5. On the **View** menu, click C/AL Code.
6. In the C/AL Editor, in the OnPreReport trigger, add code to obtain the filters. For example, if you have a filter on the **Customer** table and the filter variable that you created is named CustFilter, then add the following code.

```
CustFilter := Customer.GETFILTERS;
```

7. In Report Dataset Designer, under the Dataltem for the table to which a filter can be applied, add a new line.
8. In the **Data Type** field of the new line, select Column.
9. In the **Data Source** field of the new line, enter the name of the filter variable.
10. On the **File** menu, select Save.
11. In the **Save** dialog box, verify that Compiled is checked, and then click **OK**.
12. On the **View** menu, select Layout.
13. In Microsoft Visual Studio, in the Report.rdlc file, add a text box to display the filter header.
14. Select the filter header text box, and then on the **View** menu, select the **Properties** window.
15. In the **Properties** window, expand the **Visibility** field.
16. In the **Hidden** field, select **<Expression...>** from the drop-down list.
17. In the **Expression** window, create a conditional statement. For example, the following are the conditional statements for the two filter text boxes on report 113.

```
=IIF(First(Fields!CustFilter.Value) = "",TRUE,FALSE)
```

```
IIF(First(Fields!ItemLedgEntryFilter.Value) = "",TRUE,FALSE)
```

18. Save and compile the report.

Build the Request Page

After you upgrade the report to Microsoft Dynamics NAV 2013, you must create the request page.

A request page runs before the report starts to execute. You can use a request page to specify options and filters for a report.

You can design the **Options FastTab** of a request page by using **Request Options Page Designer** in the development environment. To access Request Options Page Designer, open Report Dataset Designer, click View, and then select Request Page.

You can design the filters on request pages by using the following report properties:

- ReqFilterHeading
- ReqFilterHeadingML
- ReqFilterFields

You can also select the following actions on a request page:

- Print
- Preview
- Cancel
- Help

The fields that you define as **ReqFilterFields** are shown on the request page. Additionally, an end-user can add more fields on which to filter to the request page.



Best Practice: *It is recommended that you add fields that the end-users of the report will frequently set filters on.*

By default, for every data item in the report, a FastTab for defining filters and sorting is created on the request page. To remove a FastTab from a request page, do not define any **ReqFilterFields** for the data item and set the **DataItemTableView** property to define sorting. The request page is displayed but there is no tab for this data item.

If a **DataItemTableView** is not defined, then users can select a sort field and sort order at run time.

If you set **UseReqPage** to No., then the report starts to print as soon as it is run. Here, end-users cannot cancel the report run. The report can be canceled. However, some pages might print.

In a complex report that uses data from several tables, the functionality can depend on a specific key and sort order. Design your reports so that end-users cannot change the sort order in a way that affects the functionality of the report.

Determining which Reports to Upgrade


This lesson will explain how to log report usage in Microsoft Dynamics NAV.

How to Log Report Usage

You can use report usage logging to identify both frequently and rarely used reports in a customer installation. This information can be useful when you must determine which Classic reports should be transformed to RDLC reports. This only works in previous version(s) of Dynamics NAV. A custom development could be an alternative to this solution.


To log report usage, you must have a hotfix installed. It can be downloaded here:

Microsoft Dynamics NAV 5.0: **KB2575296**

 [KB2575296](#)

<http://go.microsoft.com/fwlink/?LinkId=267332>

Dynamics NAV 2009: **KB2558650**

 [KB2558650](#)

<http://go.microsoft.com/fwlink/?LinkId=267333>

After you install the required hotfix, to set up report usage logging, follow these steps.

1. Create new table to be used to log report usage.

Code example

```
OBJECT Table 50000 Report Log
{
OBJECT-PROPERTIES
{
Date=11-06-12;
```

```
Time=12:36:58;
Modified=Yes;
Version List=NAV6;
}
PROPERTIES
{
}
FIELDS
{
{ 1 ; ;No. ;Integer ;AutoIncrement=Yes;
MinValue=1 }
{ 2 ; ;User ID ;Code50 ;TableRelation="User Role"."Role ID";
CaptionML=ENU=User ID }
{ 3 ; ;Report ID ;Integer ;CaptionML=ENU=Report ID }
{ 4 ; ;Report Name ;Text249 ;FieldClass=FlowField;
CalcFormula=Lookup(AllObjWithCaption."Object Caption" WHERE (Object
Type=CONST(Report),
Object ID=FIELD(Report ID)));
CaptionML=ENU=Report Name }
{ 5 ; ;Date Time ;DateTime }
}
KEYS
{
{ ;No. ;Clustered=Yes }
}
FIELDGROUPS
```

```
{  
  
}  
  
CODE  
  
{  
  
BEGIN  
  
END.  
  
}  
  
}
```

