

# Deltek Maconomy® 2.3 GA

## Upgrade Guide for Oracle

**January 30, 2017**

While Deltek has attempted to verify that the information in this document is accurate and complete, some typographical or technical errors may exist. The recipient of this document is solely responsible for all decisions relating to or use of the information provided herein.

The information contained in this publication is effective as of the publication date below and is subject to change without notice.

This publication contains proprietary information that is protected by copyright. All rights are reserved. No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, or translated into another language, without the prior written consent of Deltek, Inc.

This edition published January 2017.

© Deltek, Inc.

Deltek's software is also protected by copyright law and constitutes valuable confidential and proprietary information of Deltek, Inc. and its licensors. The Deltek software, and all related documentation, is provided for use only in accordance with the terms of the license agreement. Unauthorized reproduction or distribution of the program or any portion thereof could result in severe civil or criminal penalties.

All trademarks are the property of their respective owners.

# Contents

1	Introduction .....	1
1.1	Main Upgrade Phases .....	2
2	Prerequisites.....	3
2.1	Upgrade Kit 2.3 GA.....	3
2.2	Hardware and Resources .....	3
2.3	Upgrade Preconditions .....	3
2.4	Special Actions Prior to Upgrade.....	4
3	Upgrade Overview .....	15
4	Check Upgrade Preconditions.....	17
4.1	Check Preconditions for Upgrade.....	17
4.2	Remove Old BPM Database Objects .....	17
4.3	Change the Default Linesize Value for Oracle .....	18
4.4	Integration with Talent Management .....	18
5	Create the Unicode Database .....	19
6	Installation.....	20
6.1	Install the New Tools (TPU).....	20
6.2	Install the New Application (APU).....	20
7	Unicode Conversion .....	21
7.1	Oracle Conversion .....	21
7.2	Convert Text Strings.....	34
7.3	Restore String Length.....	35
8	Delete Old Components .....	36
8.1	Delete Old Constraints.....	36
8.2	Dump Changes to the Standard Index Set.....	36
8.3	Delete Old Indexes .....	37
8.4	Dump Materialized View Definitions .....	38
8.5	Delete Views .....	38
8.6	Delete Portal Standard Components.....	39
9	Update the Database Schema .....	41
9.1	Expand Relations.....	41
9.2	Delete Unused Columns.....	42
9.3	Convert Terminology of Setup and Production Data .....	43
9.4	Create Access Control Views .....	43
10	Update Business Logic and Components .....	45
10.1	W_19_0 Upgrade.....	45

10.2	Update Common Relations .....	47
10.3	Reinstall Standard Layouts for all Platforms .....	48
10.4	Create Indexes for Data Conversion .....	49
10.5	Convert the Portal Database .....	49
10.6	Convert Data to the New Version .....	50
10.7	Remove Indexes for Data Conversion.....	52
11	Validate Custom Components.....	53
11.1	Validate Layouts .....	53
11.2	Create Constraints.....	54
11.3	Install and Upgrade the Portal.....	54
11.4	Recreating BPM Objects .....	57
11.5	Clean Up .....	57
12	Large Databases .....	58
12.1	Drop Constraints (does not apply to 2.3 GA or LA .....	58
12.2	Dump Changes to Standard Indexes .....	58
12.3	Drop Indexes .....	58
12.4	Expand Script .....	59
12.5	Create Index for Data Conversion .....	59
12.6	Convert_Data_From_9.0 .....	59
12.7	Remove Indexes from Data Conversion.....	60
13	Appendix: Error Messages .....	61
13.1	Oracle Considerations .....	61
13.2	Using UNDO .....	61
13.3	Maconomy Errors .....	62
13.4	MConfig Errors.....	62
14	Appendix: Legacy Portal Conversion .....	63
14.1	Export Data for Conversion .....	63
14.2	Convert Exported Data .....	64
14.3	Import Converted Data .....	65
15	Appendix: NameChanger Tool .....	66

# 1 Introduction

This document describes how to upgrade from Maconomy W 9.0, service pack 5 or later (all using database schema version 2) to Maconomy 2.3 GA tools version 17. It also describes how to upgrade from Maconomy X, X+, X1 or 2.0 any service pack level, to Maconomy version 2.3 GA.

The procedure for upgrading from Maconomy W 10.0 to Maconomy 2.3 GA is similar to the procedure used to upgrade from Maconomy 9.0 to Maconomy X1. However, the procedure is substantially different from earlier procedures because these procedures also include converting the Maconomy terminology.

If you are upgrading from a version that is earlier than Maconomy W 9.0, service pack 5, you must upgrade to at least Maconomy W 9.0 service pack 5 before you can use the procedures described in this document.

This document describes each step needed for the upgrade. All examples are based on one company, and this company has the shortname **myshort**. The preferred version database is Oracle 11g. This document does not describe how to upgrade database versions.

You must execute SQL scripts using the commands that are installed by MConfig. For an Oracle database, the command is StartOracle.cmd.

The directory names used for the old application in the examples are the MConfig standard names.

The following table provides an overview of external and internal Maconomy versions.

Maconomy Version	Tools Version
9.0	9.0
X	10.0
X+	11.0
X1	12.0
2.0	15.0
2.1	16.0, SP0,SP1
2.1.1	16.0 SP2
2.1.4	16.0 SP4
2.2	17
2.3	19

## 1.1 Main Upgrade Phases

The steps covered in this guide can be divided into the following groups:

- Install tools and the application
- Delete old components
- Expand and convert databases and data
- Convert to Unicode
- Create components
- Install the new Portal
- Convert source files
- Install source files

## 2 Prerequisites

Be sure that the following prerequisites are met before you begin the upgrade process.

### 2.1 Upgrade Kit 2.3 GA

You must obtain the Upgrade Kit for 2.3 GA as well as the TPU and the APU. This document is included in the Upgrade Kit.



Always look for the latest and updated version of the Upgrade Kit on the [Deltak Maconomy Download Site](#) before beginning an upgrade.

The tools that the Upgrade Kit includes may provide automatic conversion of kernel strings and tools for converting data in the database. The Upgrade Kit consists of several SQL scripts to be run and tools to modify export files and localize customized data in the database, such as Windows and Print Layouts (MDL and MPL).

Some of the functionality that the Upgrade Kit provides is available only on Windows platforms. Thus, access to a Windows computer is required during the upgrade. The procedure is outlined in detail in the upgrade steps where this is required.

The application that you are upgrading must be upgraded to W 9.0 service pack 5 or later using the appropriate upgrade guide before following the procedures in this document.

### 2.2 Hardware and Resources

When running the Unicode conversion tools on an Oracle database it is recommended to have at least 20 gigabytes PGA size. Having a smaller PGA may result in severe performance degradation. It is not advisable to run the Oracle tools on virtual hardware unless the virtual hardware can provide comparable I/O performance to that of a physical machine. If both the new and old databases are on the same server, and there is a limited amount of memory available, it is recommended to allocate the most PGA to the Unicode database.

The recommended storage configuration is RAID 10. This is not required; however, the tools are I/O bound, and the database conversion time heavily depends on the I/O throughput.

A minimum of 4 physical CPU cores, 2.4 Ghz or higher, is required, but 12 cores or more are recommended for larger databases.



For Oracle: Check that disk space is available for the original database as well as the new Unicode database. Expect the database to grow up to 20% during the upgrade.

### 2.3 Upgrade Preconditions



Check that the UpgradePreCondition script runs without any errors or warnings before continuing the upgrade procedure. If you encounter errors, **STOP** and fix all errors before you continue.

## 2.4 Special Actions Prior to Upgrade

### 2.4.1 Ensure Instancekeys

Ensure that the **userinformation** table has **instancekeys** prior to starting the upgrade. The instance keys are used to link user, user role, and user dialog group together. This enables the administrator user to expand the data base schema to the new version. Otherwise the upgrade will fail, as this is executed by the administrator user.

To enable instance keys, follow these steps prior to upgrade:

1. Go to **Setup > System Setup > Database Relations**.
2. Select the Relation Name **User**.
3. Ensure the **Instance Keys Applied** field shows **Yes**. If so, no further action is needed.
4. If the **Instance Keys Applied** field shows **No**, click the action **Initialize Instance Keys**.

### 2.4.2 MyxL Tool for Customized Specification Languages Conversions

Between the major versions of Maconomy, syntax updates occur that may not be backwards compatible with some of the specification languages. This occurs when specification files delivered with the product in the new version are automatically converted to the new syntax, but customized specification files are not. To correct this you must run a conversion of the relevant files to convert them to the new syntax. The MyxL tool facilitates the conversion of Maconomy's specification files, including MDML, MWSL, MMSL, MNSL, and MCSL.

To convert specification files, you must:

- Check out all the specification files that are present in GitHub
- Convert the files using the MyxL tool
- Check the files back in to GitHub

### Before You Begin

Make sure that you are running the Java version corresponding to the branch with which you are working, which is Java 8 for 19 and onwards. Verify that the correct version is picked up by using the typing 'java -version' at the command prompt.

If needed, install a correct 32-bit JDK.

### Convert Specification Files

#### Convert Files Using MyxL Tool

1. Go to the TPU in the / **bin** directory and find the MyxL tool.
2. Extract the tool to a local directory.



As you extract, notice that there are **Convert{version}.exe** and **Convert{version}.ini** files present in the main directory, where **{version}** corresponds to the branch from which the tool was built. Check the version to verify that it corresponds to the correct TPU.

3. Supply various parameters to successfully run the conversion. The parameters are provided either in the Convert-19.0.101.0.ini file, or as command line arguments if the Converter is run from the command line.



- **convertdirectory** - *Mandatory*. The path to the file or directory to convert. If no other output method is specified, all files in this directory (or this file) are converted in place.
  - **writeconversion** - *Optional*. If this parameter is not set, no converted output is generated. Only a summary is written to the console and a summary report is generated. It is not set by default to prevent conversion errors. Must be explicitly set to write the conversion changes.
  - **outputdirectory** - *Optional*. A path to a directory to write the converted files to, if **writeconversion** was set. If this parameter is set the original files in the 'convertdirectory' are not altered.
  - **consolepreview** - *Optional*. If this parameter is set the result of the conversion is written to the console and a summary report is generated, if **writeconversion** was set.
  - **convertreadonly** - *Optional*. If this parameter is set read-only files are converted.
  - **debuginfo** - *Optional*. If this parameter is set debug information such as stack trace is written to the error section of the summary report.
  - **logconfigpath** - *Optional*. You can specify a path to a log configuration file here. As default the logback file provided with the Converter is used.
4. Check to verify the updated syntax of the specification language. This includes checking to ensure that all the breaking changes were fixed and no unnecessary conversions were made (meaning, that the tool does exactly what it is supposed to do and no more).

The output method precedence is:

- **outputdirectory** (**writeconversion** was set)
- **consolepreview** (**writeconversion** was set)
- **in place conversion** (none of the above parameters set and **writeconversion** was set)



The tool takes different actions depending on what parameters are supplied. If both **outputdirectory** and **consolepreview** are set, then the action associated with **outputdirectory** takes precedence.

---

5. Submit the changes back to GitHub.



#### **Ensure Correct Conversion**

Technical Consultants are responsible for checking that the conversion is correct before submitting the changes back to GitHub.

#### **If Errors Occur**

At times, errors occur during the conversion process. If you experience errors you cannot resolve, refer to the Kona conversation ([link below](#)) or create a support case attaching the error message and possibly the file that caused the error when converting.

---

## Example

The following example shows a change for 2.3 for MDML. The tool removes the **titleValue**, **firstTitleValue**, **secondTitleValue**, **titleSource**, **firstTitleSource** and **secondTitleSource** attributes from various elements and replaces those with **title** attributes that allow the use of placeholder expressions.

If the customized layouts do not include the removed attributes listed above, then no conversion is necessary.

**Note:** See the *Delttek Maconomy Language Quick Reference MDML* document for the 2.3 release for details.

## Example (Windows)

1. Insert the parameters in the Convert-19.0.101.0.ini file:

```
--convertdirectory
D:\maconomy\w_19_0test\CustomizationDir
--outputdirectory
D:\test\outputdir
--convertreadonly
--writeconversion
```

2. Insert them when running from the command line:

```
>Convert-19.0.101.0 --convertdirectory
"D:\maconomy\w_19_0test\CustomizationDir" --outputdirectory
"D:\test\outputdir" --convertreadonly --writeconversion
```

## 2.4.3 New Global and Local Dimensions

Several global and local dimensions are added in Maconomy version 2.3. Maconomy assigns a default value to all of the new dimensions on existing transaction data. You can specify the default values before the upgrade. Maconomy uses [ - ] (dash) as the default value if you do not specify a default value before the upgrade.

A consultant can execute a script before you perform the technical upgrade. The script will indicate whether there is a correctly specified conversion table for the new dimensions. Fields and variables for the existing global and local dimensions have been renamed; the same naming strategy was used in all places. You must update layouts.

### Create Option Lists

In Maconomy 2.3 LA we introduced 13 new dimensions: Specification 4-10, Local Specification 4-10, and Local Account. Due to this, during the upgrade, we can create the new specification 4-10 and local specification 4-10 lists.

However, we need to know what list names and standard names should be assigned to these new lists. Because of this you can create new option lists with this information.

#### To create an option list for new Specification dimensions:

1. Go to **Single Dialogs > Set-Up > Set-up > Option Lists**.
2. Create a new option list, and in the **Option List No.** column, select **Specification Data Conversion**. Now, each line in the table represents a new specification list.
3. Add a new line in the table as needed.

For example, if you add a new line in the table with **Name** field set to **Specification5** and **Remarks 1** field set to **Test**, then during the upgrade a new **Specification5** list is created with the name specified in **Remarks 1** field, in this case, **Test**.

**Note:** Lists for new dimensions are created even if you do not create this options list. In this case, the **Specification 5** list **Name** will be set to [ - ] (dash).

#### To create an option list for new Local Specification dimensions:

1. Go to **Single Dialogs > Set-Up > Set-up > Option Lists**.
2. Create a new option list, and in the **Option List No.** column, select **Local Specification Data Conversion**.

Now, each line in the table represents a new local specification list.

3. Add a new line in the table as needed.

For example, if you add a new line in the table with **Name** field set to **LocalSpec5** and **Remarks 1** field set to **Test** and **Remarks 2** field set to **Test 2**, then during the upgrade a new **Local Specification 5** list is created with the name specified in **Remarks 1** field, in this case **Test** and standard name specified in **Remarks 2** field.

**Note:** Lists for new dimensions are created even if you do not create this options list. In this case, the **Local Specification 5** list **Name** will be set to **Local Spec. 5** list and **Standard Name** will be set to [ - ] (dash).

#### Example of Setup

Below is an example of the setup described above.

Option Lists x

List of Option Lists

Now showing 1 - 26 << Prev Next >>

Option List No.	Description	Remarks 1	Remarks 2	Remarks 3
1	Aging Principles	Aging Principles	Used in BPM reports and in ...	
2	Local Specification Data Conversion			

Option Lists

**Option List**

Option List No. Local Specif...

