Deltek

Deltek Maconomy® 2.5

Upgrade Guide for Microsoft SQL

November 22, 2019

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 edition published November 2019.

Contents

1 Introduction

This document describes how to upgrade from 2.1 or later to Maconomy 2.5.x tools version 21.

> **Note:** For upgrades older than 2.1, a database conversion to Unicode is required. See Unicode Conversion for details.

This document describes each step needed for the upgrade. All examples are based on one company, and this company has the shortname **myshort**. The preferred version database is Microsoft SQL Server 2008 R2. This document does not describe how to upgrade database versions.

You must execute SQL scripts using the commands that are installed by MConfig. For Microsoft SQL Server the command is StartSQL.cmd. When updating a Microsoft SQL Server database, make sure that you are using the scripts that have the string “ISQL” in their names.

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

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

Maconomy Version	Tools Version
X+	11.0
X1	12.0
2.0	15.0
2.1	16.0, SP0,SP1
2.1.1	16.0 SP2
2.1.4	16.0 SP4
2.2	17
2.3	19
2.4	20
2.5	21

1.1 Main Upgrade Phases

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

- Review prerequisites and perform critical actions 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

2 Prerequisites

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

- Obtain the upgrade kit
- Prepare if this is a SQL Server 3-tier installation
- Review hardware and resources
- Check upgrade preconditions
- Perform any necessary critical actions prior to upgrade

2.1 Obtain Upgrade Kit 2.5.x

You must obtain the Upgrade Kit for 2.5.x as well as the TPU and the APU. This document is included in the Upgrade Kit.

> **Note:** Always look for the latest and updated version of the Upgrade Kit on the Deltek Maconomy Download Site before beginning an upgrade.

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

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

2.2 SQL Server 3-Tier Setup

Due to license issues, it is common to use a three-tier installation to keep license cost to a minimum. A 3-tier setup requires installation of Management Studio on the Application server.

Note: The instructions below are for Management Studio for SQLServer 2014. While there are different versions of SQLServer with more or less functionality, all versions ensure that all SQL jobs can be executed from the Application server, and the Management Studio 2014 Express package is free.

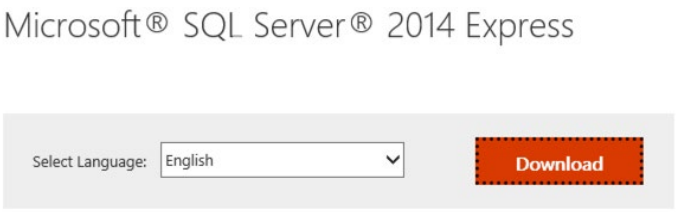
Management Studio Installation

To download Management Studio:

1. Navigate to the Management Studio to download. For example, download Management Studio for SQLServer 2014:

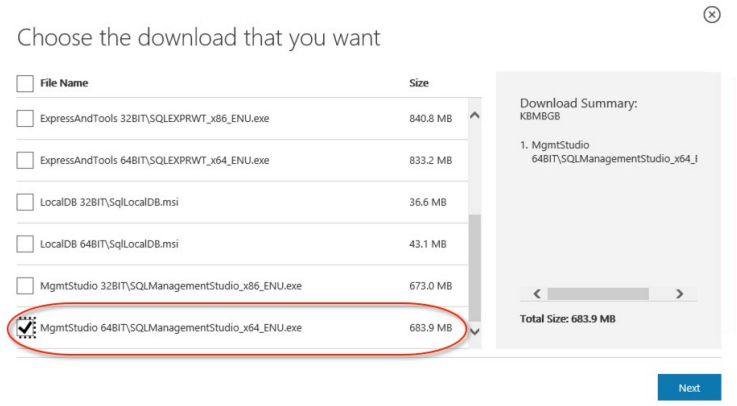
   https://www.microsoft.com/en-in/download/details.aspx?id=42299

2. Click **Download**.



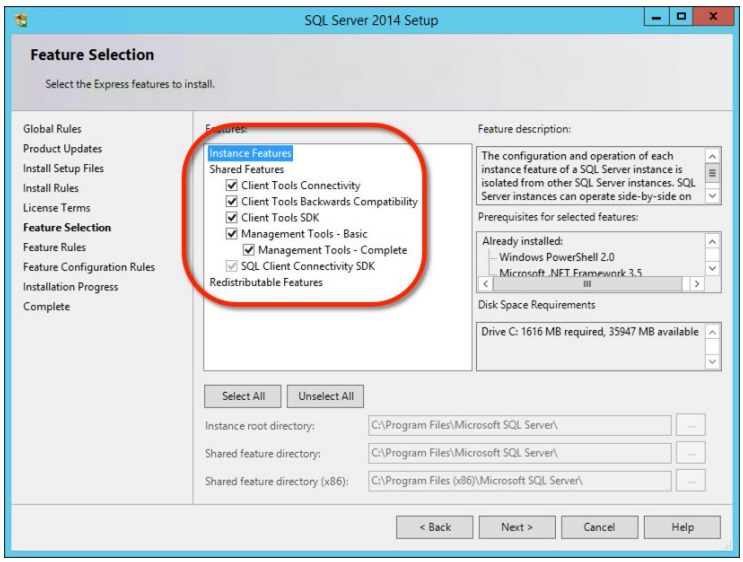
A list of download options displays.

3. Select the version to install (usually the 64-bit version) and click **Next**.



A list of features options displays.

4. Select all options under Shared Features.



Management Studio Workaround

If for some reason you are unable to install Management Studio on the Application server, as a workaround, copy the following folders to the database server:

```
<MaconomyHomeDir>\bin
<ApplicationHomeDir>\MaconomyDir\Database
```

Example:

```
E:\maconomy\bin
```

```
E:\maconomy\w_20_0\MaconomyDir\Database
```

In the Maconomy\bin directory, add the following option to StartSQLServer1:

```
<Application>.cmd: -S <DatabaseHostName>\<InstanceName>
```

If instance Name is the default, it can be ignored.

Example:

```
"c:\program files\microsoft sql server\client
sdk\odbc\110\tools\binn\sqlcmd.exe"  -S <Hostnname of database server> -U %1 -
P %2 -i "%3" -o "%4" -fi:1252
```

2.3 Review Hardware and Resources

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 recommended, but 12 cores or more are minimum for larger databases.

2.4 Check Upgrade Preconditions

Attention: 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. See the upgrade precondition section for details.

3 Critical Actions Prior to Upgrade

3.1 Approval Hierarchy for Job Quotes

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.

3.2 Enhanced Vendor Invoice Reallocation

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.

3.3 Invoicing: FinanceEntry of the A/R Control Account

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.

3.4 Increasing System Numbers

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

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

  Note: The data conversion is moderate and adds about 15 minutes for each 10 million rows in the database. See Estimates for Upgrade Time for details.
- 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

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.

Estimates for Upgrade Time

Test Database

The estimate is done for a database with 30 million rows in the 5 largest tables.

