


# Deltek Maconomy<sup>®</sup> 2.5.3

Upgrade Guide for Oracle

**July 22, 2021**



---

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 May 2021.

© 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

Overview .....	1
Process by Role .....	1
Compatibility .....	1
Revision History / What's New .....	1
MConfig .....	3
Assumptions and References .....	3
Assumptions .....	3
References .....	3
Version Names .....	4
Main Upgrade Phases .....	6
Workflows .....	7
Overall Workflow .....	7
Consultant Workflow .....	8
TIA Consultants .....	8
Business Consultants .....	11
Extension Consultants .....	12
BPM Consultants .....	13
1 Prerequisites .....	14
1.1 Obtain Upgrade Kit 2.5 .....	14
1.2 Review Hardware and Resources .....	14
1.3 Oracle Considerations .....	15
1.4 Check Upgrade Preconditions .....	15
1.5 Execute Index Tool .....	15
2 Critical Preparations for Upgrade .....	19
2.1 Approval Hierarchies .....	19
Job Quotes .....	19
Approve Quotes Action .....	19
General Journals .....	19
2.2 Enhanced Vendor Invoice Reallocation .....	19
2.3 Invoicing: FinanceEntry of the A/R Control Account .....	20
2.4 Increasing System Numbers .....	20
Table Space Capacity .....	20
Table Space Names .....	21

---

Update Custom BPM Universes .....	21
2.5    My Approvals / Approval Center Name Change .....	22
2.6    Use MyxL Tool to Convert Syntax in Files .....	22
2.7    NameChanger .....	24
2.8    Remove BPM Database Objects .....	25
3    Application Pre-Upgrade.....	26
3.1    Ensure Instancekeys in Userinformation and Approvalgroup Tables .....	26
3.2    Update Layouts for Global and Local Dimensions .....	26
3.3    Local Charts of Accounts .....	28
3.4    Reporting Structures .....	31
3.5    Change References to Depreciated Fields .....	32
4    Installation .....	33
4.1    Install the New Tools (TPU) .....	33
4.2    Install the New Application (APU) .....	33
5    Check Upgrade Preconditions .....	35
5.1    Check Preconditions for Upgrade .....	35
5.2    Change the Default Linesize Value for Oracle .....	36
5.3    Put the Application into Maintenance Mode.....	36
6    Delete Old Components.....	37
6.1    Delete Old Database Objects.....	37
6.2    Dump Materialized View Definitions .....	37
6.3    Delete Views .....	38
6.4    Delete Timestamp Triggers.....	39
6.5    Delete Portal Standard Components .....	39
7    Update the Database Schema.....	41
7.1    Expand Relations .....	41
Expand Script.....	41
Postexpand Script.....	42
7.2    Delete Unused Columns .....	43
7.3    Create Access Control Views.....	43
Create DHViews .....	43
Detach shortname from the Old Application .....	44
8    Update Business Logic and Components .....	45
8.1    [Version] Upgrade .....	45
Attaching the Shortname .....	45
Create Other Views.....	46

---

---

Errors .....	46
8.2 Update Common Relations .....	47
Update Common Relations.....	47
Create Database Objects .....	47
8.3 Reinstall Standard Layouts for all Platforms .....	48
8.4 Convert the Portal Database.....	49
8.5 Convert Data to the New Version.....	50
8.6 Create TimeStamp Triggers .....	51
9 Validate Custom Components .....	52
9.1 Validate Layouts.....	52
Screen Layout Validation on the Server .....	52
Print Layout Validation on the Server .....	52
9.2 Create Constraints .....	53
Instance Keys .....	53
Other Constraints.....	53
9.3 Install and Upgrade the Portal.....	53
Install the Web Server and Portal .....	54
Install Maconomy Server Industry Accelerator Files .....	54
Set Up the Portal Component.....	55
Validation of Portal Components .....	55
10 Post Upgrade Tasks.....	57
10.1 Recreate BPM Objects.....	57
10.2 Clean Up .....	57
10.3 Logging of Database Requests or Updates .....	57
10.4 Disable Maintenance Mode.....	58
Appendix: Error Messages.....	59
Check Fatal Error.....	59

## Overview

This document describes how to upgrade from 2.1 or later to the latest version of Maconomy.

**Note:** For upgrades older than 2.1 a database conversion to Unicode is required. Consult your Technical Consultant.

To upgrade to a version of Maconomy that is not the latest version, consult your Technical Consultant.

## Process by Role

This document indicates which consultant role typically performs each step or sub-step, and includes the roles:

- TIA Consultant
- Business Consultant
- Extension Consultant
- BPM Consultant

You can quickly move to the processes in each role by searching on the role named above, by reviewing the [workflows](#) below, or searching by consultant type.

**Note:** Some steps are performed by only one consultant type. In this case, the consultant type is listed under the top heading. For steps which are performed by multiple consultant types, the consultant type is listed by the relevant sub-step heading.

## Compatibility

Refer to the Deltek Product Support Compatibility Matrix for compatibility confirmation.

[Deltek Product Support Compatibility Matrix](#)

## Revision History / What's New

The table below shows a quick reference of recent changes.

Update	Version	Description
Updated <a href="#">Execute Index Tool</a>	2.5.3 / July 22, 2021	Updated upgrade version warning message.
Updated and removed sections	2.5.3 / June 16, 2021	Updated Execute Index Tool Removed Old Database Objects Updated Create Database Objects

Update	Version	Description
		Removed Drop Constraints for Data Conversion Updated Convert Data to the New Version Removed Recreate Constraints after Data Conversion Removed Clean Up Duplicate Conversions Updated Create TimeStamp Triggers Updated Other Constraints
<a href="#">NameChanger</a>	2.5.3	Edited for clarity, and updated Maconomy version mentioned in procedure.
Added, updated, and removed sections to support new upgrade process	2.5.3	Added <a href="#">Execute Index Tools</a> Updated <a href="#">Delete Old Database Objects</a> Removed Dump Changes to the Standard Index Set Removed Delete Old Indexes Added warning in <a href="#">Delete Timestamp Triggers</a> Updated <a href="#">Create Database Objects</a> Added <a href="#">Drop Constraints for Data Conversion</a> Added <a href="#">Recreate Constraints After Data Conversion</a> Added <a href="#">Clean Up Duplicate Indexes</a> Updated Consultant workflows
<a href="#">Overview</a>	2.5.1	Added a document Overview, including a link to the Compatibility Matrix, explanation of roles, and application references.
<a href="#">Workflows</a>	2.5.1	Added overall workflow as well as workflows by role.
[Streamlining and editing]	2.5.1	In general, with this release, we have streamlined material, removed old content, and edited for clarity.

## MConfig

MConfig is the installation and configuration tool for Maconomy. Always use the most recent MConfig version unless you have been explicitly instructed otherwise.