Description

**Remarks**

**Access Level**

Access Level

Description

Name

**User**

Created 14-12-2016 by Administrator

Changed 14-12-2016 by Administrator

Version 2

**Options**

Name	Description	Remarks 1	Remarks 2	Remarks 3
1 LocalSpec5		List Name	Standard Name	

## 2.4.4 Local Charts of Accounts

Maconomy has introduced support for local charts of accounts in version 2.3. Previously you could use one of the local dimensions as a local chart of accounts.

The new local chart of accounts dimensions are validated as a local account dimension, so you must, for example, enter a link from the local account to the global account. This is not a requirement for the local dimensions. Maconomy cannot convert a local dimension to a local account if there are local dimension values that do not have global account references.



Consultants can execute a script before performing the technical upgrade. The script displays any problems that might prevent the conversion. Those problems must be resolved before you can perform the upgrade.

To run the script:

1. Go to **SQL/Basics/LocalChartOfAccountsPreUpgrade.sql**.
2. Run **LocalChartOfAccountsPreUpgrade.sql**.

This file does not apply to upgrades from 2.3 LA to 2.3 GA.

You can still use one of the local dimensions as a local chart of accounts, but all standard reports have been changed, and all future development will assume that you use the local chart of accounts dimension, rather than a local dimension.

The purpose of the LCoA Pre-upgrade script is to verify that there will not be issues when upgrading to 2.3 in relation to the Local Chart of Account functionality introduced with this

release, ensure that dimension grouping setup is correct, and verify that deprecated fields are not used in some dialogs.

You must run this script prior to the upgrade to 2.3 LA, as it lists all the problems that must be corrected before the upgrade to avoid various problems during or after the upgrade.



Since any issues must be resolved prior to upgrade, we recommend that business consultants execute this script a few days prior to upgrade to ensure sufficient time to resolve any issues.

## Before You Begin

Before running the script you must change a setting in Preferences.

### To change the Preferences Setting:

1. Open the Maconomy client and go to **Edit > Preferences**.
2. From the dropdown list, select **Formats**.
3. In the **No. of decimals** field, select **2** from the dropdown.
4. Click **Apply**.
5. Click **OK**.

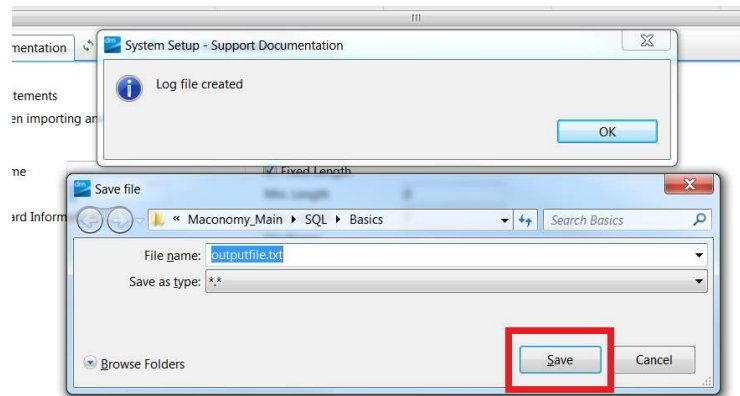
## Run the Script

### To run the script

1. Locate the script in the Upgrade Kit, which is located in the drive to which it was extracted. For example, if the Upgrade Kit was extracted to C: then find the script here:  
C:\UpgradeKit\_2.3GA\sql\LocalChartOfAccountsPreUpgrade.sql
2. Open the Workspace/Java client of the version you are upgrading. For example, if you are upgrading from 2.1 to 2.3 LA, run the script in your 2.1 Maconomy system.
3. Open the Maconomy client and go to **Setup > System Setup > Support Documentation** and choose "Import SQL Statements" in the filter.
4. In the card part **Statement No.** field, select **1**.
5. Click **Create Documentation**. A new dialog opens.
6. In the new dialog, browse to the location of the "LocalChartOfAccountsPreUpgrade.sql" file, and click **Open**.

The SQL script runs and executes various SQL statements in the database. After it is done executing the SQL statements, it produces an output file with the feedback (see image below).

7. Click **Save** to save this output file so that you can investigate and verify that the system setup satisfies the requirements for upgrade.



You must now evaluate the output and create options lists.

## Evaluate the Output

The script simply runs various SQL statements against the database to make sure that the system is ready for the upgrade.

The sample output below is color-coded to more easily indicate the content of the output and any action you must take.

- Yellow — Shows information on what the SQL statements are supposed to do.
- Green — Shows the exact SQL statements that will be run in the database.
- Red — Shows that the system has incorrect setup and / or issues that must be addressed before continuing the upgrade.

## Example Output

```
--
-----
-- Check to see if the option lists have been created for the new local spec. lists and new
-- specification entries
-----
--
select 'Could not find the option list with names for the new Local Spec. Lists' as Message
from SystemInformation
where not exists (select 1
                  from OptionList
                  where OptionListNumber = 'Local Specification Data Conversion')
union
select 'Could not find the option list with names for the new Specification entries' as Message
from SystemInformation
where not exists (select 1
                  from OptionList
```

where OptionListNumber = 'Specification Data Conversion')

```
|MESSAGE|
|-----|
|Could not find the option list with names for the new Local Spec. Lists|
|Could not find the option list with names for the new Specification entries|
```

2 rows processed

-----

#### SQL statement

In this case, we received two RED messages: **Could not find the option list with names for the new Local Spec. Lists** and **Could not find the option list with names for the new Specification entries**.

In this example, you create option lists and then you can add entries to the tables of those options lists, where you specify default names for the new dimensions. The [Create Option Lists](#) procedure is described in the Dimensions section above.

### 2.4.5 Reporting Structures

The structure of financial reports was previously specified in three different places, depending on the reporting technology. Built-in MPL reports were based on structure tables (account structures, location structures, and so on); portal universe reports were based on reporting hierarchies; and BPM reports were based on dimension groupings.

Structure tables and dimension groupings are replaced by reporting structures. Reporting structures are used for both built-in MPL reports and BPM reports.

Dimension groupings are, where possible, converted into a reporting structure. Structure tables and reporting hierarchies are not converted. You must create dimension groupings for all of the structure tables and reporting hierarchies that you need after the upgrade.

The new reporting structures are a tree structure (in the same format as finance budgets and job budgets) so it is important that lines in the existing dimension groupings are ordered in the same way as the report is printed. It is easier if you do the cleanup before the upgrade.

MPL reports and standard BPM reports are updated so that they are based on the new structures. You must update any BPM reports that you created.

### 2.4.6 References to Depreciated Fields on Customers and Vendors

The fields **Blocked**, **Blocked for Delivery**, and **Stop Payment** are removed. You cannot use target groups, linking rules, or approval hierarchies with references to these removed fields. These fields are replaced by “Active” fields, and a number of “Allowed” fields.

- **Active Status** replaces **Blocked** on Customers, Company Customers, Vendors, and Company Vendors.
- **Allow Payments** replaces **Stop Payment** on Vendors and Company Vendors.
- **Allow Delivery** replaces **Blocked for Delivery** on Customers and Company Customers.



These fields were added in version 2.1, so you can change the setup before you perform the upgrade, if you upgrade from version 2.1 or later.

## 2.4.7 Upgrading from Early Versions

If you upgrade to 2.3 GA from any version prior to Maconomy 2.1.1, you must take a special action and run an MConfig scriptlet to process the language upgrades in this release.



Run the MConfig scriptlet **prior to** installing the Service Pack APU.

**To run the MConfig scriptlet, follow these steps:**

1. Locate the MConfig scriptlet **ChangeToIsoLanguage.mcp**, which is located within the MConfigScriptlets directory of the TPU.
2. Run the script as follows:

```
<MConfig program> -r <scriptlet  
file> <application> <shortname> <password> [ <new enterprise language> ]
```



- Note the inclusion of the "-r" in the line above.
- Also note that on UNIX, you must specify the full path to the scriptlet file.

3. Enter an optional *<new enterprise language>* parameter as needed. For applications using one of the standard languages, such as the languages for which APUs contain Translations files, the scriptlet selects the corresponding new language, and the *<new enterprise language>* parameter can be omitted. Clients with applications using non-standard languages must specify the parameter.

## 2.4.8 Update Custom ControlHours.1.ms File

You should update the custom ControlHours.1.ms file in function getTableData(userValue). See 619035 for more information. Update to the following:

```
jobBudgetDialogRows = maconomy::sql(sprint('#S'SQL',  
format::toSql(userValue[userValueId].JobNumber),  
format::toSql(userValue[userValueId].BudgetType))).result.rows; // Secured  
  
select  
TaskListLine.TaskName, TaskListLine.TaskName as InstanceKey, TaskListLine.ActivityNumber,  
Activity.ActivityType, TaskListLine.Description as Text,  
nvl((select sum(nvl(NumberOf, 0))  
from JobBudgetLine  
where JobNumber = ^1  
and BudgetType = ^2  
and (BelongsToLatestApprovedRevis = 1  
or not exists (select 1  
from JobBudgetRevision  
where JobNumber = JobBudgetLine.JobNumber and BudgetType = JobBudgetLine.BudgetType)
```



```

and BelongsToLatestRevision = 1)
and TaskName = TaskListLine.TaskName), 0) as NumberOf,
nvl((select sum(nvl(NumberRegistered, 0))
from JobEntry
where JobNumber = ^1
and TaskName = TaskListLine.TaskName), 0) as NumberRegistered
from
TaskListLine, Activity, JobHeader
where
TaskListLine.TaskList = JobHeader.TaskList
and TaskListLine.ActivityNumber = Activity.ActivityNumber
and JobHeader.JobNumber = ^1
and Activity.ActivityType = 0
and (exists (select 1 from JobBudgetLine where TaskName = TaskListLine.TaskName and
JobNumber = ^1)
or exists (select 1 from TimeSheetLine where TaskName = TaskListLine.TaskName and
JobNumber = ^1))
order by
LineNumber
SQL

```

## 2.4.9 Upgrading from MAS or MCS

Starting with Maconomy 2.2, Delttek no longer releases the legacy solutions MAS and MCS. We recommend that if you are using one of these solutions and upgrading to version 2.2.x, that you implement one of the current (PSO or CPA) solutions. However, if you want to retain functionality from your current system, we have produced a package of files that can be used as a basis for customizing a standard system.

To upgrade from a legacy system and customize a standard system, follow these steps:

1. Upgrade the system using the standard procedure. However, do not install the SPU.
2. Create an Extender project for the upgraded system.
3. Extract the files from the package and copy them to the Extender project.
4. Deploy the Extender project.
5. Run the batch file  
w\_19\_0\ExtenderDeployFolder\ImportData\CopyToCustomInstallation.bat
6. In MConfig, for the field Load Shortname Data, select "Custom Data (Import only).
7. Click Install.
8. This installs system with a portal, prints, and screen layouts in the custom layer matching the former solution.



### Further Customizations

If needed, you can add further customizations to the Extender project after step 3.

#### Removing Layouts

A standalone program RemoveSolutionWindowLayouts can be used to remove the layouts from solutions left in the database.

To run the program, use a command similar to this:

```
MaconomyServer_19_0 -Smysshort -xRemoveSolutionWindowLayouts
```

The program should be run after attaching the database being upgraded to the new application, but before installing any new layouts, for instance, just after `-xRemove_Old_Layouts` command in the current chapter 10.3 Reinstalling Standard Layouts in the *DeltekMaconomy221Upgrade Guide*.

## 2.4.10 Approval of Invoice Drafts MCSInvoiceApproval

From 2.1.x, the workflow for approval of invoice drafts MCSInvoiceApproval is no longer supported as standard functionality in Maconomy.

When upgrading customers that use this workflow, separate upgrade steps are needed to facilitate the workflow. Upgrade the workflow in one of the following ways:

- **Workspace Client Users:** Re-implement the approval of invoice drafts by using approval hierarchy functionality.
- **Portal Users:** Upgrade the existing MWL workflow. This requires a manual update of the workflow. Contact development for instructions.

## 2.4.11 Removal of BPM Database Objects

When upgrading to Maconomy 2.3 it is mandatory to remove all BPM related database objects from the Maconomy database, such as materialized views, materialized view logs, database views and triggers.

Everything except materialized views can be removed by using MConfig and deselecting all checkboxes in the BPM section of shortname window. Materialized views and logs should be removed directly from the database.

When upgrading to Maconomy 2.3 or newer, any BPM-related objects must be removed from the database. To do so:

1. Deselect the all checkboxes in the BPM section found in MConfig shortname window and apply pending changes.
2. Check in the database that there are no Materialized views or Materialized view logs remaining. If Materialized views or Materialized view logs are present, delete them.

### 3 Upgrade Overview

The following table summarizes most of the steps that you must complete to perform an upgrade.

Phase	Step
Prepare	Run Upgrade Preconditions Script
Database UTF conversion	Install Oracle DMU Tool (If on Oracle DB)
	Install Maconomy dcu tools
	Install Maconomy smu tools
	Character convert the database
	Shorten offending strings
Installation	Install New Tools
	Install New 2.3 GA APU
Deleting	Run Upgrade Preconditions Script
	Delete Old Constraints
	Dump Changes to Standard Indexes
	Delete Indexes
	Dump Materialized View definitions
	Delete Materialized Views and View Logs
	Delete Views
	Delete Portal Standard Components
Converting Data	Expand and Convert Database Schema
	Recreate DHViews
	Upgrade to inicode
	Attach Shortname to 2.3 GA Application
	Recreate Views
	Update Fields and Relations
	Export Data for Conversion

Phase	Step
	Convert Exported Data
	Import Converted Data
	Create New Indexes
	Reinstall Standard Layouts
	Update Data for New Version (ConvertData)
Creating/Validating	Validate Layouts
	Recreate Constraints
Portal Installation	
Source File Conversion	
Source File Installation	

## 4 Check Upgrade Preconditions

Before starting the upgrade, you must check preconditions for the upgrade. Some versions require that specific conditions in the application be fulfilled to perform an upgrade. To check whether all of the requirements have been met, use the UpgradePreconditions script. Additionally, you must remove old BPM database objects, and change the default linesize value for Oracle.



It is a good idea to run this script well in advance of the update (for example, a week before) to let the customer know what they need to do, in addition to running it as part of the update.

The process of checking precondition differs according to the database type.

### 4.1 Check Preconditions for Upgrade

To check preconditions for the upgrade on Oracle, type the following commands:

1. `cd /data/maconomy/w_19_0/MaconomyDir/Database`
2. `StartOracleSQL.w_19_0.cmd myshort myshort UpgrPrecond.w12sv3-w190.sql > Logs/UpgradePreconditions.myshort.log`

If the upgrade precondition script returns a message like “Please create a support case,” this is due to duplicate keys in the DimensionPeriod table. You must resolve this issue before continuing with the upgrade.

The result should be 0 rows selected for all. If this is not your result, you cannot continue the upgrade until you have fixed the results. Check the Product Information for each version to see which requirements exist.



There is one upgrade precondition script for each old version.