Compare another database to the test database with the following SQL statement:

```
SELECT
   (SELECT COUNT(*) FROM FinanceEntry)
+ (SELECT COUNT(*) FROM JobEntry)
+ (SELECT COUNT(*) FROM DailyTimeSheetLine)
+ (SELECT COUNT(*) FROM TimeSheetLine)
+ (SELECT COUNT(*) FROM GeneralJournal)
;
```

Test Hardware

```
System 1
      (developer laptop used as a upper bound on how slow an upgrade
      can be on unsuitable hardware)

      OS Name:                    Microsoft Windows 7 Enterprise

      System Model:               LENOVO T560
```

Processor(s): Processor Intel(R) Core(TM) i7-6600U CPU @ 2.60GHz, 2701 Mhz, 2 Core(s), 4 Logical Processor(s) Total Physical Memory: 16. GB
System 2 (performance server used as it matches the best hardware found at customers) OS Name: Microsoft Windows Server 2012 R2 Standard System Model: ProLiant DL380 Gen9 Processor(s): 2 Processor(s) Installed. [01]: Processor Intel(R) Xeon(R) CPU E5-2643 v3 @ 3.40GHz, 3396 Mhz, 6 Core(s), 6 Logical Processor(s) [02]: Processor Intel(R) Xeon(R) CPU E5-2643 v3 @ 3.40GHz, 3396 Mhz, 6 Core(s), 6 Logical Processor(s) Total Physical Memory: 130 GB Database: Oracle Database 12c Enterprise Edition Release 12.1.0.2.0 - 64bit Production

Test Results Example

	System 2	System 1
Expand Script	17 minutes	10 hours 40 minutes
Full Data Conversion	N/A	32 hours

System 1 is a laptop and runs slowly. Even if you have a 5-year-old server hardware the computation power and IO is better than on System 1. Therefore, you should not experience more than 10 hours increase in data conversion due to the system numbers. A full data conversion is run to show that it can take up to 1/3 of the total data conversion time. As the system number data conversion is sensitive to OI performance and also subject to parallel computations, it is hardware-dependent compared to the rest of the data conversion which is more sequential and thus you should not try to extrapolate the missing Full Data Conversion on System 2 as this will run more than an hour.

The 17 minutes for the Expand Script on system 2 shows that you can expect to spend about 5 minutes for each 10 million rows in the database on similar hardware. However, you are not expected to have hardware optimized especially for high IO, as it is the case for System 2 and a slowdown factor of three is expected. Generally, you can expect to process 10 million rows in 15 minutes. If you have larger databases with hundreds of millions of rows, however, you should invest in high-preforming hardware and spend time on optimization.

The resulting 15 minutes per 10 million rows is regarded as fast and additional upgrade time due to update of system numbers is a minor concern.

Places to Run Data Conversion

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

1. Production system
2. Data warehouse

Update Custom BPM Universes

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 customer universes and customer reports:

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 meta data 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.

6.1 My Approvals / Approval Center Name Change

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

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

6.2 Instance Keys

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

6.3 Use MyxL Tool to Convert Syntax Files

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

MyxL tool facilitates the conversion of Maconomy's specification files, including MDML, MWSL, MMSL, MNSL, and MCSL.

To convert specification files, you must:

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

Before You Begin

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

If needed, install a correct 32-bit JDK.

Convert Specification Files

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.

3. 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.
   - convertdirectory — *Mandatory*. The path to the file or directory to convert. If no other output method is specified, all files in this directory (or this file) are converted in place.
   - writeconversion — *Optional*. If this parameter is not set, no converted output is generated. Only a summary is written to the console and a summary report is generated. It is not set by default to prevent conversion errors. Must be explicitly set to write the conversion changes.
   - outputdirectory — *Optional*. A path to a directory to write the converted files to, if writeconversion was set. If this parameter is set the original files in the convertdirectory are not altered.
   - consolepreview — *Optional*. If this parameter is set the result of the conversion is written to the console and a summary report is generated, if writeconversion was set.
   - convertreadonly — *Optional*. If this parameter is set read-only files are converted.
   - debuginfo — *Optional*. If this parameter is set debug information such as stack trace is written to the error section of the summary report.
   - logconfigpath — *Optional*. You can specify a path to a log configuration file here. As default the logback file provided with the Converter is used.
4. Check to verify the updated syntax of the specification language. This includes checking to ensure that all the breaking changes were fixed and no unnecessary conversions were made (meaning that the tool does exactly what it is supposed to do and no more).

   The output method precedence is:

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

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.

5. Submit the changes back to GitHub.

Note:

Ensure Correct Conversion

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

If Errors Occur

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

Example

The following example shows a change for 2.4 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 *Deltek Maconomy Language Quick Reference MDML* for the 2.4 release for details.

Example (Windows)

1. Insert the parameters into the Convert-20.0.101.0.ini file:

```
--convertdirectory
C:\maconomy\w_20_0\CustomizationDir
--outputdirectory
C:\test\outputdir
--convertreadonly
--writeconversion
```

2. Insert them when running from the command line:

```
>Convert-20.0.101.0 --convertdirectory "C:\maconomy\w_20_0\CustomizationDir" --
outputdirectory "C:\test\outputdir" --convertreadonly --writeconversion
```

6.4 Upgrade from Early Versions

If you upgrade to 2.5.x from any version prior to Maconomy 2.1.1, you must take a special action and run an MConfig scriptlet to process the language upgrades in this release.

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

To run the MConfig scriptlet, follow these steps:

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

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

Note:

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

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

6.5 Upgrade from MAS or MCS

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

To upgrade from a legacy system and customize a standard system:

1. Upgrade the system using the standard procedure. However, do not install the SPU.
2. Create an Extender project for the upgraded system.
3. Extract the files from the package and copy them to the Extender project.
4. Deploy the Extender project.
5. Run the batch file w_20_0\ExtenderDeployFolder\ImportData\CopyToCustomInstallation.bat.
6. In MConfig, for the field **Load Shortname Data**, select **Custom Data (Import only)**.
7. Click **Install**.

This installs a system with a portal, prints, and screen layouts in the custom layer matching the former solution.

Note:

Further Customizations

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

Removing Layouts

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

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

MaconomyServer_20_0 -Smyshort -xRemoveSolutionWindowLayouts

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

6.6 Remove BPM Database Objects

When upgrading to Maconomy 2.5.x, 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.

6.7 Create Constraints

Ensure that constraints exist on TIMESHEETHEADER and DAILYTIMESHEETHEADER before upgrade. If they do not already exist, you must create them using the statements below as custom constraints:

```
ALTER TABLE TIMESHEETHEADER ADD CONSTRAINT TIMESHEETHEADER1001 UNIQUE
(EMPLOYEENUMBER,PERIODSTART);

ALTER TABLE DAILYTIMESHEETHEADER ADD CONSTRAINT
DAILYTIMESHEETHEADER1001 UNIQUE (EMPLOYEENUMBER,THEDATE);
```

Note: These constraints may exist on the source system already, but with a different name. Ensure the constraints are added to your source system before doing the upgrade.

6.8 Application Pre-Upgrade

(See Appendix)

7 Installation

This section describes how to install the new TPU and the 2.5.x application to which you are upgrading Maconomy.

For the purpose of this guide we will assume two applications exist. The base application, ‘w_15_0.b’ and a target application, ‘w_20_0.t’.