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

## Assumptions and References

### Assumptions

This document describes each step needed for the upgrade. For consistency and ease of use, you can make the following assumptions:

- All examples are based on one company, and this company has the shortname **myshort**, and uses the password, **password**.
- The preferred version database is Oracle 12c. This document does not describe how to upgrade database versions.
- For the purpose of this guide we will assume two applications exist. We call the base application `<oldapplication>` and the target application `<application>`. See the full list of references below.
- We assume that the main drive is 'C' for Windows environments.
- When updating from one application to another, run scripts to ensure proper functioning of the new application version. The programs are located in the bin directory in the TPU that is installed for the application. For example, the 'C:\maconomy\tpu.NTx86.17\_0.p105.dir\bin' folder (and not the 'C:\maconomy\bin' folder, which is the global TPU folder). In this document, we call the TPU directory `<tpudir>`.

### References

In this document, we use a number of references like `<application>`, `<shortname>`, `<MConfig program>` and `<password>` to replace specific versions or details within command lines.

For example, these commands:

```
MaconomyServer.<application> -S<shortname> -xRemoveSolutionWindowLayouts
<MConfig program> -r <scriptlet file> <application> <shortname> <password>
```

might translate to:

```
MaconomyServer.w_20_0.prod -Smacoprod -xRemoveSolutionWindowLayouts
MConfig-9.5 -r DeleteAnalyzerViews w_20_0.prod myshort password
```

Reference	Examples
<code>&lt;version&gt;</code>	w_20_0
<code>&lt;oldapplication&gt;</code>	w_17_0 w_19_0.prod



Reference	Examples
<application>	w_20_0 w_21_0.prod
<applicationpath>	C:\maconomy\w_21_0 /usr/maconomy/w_21_0.prod
<shortname>	macoprod
<password>	somepassword
<tpudir>	/usr/maconomy/tpu.NTx86.20_0.p104.dir C:\maconomy\tpu.NTx86.20_0.p104.dir
<MConfig>	C:\PUs\MConfig-9.5.exe ask Darius for Linux version
<packingunits>	C:\PUs ask Darius for Linux version
<oldschema>	w170sv3
<newschema>	W210sv1

## Version Names

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

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

External Maconomy Version	Internal Maconomy Version
X	10
X+	11
X1	12
2.0	15
2.1.1	16.0 SP1
2.1.4	16.0 SP4
2.2	17
2.3	19
2.4	20

External Maconomy Version	Internal Maconomy Version
2.5	21

## Main Upgrade Phases

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

- Review prerequisites and perform critical preparations prior to upgrade
- Install tools and the application
- Delete old components
- Expand and convert databases and data
- Create components
- Install the new Portal
- Convert source files
- Install source files

## Workflows

### Overall Workflow

The following are the high-level tasks to accomplish with.

Step	Task	Consultant Role
<b><i>Pre-Upgrade Tasks</i></b>		
1	Prerequisites	TIA
2	Critical Preparations for Upgrade	Business Extension BPM TIA
3	Application Pre-Upgrade	TIA Business Extension
<b><i>Upgrade</i></b>		
4	Installation	TIA
5	Check Upgrade Preconditions	TIA
6	Delete Old Components	TIA
7	Update the Database Schema	TIA
8	Update Business Logic and Components	TIA
9	Validate Custom Components	TIA
<b><i>Post-Upgrade Tasks</i></b>		
10	Post-Upgrade Tasks	TIA

## Consultant Workflow

### TIA Consultants

**Note:** TIA Consultants perform tasks within each step. Note that in steps 2 and 3, TIA must coordinate with Business, BPM, and Extension consultants, as detailed below.



Pre-Upgrade Tasks			
Step 1	<b>Prerequisites</b>	Obtain Upgrade Kit	(TIA)
		Review Hardware and Resources	(TIA)
		Oracle Considerations	(TIA)
		Check Upgrade Preconditions	(TIA)
Step 2	<b>Critical Preparation for Upgrade</b>	Approval Hierarchies	(Business)
		Vendor Invoice Reallocation	(Business)
		Invoicing Finance Entry	(Business)
		Increasing System Numbers	(Extension)
		Increasing System Numbers	(BPM)
		Table Space Capacity	(TIA)
		Table Space Names	(TIA)
		Update Custom BPM Universes	(BPM)
		My Approvals	(Extension)
		MyxL Tool Convert Syntax Files	(Extension)
		NameChanger	(Extension)
		Remove BPM Database Objects	(TIA)
		Ensure Instancekeys in Userinformation	(TIA)
Step 3	<b>Application Pre-Upgrade</b>	Update Layouts for Dimensions	(Business)
		Local Chart of Accounts	(Business)
		Reporting Structures	(Business)
		Change References to Depreciated Fields	(Extension)

Upgrade Tasks			
Step 4	Installation	Install the New Tools	(TIA)
		Install the New Application	(TIA)
Step 5	Check Upgrade Preconditions	Check Preconditions for Upgrade	(TIA)
		Change the Default Linesize for Oracle	(TIA)
		Put the Application into Maintenance Mode	(TIA)
Step 6	Delete Old Components	Delete Old Database Objects	(TIA)
		Dump Materialized View Definitions	(TIA)
		Delete Views	(TIA)
		Delete Portal Standard Components	(TIA)
Step 7	Update the Database Schema	Expand Relations	(TIA)
		Delete Unused Columns	(TIA)
		Create Access Control Views	
Step 8	Update Business Logic and Components	[Version] Upgrade	(TIA)
		Update Common Relations	(TIA)
		Reinstall Standard Layouts	(TIA)
		Convert the Portal Database	
		Convert Data to the New Version	(TIA)
		Recreate Constraints after Data Conversion	(TIA)
		Clean Up Duplicate Indexes	(TIA)
Step 9	Validate Custom Components	Validate Layouts	(TIA)
		Create Constraints	(TIA)
		Install and Upgrade the Portal	(TIA)
Post-Upgrade Tasks			
Step 10	Post-Upgrade Tasks	Recreate BPM Objects	(TIA)
		Clean Up	(TIA)
		Advanced Logging	(TIA)
		Disable Maintenance Mode	(TIA)



## Business Consultants



**Note:** Business Consultants perform pre-upgrade tasks in steps 2 and 3 only.

Pre-Upgrade Tasks		
Step 1	Prerequisites	(TIA)
 Step 2	Critical Preparations for Upgrade	Approval Hierarchies (Business)
		Vendor Invoice Reallocation (Business)
		Invoicing Finance Entry (Business)
		Increasing System Numbers (Extension)
		Increasing System Numbers (BPM)
		Table Space Capacity (TIA)
		Table Space Names (TIA)
		Update Custom BPM Universes (BPM)
		My Approvals (Extension)
		MyxL Tool Convert Syntax Files (Extension)
		NameChanger (Extension)
		Remove BPM Database Objects (TIA)
 Step 3	Application Pre-Upgrade	Ensure Instance Keys (Business)
		Update Layouts for Dimensions (Business)
		Local Chart of Accounts (Business)
		Reporting Structures (Business)
		Change References to Depreciated Fields (Extension)
Upgrade and Post-Upgrade Tasks		
Steps 4 - 10	Upgrade and Post-Upgrade	(TIA)