2. Open Codeunit 1.
3. Open "C/AL Globals".
4. Browse to "Functions".
5. Create a new function that uses the Name: OnReportRun.
6. Open Properties and change ID to 120.
7. Now open Locals and create the parameter= ReportId with Type=Integer.
8. Select the **Variables** tab and create a ReportLog.

Now you must write the code for this new trigger.

9. Open C/AL Editor and browse to the end.
10. In OnReportRun write the following code:

Code example

```
ReportLog."User ID" := USERID;  
  
ReportLog."Report ID" := ReportId;  
  
ReportLog."Date Time" := CURRENTDATETIME;  
  
ReportLog.INSERT;
```

11. Now restart the Classic client.
12. Run several reports.
13. Now run the **Report Log** table to view the result.

Lab 6.1: Redesigning a Customer List Report

Scenario

Simon is a Microsoft Certified Partner working for CRONUS International Ltd. He is asked to upgrade the Classic version of the **Customer List** report that CRONUS used in earlier versions of Microsoft Dynamics NAV.

Exercise 1: Importing and Upgrading the Report

Exercise Scenario

Simon will upgrade the Classic report. This process deletes the request form, automatically creates an RDLC report layout, and upgrades the report to be valid for Microsoft Dynamics NAV 2013. Simon uses the following procedure to import a Classic-only report and upgrade it.

Task 1: Import the Report

High Level Steps

1. Import a Classic report.

Detailed Steps

1. Import a Classic report.
 - a. In the development environment, on the **Tools** menu, click **Object Designer**.
 - b. In Object Designer, on the **File** menu, click **Import**.
 - c. In the **Import Objects** window, browse to the report object that contains only the Classic report layout (R101.fob), and then click **Open**.
 - d. If you are prompted, click **OK** to open the Import worksheet to resolve any conflicts. Then click **OK** in the Import Worksheet to import the report.

Task 2: Upgrade the Report

High Level Steps

1. Upgrade a Classic report.

Detailed Steps

1. Upgrade a Classic report.
 - a. In **Object Designer**, click **Report**, select **Report 101, Customer List**. This is the list that you imported in the previous step, and then click **Upgrade Reports**.
 - b. In the dialog window, click **Yes** to confirm that you want to upgrade the report.

- c. In the **Tools** menu, click **Compile** to compile the report that you upgraded in the previous step.

Exercise 2: Modifying the Customer List Report RDLC Layout

Exercise Scenario

Hidden fields in the RDLC report layout will cause formatting problems when Simon prints the report because these fields are outside the width of standard paper. To eliminate the formatting problems, Simon must move these hidden fields to another location on the report that is within the width of standard paper.

Task 1: Open the Report in Visual Studio Report Designer

High Level Steps

1. Open the report in Visual Studio Report Designer.

Detailed Steps

1. Open the report in Visual Studio Report Designer.
 - a. In **Object Designer**, select report **101**, click **design**, click **View**, and then click **Layout**.
 - b. Visual Studio Report Designer opens.

Task 2: Move the hidden fields

High Level Steps

1. Move the hidden fields.

Detailed Steps

1. Move the hidden fields.
 - a. In Visual Studio, in the Report.rdlc file that is created, in the Body section of the layout, select the following hidden fields on the right side of the table:

=FIELDS!COMPANYNAME.VALUE
=FIELDS!Customer_ListCaption.VALUE
=FIELDS!CurrReport_PAGENOCaption.Value

- b. Right-click the selected fields, and then click **Cut**.
- c. Right-click the row handle of the last row, and then click **Insert Row Below** to insert a new row in the table.
- d. Right-click the first field in the new row, and then click **Paste**.
- e. On the **View** menu, click the **Properties** window.