We also assume that the main drive is ‘C’ for Windows environments.

In all of the examples used in this section, the shortname is **myshort**, and the password is **myshort**.

7.1 Install the New Tools (TPU)

You must install a new TPU when upgrading—the most recent one that belongs to the W 20.0 application.

Copy MConfig, TPU (Tools version 20), and the 2.5.x APU to the PU (PackingUnit) directory on the server. You use MConfig to specify the location of that directory by going to ‘Global Settings’ and defining the directory in ‘Packing Units Directory’.

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

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

7.2 Install the New Application (APU)

Use MConfig to install the 2.5.x 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.

Warning: Be aware that enterprise language has been changed from w_16_0 SP2. The previous languages, such as US, W_MCS, and so on, are no longer valid. Check the release notes here. Access to home.deltek.com is required.

To check the enterprise language on an existing database:

- Type the following command on a query window for the shortname in, for example, SQL Plus Management Studio:

```
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.

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

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.

8 Check Upgrade Preconditions

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

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.

8.1 Check Preconditions for Upgrade

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

You must select the appropriate script, for example:

- SQL SERVER: ISQLUpgrPrecond.w120sv12-w200.sql

The old schema version must also be taken into consideration when you upgrade the application from W 16.0. You can find the schema version in:

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

For example, when upgrading from W 12.0 schema version 12 on Oracle, you must use the ISQLUpgrPrecond.w120sv12-w200.sql script.

To check preconditions for the upgrade on SQL Server:

1. **(navigate to target application folder)**
   a. **cd c:\maconomy\w_20_0.t\MaconomyDir\Database**
   b. **c:\maconomy\bin\StartSQLServer2.w_12_0.b myshort myshort ISQLUpgrPrecond.w120sv12-w200.sql c:\Logs\ISQLUpgrdPrecond.myshort.log**

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

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

8.2 Integration with Talent Management

If you are upgrading from Maconomy 2.2.2 to 2.5.x or later, and you integrate with Talent Management, you must delete old monitors and create background tasks.

To delete standard monitors and create background tasks:

1. Use, for example, SQL Server Management Studio, to query on the Maconomy database to search for standard monitors:

```
select * from MONITOR where MONITORNUMBER = 'SynchronizeHRSmartUsers' or
MONITORNUMBER = 'ImportDTMUsers';
```

2. In the Monitors Single Dialog Workspace, locate and delete these two standard monitors: ImportDTMUsers and SynchronizeHRSmartUsers
3. See *Deltek Maconomy Integration With Talent Management Guide* for steps to create background tasks.

8.3 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 8.15 or later. 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 older Maconomy versions.

Note: If you have extensions, you must re-install 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.

To manually restart scheduled background tasks:

1. On the command line enter:

```
system.maintenance = not batch.active
```

9 Delete Old Components

9.1 Dump Changes to the Standard Index Set

Note: For large databases see Dump Changes to Standard Indexes.

Delete Old Constraints – only for 19 or newer

9.1.1 Delete Old Constraints

To delete old constraints:

1. Drop instance key constraints temporarily by typing the following:

   ```
   c:\maconomy\bin\MaconomyServer.w_12_0.b -UD -Smyshort
   ```

2. Drop the old constraints by typing the following:

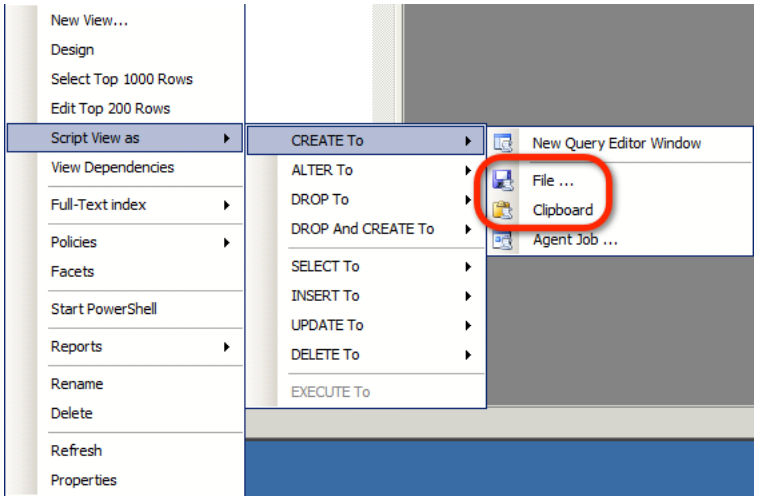
   ```
   cd c:\Maconomy\w_12_0.b\MaconomyDir\Database

   StartSQLServer2 myshort myshort ISQLDropConstraints.sql
   c:\Logs\ISQLDropConstraints.myshort.log
   ```

6.1.2 Dump Materialized View Definitions

Open Management Studio and select the materialized view. Standard materialized view names end with MV or MV_A, MV_B, and so on. (Examples are JOBENTRYJOBINVOICELINEMV, JOBENTRYJOBINVOICELINEMV_A, and JOBENTRYJOBINVOICELINEMV_B) Materialized views are called Performance Views in MConfig. If there are customized views, the easiest way to identify them is to delete all standard views and dump the remaining ones.

To dump materialized views:

1. Right-click on the view name and select either **File** or **Clipboard**.



2. Save the view.
3. Delete the view.

4. Repeat these steps for the remaining materialized views.

9.2 Delete Old Indexes

> **Note:** For large databases see Drop Indexes.

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

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

To use MConfig to delete the old indexes:

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

9.3 Delete Views

Three different types of views are in use by Maconomy:

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

You must remove the first two types before you can perform an upgrade. When updating from one application to another, the scripts must be run to ensure proper functioning of the new application version. The programs are located in the Maconomy bin directory in the TPU that is installed for the application, for example, the C:\Maconomy\tpu.NTx86.12_0.p29.dir\bin folder (and not the C:\Maconomy\bin folder, which is the global TPU folder).

The DeleteAnalyzerViews is an Mconfig scriptlet and requires MConfig to be run. MConfig 8.0 or later is required to run the scriptlet. MConfig must be started from the command line. Specify the path to MConfig, then –r (or MConfig will think it is an MConfig installation script), then the path to the script or just the name – MConfig searches for the script in the TPU that is installed for the application (that is, not the global TPU). These scripts are located in the MConfigScriptlets folder in the TPU folder, for example, C:\maconomy\tpu.NTx86.12_0.p29.dir\MConfigScriptlets. Lastly, the arguments (shortname and password) to the script must be given.

Commands to delete the views for the user:

cd C:\maconomy\tpu.NTx86.12_0.p29.dir\MConfigScriptlets