You must select the appropriate script, for example:

- **Oracle:** `UpgrPrecond.w12sv3-w190.sql`

The old schema version must also be taken into consideration when you upgrade the application from W 12.0. You can find the schema version in:

```
<path to old application>/MaconomyDir/Installation/Version.info
```

For example, when upgrading from W 12.0 schema version 3 on Oracle, you must use the `UpgrPrecond.w120sv3-w190.sql` script.

### 4.2 Remove Old BPM Database Objects

When upgrading to Maconomy 2.3 or newer, any BPM-related objects must be removed from the database. To do so:

1. Clear all the check boxes in the BPM section found in MConfig shortname window and apply pending changes.
2. Check in the database that there are no Materialized views or Materialized view logs remaining. If Materialized views or Materialized view logs are present, delete them.

### 4.3 Change the Default Linesize Value for Oracle

To avoid errors in several of the upgrade scripts, change the default linesize to 3000. We recommend you do so by changing the glogin.sql file.

To change the glogin.sql file:

1. Use the Site Profile script \$ORACLE\_HOME/sqlplus/admin/glogin.sql in UNIX, and ORACLE\_HOME\sqlplus\admin\glogin.sql in Windows.
2. Add the following line to the glogin.sql
3. `LINESIZE 3000`
4. Check that there are no other changes to the linesize value in the glogin.sql, then save and close the file.

### 4.4 Integration with Talent Management

If you are upgrading from Maconomy 2.2.2 to 2.3 GA or later, and you integrate with Talent Management, you must delete old monitors and create new background tasks.

To delete standard monitors and create background tasks, follow these steps:

1. Use the SQL query on the Maconomy database to search for standard monitors:  

```
select * from Monitor where MonitorNumber = 'SynchronizeHRSmartUsers' or
MonitorNumber = 'ImportDTMUsers';
```
2. In the Monitors Single Dialog Workspace, locate and delete these two standard monitors: ImportDTMUsers and SynchronizeHRSmartUsers
3. Refer to the *Delttek Maconomy Integration With Talent Management Guide* for steps to create background tasks.

## 5 Create the Unicode Database



Oracle version 11.2.0.2 or higher is required.

You are required to create a database using the UTF8 character set. You do not need to create the Unicode database at this stage in the upgrade process, but it makes the Unicode conversion more structured if you do. If there is a lack of disk space on the server or you are just investigating the complexity of the Unicode conversion, you can wait to create the database until you get to the stage of the upgrade process that is described in *Oracle* on page 21.

Use MConfig version 8.3 or later to create the database. In this example the old database SID is maconomy (NLS\_LANGUAGE=WE8ISO8859\_P1), and the new SID is Unicode. This name may be used by the old database if both databases are on the same hardware.

SID information	SID	unicode
	Character set	AL32UTF8
	Language	american

The rest of the database creation process has not changed. Be aware that the Unicode database may grow up to 20%.

## 6 Installation

This section describes how to install the new TPU and the 2.3 GA application to which you are upgrading Maconomy.

### 6.1 Install the New Tools (TPU)

You must install a new TPU when upgrading—the most recent one that belongs to the W 19.0 application.

Copy MConfig, TPU (Tools version 19), and the 2.3 GA APU to the PU (PackingUnit) directory on the server. You use MConfig to specify the location of that directory.

For older versions of Maconomy there is only one TPU for all systems. With MConfig, different applications can use different TPUs. You must use the new TPU and the new application; installing the new TPU as global tools is required.

To install the TPU, use MConfig, the installation and configuration tool for Maconomy. Always use the most recent MConfig version unless you have been explicitly instructed otherwise. Consult the MConfig documentation if necessary.

### 6.2 Install the New Application (APU)

Use Mconfig to install the 2.3 GA application. When upgrading a solution, you must install the new application with the same enterprise language. You select the solution as one of the final steps in the upgrade process—not when installing the APU and TPU.



Be aware that enterprise language has been changed from w\_16\_0 SP2. The previous languages, such as US, W\_MCS, and so on, are no longer valid. Check the release notes [here](#). Access to [home.deltek.com](http://home.deltek.com) is required.

To check the enterprise language on an existing database, type the following SQL command:

```
Select ENTERPRISETEXT from ENTERPRISETEXT where  
ENTERPRISETEXTNAME= 'CountryCodeForEnterpriseLanguage' ;
```

Depending on how the shortname is transferred to the new application, you probably do not need to create a shortname when installing the new application.



Deltek does not recommend attaching the existing shortname to the new application until step 10.1.1. or 7.1.4 for Oracle Unicode upgrades (You must create dependencies, reading information from the database, which may fail.) The shortname might already be attached to the application, if the new application was installed before the database dump was imported into the database. In that case, you do not need to detach the shortname from the new application; just leave the shortname attached as it is.

This concludes the installation of the new application.



## 7 Unicode Conversion

Deltek Maconomy version 2.3 GA requires that all databases are Unicode (UTF8). The conversion takes place before the Maconomy upgrade process, but be aware that no Maconomy clients before Main version 2.1 can run on a Unicode database.



Upgrade preconditions must be met before you start the procedures in this section.



We recommend that you delete unused columns before you perform the Unicode conversion. See *Delete Unused Columns* for more information.

In all examples used in this section, the former database is called **Maconomy**, and the new database is called **Unicode**. The shortname is **myshort** and the password is **myshort** in both databases.

### 7.1 Oracle Conversion

Migrating an Oracle database involves a few utilities:

- DMU — Oracle Database Migration assistant for Unicode, a tool developed by Oracle.
- DCU — Data Conversion Utility, a proprietary tool developed by Deltek. Used to enable Maconomy database migration to Unicode.
- SMU — String Manipulation Utility, a proprietary tool developed by Deltek. Used for reducing the length of offending strings (strings that exceed 255 bytes when converted into UTF8).



Deltek recommends that you install the DMU tools well in advance of the Unicode conversion to get an overview of the amount of work needed to convert the database.

#### 7.1.1 Export Metadata from the ISO Database (the Former Database)

To export metadata from the former database, complete the following steps:

1. Connect to the non-Unicode database and create a dump directory to be used by the expdp utility program from Oracle.

The expdp utility is faster than the regular export program (exp).

2. Log in using sqlplus and type the following command:

```
create directory dump_dir as 'D:\maconomy\Backup';
```

If this command fails, the dump\_dir object already exists; this is not of concern. The metadata dump file will be placed in that directory.



Check the NLS\_LANG environment parameter. It must match the setting in the MaconomyServer...I file.

- Export the metadata for the old database (the one that uses the ISO character set) by typing the following commands:

```
set NLS_LANG=AMERICAN.WE8ISO8859P1
set ORACLE_SID=Maconomy
expdp myshort/myshort dumpfile=expdp.myshort.dmp logfile=
expdp.myshort.log directory=dump_dir schemas= myshort
exclude=index,index_statistics,table_statistics,trigger,comment,post_tabl
e_action,constraint,view,materialized_view,materialized_view_log
content=metadata_only
```



The schema created in the Unicode database is without indexes, views, constraints, and materialized views. All of these are recreated during the upgrade process.

The output should look like the following:

```
Processing object type SCHEMA_EXPORT/USER
Processing object type SCHEMA_EXPORT/SYSTEM_GRANT
Processing object type SCHEMA_EXPORT/ROLE_GRANT
Processing object type SCHEMA_EXPORT/DEFAULT_ROLE
```

### 7.1.2 Import Metadata into the UTF Database

Remember to set the NLS\_LANG and the ORACLE\_SID. In the remainder of this section the new database is called unicode and the TNS name is called unicode as well. You should use a more distinctive name for customer databases.

**To import metadata into the UTF database, complete the following steps:**

- Enter the following to set the value of NLS\_LANG and the ORACLE\_SID:

```
set NLS_LANG=AMERICAN.AL32UTF8
set ORACLE_SID=unicode
```

- Create the shortname and create the dump directory:

```
Grant dba to myshort identified by mypassd;
Alter user myshort default tablespace Maconomy temporary tablespace
mactmp;
create or replace directory dump_dir as 'D:\maconomy\Backup';
```

The output should look like the following:

```
Processing object type SCHEMA_EXPORT/USER
ORA-31684: Object type USER:"MYSHORT" already exists
Processing object type SCHEMA_EXPORT/SYSTEM_GRANT
Processing object type SCHEMA_EXPORT/ROLE_GRANT
```

The following error is not of concern; it is displayed when the MYSHORT user has been created for an earlier test upgrade:

```
USER:"MYSHORT" already exists
```

- Log in to the Unicode database and create a database link to the old database:

```
create database link maconomy connect to myshort identified by myshort
using 'maconomy';
```

### 7.1.3 Install Oracle DMU

The Oracle DMU is written in Java and is supported on selected Windows- and Unix-based platforms for which the Java SE Development Kit 6 is available. The database must meet certain requirements to support the DMU as follows:

- Oracle version 11.2.0.1 patch 13 is required for the DMU.
  - You must create an Oracle password file:
    - For Windows:
 

```
orapwd file=%ORACLE_HOME%\database\PWD%ORACLE_SID%.ora
PASSWORD=<mypasswd> ENTRIES=30
```
    - For Unix:
 

```
orapwd file=$ORACLE_HOME/dbs/orapw$ORACLE_SID PASSWORD=<mypasswd>
ENTRIES=30
```
  - You must have a user with the sysdba privilege:
 

```
grant sysdba to system;
```

or

```
grant sysdba to myshort;
```
  - You must have the SYS.DBMS\_DUMA\_INTERNAL package:
 

```
cd $ORACLE_HOME
sqlplus / as sysdba
@?/rdbms/admin/prvtdumi.plb
```
  - The Oracle XDB component must be installed in the database.
- You can do this with or without Database Configuration Assistant (DBCA). However, using DBCA is the easiest way to install the XDB.
- Start dbca in a command line:
 

Click <Next> <Configuration Database Options> <Next> "Select database" <Next> <Standard Database Components> "Select Oracle XML DB" <OK> <Next> "Select Dedicated Server Mode" <Finish>

You can get the DMU installation unit at the following location:

<http://www.oracle.com/technetwork/products/globalization/dmu/downloads/index.html>

You can get the DMU installation instructions at this location:

<http://www.oracle.com/technetwork/database/globalization/dmu/learnmore/start-334681.html>

Installing the DMU involves unzipping the DMU installation unit. If all of the requirements are fulfilled, you can start the DMU tool by running dmuW64.exe. This program is located in the dmu directory structure. If the DMU is unable to start, you need a patch for Oracle.

When the DMU is started for the first time it asks for the path to the java executable.



Only JDK version 1.6 is supported (jdk1.6.0\_45 is tested and works).



DMU stores the java executable path in ~/.dmu\_jdk for Unix-based platforms. If you remove this file, the tool asks for the full path again. If the java executable is found in PATH or JAVA\_HOME is defined, it does not prompt you. On Windows, you can edit the JDK location in the file dmu\dmu\bin\dmu32.conf (DMU with 32-bit JDK) or in the file dmu\dmu\bin\dmu64.conf (DMU with 64-bit JDK) under the DMU installation folder. Look for the keyword SetJavaHome.

### 7.1.4 Clean Up the Database

Oracle version 10g and higher introduced a recycle functionality. Deleted records show up as tables with names like BIN\$.... (for example, BIN\$3yZFIL0MQaWu+5wW9eh6aA==\$0). Do not convert these tables; it is recommended that you empty the recycle bin before you generate the DMU report.



You cannot roll a PURGE statement back.

To list the contents of the recycle bin, enter the following:

```
select * from recyclebin;
```

or

```
select * from user_recyclebin;
```

To remove the entire contents of the recycle bin, enter the following:

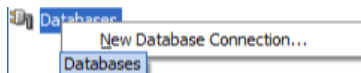
```
Purge recyclebin;
```

### 7.1.5 Generate the DMU Report

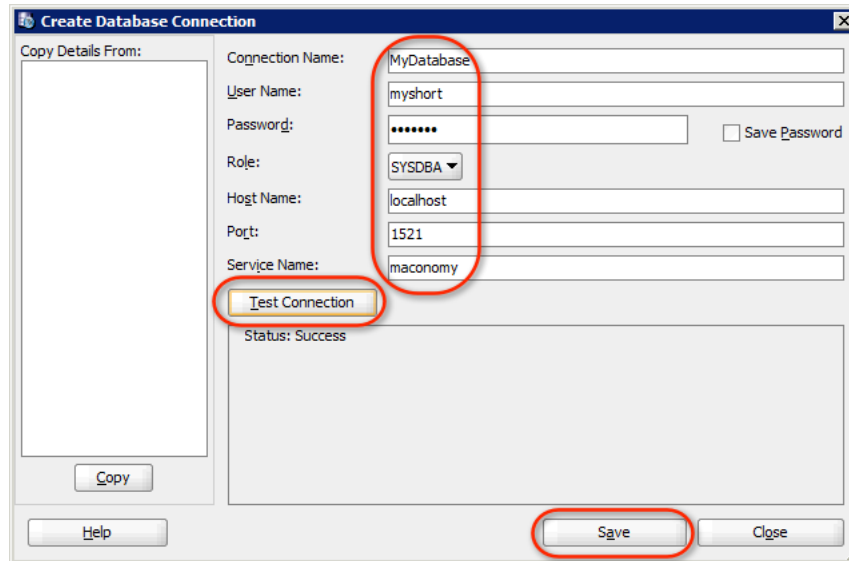
A database connection to the old database is required when you start the DMU.

**To generate the DMU report, complete the following steps:**

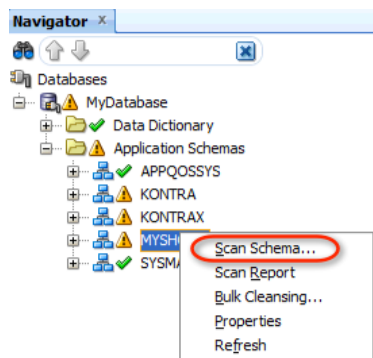
1. Right-click on **Databases** or choose **File » Create New Connection**.



2. Enter the information for the connection.

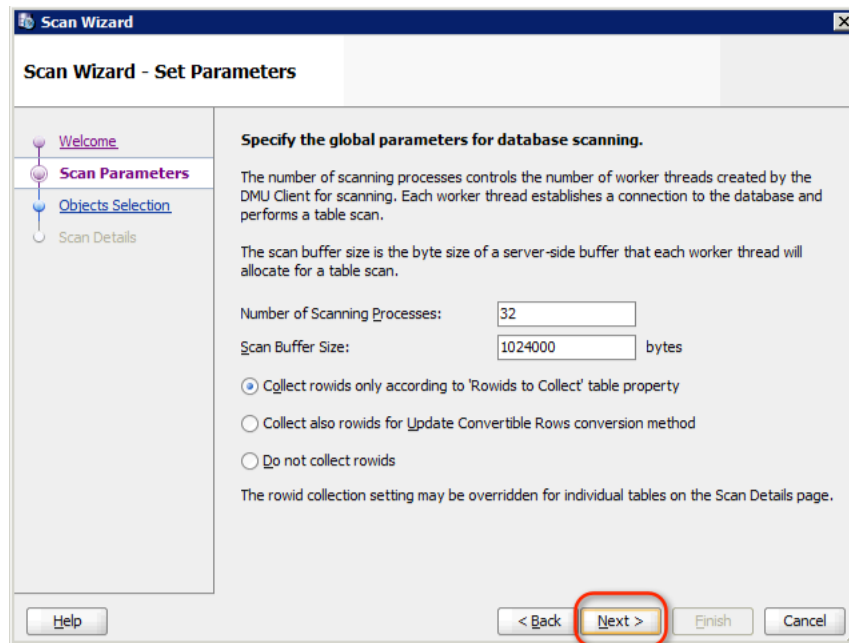


3. Click **Test Connection** to test the connection.
4. Click **Save** to save the connection.
5. Right-click on the name of the connection.
6. Right-click on the application schema name, then choose **Scan Schema**.