## Extension Consultants

**Note:** Extension Consultants pre-upgrade tasks in steps 2 and 3 only.

Pre-Upgrade Tasks		
Step 1	Prerequisites	(TIA)
 Step 2	Critical Preparation for Upgrade	Approval Hierarchies (Business)
		Vendor Invoice Reallocation (Business)
		Invoicing Finance Entry (Business)
		Increasing System Numbers (Extension)
		Increasing System Numbers (BPM)
		Table Space Capacity (TIA)
		Table Space Names (TIA)
		Update Custom BPM Universes (BPM)
		My Approvals (Extension)
		MyxL Tool Convert Syntax Files (Extension)
		NameChanger (Extension)
		Remove BPM Database Objects (TIA)
 Step 3	Application Pre-Upgrade	Ensure Instance Keys (Business)
		Update Layouts for Dimensions (Business)
		Local Chart of Accounts (Business)
		Reporting Structures (Business)
		Change References to Depreciated Fields (Extension)
Upgrade and Post-Upgrade Tasks		
Steps 4 - 10	Upgrade and Post-Upgrade	(TIA)

## BPM Consultants

**Note:** BPM Consultants perform pre-upgrade tasks in step 2.

Pre-Upgrade Tasks		
Step 1	Prerequisites	(TIA)
Step 2	Critical Preparation for Upgrade	Approval Hierarchies (Business)
		Vendor Invoice Reallocation (Business)
		Invoicing Finance Entry (Business)
		Increasing System Numbers (Extension)
		Increasing System Numbers (BPM)
		Table Space Capacity (TIA)
		Table Space Names (TIA)
		Update Custom BPM Universes (BPM)
		My Approvals (Extension)
		MyxL Tool Convert Syntax Files (Extension)
		Namechanger (Extension)
		Remove BPM Database Objects (TIA)
Step 3	Application Pre-Upgrade	Ensure Instance Keys (Business)
		Update Layouts for Dimensions (Business)
		Local Chart of Accounts (Business)
		Reporting Structures (Business)
		Change References to Depreciated Fields (Extension)
Upgrade and Post-Upgrade Tasks		
Steps 4 - 10	Upgrade and Post-Upgrade	(TIA)

# 1 Prerequisites

Be sure that the following prerequisites are met before you begin the upgrade process, including:

- Obtain the upgrade kit
- Review hardware and resources
- Check upgrade preconditions
- Perform any necessary [critical actions prior to upgrade](#)

## 1.1 Obtain Upgrade Kit 2.5

*TIA Consultant*

You must obtain the Upgrade Kit for 2.5 as well as the TPU and the APU. This document is included in the Upgrade Kit, found on DSM

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.

## 1.2 Review Hardware and Resources

*TIA Consultant*

We recommended you 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.

**Note:** 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.

## 1.3 Oracle Considerations

*TIA Consultant*

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 2.2 to Version 2.5).
- 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, 2.4 to 2.5).
- Only have limited data (less than 10 GB), in which case you probably do not have to worry about Undo/Rollback.

## 1.4 Check Upgrade Preconditions

*TIA Consultant*

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.

## 1.5 Execute Index Tool

*TIA Consultant*

Perform this task performed when the new <application> is installed (Maconomy 2.5.3 or higher) and before any modifications to the database of <oldapplication> have been performed.

Version 2.5.3 implements a new method of managing indexes and constraints.

**Perform the following steps before proceeding with the upgrade process:**

1. Copy StandardIndexesAndConstraints.json file belonging to the new application (*not* the original file being upgraded) from  
    <applicationpath>\MaconomyDir\Database  
to

- maconomy tmp folder (usually c:\maconomy\tmp)
2. Copy all files from  
Upgradekit\dbmSripts\ora directory  
to  
maconomy bin folder (usually c:\maconomy\bin)
  3. Execute the following commands (paths assume locations listed in previous steps):  
CD c:\maconomy\bin  
MaconomyServer.<application>.cmd -iMaconomyServer.<oldapplication> -  
S<shortname> --IndexScriptFile "idxm.primary.ora.txt"  
MaconomyServer.<application>.cmd -iMaconomyServer.<oldapplication> -  
S<shortname> --IndexScriptFile "getcustomindexes.txt"  
MaconomyServer.<application>.cmd -iMaconomyServer.<oldapplication> -  
S<shortname> --IndexScriptFile "consmDrop.ora.txt"

**Note:** Executing the MaconomyServer belonging to <application> using the .I file from <oldapplication> allows accurate comparison of database objects between the two applications.

4. Review the two output files, found in the C:\maconomy\Tmp folder, to ensure there are no errors during execution, that both files exist, and that dropScript.txt as well as dropCons.txt are not empty. customCreate.txt can be empty if no custom indexes exist on the database.

**Notes:**

customCreate.txt will contain objects related to maconomy\_users table in certain scenarios.

customCreate.txt otherwise will contain only non standard indexes. *This file will not be referred to moving forward in the upgrade process.*

If there is a need to recreate custom indexes, carry this out after the upgrade is complete.

Depending on the indexes, some manual adjustments may need to be done to account for changes in the Maconomy database schema.

5. Check that the following three files are created in c:\maconomy\tmp:

dropScript.txt

customCreate.txt

dropCons.txt

**Warning:** It is only possible to execute the above described procedure utilizing new MaconomyServer on the database attached to the old application. This requires that the 2.5.3 MaconomyServer is compatible with the application being upgraded.

Currently the following versions are known to work for direct upgrades: Maconomy 2.4 and Maconomy 2.3, including all maintenance releases. Also, if upgrading from Maconomy 2.3, review 2.5.3 CU1 Release Notes for additional steps.

<https://dsm.deltek.com/documentationlists/DeltekMaconomy23GA.html>

If you are using a database platform other than one supported by MaconomyServer 2.5.3+, the recommended procedure is to upgrade into Maconomy 2.5.2 using the former upgrade procedure, and subsequently patch the application to 2.5.3. If taking this course of action, ensure that the 2.5.2 application has 2.5.2 CU3 applied prior to upgrading, to avoid a known data conversion issue.

See the [Deltek Product Support Compatibility Matrix](#) as needed for details.

**Known Issue:** When upgrading directly (from any version?) to 2.5.3, there is a known performance issue that slows down data conversion and may prevent it from completing entirely on large databases. This is expected to be fixed in an upcoming CU.

## 2 Critical Preparations for Upgrade

**Warning:** This section contains actions that **MUST** be performed prior to upgrade. Note that failure to do so could result in fatal errors requiring a full restart of the upgrade process, or worse, such as production environments running with inconsistent data.

### 2.1 Approval Hierarchies

*Business Consultant*

#### Job Quotes

Maconomy version 2.5 also has a simple approval hierarchy setup in place to allow automatic submission and approval of historic job quotes. This setup will allow companies who choose to opt out of approval hierarchies for job quotes to continue using their existing workflows.

#### Approve Quotes Action

The ApproveQuote action no longer runs an extension code. This impacts customers using a standard layout (in other words, not customized) in the Quote Editing workspace (**Jobs » Budgeting » Quote**) with an extension connected to the ApproveQuote. This occurs because approval hierarchies do not actually use the ApproveQuote action but the ApproveAll actions borrowed from the Approval Groups and Approval Objects dialogs. This will not be caught during compilation or during run time, which may cause an issue where the extension code is unused and unnoticed.

If you want to continue using the ApproveQuote extensions code, you must customize the layout in the Quote Editing workspace to not use the Approval actions from the approval group, but instead use an extension instead.