```
DeleteDHViews myshort myshort MSQL

Delete3PStuff myshort myshort MSQL

c:\PUs\MConfig-8.15.exe –r DeleteAnalyzerViews w_12_0.b myshort myshort
```

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

Done!

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

Something went wrong (or nothing to delete)

It may be OK if the customer was not using the functionality. Check with the old access rights.

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.

9.4 Drop Timestamp Triggers

StartSQLServer2 **`myshort myshort`** ISQLDropTimestampTriggers.sql c:\Logs\ISQLDropTimestampTriggers.log

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.

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.

a. **`cd c:\PUs\DeltekMaconomyUpgradeKit24GA\SQL\`**

b. StartSQLServer2 **`myshort myshort ISQLDeletePortalStandard.sql c:\Logs\ISQLDeletePortalStandard.myshort.log`**

c. StartSQLServer2 **`myshort myshort ISQLDeleteStandardLayouts.sql c:\Logs\ISQLDeleteStandardLayouts.myshort.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.

10 Update the Database Schema

10.1 Expand Relations

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:** Assume that the source application is W 12.0, and a service pack with schema changes has been applied; you should use the expand script “expand.w120svX-w20svX.sql” (similarly for MSQL), where X is the current schema version number. The schema version is shown in the “<path to W 12.0 application>/MaconomyDir/Installation/Version.info file. Always double-check the expand script filename to see if you are using the correct one.
>
> You will always get error messages when running expand scripts, but it is important to check that you only receive the expected messages. Thus, it is important to enable logging **before** you run the expand script. On large databases this script may run for some time.

> **Warning:** You cannot restart the expand script. If there is an error in the expand script, it will stop. It is therefore VERY important to check if the script has processed all rows. In case of an error the script will stop. After fixing the error, remove the lines above the error and start the script again.

To run the expand script, enter the following commands:

1. `cd c:\Maconomy\w_20_0.t\MaconomyDir\Database`
2. `StartSQLServer2 myshort myshort ISQLexpand.w120sv12-w200sv2.sql c:\Logs\ISQLExpand.myshort.log`

After completion, check the expand.myshort.log file using a text editor. No errors are accepted.

Make sure that there are no error messages.

Attention: You cannot restart the expand script. If there is an error in the expand script, it will stop. It is therefore VERY important to check if the script has processed all rows. In case of an error the script will stop. After fixing the error, remove the lines above the error and start the script again.

If the existing tables do **not** match the tables from the expand script of Maconomy 2.5.x, it is a customization that you must deal with on a case-by-case basis, working with the Deltek Services team.

When upgrading to Maconomy version 2.4 a post-expand script must be executed before moving forward.

To run the post-expand script, enter the following commands:

1. `cd c:\Maconomy\w_20_0.t\MaconomyDir\Database`
2. StartSQLServer2 `myshort myshort ISQLPostExpand.sql c:\Logs\ISQLPostExpand.myshort.log`

Warning: You cannot restart the expand script. If there is an error in the expand script, it will stop. It is therefore VERY important to check if the script has processed all rows. In case of an error the script will stop. After fixing the error, remove the lines above the error and start the script again.

(Unicode appendix)

10.2 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>.

10.2.1 Create DHViews

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

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

1. `cd c:\Maconomy\w_20_0.t\MaconomyDir\Database`
2. StartSQLServer2 `myshort myshort DHISQLViewMake.sql c:\Logs\ISQLDHViewMake.myshort.log`

10.2.2 Detach shortname from the Old Application

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

To detach the shortname from the old application:

1. Open the old environment using MConfig.

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

11 Update Business Logic and Components

11.1 W_20_0 Upgrade

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

11.1.1 Attaching the Shortname

The following commands (in this step and the rest of this document) assume that the application w_20_0 has the shortname **myshort** attached (so the MaconomyServer.w_20_0 command is available).

During the attach process, you may receive an MConfig warning message claiming that Binarydocument, Textdocument and a list of Portal database tables are not as expected in the database. As long as the warning only contains the mentioned tables, it can be ignored. If there are any other tables in the warning message, please investigate before proceeding with the upgrade.

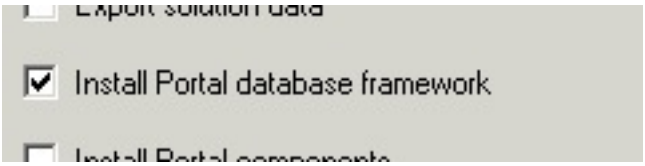
Attention: When attaching the shortname, be sure to install the new Portal database. Otherwise, the step in which you export part of the database for conversion with ModifyExportFile will fail.

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 on the new 2.4 system before new installation numbers are installed.

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

To install the new Portal database and attach the shortname:

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



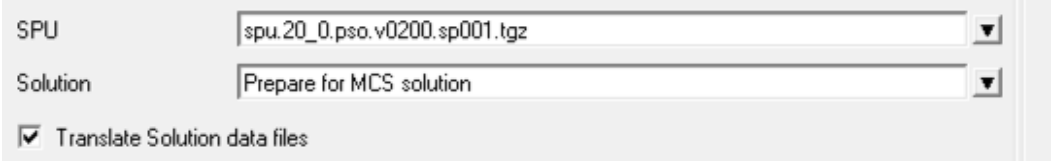
Note: If the shortname is not detached from the old application now, SQL scripts that run using StartSQLServer.cmd, and so on, may fail, so detach the shortname from the old application now.

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.

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

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

   The following figure shows an example where MCS is selected as solution.



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

11.1.2 Create Other Views

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

You must perform the following commands.

To create other views, enter the following commands:

1. `cd c:\Maconomy\w_20_0.t\MaconomyDir\Database`
2. StartSQLServer2 `myshort myshort ISQLViewMake.sql  c:\Logs\ISQLViewMake.myshort.log`
3. StartSQLServer2 `myshort myshort ISQLViewInit.sql  c:\Logs\ISQLViewInit.myshort.log`
4. `MaconomyServer.w_20_0.t -f -3 -Smyshort`
5. `Recreate3PStuff myshort myshort MSQL`
6. `MaconomyServer.w_20_0.t -Smyshort --ConvertAnalyzerFiles`

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

11.1.3 Errors

In the log file for the SQL scripts, errors like the following occur, which is normal.

```
Cannot drop the view '*', because it does not exist in the system catalog.
```

Check the log file for errors related to CREATE VIEW statements. Errors are not allowed when these statements are processed.

11.2 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.

11.2.1 Update Common Relations

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

1. `UpdateFieldsAndRelations.w_20_0 myshort myshort -NC`

No errors may occur.

> **Note:** 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.**

1. To test for a successful completion, enter the following command in an interactive SQL session or in SQL Server Management Studio:

```
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.

11.2.2 Field Length Verification

When using SQL Server you must run ISQLEnsureTypes.sql first to make sure that indexes created in this step do not get too long.

To ensure that indexes do not become too long:

1. Enter the following commands:

   ```
   cd c:\Maconomy\w_20_0.t\MaconomyDir\Database

   StartSQLServer2 myshort myshort ISQLEnsureTypes.sql
   c:\Logs\ISQLEnsureTypes.myshort.log
   ```

2. Check the log file for errors before continuing with the next steps. Note, however, that the following error messages can be ignored:

   ```
   The total row size (20322) for table '*' exceeds the maximum number of bytes per
   row (8060). Rows that exceed the maximum number of bytes will not be added.
   ```

   and

   ```
   Warning: The table '*' has been created but its maximum row size (19497) exceeds
   the maximum number of bytes per row (8060). INSERT or UPDATE of a row in this
   table will fail if the resulting row length exceeds 8060 bytes.
   ```

11.2.3 Create Indexes

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

Use MConfig to create indexes by completing the following steps:

- In the Main window, select the new application and click **Edit**.
- In the Application window, for each shortname do the following:
  - Select the shortname in the **Shortnames** list and click **Edit**.
  - In the Shortname window, click **Index management**.