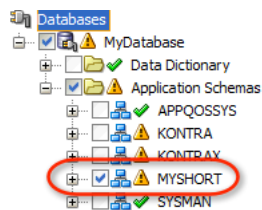


The Scan Wizard is displayed.

7. Keep the default values in the Scan Wizard and click **Next**.



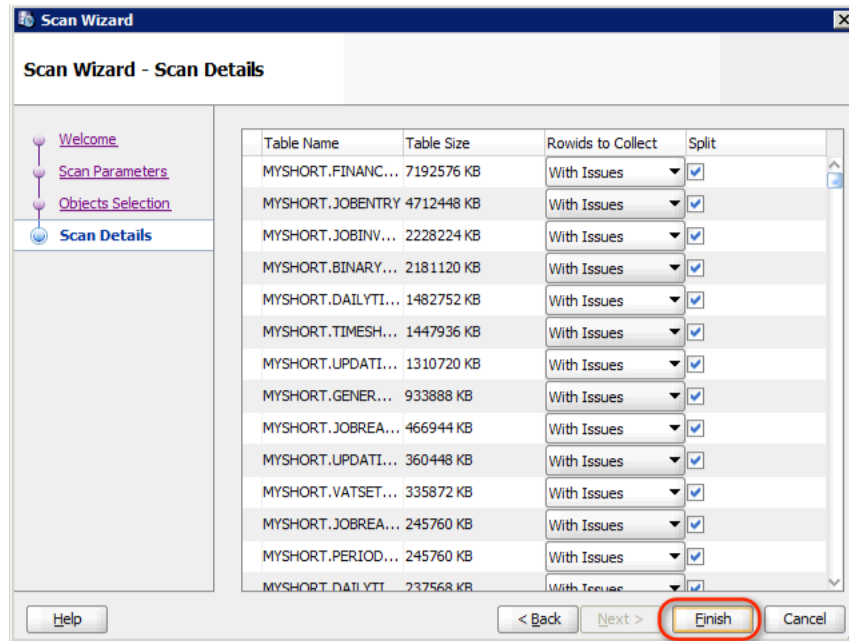
8. Expand the Application Schemas folder and select the schema that you want to analyze.



9. Click **Next**.

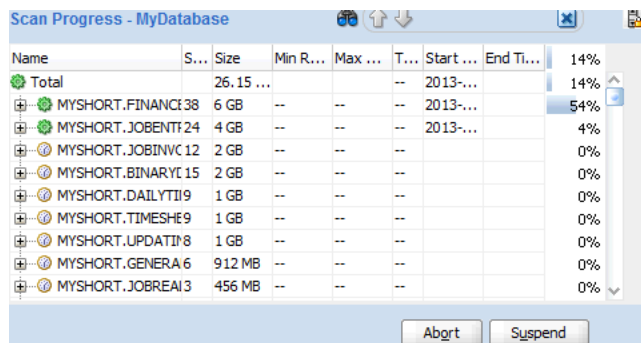
The Scan Wizard – Scan Details window is displayed.

10. Validate the details to ensure that only rows that have issues are collected.

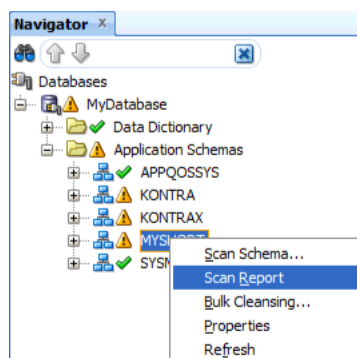


- Click **Finish** to start the database scan.

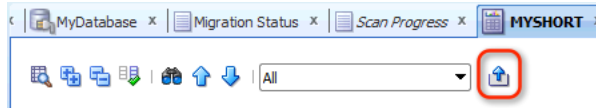
Scanning should not take more than a few minutes if the database size is less than 100 GB.



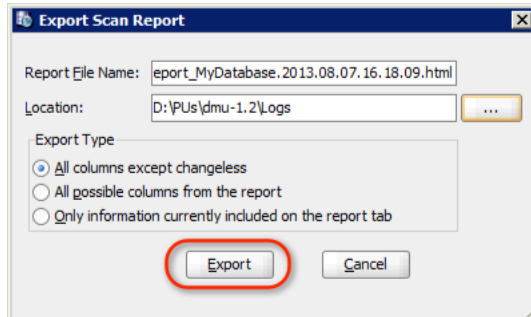
- Right-click on the schema name when the scan is finished, then choose **Scan Report**.



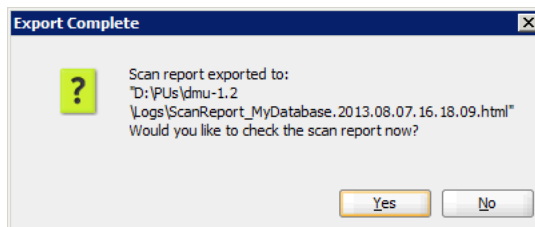
- Export the result as an HTML document by clicking the export icon.



14. Choose a file name and location for the report and click **Export**.



When the report is generated, the Export Complete dialog box enables you to view the report in a browser.



15. Click **Yes** to view the report.

The HTML report file is saved to your hard drive (such as D: in the following example) with the following path and name:

D:\PUs\dmu-1.2\Logs\ScanReport\_MyDatabase.2013.08.07.16.18.09.html

## 7.1.6 Interpreting the DMU Report

Customers' DMU reports are very different due to the way employees have entered data and which Maconomy CC\_Tables they have used. Most customers are not aware of this because it has always been a Maconomy TIA task to set up these CC\_Tables.

Most Western databases use Oracle's WE8ISO8859P1 Character set. In this character set all ASCII values between 128 and 160 (decimal) are undefined. In the past, Maconomy used these values to store some special characters. Microsoft products also use this space to store their special characters, such as a soft hyphen, and so on. When the DMU analyzes the database, all characters that are stored in these positions are shown as invalid characters. When converting these characters using Oracle's tools, they are stored in position 191, shown as "¿" in our clients.

Open the DMU report in a browser and look for this symbol: ⚠. This symbol can appear next to the table and again next to the column.

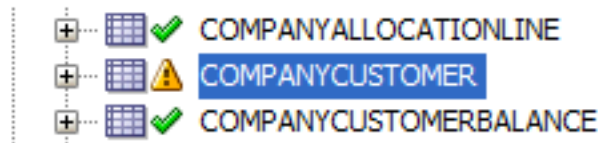


Name	Need Conversion (Scheduled)	Invalid Representation (Scheduled)	Over Column Limit (Scheduled)
COMPANYCUSTOMER	2967	15	0
ATTENTION	431	1	0
CONTACTPERSON	17	0	0
CREATEDBY	3	0	0
ELECTRONICMAILADDRESS	3	0	0
INTERCOMPANYJOBPRICELIST	0	1	0
JOBPRICELIST	0	1	0
NAME1	613	1	0
NAME2	882	4	0
NAME3	217	4	0
NAME4	63	0	0
NAME5	51	0	0
POSTALDISTRICT	613	0	0
TELEFAX	2	0	0
TELEPHONE	1	2	0
TELEX	31	0	0
VATNUMBER	1	0	0
ZIPCODE	39	1	0

In the preceding figure the table name is CompanyCustomer, and 15 rows out of 2967 need attention. The 15 rows are spread over several columns, all flagged by the exclamation point.

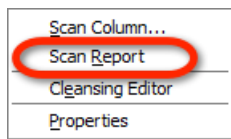
**To examine the report, complete the following steps:**

1. Open the DMU and expand the tables.
2. Look for items flagged by the exclamation point.



In this example, COMPANYCUSTOMER needs attention.



3. Expand the COMPANYCUSTOMER table, right-click on the column, and select **Scan Report**.



4. Look to see how many records need to be converted and how many invalid characters the DMU tool found.

Need No Change (Sc...	Need Conversion (Sc...	Invalid Representation...	Over Column Limit (Sc...
107395	2	3	0

Continuing the example, there are 3 invalid characters (highlighted by the red arrow).

5. Click on the number to see the records.  
Invalid characters are displayed as .
6. Double-click on a record to see the char value of the invalid character.  
In this case it is a chr(96<sub>hex</sub>) or chr(150<sub>decimal</sub>), which is the  defined in Word.



All character values are displayed in hexadecimal representation.

The char value is displayed in the following fashion.

<input type="checkbox"/> Value is NULL <input checked="" type="checkbox"/> Wrap Lines	Offset	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e
070 643 02 15		30	37	30	20	96	20	36	34	33	20	30	32	20	31	35

One way to resolve this invalid character is to substitute another one for it, like ' or chr(2).

- Click on the value **96**, change it to **27**, and click **OK**.

Offset	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e
00000000	30	37	30	20	96	20	36	34	33	20	30	32	20	31	35

- Continue with the remaining records.



You can update each record directly in the DMU tool, but this is not recommended. That process is totally manual, requires a lot of time, and cannot be reused.

The only scenario in which changing characters manually is advisable is when there are just a few records to be changed, in very large tables such as FinanceEntry and JobEntry.



Make a note about the required character change, and then make the actual change later, using the DCU configuration file. You make the change(s) in the OverwriteChars section.

### 7.1.7 DCU Utility Program

The DCU utility program generates SQL statements, PL/SQL stored procedures, and export parameter files, which are used for character conversion.

**To use the DCU utility program for character conversion, complete the following steps:**

- Create SQL scripts that change all string definitions to 765 bytes.
- Create a PL/SQL procedure to convert all tables that need special character mappings.
- Run the PL/SQL procedure for those tables.
- Create an export parameter file for the tables that do not need special mappings.



If there are errors or special needs in regard to the DCU utility program, see the *Unicode Conversion Guide*, which is included in the upgrade kit.



The DIMENSIONPERIOD table is converted with the DCU procedures regardless of what is specified in the spec file. This is done to ensure the correct conversion of the DIMENSIONKEY column. Any character replacement/conversion rules still take effect on the DIMENSIONPERIOD table if specified in the spec file, but the DIMENSIONKEY column is regenerated.

**This step fails if duplicate keys are present.** Make sure that the upgrade precondition script executed with no errors or warnings before continuing with running the DCU-generated procedures.

## 7.1.8 Install the DCU

Unpack the zip file that contains the DCU. Be sure to use the latest version of the DCU, which can be found in the latest version of the Upgrade Kit.

## 7.1.9 Configure the DCU

Before you use the dcu.exe file for the first time, edit the config.cfg file to match the current database configuration. The following is an example of a config.cfg file:

```
[General]
#threadcount indicates how many threads are to be executed in parallel minimum
4 recommended 12
#not more than 1 thread per cpu core
threadcount=32
#dblink is the name of database link which has to be created in the database
dblink=Maconomy

[DestinationDB]
#destination database logon information
ip=172.16.185.226
sid=Unicode
port=1521
username=myshort
password=myshort

[Files]
#Location of the specification file to be used with DCU
specfile=D:\PUs\dcu\spec.myshort.cfg
```



The thread count should match the total number of cores in the server.

The MakeScannerSpec utility helps you to specify the spec.cfg file. The program scans the DMU report and generates a first version of the DCU spec file.

```
Cd D:\PUs\dcu
MakeScannerSpec.exe D:\PUs\dmu-
1.2\Logs\ScanReport_MyDatabase.2013.08.07.16.18.09.html spec.myshort.cfg
```



There is a Kona space for character conversion tests where you can get the latest character mapping tables.

Be aware that there are at least two different CC\_tables:

- Version 2, where some special characters are remapped.
- Version 3, which does not remap any characters.

There are also some custom CC\_Tables, but they are believed to be rare. All data entered using the Java client does not use the CC\_Tables.

Modify the spec file that you created to match the current character mapping. The following example is a mapping for a win-1252 character set mapped in an iso-8859-p1 database. You can also use these replacements in a standard ISO database to convert some of the special

characters used in Word. By replacing these characters at this stage, strings that contain these characters may not violate the string length of 256 characters later.

```
replace=145:39;
replace=146:39;
replace=130:39;
replace=147:34;
replace=148:34;
replace=150:45;
replace=151:45;
replace=152:126;
replace=159:163;
replace=128:14844588;
replace=131:14844588;
replace 159: 49827
replace 161: 49829
```

The UTF character 14844588 is a € sign; 49827 is a £ sign; and 49829 is a ¥ sign. The rest of the signs that are remapped are special characters used in Word. It is common for customers to copy/paste text from Word and other applications directly into Maconomy dialogs.



All character values must be entered in decimal numbers, not hexadecimal.

### 7.1.10 Alter the Length of VARCHAR

You must change all strings from VARCHAR(255) to VARCHAR(765) in the unicode database. This reserves space for multi-byte characters.

**To change strings from VARCHAR(255) to VARCHAR(765), enter the following commands:**

1. `cd D:\PUs\dcu`
2. `mkdir SQLs`
3. `dcu --genstrlen SQLs\expandStrings.myshort.sql`
4. `sqlplus myshort/myshort@unicode < SQLs\expandStrings.myshort.sql > logs\expandStrings.myshort.log`



Check the log files logs\dcu.main.log and logs\expandStrings.myshort.log. You cannot have any errors or warnings.

### 7.1.11 Generate a PL/SQL Procedure for Conversion

**To generate the source code for the PL/SQL procedure in the Unicode database, enter the following commands:**

1. `dcu.exe --genproc SQLs\pl-sql.myshort.sql`
2. `sqlplus myshort/myshort@unicode < SQLs\pl-sql.myshort.sql > logs\pl-sql.myshort.log`



Check the log files logs\dcu.main.log and logs\pl-sql.myshort.log. You cannot have any errors.

### 7.1.12 Convert Tables Defined in DCU spec.cfg File

Generate the source code for the PL/SQL procedure:

```
dcu.exe -runproc
```



Check the log file logs\dcu.main.log. You cannot have any errors.

If there are errors consult the advanced documentation in the *Unicode Conversion Guide*.

**DO NOT continue if this part has errors.**

If errors occur in this stage you can run the conversion again for one of the tables that failed. An error message could look like the following:

```
[2013-08-09 13:03:24.152000] Conversion failed for table: NOTELINE
```

**To investigate errors, run the following commands in the Unicode database:**

1. `sqlplus myshort/myshort@unicode`
2. `truncate table NOTELINE;`
3. `set serveroutput on;`
4. `var ret1 number;`
5. `exec NOTELINE_CONV(:ret1);`

Fix the error and rerun `dcu.exe -runproc`. This process converts only those tables that failed. All tables that are converted are listed in the file logs\proclist.log. If a table is already converted, it is possible to remove the table in this file.