#### General Journals

If submitted general journals existed in the system prior to upgrading Maconomy to version 2.5, then a new simple approval hierarchy is created for General journal where approval on header is required, and everyone can approve.

Maconomy creates approval groups for all existing submitted general journals and sets them as approved.

### 2.2 Enhanced Vendor Invoice Reallocation

*Business Consultant*

In the vendor invoices workspace, the two dialogs Invoice Allocation and Invoice Reallocation are now combined into a single workspace, Invoice Allocation. In this workspace, the content of visible fields and actions change depending on the posted status of the vendor invoice. This change to the vendor invoices workspace could affect customized layouts and extensions that previously were dependent on the Invoice Reallocation dialog. Note that as a result, updates to customized layouts could be required.



## 2.3 Invoicing: FinanceEntry of the A/R Control Account

*Business Consultant*

For A/R Control Finance Entries, the Customer Number field now contains the Delivery Customer and not the Payment Customer (bill to) as it did previously. As part of a bug fix, the field Payment Customer for A/R Control Finance Entries now correctly shows the Payment Customer.

Reports are changed to reflect this. If you are using the field in one of your reports, note that the content of the field is changed, and it could affect the reports which you may then have to make changes to accommodate.

## 2.4 Increasing System Numbers

*Extensions Consultant*

*BPM Consultant*

With the Increase System Numbers feature, you must carry out the following actions during upgrade:

- All customizations must be upgraded, including BPM custom universes and reports.
- Data conversion

**Note:** The data conversion is moderate and adds about 15 minutes for each 10 million rows in the database.

- Update Extensions to adapt to the changed API.

### **Additional Actions:**

- **Import Programs** — Import programs have an optional double quote around the string fields and if they are left the out the field is parsed according to the type of the field which means that no import programs must be updated.
- **BPM Universes** — BPM universes must be updated where the system numbers are used.
- **Notifications** —All notifications are recalculated upon upgrade and the notifications are initially empty until a user refreshes the list or a background task refreshes the list.

## Table Space Capacity

*TIA Consultant*

The Database must have sufficient free table space in order to make a copy of the largest table with a system number. The largest table will likely be one of `FinanceEntry`, `FinanceEntry`, `DailyTimeSheetLine`, `TimeSheetLine` or `GeneralJournal` but it may be another.

## Table Space Names

*TIA Consultant*

The default setup of Maconomy stores all relations in the same table space which also happens to be the default table space in the database. If the setup is changed to have tables in different table spaces (for example, utilizing separate hard drives), then the expand script must be altered manually or a post expand script cleanup is needed to move the relations to the desired table space.

If the current relations are not in the default table space, then all changed tables mentioned will end up in the default table space. Temporarily changing the default table space to point to the desired location of the relation during the expand script can solve the issue. Otherwise manual altering of the expand script or post expand cleanup is needed.

## Update Custom BPM Universes

*BPM Consultant*

The Increase System Numbers feature changed some database fields from datatype Integer to datatype String. For all standard reports and universes, BPM defining objects on these changed fields are updated.

However, you must update these fields on custom universes and custom reports.

To update custom UNX universes:

- 1 In SAP Information Design Tool, retrieve the needed universe. This provides the DFX, BLX and CNS files.
- 2 Open the DFX file.
- 3 Run Refresh Structure. This changes the metadata of database fields for which the datatype has been changed.
- 4 Run Integrity Check on the BLX file. This highlights all objects with incorrect datatypes.
- 5 Update these highlighted objects to the new datatype (String).
- 6 Save and republish the universe.

**To update custom UNV universes:**

- 1 In SAP Information Design Tool, import the needed universe.
- 2 Open each universe dfx-file, set the connection, and run Refresh Structure. This updates the meta information about tables and fields in the database.
- 3 Open each universe blx-file and run Integrity Check.
- 4 Correct any objects or universe part that appears to fail the check.
- 5 Save the universe files and publish the universes.

**To update custom reports:**

- 1 Open a custom report in Web Intelligence, and cancel refresh.
- 2 In the report, identify the objects that have changed datatype.

In each report document, look for errors in report cells and variables (typically marked with question marks [?] ). These are references to universe objects that no longer exist.

- 3 Correct the missing parts and if needed customize the universe to include missing parts. If using custom universes, the reports should work without issues.
- 4 After the report has been corrected, refresh the report and test the results.
- 5 Save the report.

This must be done for each custom report that uses any of the corrected universes.

### Places to Run Data Conversion

Run the Data Conversion (can be updated in parallel):

1. Production system
2. Data warehouse

## 2.5 My Approvals / Approval Center Name Change

*Extension Consultant*

The workspace has changed file name / workspace name from "MyApprovals" to "ApprovalCenter."

Customized global menus and workspace files (\*.mws.xml and \*.mdml.xml) must be updated to refer to "ApprovalCenter."

## 2.6 Use MyxL Tool to Convert Syntax in Files

*Extension Consultant*

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 Maconomy version 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

**To convert files using the MyxL tool:**

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

**Note:** 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.

Supply various parameters to successfully run the conversion. The parameters are provided either in the Convert-20.0.101.0.ini file, or as command line arguments if the Converter is run from the command line.

- **convertedirectory** — *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 **convertedirectory** 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.
3. 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)

**Note:** 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.

4. Submit the changes back to GitHub.

**Notes:**

- **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, 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 2.4 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, no conversion is necessary.

**Note:** See *Deltak Maconomy Language Quick Reference MDML* for details.

## 2.7 NameChanger

### *Extensions Consultant*

In Maconomy version 2.2, many names, especially of database fields, were changed. For version 2.2 onwards, you will need to modify source files for customizations during the upgrading process. The NameChanger is a tool, released as a part of the Upgrade Kit, to facilitate this process. It is released for all server platforms, including Windows and Linux.

NameChanger processes the following file types: MDL, MDML, MDXL, MEXL, MNSL, MPL, MUL, MWSL, and RDL.

The languages MDL, MPL, and MDML are processed in an advanced way, considering for example which source files belongs to which dialogs or relations.

Run the tool with option '-h' (for "help") to see a complete list of options.

**Note:** The tool DOES NOT make all required changes to the source files. The tool may even sometimes make changes that should not have been done. You must manually verify after running the tool.

**To run NameChanger:**

1. From a command line, enter a line such as the following example:

```
NameChanger.exe -s C:\CustomizedFiles -t C:\ModifiedCustom -F 19 -T 21 -y MDL,MDML
```

This command processed all files in the specified folder tree (in this example, C:\CustomizedFiles) and stores modified files in another folder tree (in this example, C:\ModifiedCustom).

In this example, only files of types MDL and MDML are modified. All files in C:\CustomizedFiles will appear in C:\ModifiedCustomizations, some of them (for example, files of other types than MDL and MDML) possibly unchanged. The source files are for Maconomy version 19 (that is, 2.3), and the modified files for Maconomy version 21 (that is, 2.5).