- Select the **Create standard indexes** check box. If the index comparison revealed local changes to the standard set of indexes, also select the **Create additional indexes** check box and select the comparison file IndexCompareWith<shortname>.<old application>.idx in the **Index spec. files** list box. (There may be comparison files for different shortnames listed; be sure to select the right one).

- Click **OK** to return the Main window. Continue with **Next**, and so on, as for any other MConfig task (if in doubt, see the MConfig manual).

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

11.2.4 Create TimeStamp Triggers

1. Type the following commands:

   StartSQLServer2.cmd **myshort myshort**ISQLCreateTimestampTriggers.sql c:\Logs\ISQLCreateTimestampTriggers.log

11.3 Reinstall Standard Layouts for all Platforms

To reinstall standard layouts:

2. Type the following commands:

   ```
   cd c:\Maconomy\w_20_0.t\MaconomyDir\MSLScripts
   ```

   ```
   MaconomyServer.w_20_0.t -Smyshort -xRemove_Old_Layouts
   ```

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

   The –UW option will return errors because the layer functionality is removed.

4. Type the following command to reinstall standard layouts:

   ```
   MaconomyServer.w_20_0.t –Smyshort -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 Deltek Maconomy 2.5.x Release Notes for details.

11.4 Create Indexes for Data Conversion

Note: For large databases see Create Indexes for Data Conversion on page 23.

In general you do not need to create special indexes for the data conversion because the new Convert_data program creates them automatically.

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

Create the following indexes prior to upgrade to optimize data conversion.

CREATE INDEX DATACONV_IDX101 ON APPROVALLINE (INTEGERKEYVALUE1, INTEGERKEYVALUE2, INTEGERKEYVALUE3);

CREATE INDEX DATACONV_IDX102 on APPROVALHEADER (INTEGERKEYVALUE1,INTEGERKEYVALUE2);

CREATE INDEX DATACONV_IDX103 ON USERINFORMATION (EMPLOYEENUMBER);

CREATE INDEX DATACONV_IDX104 on APPROVALLINE(APPROVALGROUPINSTANCEKEY, EFFECTIVE, INSTANCEKEY) include (APPROVER,SUBSTITUTE, SUPERAPPROVER);

CREATE INDEX DATACONV_IDX105 ON APPROVALLINE (APPROVALDATE, APPROVEDORREJECTEDBY);

CREATE INDEX DATACONV_IDX106 ON COMPANYCUSTOMER (ACCOUNTMANAGERNUMBER);

CREATE INDEX DATACONV_IDX107 ON APPROVALGROUP (MAINRELATIONINSTANCEKEY);

CREATE INDEX DATACONV_IDX108 on INVOICEEDITINGHEADER(PAYMENTCUSTOMER,INVOICEEDITINGNUMBER);

CREATE INDEX DATACONV_IDX109 on APPROVALGROUP (APPROVALTYPE);

CREATE INDEX DATACONV_IDX110 on APPROVALHEADER (APPROVALTYPE,INTEGERKEYVALUE1,INTEGERKEYVALUE2);

CREATE INDEX DATACONV_IDX111 on APPROVALHEADER (APPROVALTYPE,INTEGERKEYVALUE1);

CREATE INDEX DATACONV_IDX112 on APPROVALLINE (APPROVALNUMBER,INTEGERKEYVALUE1);

CREATE INDEX DATACONV_IDX113 on JOBENTRY (TYPEOFENTRY);

CREATE INDEX DATACONV_IDX114 on VENDORINVOICEJOURNAL (POSTED,STATUS,SUBMITTEDBY);

CREATE INDEX DATACONV_IDX115 on PAYMENTMODE (REMOVEDPAYMENTMODENUMBER);

CREATE INDEX DATACONV_IDX116 on CUSTOMERPAYMENTMODE (REMOVEDCUSTOMERPAYMENTMODENU);

CREATE INDEX DATACONV_IDX117 on PAYMENTAGENTINFORMATION(REMOVEDPAYMENTAGENTNUMBER);

11.5 Convert Data to the New Version

> **Note:** For large databases see Convert_Data_From_9.0.

The purpose of the Convert_Data.* MSL scripts is to update the database with the appropriate information for the new version. This can include entering required data in new fields and/or relations or calculating the values of specific fields.

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

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

To update the current enterprise language to match those that are available in the new version:

- Run the following MConfig command:

  ```
  c:\PUs\MConfig-8.15.exe -r
  c:\maconomy\tpu.NTx86.20_0.p100.dir\MConfigScriptlets\ChangeToIsoLanguage.mcp
  w_20_0.t myshort myshort en_US
  ```

  In the preceding example, the enterprise language US is changed to en_US.

The output looks like the following:

```
MConfig will run the MCP-Scriptlet

'c:\maconomy\tpu.NTx86.20_0.p100.dir \MConfigScriptlets\ChangeToIsoLanguage.mcp'

-------------------------------------------------------------------------
c:\maconomy\tpu.NTx86.20_0.p100.dir\MConfigScriptlets\ChangeToIsoLanguage.log
Setting enterprise language to en_US for application w_20_0.t
Updating registry for application w_20_0.t
Updating Maconomy.ini
Updating database
-------------------------------------------------------------------------
Script done.
You may now close the window.
```

> **Attention:** For large databases see Convert_Data_From_9.0.

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.

To check whether there are any user-generated statistics and drop them before running Convert_Data, complete:

1. Type `SELECT count(*) FROM sys.stats WHERE user_created = 1;`

   If user-generated statistics exist, type the following to generate the required drop statements.

2. `SELECT 'drop statistics ' + OBJECT_NAME(OBJECT_ID) + '.' + name + ';' FROM sys.stats WHERE user_created = 1;`

All Platforms

```
cd c:\Maconomy\w_20_0.t\MaconomyDir\MSLScripts

MaconomyServer.w_20_0.t -xCheckEnterpriseLanguage -Smyshort
```

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

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

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

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

Four new variables are introduced in Convert_data.

- AllowIndexCreationVar — Boolean. Default value 1 (or true)

  This variable indicates whether it is allowed for Convert_data to create needed indexes during the conversion.

- TableSpaceNameVar:String — Default value ‘ ‘

  If AllowIndexCreationVar is true this variable specifies the table space for index creation.

- OracleDataBaseVar: Boolean — Default value 0 (or false)

  This variable specifies whether the database is Oracle or SQL Server. The default is SQL Server.

- TransactionTypeInitializationVar — Boolean. Default value 0 (or false)

  This variable specifies whether the optimization described in Unicode Conversion is performed. Convert_data checks the change in the expand script anyway.

Start Convert_Data by using the following commands.

All Platforms