- f. Select the new row. In the **Properties** window, under Layout, change the value of the Height property to 0.2 cm. The height 0.2 cm is sufficient to view the row in the layout, and it will not cause formatting problems when you print the report.
- g. This report has unsupported on the section trigger for Customer, Header (3). You must add the functionality for this code elsewhere in the report so that it will run in Dynamics NAV 2013. This is because code in a section trigger is not supported and will not be migrated.

Task 3: Add the Functionality from the Section Trigger

High Level Steps

1. Add the functionality from the section trigger.

Detailed Steps

1. Add the functionality from the section trigger.
 - a. In Visual Studio, on the **View** menu, click the **Properties** window.
 - b. In the Report.rdlc file, in the Body section of the layout, select the field at the top of the layout with the following value:

```
=First(Fields!Customer_TABLECAPTION_____CustFilter.Value)
```

- c. In the drop-down list for the **Hidden** field, select <Expression>.
- d. In the **Expression** window, enter the following expression:

```
=IIF(Fields!CustFilter.Value<>"",False,True)
```

- e. Click **OK**.

Exercise 3: Add Captions and Labels to the Report

Exercise Scenario

In the upgraded report the captions are a part of the dataset. Mort will now optimize the report that has the property Include Caption and uses labels.

Task 1: Change the Captions of the Report

High Level Steps

1. Enable the Include Caption property.
2. Remove the 'old' caption fields from the dataset.
3. In Visual Studio Report Designer replace the old caption fields by using the new captions.

Detailed Steps

1. Enable the Include Caption property.
 - a. In the Report Dataset Designer select the field that has the name **Customer_No_** and then click **Include Caption** to enable it.
 - b. Repeat this step for the following fields:
 - Customer_Customer_Posting_Group_
 - Customer_Customer_Disc_Group_
 - Customer_Invoice_Disc_Code_
 - Customer_Customer_Price_Group_
 - Customer_Fin_Charge_Terms_Code_
 - Customer_Payment_Terms_Code_
 - Customer_Salesperson_Code_
 - Customer_Currency_Code_
 - Customer_Credit_Limit_LCY_
 - Customer_Balance_LCY_
 - Customer_Contact
 - Customer_Phone_No_
 - Customer_Balance_LCY__Control42
2. Remove the 'old' caption fields from the dataset.
 - a. In the Report Dataset Designer remove the following lines with the name:
 - Customer_No_Caption
 - Customer_Customer_Posting_Group_Caption
 - Customer_Customer_Disc_Group_Caption
 - Customer_Invoice_Disc_Code_Caption
 - Customer_Customer_Price_Group_Caption
 - Customer_Fin_Charge_Terms_Code_Caption
 - Customer_Payment_Terms_Code_Caption
 - Customer_Salesperson_Code_Caption
 - Customer_Currency_Code_Caption

- Customer_Credit_Limit_LCY_Caption
 - Customer_Balance_LCY_Caption
 - Customer_ContactCaption
 - Customer_Phone_No_Caption
3. In Visual Studio Report Designer replace the old caption fields by using the new captions.
 - a. In Visual Studio Report Designer, in the first row of the table, right-click the text box that contains the expression: =First(Fields!Customer_No_Caption.Value) and replace it with the following expression:
Parameters!Customer_No_Caption.Value
 - b. Repeat this step for all text boxes in the first row of the label by using the corresponding parameter.
 - c. Repeat this step also the two textboxes on the first column containing the fields:
 - i. =Fields!Customer_ContactCaption.Value becomes
=Parameters!Customer_ContactCaption.Value
 - ii. =Fields!Customer_Phone_No_Caption.Value becomes
=Parameters!Customer_Phone_No_Caption.Value

Task 2: Use Labels in the Report

High Level Steps

1. In the Report Label Designer add labels.
2. In Visual Studio Report Designer replace the old caption fields by using the new labels.

Detailed Steps

1. In the Report Label Designer add labels.
 - a. Add a label with the Name: **Total_LCY_CaptionLbl** and Caption: **<Total (LCY)>**.
 - b. Remove the corresponding line in the Report Dataset Designer.
2. In Visual Studio Report Designer replace the old caption fields by using the new labels.
 - a. In Visual Studio Report Designer change the expression of the text box that contains: =First(Fields!Total_LCY_Caption.Value) to Parameters!Total_LCY_CaptionLbl.Value

Exercise 4: Saving the RDLC Report Layout

Exercise Scenario

Simon must save the RDLC layout, and he must also save and compile the changes in the Microsoft Dynamics NAV 2013 development environment.