### 7.1.13 Convert the Remaining Tables

To convert the remaining tables in the old database, export all tables that were converted in the previous section and then import them into the Unicode database. The **gendatapumpparams** parameter generates an export parameter file that Oracle's data pump program can use.

```
dcu.exe --gendatapumpparams
```

The following parameter files are generated:

- `expparams.txt` — All tables listed in the file logs\proclist.log are excluded from the dump file.
- `impparams.txt` — Parameters for importing all tables in the dump file.

On the iso database, export all unconverted tables:

```
SET NLS_LANG=AMERICAN.WE8ISO8859P1
SET ORACLE_SID=Maconomy
expdp myshort/myshort PARFILE= expparams.txt
```

Switch to the Unicode database and import the file that was generated:

```
SET NLS_LANG=american.AL32UTF8
SET ORACLE_SID=Unicode
impdp myshort/myshort PARFILE= impparams.txt
```

### 7.1.14 Attach the Shortname

For the Unicode upgrade, you must attach the shortname to the application that was installed in the Installation section. The shortname does not need to be detached from the original application as it is on a different SID and will not affect the new application. This only applies to Oracle upgrades, as in SQL Server the shortname is upgraded in place.

## 7.2 Convert Text Strings

Check whether the database contains any offending strings by running the SMU with the “- - extract” parameter. (An offending string is any string that is longer than 255 characters after the Unicode conversion.) You must fix any offending strings that are found before proceeding. The DCU (or DCUSQL) configuration file can be used with the SMU as well.

Enter the following command to find offending strings:

```
smu --extract
```

Check the offendingstrings.txt file located in the same directory as the smu.exe file.

### 7.2.1 Shorten Offending Strings

If the number of offending strings is small (fewer than 50) it can be faster just to change them manually. Alternatively, you can change the offendingstrings.txt file automatically by replacing/shortening strings listed in the file.



All strings that are listed in the offendingstrings.txt must be fewer than 255 characters long before the conversion can continue.

Based on the contents of the offending strings you must edit a file called replacements.txt. The following is an example of that file:

```
/\"/:/"/
/ \" /:/"/
/"/:/"/
/ \" /:/"/
/' /:/'/
/ ' /:/'/
/\' /:\'/
/ \ /:\'/
/- /:-/
/ - /:-/
/- /:-/
/ - /:-/
/ + /:/+/
/ = /:/=/
/million:/M/
```

Keep a record of the replacement files used to optimize this process.

To see the effect of the replacements, enter the following command:

```
smu --translate
```

Review the offending string file and try to find common patterns—which would make the strings shorter and still retain their original meanings—until all string lengths are fewer than 255 characters.

When all strings are fixed, enter the following command:

```
smu --import
```

To check that all strings are valid, enter the following command:

```
smu --extract
```

and check that the offendingstrings.txt file is empty.

## 7.3 Restore String Length

Restore database string field lengths to their original values of 255 bytes. You do this in the Unicode database.

**To restore database string lengths to their original values for Oracle only, enter the following commands in the Unicode database:**

1. `dcu.exe -- genrestoresql SQLs\RestoreStrings.myshort.sql`
2. `sqlplus myshort/myshort < SQLs\RestoreStrings.myshort.sql > Logs\RestoreStrings.myshort.log`

## 8 Delete Old Components

### 8.1 Delete Old Constraints



For large databases see *Drop Constraints* on page 58.

First, drop instance key constraints temporarily.

Enter the following command:

```
MaconomyServer.w_12_0 -UD -Smyshort
```

This part of the step is required only if the database system is Oracle. You must use the DropConstraints script that belongs to the W 9.0, W 10.0, W 11.0, or W 12.0 version—note the change of directory to that of the old application. Enter the following commands:

```
cd /data/maconomy/w_12_0/MaconomyDir/Database
StartOracleSQL.w_12_0.cmd myshort myshort DropConstraints.sql >
/data/app/maconomy/w_19_0/MaconomyDir/Database/Logs/DropConstraints.mysort.log
```



#### Other Constraints

On Oracle systems, constraints may have been created to track down errors. To check whether any constraints remain, enter the following command from an sqlplus session or create an SQL script and execute it.

Do not delete any constraints that start with “SYS” or “REPCAT.”

```
select table_name, constraint_name from user_constraints where
constraint_type = 'U';
```

Drop each constraint “C” found in a table “T” by typing the following command:

```
alter table T drop constraint C;
```

### 8.2 Dump Changes to the Standard Index Set



For large databases see *Dump Changes to Standard Indexes* on page 58.

An integral part of a Maconomy application is its indexes. Indexes are created to improve performance. For all Maconomy application versions there is a standard set of indexes. All standard indexes are created with a unique name (unique within 18 characters) that identifies the index. For example, one index might be named “Account04” and contain an index on the relation “Account” and the field “AccountNumber.”

For an installed Maconomy system, indexes may have been changed for performance purposes. Compared with the standard set of indexes, new indexes may have been added, or some standard indexes may have been deleted. The changes may be different for different short names.

When upgrading the application version, you typically want to preserve these changes. Use MConfig to dump a list of the changes. This function is implemented for MConfig version 4.1 or



later. However, you should always use the newest MConfig version, unless you have been explicitly instructed otherwise. Consult the MConfig documentation if necessary.

**To use MConfig to get a list of changes to preserve, complete the following steps:**

1. In the Main window, select the old application and click **Edit**.
  2. In the Application window, for each shortname complete the following steps:
    - Select the shortname in the **Shortnames** list and click **Edit**.
    - In the Shortname window, click **Index management**.
    - In the **Compare with application** list, select the old application again and then click **Choose**. One of the following messages is displayed:
      - **No index differences found compared with application.**
- or
- **Comparing indexes with application ... NN indexes added ... NN indexes dropped ... Index compare file 'IndexCompareWith<shortname>.<old application>.idx' generated.**
- In the latter case, "IndexCompareWith...idx" is a file that is located in the MaconomyDir/Database directory.
- Copy the file "IndexCompareWith<shortname>.<old application>.idx" to the new application's MaconomyDir/Database directory. The file then appears in the Index spec. files list in the Index management window for the new application. Later, when creating indexes, you can use this file to preserve the local changes.

## 8.3 Delete Old Indexes



For large databases see [Drop Indexes on page 58](#).

Each new version has new indexes, and you must delete all of the old ones before you can proceed with the upgrade.

Use MConfig for this purpose. Deleting indexes is supported for MConfig version 7.1 or later. However, you should always use the newest MConfig version unless you have been explicitly instructed otherwise. Consult the MConfig documentation if necessary.

**To use MConfig to delete the old indexes, complete the following steps:**

1. In the Main window, select the old application (the application to which the shortname is attached) and click **Edit**.
2. In the Application window, for each shortname complete the following steps:
  - Select the shortname in the **Shortnames** list and click **Edit**.
  - In the Shortname window, click **Index management**.
  - Select the **Remove indexes** check box. Then click **OK** and **OK** again to return to the Application window to continue with other shortnames.

## 8.4 Dump Materialized View Definitions

Large customers who are using Oracle as their database may have used materialized view functionality to speed up frequently used reports. You must identify, drop, and convert these views to the new database schema after the upgrade. A BPM consultant normally performs this task. As part of the upgrade it is important to retrieve the current view definitions.

A materialized view consists of a view log and the view itself.

**To retrieve the view log definitions, enter the following SQL command:**

```
1. select log_owner, master, rowids, include_new_values from dba_mview_logs;
```

Output from this statement should look like the following example.

LOG_OWNER	MASTER	ROW	INC
-----	-----	---	---
MYSHORT	JOENTRY	YES	YES
MYSHORT	JOBINVOICELINE	YES	YES

**To retrieve the materialized view definition, enter the following commands:**

```
1. set pagesize 10000;
2. set long 10000;
3. column refresh_mode format a12;
4. select owner, mview_name, refresh_mode, query from dba_mviews where
   owner='MYSHORT' ;
```

Output from this command may contain a lot of data. The following example is a subset of the real definition.

OWNER	MVIEW_NAME	REFRESH_MODE
-----	-----	-----
QUERY		
MYSHORT	JOENTRYJOBINVOICELINEMV	COMMIT
SELECT		
	JOENTRY.ROWID JOENTRYROWID	
	...	

## 8.5 Delete Views

Three different types of views are in use by Maconomy:

- Views used by the Maconomy Client for access control
- Views used by third-party software using ODBC access to ensure proper access control
- Views used by the Maconomy Analyzer

You must remove the first two types before you can perform an upgrade.

Use the following commands to remove the views:

```
DeleteDHViews <shortname> <password>
Delete3PStuff <shortname> <password>
```

When updating from one application to another, the scripts must be run to ensure proper functioning of the new application version. The programs are located in the Maconomy bin directory in the TPU that is installed for the application, for example, the

C:\maconomy\tpu.19.0.dir\bin folder (and not the C:\maconomy\bin folder, which is the global TPU folder).

The DeleteAnalyzerViews is an Mconfig scriptlet and requires MConfig to be run. MConfig 8.0 or later is required to run the scriptlet. MConfig must be started from the command line. Specify the path to MConfig, then -r (or MConfig will think it is an MConfig installation script), then the path to the script or just the name – MConfig searches for the script in the TPU that is installed for the application (that is, not the global TPU). These scripts are located in the MConfigScriptlets folder in the TPU folder, for example, C:\maconomy\tpu.12\_0.dir\MConfigScriptlets. Lastly, the arguments to the script must be given. These are the application name (for example, w\_12\_0.myapp) and the shortname and password.

Delete the views for the user with the following commands.

### Oracle for Windows, UNIX

```
DeleteDHViews myshort myshort
Delete3PStuff myshort myshort
MConfig-8.3 -r DeleteAnalyzerViews w_12_0.myapp myshort myshort
```

The output from the delete views scripts for all platforms is:

Done!

If the scripts did not delete any views, the following error message is displayed (on all platforms).

### Something went wrong (or nothing to delete)

It may be OK if the customer was not using the functionality. Check with the old access rights. Make sure that the ORACLE\_SID environment variable is set, or the scripts may fail with the preceding error message.

The DeleteAnalyzerViews script will report the number of views that it deletes. In case of errors, check the log file (the output of the script shows the path to the complete log file) to see the SQL that failed.

## 8.6 Delete Portal Standard Components

If you use the Portal, you must delete the standard components from the Portal, using the DeletePortalStandard.sql SQL script. This script deletes standard components and leaves only customized components in the database. This ensures that no standard components are left in the database when upgrading and converting the Portal (names may have changed).



This script is part of the Upgrade Kit, so you must change the directory to that folder.

---

**To delete standard components from the Portal if you use Oracle, enter the following commands:**

1. `cd /data/UpgradeKit/SQL`
2. `StartOracleSQL.w_12_0 myshort myshort DeletePortalStandard.sql > /data/maconomy/w_19_0/MaconomyDir/Database/Logs/DeletePortalStandard.mysh`  
or



Skip this step if you are upgrading from Maconomy W 10.0, W 11.0, or W 12.0. Only upgrades from Maconomy W 9.0 require terminology conversion.

---

You must delete standard print and window layouts from the database, using the DeleteStandardLayouts script. This ensures that there are no orphaned layouts left in the database that come from the W 9.0 application (names that have changed in later versions). The customized layouts are left in the database for conversion at a later step. To delete the standard layouts, you must run this SQL script.



This script is part of the Upgrade Kit, so you must change the directory to that folder.

**To delete standard print and window layouts from the database, enter the following commands:**

1. `cd /data/UpgradeKit/SQL`
2. `StartOracleSQL.w_12_0 myshort myshort DeleteStandardLayouts.sql > /data/maconomy/w_19_0/MaconomyDir/Database/Logs/DeleteStandardLayouts.myshort.log`

Examine the log file. Locate and correct errors.



There may be SQL errors stating that a relation (for instance PDMCOMPONENT) does not exist. You can ignore these errors. However, only errors about tables not existing can be ignored. All other errors must be corrected.

## 9 Update the Database Schema

### 9.1 Expand Relations



For large databases see *Expand Script* on page 59.

The purpose of an expand script is to update the fields and relations of the existing database to the new fields and relations.

This can include the following tasks:

- Create relations
- Add new fields to an existing relation
- Change the type of an existing field
- Move existing fields in existing relations
- Delete relations
- Delete fields from relations
- Extend field and relation names to 28 characters

An expand script is version-dependent, and you must never run “wrong” expand scripts on a database.

There are three important things to check regarding the version:

- The source version
- The source schema version
- The target version



Assume that the source application is W 12.0, and a service pack with schema changes has been applied; you should use the expand script “expand.w120svX-w160.sql”, where X is the current schema version number. The schema version is shown in the “<path to W 12.0 application>/MaconomyDir/Installation/Version.info file.

You will always get error messages when running expand scripts, but it is important to check that you only receive the expected messages. Thus, it is important to enable logging **before** you run the expand script. On large databases this script may run for some time.



You cannot restart the expand script.

**To run the expand script, enter the following commands:**

1. `cd /data/maconomy/w_19_0/MaconomyDir/Database`
2. `StartOracleSQL.cmd myshort myshort Expand.w120sv3-w190.sql > Logs/expand.myshort.log`

After completion, check the “expand.myshort.log” file using a text editor. You may find several error messages regarding attempts to drop non-existing tables. Ignore tables that end with ‘\_T’.. You should make sure that there are no other error messages.

After, check the error log.

Check the expand.myshort.log file using a text editor and search for the text “ORA-.” You should see several error messages about attempts to drop nonexistent tables. You can ignore those errors, but make sure that there are no other error messages.

Regardless of which database you are running you may see error messages in the expand script stating that the following tables do not exist:

```
DEV_CompanyCustomerCC
DEV_CompanySetupCC
DEV_EmailalertsDistribution
DEV_EmailalertsTemplates
DEV_EntrySelectionCC
DEV_ExpenseJustificationDes
```

This happens if standard extensions have not been installed in the system that is being upgraded; in that case you can ignore these messages because these tables will be created during the standard extension install (if standard extensions are needed).

Make sure that there are no other error messages.

You might also see error messages stating that the previously mentioned DEV\_ tables already exist. In that case verify that the DEV\_ table definitions in the expand script match the tables that are already in the schema. If the existing tables match the ones that are defined in the expand script, it is safe to continue.



Compare the DEV\_ tables to their definitions in Maconomy version 2.3 GA, because any newer version of Maconomy might contain changes to these tables.

If the existing tables do **not** match the tables from the expand script of Maconomy version 2.3 GA, it is a customization that you must deal with on a case-by-case basis, working with the Deltek Services team.

When upgrading to Maconomy version 2.3 or higher a postexpand script needs to be executed before moving forward

**To run the postexpand script, enter the following commands:**

1. `cd /data/maconomy/w_19_0/MaconomyDir/Database`
2. `StartOracleSQL.cmd myshort myshort PostExpand.sql > Logs/postexpand.myshort.log`

## 9.2 Delete Unused Columns

The Oracle expand script does not physically delete columns from the database. It makes the columns invisible by using the ALTER TABLE..SET UNUSED command.