2. Add the option '-p' (for "preview") to see which changes will be made prior to modifying any files.  
The log files, located in the same folder as the tool, list the changes. They also list occurrences of ambiguities where a term has multiple possible changes, as well as occurrences of terms that are removed.  
The log also lists where terms are changed to something starting with "Deprecated" or "Removed," since such terms are not likely supported in the new version.
3. Review two log files:
  - NameChangerLog.session.txt — Contains log data for the latest session.
  - NameChangerLog.accumulated.txt — Contains accumulates log data for all sessions.
4. Use the Extender's static validation to ensure that everything is correct. If the NameChanger made a mistake, it is flagged by the static validation/compilation of your MDMLs and MPLs.
5. Correct any errors flagged by the Extender manually.
6. Keep the log files and consult them later if issues later occur for certain files.

## 2.8 Remove BPM Database Objects

*TIA Consultant*

When upgrading to Maconomy 2.5, you must remove all BPM-related database objects from the Maconomy database, including materialized views, materialized view logs, database views, and triggers.

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

### **To remove BPM-related objects from the database:**

1. Deselect all 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.

## 3 Application Pre-Upgrade

### 3.1 Ensure Instancekeys in Userinformation and Approvalgroup Tables

*TIA Consultant*

Ensure that the `approvalgroup` and `userinformation` tables have `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 **Approval**.
3. Ensure the **Instance Keys Applied** field shows **Yes**. If so continue to step 5.
4. If the **Instance Keys Applied** field shows **No**, click the action **Initialize Instance Keys**.
5. Select the Relation Name **User**.
6. Ensure the **Instance Keys Applied** field shows **Yes**. If so, no further action is needed.
7. If the **Instance Keys Applied** field shows **No**, click the action **Initialize Instance Keys**.

It is important that the **Approval** relation gets instance keys prior to **User**.

### 3.2 Update Layouts for Global and Local Dimensions

*Business Consultant*

**Note:** This section applies only if you are upgrading from a pre-2.3 version. If upgrading from 2.3 LA or newer, skip this section.

Several global and local dimensions were 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 indicates 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

Maconomy 2.3 introduced 13 new dimensions: Specification 4-10, Local Specification 4-10, and Local Account. Due to this, during the upgrade, we can create the 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 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 an 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**, during the upgrade a **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** is 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 an 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**, during the upgrade a **Local Specification 5** list is created with the name specified in **Remarks 1** field, in this case **Test**, and standard name specified in the **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** is set to **Local Spec. 5** list, and **Standard Name** is set to [ - ] (dash).

**Example of Setup**

The following 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

**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	

### 3.3 Local Charts of Accounts

#### *Business Consultant*

Maconomy 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.

**Note:** 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**.

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 2.4 in relation to the Local Chart of Account functionality, 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 and later, because it lists all of the problems that must be corrected before the upgrade to avoid various problems during or after the upgrade.

**Note:** Because 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**.

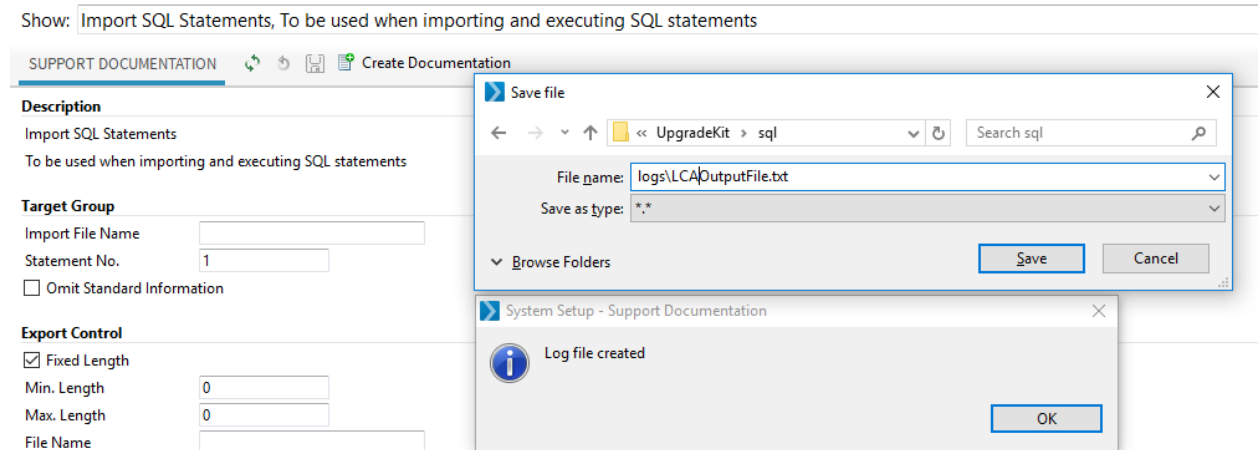
## Run the Script

### To run the script:

1. Locate the script in the Upgrade Kit:  
`<UpgradeKit->\sql\LocalChartOfAccountsPreUpgrade.sql`
2. Open the Workspace/Java client of the version you are upgrading. For example, if you are upgrading from 2.1, 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 the following image).

7. Click **Save** to save this output file so that you can investigate and verify that the system setup fulfills 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 following sample output is color-coded to more easily indicate the content of the output and any action that 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.

## 3.4 Reporting Structures

### *Business Consultant*

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 do no longer exist. You must create a reporting structure for all of the structure tables (most important is the account structure) 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.

## 3.5 Change References to Depreciated Fields

*Extensions Consultant*

Below are changed references to depreciated fields for versions 2.1 – 2.5. Change the setup before you perform the upgrade.

**The following fields were added in version 2.1:**

- **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.

**The following fields were added in version 2.5:**

- CompensationType.NumberOfPayPeriods
- CompensationAgreement.NumberOfPayPeriods
- CompensationAgreement.ValueOfAgreementPaymentCurrency
- CompensationAgreement.ValueOfAgreementCurrency
- CompensationAgreement.ExpectedValueOfAgreementCurrency

## 4 Installation

*TIA Consultant*

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

### 4.1 Install the New Tools (TPU)

You must install a new TPU when upgrading, preferably the most recent one that belongs to the application.

To install the upgrade:

1. Copy MConfig, TPU, and the APU to the <packingunits> directory on the server.
2. Use MConfig to specify the location of that directory. Go to 'Global Settings' and define the directory in the Packing Units Directory.

To use the new TPU and the new application, installing the new TPU as global tools is required.

**To install the new TPU:**

1. Start MConfig.
2. In the Tools window:
  - a. Select the new TPU for installation.
  - b. Use the new TPU as Global Tools.

### 4.2 Install the New Application (APU)

Use MConfig to install the new 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.

**To check the enterprise language on an existing database:**

1. Run this in the command prompt to login sqlplus:

**Win:**

```
set ORACLE_SID=<sid>
```

**Unix:**

```
export ORACLE_SID=<sid>
```

2. Log in the database:

```
sqlplus <shortname>/<password>
```

3. 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.

If the original application has an SPU installed and is not a legacy MAS or MCS solution please make sure to install the matching SPU on the new application and remember to prepare the system for the selected solution during the main application installation process.

This concludes the installation of the new application.

## 5 Check Upgrade Preconditions

*TIA Consultant*

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.

**Note:** 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.

### 5.1 Check Preconditions for Upgrade

**Note:** There is one upgrade precondition script for each old version.

You must select the appropriate script, for example:

- Oracle: UpgrPrecond.w12sv3-w210.sql

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

#### Win

```
<applicationpath>\MaconomyDir\Installation\version.info
```

#### Unix

```
<applicationpath>/MaconomyDir/Installation/version.info
```

For example, when upgrading from W 16.0 schema version 1 on Oracle, you must use the UpgrPrecond.w160sv1-w210.sql script.