```
cd c:\Maconomy\w_20_0.t\MaconomyDir\MSLScripts
MaconomyServer.w_20_0.t -xConvert_Data_From_9.0 -Smyshort
```

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

To check the data conversion status manually:

- Enter the following command:

  ```
  select DATACONVERSIONSTATUS from SYSTEMINFORMATION;
  ```

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

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

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

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

11.6 Remove Indexes for Data Conversion

> **Note:** For large databases see Error! Reference source not found. **on** page **Error! Bookmark not defined.**.

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

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

```
DROP INDEX APPROVALLINE.DATACONV_IDX101;
DROP INDEX APPROVALHEADER.DATACONV_IDX102;
DROP INDEX USERINFORMATION.DATACONV_IDX103;
DROP INDEX APPROVALLINE.DATACONV_IDX104;
DROP INDEX APPROVALLINE.DATACONV_IDX105;
```

```
DROP INDEX COMPANYCUSTOMER.DATACONV_IDX106;
DROP INDEX APPROVALGROUP.DATACONV_IDX107;
DROP INDEX INVOICEEDITINGHEADER.DATACONV_IDX108;
DROP INDEX APPROVALGROUP.DATACONV_IDX109;
DROP INDEX APPROVALHEADER.DATACONV_IDX110;
DROP INDEX APPROVALHEADER.DATACONV_IDX111;
DROP INDEX APPROVALLINE.DATACONV_IDX112;
DROP INDEX JOBENTRY.DATACONV_IDX113;
DROP INDEX DATACONV_IDX114.DATACONV_IDX114;
DROP INDEX PAYMENTMODE.DATACONV_IDX115;
DROP INDEX CUSTOMERPAYMENTMODE.DATACONV_IDX116;
DROP INDEX PAYMENTAGENTINFORMATION.DATACONV_IDX117;
```

12 Validate Custom Components

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

12.1 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.

12.1.1 Screen Layout Validation on the Server

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

1. `MaconomyServer.w_20_0.t -Smyshort -UVW`

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

12.1.2 Print Layout Validation on the Server

Run the following command to validate customized print layouts.

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

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

1. `c:\maconomy\bin\MaconomyServer.w_20_0.t -Smyshort -UEW`
2. `c:\maconomy\bin\MaconomyServer.w_20_0.t -Smyshort -UIW`
3. `c:\maconomy\bin\MaconomyServer.w_20_0.t -Smyshort -UVW`
4. `c:\maconomy\bin\MaconomyServer.w_20_0.t -Smyshort -UEP`
5. `c:\maconomy\tpu.NTx86.20_0.dir\jre\win32\x86\bin\java.exe -jar MPL3to4MigrationTool.jar --dir /data/maconomy/w_20_0/MaconomyDir/ExportedLayouts/`

6. `c:\maconomy\bin\MaconomyServer.w_20_0.t –Smyshort –UIP`
7. `c:\maconomy\bin\MaconomyServer.w_20_0.t –Smyshort -UVP`

12.2 Create Constraints

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

12.2.1 Instance Keys

```
c:\maconomy\bin\MaconomyServer.w_20_0.t -UC –Smyshort
```

12.2.2 Time Sheet Constraints

If any custom constraints were created, these must be removed. Specifically, if constraints for TIMESHEETHEADER and DAILYTIMESHEETHEADER were created as described in the precondition section, these must be removed:

ALTER TABLE TIMESHEETHEADER DROP CONSTRAINT TIMESHEETHEADER1001:

ALTER TABLE DAILYTIMESHEETHEADER DROP CONSTRAINT DAILYTIMESHEETHEADER1001:

12.2.3 Other Constraints

To recreate SQL Server specific constraints, enter the following commands:

```
cd c:\Maconomy\w_20_0.t\MaconomyDir\Database

StartSQLServer2 myshort myshort ISQLCreConstraints.sql
c:\Logs\ISQLCreconstraints.myshort.log
```

12.3 Install and Upgrade the Portal

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

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

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

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

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

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

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

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

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

12.4 Convert the Portal Database

To upgrade the Portal database:

1.
```
update [myshort].[dbo].[COMPONENT]
set CALLBACKSCRIPT = replace(CALLBACKSCRIPT, 'W_12_0', 'W_20_0')
where CALLBACKSCRIPT like '%W_12_0%';
```
2.
```
update [myshort].[dbo].[MACONOMYWINDOW]
set SPECFILENAME = replace(SPECFILENAME, 'W_12_0', 'W_20_0')
where SPECFILENAME like '%W_12_0%';
```
3.
```
update [myshort].[dbo].[MSCRIPTCOMPONENT]
set MSCRIPTFILENAME = replace(MSCRIPTFILENAME, 'W_12_0', 'W_20_0')
where MSCRIPTFILENAME like '%W_12_0%';
```
4.
```
update [myshort].[dbo].[PDMCOMPONENT]
set CONFIGURATIONFILE = replace(CONFIGURATIONFILE, 'W_12_0', 'W_20_0')
where CONFIGURATIONFILE like '%W_12_0%';
```
5.
```
update [myshort].[dbo].[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.

12.4.1 Install the Web Server and Portal

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

To install the web server and Portal:

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.

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.

12.4.2 Install Maconomy Server Industry Accelerator Files

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

To install the industry accelerators that are supplied with PSO SPU:

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.<Replace text>

12.4.3 Set Up the Portal Components

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

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

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

```
cd C:\maconomy\WebServers\w_20_0.t\cgi-bin\Maconomy

MaconomyPortal.myshort.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:\WebServers\Production\MaconomyPortal.

The files to be imported are located in ‘c:\maconomy\w_20_0.t\Portal\Installation’ on Windows. Component definitions and default role setup and menus are available.

Import files for industry accelerators are located in ‘c:\maconomy\w_20_0.t\IA.<Industry Accelerator>\Portal\Installation’ on Windows.

12.4.4 Validation of Portal Components

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

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

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

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

You can also export component definitions using the command window. In the following example the web server is a Windows server, and the home directory for the web server is 'C:\maconomy\WebServers\w_20_0.t\'.

Example: Export the component definition

```
cd C:\maconomy\WebServers\w_20_0.t\cgi-bin\Maconomy

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

The export file is located in:

```
C:\maconomy\WebServers\w_20_0.t\MaconomyPortal\Maconomy2_1.portal_comps.xml
```

Example: Import and validate the components

```
MaconomyPortal.myshort.en_US.exe -e -q
"filename=Maconomy2_1.portal.portal_comps.xml&username=Administrator&password=123456&im
porttype=components" Framework/import/importportal.msselect
```

13 Post Upgrade Tasks

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

13.1 Recreating 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 of 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.

13.2 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.

13.3 Check Related Product Compatibility

After upgrading Maconomy to a new version, it is important to check if all the integrations are still compatible or if they also require a new version. Integrations that should be checked after upgrade include:

- iAccess
- Touch
- People Planner

13.4 Approval Hierarchies

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.

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.

13.5 Advanced Logging

If advanced logging was configured in the original Maconomy application, please remember to migrate it to the new Maconomy application after the upgrade. For more on advanced logging configuration, please read the Maconomy System Administrator Guide.