To physically delete the columns you must run the DeleteUnusedColumns scriptlet. This is an MConfig scriptlet that is located in the TPU.

**To run the DeleteUnusedColumns scriptlet, complete the following steps:**

1. Run Mconfig from the command line with the following command:

```
MConfig-8.12 -r DeleteUnusedColumns w_nn_n myshort myshortpw
```

where:

- The first argument, w\_nn\_n, is the application name, such as w\_12\_0.
- The second argument, myshort, is the shortname.
- The third argument, myshortpw, is the password.

For small databases, you might want to run this scriptlet immediately after having attached the shortname to the new application. However, for large databases you might save time by postponing this task until later. It is recommended that you perform this task before you perform the Unicode conversion.

## 9.3 Convert Terminology of Setup and Production Data



Skip this step if you are upgrading from Maconomy W 10.0, W 11.0, W 12.0, and W 15.0. Only upgrades from Maconomy W 9.0 require terminology conversion.

This step performs the automatic terminology conversion of the contents of certain string fields, which should contain entries in kernel language. As a part of improving the English terminology used, you must run the PreTranslateKernelStrings and TranslateKernelStrings SQL-scripts. PreTranslateKernelStrings cleans up some known problems and thus prepares the system for further translation.

The execution of the TranslateKernelStrings.sql script fails when outdated or obsolete US kernel strings exist in the database. The SQL script fails in situations like these through simulation of division by zero. Before failing, the TranslateKernelStrings script outputs the list of entries that cannot be updated correctly. When the execution of the SQL script is stopped through a division by zero, the initial state of the database is reinstalled (rolled back). This enables the re-execution of TranslateKernelStrings.sql after you correct the problematic entries in the database. Contact the Maconomy Engineering team if the execution of the TranslateKernelStrings.sql script fails through a division by zero error.



This script is part of the Upgrade Kit, so you must change the directory to that folder.

**To convert the terminology of setup and production data, enter the following commands:**

1. `cd /data/UpgradeKit_19.0/SQL`
2. `StartOracleSQL.w_9_0 myshort myshort PreTranslateKernelStrings.sql > /data/maconomy/w_19_0/MaconomyDir/Database/Logs/PreTranslateKernelStrings.myshort.log`

## 9.4 Create Access Control Views

The access control views ensure that end users only have access to data that they are allowed to see. The access control itself is specified in the Maconomy application. These views are recognized by their name, which is AC<TableName>.

### 9.4.1 Create DHViews

You use the DHViewMake command to create data access security views for the client. You must run DHViewMake before you update common relations.

**To create data access security views for the client, enter the following commands:**

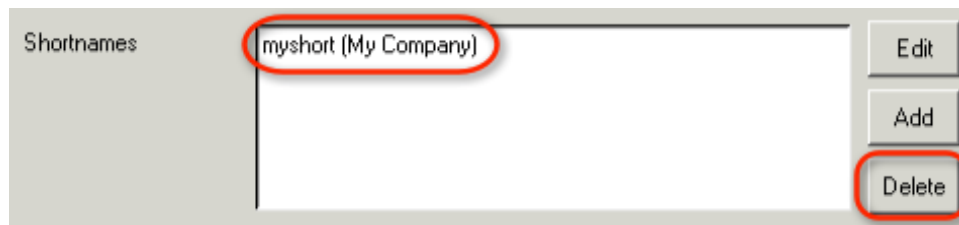
1. `cd /data/maconomy/w_19_0/MaconomyDir/Database`
2. `StartOracleSQL.cmd myshort myshort DHViewMake.sql > Logs/DHViewMake.myshort.log`

### 9.4.2 Detach shortname from the Old Application

Use MConfig to detach the shortname from the old Maconomy environment. MConfig does not touch the database or the original environment. Wait until the upgrade is finished and approved by the customer before you delete the old environment.

**To detach the shortname from the old application, complete the following steps:**

1. Open the old environment using MConfig.



2. Click **Delete** to detach the shortname.



## 10 Update Business Logic and Components



Do not continue until the Unicode conversion is finished without any errors. If there are errors in this section the upgrade may be invalid and unable to be fixed.

### 10.1 W\_19\_0 Upgrade

After the Unicode conversion has finished without error, you must remove the shortname from the old application and attach it to the new application using MConfig (if that is how the transfer of data from the old to the new application is being performed).

#### 10.1.1 Attaching the Shortname

The following commands (in this step and the rest of this document) assume that the application w\_19\_0 has the shortname **myshort** attached (so the MaconomyServer.w\_19\_0 command is available).



When attaching the shortname, be sure to install the new Portal database. Otherwise, the step in which you export part of the database for conversion with ModifyExportFile will fail.

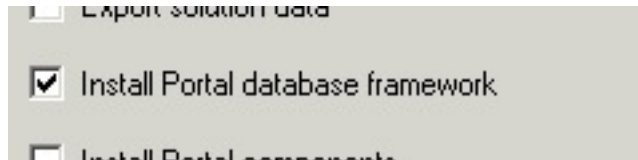


If the system that you are upgrading from has a Special.txt file in the MaconomyDir\Analyze folder, the file must be installed in the MaconomyDir\Analyze folder on the new 2.3 system before new installation numbers are installed.

The Special.txt file contains definitions of special menus, and is needed when generating the Dependencies file. If the file is missing, dialog group assignments in the database for the special menus are automatically removed.

**To install the new Portal database and attach the shortname, complete the following steps:**

1. Select the **Install Portal database framework** check box in the shortname window of the shortname that you are attaching to the new application.

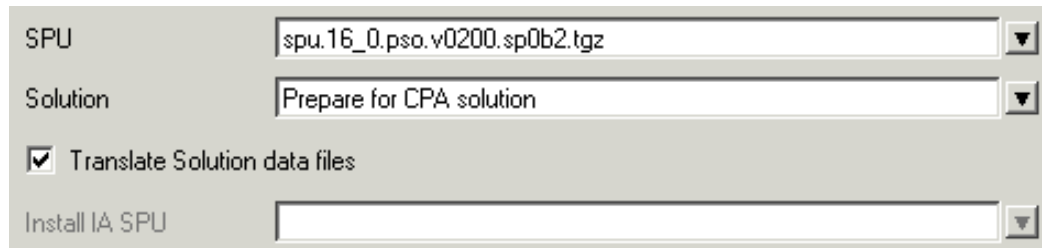


If the shortname is not detached from the old application now, SQL scripts that run using StartOracleSQL.cmd, and so on, may fail, so detach the shortname from the old application now.

If the system that you are upgrading from contains a solution, you must specify the right solution and install it when the shortname is attached (PSO or CPA).

In Maconomy, there is on Custom layer and one Extension layer. This means that adding new customizations or accelerators may overwrite already installed files. This means that the order in which accelerators are installed may be significant.

2. In MConfig go to the Application Instance window and select the SPU and the Solution. The following figure shows some examples.



The screenshot shows a configuration window with the following fields:

- SPU:** A text box containing the value `$pu.16_0.pso.v0200.sp0b2.tgz`.
- Solution:** A dropdown menu with the selected option `Prepare for CPA solution`.
- Translate Solution data files:** A checkbox that is checked.
- Install IA SPU:** An empty text box.



Be aware that these choices cannot be changed. A wrong selection in this section requires a database restore and a reinstallation of the application.

### 10.1.2 Create Other Views

Creation of data access security views for ODBC connections is a bit more complicated. First, you must create the views. Then you must add all windows to the Administrator user and recreate the internal relations between View Groups and views. Then you must recreate access to the views from the list of report users. Finally, you must clean up and reinstall the Analyzer views.

You must perform the following commands.

**To create other views, enter the following commands:**

1. `cd /data/maconomy/w_19_0/MaconomyDir/Database`
2. `StartOracleSQL.cmd myshort myshort ViewMake.sql > Logs/ViewMake.myshort.log`
3. `StartOracleSQL.cmd myshort myshort ViewInit.sql > Logs/ViewInit.myshort.log`
4. `MaconomyServer.w_19_0 -3 -Smyshort`
5. `Recreate3PStuff myshort myshort`
6. `MaconomyServer.w_19_0 -Smyshort -ConvertAnalyzerFiles`

The analyzer views previously deleted are created when running the individual analyzer for the first time; therefore, no special action is needed.

### 10.1.3 Errors

In the log file for the SQL scripts there may be some Oracle errors related to "DROP VIEW" statements.

```
ERROR at line 1:
ORA-00942: table or view does not exist
```

This is normal, and no action is required.

## 10.2 Update Common Relations

This phase ensures that certain global relations that describe the relations and fields in the database contain the correct data. This data depends on the application version only, and is therefore updated during the upgrade procedure.

The command `UpdateFieldsAndRelations` imports the tables `DatabaseField` and `DatabaseRelation` from `Newdb.dbd`. Run this script as the `maconomy` user.

### 10.2.1 Update Common Relations when Upgrading from 9.0 to 17.0



This step may take a long time because it creates instance keys and constraints after importing new field and relation descriptions. **Do not cancel it.**

To update common relations for all platforms from 9.0, enter the following command:

1. `UpdateFieldsAndRelations.w_19_0 myshort myshort`

### 10.2.2 Update Common Relations when Upgrading from 10, 11, 12, 15, 1nd 16 to 17.0

To update common relations for all platforms from 10.0, enter the following command:

1. `UpdateFieldsAndRelations.w_19_0 myshort myshort`

No errors may occur.

2. To test for a successful completion, enter the following command in an interactive SQL session:

```
select RelationName from DatabaseRelation where InternalRelationName =
'Account';
```

The result should be expressions in the enterprise language of the system (the language of all fixed texts stored in the database, such as names of system parameters, and so forth). Be sure that `UserLanguage` in `Maconomy.ini` is set to the company's enterprise language.

### 10.2.3 Create Indexes

As mentioned earlier, each version has its own indexes, and you create the indexes that belong to the new version using `MConfig`. Creating indexes is supported for `MConfig` version 8.0 or later. However, you should always use the newest `MConfig` version unless you have been explicitly instructed otherwise. Consult the `MConfig` documentation if necessary.

### 10.2.4 Create Indexes

Use `MConfig` to create indexes by completing the following steps:

1. In the Main window, select the new application and click **Edit**.
2. In the Application window, for each shortname do the following:
  - Select the shortname in the **Shortnames** list and click **Edit**.
  - In the Shortname window, click **Index management**. You might also want to check the Index tablespace if you are using Oracle and have reattached the existing shortname to the new application using `MConfig`. `MConfig` will not know where to put the indexes; you must check that they are put in the right tablespace yourself.

- Select the **Create standard indexes** check box. If the index comparison revealed local changes to the standard set of indexes, also select the **Create additional indexes** check box and select the comparison file IndexCompareWith<shortname>.<old application>.idx in the **Index spec. files** list box. (There may be comparison files for different shortnames listed; be sure to select the right one).
- 3. Click **OK** to return the Main window. Continue with **Next**, and so on, as for any other MConfig task (if in doubt, see the MConfig manual).



On large databases the creation of indexes typically runs for a long time.

## 10.3 Reinstall Standard Layouts for all Platforms

To reinstall standard layouts, complete the following steps:

1. Type the following commands:
 

```
cd /data/maconomy/w_19_0/MaconomyDir/MSLScripts
MaconomyServer.w_19_0 -Smyshort -xRemove_Old_Layouts
```
2. If upgrading to Maconomy 2.3 or higher from Maconomy 2.2 or lower, the following SQL statements need to be executed against the database before moving forward with the upgrade. If upgrading from 2.3 GA, continue to step 3.

```
ALTER TABLE TEXTDOCUMENT NOLOGGING;
ALTER TABLE TEXTDOCUMENT MODIFY (DOCUMENTDATA CLOB) LOB
(DOCUMENTDATA) STORE AS (NOCACHE NOLOGGING);
ALTER TABLE TEXTDOCUMENT MODIFY LOB (DOCUMENTDATA) (CACHE);
ALTER TABLE TEXTDOCUMENT LOGGING;
```

```
ALTER TABLE BINARYDOCUMENT NOLOGGING;
ALTER TABLE BINARYDOCUMENT MODIFY (DOCUMENTDATA BLOB) LOB
(DOCUMENTDATA) STORE AS (NOCACHE NOLOGGING);
ALTER TABLE BINARYDOCUMENT MODIFY LOB (DOCUMENTDATA) (CACHE);
ALTER TABLE BINARYDOCUMENT LOGGING;
```



This action may require up to double the tablespace used by the binary document table during the execution of this SQL.

3. Run the MaconomyServer command with the **-UW -UP** options to reinstall the standard layouts that were deleted earlier.
 

**Note:** The **-UW** option will return errors because the layer functionality is removed. Ignore these errors.
4. Type the following command to reinstall standard layouts:

```
MaconomyServer.w_19_0 -Smyshort -UW -UP
```



You should not install the layouts until after the conversion of exported data, and you must import them before running the MSL script to convert data, so you should install them in this step.

When you upgrade Maconomy, new versions of layouts are installed. If you do not install the solution because it is no longer in the release, there may not be a new version to install. See the Delttek Maconomy 2.3 GA Release Notes for details.

## 10.4 Create Indexes for Data Conversion



For large databases see [Create Indexes for Data Conversion](#) on page 49.

In general you do not need to create special indexes for the data conversion because the new Convert\_data program will create them automatically.

There are some exceptions, which are nonstandard conversion indexes that are found during the test upgrade. You must specify these indexes in the PULSE schema; you can add them at this point.

To create indexes for data conversion, type the following commands:

1. `create index DataConversion01 on APPROVALLINE (INTEGERKEYVALUE1, INTEGERKEYVALUE2, INTEGERKEYVALUE3) tablespace macindex;`
2. `create index DataConversion02 on APPROVALHEADER (INTEGERKEYVALUE1, INTEGERKEYVALUE2) tablespace macindex;`
3. `create index DataConversion03 on USERINFORMATION (EMPLOYEEENUMBER) tablespace macindex;`
4. `create index DataConversion04 on APPROVALLINE (APPROVALGROUPINSTANCEKEY, EFFECTIVE, INSTANCEKEY) tablespace macindex;`

## 10.5 Convert the Portal Database

When upgrading from Maconomy version 9.0 or 10, use *Appendix: Legacy Portal Conversion* on page 63 instead of the following instructions.

To upgrade the Portal database, complete one of the following sets of steps:

1. `update Component`  
`set Callbackscript = replace(Callbackscript, 'W_12_0', 'W_19_0')`  
`where Callbackscript like '%W_12_0%';`
2. `update Maconomywindow`  
`set SpecFileName = replace(SpecFileName, 'W_12_0', 'W_19_0')`  
`where SpecFileName like '%W_12_0%';`
3. `update MScriptComponent`  
`set Mscriptfilename = replace(Mscriptfilename, 'W_12_0', 'W_19_0')`  
`where Mscriptfilename like '%W_12_0%';`
4. `update Pdmcomponent`  
`set Configurationfile = replace(Configurationfile, 'W_12_0', 'W_19_0')`  
`where Configurationfile like '%W_12_0%';`

#### 5. update Pdmcomponent

```
set Tabfunction = replace(Tabfunction, 'W_12_0', 'W_19_0')
where Tabfunction like '%W_12_0%';
```



In rare cases, the letter W in the version name can be lowercase. For those situations, replace **W** with **w** in the preceding queries, and continue.