Task 1: Save the RDLC Layout

High Level Steps

1. Save the report.

Detailed Steps

1. Save the report.
 - a. In Visual Studio, on the **File** menu, click **Save report.rdlc**.
 - b. Close Visual Studio.
 - c. In the Microsoft Dynamics NAV 2013 development environment, select an empty line in Report Dataset Designer. You receive the following message:

The layout of report id: '101' is changed by another application.

Filename of the changed RDLC: <filename>

Do you want to load the changes?

- d. Click **Yes** to load the changes that you made to the RDLC report layout.
- e. On the **File** menu, click **Save**.
- f. In the **Save** dialog box, select **Compiled**, and then click **OK**.
- g. Close the report in Object Designer.

Lab 6.2: Redesigning a Customer - Order Summary Report

Scenario

Simon is a Microsoft Certified Partner working for CRONUS International Ltd. He is asked to upgrade the Classic version of the **Customer - Order Summary** report that CRONUS used in the earlier version of Microsoft Dynamics NAV.

Objectives

- Import and upgrade a report that has only a Classic report layout to a Microsoft Dynamics NAV 2013 report that has an RDLC layout.
- Modify the RDLC report layout.
- Save the layout and compile the report.

Exercise 1: Importing and Upgrading the Report

Exercise Scenario

Simon will now upgrade the Classic report. This process will delete the request form, automatically, create an RDLC report layout, and upgrade the report to be valid for Microsoft Dynamics NAV 2013.

Task 1: Import and Upgrade a Classic Report

High Level Steps

1. Import a Classic report.

Detailed Steps

1. Import a Classic report.
 - a. In the development environment, on the **Tools** menu, click **Object Designer**.
 - b. In Object Designer, on the **File** menu, click **Import**.
 - c. In the **Import Objects** window, browse to the report object that contains only the Classic report layout for report **R123456707**, and then click **Open**.
 - d. If you are prompted, click **OK** to open the Import worksheet to resolve any conflicts.

