



Agresso Technical Guidelines

Guidelines for Installation and Configuration of Agresso

 You will always find the latest version at <http://abwdocs.agresso.no>

- [What's New in Technical Guidelines](#)
- [Agresso Installation](#)
- [Configuring Agresso](#)
- [Upgrading Agresso](#)
- [Agresso Maintenance](#)
- [Security](#)

UNIT4 Agresso Milestone 4(5.7.0)

[Copyright © 2013 Agresso R&D AS. All Rights Reserved.](#)

Introduction

Content overview

These Technical Guidelines give you detailed information on how to install and configure Agresso. The content is divided into the following main sections:

- [What's New?](#). Explains briefly the new features, compared to previous versions.
- [Installing Agresso](#). Describes in detail how you carry out a complete Agresso installation, including database installation, client installation and server installation. You will also find information about how you can install a stand-alone demo version of Agresso on a single computer.
- [Configuring Agresso](#). Describes how you use the [Agresso Management Console](#) (AMC) to set up Agresso according to your company's needs.
- [Upgrading Agresso](#). Describes the necessary steps to upgrade from a previous Agresso version to the current version. See for example [Upgrading from 5.6.X to 5.7](#).
- [Agresso Maintenance](#). Describes necessary operating (day-to-day) [routines for your database installation](#), as well as important

stuff about [tuning](#) and security.

- **Appendix.** This section covers technical subjects and goes more into detail regarding [Agresso Database Tools](#).

Anything written in this document is in principle not legally binding in declaration of intent.

What's New in Technical Guidelines

Release and update information

Current document release:	1.0 (Milestone 4, 5.7.0, Platform 3.3)
Release date:	2013-07-25
Last updated:	View update history

Read the [release history](#) for a list of the latest modifications to technical guidelines.

Notes on the UNIT4 Agresso Milestone 4, version 1.0 release of Technical Guidelines

This new version of the Technical guidelines are updated according to the new UNIT4 Agresso Milestone 4, and introduces the following news:

- The **Agresso Management Console (AMC)** has been rewritten as a pure .NET application and given a slightly different menu structure and appearance. As a consequence, everything about [Configuring Agresso](#) in Technical Guidelines is also rewritten. Follow [this link](#) for an introduction to the new AMC.
- Some setup and configuration options previously available in the **Agresso Smart client** has been transferred to the **Agresso Management Console** - where they naturally belongs. See [Authentication, Users and Access, ACT](#).

Notes on the UNIT4 Agresso Milestone 3, version 1.1 release of Technical Guidelines

According to new EU regulations, we must explains the Self service application's use of cookies. Only an English version is made available after installation, and if you use any other language, you must translate the text. The details are described [here](#).

Notes on the UNIT4 Agresso Milestone 3, version 1.0 release of Technical Guidelines

New topics

- [Upgrade the Agresso installation](#) from the previous version.
- [Migration from Oracle to MS SQL Server](#)

Agresso.Management.PowerShell

Agresso.Management.PowerShell is a new navigation provider, providing a series of Agresso specific PowerShell objects and methods. After you have added the Agresso.Management.PowerShell snap-in, you can write scripts for efficient and secure configuration. This documentation is not included in technical guidelines, but can be found as an pdf-document installed under [.\Agresso 5.7-0\PowerShellScripts](#).

Notes on the UNIT4 Agresso Milestone 2, version 1.0 release of Technical Guidelines

Agresso Management Console

- The **Optional Services** node is renamed to **Web Service Hosts**. This is where you set up the Agresso Event Server.
- New solution for file storage in Document archive. You should upgrade AGRESSOFILE to the new service based solution.
- Web services and Windows server 2008: You need to modify folder permissions for IIS_IUSRS. See Adding Agresso Web Services

Agresso security

Configure integrated database authentication for Smart Client. See [Enable Integrated Database Authentication](#).

Notes on the Milestone 1, version 1.0 release of Technical Guidelines

.Net 4.0 is required!

Agresso Business World Route 66 is based on Agresso Platform 2.5 and requires **.Net 4.0 Full Profile**. All users of the **ABW Smart Client** must have .NET 4.0 Full Profile installed on their local computer!

Important - running Smart Client from server: Users that run the Smart Client from a server (Centrally Configured Client) may find that the Smart Client simply will not execute, with no warning.

This will happen if there is no (older) version of .NET Framework installed on the local computer, which may be the case for computers with relatively new Windows versions.

New functionality

Agresso 5.6 contains new functionality (new solutions) for the following areas covered by Agresso Technical Guidelines:

- **Logging:** The new solution for logging is detailed described in [Logging in Agresso](#) in the Appendix. Details about specific setup via **AMC** is found in Configuring Logging.
- **Process Execution Control:** You can now monitor all server queues and write status information to Windows event log.
- **ASQL:** A series of new ASQL functions are made available in ABW 5.6. See [Data Types and Functions](#) and [Data Manipulation Language \(DML\)](#) in the Appendix. For all ASQL update details, see (separate PDF-document) *Release Notes, Agresso Platform 2.2, News Introduced in the PLRA Project*.
- A New environment variable is added EXPAND_PATH_WITH_CLIENT. See Initialize and Maintaining the Business Server Environments.
- You can now set Self Service in maintenance mode from AMC, meaning that a user will get a friendly message when the self service client is taken down. See Managing Agresso Self Service from more details.
- Agresso Event Server replaces the Agresso Alert Server web service.

Updates between release dates

The Technical Guidelines allows immediate updates and corrections. By publishing the updates to our web site, the last and complete version will always be available, for all reader groups. For all important releases, the decimal in the version number - starting at 1.0 for each major release and service pack - will be increased.

You will get all update details by following the link [View update history](#) at the top of this page.

Release history - Technical Guidelines

The table lists all published versions of the Technical Guidelines, with last version on top:

Version	Technical Guidelines	Comments
5.7.0	2013-06-24	Agresso 5.7.0 (Milestone 4)
5.6.4	2013-06-13	Agresso 5.6.4 (Internal release)
5.6.4	2012-11-25	Agresso 5.6.4 (Internal release)
5.6.3	2012-07-20	Added information about the Privacy Policy law page.
5.6.3	2012-06-07	Added information about the AGR_TMP_LOC environment variable
5.6.3	2012-05-25	Agresso Business World Route 66 Milestone 3 (5.6.3)
5.6.2	2011-12-15	Oracle 11.2 workaround no longer needed from Oracle 11.2.0.3.0. Updated Finalise Upgrade documentation.
5.6.2	2011-11-28	Agresso Business World Route 66 Milestone 2 (5.6.2)
5.6.1	2011-10-25	Info on oracle 11.2 patches required by Agresso. See Agresso on Oracle .
5.6.1	2011-06-30	Removed information about "show content pane" in the web help documentation , since this options is no longer available in Agresso 5.6
5.6.1	2011-06-06	Updated Oracle update samples with a sample for merge into syntax on oracle 11.2
5.6.1	2011-05-31	New database setup requirements for MS Sql Server and Oracle , to allow the database to read external files. This is needed after applying the first Software Update for Route 66 Milestone 1. (mssql: "grant administer bulk operations to <login>", Oracle: "grant create any directory")
5.6.1	2011-05-13	Agresso Business World Route 66 Milestone 1
5.6.0	1.1, 2011-05-10	Updated security guidelines .
5.6.0	1.0, 2010-10-30	Important improvements on server.
5.5.3	1.1, 2010-05-21	Updated the SDS and Windows Firewall(client) documentation.
5.5.3	1.1, 2010-07-05	Added link to latest report engine installation on the Report Engine Installation page
5.5.3	1.1, 2010-06-05	Changes to Agresso On MS SQL Server. Updated list of things to remember when creating an Agresso database on MSSQL. "view server state" must be granted.
5.5.3	1.1, 2010-03-11	Changes to Agresso On MS SQL Server. Updated recommendation for Agresso on MSSQL. It's recommended to set Page verify to CHECKSUM.
5.5.3	1.1, 2010-01-06	Added documentation on how to configure access for SDS in the windows firewall on the server.

5.5.3	1.1, 2009-09-14	Changes to Agresso On MS SQL Server. Updated recommendation for Agresso on MSSQL (READ_COMMITTED_SNAPSHOT on). And made it clear that agrtempdb is optional.
5.5.3	1.1, 2009-04-18	ASQL documentation updated. New functions and datatypes documented, and more samples added
5.5.3	1.1, 2009-03-05	Note about agr_session view on an Oracle RAC
5.5.3	1.1, 2009-02-04	Minor corrections on database installation.
5.5.3	1.0, 2008-09-15	Built on top of Agresso Platform 1.1
5.5 SP2	1.1, 2008-04-10	Added environment variables. All available Agresso environment variables are now documented and indexed.
5.5 SP2 Update 0.2	1.1, 2007-09-14	Improved the upgrade process from 5.4 to 5.5.2
5.5 SP2	1.0, 2007-06-01	First version for ABW 5.5 Service Pack 2
5.5 SP1	1.1, 2006-03-13	Added printable PDF version of TechGuide55. Minor corrections.
5.5 SP1	1.0, 2006-02-01	First final version.
5.5	2005-07-14	First BETA version in html format.

Installation options

Complete installation versus Demo installation

Complete installation

The main topics in this section cover a complete Agresso installation in a complex environment, where you will need to install all of the following:

- Database tables
- Server components
- Client components
- Help system

Most parts of the Agresso system will be installed by the **UNIT4 Agresso** installation wizard. The exceptions are Agresso Report Engine and Agresso Web Help which must be installed separately.

Note: Please note that these guidelines do not cover the installation of a database system, i.e. **Oracle** or **MS SQL**.

The Installation wizard

Installation options

The file [SetupAgresso.exe](#) on the installation DVD, contains wizards for installation of all the Agresso components.

The main installation options are explained below:

Option	Description
UNIT4 Agresso	Allows you to install one of the following <ul style="list-style-type: none">• Complete Agresso. You can de-select unwanted components.• Smart Client. Installs the Smart Client component only. The wizard has de-selected the other components for you.• Demo. Installs the Agresso demonstration version on a single computer.
Agresso Demo	Runs scripts for configuring Agresso for demonstration purposes. This option requires that UNIT4 Agresso is installed. The Agresso Demo setup scripts will be executed from the UNIT4 Agresso installation when the "Agresso Demo" installation options is selected.
Web Help	Installs the Web help system on your web server. There is one installation for Smart Client and one for Self Service.
Bar Code	Installs the Bar code reader (currently Pegasus Barcode Xpress v5.5).
Agresso OCR	Installs optical character recognition software used by document archive.

Installation options

Complete installation versus Demo installation

Complete installation

The main topics in this section cover a complete Agresso installation in a complex environment, where you will need to install all of the following:

- Database tables
- Server components
- Client components
- Help system

Most parts of the Agresso system will be installed by the **UNIT4 Agresso** installation wizard. The exceptions are Agresso Report Engine and Agresso Web Help which must be installed separately.

Note: Please note that these guidelines do not cover the installation of a database system, i.e. [Oracle](#) or [MS SQL](#).

The Installation wizard

Installation options

The file [SetupAgresso.exe](#) on the installation DVD, contains wizards for installation of all the Agresso components.

The main installation options are explained below:

Option	Description
UNIT4 Agresso	Allows you to install one of the following <ul style="list-style-type: none">• Complete Agresso. You can de-select unwanted components.• Smart Client. Installs the Smart Client component only. The wizard has de-selected the other components for you.• Demo. Installs the Agresso demonstration version on a single computer.
Agresso Demo	Runs scripts for configuring Agresso for demonstration purposes. This option requires that UNIT4 Agresso is installed. The Agresso Demo setup scripts will be executed from the UNIT4 Agresso installation when the "Agresso Demo" installation options is selected.
Web Help	Installs the Web help system on your web server. There is one installation for Smart Client and one for Self Service.
Bar Code	Installs the Bar code reader (currently Pegasus Barcode Xpress v5.5).
Agresso OCR	Installs optical character recognition software used by document archive.

Installing and Using the Agresso Demo system

Purpose

The Agresso Demo Setup option allows the user to automatically configure Agresso on a single computer for demonstration purposes. The Demo system will automatically install MS SQL Server Express if no existing supported local Sql Server instances exist.

Install the Demo system

1. Start the **Agresso** installation from **SetupAgresso** and select the **Agresso Demo** option.
2. Follow the instructions on screen until the installation is complete.
3. Click **Finish** to start the Agresso Demo Setup scripts.

The Agresso Demo setup scripts can also be launched on an existing complete Agresso installation by using the Agresso Demo option available in **SetupAgresso.exe**. The Agresso Demo Setup scripts can be executed as many times as you want, but keep in mind that existing configurations for Self Service and Web Services (**web.config**) will be overwritten when running the setup scripts.

You can log in to the Agresso demo system as user **sysenlong** and with password **agresso**.

Installed components

Overview

The following Agresso components will be configured by the Agresso Demo Setup scripts

- Business Server with all report and process controls
- Self Service
- Self Service (Experience Packs)
- Web Services host
- Smart Client
- Self-Hosted Services
- Agresso Smart Client

Database server details

The database server settings are by default as follows:

Item	Value
MS SQL Server instance name	AGRDEMO
Database name	agrdemo570
Database login	agrdemo570
Database login password	agresso
Sa password	AgressoDemo1

The MS SQL Server instance:

When connecting to an SQL Server instance, the computer name be added as a prefix. If the instance name is AGRDEMO by default, the instance name to connect to, is therefore:

<machine name>\AGRDEMO

Using the demo installation

Installed features

The Demo system gives you full access to the standard Agresso clients (Smart client - windows application, and Self Service client - Web application), using `sysenlong` as username and `agresso` as password.

The Agresso database contains so-called template data, allowing you to test and demonstrate most of the Agresso functionality.

Requirements

- Complete UNIT4 Agresso installation
- Windows PowerShell v2.0 (Windows PowerShell is installed by default from Windows Server 2008 R2 and Windows 7)
- 64-bit Operation system, Windows Vista (6.0) or higher.
- [SQL Server 2008 requirements](#)

PowerShell v2.0 will automatically be installed if the Agresso Demo Setup scripts are started directly from SetupAgresso. Windows PowerShell v2.0 can also be downloaded from: <http://support.microsoft.com/kb/968929/en-us>.

Note: You will be unable to install SQL Server 2008 if the **SQL Server 2005 Express Tools** are installed (included with the **553 Agresso Demo** installation). You need to uninstall this feature to be able to install SQL Server 2008.

Logging (for troubleshooting purposes)

The table below shows where you find the various log files when installing Agresso Demo.

Type	Location
AgrDemo installation log-file	%TEMP%\AgrPowershellLogs\
SQL Server 2008R2 Express installation log	C:\Program Files\Microsoft SQL Server\100\Setup Bootstrap\Log

The Installation Process

Prerequisites

Supported database platforms

Agresso requires either an **Oracle** or an **MSSQL** database system.

See also [Supported platforms](#).

Sufficient access rights for installer

The account used during the installation must have sufficient rights to be able to write to the registry during the installation, both to HKEY_CURRENT_USER and HKEY_LOCAL_MACHINE. The account must also have sufficient rights to install files to the system32 directory.

Process description

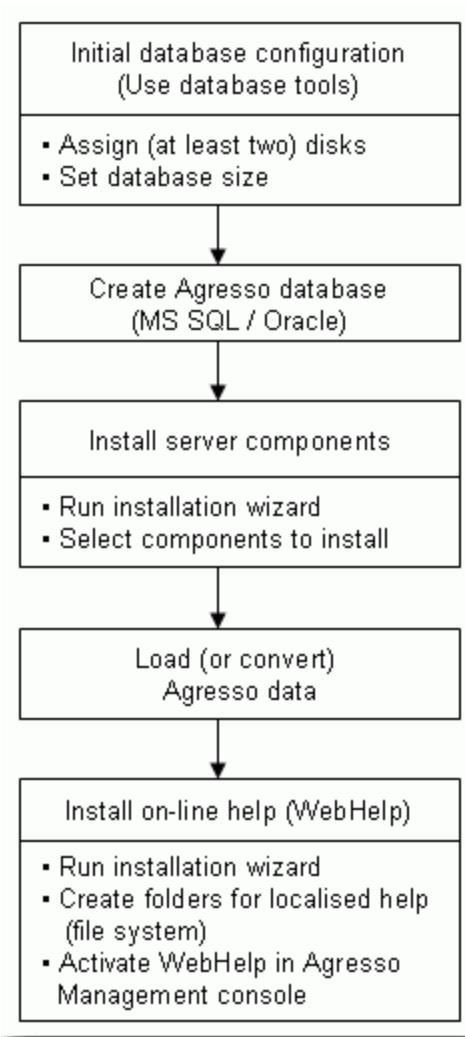
When you have the RDBMS up and running, the database must be configured according to the Agresso requirements. More than one thousand Agresso tables are in effective use, and needs to be taken care of.

Next, you must install the server components. Among the components you find **Agresso Management Console**, which is the main tool for later configuration tasks.

Finally, you complete the Agresso installation by installing the on-line help system.

Process diagram

The diagram below outlines the main activities in the installation process:



Initial Configuration

Minimum requirements

The Agresso database must have at least two disks available, one for data and one for log. It is possible to install it on a single disk, but this is only recommended if your installation is very small, or it is merely a test installation.

All update queries sent to the database causes write activity to one or more data device pages and to the log device. If both data and log are stored on the same disk, these two writing processes will interfere with each other, which may result in poor performance.

Recommended hardware setup

Storage Area Network (SAN)

SAN can be used for the database and the Agresso servers.

Standalone disks and RAID (Redundant Array of Independent/Inexpensive Disks)

The number of separate disks for data and log storage is one of the most important factors determining the performance of your Agresso system. Generally speaking, the more disks, the better performance.

The temporary tables in Agresso should also have a separate disk. If not, some of the larger server processes in Agresso may be very slow for large volumes of data.

The number of disks and the number of disk controllers determine the efficiency of your data distribution. How many disks you need depends on the size of your database, the number of concurrent Agresso users, the profile of these users (batch, on-line processing), etc. It is possible to use 7 or 8, maybe 10 disks for Agresso and your RDBMS combined. In most installations this is not feasible, and probably not a good idea anyway. Between 3 to 6 disks should be sufficient. This depends on the size of the database and the number of concurrent users.

Disk controllers

The number of disk controllers depends on the number of disks and on the speed and capacity of each controller. Generally, in a multi-disk environment, more disk controllers will speed up disk access, and thus speed up Agresso. If you install extra disks on a computer, it may be a good idea to add an extra controller. Your hardware vendor may help you determine the number of controllers necessary for your disk configuration.

RAID and disk mirroring

In general, Agresso recommends using several separate disks to speed up the application. If using RAID, make sure that at least two disks are available outside the RAID system. The database log and the temporary tables in Agresso need to have their own separate disks since both log and temp tables involve sequential write operations. For security reasons you may also want to use a separate disk for mirroring the log files. Mirroring of the disk containing the Agresso temporary tables is not required.

! Remember that RAID is not 100% safe. Thus archiving and backup is still essential.

The type of RAID to use depends on your priorities, security, the amount of available disk space and performance. RAID 1 is simple and gives maximum security (complete mirroring of all files), but it requires a lot of disk space and involves a lot of writing activity. A RAID controller with a lot of memory cache may help speed up the use of RAID.

Agresso Data Segments

Five segments for the Agresso data objects

The data objects in Agresso can be distributed among five different segments:

agrdata	Contains the Agresso tables without BLOB/IMAGE columns.
agrblob	Contains all the tables with BLOB/IMAGE.
agrdoc	Contains only the adsfileblob table (used in document archive).
agrindex	Contains the Agresso indexes.
agrscratch/agrtempdb	Contains real tables used for temporary data during the execution of server processes.
temp/agrtemp/tempdb	Contains real temporary tables used for temporary data during the execution of server processes.

Device versus segment

In some databases there are only one relevant level of data distribution. For instance, Oracle puts user tables in tablespaces. Each Agresso segment is equivalent to one tablespace. All default tablespaces in the Create Database Scripts delivered by Agresso should be created in every installation.

If you only have a few disks, there is no real point in creating a physical device for each Agresso segment (in these RDBMS's). You may create one device on each disk and optionally add multiple segments on each device. [Agresso Copy](#), used to load data files from Agresso into your database, uses the standard Agresso segments. Therefore, it is a good idea to create all these segments, even though you may have only two or three physical devices.

Number of Disks	2	3	4	5
RDBMS	1	1	1	1
LOG	2	2	2	2
AGR-SCRATCH/AGRTEMP/AGRTEMPDB/TEMP	1	3	3	3
AGRINDEX	2	2	2	5
AGRDATA	1	1	4	4
AGRLOB	1	1	4	4
AGRDOC	1	1	4	4

The differences are outlined in the table:

	Oracle	MS SQL
RDBMS	<p>The SYSTEM, UNDO, TEMP and TOOLS tablespaces.</p> <p>Moving the tablespaces UNDO and TEMP to other disks may affect performance in a large installation. Oracle control files may be placed on any disk.</p>	<p>The RDBMS Software and the MASTER device. Mirroring of master (for security) may be placed on any disk.</p>
LOG	<p>The redo-log files are normally placed on a RAID 1 (a set of three log files). If you wish to have mirroring of the log files, use another disk for the next set (MIRRORING).</p> <p>If you use archiving, the archive files may be placed on any disk (not a disk with database devices). However, the disk should have a lot of free space.</p>	<p>The log device contains the log for both the agresso and the agrtempdb databases.</p>
AGRSCRATCH/AGRTEMP/AGRTEMPDB/TEMP	<p>The tablespaces agrscratch, agrtemp and temp.</p>	<p>The data device for databases agrtempdb and tempdb.</p>

AGRTEMP/AGRTEMPDB and TEMP/TEMPDB

Most server jobs (and reports) create one or more tables for temporary storing data that exist only for the duration of the job. These database tables are usually created at the beginning of the job and dropped at the end of the job.

Much of the total writing to the database in a typical Agresso installation is writing to these short-lived tables. Moreover, both in Oracle and SQL Server all updating of these are logged to the database transaction log.

For debugging purposes it is possible to skip the drop command on these tables by setting the parameter AGR_TMP_SAVE.

From Agresso 5.5 these tables are created as temporary tables. When using SQL Server this is done by giving the table name the following prefix: ##. When using Oracle, it is done by issuing a "create global temporary table" command. Oracle use the temporary tablespace assigned to the user, and SQL Server use TEMPDB for the instance to hold the real temporary tables.

Updates to temporary tables are not logged. This gives a significant speed improvement for both database systems.

A side effect is that some of the IO is redirected to the temporary tablespace for Oracle, and to the TEMPDB database for SQL Server.

Especially for Oracle installations the reduced logging is a major improvement considered the size of archive files to back up. Installations that do log shipping to a mirror will experience a significant reduction of data to transfer.

If AGR_TMP_SAVE is switched on, the tables will be created the same way as in Agresso 5.4.

If it is required to have the old type of tables, but not to keep them when the job is completed, the parameter AGR_USE_REAL_TABLES can be set to TRUE. This turns the behaviour back to the way it was in Agresso 5.4.

About database configuration

Initial configuration of the database

Instead of giving a detailed description on how to install the Database software, Agresso recommends visiting database vendors web sites for downloading the necessary programs, patches and documentation. One advantage is that this documentation will always contain the latest information regarding installing, configuring, administering and tuning the database programs for Agresso.

There are three main steps for the installation of an database:

1. Pre-installation - information on how the OS should be installed/patched and all other requirements
2. Actual installation - information on how to install the software
3. Post-installation - information on how to configure the database components and how to get the patches

 We recommend reading the installation guide thoroughly and follow their recommendations closely.

 Agresso recommends installing the latest patch set release after a successful installation of the Database.

Agresso Management Console - your main configuration tool

When you later will configure the completed installation, you will mainly use the [Agresso Management Console](#) - a utility program that always must be installed on the Agresso server.

Agresso on MS SQL Server

General SQL Server settings

Microsoft® SQL Server™ automatically tunes many of the server configuration options, thus requiring little, if any, tuning by a system administrator. Although these configuration options can be modified by the system administrator, it is generally recommended to keep the default values, allowing SQL Server to automatically tune itself based on run-time conditions.

However, if necessary, the following components can be configured to optimize server performance:

- SQL Server memory
- I/O subsystem
- Microsoft Windows NT® options

Create an empty Agresso Database on MS SQL Server

Below, we give a short overview of things to remember when you create an Agresso database.

- **Database:** You need a database for Agresso data (Agresso).
- **Collation:** We recommend `Latin1_General_CI_AS`, but supports all collations.
- **File group:** If you are going to use File group, please refer to [Agresso Data Segments](#).
- **Authentication:** Select `SQL Server Authentication`. Agresso does not support only Windows Authentication.
- **Language:** Select `English` as language. The language for the LOGIN has to be set to English. Agresso uses English date and time formats in the database. (This does not influence the date/time format you want to see in Agresso.)
- **DB owner:** You have to set the **DB owner** manually to your Agresso login. Start **SQL Server Management Studio** and run the `sp_changedbowner` on your Agresso database.
- **DB owner:** You have to give the DB owner access to a system view:
`grant view server state to <login>`
- **DB owner:** You need to give the DB owner access to administer bulk operations:
`grant administer bulk operations to <login>`
- **Default database:** You have to set default database (Agresso) for your Agresso login.

Database properties for Agresso database

We recommend:

1. Set the recovery Model to `Full`.
2. Use the following settings:
 - Auto update statistics
 - Page verify = `CHECKSUM`
 - Auto create statistics
3. Set the parameter `READ_COMMITTED_SNAPSHOT` on by issuing the command:

```
alter database <dbname> set READ_COMMITTED_SNAPSHOT on
```

This parameter reduces or removes possible problems related to lock-waits and deadlocks.

Optional: AGR_TEMPDB

Creating a separate database for help tables is optional. If the `AGR_TEMPDB` Agresso environment variable is defined, the database specified will be used for help tables. If `AGR_TEMPDB` is not defined (default), then the Agresso database will be used for help tables.

Recommended settings for an agrtemp database:

1. Set the recovery Model to `Simple`.
2. Use the following settings:
 - Auto update statistics
 - Torn page detection

- Auto create statistics
3. Set the parameter READ_COMMITTED_SNAPSHOT on by issuing the command:

```
alter database <dbname> set READ_COMMITTED_SNAPSHOT on
```

This parameter reduces or removes possible problems related to lock-waits and deadlocks.

ODBC in a 64-bit Windows environment

Agresso supports the 64-bit version of Windows. Since Agresso Business Server and Agresso Smart Client is 32-bit, Agresso will run in the 32-bit windows subsystem (syswow64) and use the 32-bit part of the registry (wow6432node).

*Note: When creating the ODBC-datasource in a 64-bit environment, remember to start [odbcad32.exe](#) from the [windows\syswow64](#) directory and **not** the default one.*

Agresso on Oracle

Oracle Client in a 64-bit Windows environment

Agresso supports the 64-bit version of Windows. Since Agresso Business Server and Agresso Smart Client is 32-bit, Agresso will run in the 32-bit windows subsystem (syswow64). As a result of this, Agresso Business Server needs the 32-bit of the Oracle Client to run. 64-bit oracle client is required for Agresso Self Service and web services.

Note: 32-bit and 64-bit of the Oracle Client can be installed side by side.

Easy Connect

The easiest way to connect to oracle from Agresso is to use Oracle Easy Connect when creating an Agresso server or client data source. When Easy Connect is used, tnsnames.ora is not needed.

The connection string used for Oracle Easy Connect must be defined in the following format: <HOST NAME>:<PORT>/SID . You don't need to define port when the default port is used; normally the connection string format will just be: SERVER/SID.

To support use of easy connect, oracle must be configured to include EZCONNECT as naming method (HOSTNAME can also be used). Naming methods can be configured using Oracle Net Manager, or by editing sqlnet.ora manually. The NAMES.DIRECTORY_PATH section in sqlnet.ora must contain EZCONNECT to make easy connect work. If sqlnet.ora don't exist, easy connect will be used.

General Oracle settings

Although many configuration options can be modified by the DBA, it is generally recommended that these options are left with their default values, allowing Oracle to automatically tune itself based on run-time conditions.

If necessary, the following parameters can be configured to optimize performance:

- SGA - Under installation, only set the `sga_max_size` and `sga_target` and let Oracle automatically tune the SGA memory.
- CURSOR_SHARING - We have seen good performance benefits by setting set the `cursor_sharing` to `force` (default: `exact`).

Unicode support

To support Unicode characters in the database, the following settings must also be in place:

- Character set must be set to `AL32UTF8`
- The `NLS_LENGTH_SEMANTICS` parameter must be set to `char`.

Requirement

`AGR_SESSION` view must be created.

You must create a view and a synonym manually in the `SYS` schema and grant select on those to the `PUBLIC` schema.

```
REM ****
REM *** Login as SYS to run this script
REM ****
create or replace view agr_session
as select audsid, terminal, process
from v$session;

create or replace public synonym agr_session for sys.agr_session;

grant select on agr_session to public;
```

! `v$session` should be replaced with `gv$session` when using Oracle RAC.

Create tablespaces and user

Prerequisites

Before you can load tables and data into an Oracle database, you have to:

- Create tablespaces for the Agresso data
- Create an Agresso user in the database
- Grant rights to the user

Procedure

Start SQL*PLUS and login as `SYS/pwd` as `SYSDBA`. Continue as follows:

Set default destination for all data files:

```
alter system set db_create_file_dest='C:\ora_data';
```

Create tablespaces:

```
create bigfile tablespace agrdata datafile size 100m autoextend on next 50m;
create bigfile tablespace agrindex datafile size 100m autoextend on next 50m;
create bigfile tablespace agrblob datafile size 100m autoextend on next 50m;
create bigfile tablespace agrdoc datafile size 100m autoextend on next 50m;
```

```
create bigfile tablespace agrscratch datafile size 100m autoextend on next 50m;
create bigfile temporary tablespace agrtemp tempfile size 100m autoextend on next 50m;
```

! We recommend to set AGR_TMP_LOC environment variable in the Agresso Management Console to the agrscratch tablespace.

Create Agresso user:

```
create user agr56 identified by agresso
default tablespace agrdata
temporary tablespace agrtemp
quota unlimited on agrdata
quota unlimited on agrindex
quota unlimited on agrblob
quota unlimited on agrdoc
quota unlimited on agrscratch;
```

Create new ROLE for Agresso and grant session privileges:

```
create role agresso_role not identified;
grant create session to agresso_role;
grant alter session to agresso_role;
grant create table to agresso_role;
grant create trigger to agresso_role;
grant create view to agresso_role;
grant create procedure to agresso_role;
grant create sequence to agresso_role;
grant create synonym to agresso_role;
grant create any directory to agresso_role;
```

Grant privileges to Agresso user:

```
grant agresso_role to agr56;
```

Notes on Oracle 11.2

If you run Agresso on Oracle 11.2, you need to be aware that Oracle 11.2.0.1 has a serious error that will affect Agresso. This problem has been solved in Oracle 11.2.0.2 and all UNIT Agresso customers on Oracle 11.2 need to make sure that they are on Oracle patch level 11.2.0.2 or higher. Unfortunately Oracle patch level 11.2.0.2 also has an error that affects Agresso. For this error there is a work-around and if you run on 11.2, you need to do the following:

Log in as sysdba and run this query (on Oracle 11.2.0.2):

```
alter system set "_optimizer_false_filter_pred_pullup" = FALSE scope=both;
```

In Oracle 11.2.0.3 this bug is fixed and it is not necessary to define the parameter.

When installing the Oracle 11.2.0.3 client software there is a bug in Oracle installer causing the *OraOLEDB11.dll* not to be registered.

This is solved by registering the Oracle Provider for OLE DB manually:

```
regsvr32 C:\oracle\product\11.2.0\client_x86\bin\OraOLEDB11.dll
```

You can find the oracle version you are running on by running the query:

```
select * from v$version
```

You are now ready to load the template database. Information on how this is done is described in the document [Loading a New Template Database](#)

Server Installation Overview

One installation wizard

With the exception of [Web Help](#), all the UNIT4 Agresso components are handled by one installation wizard. Still, you can install Agresso components on separate servers.

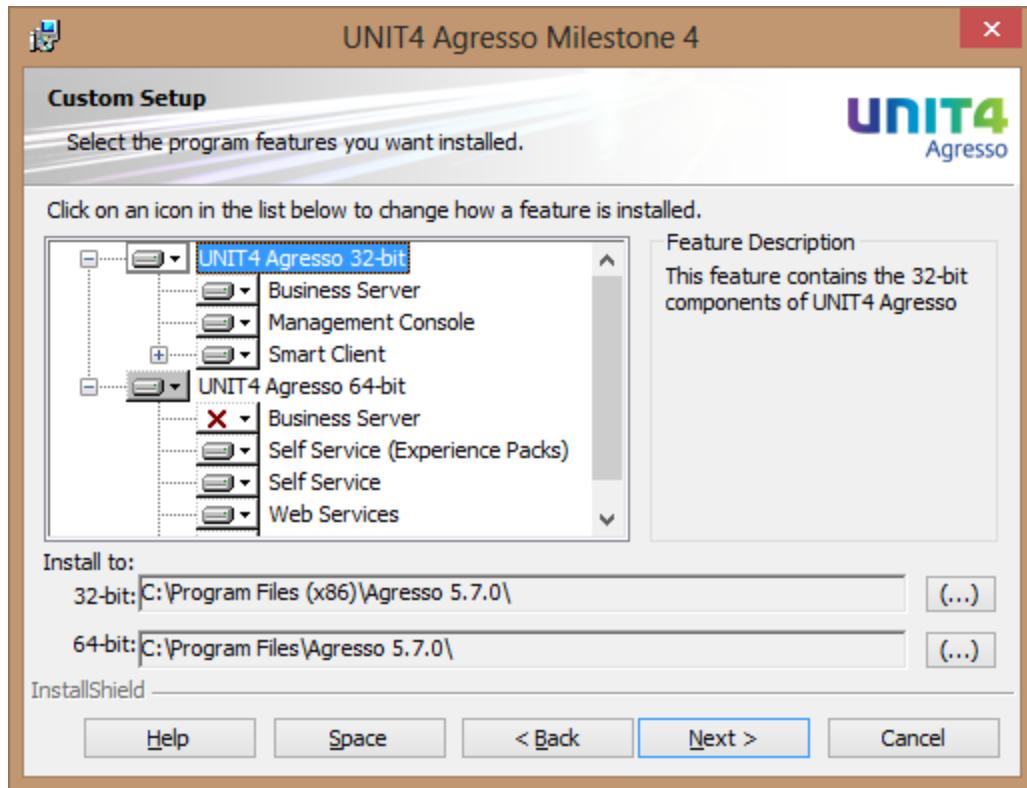
Need for configuration

The UNIT4 Agresso installation works only as a file copier. You will eventually use the **Agresso Management Console** (AMC) to set up and configure the Agresso Server according to your company's needs.

Available components

Although we recommend that you install all components, the **Custom Setup** window gives you the option to de-select certain components.

Custom Setup



The available main components are shortly described below

Component	Installation Directory	Description
UNIT4 Agresso (32-bit)	<32-bit install location>\Bin	Contains the 32-bit Agresso components. Smart Client, Agresso Business Server and Agresso Management Console. Sub components...
UNIT4 Agresso (64-bit)	<64-bit install location>\Bin	Contains the 64-bit Agresso components. Business Server, web applications and PowerShell and WMI providers required to configure the 64-bit components through AMC. Sub components...

64-bit and 32-bit components are installed to separate locations. By default Agresso 32-bit components will be installed to [c:\Program Files \(x86\)\Agresso 5.7.0](c:\Program Files (x86)\Agresso 5.7.0), the 64-bit components will by default be installed to <c:\program Files\Agresso 5.7.0>. You can customize these locations, but you are not able to change the sub-directories for the different components.

Server Step-by-step Installation

Detailed procedure

After booting the Agresso Installation DVD, [SetupAgresso.exe](#) shall start automatically.

Note: If the DVD does not start, you can locate the DVD in [Windows Explorer](#) and double-click on [SetupAgresso.exe](#).

Example: Setup Agresso



Note: Please read the Installation note first.

1. Click on UNIT4 Agresso.

2. Click **Next** until you come to the **License Agreement** window..

3. Make sure that you accept the terms in the license agreement, and click **Next**.

The program automatically extracts information like User Name and Organization from the computer.

4. Select the preferred installation option, and click **Next**.

The Setup Type window is displayed.

5. Select **Custom UNIT4 Agresso**, and click **Next**.

For a standard server installation, you leave the settings as default.

6. Click **Next**.

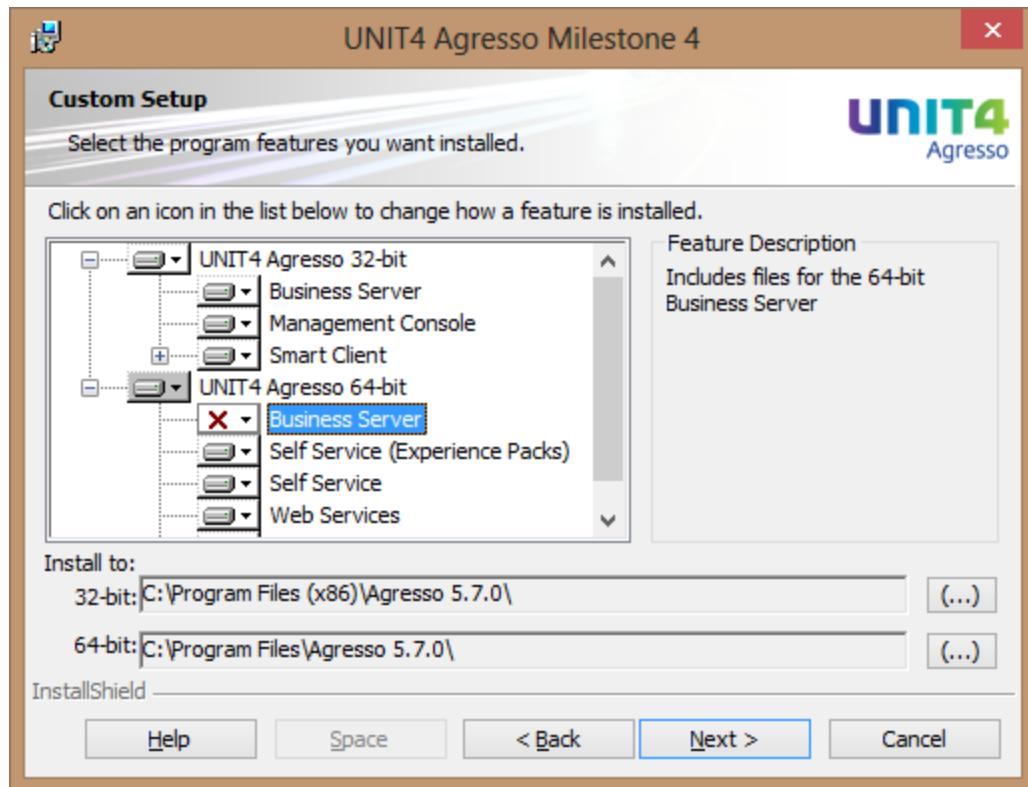
You are now ready to install the program.

7. Click **Install**.

The necessary files are copied. When completed, the installation is finished.

Installing components

Available components



The available components are shortly described below:

Component	Installation Directory	Description
Agresso WMI and Powershell Providers	<64-bit install location>\bin	64-bit WMI Providers required to configure the 64-bit Agresso components Business Server, Self Service and Web Services. The Agresso Powershell Provider is required if you want to configure or maintain the Agresso Business Server, System configuration and Update Manager through PowerShell.
Business Server	<32-bit install location>\Bin	Files required by the 32-bits Agresso Business Server
Business Server	<64-bit install location>\Bin	Files required by the 64-bit Agresso Business Server
Management Console	<32-bit install location>\Bin	Agresso Management Console AMC is required for most configuration tasks. If you need to configure 64-bit Agresso components like Self Service And Web services, you will also need to install the 64-bit WMI Providers. More..
Self Service	<64-bit install location>\Self Service	Files required to set up Agresso Self Service. More...
Self Service (Experience Packs)	<64-bit install location>\Self Service (Experience Packs)	Files required to set up Agresso Self Service (Experience Packs). More...

Smart Client	<32-bit install location>\Bin	Installs the files required to run the Agresso Smart Client. This component is also required when you are going to configure a centrally configured client .
Web Services	<64-bit install location>\Web Services\	Files required to set up the Agresso Web Services. More...

! On a 32-bit OS only the 32-bit components will be available. The Agresso server components requires Windows Server 2008 or later, and will be unavailable in the installation wizard on previous OS versions.

Note on Self Service and Web services

AMC supports creation of multiple Web Applications sharing the file set

AMC and PowerShell allows you to create multiple versions of Agresso Web Applications against the same file set. All Web Applications created in AMC or PowerShell will get a private [web.config](#) identified by the Application Pool Name. See description of [Web Sites](#).

Installation description

General: The Self Service and **Agresso Management Console** components are required to setup and configure Ageresso Self Service. The Self Service component contains all the files necessary to setup and configure the Agresso Web Server using **Agresso Management Console**. See also the [configuration section](#).

! Configuring the web application manually without using the Management Console is not recommended and not supported. This also applies for the web services.

ASP.NET Extension: Agresso Self Service and the Agresso Web Services use ASP.NET. ASP.NET web service extension is installed and enabled on the Internet Information Server when you set up Self Service or one of the web services.

.NET Framework: Installing .NET Framework will by default install ASP.NET.

Configuration

Please refer to the [configuration section](#).

Smart Client

Smart Client installation only

When you need to install the Smart Client on local computers - or if you need to reinstall it on the server - you can select Smart Client Installation from the installation DVD.

Smart Client Installation will simply install the relevant client components from the Full installation option, without asking you to de-select all the other components.

Server note: If you need to create centrally configured clients from the Agresso Management Console, you will need the Smart client on the server.

Step-by-step installation

After booting the Agresso Installation DVD, [SetupAgresso.exe](#) shall start automatically.

Note: If the DVD does not start, you can locate the DVD in **Windows Explorer** and double-click on [SetupAgresso.exe](#).

 Example: Setup Agresso



Note: Please read the Installation note first.

1. Click on UNIT4 Agresso.
2. Click **Next** until you come to the **License Agreement** window.
3. Make sure that you accept the terms in the license agreement, and click **Next**.

The program automatically extracts information like **User Name** and **Organization** from the computer.

4. Select the preferred installation option, and click **Next**.

The **Setup Type** window is displayed.

5. Select Agresso Smart Client, and click **Next**.

The Smart Client is now ready to be installed.

6. Click **Install**.

The necessary files are copied to the computer and the installation is completed.

Related topics

[Data source setup](#)

[Agresso in a Terminal Server \(Citrix\) Environment](#)

Loading a New Template Database

First time installation

General

When you install Agresso for the first time, you need to load the Agresso table definitions and template data into your database.

Some of the principles mentioned below, also apply if you need to copy Agresso data in or out of your database for other reasons.

Data distribution options

Storage spaces

There are approximately a thousand tables in UNIT4 Agresso. To be able to distribute these tables to different disk locations, Agresso uses a set of logical storage spaces.

These will be created as *tablespaces* in Oracle, and *file groups* in SQL Server. For more information see [Agresso Data Segments](#).

Copy options

When copying data into Agresso, make sure the objects are stored in the correct data space. This requires using the options **-T** and **-I** with the Agresso Copy programs for tables and indexes respectively. [The Agresso Copy programs](#) are described in detail in the Appendix.

Copying of the Agresso tables with **-T** and **-I** options assumes that the Agresso standard data spaces already are created, and the tables and indexes will be correctly distributed.

Note: It is also possible to install all objects in a single logical space. This may be a good idea for a very small installation where data distribution is not essential. We do recommend, however, that you use the [Database Tools](#) in the [Agresso Management Console](#).

Load tables into the database

To be able to start the Agresso application, data must be loaded into the database.

The file *new.zip* is located in the Agresso database script directory: *.\\Agresso 5.7.0\\DatabaseScript\\DbLoad*

Load procedure (1) using wizard

Do as follows:

1. Unzip *new.zip* (if not done already!).
2. Use the Database Copy utility from the [Agresso Management Console](#) (There is a log-in option on the right-click menu of the copy-in tool).

Load procedure (2) using copyora or copyms

Do as follows:

1. Unzip *new.zip* (if not done already!).
2. Copy the tables into the database by running the copy programs with the following parameters:

- **Oracle:**

```
copyora.exe -din -v -U<user> -P<password> -S<dbserver> "-f<path to folder containing files.lst>" -z -T -I -a5000 > copy.log
```

- **SQL Server:**

Unicode support: Note that the `u` flag must be set if your database shall support Unicode (shown as and optional `[-u]` in the command example below):

```
copyms.exe -din [-u] -v -z -U<user> -P<password> -Dbname -S<datasource> "-f<path to folder containing files.lst>" -T -I > copy.log
```

After the command has completed, check the file `copy.log` for error messages. The copy programs is located in the bin-folder of the x86 installation.