## 10.6 Convert Data to the New Version



For large databases see [Convert\\_Data\\_From\\_9.0](#) on page 59.

The purpose of the Convert\_Data.\* MSL scripts is to update the database with the appropriate information for the new version (similar to the OracleUpdate script). This can include entering required data in new fields and/or relations or calculating the values of specific fields.

You can restart Convert\_Data; that is, if it stops because of an error, you can fix the error and then restart the program. This includes restarting if the data conversion is running in the foreground on a UNIX machine and your terminal connection is interrupted for some reason.

In Maconomy version 16 ServicePack 2 the enterprise languages were changed. You therefore update the current enterprise language to match the ones that are available in the new version.

**To update the current enterprise language to match those that are available in the new version, complete the following step:**

1. Run the following MConfig command:

```
MConfig-8.12.exe -r
D:\maconomy\tpu.NTx86.19_0.dir\MConfigScriptlets\ChangeToIsoLanguage.mcp
w_19_0 <myshort> <mypassword> en_US
```

In the preceding example, the enterprise language US is changed to en\_US.

The output looks like the following:

```
MConfig will run the MCP-Scriptlet
'D:\maconomy\tpu.NTx86.19_0.dir\MConfigScriptlets\ChangeToIsoLanguage.mcp'
-----
Log file:
D:\maconomy\tpu.NTx86.19_0.dir\MConfigScriptlets\ChangeToIsoLanguage.log
Setting enterprise language to en_US for application w_19_0
Updating registry for application w_19_0
Updating Maconomy.ini
Updating database
-----
Script done.
You may now close the window.
```



This functionality requires MConfig 8.3 or later.

Before you run `Convert_Data`, check that the `UserLanguage` chosen in `Maconomy.ini` is the same as the Enterprise language of the installation, that is, the language for fixed texts that are stored in the database. You do this by running the `CheckEnterpriseLanguage` MSL script.

You must drop user-created statistics for `Convert_Data` to run; you can recreate these statistics after the script finishes.

```
cd /data/maconomy/w_19_0/MaconomyDir/MSLScripts
MaconomyServer.w_19_0 -xCheckEnterpriseLanguage -Smyshort
```

In general, MSL scripts might ask for input when run.

Variable to change (\$ to list all, <return> to continue, \$\$ to exit)

If you are prompted for this, press **Return** to continue.

If the script completes without errors, the language setting is correct; otherwise, you must change it to the Enterprise language.

Four new variables are introduced in `Convert_data`.

1. `AllowIndexCreationVar` — Boolean. Default value 1 (or true)  
This variable indicates whether it is allowed for `Convert_data` to create needed indexes during the conversion.
2. `TableSpaceNameVar:String` — Default value ' '.  
If `AllowIndexCreationVar` is true this variable specifies the table space for index creation.
3. `OracleDataBaseVar`: Boolean — Default value 0 (or false).  
This variable specifies whether the database is Oracle or SQL Server. The default is SQL Server.
4. `TransactionTypeInitializationVar` — Boolean. Default value 0 (or false).  
This variable specifies whether the optimization described in *Unicode Conversion* on page 21 is performed. `Convert_data` checks the change in the expand script anyway.

Start `Convert_Data` by using the following commands.

```
cd /data/maconomy/w_19_0/MaconomyDir/MSLScripts
MaconomyServer.w_19_0 -xConvert_Data_From_9.0 -Smyshort
```



The script may display lines that state that it is converting, for instance from 9.0 SP 5 to X, while the application installed is on, for instance, X1 SP 2. This is expected behavior, and these conversion steps will be skipped quickly.

Use the following command to check the data conversion status manually:

```
select DATACONVERSIONSTATUS from SYSTEMINFORMATION;
```

If no language is specified in `Maconomy.ini`, the default language of the Maconomy version is selected (US). If in doubt about the enterprise language, select some fixed texts from the database.

```
select DESCRIPTION from SYSTEMPARAMETER where INTERNALNAME < '@B';
```

This should print some texts that are all in the enterprise language. If you are changing the setting for `UserLanguage`, remember to set it back after you run `Convert_Data`.



On large databases this script typically runs for a long time.

**To generate a log file on a UNIX system by starting the ConvertData program, complete the following steps:**

1. Enter the following command:

```
MaconomyServer.w_19_0 -xConvert_Data_from_9.0 -Smyshort < Import.Data >&!  
../Database/Logs/ConvertData.myshort.log &
```

Where Import.Data is a text file created with a text editor and contains only one blank line.

## 10.7 Remove Indexes for Data Conversion



For large databases see [Remove Indexes from Data Conversion](#) on page 60.

You should remove all indexes created for use. If you created the indexes DataConversion01, DataConversion02, and DataConversion03, execute the following commands from an interactive SQL shell (or write a script and execute it).

**To remove indexes for data conversion for all databases, enter the following commands:**

1. `drop index DataConversion01;`
2. `drop index DataConversion02;`
3. `drop index DataConversion03;`



# 11 Validate Custom Components

All standard components are now upgraded, but there may still be customized layouts and Portal components that need to be validated by the new Maconomy version.

## 11.1 Validate Layouts

Modified layouts might need adjustments in the new application version if the following conditions are true:

- Database fields have been removed or renamed.
- Variables have been removed or renamed.
- Cursors have been removed or renamed.
- New mandatory fields have been added.
- The structure of a print layout's parent layout has changed.

Most of the possible problems (renamed fields and relations) should have been handled in the step that converts the database export files that also contain the layouts, so layouts should not fail to validate because of name changes.

However, you must still perform a validation of all modified screen and print layouts against the new application. You perform screen layout validation on the server only—there is no way to validate screen layouts from the Windows client.

You perform print layout validation on the server or from a Maconomy Client.

You must convert all layouts to MPL4. For custom layouts, you do this by running a Java script before importing them. The Java Runtime Environment is part of the TPU.

### 11.1.1 Screen Layout Validation on the Server

**To validate all screen layouts for all platforms, enter the following commands:**

1. `MaconomyServer.w_19_0 -Smyshort -UEW`
2. `MaconomyServer.w_19_0 -Smyshort -UIW`
3. `MaconomyServer.w_19_0 -Smyshort -UVW`

This process validates all of the modified screen layouts in the database for the selected company.

### 11.1.2 Print Layout Validation on the Server

Run the following command to validate customized print layouts.

You can find the `MPL3to4MigrationTool.jar` tool in the `TPU\JavaMPL` directory. If you see the `MPL3MigrationTool.jar` file instead, you should be using a newer TPU. The `MPL3to4MigrationTool.jar` tool converts MPL 3 layouts to MPL 4. Any MPL 1 or MPL 3 layouts remain untouched.

**To validate customized print layouts for all platforms, enter the following commands:**

1. `MaconomyServer.w_19_0 -Smyshort -UEP`
2. `java -jar MPL3to4MigrationTool.jar --dir /data/maconomy/w_19_0/MaconomyDir/ExportedLayouts/`

3. `MaconomyServer.w_19_0 -Smyshort -UIP`
4. `MaconomyServer.w_19_0 -Smyshort -UVP`

## 11.2 Create Constraints

This step introduces consistency constraints that will be checked by the database system during use.

### 11.2.1 Instance Keys

#### All Platforms

```
MaconomyServer.w_19_0 -UC -Smyshort
```

### 11.2.2 Other Constraints

Oracle Specific constraints:

```
cd /data/maconomy/w_19_0/MaconomyDir/Database
StartOracleSQL.cmd myshort myshort CreConstraints.sql >
Logs/CreConstraints.mysort.log
```

Check the log file for errors.

- There may be errors concerning constraints that cannot be created because they already exist. You can ignore those errors.
- There may be errors concerning constraints that cannot be created because they are not fulfilled by the existing data (ORA-02299). These errors are not severe, but you must reported them via a support case to the Maconomy Customer Care, including the log file.
- The error “ORA-22661: such unique or primary key already exists in the table” is a known issue that you can ignore if it appears during the creation of constraints.

The upgrade can continue, and the system can be put into operation, no matter which errors have occurred during the installation of constraints.



Create a support case with Deltek if any errors other than “constraint already exists” occurred. Ignoring constraint creation errors can lead to database inconsistencies.

## 11.3 Install and Upgrade the Portal

If you use the Portal, you must upgrade it to the newest Portal version. You use MConfig to install the new Portal that belongs to the upgraded application and run some scripts and tools to ensure that the sure the database is converted properly. In addition, you must upgrade customized files; however, the process for doing that is not covered in this document. The following information describes how to update the Portal.

In Maconomy, there is *one* Custom layer and *one* Extension layer. This means that adding new customizations or accelerators may overwrite already installed files. This means that the order in which accelerators are installed may be significant.

In the early PSO solutions, industry accelerators were considered customizations and treated as such; that is, they were installed in the custom layer on both the Maconomy server and the web server. This means that “standard” customizations are potentially mixed with “true” customizations.

Since PSO 1.4, industry accelerator files have been installed in a separate Extension layer.

In both cases, Portal components from accelerators are still marked as “not standard” in the Portal Designer.

When upgrading a system from a PSO solution prior to 1.4, you can take **one** of two possible paths for the upgrade:

- Keep the entire custom layer “as is” or
- Remove the accelerator parts and keep only true customizations

If you choose the first option, you should transfer all of the files in the custom layer to the new application both on the Maconomy server and the web server. Going forward, anything that is in the custom layer will be treated as true customizations. No service packs will be released supporting accelerators on the custom layer, and they may hide updated layouts and configurations in the extension layer released in later service packs.

If you choose the second option, the procedure for removing accelerator files from the custom layer depends on the previously installed industry accelerator. You should include this in the upgrade of customizations, which is not covered by this document.

### 11.3.1 Install the Web Server and Portal

You must install the new web server and the Portal for the new application, using MConfig.

**To install the web server and Portal, complete the following steps:**

1. In MConfig, use the Web Products screen to set up web products for the application.
2. Select Install Portal database in shortnames and Install Portal components in shortnames.



Be sure to force the import of Portal components and upgrade of the database schema for the shortname in question.

3. Select **Install Unconditionally** in **(Re)Install Application component files**.
4. If you are installing industry accelerators, select the first accelerator in **Custom Portal files directory**.
5. Click **OK** in the main screen to install.
6. Repeat for each accelerator.

### 11.3.2 Install Maconomy Server Industry Accelerator Files

You can install the industry accelerators that are supplied with the PSO SPU by using the data import feature in MConfig:

**To install the industry accelerators that are supplied with PSO SPU, complete the following steps:**

1. Access MConfig.
2. On the shortname screen, select **Solution Setup Data (import only)** in **Load shortname data**.
3. In **Data configuration options**, select the appropriate industry accelerators and select **Accelerator Runtime Files Only**. This installs files—but no data—in the database.



For third-party accelerators, select **IA.<Accelerator Name> (Import Only) in Load shortname data**. See the documentation for the accelerator for data configuration options, and verify that the upgrade without import option is supported.

### 11.3.3 Set Up the Portal Component

You can install updated versions of solution and industry accelerator components if appropriate. This may be necessary when upgrading from a “native” solution to a PSO solution or when upgrading from a pre-PSO1.4 solution if the “custom” industry accelerators are exchanged for the new standard accelerators.

Use the Portal Designer import features to install updated versions of solution and industry accelerator components, using **one** of the following methods:

- Inside the Portal — Open the Portal Designer » Import / Export component, choose Import, and select the files to be imported.
- From the command line — Type the following commands:

```
cd C:\WebServers\Production\cgi-bin\Maconomy
MaconomyPortal.myshort.W.exe -e -q
"filename=solution_module_comps.xml&username=Administrator&password=123456&i
mporttype=components" Framework/import/importportal.ms
```



If you import from the command line, you must copy the import files to a location that the web server can see, for example, C:\WebServers\Production\MaconomyPortal.

The files to be imported are located in maconomy/w\_19\_0.myapp/Portal/Installation. Component definitions and default role setup and menus are available.

Import files for industry accelerators are located in maconomy/w\_19\_0/IA.<Industry Accelerator>/Portal/Installation

### 11.3.4 Validation of Portal Components

You must manually validate the customized Portal components for the upgraded Portal and database schema.

**To manually validate customized Portal components, complete the following steps, using the Portal Designer:**

1. Choose the component.
2. Click **Update** without changing anything.

If you have a large number of components to validate manually, it is easier to export all of the components using the Portal Designer export menu and then import the components again. Importing also performs the validation. You must export and import after the upgrade—exporting before the upgrade and importing after can cause unpredictable behavior.

You can also export component definitions using the command window. In the following example the web server is a Windows server, and the home directory for the web server is C:\WebServers\Production'.

Example: Export the component definition

```
cd C:\WebServers\Production\cgi-bin\Maconomy
```

```
MaconomyPortal.myshort.W.exe -e -q
"filename=Maconomy2_1.portal_comps.xml&username=Administrator&password=123456&e
xporttype=components" Framework/export/exportportal.ms
```

The export file is located in:

```
C:\WebServers\Production\MaconomyPortal\Maconomy2_1.portal_comps.xml
```

**Example: Import and validate the components**

```
MaconomyPortal.myshort.W.exe -e -q
"filename=Maconomy2_1.portal_comps.xml&username=Administrator&password=123456&imp
orttype=components" Framework/import/importportal.msselect
```

At this point you should update the database statistics for oracle by running the following commands:

```
begin
dbms_stats.delete_schema_stats( ownname=> '"MYSHORT"' );
end;
/

begin
dbms_stats.gather_schema_stats(
ownname=> '"MYSHORT"' ,
options=> 'GATHER AUTO');
end;
/
```

## 11.4 Recreating BPM Objects



This section is only relevant if the original system has BPM.

To recreate all BPM-related database objects, use the “Business Performance Management” section in the shortname window in MConfig. At this point in the upgrade process, all the check boxes in the BPM section should be unchecked. To recreate all BPM database objects, select the relevant values and proceed with the installation. - .



Skipping this step results in the database containing invalid BPM objects that do not match the upgraded schema.

## 11.5 Clean Up

After you have installed a new version you should make sure that no users can connect to the old Maconomy application version. You can decide to completely remove the old version or just to disable it (by changing the port number). Deltek recommends that you remove the version completely using MConfig.

## 12 Large Databases

When upgrading databases larger than 25 GB (datafiles only) Deltek recommends that you read the following information carefully to optimize the upgrade. The philosophy of this approach is to not drop things that you can reuse later in the upgrade procedure.



This should only be considered if upgrading from Maconomy version 11, 12 or 15 to Maconomy version 2.3 GA.

Upgrades from Version 8 may benefit from a special upgrade kit and upgrade procedure. Contact Deltek Engineering for further information.

### 12.1 Drop Constraints (does not apply to 2.3 GA or LA)

Not all constraints need to be removed when upgrading to older versions of Maconomy (2.2 or older) Therefore steps 8.1 and 8.2 can be replaced.