Task 2: Upgrade a Classic Report

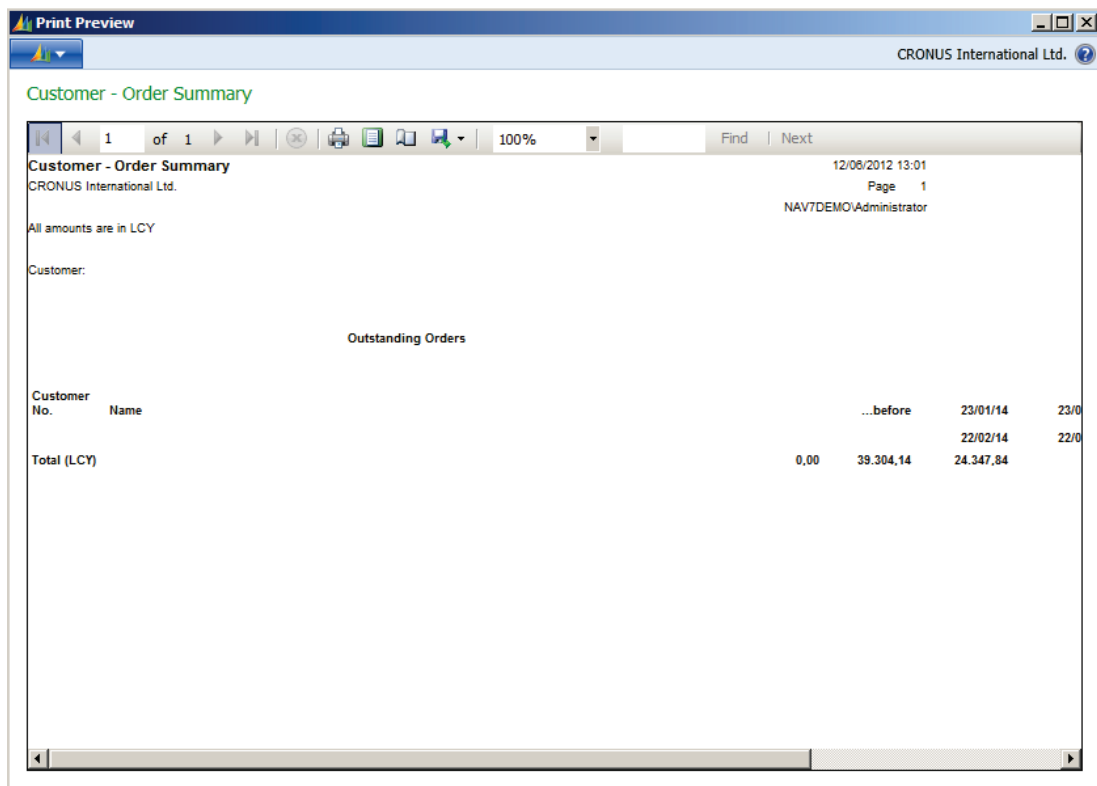
High Level Steps

1. Upgrade a Classic report.

Detailed Steps

1. Upgrade a Classic report.
 - a. In Object Designer, click **Report**, select **Report 123456707, Customer Order Summary** (this is the report that you imported in the previous step), and then click **Tools, Upgrade Reports**.
 - b. In the dialog window, click **Yes** to confirm that you want to upgrade the report.
 - c. In the **Tools** menu, click **Compile** to compile the report that you upgraded in the previous step.

The "Customer – Order Summary Print Preview" illustration shows the RDLC report layout for **report 123456707, Customer Order Summary**, after you upgrade the report but before you perform any manual modifications.



Exercise 2: Modifying the Customer - Order Summary Report RDLC Layout

Exercise Scenario

The RDLC report layout for the **Customer Order Summary** report that is created when Simon upgraded the report does not contain all the lines that are required to be displayed in the report. Simon must make some modifications to the report layout.

Task 1: Open the Report in Visual Studio Report Designer

High Level Steps

1. Open the report in Visual Studio Report Designer.

Detailed Steps

1. Open the report in Visual Studio Report Designer.
 - a. In Object Designer, select report 123456707, click **Design**, click **View**, and then click **Layout**.
 - b. Visual Studio Report Designer opens.

Task 2: Move Header Fields Out of the Table

High Level Steps

1. Move header fields out of the table.

Detailed Steps

1. Move header fields out of the table.
 - a. In Visual Studio, in the Body of the report.rdlc file, move the table down in the layout to provide space for two fields in the table.
 - b. Select the **First(Fields!Outstanding_OrdersCaption.Value)** field, right-click the field, and then click **Cut**.
 - c. Right-click the area in the table and then click **Paste** to paste the **First(Fields!Outstanding_OrdersCaption.Value)** value in a text box that is in the table.
 - d. Select the **First(Fields!EmptyString.Value)** field, right-click, and then select **Cut**.
 - e. Right-click the area that is in the table, under the text box in which you performed the previous past operation and then click **Paste** to paste the **First(Fields!EmptyString.Value)** value in a text box.
 - f. Select the first two rows of the table from which you cut fields. Right-click the row handle of the two rows, and then select **Delete Rows**.

The layout that is created automatically contains some fields that you must move. The first field is the **SalesAmtOnOrderLCY_3.Value** field. This field is in the wrong location. The other fields that can cause problems in the printed layout of the report are the three hidden fields.

Task 3: Update the Layout

High Level Steps

1. Update the layout.

Detailed Steps

1. Update the layout.
 - a. In Visual Studio, on the last row of the table, right-click the field that uses the value **=Sum(Fields!SalesAmtOnOrderLCY_3.Value)**, and then click **Cut**.
 - b. Right-click the empty field in the last row of the table that is between the **=Sum(Fields!SalesAmtOnOrderLCY_2.Value)** and **=Sum(Fields!SalesAmtOnOrderLCY_4.Value)** fields, and then click **Paste**.
 - c. Expand the third row of the table.
 - d. Decrease the width of the third column of the table.
 - e. Select the three hidden fields on the third row of the table. These hidden fields are in the rightmost columns of the table. The values in the hidden fields are as follows:

=Fields!COMPANYNAME.Value
=Fields!Customer__Order_SummaryCaption.Value
=Fields!CurrReport_PAGENOCaption.Value

- f. Right-click the three hidden fields, and then click **Cut**.
- g. Right-click the empty field in the first column of the second row of the table, and then click **Paste** to paste the three fields.