#### To check preconditions for the upgrade on Oracle:

1. Navigate to the target application folder and type the following commands:

##### Win:

```
cd <applicationpath>\MaconomyDir\Database
StartOracleSQL2 <shortname> <password> UpgrPrecond.<oldschema>-
<newschema>.sql c:\Logs\UpgrdPrecond.<shortname>.log
```

##### UNIX:

```
cd <applicationpath>/MaconomyDir/Database
StartOracleSQL2 <shortname> <password> UpgrPrecond. <oldschema>-
<newschema>.sql Logs/UpgrdPrecond.<shortname>.log
```

2. Verify results in the log file.

The result should be 0 rows selected in all cases. 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.



If the upgrade precondition script returns a message such as “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.

## 5.2 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 default linesize value:**

1. Edit **one** of the following Site Profile script glogin.sql files, depending on your operating system:

**Win:**

```
%ORACLE_HOME%\sqlplus\admin\glogin.sql
```

**Unix:**

```
$ORACLE_HOME/sqlplus/admin/glogin.sql
```

2. Add the following line to the file **glogin.sql**:

```
set LINESIZE 3000
```

3. Check that there are no other changes to the linesize value in the glogin.sql file, then save and close the file.

## 5.3 Put the Application into Maintenance Mode

Prior to starting the upgrade, make sure the Maconomy application you are upgrading is set to Maintenance Mode using MConfig. This is required to ensure background tasks are not executed during the upgrade process. The new Maconomy application is in maintenance mode by default as a new installation. There is no maintenance mode for Maconomy versions earlier than 2.3.

**Note:** If you have extensions, you must reinstall these after installing a service pack before de-selecting the Maintenance Mode field.

**To enable Maintenance Mode:**

1. In the Application window, select the **Maintenance Mode** field.
2. Click **OK > Next > Next** to apply.
3. Complete your maintenance tasks.
4. Go back to the Application window, and de-select the Maintenance Mode field.
5. Click **OK > Next > Next** to apply.

## 6 Delete Old Components

*TIA Consultants*

### 6.1 Delete Old Database Objects

**To delete old database objects:**

1. Run the following commands:

```
cd c:\maconomy\tmp
```

```
StartOracleSQL <shortname> <password> dropScript.txt > c:\Logs\dropScript.log
```

```
StartOracleSQL <shortname> <password> dropCons.txt > c:\Logs\dropCons.log
```

Once executed, check the log. Acceptable errors include those indicating not able to drop objects because they do not exist. Other errors should be investigated.

**Note:**

**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;
```

### 6.2 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:**

1. Log into sqlplus.
2. Enter the following SQL command:

```
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
MYSHORT                                JOBINVOICELINE                            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
      ...

```

## 6.3 Delete Views

Three different types of views are in use by Maconomy:

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

You must remove these before you can perform an upgrade.

**To delete the views for the user run the following commands:**

**Win:**

```

cd <tpudir>\MConfigScriptlets
DeleteDHViews <shortname> <password>
Delete3PStuff <shortname> <password>
<MConfig> -r DeleteAnalyzerViews <application> <shortname> <password>

```

**UNIX:**

```

cd <tpudir>/MConfigScriptlets
DeleteDHViews <shortname> <password>
Delete3PStuff <shortname> <password>
<MConfig> -r DeleteAnalyzerViews <application> <shortname> <password>

```

The output from the delete views scripts for both 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.

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.

## 6.4 Delete Timestamp Triggers

### Win:

```
cd c:\Maconomy\<oldapplication>\MaconomyDir\Database
StartOracleSQL <shortname> <password> DropTimestampTriggers.sql
c:\Logs\DropConstraints.sqlshort.log
```

### UNIX:

```
cd /data/maconomy/<oldapplication>/MaconomyDir/Database
StartOracleSQL <shortname> <password> DropTimestampTriggers.sql >
Logs/DropConstraints.<shortname>.log
```

**Warning!** Maconomy version 2.5.1 as well as 2.5.1 CU1 and CU2 contains incorrectly named timestamp triggers. To remove them, execute the same command as for dropping standard triggers, except with the following SQL file:

DropTimestampTriggersWithT.sql

**Note:** Executing this step on systems that do not have faulty triggers is harmless.

## 6.5 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).

**Note:** 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:

### Win:

```
cd \PUs\DeltekMaconomyUpgradeKit24GA\SQL\
StartOracleSQL.<oldapplication> <shortname> <password>
DeletePortalStandard.sql > c:\Logs\DeletePortalStandard.<shortname>.log
```

### UNIX:

```
cd /data/DeltekMaconomyUpgradeKit24GA/SQL
```

```
StartOracleSQL.<oldapplication> <shortname> <password>  
DeletePortalStandard.sql > Logs/DeletePortalStandard.<shortname>.log
```

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 (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.

**Note:** 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:

**Warning:** Do not run DeleteStandardLayouts when upgrading to Maconomy 2.4 or later as it will cause errors in the upgrade process.

**Win:**

```
cd \PUs\DeltekMaconomyUpgradeKit24GA\SQL\  
StartOracleSQL.<oldapplication> <shortname> <password>  
DeleteStandardLayouts.sql > c:\Logs\DeletePortalStandard.<shortname>.log
```

**UNIX:**

```
cd /data/DeltekMaconomyUpgradeKit24GA/SQL  
StartOracleSQL.<oldapplication> <shortname> <password>  
DeleteStandardLayouts1.sql > Logs/DeletePortalStandard.<shortname>.log
```

Examine the log file. Locate and correct errors.

**Note:** 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.

## 7 Update the Database Schema

*TIA Consultant*

### 7.1 Expand Relations

**Note:** For large databases see [Expand Script](#).

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

**Note:** You should use the expand script “expand.<oldschema>-<newschema>.sql” The schema version of the old application is shown in the “<OldApplicationPath>/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. This script may run for several hours depending on database size and the version span to cover.

### Expand Script

**Warning:** You cannot restart the expand script.

**To run the expand script:**

1. Enter the following commands:

**Win:**

```
cd c:\maconomy\<application>\MaconomyDir\Database
StartOracleSQL <shortname> <password> expand.<oldschema>--<newschema>.sql >
c:\Logs\Expand.<shortname>.log
```

**UNIX:**

```
cd /data/maconomy/<application>/MaconomyDir/Database
StartOracleSQL <shortname> <password> expand.<oldschema>--<newschema>.sql >
Logs/expand.<shortname>.log
```

2. After completion, check the “expand.<shortname>.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.
3. After, check the error log. Check the expand.<shortname>.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).

4. 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.

**Attention:** Compare the DEV\_ tables to their definitions in Maconomy version 2.5, 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 the target version, it is a customization that you must deal with on a case-by-case basis, working with the Deltek Services team.

## Postexpand Script

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

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

**Win:**

```
cd c:\maconomy\<application>\MaconomyDir\Database
StartOracleSQL <shortname> <password> PostExpand.sql >
c:\Logs\PostExpand.<shortname>.log
```

**UNIX:**