In versions 11 and 12 constraints are not allowed in the following tables:

- JobInvoicing
- Jobregistration
- Dimensioncombusage

**To drop constraints in an Oracle database, complete the following steps:**

1. Enter the following SQL command to find the constraints' names:

```
select table_name, constraint_name from user_constraints where
constraint_type = 'U' and table_name in ('JONINVOICING',
'JOBREGISTRATION', 'DIMENSIONCOMBUSAGE');
```

2. Enter the following SQL command to drop each constraint "C" found in a table "T":

```
alter table T drop constraint C;
```

### 12.2 Dump Changes to Standard Indexes

When you dump index changes from MConfig check whether there are any indexes on the following combination of table name and column name. If there are any, drop them.

- JobInvoiceLine: NOTINCLUDEDINFIXEDPRICE
- JobEntry: NOTINCLUDEDINFIXEDPRICE

**To find and drop indexes in an Oracle database, enter the following command:**

```
1. select 'drop index '||index_name||';' from user_ind_columns where
column_name='NOTINCLUDEDINFIXEDPRICE' group by index_name;
```

Execute the output from the preceding script to drop indexes.

### 12.3 Drop Indexes

You are not required to drop all indexes. They are recreated at a later stage. See the preceding section for information regarding which indexes to delete.

## 12.4 Expand Script



You can speed up upgrades from Maconomy version X or earlier by optimizing the expand script.

To optimize the expand script, complete the following steps:

1. Replace the following:

```
TRANSACTIONTYPE      VARCHAR (FULLFIELDLENGTH)      DEFAULT ' '
```

with:

```
TRANSACTIONTYPE      VARCHAR (FULLFIELDLENGTH)      DEFAULT 'Standard'
```



This change only works if you also perform the changes that are described in [Convert\\_Data\\_From\\_9.0](#).



If the enterprise language is NL, “Standard” is replaced with “Staandard.” It is extremely important that you make the change in the correct expand script. There are 44 entries that must be changed.

## 12.5 Create Index for Data Conversion

You are not required to create any indexes for the data conversion. Convert\_data creates the necessary indexes.

## 12.6 Convert\_Data\_From\_9.0

Four new variables are introduced in Convert\_data\_From\_9.0. You access these variables by entering a \$ sign after the MSL program is started.

The four variables are:

- ALLOWINDEXCREATIONVAR — Type Boolean; default value 1 (Do not create indexes). Indicates whether Convert\_Data is allowed to create indexes for the data conversion.
- TABLESPACENAMEVAR — Type String; default value ‘ ’. Specifies the temporary index tablespace name.
- ORACLEDATABASEVAR — Type Boolean; default value 0. Specifies the database type: 0 = Oracle; 1 = SQL Server.
- TRANSACTIONTYPEINITIALIZEDVAR — Type Boolean; default value 0. If TransactionType is changed in the expand script you must change this variable to 1.



If you do not change any of these variables Convert\_data works as in previous versions.

## 12.7 Remove Indexes from Data Conversion

You are not required to remove any indexes generated by the Convert\_data program because they are automatically removed.



## 13 Appendix: Error Messages

There are two distinct groups of errors that you may run into during an upgrade:

- Oracle errors typically related to rollback space
- Maconomy errors typically related to "ConvertData"

The following two sections cover these in more detail.

### 13.1 Oracle Considerations

Before you perform an upgrade there should be at least 30% free space in the Maconomy tablespace, or if auto extension is used, at least 30% free disk space (to extend tablespaces).

Equally important, enough space should be allocated for the rollback tablespace or enough UNDO\_RETENTION time if undo is used.

How much space you need depends on the following:

- The version you are upgrading from.
- The version you are upgrading to.
- The size of the data.
- The allocation of the data between different tables.

You need more space if you are:

- Moving over several versions (for instance from Version 9 to Version 2.1).
- Have a lot of data (more than 50 GB).
- Have a concentration of data in specific tables. You can check this by looking at the log-file from an export of the company.

You need less space if you:

- Move between versions that are very close (for example, X1 to 2.1).
- Only have limited data (less than 10 GB), in which case you probably do not have to worry about Undo/Rollback.

### 13.2 Using UNDO

Using UNDO is much easier than ROLLBACK and is highly recommended. All you need to worry about is having enough UNDO tablespace and setting the UNDO\_RETENTION time init.ora parameter correctly. The latter may be guesswork, but for large databases an UNDO tablespace of 10GB and a retention time of 4 hours (14400 seconds) is not unusual. After running the update script, resize the UNDO tablespace to normal and set retention time back to normal. Note that the UNDO\_RETENTION parameter can be set dynamically using the ALTER SYSTEM command.

#### 13.2.1 Rollback Segments

Instead of creating or changing Rollback Segments, clients that run Oracle 10g or higher should use Undo Segments.

### 13.2.2 Rule-Based Indexing during Upgrade

It is very important to set the Oracle Instance to operate with Rule-Based Indexing. You do this by adding the following line to the init.ora configuration file:

```
optimize_mode=rule
```

You must then restart the Oracle Instance.

Remember to remove the setting if you are running Cost-Based Indexing. Always remember to update the statistics after the upgrade if you are running Cost-Based Indexing.

## 13.3 Maconomy Errors

If unexpected errors occur, they usually happen while running the ConvertData MSL-program. They can look something like the following:

```
Check_Fatal, Routine 'STATUS33CREATECOMPANYCUSTOMER', line 68, message"
Script Terminated with an error: 'Maconomy system error:
STATUS33CREATECOMPANYCUSTOMER line 68: '
```

Normally these require assistance from Application Development; if you have such an error call Customer Support Services.

## 13.4 MConfig Errors

During an upgrade you use MConfig for the following:

- Dumping local changes to the standard set of indexes.
- Deleting existing indexes.
- Creating indexes.

For these tasks, MConfig “knows” which error messages are acceptable and which are not. Thus, you should not expect error feedback during these actions (and of course not program crashes, either). If errors occur, call Deltek Support.

## 14 Appendix: Legacy Portal Conversion

### 14.1 Export Data for Conversion

You must export some specific relations in the database. These relations contain customized windows and print layouts and Portal components, the system maintenance log, user settings, and customized workflow SQL.



Upgrading from Maconomy 10.0, 11.0, 12.0, and 15.0 only requires the Portal database to be exported for conversion, whereas upgrading from Maconomy 9.0 has many more tables included in the conversion. Use the correct conversion procedure for each upgrade path (if you are upgrading from 9.0, perform the extra steps, or if not, proceed as usual).

The change of directory to where the Upgrade Kit is located makes it easier to locate the database export file in the next step. If the Maconomy application is running on a Unix server, the database export file must be transferred to a Windows computer for further processing.

If you forgot to install the new Portal database (check the “Install Portal database framework” in the shortname window) when attaching the shortname to the new application, you may get errors like the following:

```
Exporting MACONOMYWINDOW      ..      . Records exported:      0

PPU_DebugStr:
DIE_Export: Fetching records from 'Maconomywindow' failed.
Message: 'ORA-00904: "LAYOUTSPEC": invalid identifier'
```

To fix this, start MConfig and apply that change for the new application. Then redo this step.

If some of the relations are missing in the command line, or the command line contains a spelling mistake, the relation is not exported to the database dump, and thus is not converted. The existence of the relation in the database dump is checked during the conversion. See the next step for details.

#### 14.1.1 Back Up Portal Database

To back up the relations to convert when upgrading from Maconomy 10, 11, or 12, enter the following commands:

1. `cd C:\UpgradeKit`
2. `MaconomyServer.w_19_0 -Smyshort -Eunconverted.dbd -R"@Portal"`

#### 14.1.2 Export Data when Upgrading from 9.0 to 17.0



If the Universe Reporting module has been activated, you might need to convert the BinaryDocument table as well. This table can be quite large, and it should only be included in the export if there is data to be converted.

To check whether the BinaryDocument table should be included, complete the following steps:

1. Enter the following command:
 

```
select count(*) from BinaryDocument where ( DocumentGroup='ROE' and
DocumentSubgroup='XUSER' ) or ( DocumentGroup='ROE' and
DocumentSubgroup='XRO' );
```
2. If the count(\*) returns any rows, include the BinaryDocument table.

To export the relations to convert when upgrading from Maconomy 9.0, complete the following steps:

1. `cd C:\UpgradeKit`
2. `MaconomyServer.w_19_0 -Smyshort -Eunconverted.dbd -R"@Portal,TextDocument,Document,PrintoutLayout,Windowlayout,DialogGroup,TargetGroupSpecTypeTable,Monitor,NotificationType,SystemMaintenanceLog,LifeCycle,LifeCycleDefinition,LifeCycleTransitionDefinition,LifeCycleRuleDefinition,LifeCycleStage,LifeCycleStageDefinition"`



If the BinaryDocument should be included, enter the following command:

```
MaconomyServer.w_19_0 -Smyshort -Eunconverted.dbd -R"@Portal,TextDocument,BinaryDocument,Document,PrintoutLayout,Windowlayout,DialogGroup,TargetGroupSpecTypeTable,Monitor,NotificationType,SystemMaintenanceLog,LifeCycle,LifeCycleDefinition,LifeCycleTransitionDefinition,LifeCycleRuleDefinition,LifeCycleStage,LifeCycleStageDefinition"
```

## 14.2 Convert Exported Data

To convert the exported data, you must run the ModifyExportFile tool on the relations that you exported. ModifyExportFile is located in the UpgradeKit directory in the Tools subfolder. Be sure to change to that directory. ModifyExportFile requires the LochLomond tool to be present in that directory, and it must also be able to find the dictionaries to translate with.

The conversion of the exported data is based on conversion scripts, which are version-dependent. Use the conversionspec.w90-w150.txt script when upgrading from Maconomy 9.0. Use conversionspec.w110-w160.txt or conversionspec.w120-w160.txt when upgrading from Maconomy 10.0, 11.0, 12.0, or 15.0. You specify the script file on the command line as shown in the following information.

The result of the conversion will be in the converted.dbd file. The LocalizedFields.log file lists fields that have been localized (for reference).

Run the following command to convert (note that the command assumes that the database export file is located in C:\UpgradeKit-17.0).



The ModifyExportFile tool is part of the Upgrade Kit, so you must change directory to that folder.

To convert exported data on Windows, enter the following commands:

1. `cd D:\UpgradeKit-17.0\Tools`
2. `ModifyExportFile -InputFile ..\unconverted.dbd -OutputFile ..\converted.dbd -RelationFile conversionspec.w120-w160.txt -MaconomyFolder ..\AppFolder`

To convert exported data on Unix, complete the following steps:

1. Transfer unconverted.dbd to a Windows computer that also has the Upgrade Kit unpacked.
2. Enter the following commands:
 

```
cd D:\UpgradeKit-17.0\Tools
ModifyExportFile -InputFile ..\unconverted.dbd -OutputFile ..\converted.dbd -RelationFile conversionspec.w120-w160.txt -MaconomyFolder ..\AppFolder
```

### 3. Transfer converted.dbd to the Unix server.

The output from this command lists all relations in the file and states whether any conversions are required for the relation. The output will resemble the following:

```
WORKFLOWITEM          - unselected found in import file
WORKFLOWITEM          - not exported, not in relation file
```

This is expected. The end result is that XXX relations have been processed.

```
No. of relations in output file :
- Exported from input           :      4
- Skipped from input           :     22
Totalling                       :     26
```

The converted.dbd output file will likely be smaller than unconverted.dbd because not all relations in the input file will be present in the output file. The LocalizedFields.log file will show the fields that have been localized.

A uniqueness test is performed converting the unconverted.dbd database file. In some cases special customizations in the Portal will cause uniqueness problems in certain component field tables.

The following shows an example of how uniqueness problems are reported:

Uniqueness problem in table XXX, key YYY



Contact Customer Support if ModifyExportFile fails with the preceding message. You cannot continue the upgrade until the problems are fixed.

The unconverted.dbd database file is checked for the existence of the relations for conversion, and a warning is issued if one or more tables are missing. Spelling mistakes are the typical reason for missing tables in the export file.

ModifyExportFile issues the following warning if relations are missing:

```
**WARNING: There were relations in the specification file that were not present
in the dump file. Please check if the following relation(s) should be in the
dump file, or make sure that you really don't want these relations to be
converted:
```

```
BinaryDocument
PrintoutLayout
```

The BinaryDocument relation might be missing if you excluded the relation in the export command. If any other relations are missing, you must export the data again, including the missing relations.

## 14.3 Import Converted Data

You must now import the converted layouts and Portal components again, truncating the existing relations.

**To reimport the converted layouts and Portal components, enter the following commands:**

1. `cd C:\UpgradeKit-17.0`
2. `MaconomyServer.w_19_0 -Smyshort -Zt -Iconverted.dbd`

This completes the conversion of Portal data for the upgrade to W 17.0 and the necessary terminology conversion for certain relations.

## 15 Appendix: NameChanger Tool

In Maconomy version 2.3 GA many names, especially those of database fields, have been changed. It can therefore be expected that modifying source files for customer customizations can be a complex task during the upgrade process.

To help to simplify this work the NameChanger tool is released as a part of the Upgrade Kit (it is in the Tools folder). It is provided for all server platforms—Windows, Linux, AIX, and SunOS.

The following is an example of how to use the NameChanger tool:

```
NameChanger.exe -s C:\CustomizedFiles -t C:\ModifiedCustomizations -F 15 -T 17
-y MDL,MDML
```

This command processes all files in the folder tree C:\CustomizedFiles and stores modified files in the folder tree C:\ModifiedCustomizations. In this example only files of types MDL and MDML are modified. All files in C:\CustomizedFiles appear in C:\ModifiedCustomizations, some of them (for example, files of types other than MDL and MDML) possibly unchanged. The source files are for Maconomy version 15 (that is, 2.0), and the modified files are for Maconomy version 17 (that is, 2.3 GA).


If you add the option “-p” (for “preview”) you can see which changes would be done, without any creating any modified files.

The file types that are currently handled are MDL, MDML, MDXL, MEXL, MNSL, MPL, MUL, MWSL, and RDL.

Run the NameChanger tool with the option “-h” (for “help”) to see a complete list of options.

The log file, which is located in the same folder as the NameChanger tool, lists the changes. It also lists occurrences of ambiguities where a term has multiple possible changes. The log file also lists where terms have been changed to something starting with “Deprecated” or “Removed,” since such terms are likely not to be supported in the new version.

Note that Deltek does not guarantee that the NameChanger tool makes all of the required changes to the source files. This tool might sometimes make changes that should not have been made. You must still manually verify the changes. However, tests show that at least when upgrading to version 2.3 GA, the number of correct translations by far outnumbers the number of mistakes.



Deltek is the leading global provider of enterprise software and information solutions for professional services firms, government contractors, and government agencies. For decades, we have delivered actionable insight that empowers our customers to unlock their business potential. Over 14,000 organizations and 1.8 million users in approximately 80 countries around the world rely on Deltek to research and identify opportunities, win new business, optimize resource, streamline operations, and deliver more profitable projects. Deltek – Know more. Do more.®

[deltek.com](http://deltek.com)