13.6 Disable Maintenance Mode

At this point, the Maconomy application should be put out of maintenance mode using MConfig 8.15 or later. Putting the system out of maintenance mode will re-enable background tasks.

14 Hints, Tips and Tricks

14.1 Dump Changes to Standard Indexes

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

- JobInvoiceLine: NOTINCLUDEDINFIXEDPRICE
- JobEntry: NOTINCLUDEDINFIXEDPRICE

To find and drop indexes in SQL Server, enter the following commands:

```
select
   convert(varchar(128),sysindexes.name) "INDEX_NAME"
from
   sysindexes, sysindexkeys, syscolumns
where
   sysindexes.indid = sysindexkeys.indid and
   sysindexes.id = syscolumns.id and
   sysindexkeys.id = syscolumns.id and
   sysindexkeys.colid = syscolumns.colid and
   sysindexes.indid <> 255 and
   sysindexes.indid > 1 and
   sysindexes.name not like '%\_Sys\_%' escape '\' and
   syscolumns.name = 'notINCLUDEDINFIXEDPRICE'
order by
   sysindexes.name,  sysindexkeys.keyno
```

Execute the output from the preceding script to drop indexes.

14.2 Drop Indexes

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

14.3 Expand Script

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

To optimize the expand script:

- Replace the following:

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

  with:

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

Note: This change only works if you also perform the changes that are described in Convert_Data_From_9.0.

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

14.4 Convert_Data_From_9.0

Four new variables are introduced in Convert_data_From_9.0. You can access these variables by entering a $ sign after the MSL program is started.

The four variables are:

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

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

15 Appendix: Error Messages

15.1 Maconomy Errors

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

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

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

15.2 MConfig Errors

During an upgrade you use MConfig for the following:

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

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

16 Appendix: NameChanger Tool

In Maconomy version 2.5.x many names, especially those of database fields, have been changed. It can therefore be expected that modifying source files for customer customizations can be a complex task during the upgrade process.

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

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

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

This command processes all of the files in the folder tree C:\CustomizedFiles and stores modified files in the folder tree C:\ModifiedCustomizations. In this example only files of types MDL and MDML are modified. All files in C:\CustomizedFiles appear in C:\ModifiedCustomizations, some of them (for example, files of types other than MDL and MDML) possibly unchanged. The source files are for Maconomy version 15 (that is, 2.0), and the modified files are for Maconomy version 17 (that is, 2.5.x).

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

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

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

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

Note that Deltek does not guarantee that the NameChanger tool makes all of the required changes to the source files. This tool might sometimes make changes that should not have been made. You must still manually verify the changes. However, tests show that at least when upgrading to version 2.5.x, the number of correct translations by far outnumbers the number of mistakes.

17 Appendix: Unicode Conversion

Deltek Maconomy version 2.5.x requires that all databases are Unicode (UTF8). The conversion takes place before the Maconomy upgrade process, but be aware that no Maconomy clients before Main version 2.1 can run on a Unicode database.

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

Note: We recommend that you delete unused columns before you perform the Unicode conversion.

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

17.1 Unicode Data Conversion

The Maconomy application has always used the UTF8 character set in SQL server. Download dcusql.zip, which is the latest version of the DCU for SQL Server.

17.1.1 Install and Configure DCUSQL

Unzip the dcusql and edit the config.cfg file to match the current MSQL credentials. A configmsql.cfg file might look like the following:

```
[DestinationDB]
ip=172.16.185.226
instance=default
database=myshort
username=myshort
password=myshort
```

17.1.2 Convert Strings

To run the dcusql:

- Enter the following command:

  cd C:\PUs\DeltekMaconomyUpgradeKit24GA\UnicodeTools\dcusql\dcusql

  dcusql

If the config.cfg file is correct, the following files are generated:

- dropconst.sql, which drops all constraints in the database.
- dcusql.exe dropconst
- convert.sql, which converts all VARCHAR columns to NVARCHAR (national character set). This conversions all strings to UTF16 characters based on the collation specified for the database
- dcusql.exe convert
- makeconst.sql, which recreates the constraints.

- dcusql.exe makeconst

–

Attention: If any errors occur during the execution of convert.sql, you should fix them and run convert.sql again. Likely errors are views/indexes/constraints/triggers that have not been dropped.

To fix these errors, save the definitions of the views to be dropped, drop all of the views/indexes/constraints, re-run convert.sql without errors, and recreate the views.

17.2 Convert Text Strings

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

To find offending strings:

- Enter the following command:

  ```
  cd C:\PUs\DeltekMaconomyUpgradeKit24GA\UnicodeTools\smu\smu
  smu --extract
  ```

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

17.2.1 Shorten Offending Strings

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

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

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

```
/‟/:/"/
/ ‟ /:/"/
/”/:/"/
/ ” /:/"/
/’/:/'/
/ ’ /:/'/
/‘/:/'/
/ ‘ /:/'/
/–/:/-/
/ – /:/-/
/—/:/-/
/ — /:/-/
/ + /:/+/
/ = /:/=/
/million/:/M/
```

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

To see the effect of the replacements:

- Enter the following command:

  ```
  smu --translate
  ```

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

When all strings are fixed, import them.

To import all strings:

- Enter the following command:

  ```
  smu --import
  ```

To check that all strings are valid:

- Enter the following command:

  ```
  smu --extract
  ```

  and check that the offendingstrings.txt file is empty.

18 Appendix: Application Pre-Upgrade

18.1 Refresh Database Views after ETL Upgrade

Users are unable to use Analysis reports that utilize the Job Cost universe unless they do a full refresh of the database views. This is due to mapping errors on the database views after ETL has modified the tables.

To refresh database views, run the following script:

```
SQLRefreshView.sql
```

18.2 Add Scripts for Missing Validations

Customers upgrading to 2.3 LA or later from versions older than 2.3 LA are missing validations for company vendors. The following scripts return the vendor number and company number for which company vendors are missing.

> **Note:** Running these SQL scripts in the database may return some rows with columns named VN and CN, where VN is vendor number, and CN is company number.
>
> You must create company vendors with VN - vendor number and CN - company number before beginning the upgrade.