```
cd /data/maconomy/<application>/MaconomyDir/Database
StartOracleSQL <shortname> <password> PostExpand.sql >
Logs/postexpand.<shortname>.log
```

## 7.2 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 MConfigScriptlets under <tpudir>.

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

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

**Win:**

```
<MConfig> -r <tpudir>\DeleteUnusedColumns <application> <shortname> <password>
```

**UNIX:**

```
<MConfig> -r <tpudir>/DeleteUnusedColumns <application> <shortname> <password>
```

## 7.3 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>.

### 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:**

**Win:**

```
cd c:\maconomy\<application>\MaconomyDir\Database
StartOracleSQL <shortname> <password> DHViewMake.sql >
c:\Logs\DHViewMake.<shortname>.log
```

**UNIX:**

```
cd /data/maconomy/<application>/MaconomyDir/Database
StartOracleSQL <shortname> <password> DHViewMake.sql >
Logs/DHViewMake.<shortname>.log
```

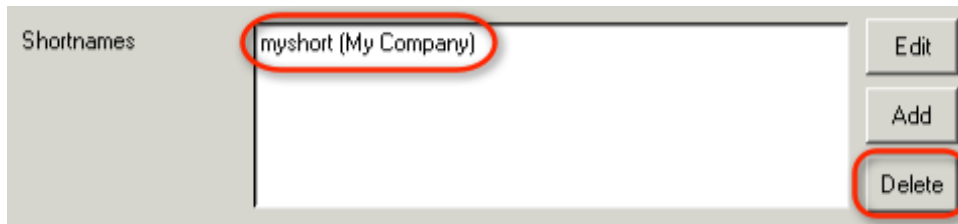


## Detach shortname from the Old Application

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.

## 8 Update Business Logic and Components

*TIA Consultant*

### 8.1 [Version] Upgrade

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).

#### Attaching the Shortname

The following commands (in this step and the rest of this document) assume that the application has the shortname attached (so the MaconomyServer.<application> command is available).

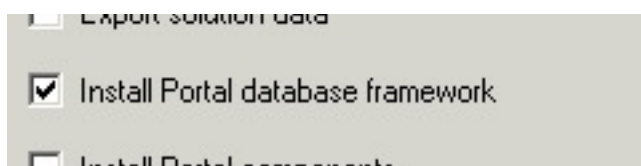
**Attention:** When attaching the shortname, be sure to install the new Portal database. Otherwise, the step in which you export part of the database for conversion with ModifyExportFile will fail.

**Note:** 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 in the new application 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:

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



**Note:** 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 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.

2. In MConfig go to the Application Instance window and select the SPU and the Solution.

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

## 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:**

### Win:

1. `cd c:\maconomy\<application>\MaconomyDir\Database`
2. `StartOracleSQL <shortname> <password> ViewMake.sql > c:\Logs\ViewMake.<shortname>.log`
3. `StartOracleSQL <shortname> <password> ViewInit.sql > c:\Logs\ViewInit.<shortname>.log`
4. `MaconomyServer.<application> -f -3 -S<shortname>`
5. `Recreate3PStuff <shortname> <password>`
6. `MaconomyServer.<application> -S<shortname> --ConvertAnalyzerFiles`

### UNIX:

1. `cd /data/maconomy/<application>/MaconomyDir/Database`
2. `StartOracleSQL <shortname> <password> ViewMake.sql Logs/ViewMake.<shortname>.log`
3. `StartOracleSQL <shortname> <password> ViewInit.sql Logs/ViewInit.<shortname>.log`
4. `MaconomyServer.<application> -f -3 -S<shortname>`
5. `Recreate3PStuff <shortname> <password>`
6. `MaconomyServer.<application> -S<shortname> --ConvertAnalyzerFiles`

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

## Errors

In the log file for the SQL scripts there may be some Oracle errors related to DROP VIEW statements, such as:

```
ERROR at line 1:
```

```
ORA-00942: table or view does not exist
```

This is normal, and no action is required.

## 8.2 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.

### Update Common Relations

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

1. `UpdateFieldsAndRelations.<application> <shortname> <password> -NC`

No errors may occur.

**Warning:** 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 under any circumstances as the step is not repeatable and if killed mid execution will require the upgrade to be restarted or at the very least the Database would have to be restored from a backup prior to the execution of this step if such a backup exists.**

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.

### Create Database Objects

To create indexes and other database objects, execute the following commands:

```
cd c:\maconomy\bin
```

```
MaconomyServer.<application>.cmd -S<shortname> --IndexScriptFile "idxm.secondary.ora.txt"
```

```
cd c:\maconomy\tmp
```

```
StartOracleSQL <shortname> <password> createScript.txt > c:\Logs\createScript.log
```

Once executed, check the log. There may be errors concerning objects that cannot be created because they already exist. You can ignore those errors. Other kinds of errors should be investigated.

**Notes:** There is a post upgrade duplicate cleanup step that will clean up duplicated indexes. On large databases, the creation of Database Objects typically runs for a long time.

**Warning:** Executing idxm.secondary.msql.txt via MaconomyServer may produce errors like these:

Error: Index FINANCEENTRY01 found in hints files cannot be found in the database

Error: Index INVOICETRANSFERJ01 found in hints files cannot be found in the database

These errors are meant to inform you of missing database objects. Note that the resulting createScript.txt file will have the required statements to create the missing objects.

## 8.3 Reinstall Standard Layouts for all Platforms

To reinstall standard layouts:

1. Type the following commands:

**Win:**

```
cd c:\maconomy\<application>\MaconomyDir\MSLScripts
MaconomyServer.<application> -S<shortname> -xRemove_Old_Layouts
```

**UNIX:**

```
cd /data/maconomy/<application>/MaconomyDir/MSLScripts
MaconomyServer.<application> -S<shortname> -xRemove_Old_Layouts
```

2. If upgrading to Maconomy 2.3 or higher from Maconomy 2.2 or lower, the following SQL statements must be executed against the database before moving forward with the upgrade. If upgrading from 2.5, 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;
```

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

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.

3. Type the following command to reinstall standard layouts:

```
MaconomyServer.<application> -S<shortname> -UW -UP
```

**Warning:** 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 Release Notes for details.

## 8.4 Convert the Portal Database

To upgrade the Portal database, complete the following sets of steps (replace the base and target application main version numbers accordingly):

```
update Component
set Callbackscript = replace(Callbackscript, 'W_12_0', 'W_20_0')
where Callbackscript like '%W_12_0%';
update Maconomywindow
set SpecFileName = replace(SpecFileName, 'W_12_0', 'W_20_0')
where SpecFileName like '%W_12_0%';
update MScriptComponent
set Mscriptfilename = replace(Mscriptfilename, 'W_12_0', 'W_20_0')
where Mscriptfilename like '%W_12_0%';
update Pdmcomponent
set Configurationfile = replace(Configurationfile, 'W_12_0', 'W_20_0')
where Configurationfile like '%W_12_0%';
update Pdmcomponent
set Tabfunction = replace(Tabfunction, 'W_12_0', 'W_20_0')
where Tabfunction like '%W_12_0%';
```

**Note:** 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.

## 8.5 Convert Data to the New Version

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.