Note: It is recommended that you move these hidden fields from the right side of the table in all reports for which you create an RDLC report layout. Although the fields are hidden, white space is still allocated on the page for the fields. In some cases, this causes problems with the print layout because of the width of the paper and the width of the table.

- h. Select the last three columns of the table from which you cut the hidden fields.
- i. Right-click the three columns, and then click **Delete Columns**.

The next step is to add detail lines and add data to those lines in the table so that the same information that is in the Classic client report layout is in the RDLC report layout.

Task 4: Add Data to Detail Lines

High Level Steps

1. Add data to detail lines.
2. Obtain the sum of the lines in the dataset for each customer, by adding a grouping property. Add the grouping property for each detail row that you added.
3. Use the request page to select if the values should be displayed in local currency (LCY) or in the company's own currency. The first detail line that you added in step 3 shows the values in the local currency. The second detail line shows the values in the company's currency. Then select which values will be printed in the report that is based on the selection in the request page, you must add a conditional statement to the visibility property. Now set the visibility property.
4. Show the All Amounts are in the LCY caption only when the values are displayed in local currency.
5. Show the customer filter caption only when a customer filter is applied.

Detailed Steps

1. Add data to detail lines.
 - a. Click in the table to make it active.
 - b. Right-click the blank detail row, and then click **Insert Row > Inside Group - Below**.
 - c. Add the following values to the first blank detail line:
 - i. Customer_No_
 - ii. Customer_Name
 - iii. Leave the third column blank.
 - iv. SalesAmtOnOrderLCY_1_Control31
 - v. SalesAmtOnOrderLCY_2_Control32
 - vi. SalesAmtOnOrderLCY_3_Control33
 - vii. SalesAmtOnOrderLCY_4_Control34
 - viii. SalesAmtOnOrderLCY_5_Control35
 - ix. SalesOrderAmountLCY_Control36

To add these values, follow these steps.

- d. Right-click the field of the detail line, and then click **Expression**.
- e. In the **Expression** window, under Category, select Fields (DataSet_Result).

- f. Under Field, double-click the value from the previous list.
- g. An expression is added to the text box.

For each amount value, the data that is shown must be the sum of the lines in the dataset.

- h. To create the sum, change the expression to Sum(<expression>).
For example, change
=Fields!SalesAmtOnOrderLCY_1_Control31.Value to
=Sum(Fields!SalesAmtOnOrderLCY_1_Control31.Value)
- i. Click **OK**.
- j. For each value, you must modify the font and the format.

To format the data, follow these steps.

- k. Right-click the field, and then click **Properties**.
 - l. In the **Text box Properties** window, click the **Number** tab.
 - m. If the field is an amount field, click **Custom** in **Category** and next to **Custom format**, click the Expression button.
 - n. In the **Expression** window, under Category, select Fields (DataSet_Result).
 - o. Under Fields, select the Format value of the field that you selected. For example, select SalesAmtOnOrderLCY_1_Control31Format.
 - p. Click **OK**.
 - q. In the **Textbox Properties** window, click the **Font** tab.
 - r. Under Size, enter 7pt.
 - s. Add the following values to the second blank detail line:
 - i. Customer_No_
 - ii. Customer_Name
 - iii. Sales_Line__Currency_Code_
 - iv. SalesAmtOnOrder_1_
 - v. SalesAmtOnOrder_2_
 - vi. SalesAmtOnOrder_3_
 - vii. SalesAmtOnOrder_4_
 - viii. SalesAmtOnOrder_5_
 - ix. SalesOrderAmount
2. Obtain the sum of the lines in the dataset for each customer, by adding a grouping property. Add the grouping property for each detail row that you added.
- a. Select the row.
 - b. In the **Row Groups** window, select Group Properties.