Re-creating database views

To make sure all database views in Agresso are correct, you may want to re-create the views. Views can be re-created from the Copy In node in AMC. To re-create views, locate the Copy In node in under Database Tools, right-click and select All Tasks | Recreate views.

Database Tools | Database Copy | Copy In | All Tasks | Recreate views ...

You may also re-create views using the copy tools from command line:

create an empty directory with an empty file called `files.lst`. In this new directory, run
`copy<db>.exe -din <logon information> -v`.

Create and re-create triggers and procedures

To make sure all database functions and procedures in Agresso are correct, you can re-create them. To re-create views, locate the Copy In node in under Database Tools, right-click and select Re-create triggers/procedures.

Database Tools | Database Copy | Copy In | All Tasks | Re-create triggers/procedures...

You can also re-create procedures from command line using the Agresso copy tools. To do this, you must create an empty directory with an empty file called `files.lst`. In this new directory, run

```
copy<db>.exe -din <logon information> -z
```

Report Engine Installation

It's recommended that the latest version of report engine is installed. The latest Agresso Report Engine version can be downloaded from the Agresso Updates page: https://abwupdates.agresso.com/updates.aspx?product_area=REP

For updated information on how to install and configure the latest version of Agresso Report Engine, please refer to the Installation Notes document distributed with Report Engine.

WEB HELP INSTALLATION OVERVIEW

Options

The help system comes in two versions:

1. Web based help (WebHelp), where the help files (html-files) are located on a web server and uses a Web browser for display. There is one Web Help installation for Agresso Smart Client (SCHELP), one for Agresso Self Service (SSAHELP) and one for Self Service With Experience Packs (X2HELP). You can access these help files from many different UNIT4 Agresso installations as long as a reference to the WebHelp server is added to the database through a configuration in **Agresso Management Console**.

To install WebHelp, see [WebHelp Installation](#).

After installation, you need to publish the WebHelp system using AMC. See [WebHelp Configuration](#).

2. Compiled help (HTMLHelp), where the help files (the same html-files as for WebHelp) are compiled and packed into CHM files. These are installed on the network share during the Agresso installation. The compiled help system is faster than Web based help, but can be accessed from the Smart Client only.
HTMLHelp does not require a specific installation, since the CHM files are copied to the correct directory during the Agresso installation.
Configuration of the Agresso HTMLHelp is described in HTMLHelp Configuration.

Language and installation folders

Language alternatives

Agresso offers the following language versions for Web Help:

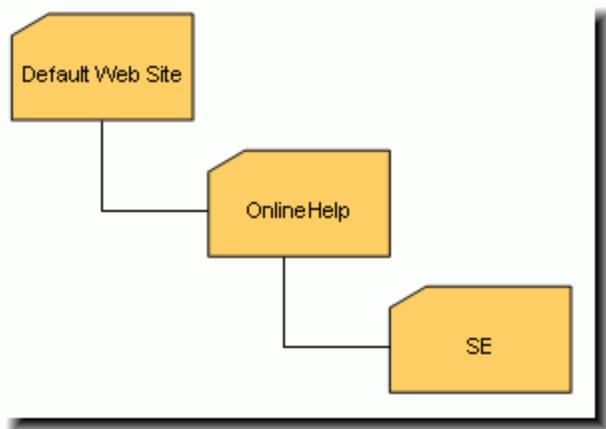
Installation folders

During installation, you must select the web site to use, and give a name for the Web Help entry point (web application).

If you have made these selections during installation:

- Language: `SE`
- Web site: `Default Web Site`
- Web application name: `SCHELP`

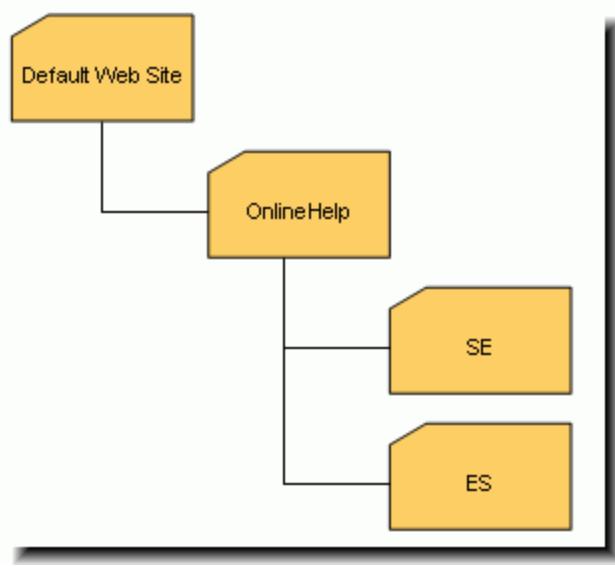
the following structure will be created in IIS:



The help files will be available in the SE folder.

Copying language folders

In cases where you must have several language codes available - with the same set of default help files - you can simply copy the folder containing the default files and rename it to the desired language code.



WebHelp Installation

Wizard

You will use the [Web Help for Smart Client](#) and [Web Help for Self Service](#) link on the installation DVD to install the Help system.

Install WebHelp

1. Click **Next** until you reach the **Customer Information** dialog in the installation.

2 Do as follows:

- Enter correct **User Name** and **Organisation** (or accept default values)
- Select **Anyone who uses this computer**
- Click **Next**

Note: If you selected default help language in step 1, you will now have to select the language code for the installation. The selected language code will give name to the folder where the help files will be installed.

3. Select language code and click **Next**

If this is your first installation, the **Web Server Setup** dialog will display. Otherwise, the already selected setup will be used, and you will only have to complete the installation - from step 6 below.

4. Select a **Web Site** and type in the name you want for the Online Help application. Then click **Next** to bring up the **Destination Folder** dialog.

5. Select the folder (in the file system) where you want your Online Help (root) catalog to reside, and click **Next**. The selected folder will be used as the entry point catalog for the language code folders in the Online Help installation.

6. Click **Install** to start the installation.

Configuration

When Agresso WebHelp is installed you must use the **Agresso Management Console** to configure the help system. See [Help Configuration](#).

Other deployment options

If you want extract the help files from the installation without installing the product, this can be done using the following command:

```
msiexec /a "<path to msi-file>" /qb TARGETDIR="C:\WebHelpExtracted"
```

In this sample the files will be extracted to `C:\WebHelpExtracted\Agresso Web Help\SSAHelp\XX`. The XX folder can then be renamed to the correct language code, and a web application can be created manually if needed.

Logging and Troubleshooting

Logging

Installation logging can be turned on by manipulating the following registry value:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\Installer\Logging
```

Setting this value to `voicewarmup` will turn on verbose logging of the installation. Windows Installer log files are generated in the local temp directory of the user running the installation.

You can also specify logging as a parameter in the `msiexec`. The sample below installs UNIT4 Agresso and creates a log file at a custom directory:

```
msiexec /I <path to msi package> /l*v c:\MyLogFile.log
```

If you experience problems with the UNIT4 Agressoinstallation, please run the installation with verbose logging. The log file can be used for analyzing the problem.

Silent Installation

Install from command prompt

Windows Installer has an option to run a silent installation.

A silent installation means running the installation in the background, with no user interface. All selections usually performed through the GUI during the installation process, must now be set from the command line.

Silent installation

Complete UNIT4 Agresso

```
msiexec.exe /i "UNIT4 Agresso (64-bit).msi" /quiet ALLUSERS=1 INSTALLLEVEL=150 INSTALLDIR="C:\Agresso_x64\" INSTALLDIRX86="C:\Agresso_x86\" /l*v %TEMP%\AgrCompleteInst.log
```

Smart Client

```
msiexec.exe /i "UNIT4 Agresso\AGR x86 Setup\UNIT4 Agresso (32-bit).msi" /quiet ALLUSERS=1 INSTALL-LEVEL=75 /l*v %TEMP%\AgrSmartClientInst.log
```

Agresso Demo

```
C:\Windows\system32\WindowsPowerShell\v1.0\powershell -executionPolicy Unrestricted AgrDemoSetup.ps1 -Silent $true
```

General syntax for silent installation

```
msiexec /i <path to msi package> /quiet
```

For more options see microsofts documentation of standard Windows Installer [commandline options](#).

Configuring Agresso

Agresso Management Console (AMC)

This section of Technical guidelines describes the functionality of the [Agresso Management Console](#), and how you use the console to set up and maintain your system configuration(s).

Scripting with Agresso.Management.PowerShell

To facilitate the configuration and setup of UNIT4 Agresso installations, Agresso has created a set of Windows PowerShell objects with new and extended cmdlets. It comes as a PowerShell snap-in: *Agresso.Management.PowerShell*.

See [Agresso.Management.PowerShell Reference](#).

Introduction to Agresso Management Console

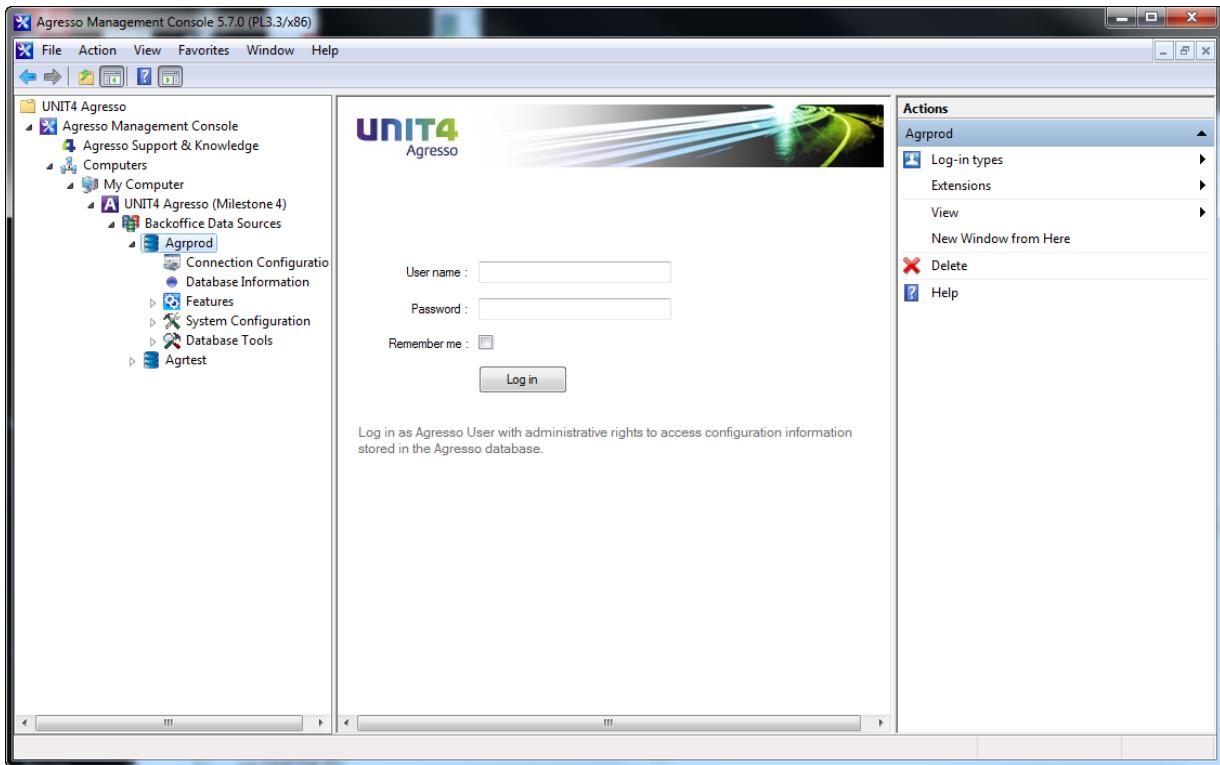
Installation, Configuration and Maintenance

The **Agresso Management Console**, AMC, is the main tool for managing your Agresso installation(s). AMC complies with international standards for software management, and is based on:

- **Microsoft Management Console** (MMC), which defines a standard user interface for managing applications, and
- **Windows Management Instrumentation** (WMI), which is a standard technology for accessing management information in an enterprise environment.

Nodes and panes

Click to see an example of the AMC interface



In the left pane you can navigate among servers and installed Agresso software. An item in the left pane is referred to as a *node*. When you select a node, the *main view* (in the middle) presents details about the selected node, while the right pane offers the available *actions*. The actions are also available via a context sensitive menu for the selected node (right-click).

Remote administration

AMC and Powershell are installed as part of the server software but can also be run on a workstation to manage remote computers.

Enable remote administration: To enable remote administration, you must install AMC/Agresso Powershell on the workstation, and - in AMC - add the remote server(s) to the Computers node.

AMC will now allow you to connect to them through WMI. See [Enable remote administration](#).

Tables

To get an overview of database tables used by the management console, see [Tables used by AMC](#).

Enable Remote Administration

Three tasks

When you enable remote administration, you can basically perform all configuration tasks for a server, from your own office computer.

Before you can do that, you must install **AMC** on your local computer, and make sure that there are no security restraints. The complete process consists of the following tasks:

1. Install AMC locally.
2. Solve security issues.
3. Enable remote administration.

Install AMC locally

To install AMC locally, you need the Agresso installation DVD. Select [UNIT4 Agresso](#) installation and de-select the components you don't need.

Solve security issues

Remote administration through the **Agresso Management Console** is based on integrated security. You will need administrator rights to be able to do any changes to the remote computer just as you need administrator rights to do any changes to your local computer. This is governed by AMC's use of Windows Management Instrumentation (WMI).

Procedure

To set up remote access right to the WMI service, you must first open the **Computer Management** Console - on the computer (server) you want remote access to - found under [Control Panel/Administrative Tools](#).

1. Activate the **WMI Control Properties** window:

- a. Expand the **Services and Applications** node,
- b. right-click the **WMI Control node**,
- c. and select **Properties**.



2. Activate the **Security for Root** window:

- a. Activate the **Security property** tab,
- b. select the Root node,
- c. and click the **Security** button

Security for Root



3. Make sure that your user, or a group of users where you belong, has the **Remote Enable** permission set to **Allow**.

4. Save your work.

If no errors occurred, your access rights will now be in order.

Enable Remote Administration

Two tasks

To enable remote administration, you must

- 1) add the remote computer to your local AMC, and then
- 2) connect to the remote computer.

Add the remote computer to AMC

To enable remote administration of your Agresso installation, do as follows:

- 1.** Open AMC on local computer.
- 2.** Right click the **Computers** node and select **Add**. A wizard will guide you through the setup process.

As a result, the remote computer will appear as a new node under **Computers**.

Connect to the remote computer

When you click on the new computer node, you are prompted to log in.

The Agresso installation can now be managed from your local computer.

Backoffice Data Sources

Data source

The **BackOffice Data Sources** node in **AMC** is used as an entry point for the server installation (which will connect to this data source).

Logically, a data source represents an Agresso database (an Agresso installation), and you will always use the data source as an entry point when you configure the installation.

A data source is uniquely identified by a name (e.g. *Agitest*), and holds all necessary properties needed by the Agresso server software to connect to a data base. The data source name is the key to all configuration of a running Agresso system.

AMC allows you to create and maintain an unlimited set of data sources (Agresso systems). Typically, we find two data sources on a local site, one for test and one for production. Hosting environments may have tens (or hundreds) of data sources, representing a multitude of Agresso systems.

Creating a data source

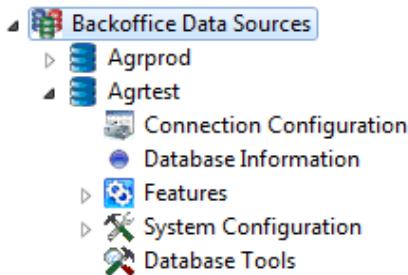
! The **New Data Source Wizard** will need the name (or IP address) of the database server, as well as the name of the database. You must have this information ready before you start.

Procedure

The procedure is simple:

Select the **New** action from the **Backoffice Data Sources** node and follow instructions.

The new data source - here called **Agitest** - will appear as a new node, with several child nodes.



The new nodes

The new nodes are shortly described below:

- The new node for the data source (**Agitest** above): Allows you to log in to Agresso (as administrative user). Login is required to get access to information and settings stored in the database. See [below](#).
- [Connection Configuration](#) - with basic connection information. Can be updated..
- [Database information](#) - gives you connection information and details about the database.
- [Features](#) - for configuration of server features, like Business server(s), Web services/applications and Centrally configured clients.

- [System Configuration](#) - for Agresso configuration (mail, logging etc)
- [Database Tools](#) - with options for database copy, updates etc.

Log-in types

Default authenticator

Agresso is delivered with a set of standard authenticators, available under the Authentication Setup node in AMC (System Configuration), and also available under System administration in the Smart client.

You will need to select a default authenticator for **Management**, valid for both Agresso Management Console and Agresso Management PowerShell.

We recommend an authenticator without a user interface, for example Windows Authentication.

Default log-in type

AMC allows you to select one of three authentication options, as the default log-in type:

- **Agresso Log-in:** Requires that you log in as an administrative Agresso user, i.e. a user with *Administrator* checked ([User master file](#), Smart client). Will always require user name and password.
- **Integrated Log-in:** Single sign-on (supported by Windows Authentication). Recommended. SSO user must be identified as such in [User master file](#).
- **Database Log-in:** Offers two options. You can either log in as a specific database user, with user name and password, or you can use Windows authentication, and thus get hold of the database user via the domain user.

AMC will always remember the latest log-in type.

Connection Configuration

Purpose

Allows you to view and edit basic information about the database connection:

- Database server.
- Database.
- User.
- Parameter settings.

Main actions

The relevant actions are:

- **Save** (if you have made changes).
- **Test Connection**. Will test the connection after property changes, before save.

Database Information

Prerequisite

Unless you are logged in to Agresso, no information is available.

Purpose

Displays information about the database used by Agresso:

- the connection,
- the database system (RDBMS),
- the database,
- system data updates (if any). You will find the name of all update packages installed via [Update Manager](#) since last Milestone - for information only.

Main actions

None.

Features

Purpose

Shows what you have installed on the server. You must explicitly activate the relevant options - for the current data source.

Child nodes

If everything is installed, you will get access to the following nodes:

- [Business Server\(s\)](#), both 32 and 64 bit if required.
- [Self-Hosted Services](#).
- [Web Applications](#).
- [Centrally Configured Clients](#).

Main actions

None. You will always go via the child nodes.

Support for both 32-bit and 64-bit Business Servers

A 32-bit and 64-bit Business Server can fully coexist and run on the same server environment without the two services conflicting. AMC and Agresso Management PowerShell will detect and support such a scenario.

The server controllers (server queues) are assigned to the Business Server instances based on the server name and architecture type. This means that two Business Server instances can run on the same server if they have different architectures.

You can for example set up a dedicated 64-bit Business Server to serve one set of the server controllers such as TPS, DWS, and Workflow; and a separate 32-bit Business Server to serve the report queues.

Business Server

You will get a Business Server must be initialised.

Purpose

Allows you to manage the Agresso Business server - i.e. the Windows service handling all server operations.

- Start / stop the service.
- Select start-up option.
- Select [Log-on alternative](#).
- Set path to the service's .exe file.

In addition, you get shortcuts to a series of configuration options, otherwise available via child nodes' child nodes:

- Logging
- Native Connection
- Activity Monitoring
- Server Controllers (Reports, Times Processes, Service Processes)
- Server Printing
- Other Tasks (Environment variables, Mail)

Main actions

Start / Stop (the service)

Log-on alternatives

You can select between a

- Built-in System Account:
 - Local Service:
 - Local System:
 - Network Service
- Specific account. In this case you will search for the account and must enter user name and password.

Report and Process Control

Prerequisite

Unless you are logged in to Agresso, no information is available.

Purpose

Gives you access to setup and configuration screens for three types of controllers:

- [Report Queues](#)
- [Timed Processes](#)
- [Service Processes](#)

Main actions

Add Report and Process Controller. Allows you to add controllers of all types - in one go.

Tables

Information about reports and processes are stored in the table [aagserverqueue](#).

Architecture

If you have initialized both x86 and x64 versions of the Business Server, you can change the architecture for each individual process (after they have been added).

If you select a certain process setup for a business server and change the default architecture, the process will be moved to the other business server (node).

Note that some custom reports may be written for x86 only.

See also [Running Processes on Multiple Computers](#).

Report Queues

Prerequisite

You must be logged on.

Purpose

Manage queues. Each new queue will appear as a child node.

Main actions

- **New...** - add a predefined queue. Only one is available: DEFAULT.
- **New User Defined** - add a new serial or parallel queue

Main actions on child nodes

- **Disable/ Enable**
- **Set as Default** - will set the current queue as default for all Agresso reports using Report queues.
- **Save** - saves any new settings.
- **Reload** - reloads the default settings.
- **Delete** - removes the selected queue (and node).

About Report queues

A report queue consists of one or more jobs that are ordered from an Agresso client (by a user). Normally, the user will use report parameters to provide the correct data for a job, and the Business Server will retrieve these data from the database before running the job.

There are two types of report queues:

- **Serial queue:** A queue running one report at a time. This assures that reports ordered in a specific order also will run in the same order.
- **Parallel queue:** A queue that can run two or more reports simultaneously. Time-consuming reports will thereby not completely block other reports, given available time slots.

You must, however, ensure that reports running simultaneously are designed to do so. Use the report/server queue filter available in the Agresso Smart Client to do this.

Example: It is safe to put all reports marked as listing only (not update) into the same parallel report queue.

DEFAULT

Prerequisite

Unless you are logged in to Agresso, no information is available.

Purpose

The default report queue.

Properties

You can change the **Type** (Serial / parallel - and number of parallel slots), as well as [Architecture](#).

The **Description** property is displayed as report queue name for the users.

Main child nodes

Activity Monitoring

Gives you the same information as the general [Activity Monitoring](#) node, but filtered by the current report queue only.

Logging

Allows you to set [logging options](#) for each queue.

Recreate: If you select the **Recreate** action, all options will be reset to the default settings for Report Queue.

Environment Variables

Allows you to override [global settings](#).

Mail Configuration

Allows you to override [mail setup for Business server](#).

Instrumentation

General: Sometimes, when a report takes a long time (for example that it never finishes), the server needs to take some action. When we enable instrumentation for a report, we can define a rule that identifies a potential error and a certain error, as well as the actions to take.

Logging: All incidents are written to the Event Log Destination with special event ids. See Actions below

Actions: The following actions are defined:

- **Log potential hang:** If the process has not finished after a number of minutes, this will be written to the log as a warning, with event id = 2039
- **Log process as hanging:** If the process has not finished after a (higher) number of minutes, an error will be logged. Severity = Error. Event id = 2040.
- **Auto terminate:** If set, and the process is logged as hanging, the Business server will terminate the process, and write this to the log. Severity=Error. Event id=2042 or 2041 (couldn't stop the process).

Without instrumentation, no actions will be taken, and a report may hang for days.

Alerts

Allows you to override default settings for [Alerts](#).

Main actions

Disable / **Enable** - dependent on the current state.

Timed processes

Prerequisite

Unless you are logged in to Agresso, no information is available.

Purpose

Set up and configure timed processes.

Main actions

New... - add a predefined process.

New User Defined - add a process defined by yourself.

About Timed Processes

A timed process consists of one or more jobs that are predefined by the Agresso Business Server, and which will be processed by an Agresso Processing Server program.

A timed process runs (on the server!) under control of the Agresso Business server, at defined intervals. Normally, a process will parse certain tables and check if the necessary conditions for further processing are met. If so, the process will run the job(s).

Agresso processes

The Agresso Business Server controls the following Timed Processes, each with its own settings:

Scheduler

The Scheduler is used to start and stop defined jobs - according to a specified schedule.

Invoice Matching Server

The Invoice Matching Service is a server component to match purchase invoice transactions to purchase orders. Registered invoices without missing goods received are matched by the IMS when receipt is done. There should never be more than one IMS running per database.

ACRLS Server (AAS)

The ACRLS Server uses the changed data from 'Amendment logging' triggers to update shadow tables, move the data to a history table, and update the import table for IntellAgent (this functionality is not utilized by IntellAgent at the moment).

Transaction Processing Server (TPS)

The Transaction Processing Server is a server component configured to perform transaction processing. There should never be more than one TPS running per database. The server runs at configurable intervals.

Logistics Invoice Processing Server (ALGIPS)

The Logistics Invoice Processing Server is a server component configured to perform invoice processing. There should never be more than one ALGIPS running per database. The server runs at configurable intervals.

Logistics Stock Processing Server (ALGSPS)

The Logistics Invoice Processing Server is a server component configured to perform stock processing. There should never be more than one ALGSPS running per database. The server runs at configurable intervals.

Data Warehouse Server (DWS)

The Data Warehouse Server is a server that does all collection of transactions and updating of balance tables. There should never be more than one DWS running per database. The server runs at configurable intervals.

IntellAgent Processing Server (AINAPS)

AINAPS is processing all events set up in the IntellAgent module. It will also process the alerts generated by workflow, as well as delay steps set up in the workflow processes.

Tuning: If required, you can set up several AINAPS server queues to share the workload, and to assign events to the different queues.

The execution frequency of AINAPS will determine how often an event can be processed (regardless of the schedule defined on each event, the event can not be processed more frequently than the AINAPS is running), so this needs to be considered when setting the frequency. Normally it should run quite frequently (every 3-5 minutes), so that IntellAgent will be able to rapidly respond to changes.

Note: Workflow alerts will only be affected by the run frequency of the AINAPS if the alerts are aggregated (in the Alert setup screen). If not, the workflow service will start the AINAPS immediately.

Message Server (AMS)

The Message Server (AMS) is responsible for processing all messages in the acrmESSAGEQUEUEHEAD and acrmESSAGEQUEUEDET tables. This includes sending new messages to the interfaces defined in AMC, receiving delivery status for sent messages and posting this back to the tables. It will also re-send messages that previously have failed. Currently, AMS is only used for SMS messages generated by IntellAgent. AMS is also used for processing mails in the Agresso Mail Queue.

AMS should be set up to run frequently (every 3-5 minutes) to ensure that messages are rapidly delivered, and that it frequently will try to resend messages that previously have failed.

The AINAPS will order the AMS directly, and messages from IntellAgent are therefore not affected by the execution frequency, except for retries for failed messages.

RESRATE Server (ARS)

The Agresso RESRATE Server is vital to all HRMS/Payroll/Project customers.

ARS is responsible for producing and maintaining resource rates stored at the lowest possible level (by value reference, by resource, by dates) in the new table ahsresrate.

The purpose is to avoid the heavy workload of processing the rate-hierarchy/rate-setup-logic each time a rate is retrieved. Instead the rates are pre-processed by the RESRATE Server and made directly available on ahsresrate (one query away). The RESRATE server is monitoring all rate-relevant changes made in the system and synchronizing ahsresrate.

ARS should be set up to run frequently (every 3 minutes is suggested) to avoid long delays before changes are processed and rates updated (for example when a resource's pay step is changed in HS01). The ARS will stop immediately if no rows are found in the ahsrestrigger table (so the overhead of running the RESRATE server is often only one query against ahsrestrigger).

(Timed process)

Prerequisite

Unless you are logged in to Agresso, no information is available.

Purpose

You will get one node (of this type) for each added process.

Properties

You can change the **Execution Timer** (number of minutes), as well as [Architecture](#).

Main child nodes

Activity Monitoring

Gives you the same information as the general [Activity Monitoring](#) node, but filtered by the current report queue only.

Logging

Allows you to set [logging options](#) for each queue.

Recreate: If you select the **Recreate** action, all options will be reset to the default settings for Report Queue.

Environment Variables

Allows you to override [global settings](#).

Mail Configuration

Allows you to override [mail setup for Business server](#).

Instrumentation

General: Sometimes, when the process takes a long time (for example that it never finishes), the server needs to take some action. When we enable instrumentation for the process, we can define a rule that identifies a potential error and a certain error, as well as the actions to take.

Logging: All incidents are written to the Event Log Destination with special event ids. See Actions below

Actions: The following actions are defined:

- **Log potential hang:** If the process has not finished after a number of minutes, this will be written to the log as a warning, with event id = 2039
- **Log process as hanging:** If the process has not finished after a (higher) number of minutes, an error will be logged. Severity = Error. Event id = 2040.
- **Auto terminate:** If set, and the process is logged as hanging, the Business server will terminate the process, and write this to the log. Severity=Error. Event id=2042 or 2041 (couldn't stop the process).

Without instrumentation, no actions will be taken, and a process may hang for days.

Alerts

Allows you to override default settings for [Alerts](#).

Main actions

Disable / **Enable** - dependent on the current state.

Service Processes

Prerequisite

Unless you are logged in to Agresso, no information is available.

Purpose

Manage service processes.

Main actions

New... - add a predefined process.

New User Defined - add a process defined by yourself.

Service Processes in Agresso

Currently, there are two available service processes:

-  Workflow

The Workflow service runs when the Business server runs, and manages all the Agresso workflow tasks. The workflow service is often referred to as the *Workflow engine*.

-  Database Logging Service

Reads a message queue and inserts rows into table `acrlog`. Can be used with the Message Queue logging destination. Only enable this server queue if you are going to use the Message Queue for logging.

(Workflow and DbLogService)

Prerequisite

Unless you are logged in to Agresso, no information is available.

Purpose

You will get one node (of this type) for each added process. Currently, there are two service processes:

- **Workflow** (the workflow engine)
- **DbLogService** - takes information from a message queue and updates a database table.

Properties

You can change the [Architecture](#).

(DbLogService only: The **Process Parameters** property allows you to change the message queue to read from (if there are more than one. See [Reference manual - Logging](#)).

Main child nodes

Activity Monitoring

Gives you the same information as the general [Activity Monitoring](#) node, but filtered by the current report queue only.

Logging

Allows you to set [logging options](#) for each queue.

Recreate: If you select the [Recreate](#) action, all options will be reset to the default settings for Report Queue.

Environment Variables

Allows you to override [global settings](#).

Mail Configuration

Allows you to override [mail setup for Business server](#).

Alerts

Allows you to override default settings for [Alerts](#).

Main actions

Disable / **Enable** - dependent on the current state.

Running Processes on Multiple Computers

Divide the workload

One way to control the processing load for large installations is to divide the server queues on two or more computers.

To set up a second computer, do as follows:

1. Install the Agresso Server software on a second computer and create a Backoffice Data Source connecting to the Agresso database.
2. Initialize the Business Server Environment on the computer.

The aagserverqueue table: The column `server_name` in `aagserverqueue` holds the name of the server where the queues are supposed to run. The **Server Queues** node in AMC will list all server queues configured for a given database. The name of the remote computer running the queues is found in the column `Host`.

The settings stored in the registry of the remote computer will not be directly configurable. This would have required AMC to hold a remote WMI connection to the host computer.

To configure these remote settings, you can simply add the second computer to AMC:

1. Locate, and then right-click on the **Computers** node in AMC.
2. Select **All Tasks | Add/Remove Computers**

Now you can manage the remote server queues from here.

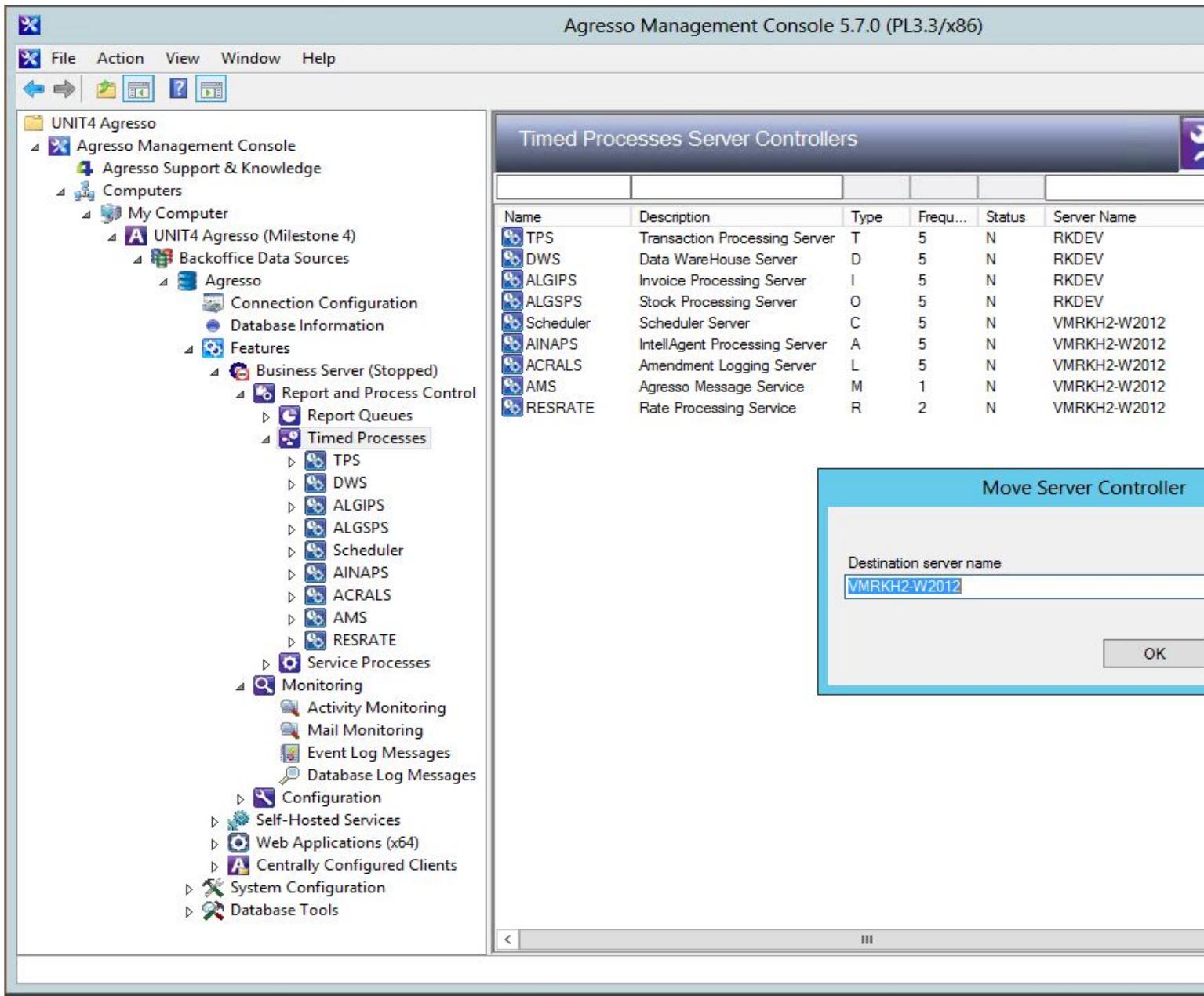
Moving a server queue from one computer to another

! We recommend to stop the services before moving a queue to avoid a situation where two services are serving the same queue, although a service will normally discover that a new queue has been added to the list.

When the Business Server service is started, it always checks the table `aagserverqueue` to find all server queues assigned to it. To move a server queue from one computer to another, follow this procedure:

1. Right-click on the server queue node you want to move (in AMC) and select **All Tasks | Move Queue**.

Example: Move Queue



This will open the **Server Name** dialog.

2. Enter the name of the computer you want the queue to be moved to and click **OK**.

Note: This will not move any of the locally overridden environment variables, as they might be dependent of local folder locations and configuration. If the queue requires individual settings, these must be re-configured at the new host computer after the move is completed.

Monitoring

Purpose

Provides access to four areas:

- [Activity Monitoring](#)
- [Mail Monitoring](#)
- [Event Log Messages](#)
- [Database Log Messages](#)

Main actions

None.

Activity Monitoring

Prerequisite

Unless you are logged in to Agresso, no information is available.

Purpose

Displays information about current or historical Report and Process Controls executed on one or more servers.

Main actions

Report queues

Display log (when report has completed)

Rerun (when report has completed)

Kill (when report is executed)

Timed Processes

Hold (Prevent scheduled process from executing)

Run (Run scheduled process now)

Service Processes

Kill (Abort process)

Display Log (display service log)

Mail Monitoring

Prerequisite

Unless you are logged in to Agresso, no information is available. The [Agresso Mail Queue](#) must be configured and used.

Purpose

Displays information about Sent /Unsent mail messages.

Main actions

Edit (Edit a mail in queue)

Delete (Delete a mail in queue)

Hold /Send (Hold or send a mail in the queue)

Event Log Messages

Prerequisite

An Agresso Event Log destination (Trace listener) must be configured.

Purpose

Displays Agresso event log messages.

Main actions

Property (Show event log details)

Database Log Messages

Prerequisite

Unless you are logged in to Agresso, no information is available.

Purpose

Displays information from the database log.

Main actions

None.

Configuration

Prerequisite

Unless you are logged in to Agresso, no information is available.

Purpose

Gives you access to child nodes representing the general configuration areas for the Business Server:

- [ODBC Connection](#)
- [Environment Variables](#)
- [Clean-up Routines](#)
- [Blob Management](#)
- [Temporary Tables](#)
- [Mail Integration](#)
- [Advanced Logging](#)
- [Logging](#)
- [Printers](#)
- [Mail Queue](#)
- [Alerts](#)

Main actions

None.

ODBC Configuration

Prerequisites

Note the following:

- Relevant for MSSQL databases only.
- Unless you are logged in to Agresso, no information is available.

Purpose

Allows you to select - or create a new - ODBC data source for the Business server processes.

When the business server is initialized, a new ODBC data source are created automatically. If you need to change it after you have run the initialization wizard, you do this here.

Main action

New - a wizard will guide you through the setup.

Environment Variables

Purpose

Displays the environment variables set when the Business server was initialized- with their current value. Some of these will be changed during other configuration tasks, but can be Reset to their original value, if required.

Main actions

Property: When you select a variable, you can change the current value.

Add. Adds a new variable., for example needed by a third party application.

Reset. Allows you to add missing (previously deleted), or reset all default (Agresso) variables to their original value.

Tools. Gives you three options for quick updating of related environment variables:

- **Set Output Locations:** Allows you to change the root folder for all server output.
- **Set Run-time Locations:** Allows you to change default folder for binary files (e.g. AGRESSO_EXE).
- **Set Resource Locations:** Allows you to change root folder for server resources.

Environment variables

Below, we give a short description of all environment variables. Note that all of them are used by the Milestone 4 version.

Folder locations

The following environment variables will be created and set to the following locations. <path> refers to the Agresso installation path.: :

Environment variable	Default path	Description
AGRESSO_EXE	<path>\bin	Folder for Business server binary files (should always be .\bin, so keep the default value)
AGRESSO_REPORT	<path>\Report Writer\	Folder for Agresso report writer and Agresso Excelerator layout files
AGRESSO_COM	<path>\Command Files\	Folder for command files used for printing, file handling, etc
AGRESSO_CUSTOM	<path>\Customised Reports\	Folder for customised report layout files
AGRESSO_STYLE-SHEET	<path>\Stylesheets\	Folder for XML stylesheets
AGRESSO_PRINT	<path>\<datasource name>\Report Results\	Folder for report output files
AGRESSO_LOG	<path>\<datasource name>\Server Logging\	Folder for report and process logging
AGRESSO_EXPORT	<path>\<datasource name>\Data Export\	Folder for exported data files
AGRESSO_OCR	<path>\<datasource name>\OCR Export\	Folder for OCR export files
AGRESSO_IMPORT	<path>\<datasource name>\Data	Folder for files to be imported into the system

Environment variable	Default path	Description
	Import\	

Other variables

Note that some of the variables below are not used - or created - by Agresso Milestone 4:

Environment variable	Description
AGR_TMP_LOC	Database location to use for tables created and used temporarily by server jobs when they are not creating them as database temporary tables. For Oracle the value is a tablespace name, for Sql Server a filegroup name is used. If AGR_USE_REAL_TABLES or AGR_TMP_SAVE is true, the AGR_TMP_LOC will be used if defined.
AGR_ALLOW_ERR	Allow errors during batch input(GL07)
AGR_DEBUGGING	This variable is not used in Milestone 4. Logging is configured through the new logging framework that can be configured for each server queue from AMC.
AGR_LANGUAGE	Language setup code
AGR_LOG_FORMAT	Default log file format. If the value is set to 1, xml logging will be used.
AGR_TEMPDB	By default this variable is not defined. This variable must be defined if a separate database should be used for help tables. See also: Agresso On MS SQL Server
AGR_TMP_SAVE	Save temporary tables. Temporary tables will not be deleted when a server job is complete. It's recommended to configure this setting from the logging configuration page.
AGR_USE_REAL_TABLE	Use real database temporary tables. This is only recommended for trouble-shooting purposes due to performance issues. (it's recommended to configure this setting from the logging configuration page)
AGRESSO_SCRATCH	A directory used as a temporary location for reports. By default this parameter is not defined and Windows TMP environment variable will by default be used.
EXPAND_PATH_WITH_CLIENT	If set to <code>True</code> , the environment variables AGRESSO_PRINT, AGRESSO_EXPORT, AGRESSO_IMPORT, and AGRESSO_OCR, will be expanded with the client code of the report order - if the client code exists. Example: <code>c:\agresso\Data Files\mydatasource\EN\Report Results</code> for client EN.

Clean-up Routines

Prerequisite

Unless you are logged in to Agresso, no information is available.

Purpose

Tells the Business server how it shall handle report history.

Properties

Mode: Allows you to clean up based on

- Number of days - the orders are kept for the number entered.
- Number of report orders - the <number entered> reports are always kept.

Delete files: If checked, also related LIS files and logs will be deleted.

Process info keep: Determines the number of hours items in the [Activity monitor](#) shall be kept.

Main actions

Save. Saves any changes.

Reload.

Blob Handling

Prerequisite

Unless you are logged in to Agresso, no information is available.

Purpose

Allows you set a few properties regarding blobs.

If you enable Blob storage, you may also set a maximum size (in Kb) for blobs.

Main actions

Save. Saves any changes.

Reload. Reloads default settings.

Temporary Tables

Purpose

Allows you to set up rules for how the system shall handle temporary tables.

You can set two properties:

- **Use Real Tables:** If checked, temporary tables will be real tables, not <hva da?>
- **Keep Temporary Tables:** If checked, all temporary tables will be kept (not deleted) for <how long??>

Main action

Save (changes).

Mail Integration

Prerequisite

Unless you are logged in to Agresso, no information is available.

Purpose

You can simply enable / disable mail functionality from the Business server.

Main action

Save (new setting).

Advanced Logging

Purpose

Allows you to modify two properties:

- Log Time Usage
- Default Log-file Encoding

Main actions

Save (changes).

Logging

Prerequisite

Unless you are logged in to Agresso, no information is available.

Purpose

Used to set the default logging properties, stored in the *AgrBusinessServer.logging.config* file. These can always be overridden for various areas.

See [Reference Manual Logging](#).

Main actions

Save (default settings).

Printers

Prerequisite

Unless you are logged in to Agresso, no information is available.

Purpose

Allows you to add printers to the installation. Agresso provides three printer types, based on the [default print handlers](#). The DEFAULT printer (Local print, see below) is already installed.

- Windows printer - based on Agresso Excelerator Print Handler. See [Setting up a Windows printer](#) below.
- Local print - based on Agresso Report Writer Print Handler.
- Customized print of Agresso reports - based on Agresso Print Command Executor. See [Adding a printer based on Command Executor](#) below.

Main action

Add (a printer of a certain type). You can add any number of printers.

Setting up a Windows printer

When you have added a windows printer, you use the **Print Control** node to define the default settings for a user. You can also decide to hide certain properties, or keep the default settings as read only.

Setting up a printer based on Command Executor

When you have added a new printer based on Agresso Print Command Executor, you must use the **Print Control** node to define the configuration. For this purpose, you have access to a series of macros, as a means to build the **ParameterFormat** property.

ParameterFormat for pre-552 command files

To execute a command file using the fixed parameters as required before Agresso 5.5 sp2, use the following command line (note the quotes!):