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.

### Win:

```
cd c:\maconomy\<application>\MaconomyDir\MSLScripts
MaconomyServer.<application> -xCheckEnterpriseLanguage -S<shortname>
```

### UNIX:

```
cd /data/maconomy/<application>/MaconomyDir/MSLScripts
MaconomyServer.<application> -xCheckEnterpriseLanguage -S<shortname>
```

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).  
Do not change this unless explicitly directed to do so by Engineering.

**Prior to running data conversion, the following constraint must be created either by using SQLPLUS or SQL Developer:**

```
ALTER TABLE DATABASERELATION ADD CONSTRAINT IKDATABASERELATION
UNIQUE(INSTANCEKEY);
```

Start Convert\_Data by using the following commands.

**Win:**

```
cd c:\maconomy\<application>\MaconomyDir\MSLScripts
MaconomyServer.<application> -xConvert_Data_From_9.0 -S<shortname>
```

**UNIX:**

```
cd /data/maconomy/<application>/MaconomyDir/MSLScripts
MaconomyServer.<application> -xConvert_Data_From_9.0 -S<shortname>
```

**To check the data conversion status manually:**

1. Use the following command (with sqlplus):

```
select DATAVERSIONSTATUS from SYSTEMINFORMATION;
```

2. 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.

3. If you are changing the setting for UserLanguage, remember to set it back after you run Convert\_Data.

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

**To generate a log file on a UNIX system:**

In the ConvertData program, enter the following command:

```
MaconomyServer.<application> -xConvert_Data_from_9.0 -S<shortname> <
Import.Data >&! ../Logs/ConvertData.<shortname>.log &
```

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

## 8.6 Create TimeStamp Triggers

At this stage, timestamp triggers must be recreated if the application uses them.

**To recreate timestamp triggers:**

1. Using the the shortname window in MConfig, select the **Enable Transaction Timestamps** checkbox:

Business Performance Management

☐ Enable Reporting

☐ Create performance views Views

☐ Enable Business Intelligence

☒ Enable Transaction Timestamps

CALENDARAYPV  
JOBENTRYJOBINVOICELINEPV  
ACCOUNTPV



## 9 Validate Custom Components

*TIA Consultant*

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.

### 9.1 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.

### Screen Layout Validation on the Server

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

```
MaconomyServer.<application> -S<shortname> -UVW
```

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

### 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 enter the following commands:**

1. MaconomyServer.<application> -S<shortname> -UEP

**UNIX:**

```
java -jar MPL3to4MigrationTool.jar -dir  
/data/maconomy/<application>/MaconomyDir/ExportedLayouts/
```

**Win:**

```
java -jar MPL3to4MigrationTool.jar -dir  
c:\maconomy\<application>\MaconomyDir\ExportedLayouts  
2. MaconomyServer.<application> -S<shortname> -UIP  
3. MaconomyServer.<application> -S<shortname> -UVP
```

## 9.2 Create Constraints

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

### Instance Keys

**All Platforms**

```
MaconomyServer.<application> -UC -S<shortname>
```

### Other Constraints

Oracle-specific constraints:

```
cd c:\maconomy\bin  
MaconomyServer.<application>.cmd -S<shortname> --IndexScriptFile "consmCreate.ora.txt"  
  
cd c:\maconomy\tmp  
StartOracleSQL <shortname> <password> creCons.txt > C:\logs\creCons.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.

**Warning:** Create a support case with Deltek if any errors other than "constraint already exists" occurred. Ignoring constraint creation errors can lead to database inconsistencies.

## 9.3 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 that support 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.

## 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:

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.

**Note:** Be sure to force the import of Portal components and upgrade of the database schema for the shortname in question, by selecting ‘Unconditionally’.

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.

## 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:

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.

**Note:** 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.

## 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:

**Win:**

```
cd C:\maconomy\WebServers\<application>\cgi-bin\Maconomy
MaconomyPortal.<shortname>.en_US.exe -e -q
"filename=solution_module_comps.xml&username=Administrator&password=123456&importtype=components" Framework/import/importportal.ms
```

**Note:** 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:\maconomy\WebServers\<application>\MaconomyPortal'. The files are located in 'C:\maconomy\<application>\Portal\Installation'

Check that the 'MaconomyPortal.<shortname>.en\_US.exe' file exists and that you are using the correct one for your language settings. Ex.: for Danish, you should use 'MaconomyPortal.<shortname>.da\_DK.exe' instead.

The files to be imported are located in 'c:\maconomy\<application>\Portal\Installation' on Windows. Component definitions and default role setup and menus are available.

Import files for industry accelerators are located in 'c:\maconomy\<application>\IA.<Industry Accelerator>\Portal\Installation' on Windows.

## 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:\maconomy\WebServers\<application>\'.

**Example: Export the component definition**

```
cd C:\maconomy\WebServers\<application>\cgi-bin\Maconomy
MaconomyPortal.<shortname>.en_US.exe -e -q
"filename=Maconomy2_1.portal_comps.xml&username=Administrator&password=123456&exporttype=components" Framework/export/exportportal.ms
```

The export file is located in:

```
C:\maconomy\WebServers\<application>\MaconomyPortal\Maconomy2_1.portal_comps.xml
```

**Example: Import and validate the components**

```
MaconomyPortal.<shortname>.W.exe -e -q
"filename=Maconomy2_1.portal_comps.xml&username=Administrator&password=123456&importtype=components" Framework/import/importportal.msselect
```

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

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

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

## 10 Post Upgrade Tasks

*TIA Consultant*

This section covers various cleanup tasks and items to check once the main upgrade process is complete and before allowing users onto the new Maconomy application.

### 10.1 Recreate BPM Objects

**Note:** 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.

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

### 10.2 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.

### 10.3 Logging of Database Requests or Updates

*Application Consultant*

If “Advanced Logging” is used (add-on 32 and/or 33 is active) in the original Maconomy application, you must migrate it to the new Maconomy application after the upgrade.

**To migrate Advanced Logging to the new application:**

1. Check that the following names of tables and columns in the specification files are still valid after the upgrade:  
.../MaconomyDir/Definitions/RequestLog.cnf  
and  
.../MaconomyDir/Definitions/UpdateLog.cnf
2. Correct the names if necessary.
3. If a specification file is modified it must be stamped by HQ-Distribution ([HQ-Distribution@deltek.com](mailto:HQ-Distribution@deltek.com)).

## 10.4 Disable Maintenance Mode

*TIA Consultant*

At this point, the Maconomy application should be put out of maintenance mode using MConfig. Putting the system out of maintenance mode will re-enable background tasks.

**To disable Maintenance Mode:**

- 1 Go to the Application window.
- 2 Deselect the **Maintenance Mode** field.

## 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](#)

### Check Fatal Error

If you get a Check Fatal error, call Deltek Support.



---

## About Deltek

Better software means better projects. Deltek is the leading global provider of enterprise software and information solutions for project-based businesses. More than 23,000 organizations and millions of users in over 80 countries around the world rely on Deltek for superior levels of project intelligence, management and collaboration. Our industry-focused expertise powers project success by helping firms achieve performance that maximizes productivity and revenue. [www.deltek.com](http://www.deltek.com)