SQL 1

select distinct

```
RequisitionHeader.SuggestedSupplierNumber VN,
RequisitionHeader.CompanyNumber CN
```

from

```
RequisitionHeader
```

where

```
RequistionHeader.SuggestedSupplierNumber <> ' '
and not exists (select 1
```

from

```
CompanyVendor
```

where

```
RequisitionHeader.SuggestedSupplierNumber=CompanyVendor.VendorNumber
and RequisitionHeader.CompanyNumber=CompanyVendor.SettlingCompany);
```

SQL 2

select distinct

```
RequestForQuoteHeader.SupplierNumber VN
RequestForQuoteHeader.CompanyNumber CN
```

from

```
      RequestForQuoteHeader
where
      RequestForQuoteHeader.SupplierNumber <> ' '
      and not exists (select 1
from
      CompanyVendor
where
      RequestForQuoteHeader.SupplierNumber=CompanyVendor.VendorNumber
      and RequestForQuoteHeader.CompanyNumber=CompanyVendor.SettlingCompany);
```

SQL 3

```
select distinct
      PurchaseOrderHeader.SupplierNumber VN
      PurchaseOrderHeader.CompanyNumber CN
from
      PurchaseOrderHeader
where
      PurchaseOrderHeader.SupplierNumber <> ' '
      and not exists (select 1
from
      CompanyVendor
where
      PurchaseOrderHeader.SupplierNumber=CompanyVendor.VendorNumber
      and PurchaseOrderHeader.CompanyNumber=CompanyVendor.SettlingCompany);
```

SQL 4

```
select distinct
      VendorInvoiceJournal.SupplierNumber VN
      VendorInvoiceJournal.CompanyNumber CN
from
      VendorInvoiceJournal
where
      VendorInvoiceJournal.VendorNumber <> ' '
      and not exists (select 1
from
      CompanyVendor
where
      VendorInvoiceJournal.VendorNumber=CompanyVendor.VendorNumber
```

```
      and
VendorInvoiceJournal.SettlingCompany=CompanyVendor.SettlingCompany);
```

18.3 Ensure Instancekeys in Userinformation and Approvalgroup Tables

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.

If the **Instance Keys Applied** field shows **No**, click the action **Initialize**

18.4 Update Layouts for Global and Local Dimensions

> **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**.

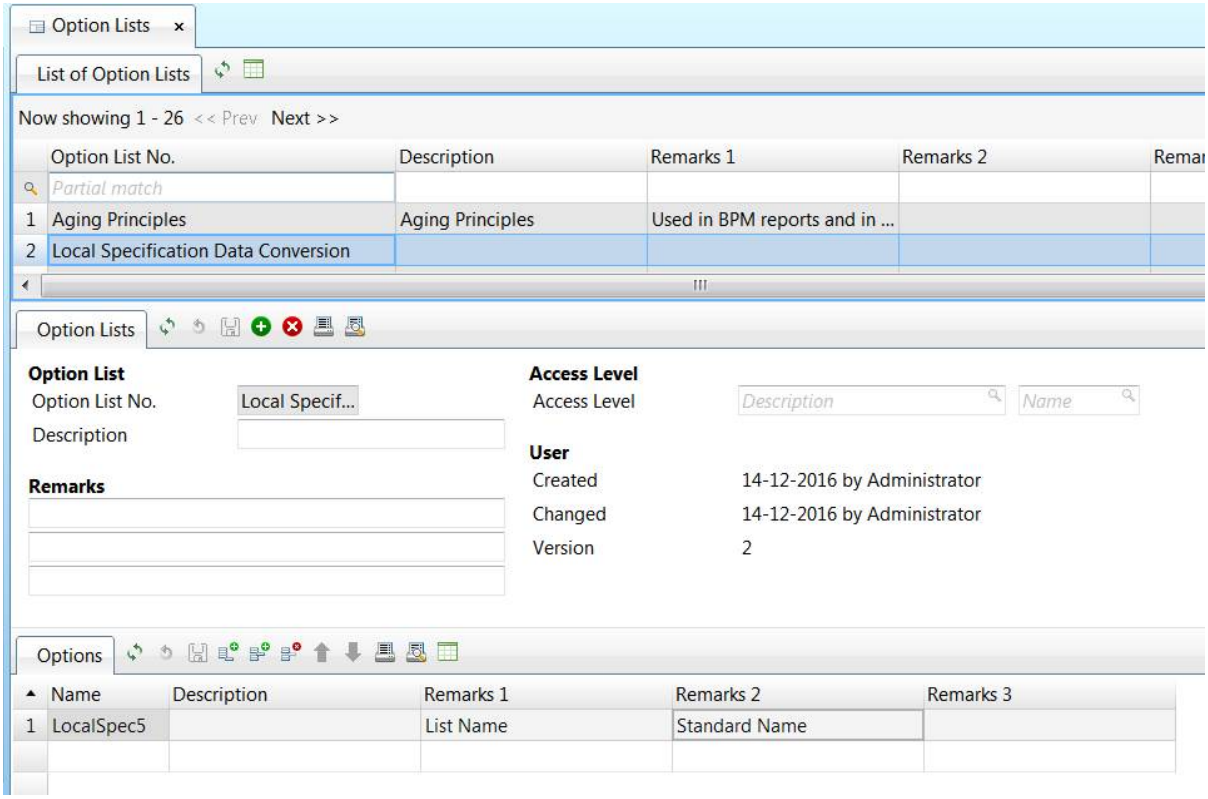
   Now, each line in the table represents a new local specification list.
3. Add a new line in the table as needed.

   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.



18.5 Local Charts of Accounts

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**.

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

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

The purpose of the LCoA Pre-upgrade script is to verify that there will not be issues when upgrading to 2.4 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**.

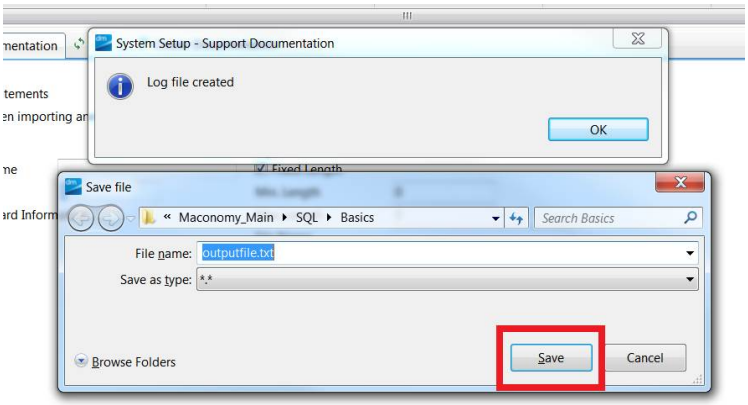
Run the Script

To run the script:

1. Locate the script in the Upgrade Kit, which is located on the drive to which it was extracted. For example, if the Upgrade Kit was extracted to C: find the script here:

   C:\UpgradeKit_2.3GA\sql\LocalChartOfAccountsPreUpgrade.sql
2. Open the Workspace/Java client of the version you are upgrading. For example, if you are upgrading from 2.1 to 2.3 LA, run the script in your 2.1 Maconomy system.
3. Open the Maconomy client and go to **Setup » System Setup » Support Documentation** and choose **Import SQL Statements** in the filter.
4. In the card part **Statement No.** field, select **1**.
5. Click **Create Documentation**. A new dialog opens.
6. In the new dialog, browse to the location of the LocalChartOfAccountsPreUpgrade.sql file and click **Open**.

   The SQL script runs and executes various SQL statements in the database. After it is done executing the SQL statements, it produces an output file with the feedback (see the following image).
7. Click **Save** to save this output file so that you can investigate and verify that the system setup satisfies the requirements for upgrade.



You must now evaluate the output and create options lists.

Evaluate the Output

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

The 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. The Create Option Lists procedure is described in the Dimensions section above.

18.6 Reporting Structures

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

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

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

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

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

18.7 Change References to Depreciated Fields on Customers and Vendors

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

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

18.8 Update Custom ControlHours.1.ms File

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

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

18.9 Facilitate Workflow of MCSInvoiceApproval

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

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

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

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