```
"$AGRESSO_COM$$EXECUTABLE" "$FORMNAME" "$PRINTERQUEUE" "$COPIES" "$DOCUMENTNAME" "$REPORTNAME" "$DOCUMENTEXTENTION" "$DEBUGFLAG" "$PORTNAME"
```

Folder macros for locating the command file

Note the following:

- If the command file to run is located in the Agresso Command File folder, use \$AGRESSO_COM\$.
Example: "\$AGRESSO_COM\$mycommandfile.cmd".
- If the command file to run is located in the Agresso BIN folder, use \$AGRESSO_EXE\$.
Example: "\$AGRESSO_EXE\$mycommandfile.cmd".
- If the command file is placed in any other location, use the full path.
Example: "c:\program files\mycommandfile.cmd".

Available macros

The available macros are described in the table below:

Macro (parameter)	Value
\$AGRESSO_COM\$	The path to the AGRESSO command folder (ends with a backslash)
\$AGRESSO_EXE\$	The path to the AGRESSO bin folder (ends with a backslash)
\$AGRESSO_LOG\$	The path to the AGRESSO log folder (ends with a backslash)
\$AGRESSO_PRINT\$	The path to the Report Output folder (ends with a backslash)
\$COPIES\$	Number of copies to print. The value is taken from CR29.
\$DATASOURCE\$	Name of the Agresso datasource
\$DEBUGFLAG\$	Debug flag. Value <code>DEBUG</code> if debug level is set to Add Queries or higher, otherwise <code>NODEBUG</code>
\$DEVICE\$	The value from the print_queue column of aagprintdef
\$DOCUMENTEXTENTION\$	Document filename extension, including leading period. This is the extension of the report type, e.g. .LIS.
\$DOCUMENTNAME\$	Name and full path of the document to be printed. E.g. GL12a_4.LIS
\$EXECUTABLE\$	The name (no path) of the third party application or command file to execute (for an external print handler). The value is set on the Configuration tab for the Printer Definition.
\$FORMNAME\$	Formular name – if defined for the report.
\$LOGFILE\$	Name and full path to the log file as defined in the Agresso Log.
\$ORDERNO\$	The order number of the report.
\$PORTNAME\$	Port name as set in Printer Port Name in Print Execution Configuration.
\$PRINTER\$	Name of the Agresso Printer Definition
\$PRINTERDEST\$	The value from the destination column of aagprintdef
\$PRINTERQUEUE\$	The value from the print_queue column of aagprintdef (as \$DEVICE\$)
\$PROCESSNAME\$	Name of the process executing the application/command file
\$REPORTCOLS\$	The number of columns in the report
\$REPORTNAME\$	The name of the report to print (e.g. GL12)
\$REPORTROWS\$	The number of rows to print per page
\$SERVERQUEUE\$	Name of the server queue executing the report

Important: If you know that the substituted macro value may contain blanks (space characters), the macro must be enclosed in quotes, e.g. "\$DOCUMENTNAME\$". To be on the safe side, you can use quotes for all macros.

DEFAULT

Prerequisite

Unless you are logged in to Agresso, no information is available.

Purpose

The default report queue.

Properties

You can change the **Type** (Serial / parallel - and number of parallel slots), as well as [Architecture](#).

The **Description** property is displayed as report queue name for the users.

Main child nodes

Activity Monitoring

Gives you the same information as the general [Activity Monitoring](#) node, but filtered by the current report queue only.

Logging

Allows you to set [logging options](#) for each queue.

Recreate: If you select the **Recreate** action, all options will be reset to the default settings for Report Queue.

Environment Variables

Allows you to override [global settings](#).

Mail Configuration

Allows you to override [mail setup for Business server](#).

Instrumentation

General: Sometimes, when a report takes a long time (for example that it never finishes), the server needs to take some action. When we enable instrumentation for a report, we can define a rule that identifies a potential error and a certain error, as well as the actions to take.

Logging: All incidents are written to the Event Log Destination with special event ids. See Actions below

Actions: The following actions are defined:

- **Log potential hang:** If the process has not finished after a number of minutes, this will be written to the log as a warning, with event id = 2039
- **Log process as hanging:** If the process has not finished after a (higher) number of minutes, an error will be logged. Severity = Error. Event id = 2040.
- **Auto terminate:** If set, and the process is logged as hanging, the Business server will terminate the process, and write this to the log. Severity=Error. Event id=2042 or 2041 (couldn't stop the process).

Without instrumentation, no actions will be taken, and a report may hang for days.

Alerts

Allows you to override default settings for [Alerts](#).

Main actions

Disable / **Enable** - dependent on the current state.

Troubleshooting

Report log

Log file

Printing steps are logged in the report's log file, for example *g12-4.log*.

Log content

The log tells you:

- the name of the file to print,
- how the front page is handled
- the plug-in that is used
- any critical errors that occur during printing.

Parameter for additional details: You can set the parameter **Add queries to log file** to get additional details about the printing steps, as well as a reference to a separate Plug-in log file. This log file's name format is:

<file to print>.printing.log (example: *g12_4.lis.printing.log*)

and is stored in the AGRESSO_LOG folder.

AgrPrintExecuter: The external print handler AgrPrintExecuter will also log to an own log file for printing when **Add queries to log file** is turned on. It logs the command format as well as the substituted values representing the command to execute.

Common Errors

We have identified and described a few common errors that can occur:

1

Error message: Could not open device '\\FP47OSLO\K4S_Ricoh1035x' for printing

Explanation and solution: The Agresso Business Server is running in the context of a network user not having the printer device attached or as Local System. Windows Printer Devices are stored in the registry per user. A service running within the Local System context would not be able to see such information.

Assure you log on the computer running Agresso Business Server as the Network user the same user the service logs on as. Now, add the Windows Printer Device using the Printers applet in the Control Panel.

If the service logs on as Local System, change this to a network user and configure the printer for this user. It is recommended that an own network user is created for Agresso Business Server for this use.

2

Error message: Printer plug-in 'AgrPrintReport' not a registered plug-in for printing

Explanation and solution: The Print plug-in has been removed after the Printer definition was created.

You must re-create the Printer definition or use **AG12 (Printers)** in the Smart Client to create the Printer definition if you do not want to use the Print Plug-in.

3

Error message: <application name> is not recognized as an internal or external command, operable program or batch file

Explanation and solution: The parameters (via macros) are incorrect. The folder path may be wrong (you may have used \$AGRESSO_COM\$ instead of \$AGRESSO_EXE\$) or you may have forgotten quotes around macros.

Use the log to get a copy of the command that generated the error. Then try to run the command from a command window and study the error messages.

Printer Setup in Agresso

New basic solution

From version 5.5 SP2, Agresso offers a flexible and consistent solution for printer setup and configuration through the **Agresso Management Console**. Old printer definitions can still be configured using the **AG12** screen (**Printers**) in the Smart Client, but you can also use AMC to upgrade your old definitions, and thereby integrate them into the new solution.

Printer Definitions

The **Printers** node is the parent node for all Printer definitions. A Printer definition can either refer to an existing network printer (Windows printer), or to a program (external print handler) developed for special print purposes. Or it can simply refer to the users default, local print setup.

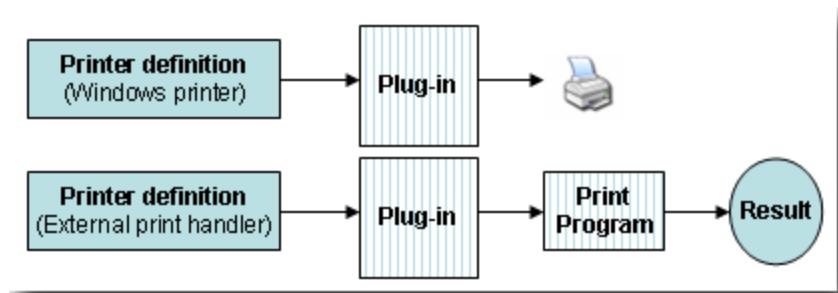
Windows printer: A Windows printer definition will automatically collect printer properties (paper size, orientation, etcetera) from available printers, and you can set property values directly from AMC.

External print handler: An external print handler takes care of special print operations (normally printing that goes to other devices than a windows printer), and is basically a manager for an external program that performs the print job.

You can have several printer definitions using the same available printer or print handler. By tailoring the printer definitions' properties (parameters), you can fine tune the output for different report types.

Print Plug-in

An Agresso printer definition uses a Print plug-in as the means to deliver the print result. The print result is based on Agresso output data and administered by a printer definition. When a report is printed, the report generator identifies the attached printer definition, which in turn activates the attached plug-in to produce the result:



Plug-ins and report types

A plug-in will at least be able to handle one report type, where a report type – also called document type – is identified by the report file extension., such as LIS, XLS an so on.

A printer definition intended to support several report types (for example a windows printer), may therefore be associated with several plug-ins, one for each report type. When a report of a certain type is printed, the plug-in set up to support this report type will take control.

Default plug-in for report type: If you introduce a new plug-in which supports a report type already handled by another (standard Agresso) plug-in, you must set a default plug-in for the report type(s) in question. You can override this default setting by choosing another plug-in on printer definition level.

Custom plug-ins

The use of plug-ins allows you to add print functionality when needed, independent of official Agresso releases. When a new, customised report requires special print handling, you can develop a plug-in to produce the print result based on the report data, and add a new printer definition – using this plug-in – as the printer definition for the new report type.

Note: Currently, you will need Agresso assistance to develop new plug-ins.

Upgrading existing printer definitions

You can upgrade the existing definitions from AMC. When upgrading a definition for a Windows printer, the printer properties (paper size, orientation etcetera) can immediately be (re-) set from AMC. When upgrading an external print handler (command file), you will automatically get the correct parameters set in AMC.

Note: **AG12** is still available, but should no longer be used for printer definitions.

New table for print configuration data

Printer definitions are stored in the table **aagprintdef**. When you create Printer definitions from AMC, or upgrade an existing definition, data will be added to the (new) table **aagsystemconfig**. A printer definition not found in **aagsystemconfig** will be treated as *old* (pre-5.5.3).

Report formats

A report format is always part of a printer definition and is identified by the number of columns and rows per page.

Standard Agresso formats

By default, all printer definitions (including external print handlers) are set up to handle all the standard Agresso formats. By unchecking a report format, you exclude reports using this format from the printer definition:

Report Formating for Printer Definition					
Report Formats					
	Print	Columns	Rows	Report Flags	Form
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	80	66		default
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	132	66		default
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	146	66		default
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	186	66		default

 [Add User Defined Report Format](#)

User defined report formats

You can create custom report formats, which can be required for custom report definitions. You can only delete custom report formats.

Report types (document types)

Standard types

Agresso standard installation supports eight different report types, identified as follows:

Report type (extension)	Description
AGM	Agresso message type. An AGM report may be generated when something unexpected occurs - for instance that there are no data to print. Text format.
FPG	The report's front page, i.e. attribute values and parameters used when the report was ordered. Text format.
LIS	The standard Agresso report format. Text format.
LOG	Log files. Text format.
PDF	PDF format, can be read by Adobe Acrobat Reader.
RDF	Report Creator format.
TXT	Text format.
XLS	Excel format.

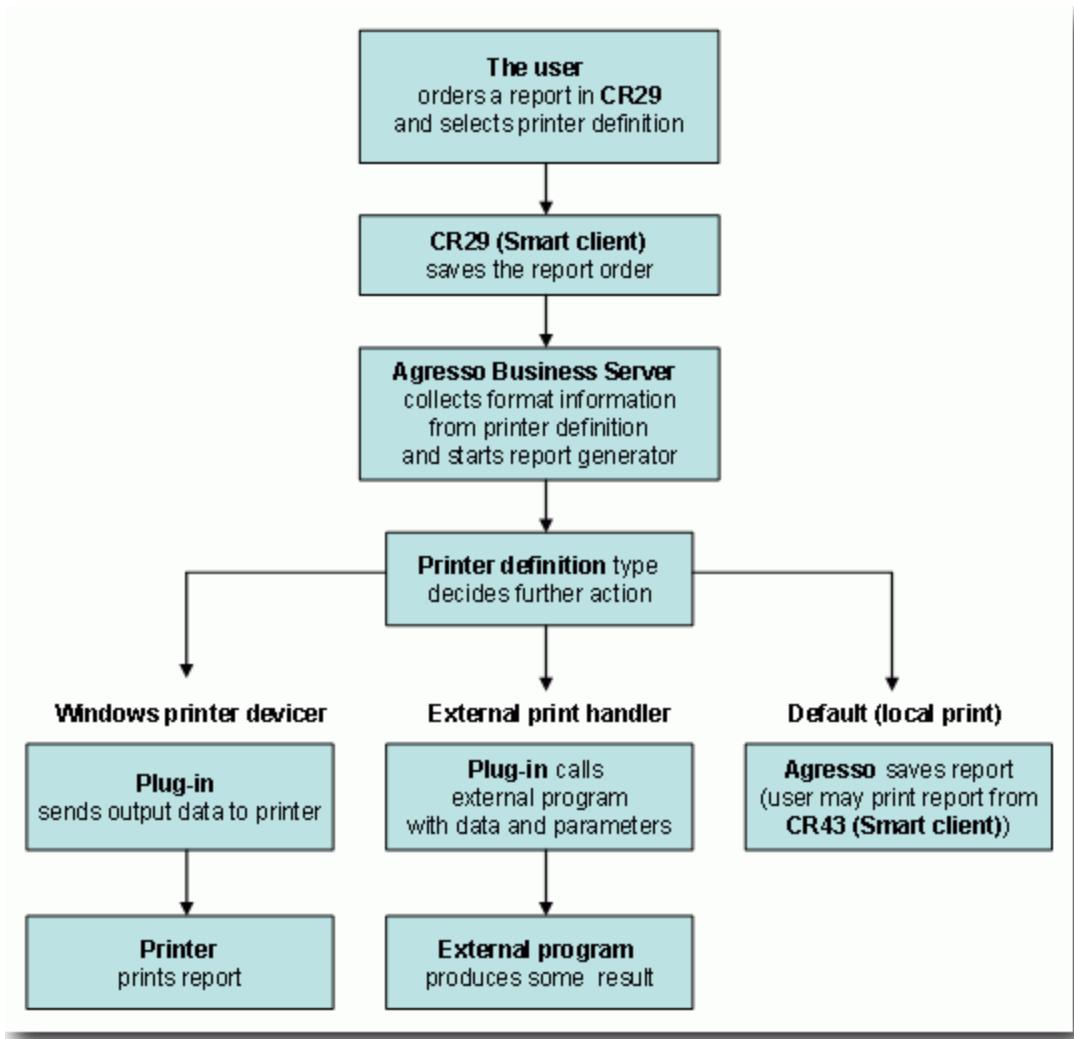
Plug-ins and report types

Agresso comes with a set of plug-ins, covering all Agresso report types, and with no overlap. The table below gives an overview:

Plug-in	Implementation File	Report (File) Types Supported
Agresso Report Print	AgPrintReport.dll	*.LIS, *.FPG, *.AGM, *.LOG, *.TXT
Agresso Excelerator Report Print	AgPrintExcelerator.dll	*.XLS, *.RDF, *.PDF
Agresso External Print Executer	AgPrintExecuter.dll	All

Summary: Print Process Diagram

The diagram below identifies the main elements involved when an Agresso user orders a report:



Mail Queue

Prerequisite

Unless you are logged in to Agresso, no information is available.

Before you can add a new mail queue, you must have defined at least one [system profile](#), not used by another queue.

Purpose

Add new mail queues.

Main actions

Add - a wizard will guide you through the setup.

Delete - available when you have selected a mail queue (previously added).

Alerts

Alert

An alert is basically an e-mail sent to one or more recipients on basis of an error. Currently, you can set up alerts for four error types:

- Technical
- Functional
- No rows
- Other

See [Errors and error types](#) below.

If required, the relevant part of the log file can be attached to the mail.

Prerequisite

Note the following:

- Unless you are logged in to Agresso, no information is available.
- You must have configured a System profile for the Business Server before you can configure alerts.
- The Business Server must be up and running.

Purpose

Allows you to define alerts when an error occur in the system.

Main actions

Save (your settings).

Errors and error types

The Business Server service receives a return value from executed processes. If the return value indicates that an error has occurred, the service will check if alerts are enabled for the error code, and. If so, it will send the alert to the target address (mail address or phone number).

You can configure up to four different alerts, one for each of the following error types:

Error type	Description
TECHNICAL ERROR	Indicates a general error that the administrator should look into. Examples: The database connection is broken or the process fails to run.
FUNCTIONAL ERROR	Indicates that required data is missing, making it impossible to complete a report or a process.
NO ROWS	Occurs when a report returns zero rows. Note: This is not really an error; there is just not any data. If you set up an alert for this error, it is recommended to only send it to the report owner.
OTHER ERRORS	Covers all other errors, for instance if a process is killed in AMC.

When locating the settings to use, the **AgrBusinessServer** service will first look for the settings at the Server queue level. If none exists, it will look for settings at the Business Server level.

Self-Hosted Services

Purpose

Allows you to install and maintain two services:

- [DocArchiveFileStorage](#)
- [EventServer](#)

Main action

Add (a service).

Consequences when adding DocArchiveFileStorage

When you add **DocArchiveFileStorage**, both the Web server and the database will be updated.

Web server

During installation, the following will happen on the Web server:

- The exact address of the web service will be available through the service host.
- The file `<datasource name>.AgrEventServer.exe.config` will either be created or updated, with a new `<service>` node, holding configuration properties for the service.

Database

The database - available for all clients - will always be updated:

The Common parameter FILESTORAGESERVICE_URL will be added to the system setup and given a value pointing to the web server. The web service is not identified, however.

DocArchiveFileStorage

Purpose

Allows you modify properties for the service. You will do this via the child nodes.

Main actions

Start / Stop (the service).

Save (property changes).

Child nodes

DocArchive Settings

You can set two properties:

- **DocumentArchiveRoot** - the folder to be used for file storage.
- **UseWeekNumber**.

Note: If you already have a document system using Web service file storage based on Agrezzo 5.5.3 Update 10 or Agrezzo Milestone 1, update 1, you must make sure that the **Document Archive Root** is set to the folder already in use and that **UseWeekNumber** has the same value as before.

Endpoint Information

Displays installation information.

Logger

Allows you to override default logging. See [Configuration / Logging](#).

Event Log Messages

Displays Agresso event log messages (An Agresso Event Log destination must be configured).

If you update an existing Document system to new file storage solution

Do as follows:

Open the Agresso Smart client and do as follows:

1. Open the **Document system** window (System Administration / System setup / Document archive)
2. Do you already have a document system for file storage based on Agresso 5.5.3, update 10?

If Yes:

Change the name in the **Dll name** column to `AgrDSFileStorage.dll`. This will ensure that the new solution will be used.

If No:

- Add a new document system (with a meaningful description. The description will be used to identify the document system in the **Document type** window)
- Make sure that **Dll name** is set to `AgrDSFileStorage.dll`.

You can now create document types (based on this document system) using the new solution.

If you will upgrade from AGRESSOFILE to new file storage solution

Keep AGRESSOFILE as name of the document system

AGRESSOFILE used a shared folder on a server for document storage, and you can upgrade to the new version without changing the storage area.

You need to do the following:

1. Remove any unnecessary access rights to the shared folder.
2. Grant full access (to the shared folder) to the web server administrator (i.e. the user that owns the new Web service).
3. Install the new solution using AMC and set correct path in **DocumentArchiveRoot**. See above.
4. Modify the settings for AGRESSOFILE: Change **Dll Name** to `AgrDSFileStorage.dll`.
5. Remove any settings for the DS_NATIVE_ system parameters. They will no longer be used.

This will ensure that all document types using the AGRESSOFILE document system are unaffected. AGRESSOFILE now will use the new component, `AgrDSFileStorage.dll`, when storing and retrieving documents.

EventServer

About Agresso Event Server

The Agresso Event Server service keeps track of the report activity controlled by the Agresso Business Server. The Agresso Event Server replaces the Agresso Alert Server web services used in previous Agresso releases.

The service allows the Smart client to fetch information about report status without querying the database. The **Ordered reports** pane in the Smart client is dependent on this service to be able to show the updated report order status.

Purpose

Allows you to start and stop the Event server service, and set general properties.

Main actions

Start / Stop (the service).

Save (property changes)

Child nodes

Endpoint Information

Displays installation information.

Logger

Allows you to override default logging. See [Configuration / Logging](#).

Event Log Messages

Displays Agresso event log messages (An Agresso Event Log destination must be configured).

Additional information

The Smart client uses the Common parameter ALERTSERVER_URL to locate the path to the service. By default the Smart client polls the service for changes every 10th second. The poll frequency in the smart client can be configured by adding the REPORT_TIMEOUT common parameter.

A windows firewall exception is added when the service is created.

Default Web Site

Prerequisite

Unless you are logged in to Agresso, no information is available.

Purpose

Manage the site, i.e. start or stop the site and handle web applications. The following application types are available:

- [Agresso Self Service Web Application](#) - the standard Web client.
- [Agresso Self Service Web Client \(Experience Packs\)](#) - the new Web client, not available before Milestone 4.
- [Agresso Web Service Host Application](#) - required to install Web services.
- [Agresso ReportEngine Web Application](#) - used internally by Agresso Report Engine.

Main actions

The following actions are of interest:

Add... - used to add a Web application. A Wizard will provide the available applications.

In order to add Web services, you must first add the [Agresso Web Service Host Application](#).

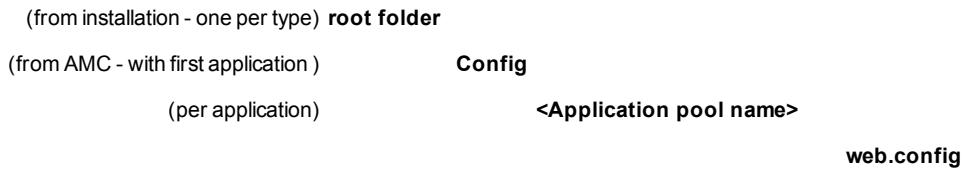
Start site/ Stop site (the Default Web Site)

Shared file set

When you add a web application (ReportEngine not included!), the wizard will propose a default location (Physical Folder) for the application files, created as part of the installation. Each application type will have a separate root folder.

If you add a new application, of the same type as one previously added, the wizard will now propose the same root folder for the application files, and only create a new sub-folder for the application specific properties (e.g. `ServerDataSource`).

The folder structure is as follows:



Benefits

File maintenance and upgrades will be simplified by using shared files.

Restraints

All applications will share the same `web.config`, with the exception of private `appsettings`, stored in the `web.config` under `<Application pool>`. If you update general configuration for one application, you will also update the configuration for all applications sharing the same file set.

Self Service Client

Purpose

Allows you to add the Self service client.

Main action

Delete. Removes the client.

Main child nodes

User message

Allows you to add a default message, displayed to active users(For example before the system is shut down). The message will remain until it is disabled.

Maintenance

Allows you to set the client in offline mode (and set it back to online), and display a default message for users trying to log on.

(Settings are stored in the [App_offline.htm](#) file in the applications root folder):

Application Configuration

Shows the general Application Configuration.

UI Behaviour

Defines the error detail level to display to the users when an error occurs. You can override default `appSettings` (from main [web.config](#)).

Miscellaneous

Allows you override default `appSettings` (from main [web.config](#)).

Application Pool

Displays Application Pool properties (from IIS).

Protocol

Shows current protocol settings (HTTP/HTTPS/HTTPS (login)). If changed, the [web.config](#) will be updated (the mode and secure attributes (for `<sslRedirect>` and `<file>`)).

Note that if you create multiple Web Service Host Applications with a [shared file set](#), they will also share the protocol settings.

Authentication

Allows you to see and change authentication setup.

Note that you must have enabled the correct authenticator for **Self service** via the [Authentication setup](#) node.

You must also have turned on **Windows Authentication** for IIS before you can enable Windows Authentication for a web application. Go via **Control Panel>>Programs and Features>>Turn Windows features on or off**. Check **Windows Authentication** under **Internet Information Services>>World Wide Web Services>>Security**.

Logger

Allows you to enable / disable logging for the Web services. When you select the node, you get access to all log options via the main panel.

See [Configuration / Logging](#) for more details.

Event Log Messages

See [Event Log Messages](#) (under the Business Server/Monitoring node).

Self Service Experience Packs

Purpose

Allows you to add the new Self Service With Experience Packs container.

Main action

Delete. Removes the container.

Main child nodes

User message

Allows you to add a default message, displayed to active users(For example before the system is shut down). The message will remain until it is disabled.

Application Configuration

Shows the general Application Configuration.

UI Behaviour

Defines the error detail level to display to the users when an error occurs.

Application Pool

Displays Application Pool properties (from IIS).

Protocol

Allow you to change the site bindings . Can also be done via IIS.

Authentication

Allows you to see and change authentication setup.

Note that you must have enabled the correct authenticator for **Self service** via the [Authentication setup](#) node.

You must also have turned on **Windows Authentication** for IIS before you can enable Windows Authentication for a web application. Go via **Control Panel>>Programs and Features>>Turn Windows features on or off**. Check **Windows Authentication** under **Internet Information services>>World Wide Web Services>>Security**.

Logger

Allows you to enable / disable logging for the Web services. When you select the node, you get access to all log options via the main panel.

See [Configuration / Logging](#) for more details.

Event Log Messages

See [Event Log Messages](#) (under the Business Server/Monitoring node).

Web Service Host Application

Purpose

The Agresso Web Service Host Application node

- allows you to add and maintain all the externally available web service versions delivered by Agresso, and
- provides you with child nodes holding general IIS configuration options, in addition to standard Agresso specific configuration.

The complete configuration is stored in a set of *.config* files, stored under **Physical Folder** (a Web Service Host property).

Main action

Delete. Removes the host.

Main child nodes

Application Configuration

Show the general settings for the Host Application in the main pane. If available, you can change

- Data source.
- Default language.

Application Pool

Displays Application Pool properties (from IIS).

Protocol

Shows current protocol settings (HTTP/HTTPS). If changed, the *web.config* will be updated (the `<security mode>` attribute under `<basicHttpBinding>` will be updated).

Note that if you create multiple Web Service Host Applications with a shared file set, they will also share the protocol settings.

Authentication

Allows you to see and change authentication setup.

Note that you must have enabled the correct authenticator for **Web services** via the Authentication setup node.

You must also have turned on **Windows Authentication** for IIS before you can enable Windows Authentication for a web application. Go via **Control Panel>>Programs and Features>>Turn Windows features on or off**. Check **Windows Authentication** under **Internet Information services>>World Wide Web Services>>Security**.

Logger

Allows you to enable / disable logging for the Web services. When you select the node, you get access to all log options via the main panel.

See Configuration / Logging for more details.

Event Log Messages

See Event Log Messages (under the Business Server/Monitoring node).

Web services

You use the Web services child node to add (or remove) Agresso Web service versions. Web services are listed in the main pane.

Main actions: It offers two main actions:

- **Add** - allows you to add one or more web services (not already added).
- **Recreate** - removes and re-creates all web services.

Actions per Web service: Test, View properties, Delete.

ReportEngine Web Application

Purpose

The ReportEngine Web Application node

- allows you to manage the Report Engine web application
- provides you with child nodes holding general IIS configuration options, in addition to standard Agresso specific configuration.

The complete configuration is stored in [web.config](#) file, stored under Physical Folder (a ReportEngine property).

Main action

Delete. Removes the application.

Child nodes

Application Configuration

Show the general settings for ReportEngine in the main pane. If available, you can change the [ServerDataSource](#).

Application Pool

Displays Application Pool properties (from IIS).

Protocol

Allows management of the web site bindings.

Authentication

Allows you to see and change authentication setup.

Note that you must have enabled the correct authenticator for [Web services](#) via the [Authentication setup](#) node.

You must also have turned on [Windows Authentication](#) for IIS before you can enable Windows Authentication for a web application. Go via [Control Panel>>Programs and Features>>Turn Windows features on or off](#). Check [Windows Authentication](#) under [Internet Information services>>World Wide Web Services>>Security](#).

Publish

Saves the URI for the application to the database table [aagserviceuri](#). This allows other applications to discover the Web application by looking up the [REPORTENGINE](#) service id.

Centrally Configured Clients

Purpose

Allows you to manage Centrally Configured Clients (CCCs).

The CCC master installation

All required CCC software was (by default) installed in the bin folder when you installed the Agresso Smart Client on the server. This is referred to as the *master installation* for all centrally configurated clients.

Updates: Service packs are installed on top of the master installation. AMC has functionality to refresh the CCC files based on the master installation. See [File Maintenance](#) for information on how to update existing CCCs with new files.

A note on .NET Framework

Please note that all (local) users of the shared client must have the correct .NET Framework version installed.

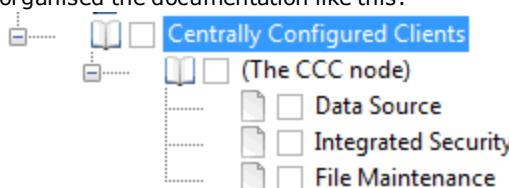
Main actions

Add... / **Remove...** (client).

Add

When adding a new CCC, a wizard will guide you through the installation. As a result, you get the following:

- A new node under Centrally Configured Clients, with the name of the new CCC (The CCC node). In these Technical Guidelines, we have organised the documentation like this:



- A new shared folder with all required files. Database connection are stored in `..\config\agresso32.ini`. See also [File Exclude List](#) below.

The File Exclude List

Not all files in the master installation should be distributed to the client users. Some files might be harmful or confusing when executed on the client computers. To prevent files from being copied from the master bin folder to the network share bin folder, you can edit the `AgrCCCFileList.xml`, located under the `<shared CCC folder>\Config`. directory.

Updates

The content of `AgrCCCFileList.xml` will also be used each time you update or refresh the CCC installation. See [File Maintenance](#).

Data Source

Purpose

The Data Source node displays the database related content of the `agresso32.ini` file.

Main actions

Save. You can make changes to the data source configuration.

DriverFile (MSSQL only): Note that ODBC drive file must be installed on all computers accessing the CCC.

Integrated Database Authentication

[Open Reference manual](#)

The link above will open a PDF-version of the Reference manual for Integrated Database Authentication.

File Maintenance

Purpose

Lets you update the CCC installation with a new set of files.

This is mainly relevant after you have installed a new update from Agresso.

Main action

Start - will need your confirmation before it installs the current files from the [master installation](#).

System Configuration

Prerequisite

Unless you are logged in to Agresso, you will not get access to the child nodes.

Purpose

Gives you access to the following areas (via child nodes):

- [License](#)
- [Common Parameters](#)
- [Task Scheduler](#)
- [Mail](#)
- [Print Integration](#)
- Collaboration Integration
- [Logging](#)
- [Authentication](#)
- [Users and Access](#)
- [ACT](#)
- Help Configuration

Main actions

None. You go via the child nodes.

License

Prerequisite

You must be logged in to Agresso .

You will have received the new license by email. Before you can add the new license to AMC, you must have saved it somewhere on disk.

Purpose

Allows you to update (or add) an Agresso license to the database. It also allows you to view the licensed modules and the number of users registered for each module.

If you have a demo license installed, this node will inform you if the license will soon expire, or if it has already expired.

Main actions

Update - will start the **Update license wizard**. You will then browse for the new license file.

Common parameters

Prerequisite

Unless you are logged in to Agresso, no information is available.

Purpose

Allows you to view and update common parameter settings, previously only available from the standard Agresso client.

Main actions

- **New** - create a new common parameter. Note that common parameters are used by Agresso code. Unless you have a custom component - for example - that will use the new common parameter, it will have no effect.
- **Override** - change parameter settings.
- **Delete** - delete a parameter. Only possible for new or changed parameters. When you delete a changed parameter, it will be reset to the original settings.

Task Scheduler

Prerequisite

Unless you are logged in to Agresso, no information is available.

Purpose

You use **Task Scheduler** to define new tasks - or jobs - that shall be executed on the business server. A task can do one of the following:

- Run a program.
- Send an e-mail.
- Display a message.

You can define a one time task, or a recurring task, with one or more trigger conditions.

Main action

Add - a wizard will guide you through the setup process. Below, we give a short description of a few properties, related to dialogs in the wizard.

Security settings

Configure the privileges the process should be executed with.

Triggers

Note that you can use several conditions to trigger the task, and all will be active. This means that a Daily condition may overlap a Weekly condition etc.

If you synchronize across time zones, the task will

Other settings (three dialogs)

You may specify that the task shall run only when the computer is idle (note that this may be in conflict with logged on user) and whether it shall not run when the computer is powered by a battery only. In addition, you may set some additional properties related to possible errors or conflicts that may occur.

Mail

Prerequisite

Unless you are logged in to Agresso, no information is available.

Purpose

The Mail node is the entry point for all mail configuration, both for the clients and server side mail. You get access to two child nodes:

[Mail Integration](#) - used to install and configure plug-ins.

[Mail Profiles](#) - used to configure and create mail profiles for various use.

Mail Integration (plug-ins)

Mail system configuration is dependent on at least one mail system interface plug-in, ensuring correct communication between your existing mail system and Agresso. Currently, Agresso comes with [four mail plug-ins](#). You can also add your own.

Custom plug-ins: Agresso Mail configuration also supports custom mail system plug-ins, provided that the plug-in is compatible with some important interface requirements. These requirements are currently not published, but can be obtained from Agresso consultants.

Mail Profiles

Profile Template: A profile template is a default mail profile setup, used to create new working profiles. A template has a set of property values relevant for a certain plug-in, required when the plug-in shall connect to the mail system.

When you have defined a template for a certain plug-in, you will use this to create the actual profiles for users and server side programs (system).

User profiles: User profiles will automatically be configured first time users try to send mail, based on the profile template.

System profiles: If the configuration shall support server side mail, you must always define details for the System profile - or confirm the use of the default settings.

Supported mail systems

Agresso currently supports the following mail systems:

Mail System	Description
SMTP (Simple Mail Transfer Protocol)	<p>For server-side mail integration, like distributing reports on mail from the Agresso Business Server.</p> <p>Smart Client users: SMTP does not have a user interface. When Smart Client users are set up with an SMTP profile, Agresso has provided a mail interface contained in the library AgrMailUI.DLL.</p> <p>AgrMailUI contains an Agresso integrated Address Book and a Compose Mail dialog with text and HTML capabilities.</p>
Simple MAPI (Messaging Application Programming Interface)	<p>Outlook Express and Windows mail implements Simple MAPI only.</p> <p>Simple MAPI is also supported by other messaging applications like Lotus Notes and Novell Groupwise.</p> <p>Note: Simple MAPI is <i>not recommended</i> for server-side mail integration.</p>
Extended MAPI	<p>Provides an extensive set of functions used to create mail-enabled applications. The full function library is known as MAPI 1.0 or Extended MAPI. Extended MAPI allows complete control over the messaging system on the client computer, creation and management of messages, management of the client mailbox, service providers, and so forth.</p> <p>Microsoft Outlook supports Extended MAPI.</p>

AgrMailUI - the Agresso provided user interface for SMTP users

Agresso Mail User Interface provides a user interface for Smart Client users that are set up with SMTP as mail profile.

Dialogs

AgrMailUI provides the user with two dialogs:

- A standard **Send mail** dialog - with functionality found in most mail clients.
- An **Address book** - integrated with the Agresso **User master file**.

Agresso table for contacts - aagcontacts

Using AgrMailUI, both personal contacts and global contacts are added to the table *aagcontacts*.

Global contacts: Global contacts are added with an asterisk (*) in the user_id column, meaning that the contact will be available for all Agresso users.

Personal contacts: When a personal contact is added, the User Id of the user making the registration are added in the user_id column, meaning that the contact only will be available to this user.

Valid address formats

As a general rule, all mail addresses should be stored in SMTP format. The following address formats are all valid:

[Display Name] SMTP:mail.address@mycompany.com

[Display Name]mail.address@mycompany.com

SMTP:mail.address@mycompany.com

mail.address@mycompany.com

Mail Integrations

Prerequisite

Unless you are logged in to Agresso, no information is available.

Purpose

Gives you access to the various mail integration plug-ins delivered by Agresso - as well as any custom components added by yourself.

The default plug-ins are described below (all plug-ins will appear as child nodes):

- [Agresso SMTP Server Integration](#) - the recommended choice for all server side mail. Should be used as the executing system, along with Agresso Mail Queue Post Integration.
- [Agresso Mail Queue Post Integration](#) - will not intermediately send the mail but store the mail in a register handled by Agresso Message Server (AMS). See [About mail queues](#) below.
- [Agresso Extended MAPI Client Integration](#) - handles server side mail, but can currently not be used as the executing system for Agresso Mail Queue.
- [Agresso Simple MAPI Client Integration](#) - for users with MAPI compatible mail clients on their desktop. Can not be used for server side mail.

Actions

Add - allows you to add a custom mail integration plug-in - or a deleted Agresso plug-in.

About mail queues

As a general rule, all mail in Agresso should go via a mail queue.

A mail queue provides extensive error handling and ensures full control of mail status. A mail queue can be [monitored](#) and maintained using tailored screens in AMC. You can also use Agresso Management PowerShell for automation.

See [Mail Queue](#) for more details.

Agresso SMTP Server Integration

Prerequisite

Unless you are logged in to Agresso, no information is available.

Purpose

[Agresso SMTP Server Integration](#) is the recommended choice for all server side mail. It should be set up as the mail executor (mail protocol), monitored by Agresso Server Queue.

Actions

Delete - allows you to remove an integration plug-in from the configuration. Can be added later.

Agresso Mail Queue Post Integration

Prerequisite

Unless you are logged in to Agresso, no information is available.

Purpose

[Agresso Mail Queue Post Integration](#) - will not intermediately send the mail but stores the mail in a register handled by Agresso Message Server (AMS). The AMS server queue uses one (or more) of the other plug-ins to actually send mail, and the mail will be kept in the database until it is successfully sent. AMS supports re-send options, error handling, status inspection and so forth

Actions

Delete - allows you to remove an integration plug-in from the configuration. Can be added later.

Agresso Extended MAPI Client Integration

Prerequisite

Unless you are logged in to Agresso, no information is available.

Purpose

[Agresso Extended MAPI Client Integration](#) handles server side mail, but can currently not be used as the executing system for Agresso Mail Queue.

Actions

Delete - allows you to remove an integration plug-in from the configuration. Can be added later.

Agresso Simple MAPI Client Integration

Prerequisite

Unless you are logged in to Agresso, no information is available.

Purpose

[Agresso Simple MAPI Client Integration](#) for users with MAPI compatible mail clients on their desktop. Can not be used for server side mail.

Actions

Delete - allows you to remove an integration plug-in from the configuration. Can be added later.

Mail Profiles

About Mail Profiles

A mail profile contains a mail from-address as well as a set of properties related to a selected plug-in.

All active profiles must be based on a Profile template.

Profile templates

A profile template is a model configuration, and used to create actual profiles. All profiles will inherit the general configuration from the template.

A Profile template can be used for both user profiles and system profiles. Some properties, related to what the user can see and modify, are only relevant for user profiles.

User Profiles

A user profile is a mail profile for a Smart client user, initially only identified as a template. When a user tries to send mail for the first time, the selected template will prompt the user to fill in personal information, and then create a unique profile for the current user.

System Profiles

You need a system profile if you intend to send server side mail.

Profile Templates

Profile templates

General

Depending on whether your mail system is intended for system reporting, for ordinary use by employees, or both, you must implement at least one configuration profile. Regardless of the number of profiles you need, and the eventual user types intended for the profiles, you follow the same basic steps for each.

Business server restrictions

A mail system (intended for the Business server) should as a rule be based on Agresso Mail Queue, but SMTP or Extended MAPI can also be used.

Extended MAPI: If you will use an Extended MAPI plug-in, **Microsoft Outlook** must be installed on the server and configured for the account to be used by the Business server.

Main tasks

Creating a new Configuration profile, involves four small tasks:

- A. Select a mail system integration plug-in and add it as a new node to AMC.
- B. Select the new node and add a new Configuration profile node to AMC.
- C. Select the Configuration profile node and define the default setup on the Mail Configuration page
- D. Add at least one System profile.

Create a plug-in node

A. Add Mail Integration plug-in

1. Right-click the **Mail Integrations** node and select **Add**

2. Select the desired plug-in and click **Finish**

B. Add Profile template

The next step is to add the configuration profile (the system and user profiles come automatically):

1. Right-click **Mail Profiles\Profile Templates** node and select **New**.

A dialog box is displayed where you can select the mail integration plugin to use.

2. Enter a meaningful name and click **Finish**.

C. Configure the Profile template

1. Select the node to view the different configuration options. You can control which settings the user should see, and which settings the user should be able to change.

Page details

SMTP Server Connection

Parameter	Value	Description	Read only	Hide from UI
HostName	127.0.0.1	Host Name/Address	<input checked="" type="checkbox"/>	<input type="checkbox"/>
PortNumber	25	Port Number	<input checked="" type="checkbox"/>	<input type="checkbox"/>
AutoConnect	<input checked="" type="checkbox"/>	Automatically connect to internet	<input type="checkbox"/>	<input checked="" type="checkbox"/>
LocalBoundAddress	Any IP Address	Local Bound Address	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
UseSSL	<input type="checkbox"/>	Use SSL	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
[MailUI]	AGRESSO Mail User Interface	Mail Interface	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
TimeOut	60	Timeout (in seconds) during data send to server	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

Authentication

Parameter	Value	Description	Read only	Hide from UI
AuthenticationMethod	Authentication Login	Mail Server Authentication Methods	<input checked="" type="checkbox"/>	<input type="checkbox"/>
UserName		Mail Account Name (at the SMTP Server)	<input type="checkbox"/>	<input type="checkbox"/>
Password	*****	Mail Account Password	<input type="checkbox"/>	<input type="checkbox"/>

Mail Options

Parameter	Value	Description	Read only	Hide from UI
DefaultEncoding	Unicode (UTF-8)	Default Mail Encoding	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
MIMEEncoded	<input type="checkbox"/>	Send all messages with MIME encoding	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
ValidateRecipients	Skip e-mail address validation	Validate e-mail addresses	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
AlternativeBodyDefaultText	If you see this text then your mail client is not mime/html compliant	Default text to show if mail client is not mime/html compliant	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
AlternativeBodyHandling	Default alternative body	Alternative body	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

Proxy server

Parameter	Value	Description	Read only	Hide from UI
ProxyType	No proxy server	Type	<input checked="" type="checkbox"/>	<input type="checkbox"/>

Descriptions

You set default parameter values for **Authentication**, **Mail Options** and **SMTP Server Connection** (in this example). The various check boxes are used to set rules for client users, when they try to send Agresso mail for the first time.

Authentication (SMTP)

AgrSMTP supports the following authentication methods:

- [No login](#) - SMTP server looks at your IP address to ensure you are on the RCN network.
- [Authentication Login](#) - The authentication data is sent to the SMTP server as Base64 encoded text.
- [NTLM \(NT LAN Manager\)](#) – Uses Microsoft Authentication Protocol to authenticate the user (Agresso user or server side software)

trying to access the SMTP server to send mail.

Note: To be able to use this protocol when sending mail from the Business Server, assure the Business Server service logs on as a valid network user.

- **Cram MD5 Method** - CRAM-MD5 is a challenge-response authentication mechanism to authenticate the SMTP connection. It encrypts the client's user name and password and is more secure than Plain Login and Authentication Login.
- **Plain Login** – The authentication data is sent to the SMTP server as plain text.

Default Mail Encoding

The value selected for **Default Mail Encoding** must correspond to the value given by Agresso system parameter DEF_FILE_ENCODING. The following two values are most relevant:

If DEF_FILE_ENCODING = **ANSI**, Default Mail Encoding shall be set to **Western European (ISO)**.

If DEF_FILE_ENCODING = **UTF8**, Default Mail Encoding shall be set to **Unicode (UTF-8)**.

2. Change the values to fit your local environment.

Example: When setting up an SMTP mail integration you typically set which type of Authentication the mail server is using as well as the SMTP server's host name or IP address where the mail server can be reached. We recommend that you fill in the fixed values for your environment and select **Read Only** and **Hide for User** for these values.

D. Add a system profile

If the mail configuration profile is to be used from the server (the **Not Empty** box is checked - see Page example above), you need to create a system profile.

1. Right click the **System Profiles** node and select **New**

2. Type in the system profile name and click **Next**.

Note: This is the name you will use when you later link the implementation to the Business server.

3. Type in the Display name (for the email address) and the email address to use, then click **Next**.

4. Set the correct parameters and you are done.

Note: It is mandatory to have a correct mail account name and password.

Result

You have now created a new, general configuration profile, in principle available both for Smart Client users and for the Agresso Business Server.

System Profiles

System Profiles

If your implementation shall support server side mail, you will need to create at least one system profile that will be the identifiable mail sender. The system profile must match a registered user on the mail server.

Remember

Only SMTP and Extended MAPI are supported as mail systems for Agresso Business Server.