- c. In the **Properties** window, click the **Add** button.
 - d. In the **Group on** field select Fields!Customer_No_.Value, and then click **OK**.
3. Use the request page to select if the values should be displayed in local currency (LCY) or in the company's own currency. The first detail line that you added in step 3 shows the values in the local currency. The second detail line shows the values in the company's currency. Then select which values will be printed in the report that is based on the selection in the request page, you must add a conditional statement to the visibility property. Now set the visibility property.
- a. Select the first of the two rows that you added. This row has the LCY values. In the **Properties** window, expand Visibility. In the drop-down list for Hidden, select <Expression>.
 - b. In the **Expression** window, change the expression to the following so that the row is hidden if Fields!PrintAmountsInLCY.Value is not true:

```
= not first(Fields!PrintAmountsInLCY.Value)
```

- c. Select the second of the two rows that you added. This row does not have the LCY values.
- d. In the **Properties** window, expand Visibility.
- e. In the drop-down list for **Hidden**, select <Expression>.
- f. In the **Expression** window, change the expression to the following:

```
=first(Fields!PrintAmountsInLCY.Value)
```

4. Show the All Amounts are in the LCY caption only when the values are displayed in local currency.
- a. Right-click the field
=First(Fields!All_Amounts_are_in_LCYCaption.Value), and then select Properties.
 - b. In the **Textbox Properties** window, click the **Visibility** tab.
 - c. Under Initial visibility, select Expression, and then enter the following expression in the text box:

```
=not First(Fields!PrintAmountsInLCY.Value)
```

- d. Click **OK**.

5. Show the customer filter caption only when a customer filter is applied.
 - a. Right-click the field **=First(Fields!Customer_TABLECAPTION_____CustFilter.Value)**, and then click **Properties**.
 - b. In the **Textbox Properties** window, click the **Visibility** tab.
 - c. Under Initial visibility, select Expression, and then enter the following expression in the text box:

```
=First(Fields!CustFilter.Value, "DataSet_Result")= ""
```

Exercise 3: Saving the RDLC Report Layout

Exercise Scenario

Simon must now save the RDLC report layout, and he must also save and compile the changes in the Microsoft Dynamics NAV 2013 development environment.

Task 1: Save the Report

High Level Steps

1. Save the report.

Detailed Steps

1. Save the report.
 - a. In Visual Studio, on the **File** menu, click **Save report.rdlc**.
 - b. Close Visual Studio.
 - c. In the Microsoft Dynamics NAV 2013 development environment, select an empty line in the report in Object Designer. You receive the following message:

The layout of report id: '123456707' is changed by another application.

Filename of the changed RDLC: <filename>

Do you want to load the changes?

- d. Click **Yes** to load the changes that you made to the RDLC report layout.
- e. On the **File** menu, click **Save**.
- f. In the **Save** window, select the **Compiled** check box, and then click **OK**.
- g. Close the report in Object Designer.

Module Review

Module Review and Takeaways

This module described the upgrade process for different report types:

- A Hybrid Report
- A Processing Only Report
- A Classic Report

Furthermore several tips, tricks and recommendations were provided concerning performance and optimization of the dataset and report layout.

The module also covered how to determine which reports to upgrade and concluded with practical labs and real life examples on how a List report and an Order Summary report should be upgraded.

Test Your Knowledge

Test your knowledge with the following questions.

1. Which report types cannot be upgraded to Microsoft Dynamics NAV 2013?
 - () Classic
 - () Hybrid
 - () Textual
 - () ProcessingOnly
2. Which menu item is used to upgrade a report to the Microsoft Dynamics NAV 2013 RoleTailored client?
 - () Tools > Upgrade Reports
 - () Tools > Report Upgrade
 - () View > Layout
 - () View > Upgrade Report

3. How is report usage logged in Microsoft Dynamics NAV?
- Report usage is logged automatically.
 - Report usage is logged through the change log functionality.
 - A hotfix must be installed , depending on the version of Microsoft Dynamics NAV, after which you can create a log table and modify a Codeunit 1.
 - Report usage cannot be logged in Microsoft Dynamics NAV.

Test Your Knowledge Solutions

Module Review and Takeaways

1. Which report types cannot be upgraded to Microsoft Dynamics NAV 2013?
 - Classic
 - Hybrid
 - Textual
 - ProcessingOnly
2. Which menu item is used to upgrade a report to the Microsoft Dynamics NAV 2013 RoleTailored client?
 - Tools > Upgrade Reports
 - Tools > Report Upgrade
 - View > Layout
 - View > Upgrade Report
3. How is report usage logged in Microsoft Dynamics NAV?
 - Report usage is logged automatically.
 - Report usage is logged through the change log functionality.
 - A hotfix must be installed , depending on the version of Microsoft Dynamics NAV, after which you can create a log table and modify a Codeunit 1.
 - Report usage cannot be logged in Microsoft Dynamics NAV