Agresso Server Queue

If you want Agresso Server Queue to handle your mail, you must link the System profile for the *server queue configuration profile* to the Business server.

User Profiles

When you click on the **User Profiles** node in AMC, all users registered with the active configuration profile will be listed in the right hand panel. This will be empty immediately after creation of the configuration profile.

When the users defined with this profile first try to send mail, they will be prompted to enter (or confirm) a few values - according to the valid user profile. Users will thus create their own profiles when required.

Main action

New - allows you to create a new profile, on behalf of any user.

Print Integration

Purpose

Gives you access to two child nodes for print administration:

[Print Handlers](#).

[File Type Associations](#).

Main actions

None.

[Read more about printer setup](#).

Print Handlers

Purpose

Allows you to manage print handlers used in the installation.

Main actions

Add - allows you to add a new print handler. A wizard will guide you through the installation.

You will find three standard print handlers:

- Agresso Report Writer Print Handler
- Agresso Excelerator Print Handler
- Agresso Print Command Executor. See [Print Command Executor properties](#) below.

When you have added a print handler, you can just view the print handler properties, not change their values.

Plug-in capabilities for Report Writer and Excelerator

In addition to the supported [file types](#), you can see the **Plug-in capabilities**. They represent what you can set up as default printer settings in AMC. and what you want to display as user configurable properties.

Macros for Print Command Executor

The Print Command Executor provides a series of macros, which you can use when you set up a specific printer based on this print handler. See [Setting up a printer based on Command Executor](#).

File Types Associations

Purpose

Displays information about the default print handler used by the various file types.

Unless you have added custom print handlers, as an alternative for a file type, the settings cannot be changed.

See also [Printer setup in Agresso, Report types](#).

Main actions

None.

Collaboration Integration

Prerequisite

Unless you are logged in to Agresso, no information is available.

Purpose

Add new integration plug-ins.

We deliver one default plug-in for Microsoft Exchange.

Main actions

Add - a wizard will guide you through the setup. If you want to add a custom plug-in (other than Microsoft Exchange), you need to browse the file system for the [.dll](#).

Delete - available when you have selected a plug-in (previously added).

Exchange Server Integration

Purpose

Part of the setup for integration between Agresso Assignments and Microsoft Exchange. When everything is installed and correctly configured, *assignments* in Agresso can be automatically copied to the relevant user(s) Outlook calendar as *appointments*.

Main action

Add / Delete - allows you to add (delete) an integration profile for server integration. You will only need to give the profile a name.

Configuring an Integration Profile

When you have added a new profile, it will appear as a child node, and the profile properties will appear in the main pane (when you click on the child node).

Properties for Exchange Server Connection

You can enter or modify the following properties for the **Exchange Server Connection**:

Property	Description
Uri	The address to the Exchange web service. Can be blank if the Exchange server supports Auto-discovery.
Version	Refers to the Exchange web service version.
Agresso User Email Filter	A "wildcard" email address, but without wildcard symbols. Example: If the setup shall be valid for all users belonging to a certain domain, enter the domain only: <code>mydo-main.com</code> . If you need more than one domain, use ';' as separator: <code>mydomain.com; yourdomain.com</code> .
Read Only	Must be <code>un-checked</code> if you want to integrate Agresso <i>assignments</i> with Outlook <i>appointments</i> .

Properties for Authentication

The **Authentication** properties are used to identify the "super-user" which will access the Exchange Web service and also appear as the sender, when an assignment is copied as an Outlook appointment:

Property	Description
Username	The email address to the super user who will access the Exchange Web service (and manage appointments).
Password	The super users Exchange password.
Domain Name	

Logging

Prerequisite

Unless you are logged in to Agresso, no information is available.

Purpose

Gives access to all elements used for logging configuration, available as child nodes (under [Template Configuration](#)).

Main actions

None.

Reference

The logic behind the logging system is described in detail in [Reference Manual - Logging \(PDF\)](#).

Template Configuration

Logging template

A logging template is a model setup for logging, to be used when you create logging configuration for different Agresso components.

Purpose

Allows you to configure the various components used in logging:

- [Trace Listeners](#) - the destinations for the log entries.
- [Log Filters](#). - defines the log filters.
- [Log Formatters](#). Determines what to write to the log, and how it shall be formatted.
- [Severity Filters](#). Establishes the connection between a severity level and a destination (Trace Listener).

Main actions

Reset - will reset any custom configuration (made by you), back to Agresso default.

Trace Listeners

Trace Listeners and Formatters

A **Trace Listener** is a destination for the log entry - i.e. the place where the log message is written. A Trace Listener must always be set up with a [Log Formatter](#), determining what to write, and how to format it.

Purpose

The Trace Listener node allows you to configure the pre-defined log destinations.

Log Filters

Delete this text and replace it with your own content.

Log Formatters

Log Formatter

A formatter is a specialised component (implemented as a plug-in) which takes any log entry and converts it to a format suitable for a given destination type.

Purpose

Gives you access to the standard Agresso formatters, as child nodes.

Severity Filters

Delete this text and replace it with your own content.

Configuration Files

Purpose

Gives an overview of, and access to, the logger configuration files currently used by the system.

The files are displayed as child nodes, and you must select a child node in order to change the current configuration in that file.

Main actions

New - add a new configuration file.

Delete - to delete a configuration file, you must select the relevant file in the main (middle) panel.

Working with configuration files

When you select a child node to change the configuration, the main panel will display the current settings. See [Reference manual - Logging](#) for details.

Authentication

Purpose

The Authentication node gives you access to the following child nodes:

- [Authentication Setup](#).
- [Authenticators](#).
- [Password Policy](#).
- [Password Reset](#).

Main actions

None.

Reference

See also [Authentication and Authenticators](#).

Authentication Setup

Prerequisite

Unless you are logged in to Agresso, no information is available.

Purpose

The Authentication setup window gives you access to all (loaded) authenticators and allows you to:

- activate authenticators for a certain platform
- set default authenticator for a platform and, optionally, any alternative authenticators to be used if the default one fails.

Activate an authenticator

Do as follows:

1. Choose the platform you want to work with.
2. Pick the authenticators you will activate.
3. Set priority for all activated authenticators (set default, move up/down)

Authenticators

Prerequisite

Unless you are logged in to Agresso, no information is available.

Purpose

The **Authenticators** window lets you configure the current system by selecting specific authenticators to be used for the available clients. The window allows you to:

- Load custom authenticators into Agresso and, optionally, deploy them automatically
- Remove (or delete) custom authenticators from the system
- View the list of currently available authenticators
- View details such as assembly and class for each authenticator

Fields

(Checkbox)

Selection column. Must be selected before you can set properties for custom authenticators.

System authenticator properties can not be changed.

Name

The internal name of the authenticator as found in the .dll and cannot be changed.

Platform

Indicates which platforms (Smart client, Self service application) the authenticator is made for. Cannot be changed.

Title id

The ID of the title (as used in the Agresso title system in AgrDataTools) to be used for the Display name. This is only relevant if the authenticator comes with a user interface.

Description

The description that is displayed to a user if the authenticator has a user interface. Can be modified for custom authenticators.

Display name

Same as Description.

UI

Indicates whether the authenticator comes with a user interface or not. Cannot be changed.

ACT

Indicates whether the authenticator shall be deployed by ACT deployment technology or not. If not checked, it means that the authenticator must reside in the correct bin folder for the supported platforms.

System authenticators are deployed as part of the installation, and the deployment options cannot be changed.

Type

Indicates whether the authenticator type is system (S) or custom (C).

Password policy

Prerequisite

Unless you are logged in to Agresso, no information is available.

Purpose

This window is used to configure the password policy for user login passwords on all clients. Password policies can be configured based on a flexible set of rules to allow the password strength to be tailored to match the organisation's security policies.

The password rules consist of:

- General settings for configuring the length of time the password is valid, the number of failed login attempts a user can make before being locked out of the system and how long the user is locked out etc.
- Basic rules which set the password length and whether the user's username can be included in the passport.
- Advanced rules which configure the use of upper and lowercase characters, digits and special characters.

Explanation of fields

The password policy configuration is divided into three sections: General settings, Basic rules and Advanced rules.

General settings

Password duration (days)

Sets the number of days the password is valid for before it expires. A value of 0

— Password duration is set and the password will expire after this time.

— Passwords are valid indefinitely.

Warn before expiry (days)

Sets the number of days before password expiry that an expiry warning will be sent to users.

— A warning is sent to the user that the login password will expire in the number of days set in the Password duration (days) field.

— No expiry warning is sent to the user.

Number of tries

Sets the number of unsuccessful login attempts a user can make with the wrong password before he or she is locked out of the system.

— A check is made on the number of unsuccessful login attempts using a wrong password and the user is locked out after too many failed attempts.

— No check is made on the number of unsuccessful login attempts using a wrong password and the user can continue to make login attempts without being locked out of the system.

Lockout duration (minutes)

Specifies how long users will be locked out of the system after making too many unsuccessful login attempts with the wrong password. After this time users can attempt to log in again.

Use advanced rules

Sets if the password policy makes use of the parameters in the Advanced rules section.

Basic rules

Minimum length

Sets the minimum number of characters that the user passwords must contain.

— A limit is placed on the minimum number of characters passwords must contain.

— No limit is placed on the minimum number of characters that passwords can contain.

Allow username in password

Sets if the user's username can be contained in the login password.

— The user's username can be included in the login password.

— The user's username cannot be included in login password.

Advanced rules

Number of rules to satisfy

Sets how many of the active advanced rules the password must satisfy to be accepted as valid.

Minimum uppercase letters

Sets the minimum number of uppercase letters the password must contain.

— A check is made on the number of uppercase characters contained in the password and the password must contain the minimum number specified to be valid. Setting the number to 0 when the field is enabled means that uppercase characters are not allowed in the password.

— No check is made on the number of uppercase characters contained in the password. The user can include uppercase characters in the password but their use is not mandatory.

Minimum lowercase letters

Sets the minimum number of lowercase letters the password must contain.

— A check is made on the number of lowercase characters contained in the password and the password must contain the minimum number specified to be valid. Setting the number to 0 when the field is enabled means that lowercase characters are not allowed in the password.

— No check is made on the number of lowercase characters contained in the password. The user can include uppercase characters in the password but their use is not mandatory.

Minimum digit characters

Sets the minimum number of digit characters (0 - 9) the password must contain.

— A check is made on the number of digit characters contained in the password and the password must contain the minimum number specified to be valid. Setting the number to 0 when the field is enabled means that digit characters are not allowed in the password.

— No check is made on the number of digit characters contained in the password. The user can include digit characters in the password but their use is not mandatory.

Minimum special characters

Sets the minimum number of special letters the password must contain.

— A check is made on the number of special characters contained in the password and the password must contain the minimum number specified to be valid. Setting the number to 0 when the field is enabled means that special characters are not allowed in the password.

— No check is made on the number of lowercase characters contained in the password. The user can include special characters in the password but their use is not mandatory.

Special characters allowed

Sets which special characters are allowed in user passwords.

Password Reset

Prerequisite

- Unless you are logged in to Agresso, no information is available.
- [Agresso Mail Queue](#) configuration must exist

Purpose

Allows you to select a mail profile (default) when responding to a *Reset password* request.

Users and Access

Purpose

Gives access to the [Self Service Start Page](#) node, which allows you to select a default start up page for Self service users.

Note that you still have use the Agresso Smart client (System administration) to assign access rights to users.

Main actions

None.

Self Service Start Page

Purpose

Unless you are logged in to Agresso, no information is available.

Purpose

You use this node to set a default start up page for Self service users, as an alternative to the general welcome page.

Note that you can only set up one default start page, valid for a set of selected clients and/or a set of selected roles.

Selections

First, you select the start page for a given (start) module. Next, you decide where to use it:

- If you select one or more clients, the start page will be valid for all users of the selected clients. If you select none, the start page will only be valid for selected roles (if any).
- If you select one or more roles, the start page will be valid for all members of the selected roles, irrespective of clients. If you select no roles, the start page will only be valid for users of selected clients (if any).
- If you don't select any clients or roles, the start page will not be used.

ACT

Purpose

Gives access to

- [ACT Setup](#) and
- [ACT TopGen Setup](#),

windows previously (before Milestone 4) only available via the **Agresso Smart client**.

ACT Setup

ACT

Agresso Customisation Tools (ACT) was introduced with Agresso 5.5 and consists of a set of basic interfaces, methods and events, interacting seamlessly with Agresso Platform components, the Smart client and the server.

Developers' tool

ACT is for developers only, who need to tailor the Agresso functionality to certain company specific needs, not covered directly - or in the right company context - in the standard version of Agresso Business World. In principle, ACT gives the external developer access to the whole of Agresso - from the inside.

ACT Setup

When you work with ACT, you will first and foremost utilize MS .NET Visual Studio Integrated Development Environment. There you will find the ACT libraries, and write your customisation.

When the customisation is complete, and when you need to administer the customisation in the Agresso context (install, upgrade, remove etc), you need to access the ACT Setup window.

Related topics

The following documents are available in .pdf format:

- Developer's Reference, Agresso Customisation Tools - Getting started - for developers of ACT components.
- **Reference Manual, Customisation Tools** - for personnel responsible for loading and setup of ACT components.

Assembly setup

General section

Assembly name

The name of the dll (assembly). Must be unique.

Status

The assembly's status. Valid values are:

N - Normal load.

F - Loaded from the file specified in Path.

L - Loaded from database.

D - Deploy.

S - Secure load.

C - Closed. The assembly will not be loaded.

The Find button is used to search for and list existing assemblies. If any text and wildcards are entered in the Assembly name and Status fields, these values will be used as search criteria.

Matching assemblies will be listed in the Assembly section.

The Assembly detail section

Name

Short name of the assembly.

Assembly name

Full assembly name, including version.

Description

Free text.

Status

Same as above.

Path

Relative or absolute path to where the .dll shall be stored. If blank, the default path will be Agresso executing directory.

File name

Name of .dll file.

File size

The .dll size.

The Assembly section (table columns)

The Assembly section lists information about all the currently loaded assemblies:

Name

Short name of the assembly.

Description

Free text.

Status

Current status - see above.

In database

Indicates whether the project is registered or not.

Upload

When clicked, the .dll will be uploaded to the Agresso database.

Client extensions

Two sections

The tab consists of two sections, where one row in the Extension section can be represented by one or more rows in the Usage section.

The Extension section

The following fields are found in the Extension section:

Assembly

Assembly name.

Class

Name of class that extends the ACT interface.

Description

Project description. Can be edited directly in the table.

Project type

Interface reference.

SeqNo

The sequence number deciding when this project shall be loaded - if more than one project are set up to be loaded.
Default value = 0 (random load sequence).

Status

Valid values are:

N - will be loaded.

C - will not be loaded.

The Usage section

The Usage section lists the various screens and menus

Screen

Indicates form id of the customised AGRESSO window - if the main class extends the IProjectForm interface. If the main class extends the IProjectCustomForm interface, the custom form id is displayed.

Menu ID

Any given menu Id.

System setup

The system setup code.

Status

The status for this use of the selected project. Valid values are:

N - will be loaded.

C - will not be loaded.

R - Required (compulsory). Must be loaded if AGRESSO - or the form - shall start.

TopGen extensions

Two sections

The tab consists of two sections, where one row in the Extension section can be represented by one or more rows in the Usage section.

The Extension section

The following fields are found in the Extension section:

Assembly

Assembly name.

Class

Name of class that extends the ACT interface.

Description

Project description. Can be edited directly in the table.

Project type

Interface reference.

SeqNo

The sequence number deciding when this project shall be loaded - if more than one project are set up to be loaded.
Default value = 0 (random load sequence).

Status

Valid values are:

N - will be loaded.

C - will not be loaded.

The Usage section

The Usage section lists the various screens and menus

Screen

Indicates form id of the customised Agresso window - if the main class extends the IProjectForm interface. If the main class extends the IProjectCustomForm interface, the custom form id is displayed.

Menu ID

Any given menu Id.

System setup

The system setup code.

Status

The status for this use of the selected project. Valid values are:

N - will be loaded.

C - will not be loaded.

R - Required (compulsory). Must be loaded to avoid error messages.

Framework extensions

Two sections

The tab consists of two sections, where one row in the Extension section can be represented by one or more rows in the Usage section.

The Extension section

The following fields are found in the Extension section:

Assembly

Assembly name.

Class

Name of class that extends the ACT interface.

Description

Project description. Can be edited directly in the table.

Project type

Interface reference.

SeqNo

The sequence number deciding when this project shall be loaded - if more than one project are set up to be loaded.
Default value = 0 (random load sequence).

Status

Valid values are:

N - will be loaded.

C - will not be loaded.

The Usage section

(Manager)

Name of the manager class.

(Entity)

Name of managed entity.

Status

The status for this use of the selected project. Valid values are:

N - will be loaded.

C - will not be loaded.

R - Required (compulsory). Must be loaded to avoid error messages.

Server extensions

Two sections

The tab consists of two sections, where one row in the Extension section can be represented by one or more rows in the Usage section.

The Extension section

The following fields are found in the Extension section:

Assembly

Assembly name.

Class

Name of the class that extends the ACT interface.

Description

Project description. Can be edited directly in the table.

Project type

Interface reference.

SeqNo

The sequence number deciding when this project shall be loaded - if more than one project are set up to be loaded.
Default value = 0 (random load sequence).

Status

Valid values are:

N - will be loaded.

C - will not be loaded.

The Usage section

Report

Name of the server process or report.

Variant

Any specific report variant.

System setup

The system setup code.

Status

The status for this use of the selected project. Valid values are:

N - will be loaded.

C - will not be loaded.

R - Required (compulsory). Must be loaded if the server process shall start.

Named events

Table

The following columns are displayed:

Assembly

Assembly name.

Class

Name of class that extends the ACT interface.

Event family

The family name – given by the developer.

Event name

Event name – as given by the developer.

Description

Description – as given by the developer.

Project type

Currently not in use..

SeqNo

The sequence number deciding when this project shall be loaded - if more than one projects shall be loaded. Default value = 0 (random load sequence).

Status

The project's status. Valid values are:

N – Normal. Will be loaded.

C – Closed. Will not be loaded.

Service extensions

Two sections

The tab consists of two sections, where one row in the Extension section can be represented by one or more rows in the Usage section.

The Extension section

The following fields are found in the Extension section:

Assembly

Assembly name.

Class

Name of class that extends the ACT interface.

Description

Project description. Can be edited directly in the table.

Project type

Currently not in use.

SeqNo

The sequence number deciding when this project shall be loaded - if more than one project are set up to be loaded.
Default value = 0 (random load sequence).

Status

Valid values are:

N - will be loaded.

C - will not be loaded.

ACT TopGen Setup

Purpose

You use **ACT TopGen Setup** for two purposes:

1. Load your custom screen definitions into the Agresso system (database) and map it to a menu item.
2. Set up rules for use in the Agresso clients.

General functionality

The window has three sections:

- You use the Search section to load screen definitions already mapped to a menu item. All fields are used as search criteria, and wild-cards are allowed.

- The Location details section displays information about the location of a TopGen definition. It can either be in the database or in a file.
- The TopGen page location section holds mapping information between TopGen screens, their aliases and menus where they shall be available.

A note about ACT and component loading

When a menu item (identified in Menu id) is activated, it will be linked to the screen definition in the Alias column, but ACT will load the screen definition identified in the Frame name column. Normally, therefore, the content of Alias and Frame name will be identical.

Example

You can test a new screen by using an existing menu item and the standard screen definitions as alias, but actually load your own screen - identified in Frame name - when the menu item is activated.

Explanation of fields

Search section

You use the following fields to restrict searching for previously defined mapping details:

Frame name

TopGen screens previously defined in the Frame name column in the TopGen location table.

Alias

TopGen aliases previously defined in the Alias column in the TopGen location table.

Menu id

Menu ids previously defined in the Menu id column in the TopGen location table.

Status

Status for the row of mapping details. Valid values are Active and Closed.

Location details section

This section is only of practical interest if you, for some reason, keep the TopGen definition in a file.

The following fields are available:

Storage

Storage type. Can be database or file.

Path

Path to file (when storage is File).

File name

Name of file.

TopGen page location section

The following columns are available:

Frame name

Name of the TopGen screen - as defined in the TopGen editor. When the menu id is activated, this screen will be loaded.

Alias

Name of the TopGen screen that is identified in the menu. Normally the same as the Frame name (default value).

Menu id

A menu id where the mapping details shall be valid.

Status

Current status for the row of mapping details.

Buttons: The buttons are explained as follows:

Add

Adds a new row for mapping details.

Delete

Deletes selected rows. Underlying screen definitions are not affected.

Delete from database

Removes selected rows from the setup. The underlying screen definition(s) will be removed from the database.

Menu commands

Tools menu

Import

The import command lets you browse the file system and select one or more TopGen screen definitions for import.

New menu

The New menu command is only active if you have at least one row in the TopGen page location section. The row can be empty.

New menu opens the User defined Self service menu page, allowing you to add new menu items - for your custom screens - to the Self Service menu.

Related topics

The [User defined](#) Self service menu page

Help Configuration

Prerequisites

1. Unless you are logged in to Agresso, no information is available.
2. Agresso WebHelp must be installed on a Web server, available for all users. See [WebHelp Installation](#).

Purpose

Help Configuration gives you access to the help systems for the various Agresso clients via child nodes:

- Smart Client Help (SCHELP).
- Self Service Help (SSAHELP).
- Self Service (Experience Pack) Help (X2HELP).

When you select a child node, you can publish - or withdraw - the relevant Help.

HTML based help system

Agresso WebHelp is an html-based help solution compatible with all major browsers.

Installation paths and language

There may be several language variants of the Help system, located in language specific folders.

Examples: The URL for the English (UK) version can be: <http://<Web Server name>/SCHELP/EN/SCmaster.htm>.

The URL for the Swedish version can be: <http://<Web Server name>/SCHELP/SE/SCmaster.htm>.

Note: Make sure you remember the server name as well as the virtual path (URL) to the Agresso Web Help system!

Database Tools Overview

Main actions

None. You will use one of the child nodes:

- **[Database Copy](#)**, offers tools for copying data to and from an Agresso database.
- **[Run Scripts](#)** (Agresso SQL Scripts), allows you to run scripts in Agresso SQL (ASQL) directly from the Management Console.
- **[Agresso Index](#)**, allows you to regenerate Agresso indexes.
- **[Update Manager](#)**, allows you to run specific update scripts enclosed in dlls.
- **[Blob Viewer](#)**, allows you to browse, view and delete blobs in the Agresso database.

Database Copy

Reference

For all details about Database Copy, please refer to the documentation starting with [Agresso Copy](#) in the Appendix.

Purpose

Agresso Copy is used to load tables into a new Agresso database (Copy In) and to copy tables from one database to another (Copy Out). The tools are based on the same functionality as the CopyMS and CopyOra command line based tools. AMC adds a user-friendly layer on top of the basic tools to make them easier to use.

Note: Some of the more advanced options found in the basic tools might not be available from within AMC.

Run Scripts

Reference

For all details about the Agresso SQL Scripts (ASQL), please refer to the documentation starting with [Agresso SQL Program Overview](#) in the Appendix.

Purpose

The **Run Script** database tool is used when you need to execute SQL statements written in either Agresso SQL syntax or native database syntax. The statements are stored in one or more ASCII file with [.asq](#) file extension.

Update Manager

The Update Manager is a general tool for database updates of Agresso system tables between service packs. The updates are packed as version controlled dlls, and will either come on the installation DVD (for example: Localisation updates) or be made available from our web site.

The use of Update Manager is quite straightforward: First, you browse for the update file ([xxx.update.dll](#)) and add it to the **Update Manager** node. Next, you run the update.

Blob Viewer

The Agresso Blob Viewer allows you to browse for, view and delete blobs found in your Agresso database.

The Blob Viewer will display all the Agresso tables containing blobs. When you select a blob table node, AMC will list all rows found in the blob table in the main view. Selecting one of these nodes enables you to execute, save or delete the blob you have selected.

Database Copy

Purpose

Copy table data to files - or data from files to Agresso tables.

Main Actions

None. You will use one of the child nodes for copying:

- [Copy In](#)
- [Copy Out](#)

Copy In

Purpose

Copy data and indexes stored in files into the database. You will therefore always start by selecting either a file list (*.lst file) or data files representing Agresso tables (a*.* files).

The selection can be saved by right-clicking the Copy In node and select "Save to File".

The last selection will automatically be saved to a file named "_cifiles.lst" when the copy operation is started.

Main actions

The following main actions are available:

- **Add From List File**

Add a predefined selection of data files to the selection list. The function let you to browse for an .lst file containing a list of datafile names. All files listed in the .lst file will be populated in the selection view (right pane).

- **Add From File(s)**

You browse for one or more data files and add them to the selection list. You can save a set of already loaded data files to disk, as a .lst file, for later use (with Add From List File).

- **Start Copy...**

Starts the Agresso Database Copy Wizard. This option is only available when one or more files are selected.

- **Rerun Copy**

Allows you to run the Agresso Database Copy Wizard with the same settings as last Start Copy....(the **Finish** button is enabled on the wizard's first page).

Copy Out

Purpose

Allows you to copy data from a selection of database tables, to files in a folder.

Main Actions

You will first make a selection of tables to be included in the copy operation. The selection can be defined in a .lst file, or you can browse the database for tables to include. You can then copy the selected table definitions and data to the file system.

The following main functions are available:

- **Add From List File**

Add a predefined selection of tables to the selection list. The function let you to browse for an .lst file containing a list of table names.

- **Add From Database**
Select tables directly from the database.
- **Start Copy..**
Starts the Agresso Database Copy Wizard. This option is only available when one or more tables are selected.
- **Rerun Copy**
Starts the Agresso Database Copy Wizard with **Finish** button enabled from the first page. Allows you to start the copy operation using the same options used in last **Start Copy...**.

Advanced settings

Note the following advanced options:

- If you select **Treat views as tables** Views are copied out in the same way as tables. Note, they can not be copied in as views again.
- If you select **Turn off trigger warnings** and the table being copied contains a trigger, the definition is printed to the output window.

Run Scripts

Purpose

Run one ore more scripts.

Main Actions

The following actions are available when right-clicking the **Run Scripts** node:

- **Add From File List**
Browse for a **.lst** file containing a list of script-file names. All files listed in the selected file will be displayed in the main view.
- **Add File(s)**
Browse for files to add to the selection.
- **Start**
Starts the **Run ASQL Scripts Wizard**. Only available when one or more script-files are loaded.
Advanced settings: The wizard has a tab with Advanced settings which allows you to terminate the current session if errors are encountered. This is only used when running more than one asq file.
It is used in conjunction with the "on error exit" command. The scope of the "on error exit" command is only the current file. Thus, if an error occurs after this command, **asql.exe** will skip the rest of the current file, but continue with the next one. However, if you want **asql.exe** to cancel all files when an error occurs, use this parameter.
- **Rerun**
Allows you to rerun the last script file(s).

Agresso Index

Purpose

The **Agresso Index** tools enables you to regenerate indexes defined in **aysindex** (system indexes) and **aagindex** (user defined indexes).

If a defined index is missing, **Agresso Index** will re-create it for you. The **Agrindex.exe** application can also be run from command line as described in the [appendix](#).

Main actions

The following actions are available:

- **Add tables** - collect the tables you want to regenerate indexes for (if no tables are selected, indexes for all tables will be regenerated)
- **Remove from selection** - allows you to remove one or more loaded tables.
- **Remove all** - removes all loaded tables.
- **Start** - regenerates indexes for the loaded tables.

Update Manager

Purpose

Add an update package to the Agresso installation.

Update package files

The update manager works with update packages, one at a time, and an update package always comes as an `*.update.dll` file. The packages can either come on an upgrade DVD, or they can be downloaded from a web site.

Updates overview

The [Database Information](#) node will give you an overview of all updates.

Main action

The following action is available:

- **Add Update Package**. A wizard will guide you through the process.

Related topics

For more information about update packages and steps, see [Update Manager details](#).

Database patches and version control

The [Data source](#) node (**Configuration** tab) in the [Agresso Management Console](#) contains detailed database version information:

The [Update level](#) property gives you the version and name of the last successfully applied update (patch).

Update Manager details

Overview

This topic describes two important aspects of [Update Manager](#):

- *The Update package*, consisting of update steps and properties.
- *The Database update storage* (database tables used by [Update Manager](#))

Update package with steps

Package structure

An *update package* is the main object for the Update Manager, and contains one or more *update steps*. The diagram below shows the structure of three update packages, with various steps:

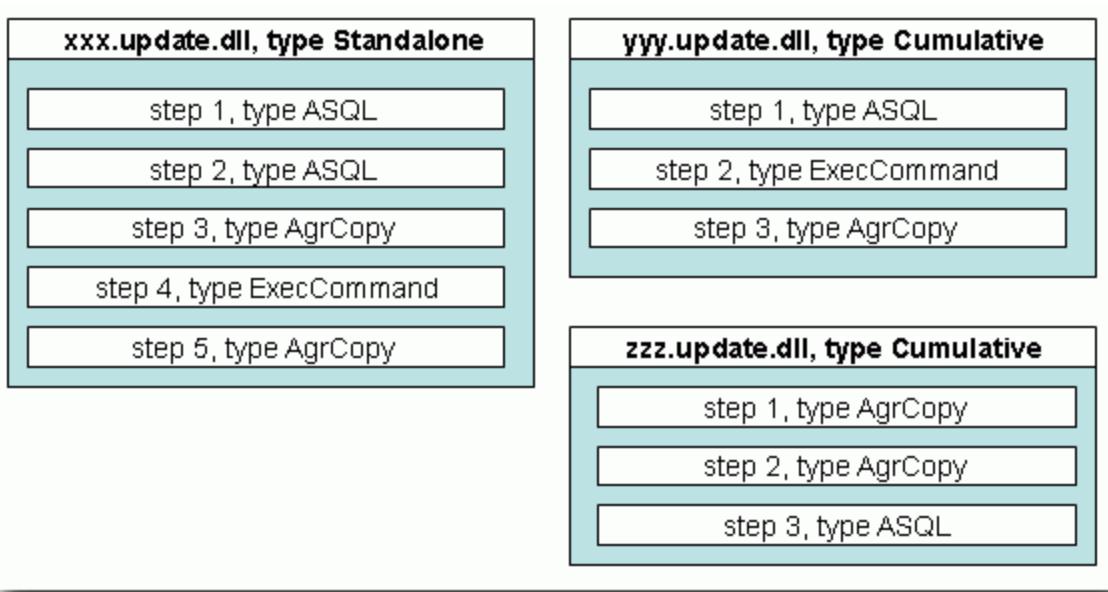


Diagram comments: The diagram illustrates that there are two types of packages, either Standalone or Cumulative. Cumulative package types will update the Update level property for the detailed database version, and cumulative packages must be added in the correct sequence. A cumulative package that requires an update to be run first, will have the status **On Hold**.

The diagram also illustrates that a step can be of any of three types:

- ASQL - a script file
- AgrCopy - an Agrezzo Copy data file
- ExecCommand - a command file or an executable.

Package properties

When you open the **Update Manager**, the loaded updates (not necessarily executed yet) will be listed in the main panel and also appear as child nodes to **Update Manager**.

By right-clicking one of the child nodes, you can select **Properties** to get additional information about the selected package.

Step properties

General properties: By selecting an update package node, the main properties for the various steps will be listed in the main view.

Step properties: You can select **Properties** to get additional information about the selected step. The following properties are available for all step types:

Step property	Description
Allow change	You are allowed to open the Step for edit before run
Allow error	Flags if a Step is critical. An Update Package cannot finish if a critical step fails. Contact Agrezzo Maintenance if this happens. Agrezzo Maintenance will send a new Upgrade Package to solve this issue.
Allow re-run	The Step can be re-run without creating any harm to your data.

The following properties are available for steps of type AgrCopy only:

AgrCopy step property	Description
Re-create views	The step will re-create views based on definitions in the aagview when turned on.
Re-create procedures and triggers	The step will re-create procedures and triggers when turned on.
Use Where-clause in data file	The step uses a Where-clase in the AGRESSO Copy file to target update of special rows.
Append data only	AGRESSO Copy will only append data to the table and will not delete any existing rows.
Additional Parameters	Additional parameters used to control the AGRESSO Copy process.

Step actions: In addition to properties, the menu offers the following options:

- **Save Log to File** – Export the log file to the file system.
- **View/Edit** – View or Edit the step. You can modify the step, if the step allows changes.
- **Uninstall** - Only available for experience packs

Database update storage

Description

The **Update Manager** stores all update packages as blobs and keeps history and logs for all applied database updates.

The use of blobs assures that no information or files are stored locally, at the computer running AMC. Once an update package is added to the **Update Manager**, it is available from any computer running AMC. This also applies to the log files.

Tables in use

Update package information: The table `acrupgradehead` holds header information about all database updates added along with status information.

Update step information: The table `acrupgradedetail` holds detailed information about each step in the update package.

Log files and step modifications: The log files, as well as information about modified steps, are stored as blobs in the table `acrupgradeblob`.

These tables may be collectively referred to as the *Database update storage*.

Blob Viewer

Prerequisite

Unless you are logged in to Agresso, no information is available.

Purpose

Gives you access to all blob tables.

Main actions

None. All actions are for the blob tables or a specific (selected) blob.

Actions on blob table

When you select a blob table node, the table content (the blobs) will be listed in the main panel. You get access to two main actions:

New - allows you to add a file to the table.

Export - allows you to export all blobs as files. Default output folder is My Documents\Blob Export\<table name> (can be changed).

Note that you can use the **Export formatting tags** in the wizard to construct a **Format** for the output file names, for example like this:
[ID]. [TYPE].

Actions on a selected blob

When you select a blob in the main table, the following main actions are available:

Properties (blob details)

Delete

Export (i.e. save as file)

View (open in associated viewer)

Update (i.e. replace the current content with another file.)

Upgrading from 5.6.X to 5.7.0

General upgrading procedure

This section describes the required steps to upgrade from a 5.6 version to 5.7.0.

Prerequisites

- The new version of **Agresso Business World** must have been successfully installed before you start the upgrade process.
- We assume that you have profound knowledge of the Agresso installation and the RDBMS used.

Prepare the upgrade

Backup the database

Before you run the upgrade wizard, you should back up your Agresso data. To be completely safe, you should also use the **Agresso Copy** program to make a complete copy of all database tables.

Verify disk space

Available disk space needed for the upgrade is approximately the size of the largest table + 10%.

See the list of table changes in the Appendix.

Create a Server Data Source and initialize Business Server environment

Use the **Agresso Management Console** (AMC) to create a new data source connected to the old database.

When the connection is up and working, you must also initialise the Business Server environment (select the **Business server** node in **AMC**) and then **Initialise Business Server**).

Now you can run the upgrade wizard. See [Upgrading Procedure 5.6.x to 5.7.0](#).

Amendment tables

Logging

If amendment logging is turned on for an Agresso table, an amendment table (or shadow table) are created and continuously updated with all table changes.

Note: During upgrade, all amendment logging will be turned off (by the wizard), and switched on again when the convert script is complete.

Table for amendment tables

All amendment tables are defined in the table [aagamendlog](#).

Naming standard: An Agresso table name is constructed from the structure `a<module><identification>`, while an amendment table is extended with the letters `shd` between `<module>` and `<identification>`.

If you turn on amendment logging for `acrclient`, the amendment table `acrshdclient` will be created and added to [aagamendlog](#).

Recreate active logging triggers

If some of the shadow tables are not converted correctly, and the upgrade wizard is unable to fix the error in the database check. The shadow tables can be re-created by renaming or dropping the existing table, and then re-enable amendment logging for the table in the **Activation of logging server** (AG30) page in the Smart client.

When errors occur

It is important that all errors encountered during the upgrade process are reported back to UNIT4 Agresso.

If you, as a partner or subsidiary, have access to the SDE system, report the errors as SDE tickets.

*Do **not** report the problem if it is related to duplicates in the database, full disk, full tablespace or similar!*

Check user defined queries after upgrading

As part of the system upgrade, the ASQL parser has been improved with stronger syntax check and more consistent error handling. If a complex query contains an error, no part of the query will be executed. Instead, ASQL will report a parser error.

Previously, valid ASQL appearing before the error would have been executed, while the rest would be ignored. Now, the query will not run at all.

It is therefore important that you run all user defined queries after the upgrade.

Upgrading procedure 5.6.x to 5.7.0

The upgrade wizard

You will use the wizard located in

[.\DatabaseScript\DbTypeUpgrade\UpgradeWizards.exe.](#)

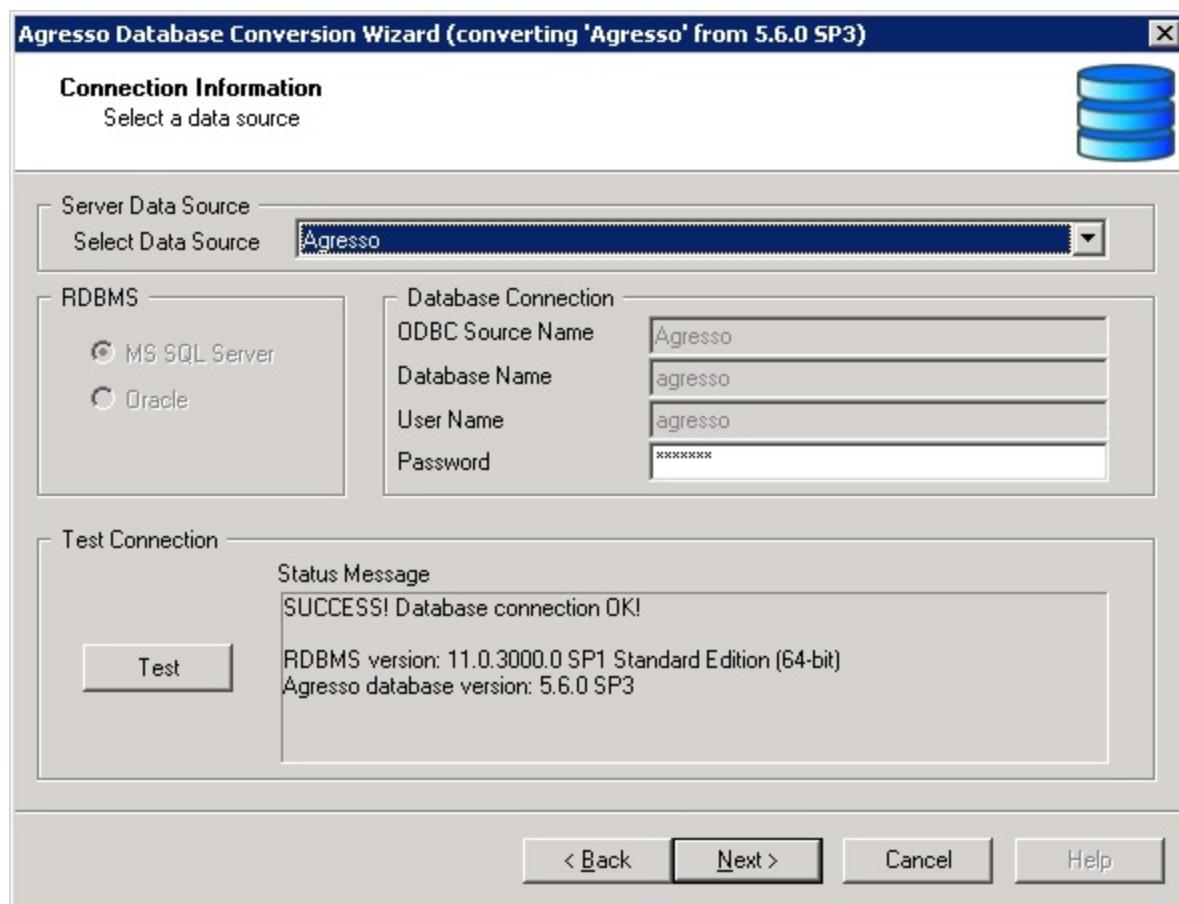
Start the Wizard

- A. Start [UpgradeWizards.exe](#)
- B. Select **Convert from 5.6**

Procedure

1. When the upgrade wizard is up and running, select Step 1, [Main Upgrade Wizard](#). Click **Next** to display the **Connection Information** dialog

 Upgrade Wizard - Connection Information

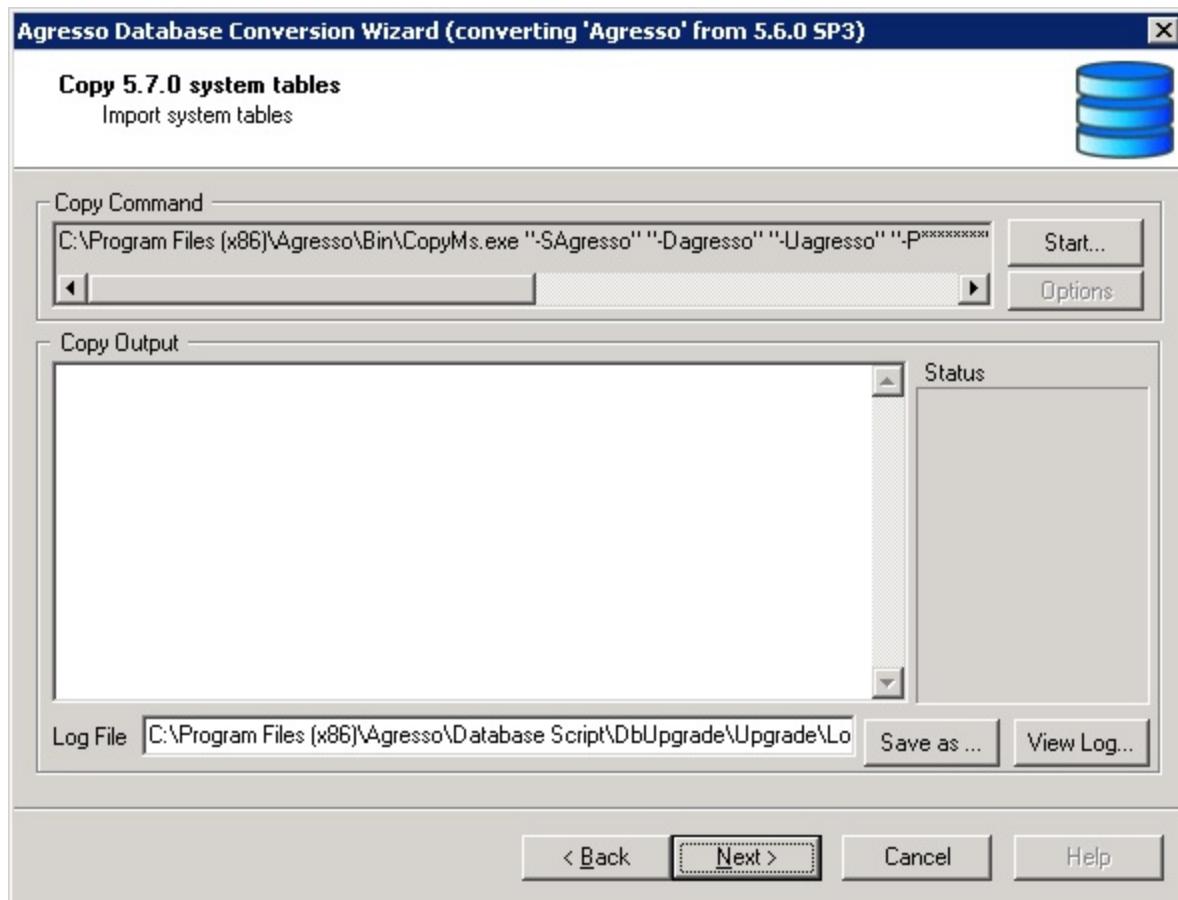


2. Select the data source, enter password, and click **Next**.

If all goes well, the wizard will display the conversion database name. Click **Next** to continue.

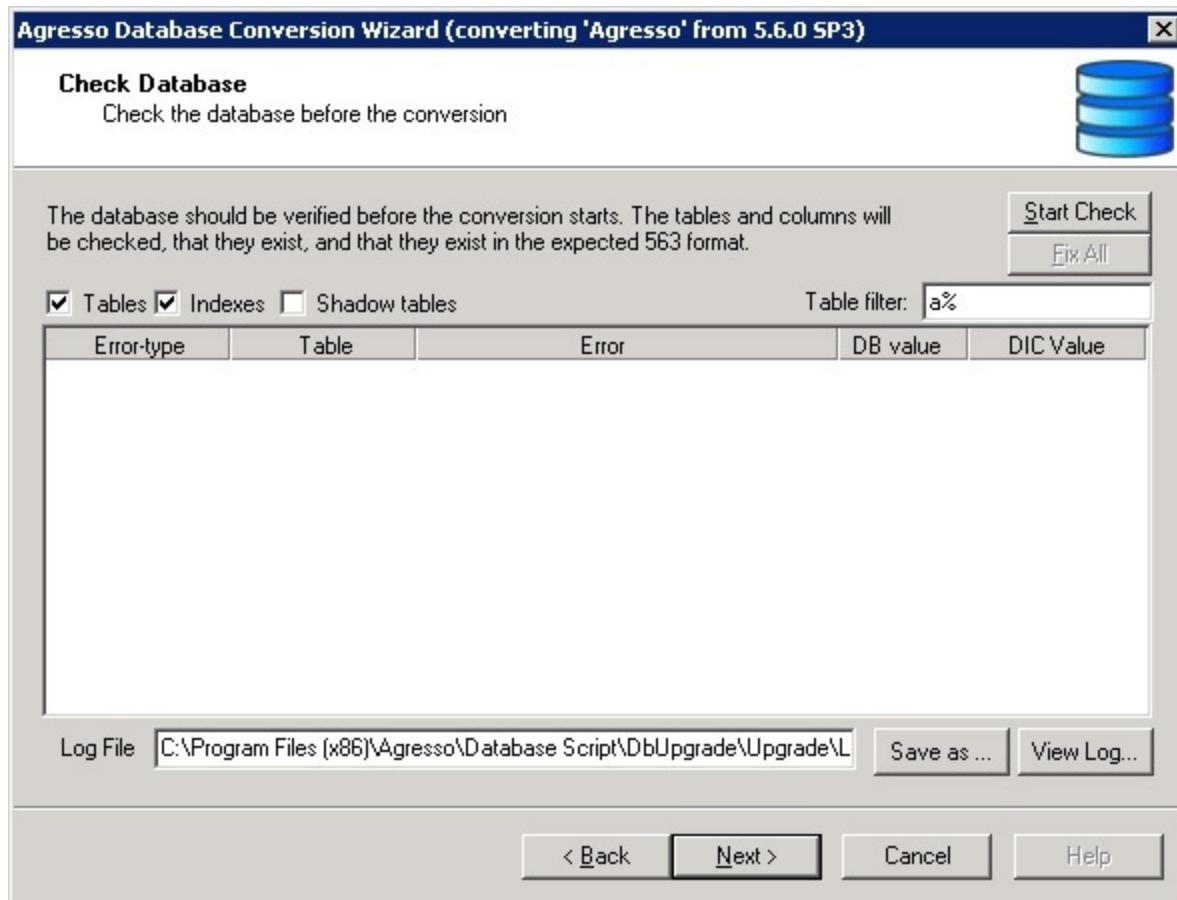
The wizard takes you to the **Copy system tables** step.

 Upgrade Wizard - Copy in system tables



3. Click **Start** to recreate the system and directory tables. When completed, click **Next** to continue with **Check Database**.

 Upgrade Wizard - Check Database



This step verifies that the database is in the expected 5.6 format. This check might take some time.

4a. Click **Start Check**

If you find differences, try the **Fix All** button. The Fix All might take some if tables with a lot of data needs to be fixed.

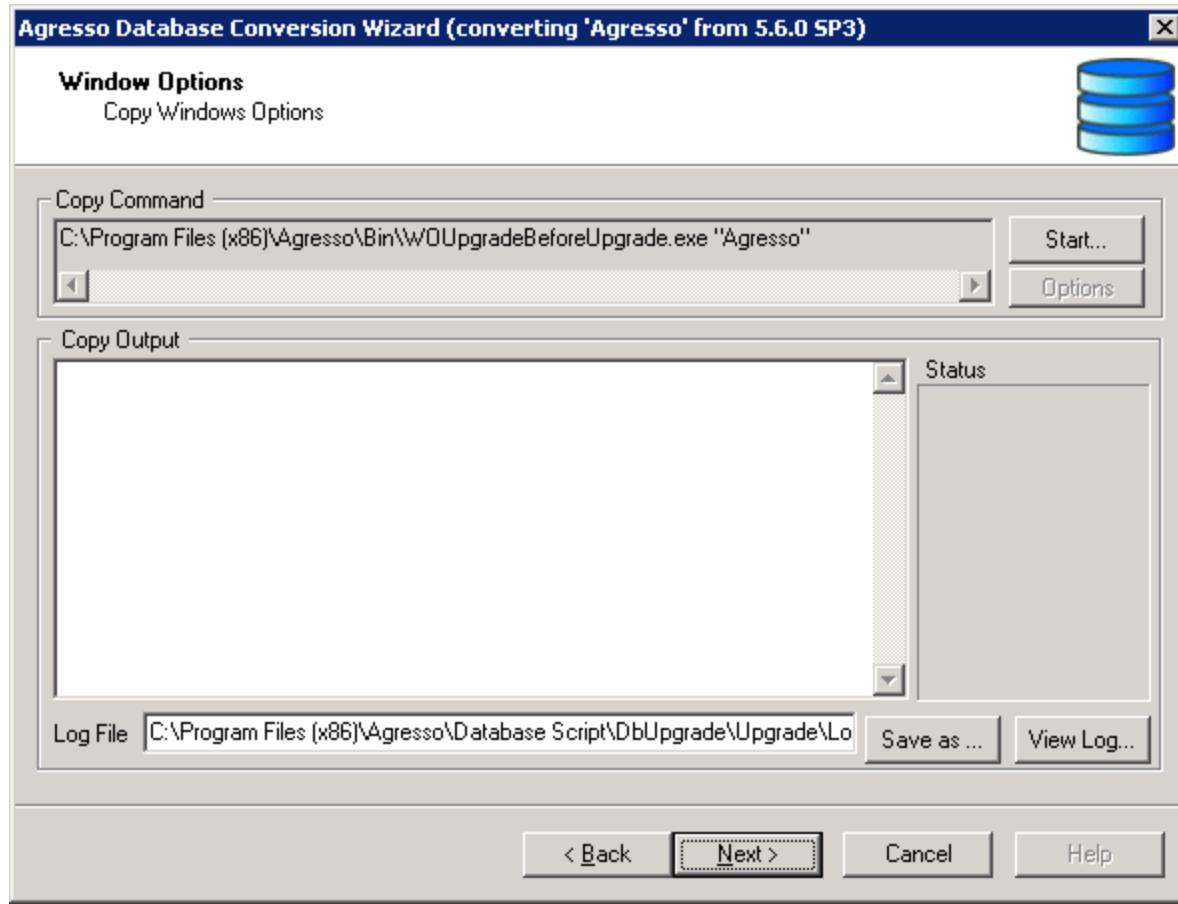
It is important to have a fixed database before continuing with the upgraded.

The following differences can be ignored:

- Columns in the table are not found in the dictionary.
- Columns are longer than expected.
- Errors on indexes (indexes will be recreated in a later step).

4b. Click **Next** to continue. You are ready to copy window options.

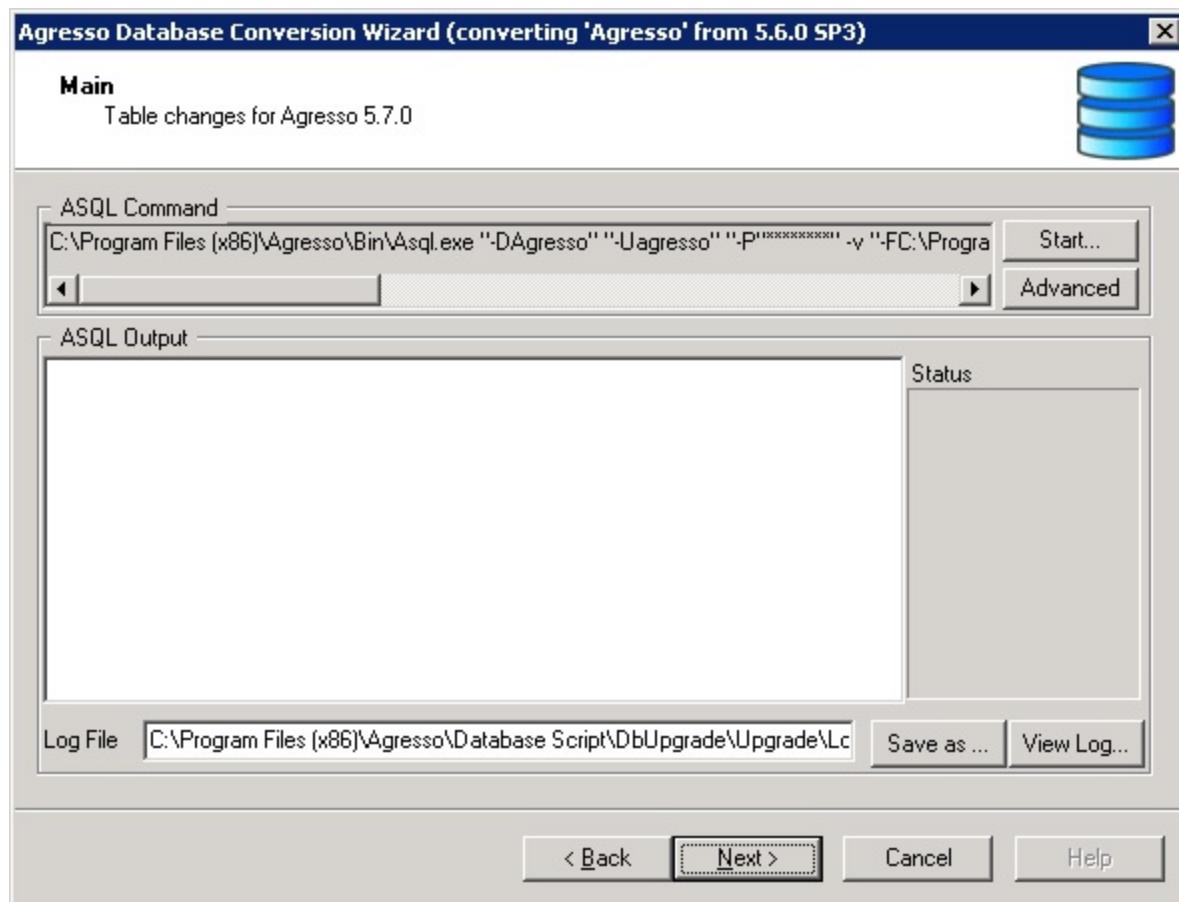




5. Click **Start** to run the script, then (when the script is finished) click **Next** to continue.

You can now introduce the main changes in the database structure:

Upgrade Wizard - Main table changes

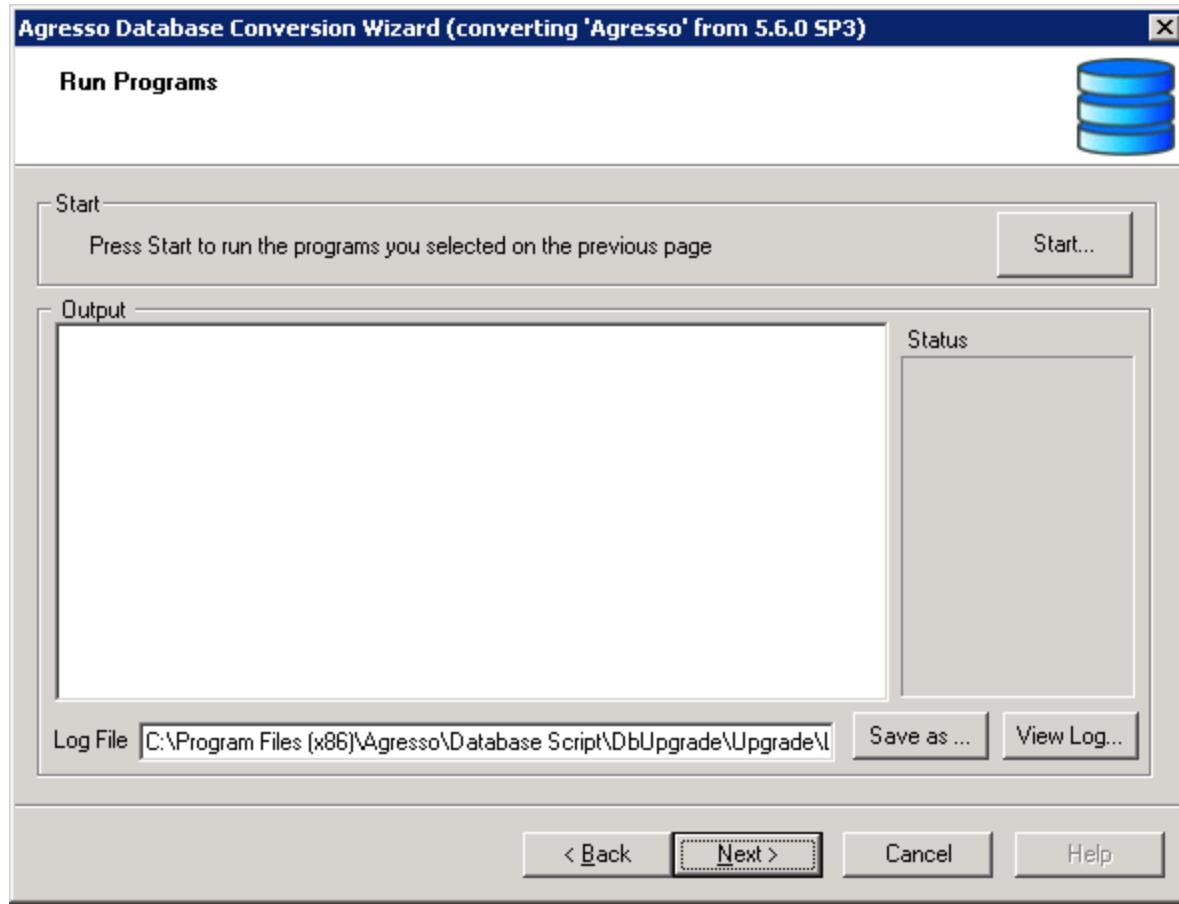


6. Click **Start** to run the script, then (when the script is finished) click **Next** to continue.

The next step is named **Select programs to run**. The available options depend on the version you are upgrading from.

As a rule, you should let the wizard make the selection for you, then click **Next**. This will take you to the **Run programs** step:

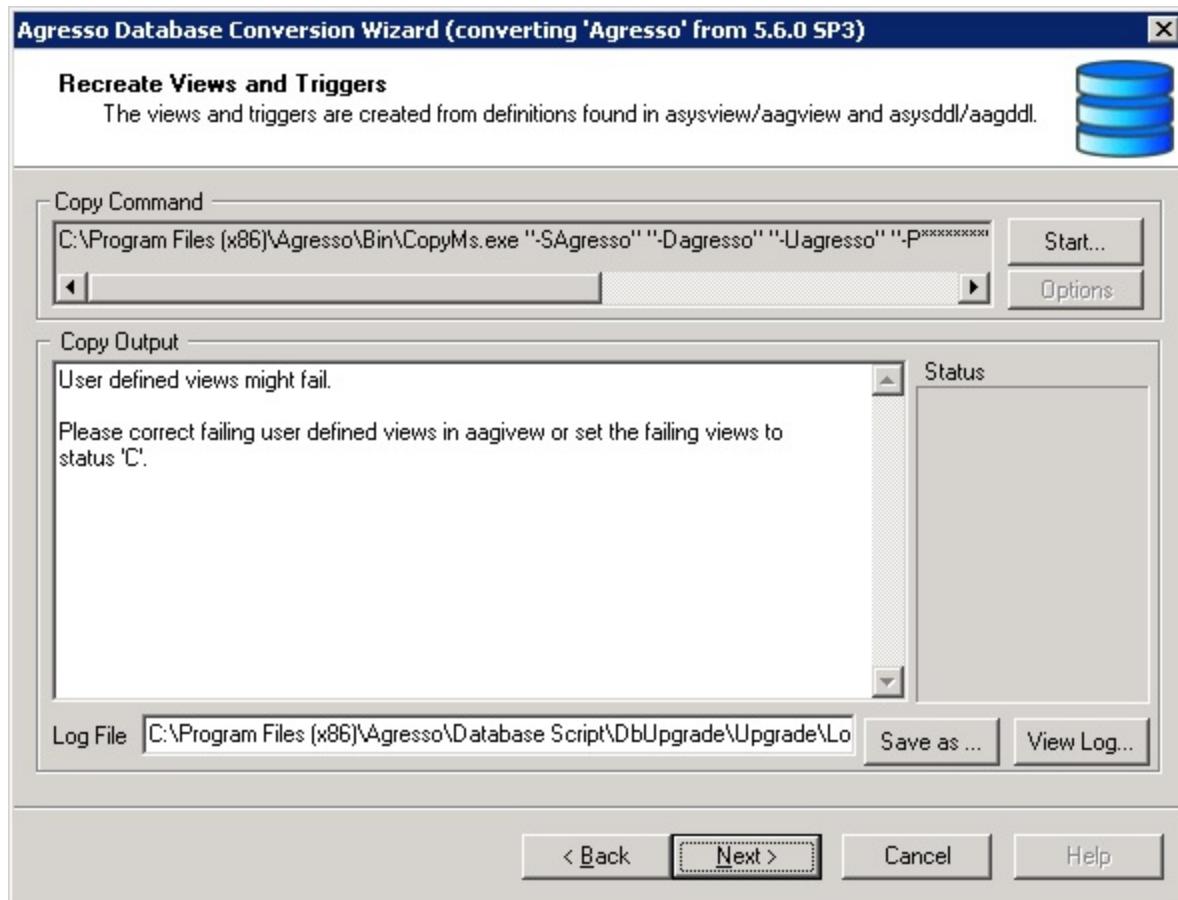
Upgrade Wizard - Run programs



7. Click **Start** to run the selected program. When finished, click **Next**.

This will take you to the **Recreate Views and Triggers** step.

Upgrade Wizard - Recreate Views and Triggers

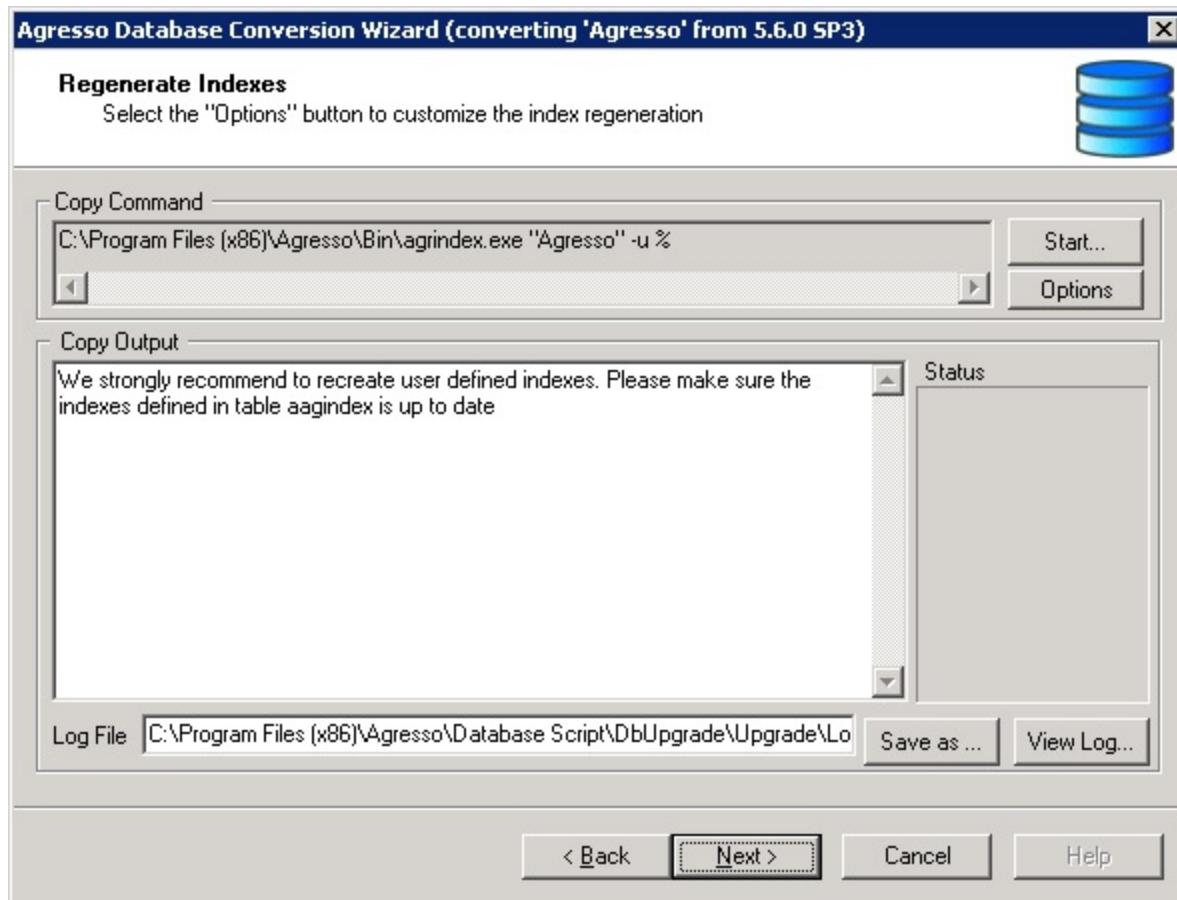


User defined views and triggers might fail if data structure has been changed in the latest release on the tables referred to. Definition of the views and triggers

can be found in the tables aagview and aagddl. See in the Appendix for table/column changes.

- 8.** Click **Start** and **Next** in the next windows, in order to restore system and user defined views and triggers. This will bring you to the final step.

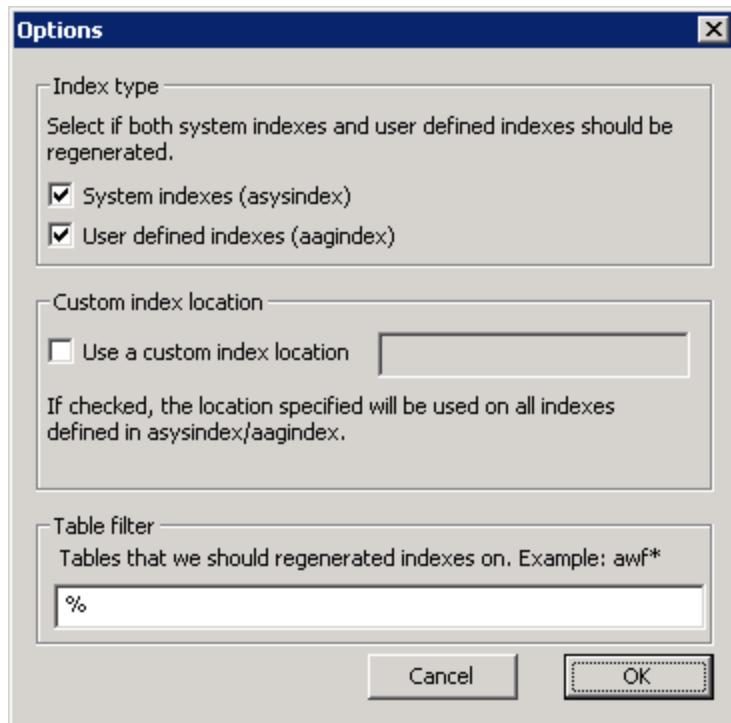
Upgrade Wizard - Regenerate Indexes



9. Click **Start** to regenerate the indexes. The indexes are important for performance and for preventing duplicates in the database.

Duplicates: If an index is created without the unique flag due to duplicate rows, remove the duplicates, and rerun **Recreate indexes**.

10. Click the **Options** button to open the Options dialog:

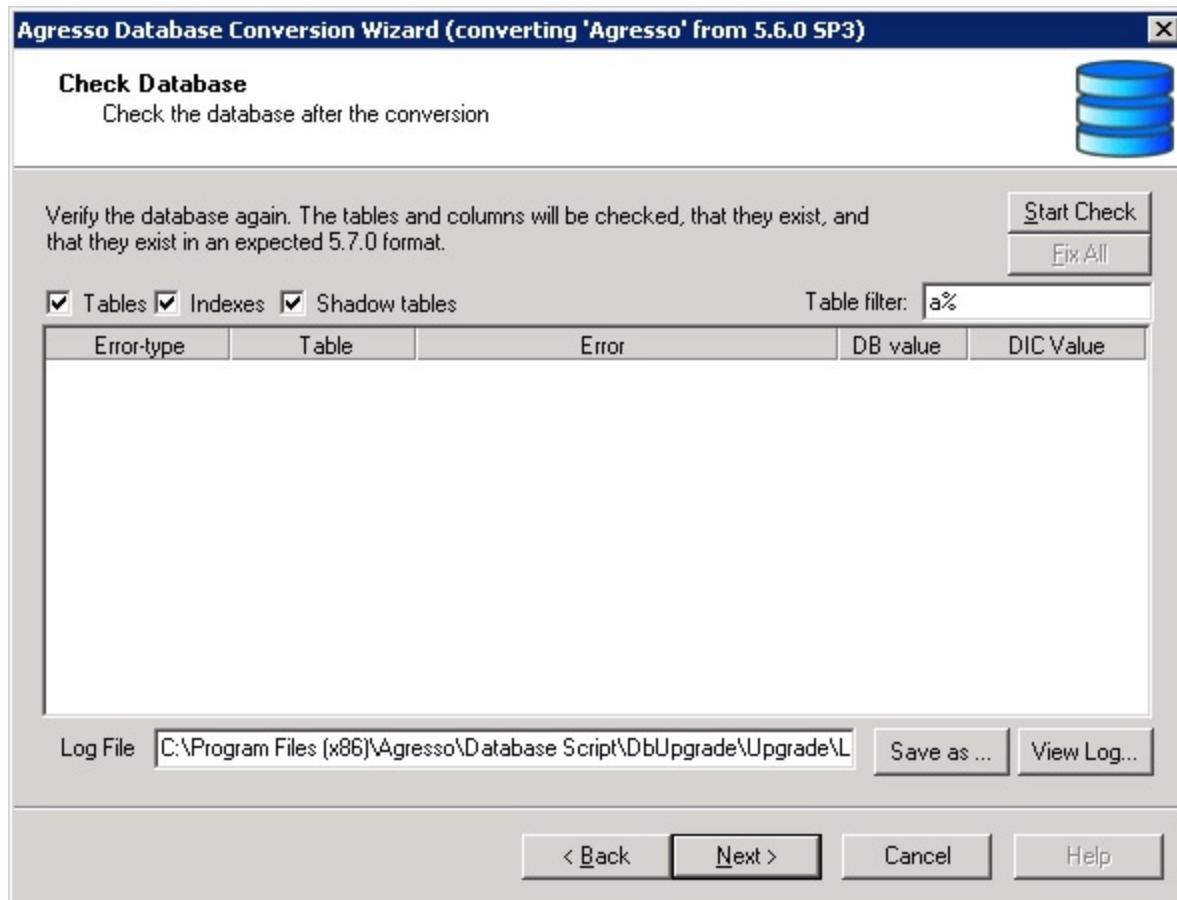


A note about indexes

User defined indexes (stored in [aagindex](#)) are recreated by default. Due to table changes, you should also change these, to avoid reduced performance. For details. see [AgrIndex](#).

- 11.** Select your options and click **OK**. Back in the **Regerate indexes** dialog, click **Next**. You should now check the database after con-
version.

Upgrade Wizard - Check Database



12. Click **Start Check**.

Finalise Upgrade

Overview

To complete the upgrade, you should:

- Check the converted Browser templates, and correct any errors.
- Clean up duplicates and add indexes.
- Correct all user defined views due to the table changes (see Appendix, Agresso Data Dictionary)
- Remove all tables no longer in use.

Check and correct Browser templates

Tools are provided to administer and ease the upgrade process for Browser templates:

- A Browser checker utility, BrowserTemplateChecker.exe, introduced in 5.5 Service Pack 1 that is run after upgrade.
- Check the log file produced during the upgrade. In this log, detailed information can be found of the Browser templates which need to be adjusted to run correctly on the upgraded version of UNIT4 Agresso. To fix the problem, open the browser template, make the necessary changes and save.

Indexes and duplicates

When running the database check step in the wizard, there might be indexes missing due to duplicates.

Indexes might be very important, and missing or wrong indexes can lead to very poor and slow performance.

See log file .\Database Script\DbUpgrade\Upgrade\Log\logindex_<database>.log

Use any database tool to find the duplicates (for example SQL Server Management Studio (mssql), SQL Developer (Oracle)).

Example on how to find the duplicates:

```
select distinct client, attribute_id from agldimension
group by client, attribute_id
having count(*) > 1
order by client, attribute_id
/
```

Delete/change so there are no duplicates in the table, and re-create the index.

Look up in the Appendix, Agresso Data Dictionary to see the correct index definition. Create the index using the preferred DBA Tool.

User defined views

The table structure is changed from release to release. Ensure all user defined objects are changed according to the new structure (See Appendix Agresso Data Dictionary)

Change the definition and recreate the views

See log file .\Database Script\DbUpgrade\Upgrade\Log\logviews_<database>.log

Amendment logging

Amendment logging triggers needs to be recreated after the upgrade. Use the Smart client to open the **AG30 Activation of logging server** from **System Administration | Data Control | Amendment logging**, and regenerate active logging triggers.

Remove tables not longer in use

The script *drop.asp* in the *Scripts* directory will drop all old and temporary tables no more used by the application.

Note: This script must be run when the upgrade has been completely verified, and you are sure that none of the old tables are needed for backup purposes.

Create System Attributes

Description

The wizard **Create System Attributes** will generate all the fixed attributes currently missing in your Agresso installation.

The wizard is part of Step 6 - **Run binary programs** in the main system upgrade wizard, but is also available in a stand alone version, available in the file *CreateAttributesWizard.exe*.

Standard (fixed) attributes only

The wizard updates only attributes delivered as part of the standard **UNIT4 Agresso** installation (with attribute id from **A0** to **LZ**).

Other attribute types, for example localisation specific attributes or custom attributes, will not be updated by the wizard.

All clients will be updated

Attributes are client dependent and the wizard will generate one attribute instance per client. Attribute names and descriptions will comply with the selected language (Company Information screen, CR01) for the various clients.

Run the wizard

The wizard is very straightforward, with no complex options. Just follow the instructions on screen.

NOTES ON UPGRADING FROM 5.5.X TO 5.7.0

Prerequisites

- The new version of **Agresso Business World** must have been successfully installed before you start the upgrade process.
- We assume that you have profound knowledge of the Agresso installation and the RDBMS used.

Prepare the upgrade

Backup the database

Before you run the upgrade wizard, you should back up your Agresso data. To be completely safe, you should also use the [Agresso Copy](#) program to make a complete copy of all database tables.

Verify disk space

Available disk space needed for the upgrade is approximately the size of the largest table + 10%.

See the list of table changes in the Appendix.

Create data source and initialize Business Server environment

Use the [Agresso Management Console](#) (AMC) to create a new data source connected to the old database.

When the connection is up and working, you must also initialise the Business Server environment (select the **Business server** node in **AMC**) and then **Initialise Business Server**.

Now you can run the upgrade wizard. See [Upgrading Procedure](#).

amendment tables

Logging

If amendment logging is turned on for an Agresso table, an amendment table (or shadow table) are created and continuously updated with all table changes.

Note: During upgrade, all amendment logging will be turned off (by the wizard), and switched on again when the convert script is complete.

Table for amendment tables

All amendment tables are defined in the table [aagamendlog](#).

Naming standard: An Agresso table name is constructed from the structure `a<module><identification>`, while an amendment table is extended with the letters `shd` between `<module>` and `<identification>`.

If you turn on amendment logging for `acrclient`, the amendment table `acrshdclient` will be created and added to [aagamendlog](#).

Recreate active logging triggers

If some of the shadow tables are not converted correctly, and the upgrade wizard is unable to fix the error in the database check. The shadow tables can be re-created by renaming or dropping the existing table, and then re-enable amendment logging for the table in the [Activation of logging server](#) (AG30) page in the smart client.

When errors occur

It is important that all errors encountered during the upgrade process are reported back to UNIT4 Agresso.

If you, as a partner or subsidiary, have access to the SDE system, report the errors as SDE tickets.

Do not report the problem if it is related to duplicates in the database, full disk, full tablespace or similar.

Check user defined queries after upgrading

As part of the system upgrade, the ASQL parser has been improved with stronger syntax check and more consistent error handling. If a complex query contains an error, no part of the query will be executed. Instead, ASQL will report a parser error.

Previously, valid ASQL appearing before the error would have been executed, while the rest would be ignored. Now, the query will not run at all.

It is therefore important that you run all user defined queries after the upgrade.

Upgrading Procedure

The upgrade wizard

You will use the wizard located in

[.\DatabaseScript\DbTypeUpgrade\UpgradeWizards.exe](#).

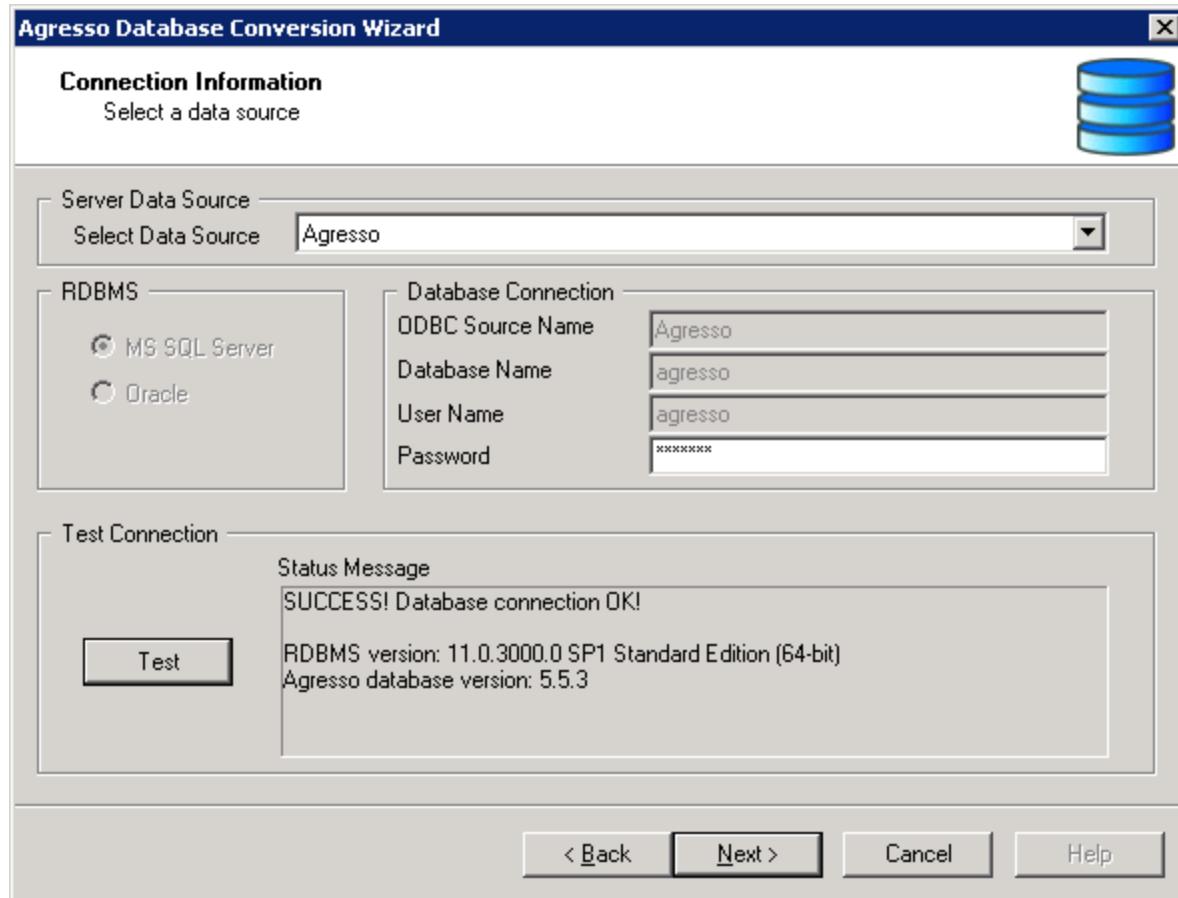
Start the Wizard

- A. Start [UpgradeWizards.exe](#)
- B. Select **Convert from 5.5**

Procedure

1. When the upgrade wizard is up and running, select Step 1, [Main Upgrade Wizard](#). Click **Next** to display the **Connection Information** dialog

 Upgrade Wizard - Connection Information



2. Select the data source, enter password, and click **Next**.

If all goes well, the wizard will display the conversion database name. Click **Next** to go to the **Window Options** step.
Note: the **Window Options** step is only relevant if you convert from 5.5.2 or later

2A: Do you convert from 5.5.2 or 5.5.3?

If YES:

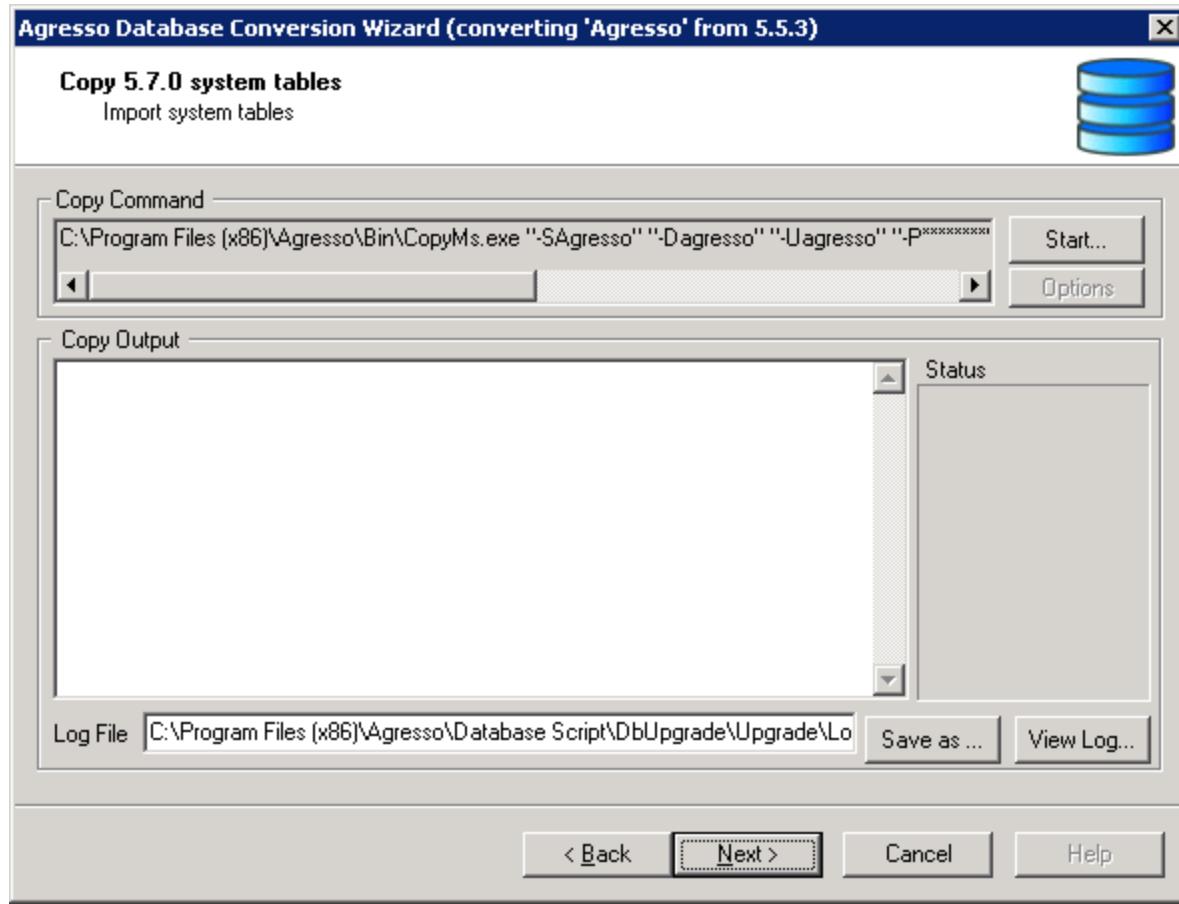
- Click **Start** in the **Window Options** dialog and wait for the wizard to complete.
- Click **Next** to continue.

If NO:

- Click **Next** to continue.

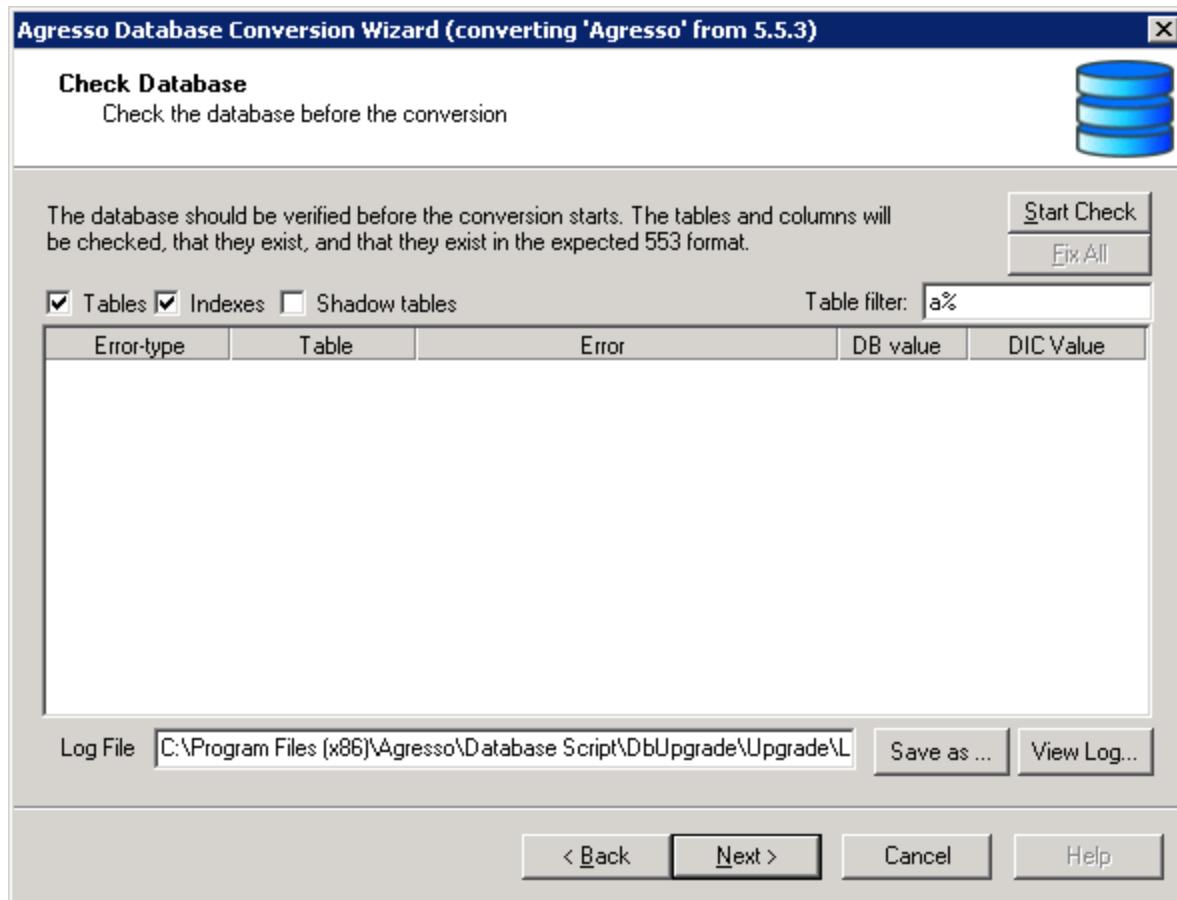
The wizard takes you to the **Copy Import system tables** step.

Upgrade Wizard - Copy in system tables



3. Click **Start** to recreate the system and directory tables. When completed, click **Next** to continue with **Check Database**.

 Upgrade Wizard - Check Database



This step verifies that the database is in the expected 5.5 format. This check might take some time.

4. Click Start Check

If you find differences, try the Fix All button. The Fix All might take some if tables with a lot of data needs to be fixed.

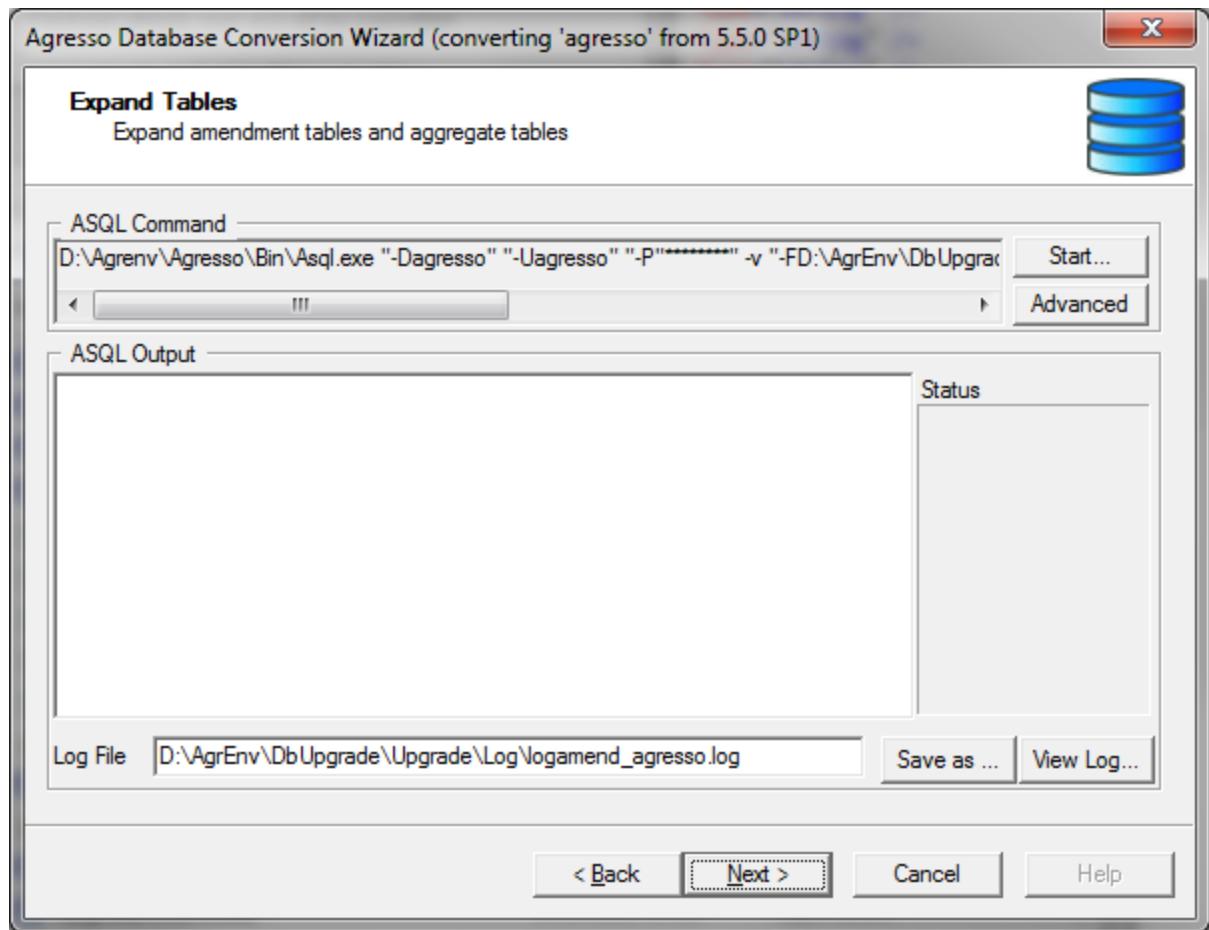
It is important to have a fixed database before continuing with the upgraded.

The following differences can be ignored:

- Columns in the table are not found in the dictionary.
- Columns are longer than expected.
- Errors on indexes (indexes will be recreated in a later step).

The next step is **Expand Tables**.

Upgrade Wizard - Expand Tables (from 5.5 SP1 only)



4. A Do you convert from 5.5.1?

If YES:

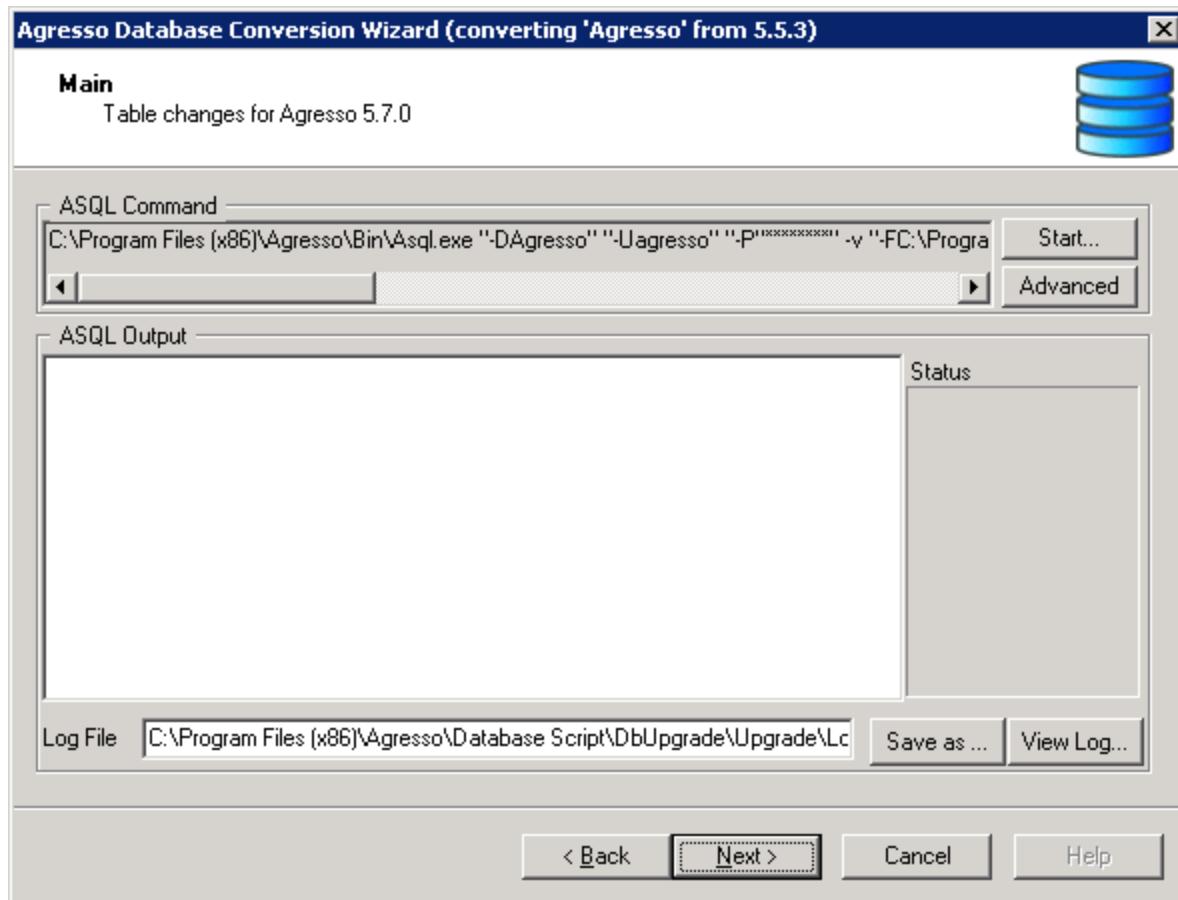
- Click **Start** in the **Expand Tables** dialog and wait for the wizard to complete.
This step expand columns in the amendment tables and the aggregate tables. It may take some time.
- Click **Next** to continue.

If NO:

- Click **Next** to continue.

You can now introduce the main changes in the database structure:

Upgrade Wizard - Main - Table changes for Agresso

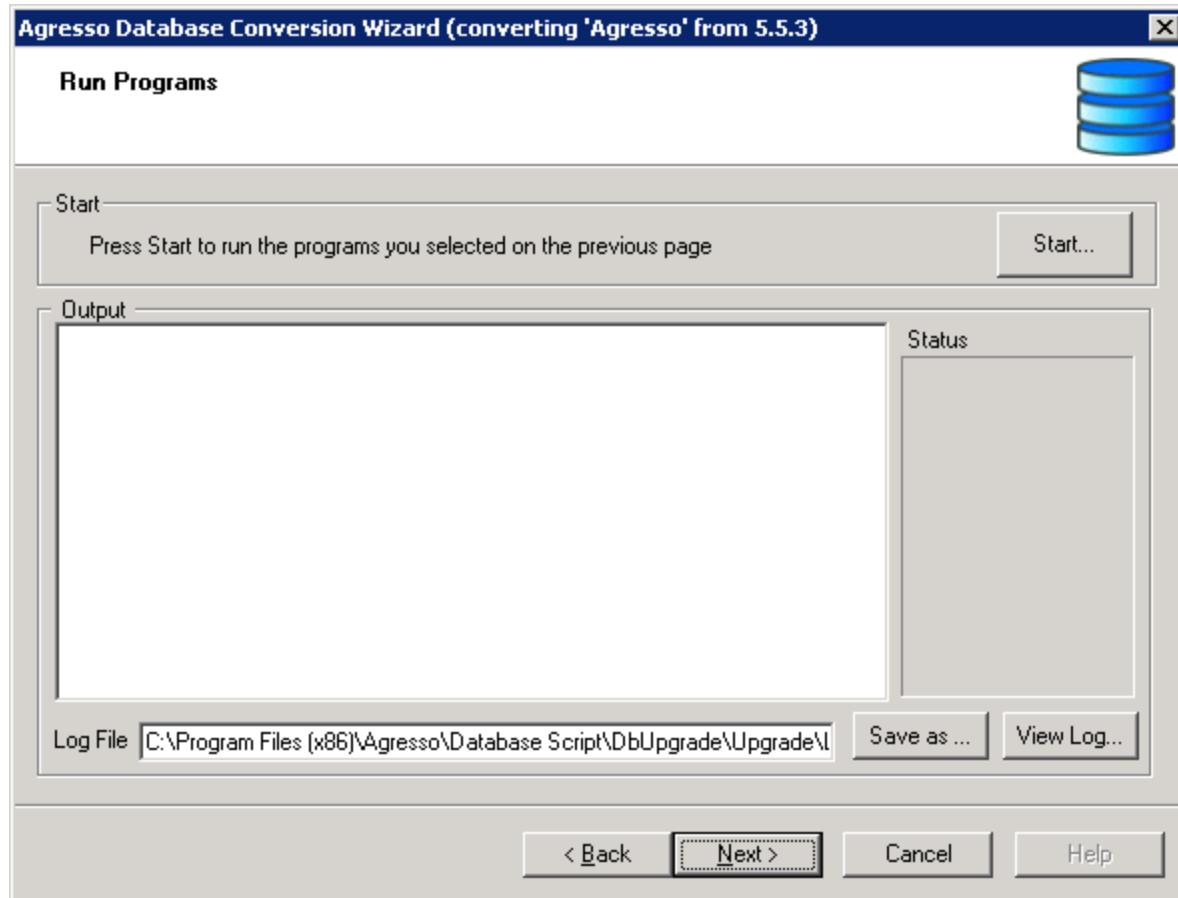


5. Click **Start** to run the script, then (when the script is finished) click **Next** to continue.

The next step is named **Select programs to run**. The available options depend on the version you are upgrading from.

As a rule, you should let the wizard make the selection for you, then click **Next**. This will take you to the **Run programs** step:

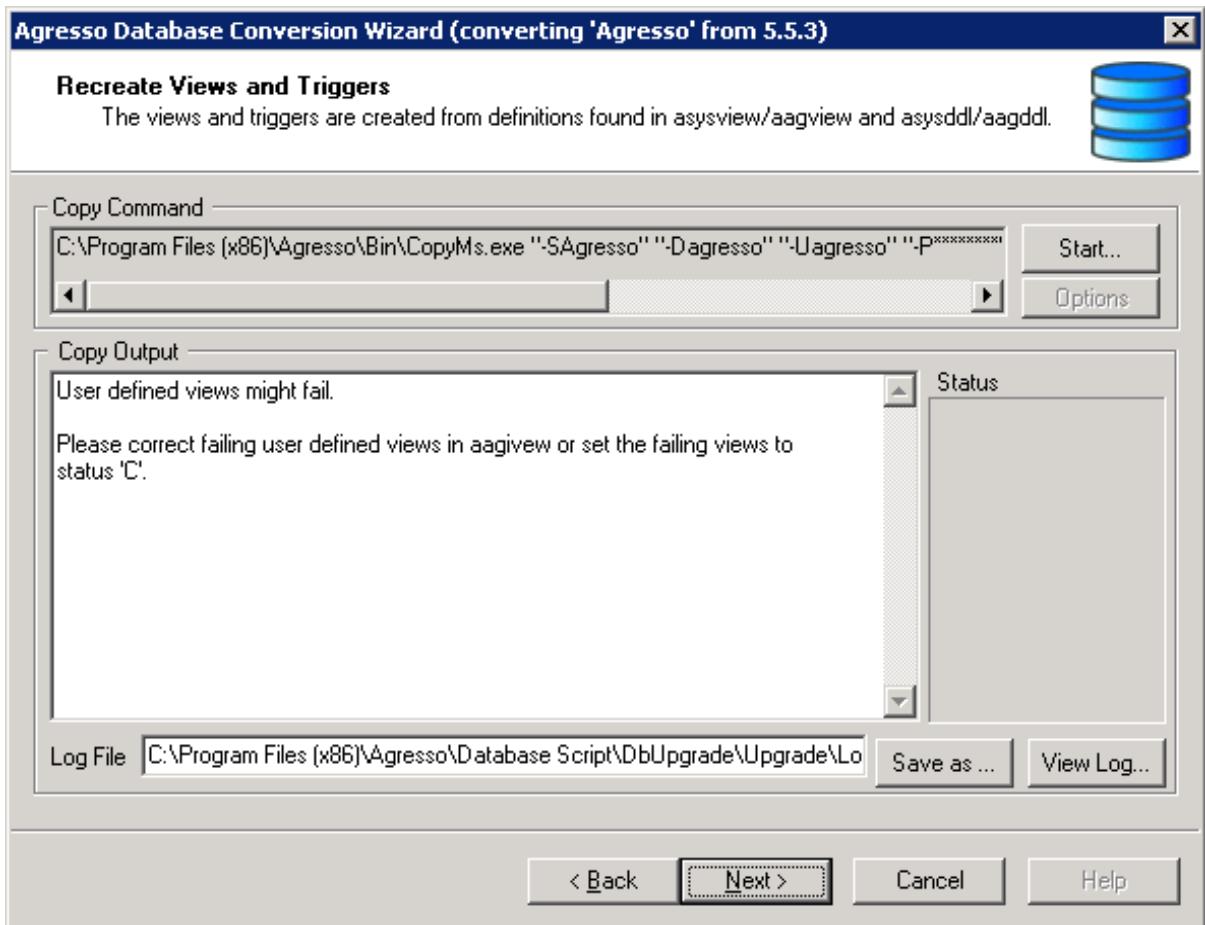
Upgrade Wizard - Run programs



6. Click **Start** to run the selected program, and then **Next** when the programs are completed.

This will take you to the **Recreate Views and Triggers** step.

Upgrade Wizard - Recreate Views and Triggers

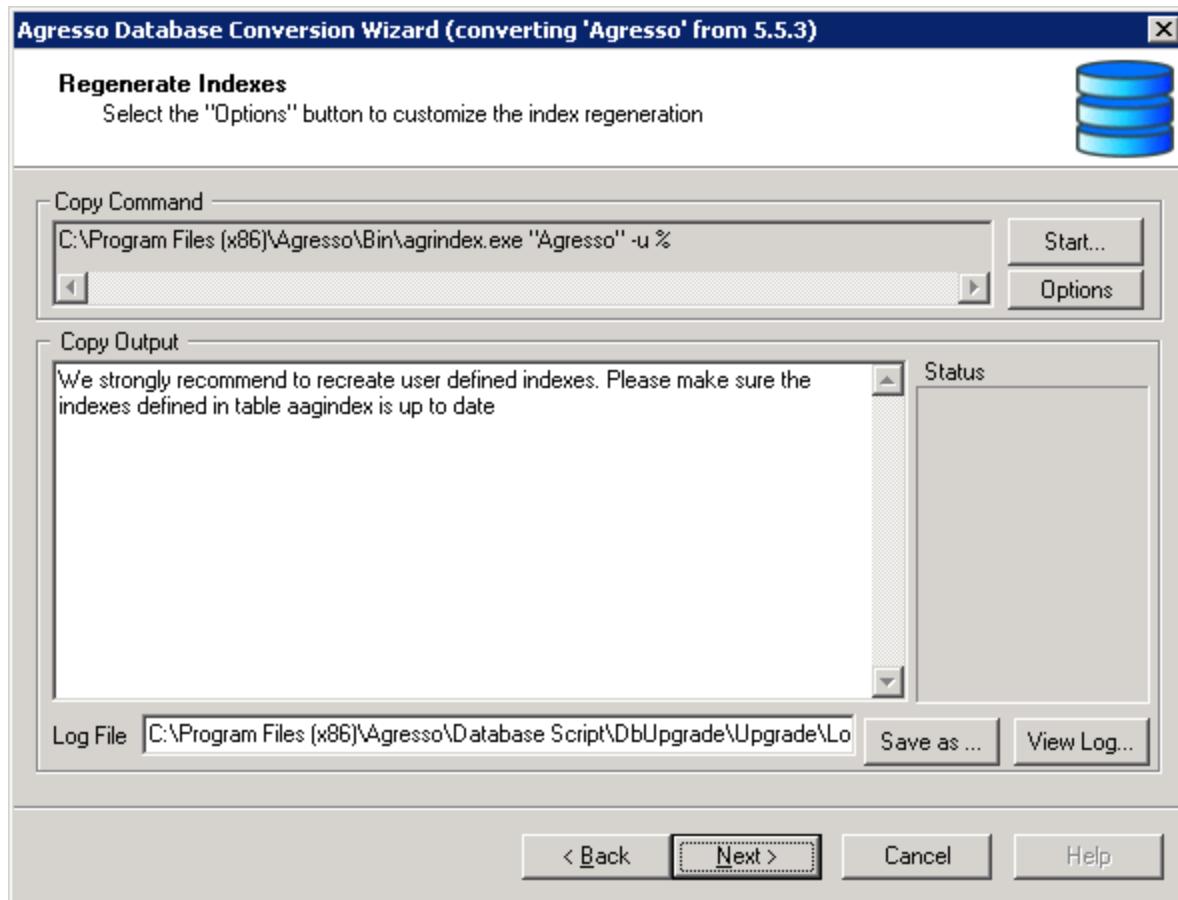


User defined views and triggers might fail if data structure has been changed in the latest release on the tables referred to. Definition of the views and triggers

can be found in the tables aagview and aagddl. See in the Appendix for table/column changes.

7. Click **Next** and **Start** in the next windows, in order to restore system and user defined views and triggers. This will bring you to the final step.

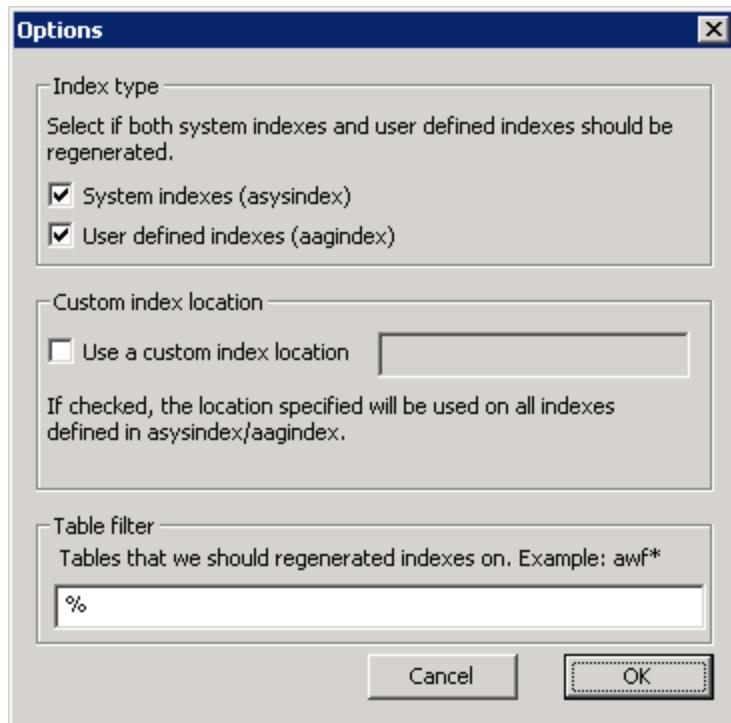
Upgrade Wizard - Regenerate Indexes



8. Click **Start** to regenerate the indexes. The indexes are important for performance and for preventing duplicates in the database.

Duplicates: If an index is created without the unique flag due to duplicate rows, remove the duplicates, and rerun **Recreate indexes**.

9. Click the **Options** button to open the Options dialog:

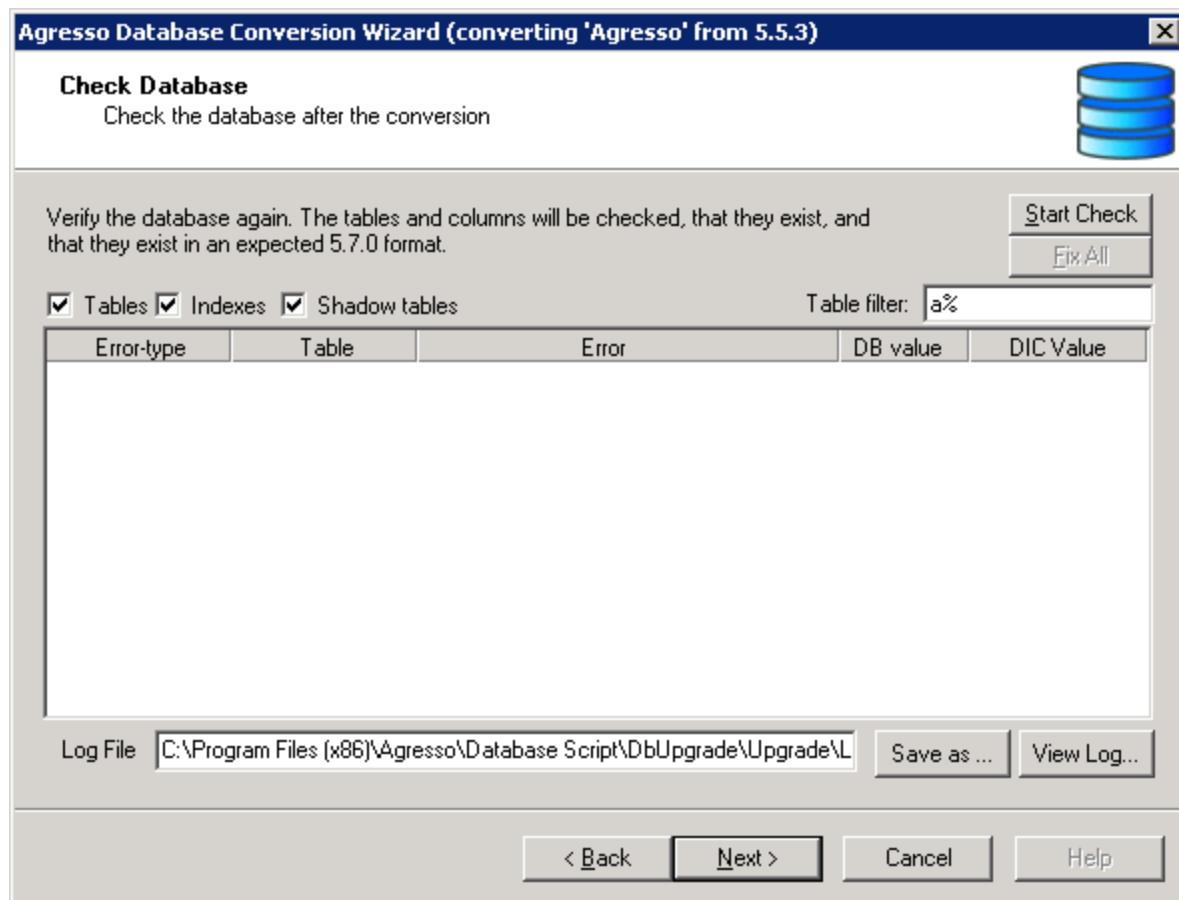


A note about indexes

User defined indexes (stored in [aagindex](#)) are recreated by default. Due to table changes, you should also change these, to avoid reduced performance. For details. see [AgrIndex](#).

- 10.** Select your options and click **OK**. Back in the **Regerate indexes** dialog, click **Next**. You should now check the database after con-
version.

Upgrade Wizard - Check Database



11. Click **Start Check**.

The database upgrade is now finished!

Finalise Upgrade

Overview

To complete the upgrade, you should:

- Check the converted Browser templates, and correct any errors.
- Clean up duplicates and add indexes.

- Correct all user defined views due to the table changes (see Appendix, Agresso Data Dictionary)
- Remove all tables no longer in use.

Check and correct Browser templates

Tools are provided to administer and ease the upgrade process for Browser templates:

- A Browser checker utility, BrowserTemplateChecker.exe, introduced in 5.5 Service Pack 1 that is run after upgrade.
- Check the log file produced during the upgrade. In this log, detailed information can be found of the Browser templates which need to be adjusted to run correctly on the latest version of Agresso. To fix the problem, open the browser template, make the necessary changes and save.

Indexes and duplicates

When running the database check step in the wizard, there might were indexes missing due to duplicates.

Indexes might be very important, and missing or wrong indexes can lead to very poor and slow performance.

Check the log files if there has been any errors while creating the indexes.

See log file .\Database Script\DbUpgrade\Upgrade\Log\logindex_<database>.log

Use any database tool to find the duplicates (for example SQL Server Management Studio (SqlServer), SQL Developer (Oracle)).

Example on how to find the duplicates:

```
select distinct client, attribute_id from agldimension
group by client, attribute_id
having count(*) > 1
order by client, attribute_id
/
```

Delete/change so there are no duplicates in the table, and re-create the index.

Look up in the Appendix, Agresso Data Dictionary to see the correct index definition. Create the index using the preferred DBA Tool.

User defined views

The table structure is changed from release to release. Ensure all user defined objects are changed according to the new structure (See Appendix Agresso Data Dictionary)

Change the definition and recreate the views

See log file .\Database Script\DbUpgrade\Upgrade\Log\logviews_<database>.log

Amendment logging

Amendment logging triggers needs to be recreated after the upgrade. Use the Smart client to open the **AG30 Activation of logging server** from **System Administration | Data Control | Amendment logging**, and regenerate active logging triggers.

Remove tables not longer in use

The script *drop.asp* in the *Scripts* directory will drop all old and temporary tables no more used by the application.

Note: This script must be run when the upgrade has been completely verified, and you are sure that none of the old tables are needed for backup purposes.

Create System Attributes

Description

The wizard **Create System Attributes** will generate all the fixed attributes currently missing in your UNIT4 Agresso installation.

The wizard is part of Step 6. **Run binary programs** in the main system upgrade wizard, but is also available in a stand alone version, available in the file *CreateAttributesWizard.exe*.

Standard (fixed) attributes only

The wizard updates only attributes delivered as part of the standard **UNIT4 Agresso** installation (with attribute id from A0 to LZ).

Other attribute types, for example localisation specific attributes or custom attributes, will not be updated by the wizard.

All clients will be updated

Attributes are client dependent and the wizard will generate one attribute instance per client. Attribute names and descriptions will comply with the selected language (Company Information screen, CR01) for the various clients.

Run the wizard

The wizard is very straightforward, with no complex options. Just follow the instructions on screen.

Introduction to Upgrading

Important changes

Changes in database

The Agresso upgrade from version 5.4 to 5.7.0 affects most of the Agresso tables. Many tables have new columns, and, to support more intuitive field values, several (old) column definitions have been modified. For example:

- Transaction series are expanded to 12 digits. Bigint(MSSql) and number(20) (Oracle).
- Account numbers are expanded to 25 characters.
- All attribute values can now hold up to 25 characters.
- Descriptions, names, file_names etc. are expanded to 255 characters.
- All supplier_id's and customer_id's (apar_id's) are converted to alphanumeric, 12 characters.
- Where client exists in the index, we have moved the client column to be listed first in the index list in order to improve the performance on index access.

Note: Although a large amount of columns have been redefined to allow extended values, this will not necessarily (or: immediately) affect the database size. The columns have been redefined from datatype **char** to **varchar**, allowing a more dynamic utilisation of the disk space.

Functional changes

Some functional areas have been completely rewritten, as a means to keep up with both functional and technological requirements. Affected areas are, for example:

- Document archive - which now can be utilised from all over Agresso.
- Authorisation and access rights - where the introduction of a new Role concept has required a new data model and implementation.
- Workflow - a new Agresso workflow engine has replaced the previous implementations (Compello).

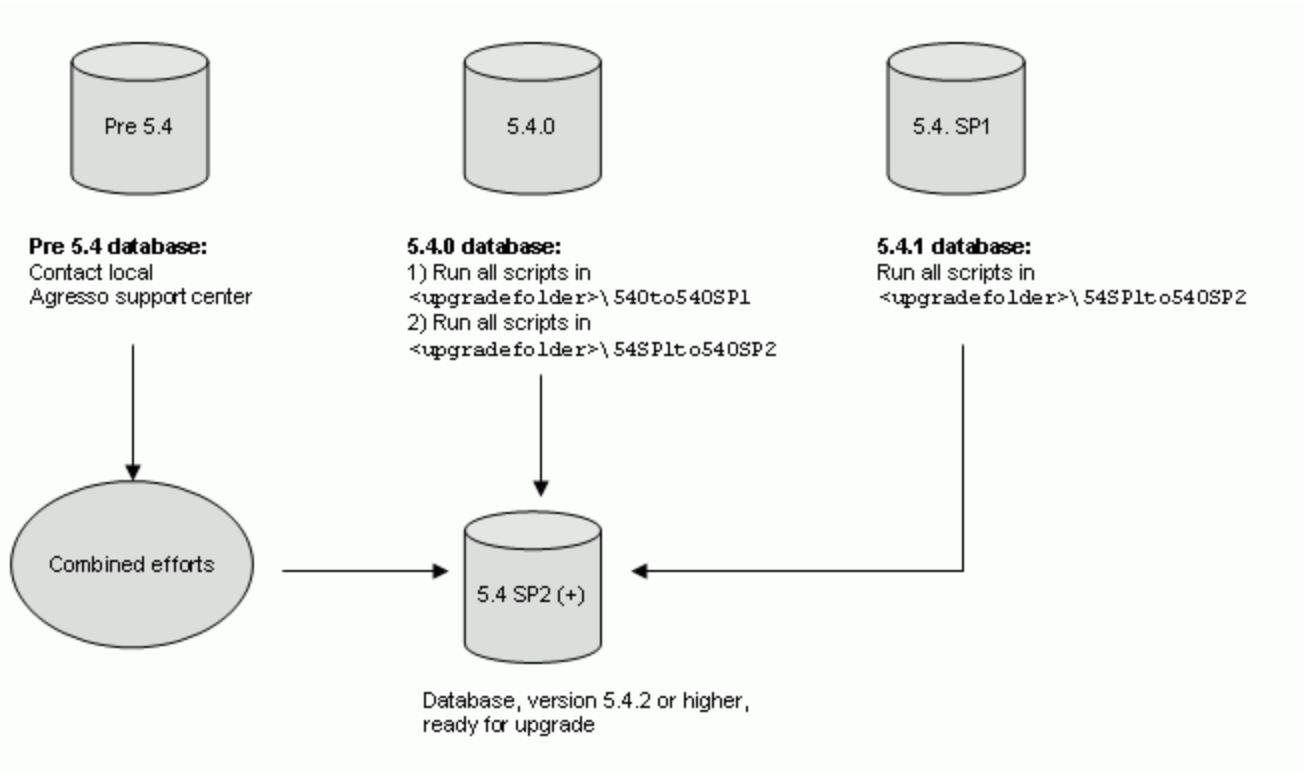
Database updates

Please note:

- If your database is older than Agresso version 5.4 SP2, you must contact your local Agresso support center. In that case, you need assistance to make an additional, basic upgrade.
- The new version of Agresso must have been successfully installed before you start the upgrade process.
- We assume that you have profound knowledge of the Agresso installation and the RDBMS used.
- **Upgrade scripts and wizards:** The upgrade scripts and wizards are found in the folder [.\DatabaseScript\DbUpgrade](#) on the upgrade DVD. It will be referred to as [`<upgrade folder>`](#).

Diagram

The required database updates are shown in the following diagram:



Find correct database version

If you need to find the correct database version, execute the following sql statement:

```
select text1,text2,text3 from syssetup where name = 'BASE_VERSION'
```

The main upgrade stages

There are four main upgrade stages:

1. Installation of the new UNIT4 Agresso software, database upgrade and data conversion. We refer to this stage as the Main system upgrade.
2. Additional system areas upgrades - dependent on the areas included in your license.
3. Functional areas (module) upgrades - dependent on modules used by your installation.
4. Completion.

Test versus Production

In the upgrade process, we make a distinction between upgrade of a *Test system* and upgrade of the *Production system*. For both systems, however, you must go through all the 4 stages mentioned above.

Test

The test upgrade comes first, and results in a complete, fully functional Agresso system. You will run through the complete upgrade process, and perform all tasks: run the upgrade wizards, and use the **Agresso Management Console** and the **Agresso Smart Client** to set

up and configure elements not handled by the wizards. When the Test system is found in order, you are ready for the Production update.

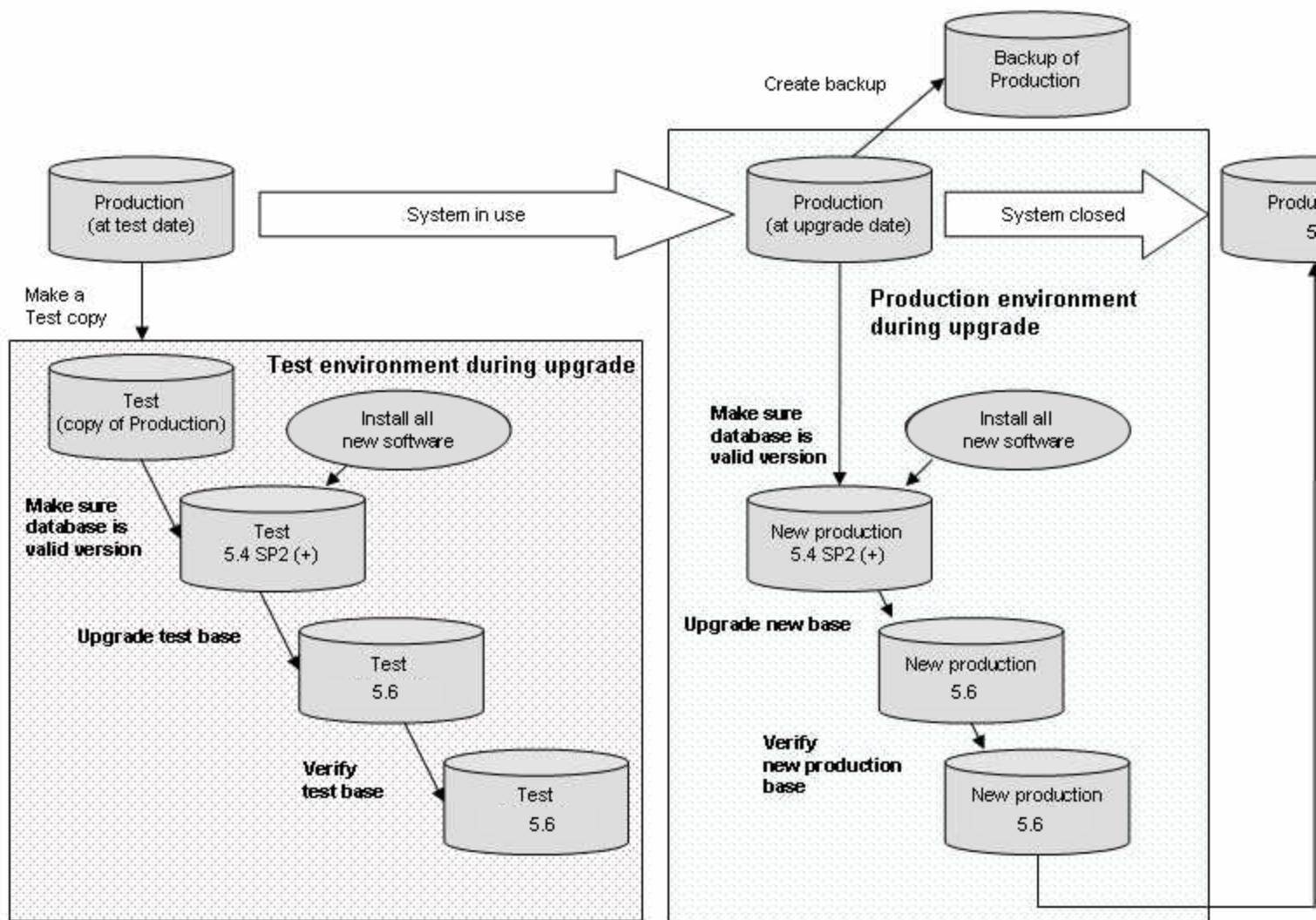
Production

The Production upgrade is basically a repetition of the Test upgrade process, with one important difference: Most of the manual setup for the *additional system areas* in the Test environment, can be copied directly into the Production database. The wizard **Copy Setup Between Databases** will facilitate the production upgrade process.

Note: If you do not complete a Test upgrade first, but works directly with the Production system, you will have to follow the detailed upgrade process as described under *Test System Upgrade (all tasks)*. With one system only, you will not have any previous settings to copy.

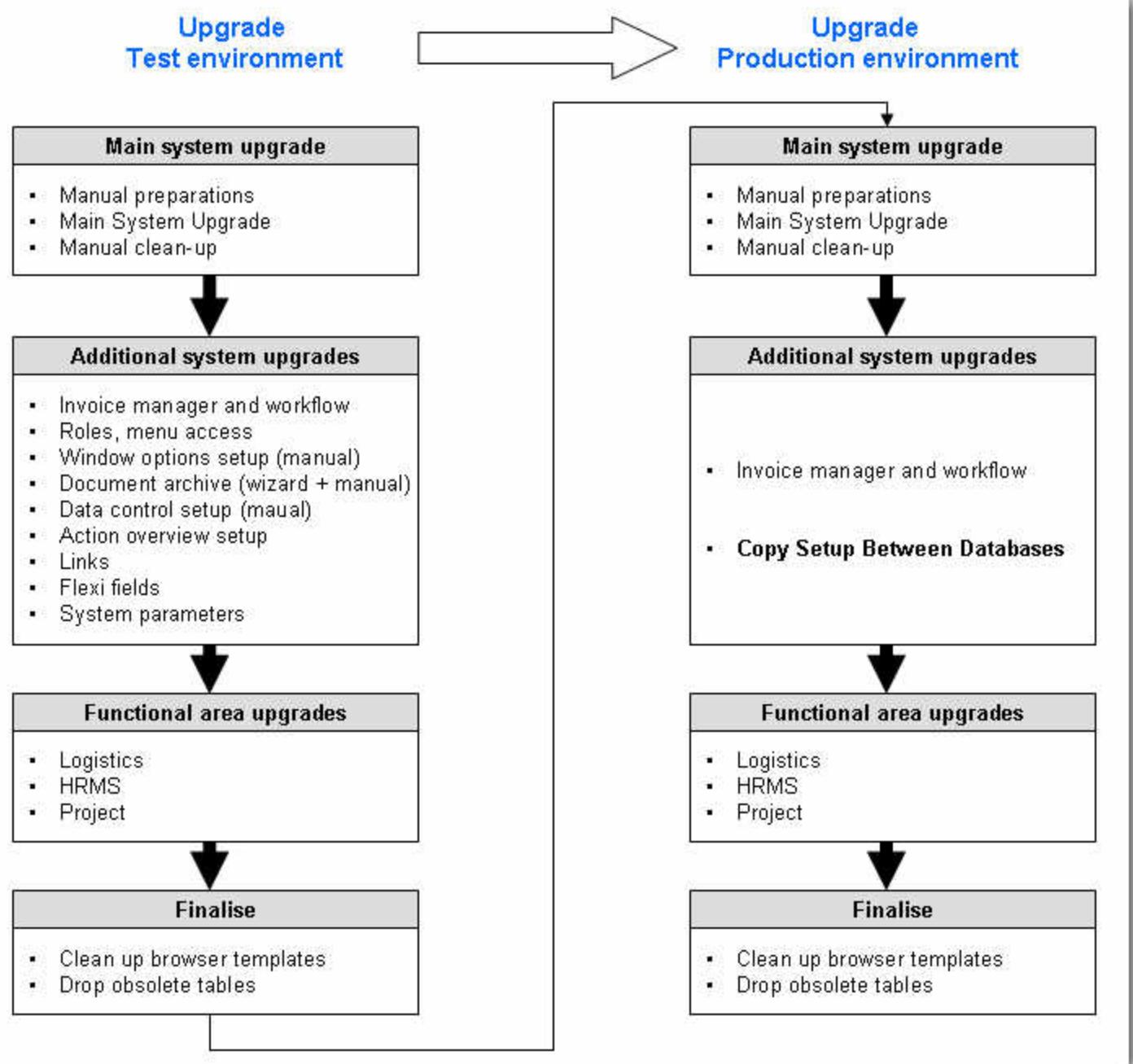
Upgrade overview diagram

The diagram below outline the complete upgrade process. The time aspect is indicated by showing the activities from left to right. There should be considerably less time involved in the final upgrade of the production database, compared to the test upgrade.



Test and production tasks

When we reduce the scope to the upgrade of a 5.4 SP2(+) system, the main tasks are as follows:



Wizards

All upgrade processes are wizard based. For some areas, however, it will also be necessary to perform additional, manual actions.

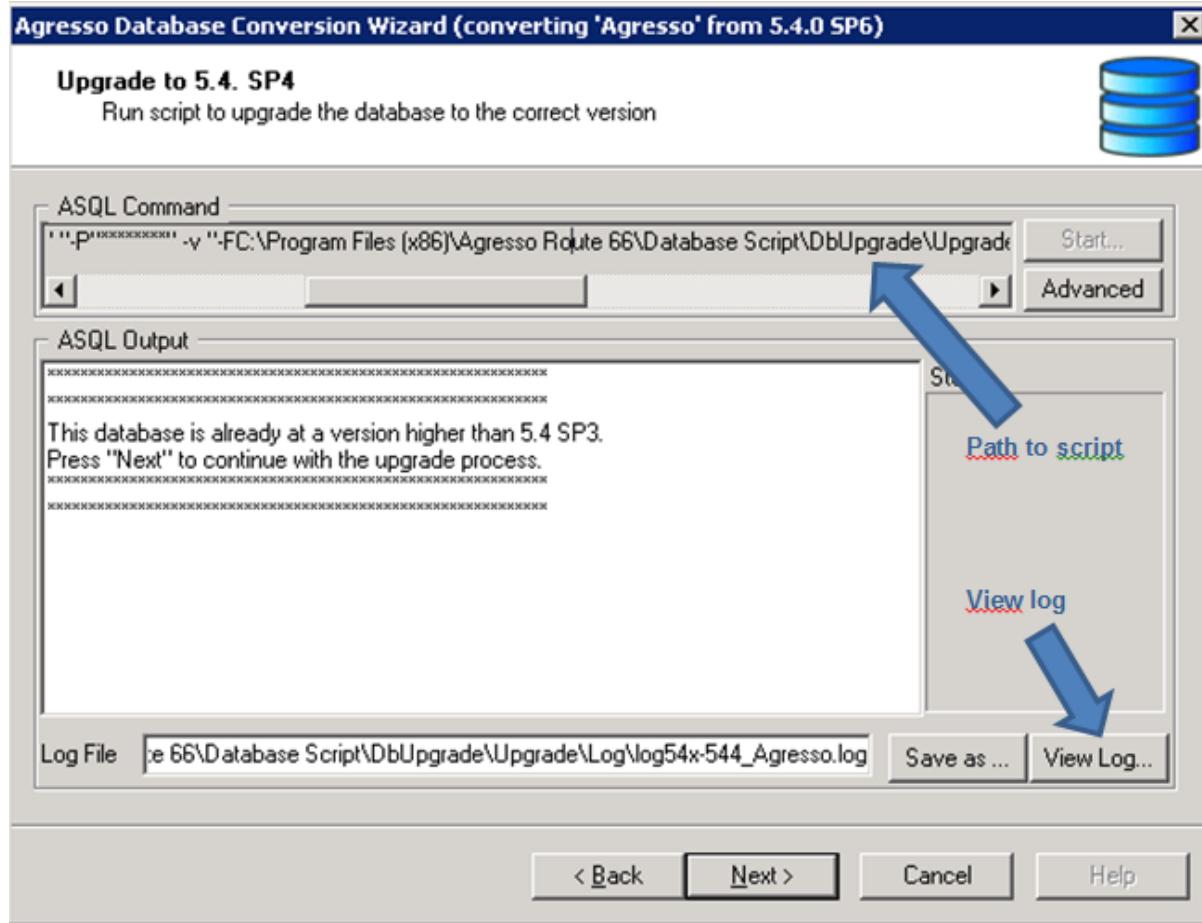
Location

The upgrade wizards are found in the UNIT4 Agresso installation directory, at the following location:

<INSTALL LOCATION>\DatabaseScript\DbUpgrade\Upgrade

Script details

The upgrade wizards run one or more scripts. In the wizard dialog, you will find the path to the current script. After the script has run, you can study the log:



The upgrade user

Temporary user for all clients

During the main upgrade (when upgrading from 5.4x only), a temporary Agresso user, `upgr55`, is created, enabling you to log on to Agresso, and set up roles and access rights for the existing users. This user is created for every client in the database, the password is `upgr55`.

When errors occur

It is important that all errors encountered during the upgrade process are reported back to Unit4 Agresso

If you, as a partner or subsidiary, have access to the SDE system, report the errors as SDE tickets.

Do not report the problem if it is related to duplicates in the database, full disk, full tablespace or similar.

Upgrade overview - Test

Process description

The upgrade process presented here, starts when you have a valid 5.4 SP2(+) system in your test environment. For database upgrades, see [Upgrade Process Overview](#).

The Test upgrade goes through four stages:

1. The Main system upgrade wizard performs all the required, basic upgrades of your system: It will add new tables, and convert old tables and data to the new formats - as far as it goes.
2. If relevant, you will run the Workflow wizard. This will convert data from Invoice Manager to the new Workflow solution, and also create new roles according to the new role structure.

Next, and this task is always required in stage 2, you must establish the new authorisation structure, giving the users access rights in accordance with the new Role concept. During upgrade og access rights and roles, you may also want to modify the roles automatically created by the workflow wizard.

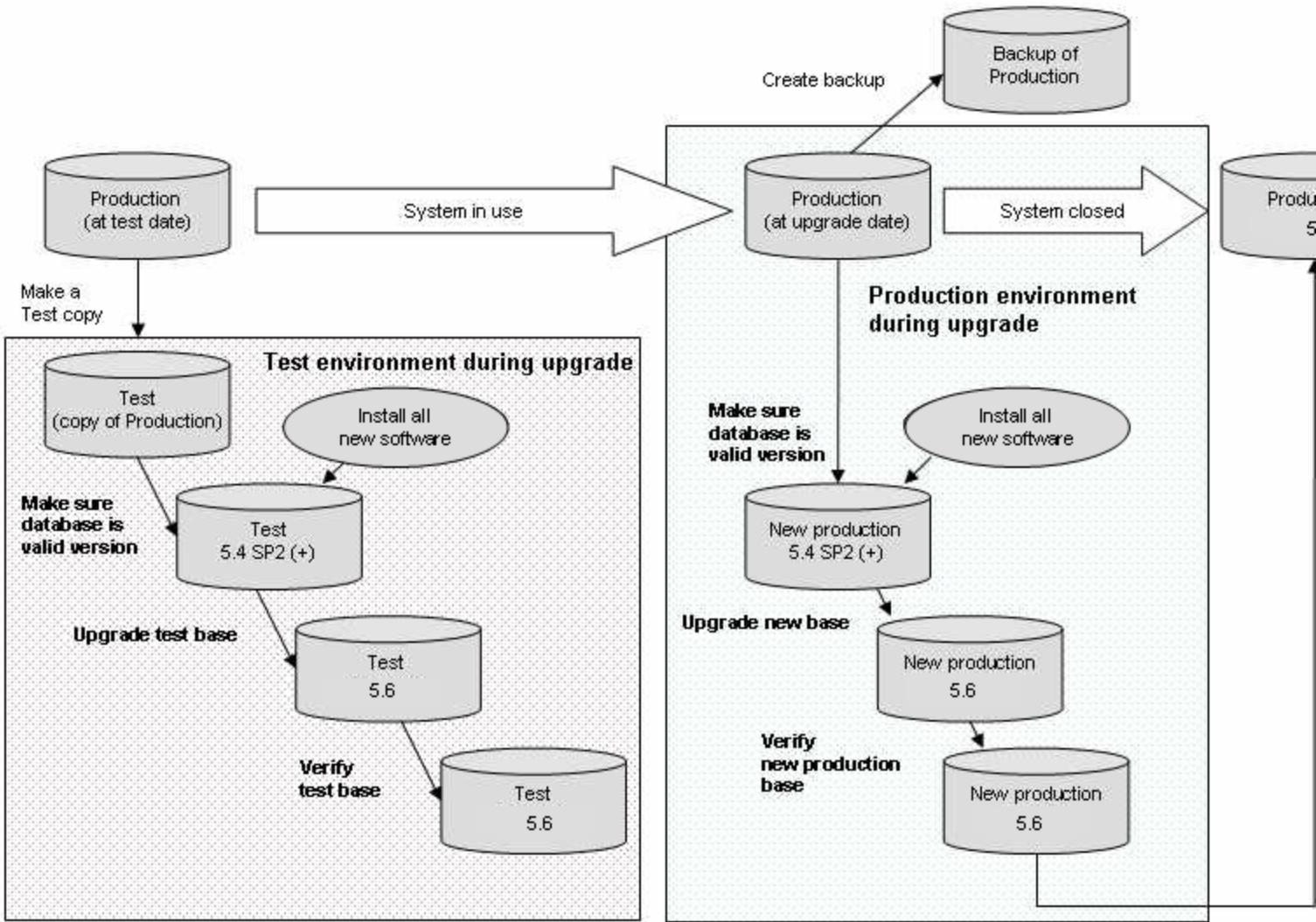
Note: You can either create the new role structure from scratch, or you can use a wizard to convert the old groups, web roles in one go. See [Authorization and Roles](#) for details.

Further - you will need to perform upgrade tasks for the remaining system areas used by your installation, such as Document archive, Data control and Invoice follow up (the latter is handled by Action overview in Agresso 5.7.0).

3. You must run upgrade wizards and perform manual setup for a few functional areas (or modules), where new implementations in the latest version of Agresso prevents automatic upgrading. Modules not in use can be ignored.
4. Finally, you finish the upgrade by cleaning up browser templates and remove tables not longer in use.

Overview diagram

The diagram below summarises the main tasks required for a complete upgrade - intended for the Test environment:



Manual setup tasks

There are several places where we have explicitly stated that manual setup is required - in addition to - or as an alternative to - running a wizard.

Some of these tasks may take time, and can in practice go on for several months, and long after the subsequent tasks, as outlined in the diagram above, has been completed.

Please note: The manual setup tasks is not described in these Technical Guidelines (you will see the details when you enter the detailed descriptions). The required documentation is found in the relevant Release Notes for the area in question.

Important note

If you plan to use the **Copy Setup Between Databases** wizard to transfer users, roles, menu access and optional setup from Test to Production, be aware of the following:

The wizard will copy all users, passwords, roles etcetera from the test (source) database to the production database. The production data will be replaced with data from the source database, with two important consequences:

- New users in the production system (after the test upgrade started), will not be transferred - unless you make sure that they also are registered in the test system.
- Passwords changed in the production system after the test upgrade started, will be overwritten by the old passwords.

Prepare the Upgrade

Install UNIT4 Agresso software

When your Test database is in the correct 5.4 version, you should install all UNIT4 Agresso software.

Verify disk space

Available disk space needed for the upgrade is approximately the size of the largest table + 10%

Note: Expanding the maximum number of characters allowed in a field does not have any immediate impact on the database size. The expanded columns were previously of data type char, while they now are redefined to varchar.

Create data source and initialise Business Server environment

Use the **Agresso Management Console** to create a new data source connected to the old database.

When the connection is up and working, you must also initialise the Business Server environment (select the **Business Server** node in AMC and then **Initialise Business Server**). You do not need a database connection to initialize the Business Server Environment in AMC.

Rename or delete amendment tables

Logging

If amendment logging is turned on for an Agresso table, an amendment table (or shadow table) are created and continuously updated with all table changes.

Note: During upgrade, all amendment logging will be turned off (by the wizard), and switched on again when the convert script is complete.

Table for amendment tables

All amendment tables are defined in the table [aagamendlog](#).

Naming standard: An Agresso table name is constructed from the structure `a<module><identification>` (example: `a cr client = acrc-client`) while an amendment table is extended with the letters `shd` between `<module>` and `<identification>`.

If you turn on amendment logging for `acrclient`, the amendment table `acrhdcclient` will be created and added to `aagamendlog`.

Old amendment tables

During previous releases, the amendment tables have not been upgraded. When upgrading to 5.7.0, these tables will be checked, which may lead to a large number of errors.

Recommendation

To avoid problems with existing amendment tables, we recommend that you rename (or delete) all amendment tables before you start the upgrade process. Thereby, the upgrade wizard will not find them, and they will consequently not generate any errors.

If you need the amendment tables for historical reasons, they will always be available in your database copy.

To remember

Custom object definitions

The main system upgrade wizard will initially check the database for any custom objects, and write a detailed report containing all the old definitions (scripts). During the upgrade process, all custom objects will be removed.

The generated report, named [*user_def_obj.txt*](#), will be saved to the default log directory. You will need this report, as well as a detailed knowledge of the new table definitions (see Table changes in latest version) to restore previous functionality in the latest version of the UNIT4 Agresso database.

The *custom objects* that will be removed, are:

- triggers and indexes,
- procedures,
- shadow tables,
- user defined views.

Run the Upgrade Wizard

The upgrade wizard

You find the upgrade wizard at the following location:

[.\Agresso 5.7.0\DatabaseScript\DbUpgrade\UpgradeWizards.exe](#)

Upgrade scripts and log files

Logging and error handling

The **Log File** will, when the script has run, contain a description of what happened. Each statement are listed, along with status after execution.

Note: You should - as a general rule - always look at the log file when the script is finished. If any errors occurred, they must be corrected, and the script must be run again - *before* you continue with the next upgrade step.

When we describe the actions for each step below, we *do not* ask you to view the log files. We take it for granted that you will do so!

Logging details

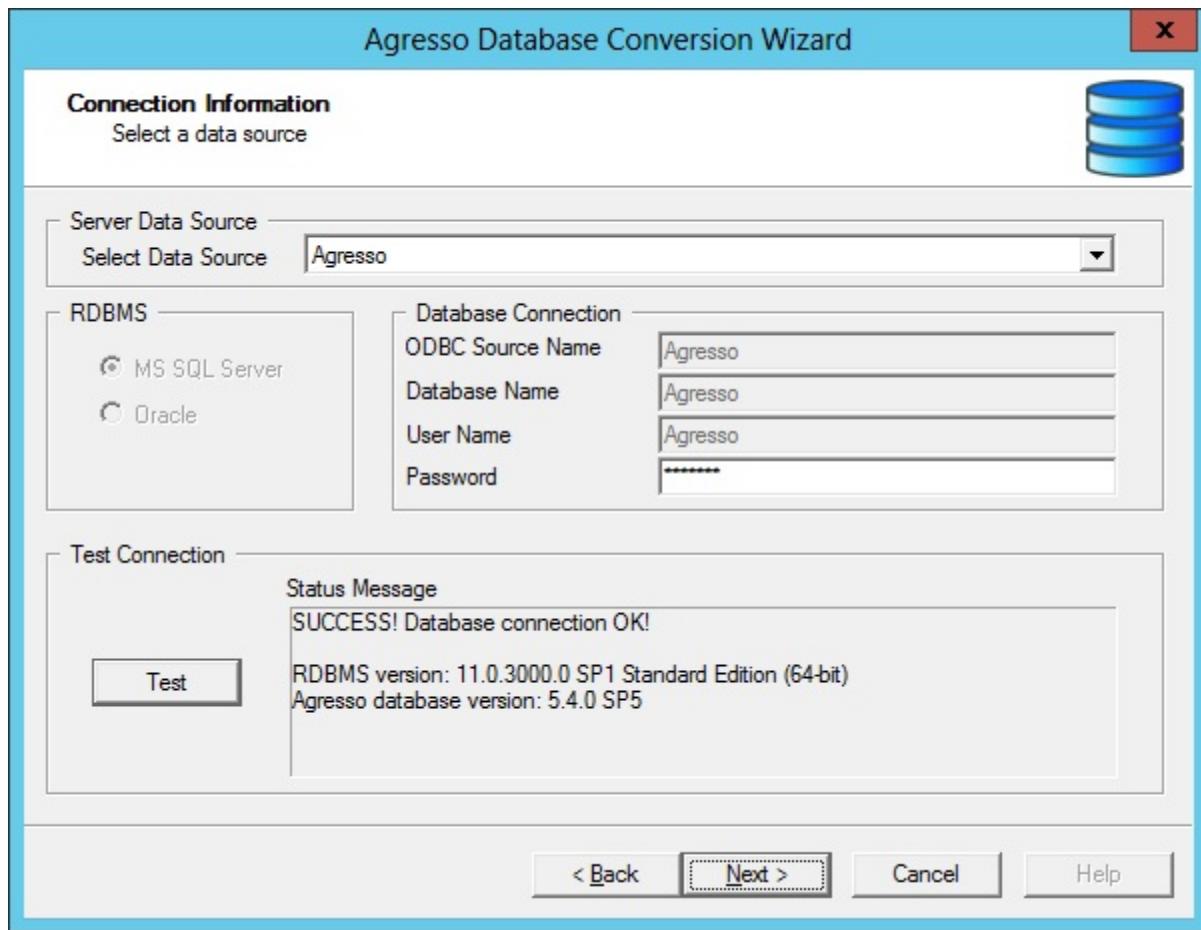
You can define the level of details to be listed in the log file by using the **Advanced** button available in the various wizard dialogs.

Run the upgrade wizard

1. When the upgrade wizard is up and running, select **Convert from 5.4**. Then select Step 1 [**Main Upgrade Wizard**](#). The **Connection Information** dialog will be displayed.



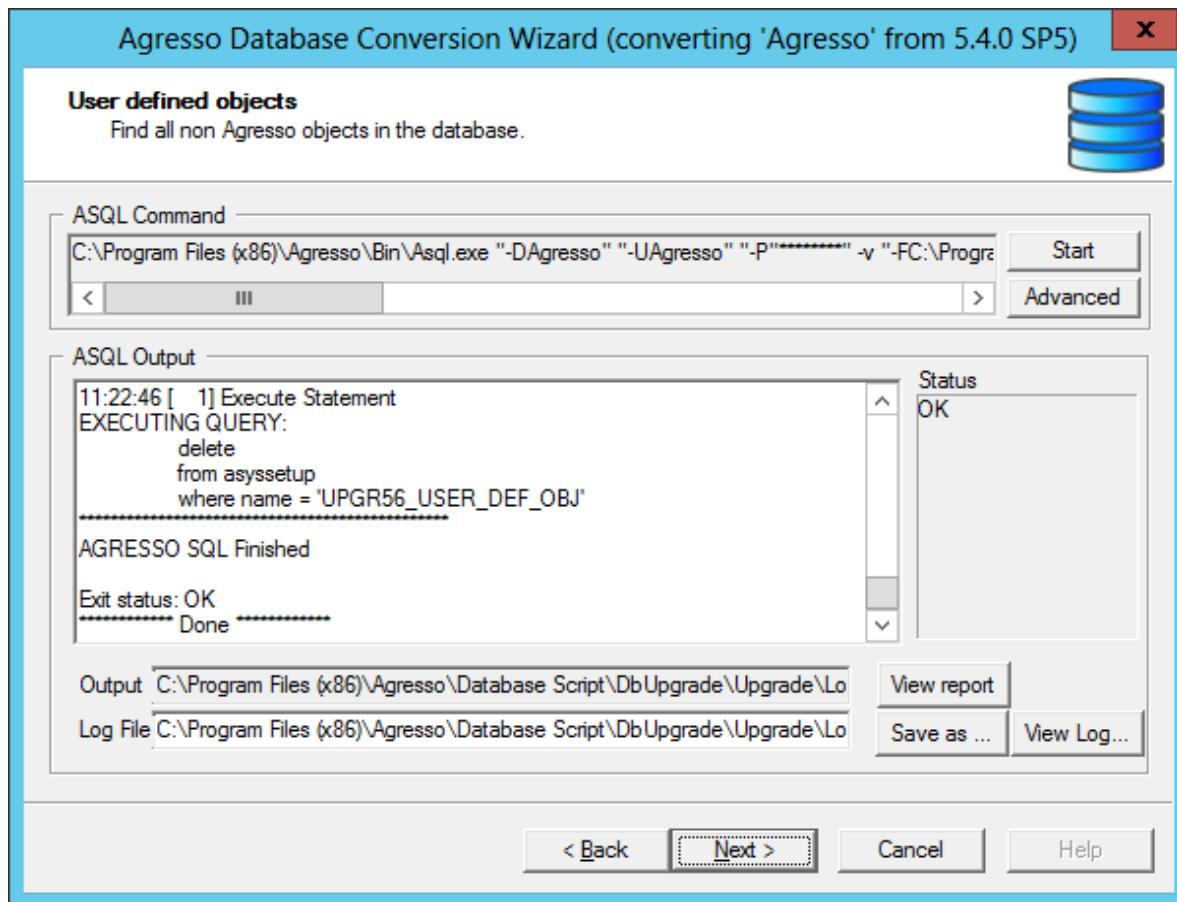
Upgrade Wizard - Connection Information



2. Select the data source, enter password, and click **Next**.

Note: If you get an error message, just accept the proposed action - and make a note of it! The error is not significant for the upgrade, but it will be when running Agresso later.

Upgrade Wizard - User defined objects



More step 3 details

Note: The report containing the custom object definitions are shown in the **ASQL Output** field. You will need this later.

Important: Do not continue if any errors are found in the database. These may be:

- **Null values** - these must be fixed before you continue. Use copyms/copyora to copy the table with the NULL values out, and then into the database again.
- **Duplicates**. Do not continue with the upgrade if there are duplicates in any of the tables
 - [aaguser](#),
 - [acruserinfo](#) or
 - [aaguserfunc](#).

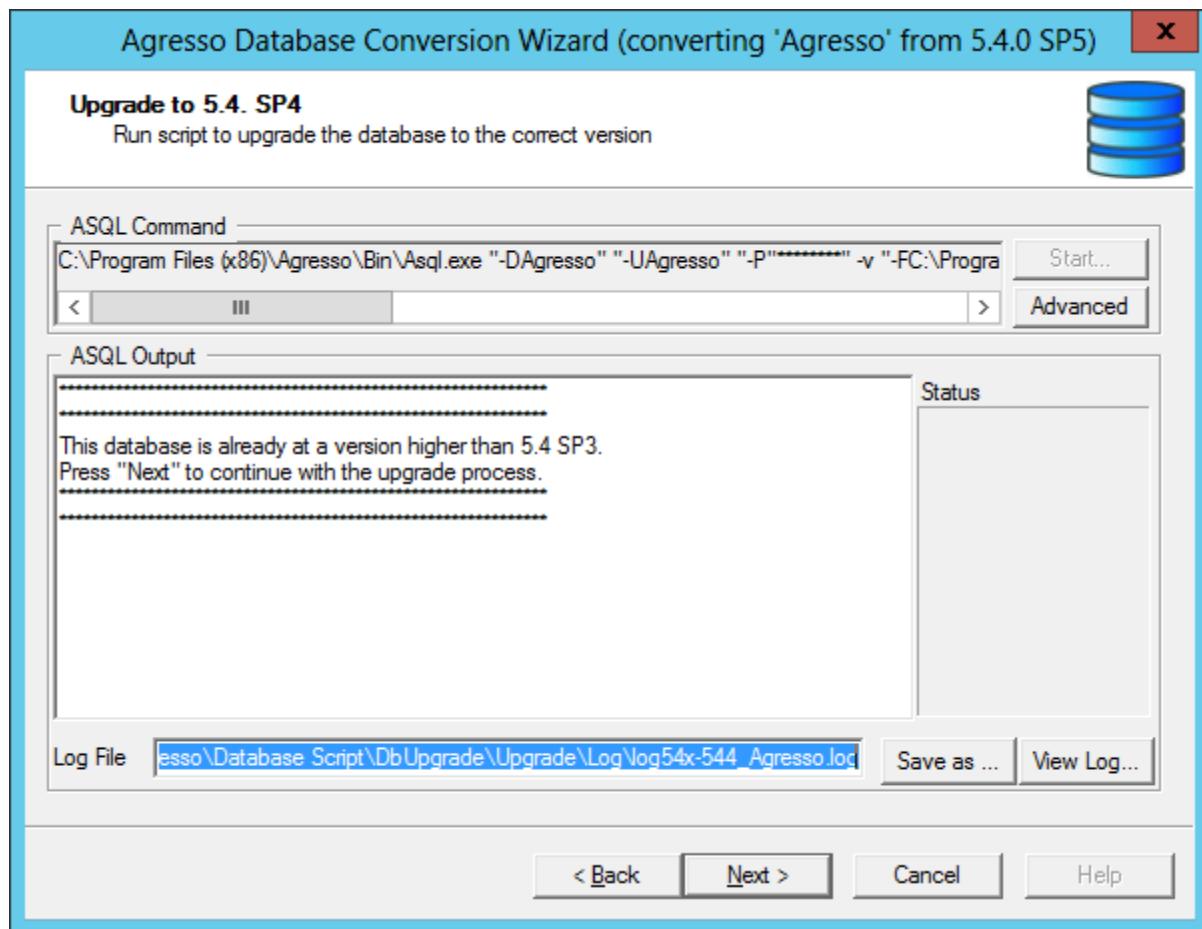
You must find the duplicates and remove them (or rename them).

Note: You must re-run the User defined objects step until there are no errors.

3. Click **Start** to run the script.

Important: You must correct all reported errors and make sure that everything is correct (re-run the script - until no errors are reported), before you can proceed to the next step.

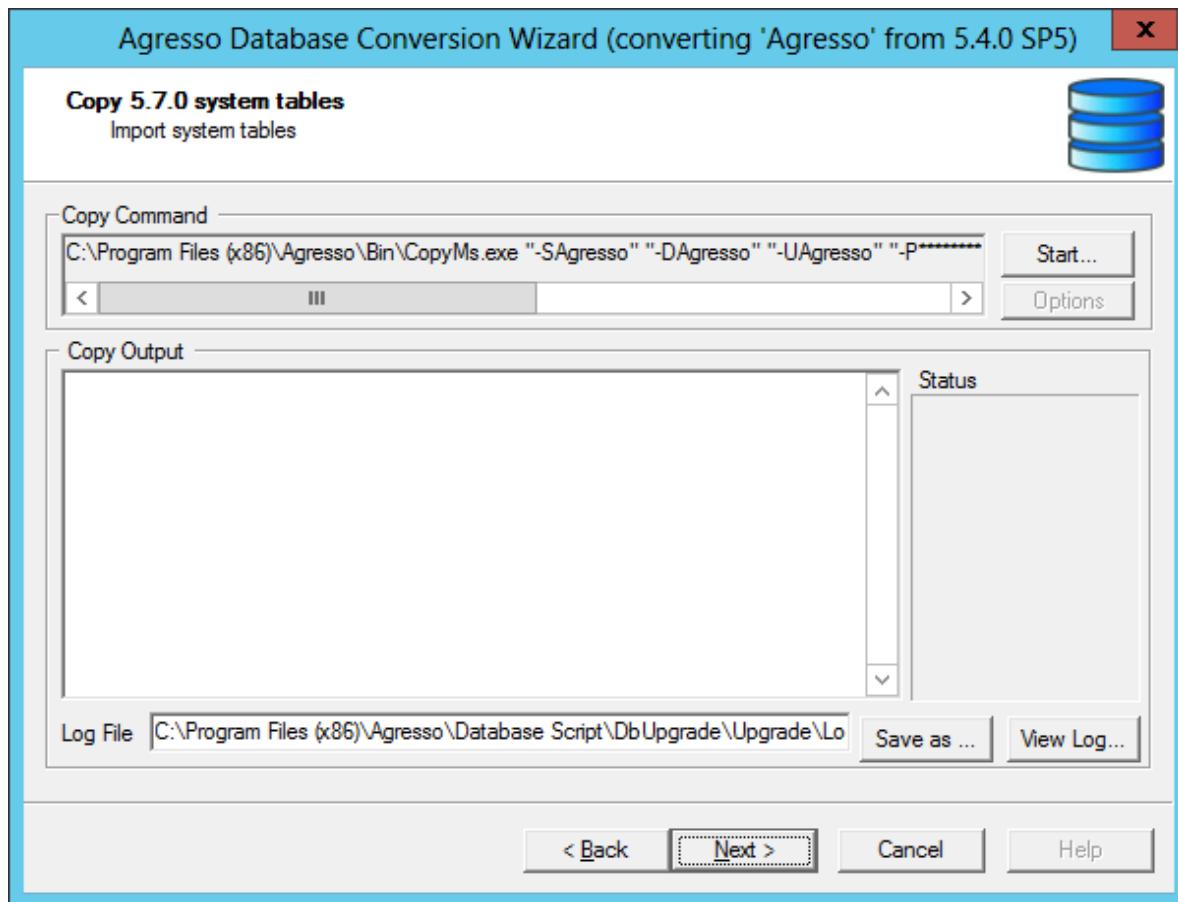
 Upgrade Wizard - Upgrade to 5.4 SP4



4. If your database needs upgrading, click **Start**. Click **Next >** to continue.

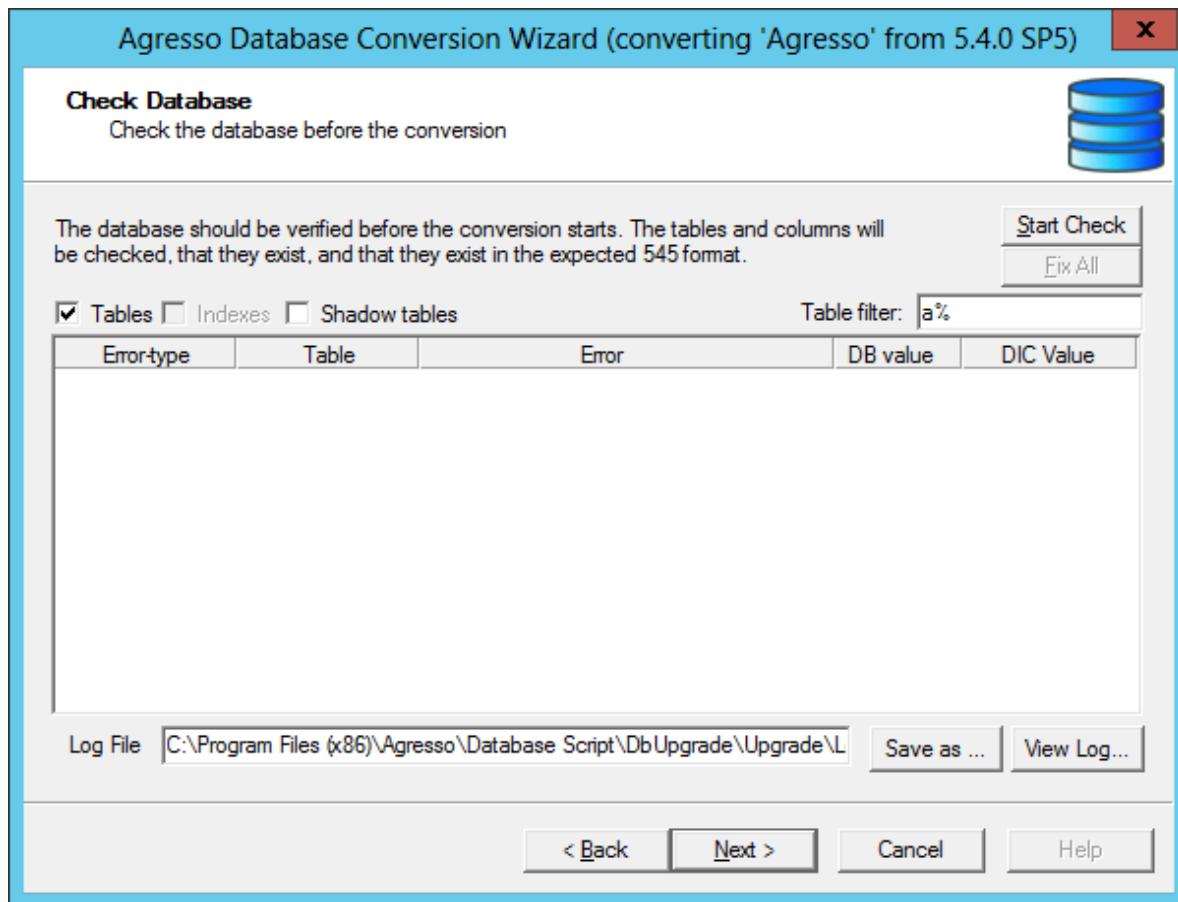
You are ready to restore the Agresso system tables, some awf* tables, and dictionary tables.

 Upgrade Wizard - Copy in System Tables



5. Click **Start** to run the script, then **Next** to continue.

Upgrade Wizard - Check database

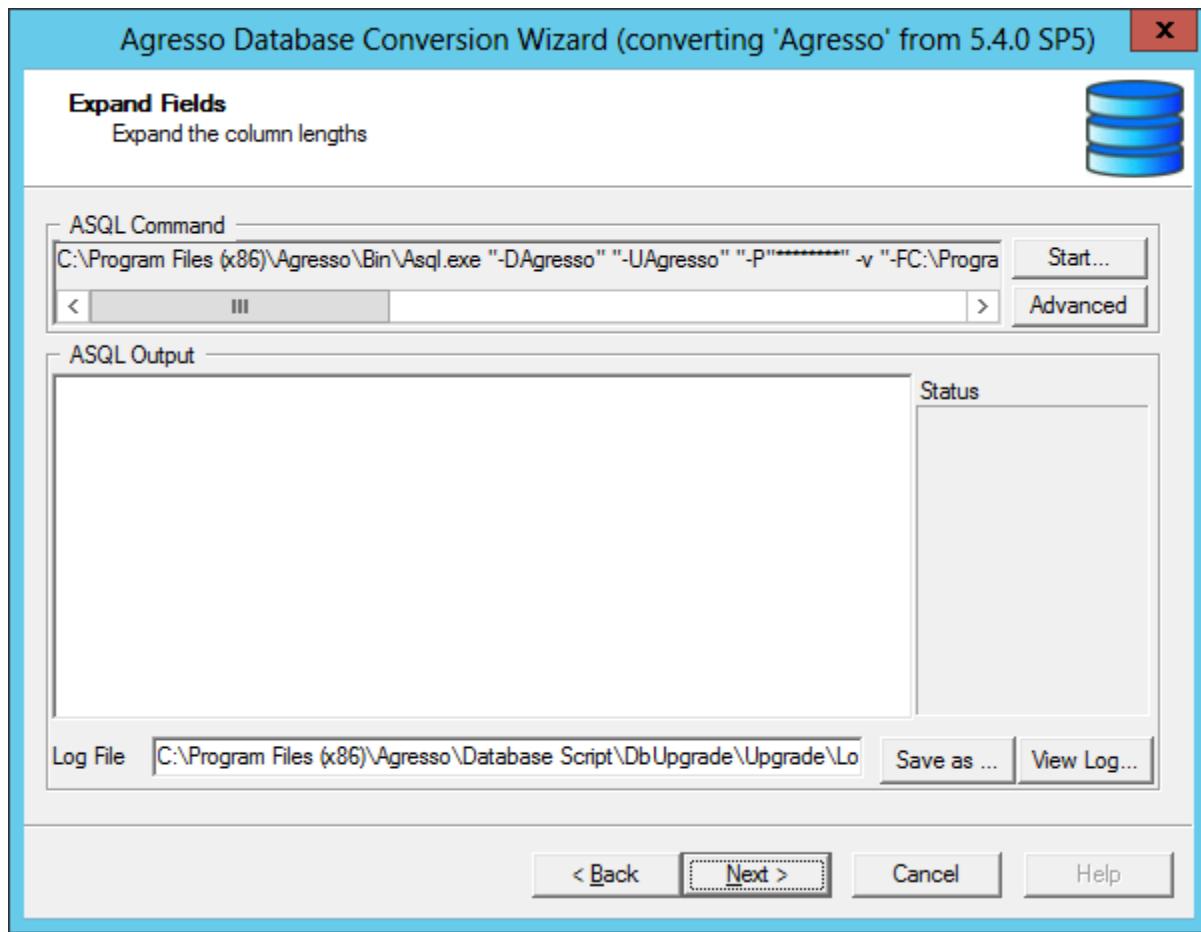


This step verifies that the database is in the expected 5.4 format. This check might take some time.

6. Click **Start Check** to check the database. Click **Next >** to continue when detected issues are verified or solved.

Next, the table columns will be expanded:

[Upgrade Wizard - Expand Fields](#)



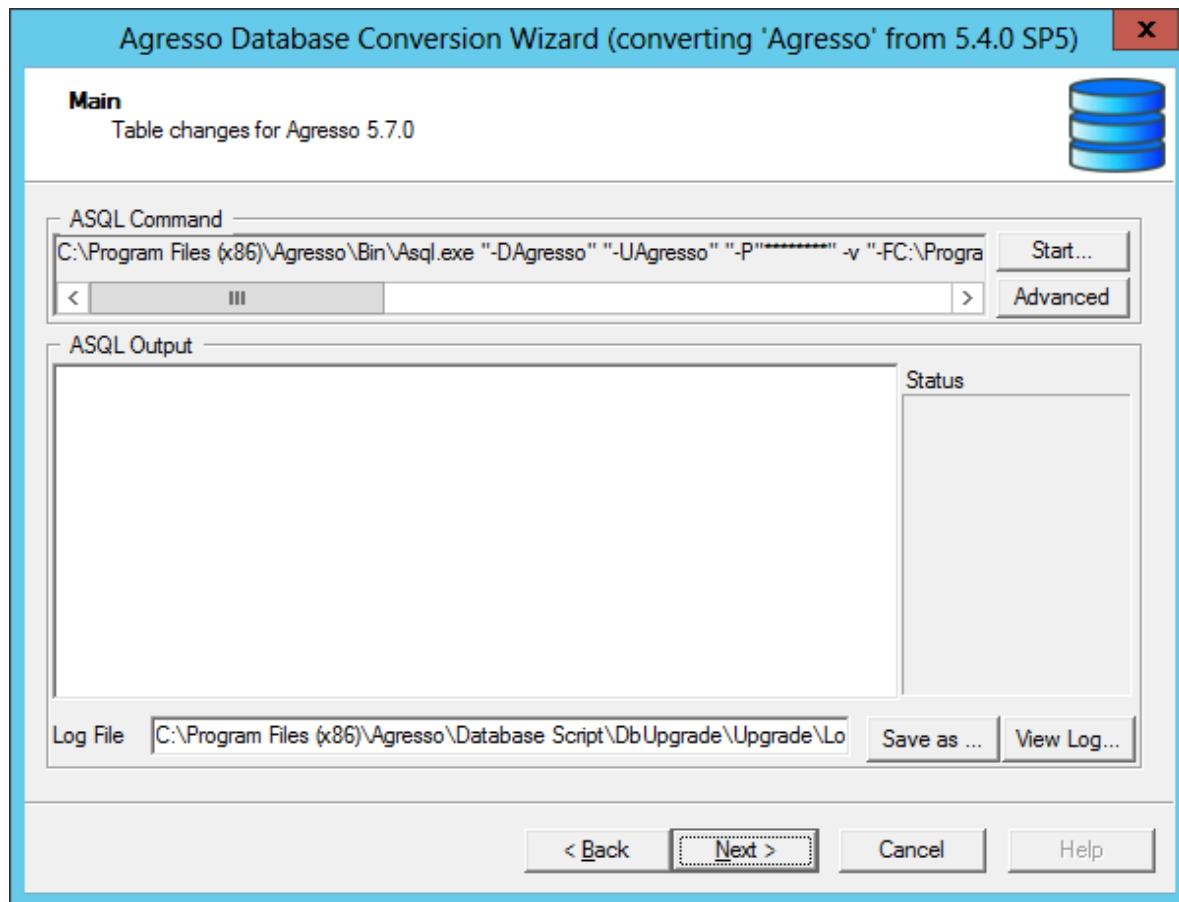
7. Click **Start** to run the script, then (when the script is finished) click **Next** to continue.

Note: This step might take some time.

Duplicates should be removed: If duplicates are found, it is recommended that the duplicates are removed before you continue. You do not have to rerun this step after the duplicates are removed.

You can now introduce the main changes in the database structure:

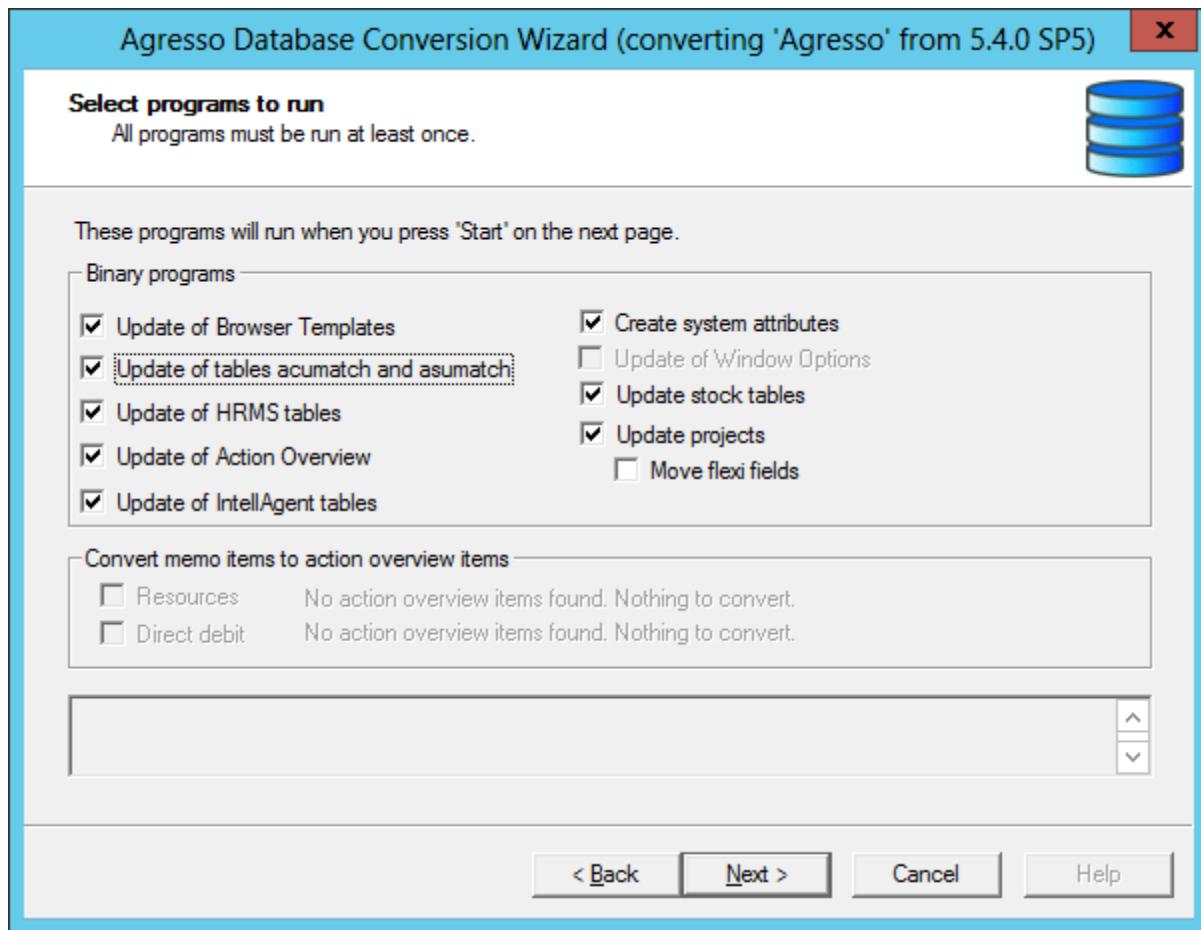
Upgrade Wizard - Main - Table changes for Agresso



8. Click **Start** to run the script, then (when the script is finished) click **Next** to continue.

The final changes to the database structure requires that you run some additional programs.

Upgrade Wizard- Run binary programs

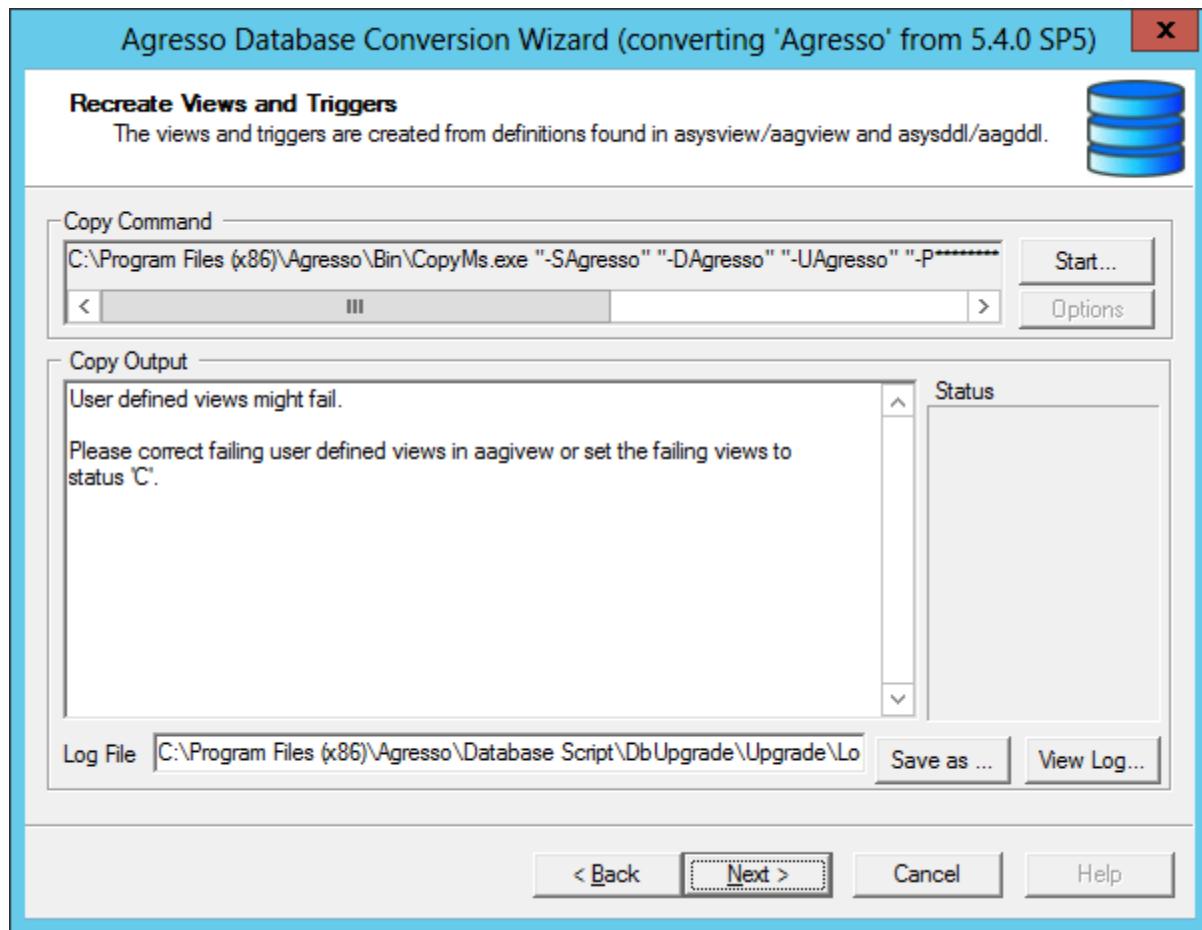


9. Select all programs listed and click **Next**.

Note: All programs must be run , but not necessarily at the same time. It will not do any harm if they are run more than once.

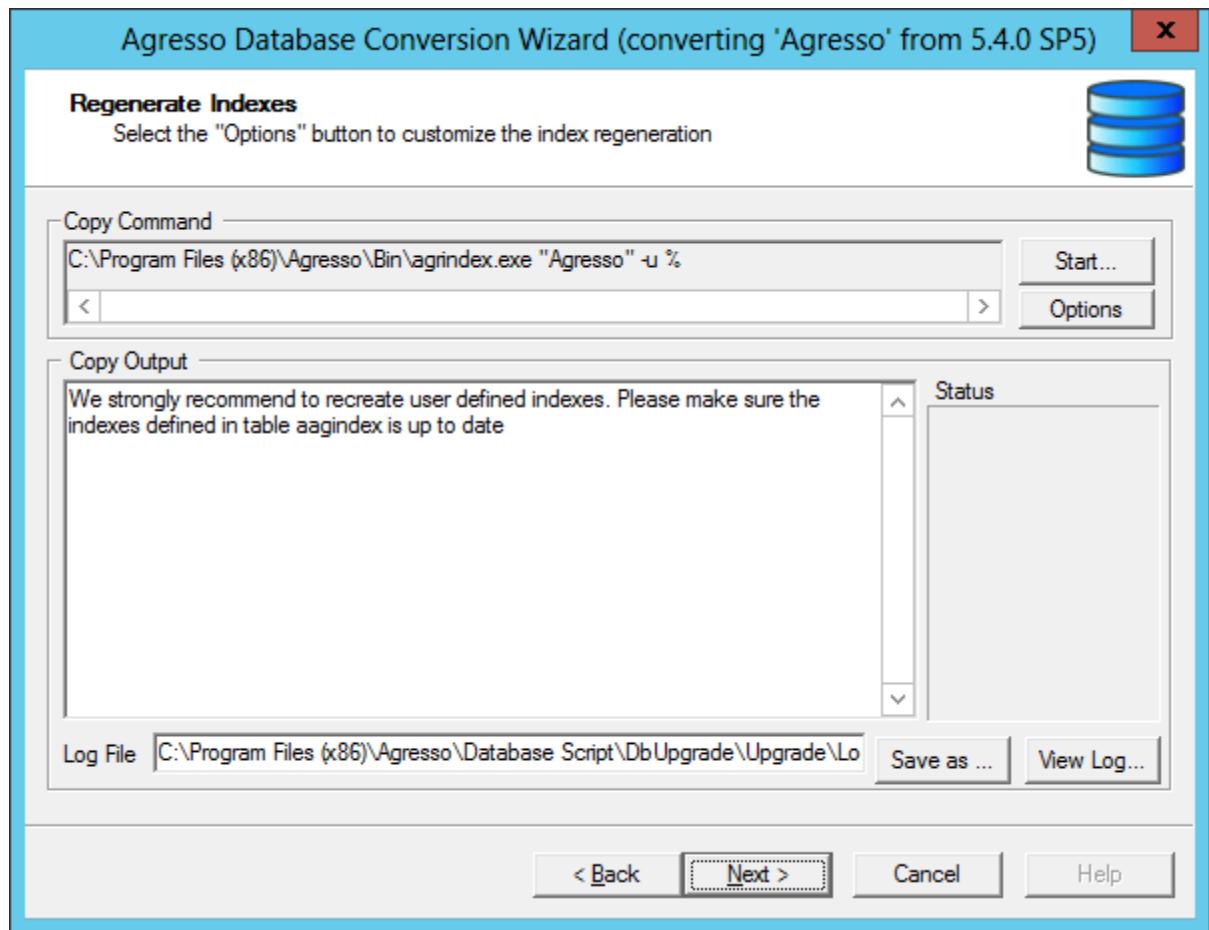
10. Click **Start** to run the selected upgrade programs, and then **Next** when the programs are completed.

 Upgrade Wizard - Recreate Views and Triggers

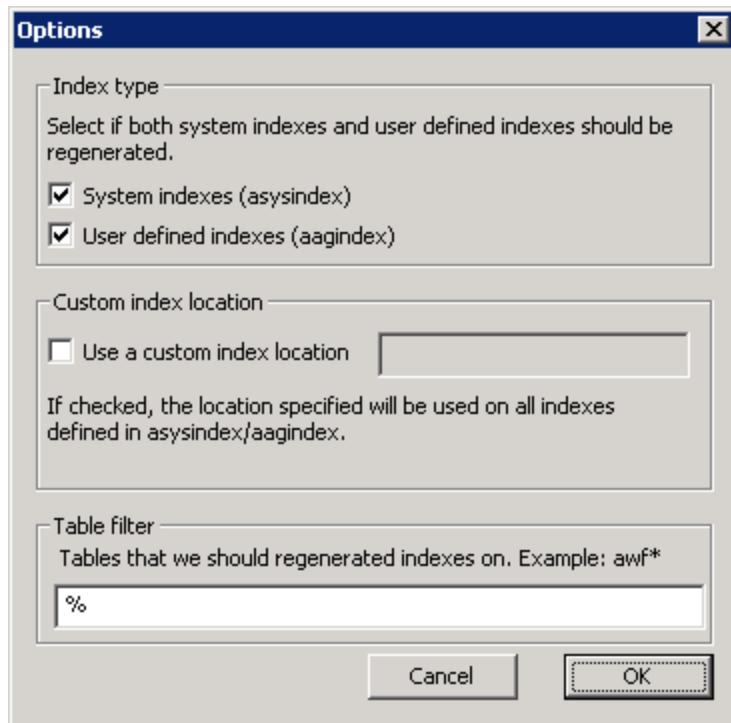


- 1.1.** Click **Start** and **Next** in the next windows, in order to restore system views and indexes. This will bring you to the final step.

Upgrade Wizard - Regenerate Indexes



12. Click the **Options** button to open the **Options** dialog:



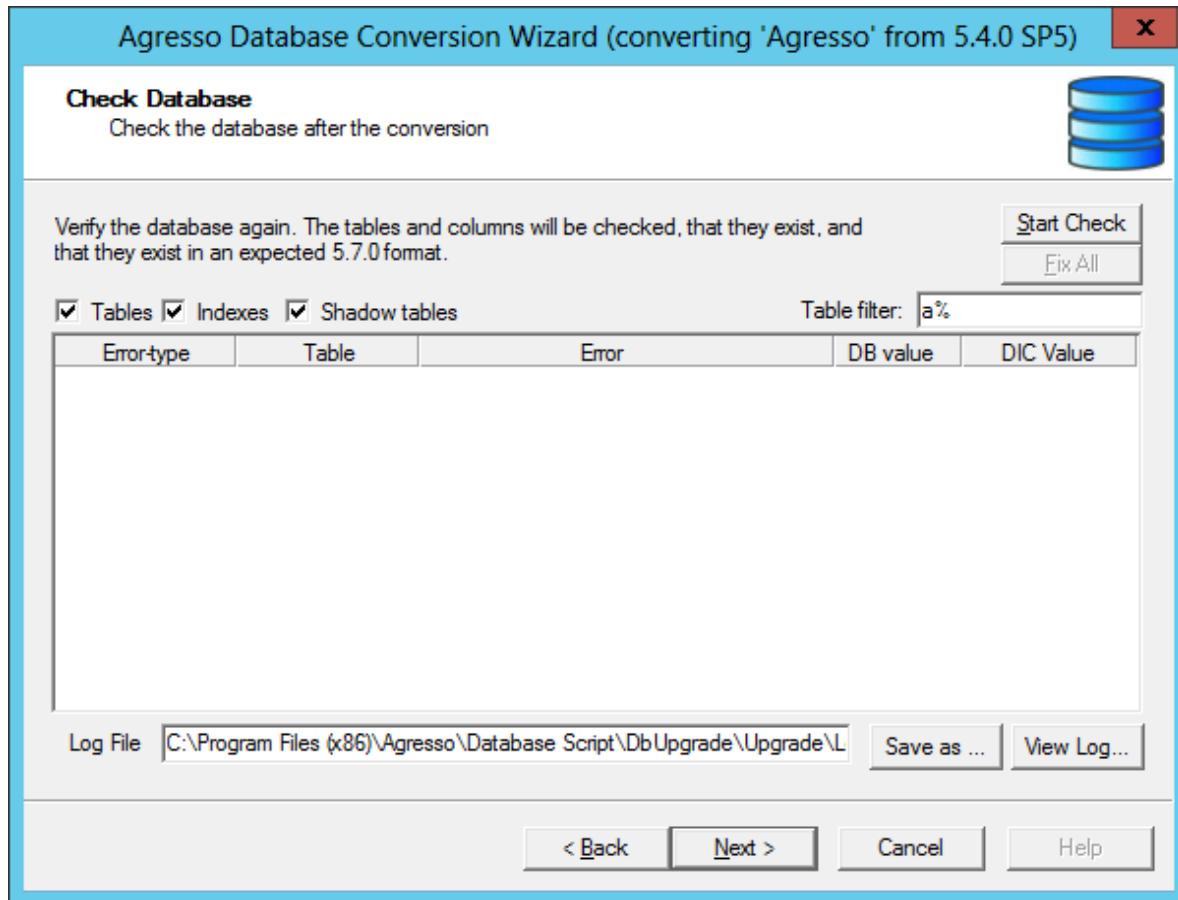
A note about indexes

User defined indexes (stored in [aagindex](#)) are recreated by default. Due to table changes, you should also change these, to avoid reduced performance. For details. see [AgrIndex](#).

13. Select options and click **OK**. Then click **Next** to continue with next step (Check Database).

This check might take some time.

 Upgrade Wizard - Check Database



14. Click **Start Check**.

This check compares the table structure, tables, columns, index with the format defined in Agresso's agrsys* tables.

If any differences occur, double-click on the error line to correct it. Or try the Fix All button. The Fix All might take some if tables with a lot of data needs to be fixed.

It is important to have the database with all the tables, columns and indexes in an expected format.

If there are tables with duplicates, clean up the duplicates and regenerate the index. A missing index might be crucial for the performance.

The following differences can be ignored:

Columns in the table are not found in the dictionary.

Columns are longer than expected.

Complete the Main Upgrade

Prerequisites

The output file *user_def_obj.txt* will contain all necessary information for restoring user defined views and objects. It will also list all duplicate user Ids (Oracle only).

In order to restore previous, custom functionality, you will require *user_def_obj.txt*, but also a detailed description of the database structure of the latest version of Agresso. See relevant Table changes in the Appendix (Agresso Data Dictionary).

Tasks

To complete the main system upgrade, you should do the following:

1. Restore user defined objects and views
2. Register license

Restore user defined objects and views

A note about amendment logging

During the upgrade, all amendment logging are stopped.

Update tasks

The table below describes how custom objects can be upgraded:

Object type	Update tasks
Triggers and indexes	Use a database tool to change the columns and data types to fit to the new database structure.
Procedures	Use a database tool to re-define the procedures according to the new database structure.
Amendment logging (shadow tables)	Amendment logging triggers needs to be recreated after the upgrade. Use the Smart client to open the AG30 Activation of logging server from System Administration Data Control Amendment logging , and regenerate active logging triggers.
User defined views	Use the Smart client to open AG17 Database view definition . Zoom into each query, make the necessary changes, and remember to save the changes to have the views recreated.

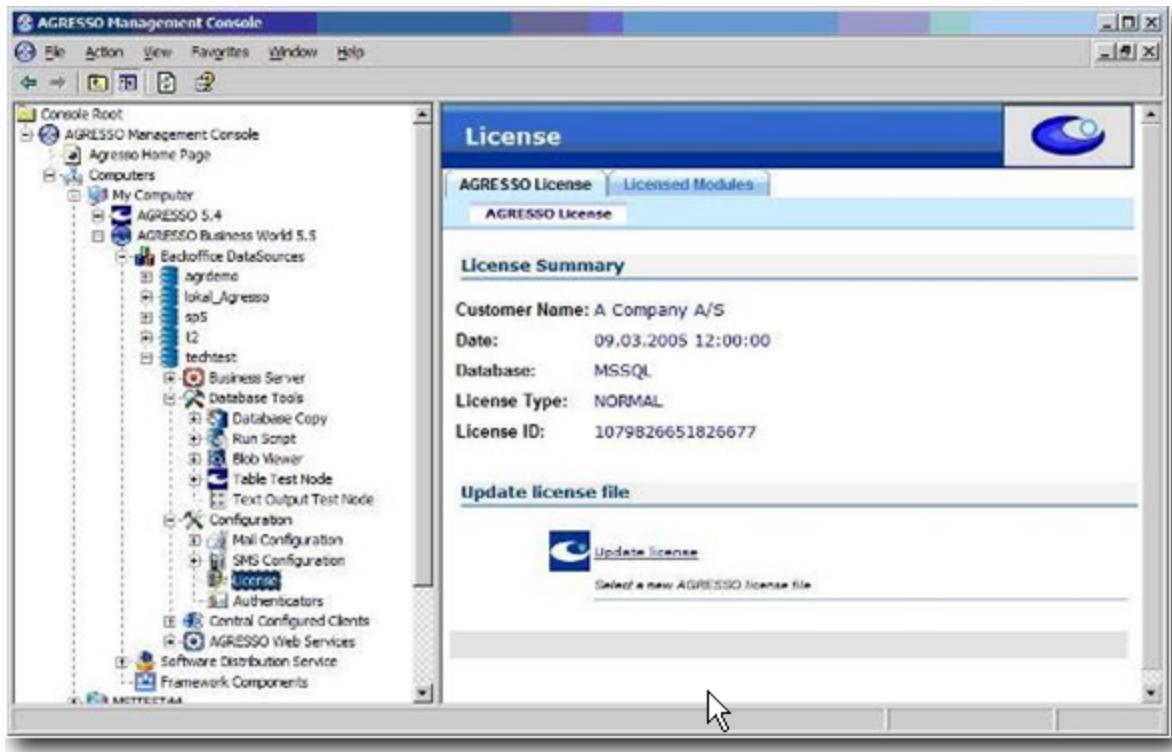
Register license

View licence

The license node in AMC allows you to add a new Agresso license to the database. It also enables you to view the licensed modules and the number of users registered for each module.

If you have a demo license installed, this node will inform you if the license will soon expire, or if it has already expired.

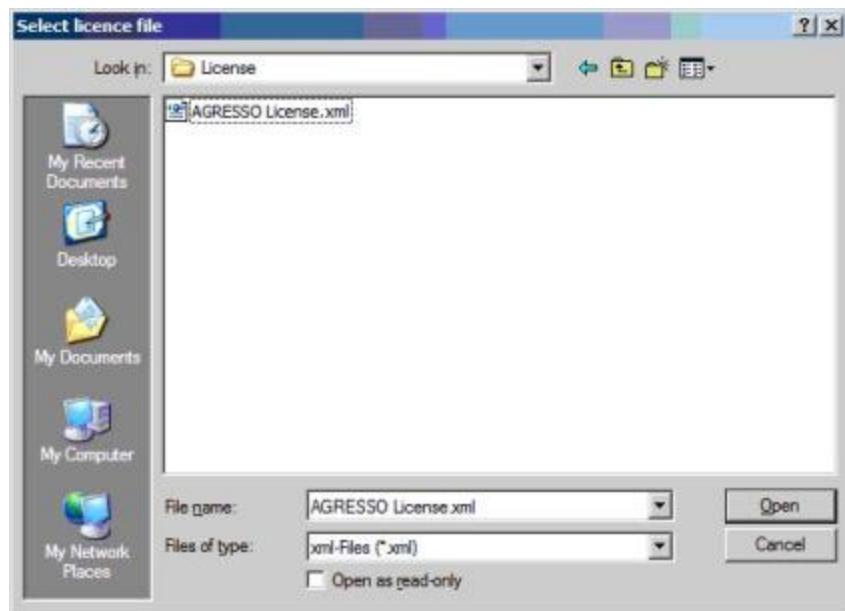
 [View License](#)



Add License

To add a license you can either right-click on the license node and select **Add License** from the **All tasks** context menu. Or you can click on the [Add License](#) or [Update License](#) link found in the right pane of the management console. Then you will be prompted to select a license file.

Add License



You can also add a license using the Agresso Client Configuration tool.

Note: You need to enter a *new license* when upgrading from 5.4. to make all parts of Agresso work. If required, contact your local Agresso support office.

Login info

Your are now ready to test your UNIT4 Agresso application with the following credentials (Note: `upgr55` is correct name after upgrading):

Username:	upgr55
Client:	<your client>
Password:	upgr55

Additional System Upgrades

You can now proceed with Additional system upgrades and setup.

Invoice Manager and Workflow

Agresso Workflow

New workflow solution

The workflow module (introduced in version 5.5 of ABW) replaces the Compello system. The new workflow module is based on an entirely new data model. Some Compello tables are kept to show historic workflow maps and are renamed to `ac<compello table name>`.

Wizard parameters

The Workflow upgrade wizard converts the data in the Compello tables to the new data model and defines processes in the new workflow system to match the existing set up in Compello.

Note: The wizard also makes it possible to upgrade rules from Compello. This is, however, NOT recommended, as the new workflow solution allows you to create far more advanced and generic rules than before.

Parameters

During the upgrade, you will be prompted to fill in some necessary parameters (see 3. below). These are explained as follows:

Parameter	Default value	Description
Client	*	The Agresso client to upgrade. Wildcards can be used (for example: *)..
Run as user		The user that will run the upgrade wizard.
Date From	19000101	Format: YYYYMMDD. Older document, transactions and images will be ignored.
Date To	20990101	Format: YYYYMMDD. Newer documents, transactions and images will be ignored.
Convert Invoice Man-	Off	Select if fixed registers from Compello shall be upgraded. If checked:

ager registers		<ul style="list-style-type: none"> Users – Users will be created and the workflow flag will be checked for them. Groups – will be converted to Roles Substitutes and supervisor definitions
Convert Invoice Manager rules	Off	<p>Only selectable if the Convert Invoice Manager registers is selected or the registers were upgraded in a previous run of the wizard.</p> <p>If selected, all Compello rules will be upgraded to the new rules tables.</p> <p>Note: This is not recommended (see The upgrade wizard above).</p>
Invoice Manager maps	Off	<p>If selected, historic transactions and Compello map data will still be available. The old maps can be viewed from the new map viewer.</p>
Create template workflow processes	Off	<p>If selected, all template workflow processes will be copied to the clients chosen for upgrade. The template processes are drafts and will need to be committed by the user chosen in Run as user (see above) to become active.</p> <p>Note: This will overwrite any current workflow processes for the clients being upgraded.</p>
Convert Invoice Manager archive	Off	<p>If selected, Compello images will be upgraded to the new solution, and UNC-path (below) will be enabled.</p>
UNC-path		<p>UNC path to locate existing Compello documents. NB! Must be UNC path!</p> <p>Example: <code>\compelloarchive\images</code></p>

Upgrade Invoice manager

Upgrade consequences

The result of the upgrade processes described below is that

- the Compello COM application is completely removed,
- users will have direct access to the file share.

Dangers: The Compello document location must be configured as a file share (UNC name) with full access for everybody using the document archive. The share can be hidden, but there is no way of stopping an advanced user from mapping the drive and accessing the documents.

In practice, no one needs more than read-only access to Compello documents, and the documents are scrambled to prevent casual browsing.

Protection: We have tried combine these requirements, and reduce the security risks, by creating a dummy Windows user, which Agresso will use to access the documents. The system parameters DS_NATIVEFILE_USER and DS_NATIVEFILE_PWD allows you to configure this user. The dummy user (and no other) needs full access to the file share. When you run the document archive system within Agresso, the rights of this dummy user are picked up. Any attempt to map the file share outside of Agresso will be denied.

New document archive and old images

With the new document archive, customers with historical Compello images can select one of the following options:

- Leave existing documents in the Compello folders as read-only, for historical access only
- Move the documents into the Agresso Document archive (and remove them from the Compello database).

Required upgrade

To be able to view Compello documents in Agresso, you must at least:

- 1.** Run the upgrade wizard to convert the database tables. (This can be done in stages by date etc.)
- 2.** Configure permissions for the Compello file share.
- 3.** Create a document type to access the Compello documents, for example Compello Invoices. This should belong to the GL Transaction key, and have the document system Compello selected.

This will change the pointers to the data only. The image files are not moved.

Complete migration

You may want to make a full migration for the following reasons:

- To have one document type only for all invoices, both old and new
- To be able to use the new Agresso document archive features

If you prefer a complete migration and remove the Compello archive, you can use the server process **DS01 Document copy** to move documents between document types. If the document types are based on different storage systems, as in this case, it automatically uses the drivers to do the conversion.

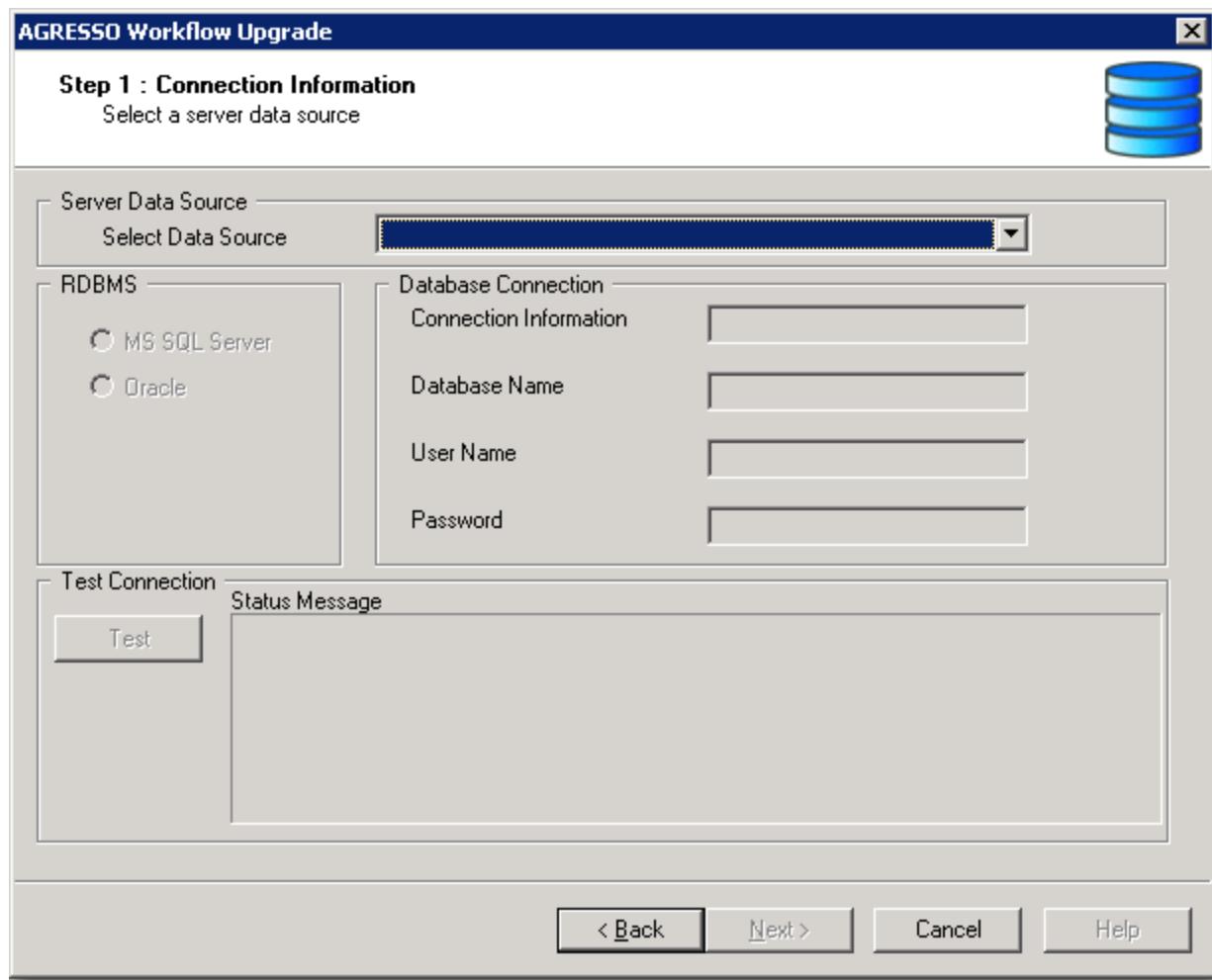
Run the wizard

Do as follows:

- 1.** Start the upgrade wizard and move to the Step 1 dialog:



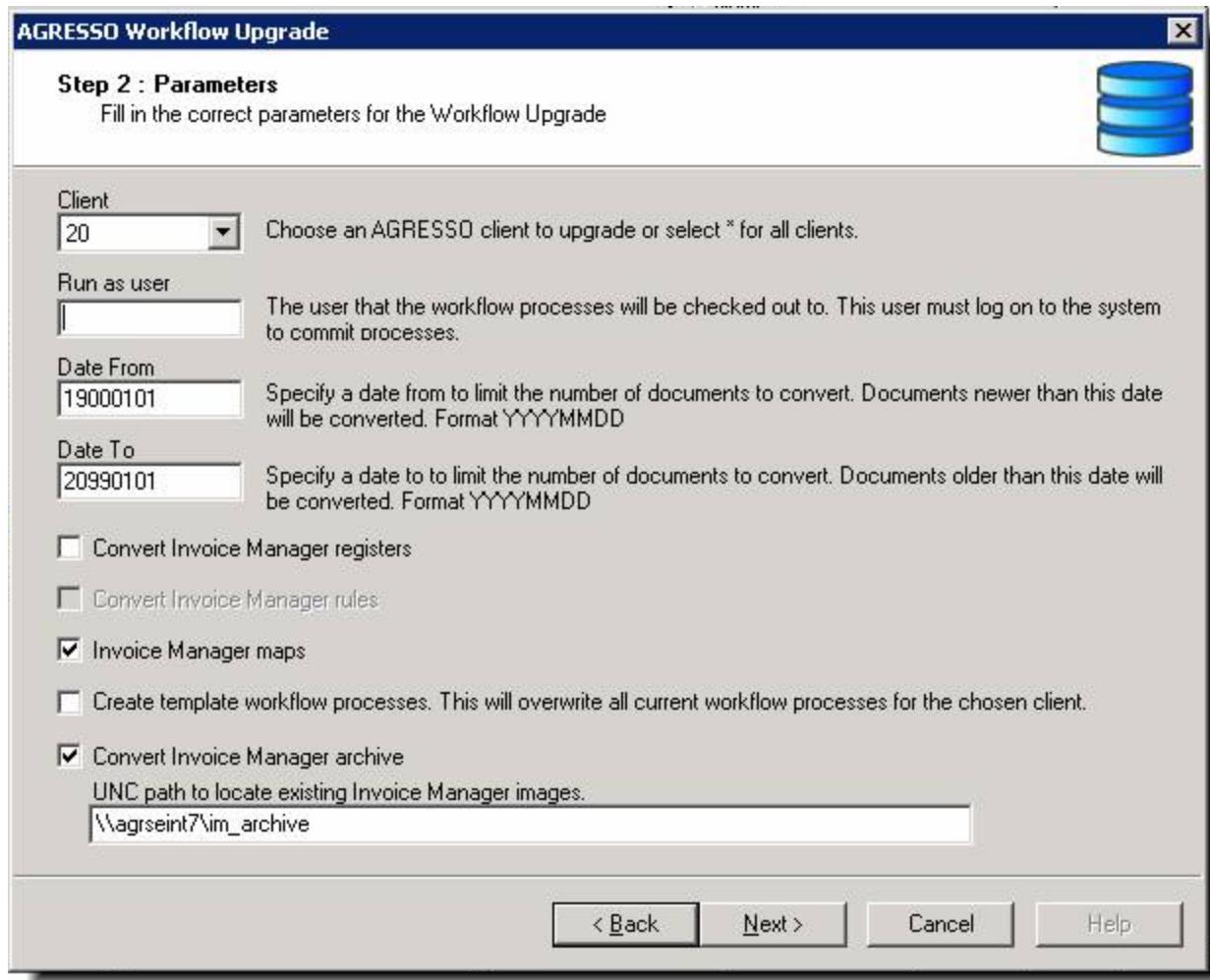
[Workflow Upgrade - Step 1](#)



2. Select correct data source from the drop-down list, fill in the correct password and click **Next**.

You are now prompted to set necessary parameter values.

Parameters - Step 2



3. Make sure the parameter settings are correct, and click **Next**.

You are ready to run the upgrade scripts.

4. Click **Start** to run the script and the **Next** to continue.

The (basic) upgrade is completed and you can close the wizard.

AUTHORIZATION AND ROLES

The ABW authorisation structure

ABW access for users and roles

A new and flexible structure for users and roles, with regard to access rights, was introduced in ABW 5.5.

An ABW user can belong to several roles, and the user's access rights will be the sum of all the roles' access rights. In addition, a user can have additional access rights independent of roles.

User type per client

ABW also supports the concept of user types. A user type is a definition of the user's general relationship to a specific client – independent of roles. This relationship is established by linking the user to another data register in Agresso (Resources, Customers), thus making it possible for Agresso to understand when a certain user is identical with a certain entity in the linked register.

Example: If user myname is set up with user type Resource and linked to the resource 12345 in Resource master file, Agresso will allow myname to enter hours worked for resource 12345.

It is not required to define any user type, and for each possible user type value, only one specific relationship can be defined per client. The same user can have different user type relationships to different clients.

Resources only: Currently only Resource is fully supported as user type.

User type values

A user type is defined by linking the user to a registered entity in one of the Agresso master files. The table below describes the valid user types and linked registers:

User type value

Means that the user id will be linked to...

Customer

Customer id in Customer master file. Not currently supported.

Supplier

Supplier id in Supplier master file. Not currently supported.

Resource

Resource id in Resource master file.

Applicant

Applicant no in Applicant master file. Not currently supported.

Student

Not in use the standard ABW application.

ABW 5.4 to 5.6 conversion challenges

The 5.4 solution

ABW 5.4 supported users, authorisation groups, user groups, web roles – and resources. Due to the real world needs, many users were set up with their private groups or web roles, and the number of groups could be very large.

The new role concept was introduced to straighten out this complexity.

Utilising roles

A role is an abstract entity with certain access rights to Agresso, and meant to correspond to the real roles a person can enter into in her professional life.

Example: As a sales representative, you will share several properties – and functional needs – with other sales representatives. Your company will therefore create the role Sales representative in Agresso, and grant (all members of) this role the required access rights.

Incompatible data structures

The new data structures (tables) introduced with ABW 5.5, differ from the 5.4 structures with regard to user authorisation.

When you run the default upgrade programs, the previous user data will be converted, but as there are no roles to convert, there is no easy way to establish a new role based authorisation structure.

Since the user type in ABW 5.6, such as resource, is linked to a user through role membership, you will also lose the connection between user and (the 5.4) resource.

Upgrade alternatives

Upgrade from scratch

The safe but somewhat labour-intensive way to proceed is to skip the Authorization upgrade wizard after the main system upgrade wizards have been run. Instead, you must create a new role structure according to your company's needs, and then link the various users to the roles they shall belong to.

When the role structure is in place, you can run the Re-connecting resources and users wizard to re-establish the link to the resource user type.

Automatic upgrade

The installation package comes with an upgrade wizard - Authorization upgrade - that will help you to quickly establish a role structure. This wizard will

- automatically create roles that correspond to the previous groups (user groups, authorisation groups, web roles)
- keep the links between user ids and resources.

The result, however, is that you will get an authorisation structure which is just as complex as it was in 5.4. Each 5.4 group or web role will be converted to a new role in ABW 5.6. This may be practical for some companies, although there will probably remain some manual clean-up tasks (after the wizard has run) before the new role concept can be fully utilised.

Upgrade from scratch

Description

During default upgrade, the upgrade programs created a temporary table, upgconntmp, where the 5.4 links between user id and resource id is stored – waiting for the roles to be defined.

Reference: See Release Notes for ABW 5.5 SP1, Data control and User access for details about role management.

When you have defined the roles, and linked users to their respective set of roles, the system is ready for the Re-connecting resources and users wizard. This will re-establish the user – resource connection from 5.4, within the defined role structure.

Explanation: Since the autorisation data structures are different from 5.4, it gives no meaning to partially populate this structure automatically before the roles are established.

The aaguserdetail table

The aaguserdetail table contains one row per user, role and client. Two empty columns, Attribute id and Dim value, await the updates from the Re-connecting resources and users wizard.

Wizard

The wizard consists of two main (compiled) scripts, with the following tasks:

1. For all users in the temporary table upgconntmp:
 - a) locate all rows with the corresponding user id in the new table aaguserdetail, and
 - b) fill in the resource id (in the Dim value column) from upgconntmp for all rows.
2. Delete the upgconntmp table.

Running automatic upgrade

Consequences of automatic upgrade

If you want quick results, and prefer to run the Authorization upgrade wizard, you will probably end up with at least one role for each user, and a very complex authorisation structure, difficult to maintain. Still, if you are not dependent on advanced authorisation management, the upgrade wizard is there for you.

The wizard

The Authorization upgrade wizard is available in the .\DatabaseScript\DbUpgrade\Upgrade folder. It will upgrade Authorization groups, User groups and Web roles to the new role concept. If you upgrade Authorization groups access, the Smart client will also be upgraded.

Important: The access rights for upgr55 will be removed. Access to the Self service client can not be upgraded because the menu structure has been completely changed.

Document Archive Upgrade

New Document archive solution

The Agresso Document archive was completely rewritten with Agresso 5.5. The document archive is now an integrated part of a series of Agresso windows, and documents of any type (.doc, .pdf, .jpg etc.) can now be attached to most of the Agresso object's.

These profound changes require that all the previous documents must be converted.

Please refer to *Agresso 5.5 Release Notes, Document Archive* for details about technical and functional aspects of the new solution.

Invoice Manager Reference

See the topic [Workflow and Invoice Manager](#) for information on how to interface your old Compello images with the new Agresso document archive.

Running the upgrade wizard

Necessary preparations

During the upgrade, you will need to make a few selections in order to proceed (see 3. below). The parameters you must enter are explained in the following table:

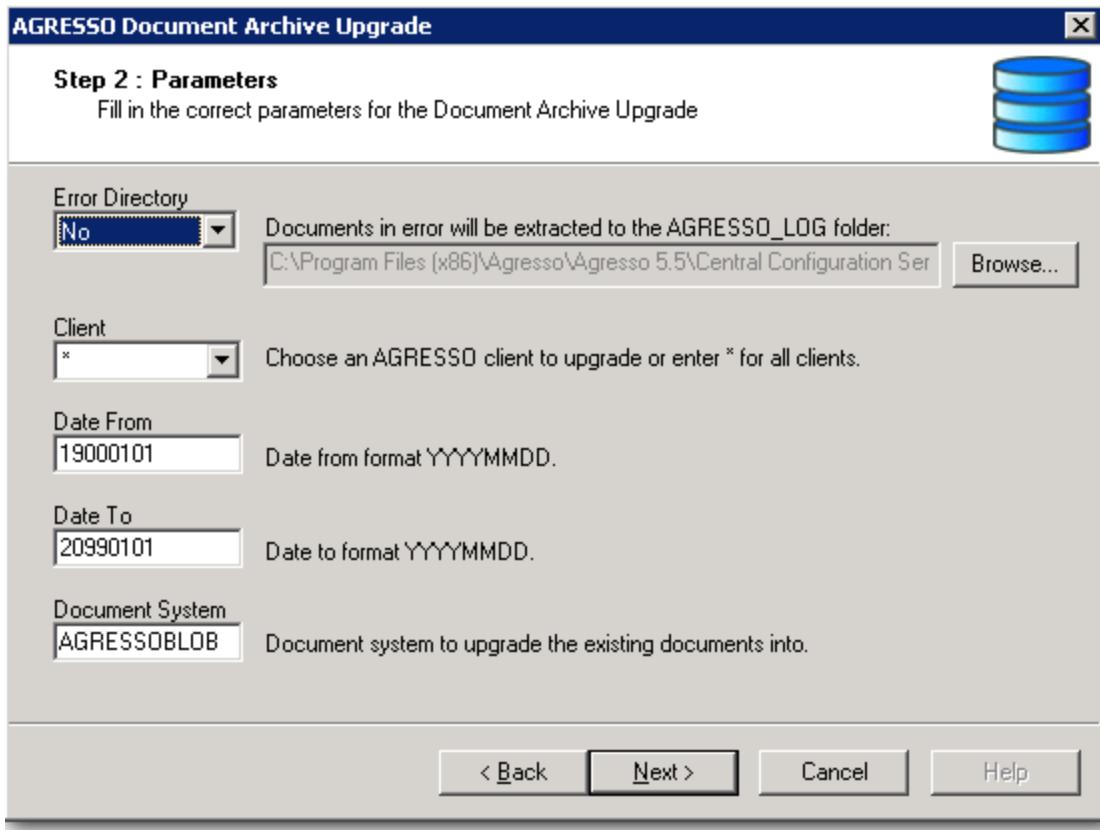
Field	Default value	Description
Error Directory	No	<p>Yes or No.</p> <p>If Yes, erroneous documents will be extracted to the server logging (AGRESSO_LOG) directory for manual fixing.</p> <p>If No erroneous documents will just be reported in the log (not extracted).</p>
Client	*	The Agresso client to upgrade. An asterisk (*) means all clients.
Date from	19000101	All documents registered at or after Date from will be included. Older documents will be ignored. Format: YYYYMMDD.
Date to	20990101	All documents registered before or at Date to will be included. Newer documents will be ignored. Format: YYYYMMDD.
Document system	AgressoBLOB	The new document archive can be connected to many document systems, including third party archives. You must choose a document system to upgrade the existing documents into. The default value is the new Agresso database archive.

Upgrade procedure

To upgrade to the new document archive solution, you will first start the Upgrade wizard. Continue as follows:

1. Double-click the [Document Archive](#) link in the Database Upgrade Wizards screen. After reading the introduction, click Next.
2. Enter connection information and click Next to prepare for step 2.

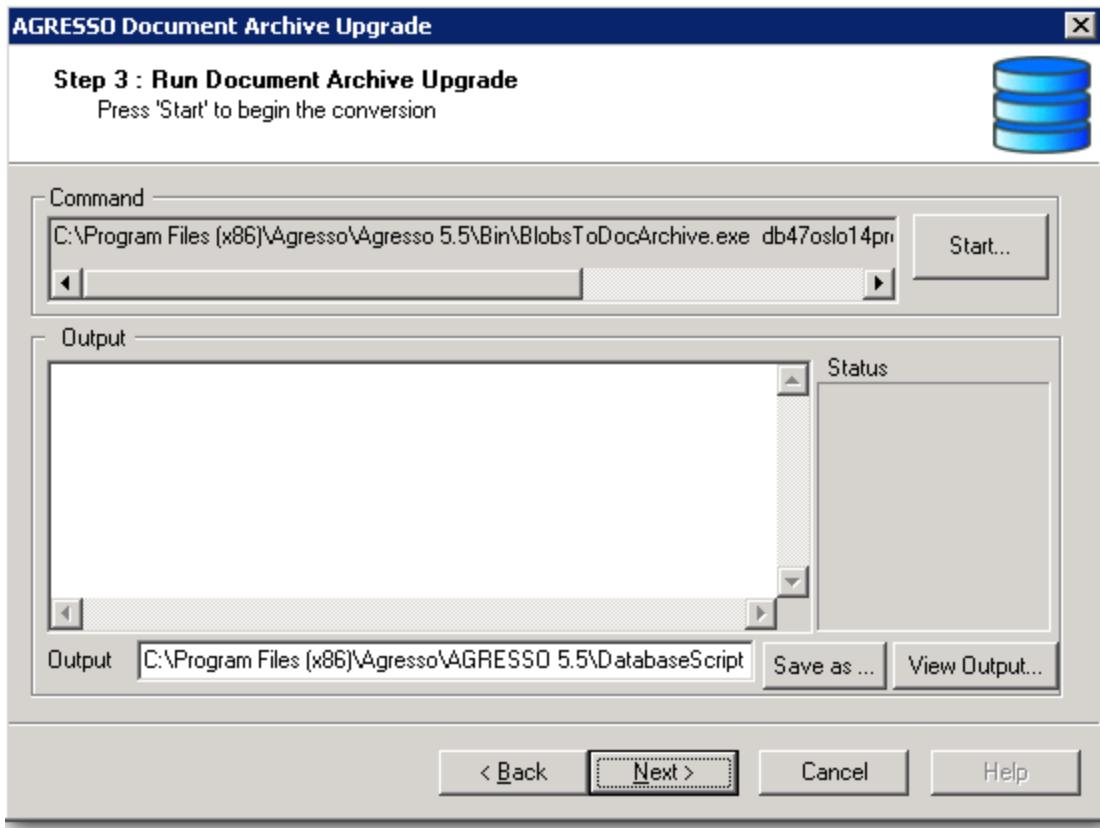




3. Fill in the required parameters and click **Next**.

You will now have to verify the path to the log file (output) made by the upgrade scripts.

 Document Archive Upgrade - Step 3 #1



4. Make sure that the output path is correct, and run the upgrade by clicking the **Start** button.

5. When the conversion is done, do as follows:

- a. Click **View Output** to study the conversion results
(we presume everything is in order)
- b. Click **Next** to bring up the final window.

6. Click **Finish**

Note: If necessary, check the log file once more and make sure everything is correct.

Manual setup tasks - Test

The following areas require manual setup:

- From View options to Window options - See Release Notes for User Interface, ABW 5.5 SP1.
- Data control - see Release Notes for Common and System Administration, ABW 5.5 and 5.5.1 combined.
- Management of open items and Action overview - see Release Notes for Action Overview, ABW 5.5 and 5.5.1 combined

Responsibles Upgrade

Changes in data structure

The new solution for users and roles has required a completely new implementation of the *Responsible* concept in the Logistics module. Previously, the *algresponsible* table hold information about responsible codes and the *resource_id* (not the *user_id*) that was linked to the code.

Agresso 5.5 introduced a solution where a responsible belongs to a certain role, and where role membership is based on the *user_id*.

Upgrade of orders created by Agresso 5.4x

The main purpose of this Responsibles Upgrade, is to convert active orders (created in the various logistics modules) in such a way that the responsible person still can be identified when the order is further processed.

Note

The upgrade will *not* convert the old responsible resources into new responsible *roles*.

Upgrade tasks

The main upgrade tasks are performed by the responsible upgrade wizard, and are described as follows:

Task	Description
Automatic code matching	Match resource_ids from the <i>algresponsible</i> table with the 5.7.0 users. If a user can be uniquely identified by a resource_id, the upgrade wizard will establish a new connection between the 5.4 responsible code and the 5.5 user id.
Manual matching	For all unmatched resource_ids, you are prompted to manually enter user_ids. There is no requirement that the new user really takes part i 5.5 responsible role. This task can be repeated several times, until all (old) resource_ids have a valid match.
Automatic order upgrade	The wizard will search through all relevant orders and convert all orders with a responsible codes where there has been a sucessful match. Log: The result will be written to the log, and can be viewed in the table <i>algprophead</i> .
Manual order update	For all the orders that were not converted, you will have to use the registration windows for the various order types, to set the correct responsible. Note: This requires that the various responsible roles are set up in the Responsible Setup window in the Smart client. See <i>Agresso 5.5 Release Notes, Logistics – Common logistics</i> .

Run the upgrade wizard

When the **Responsibles Upgrade wizard** is up and running, navigate to the Step 1 window, and do as follows:

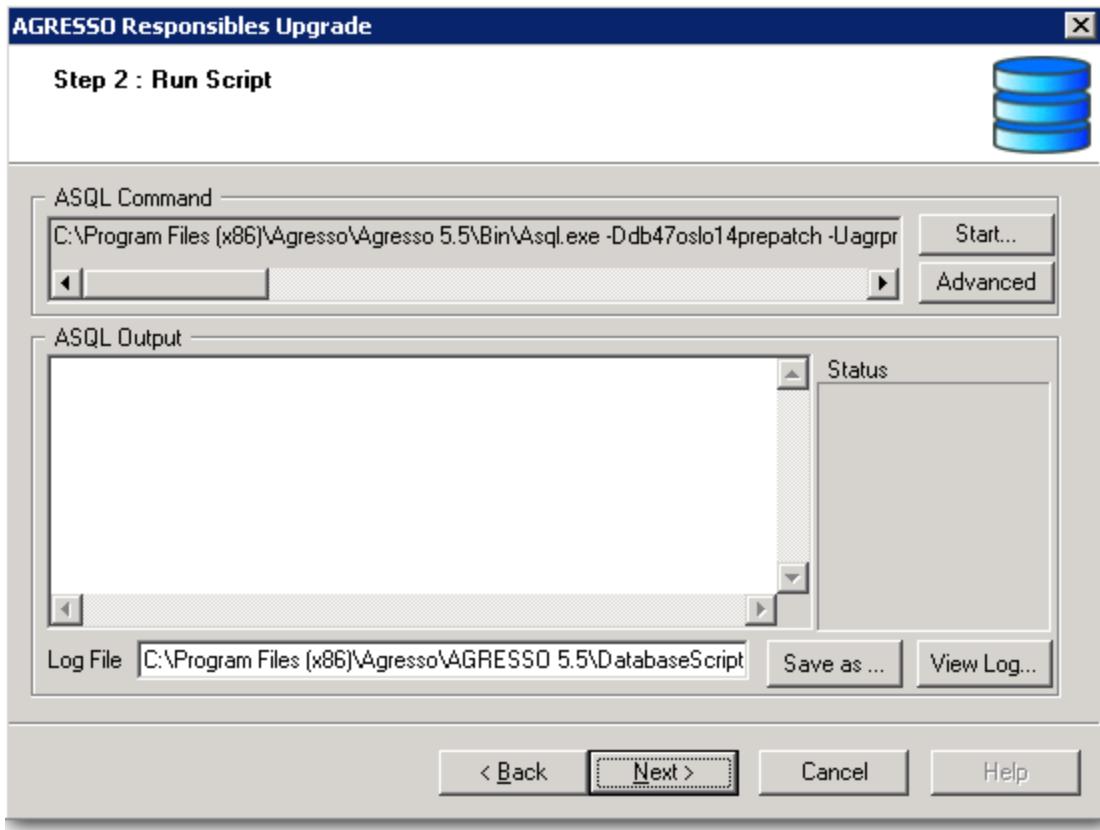
1. Select correct data source from the drop-down list, fill in the correct password and click **Next**.

You are now ready to run the upgrade scripts.

Note: If you already have run the upgrade scripts once, and in addition performed some manual matching (3. below), a re-run of the wizard will remove all manual updates!



Responsibles Upgrade - Step 2

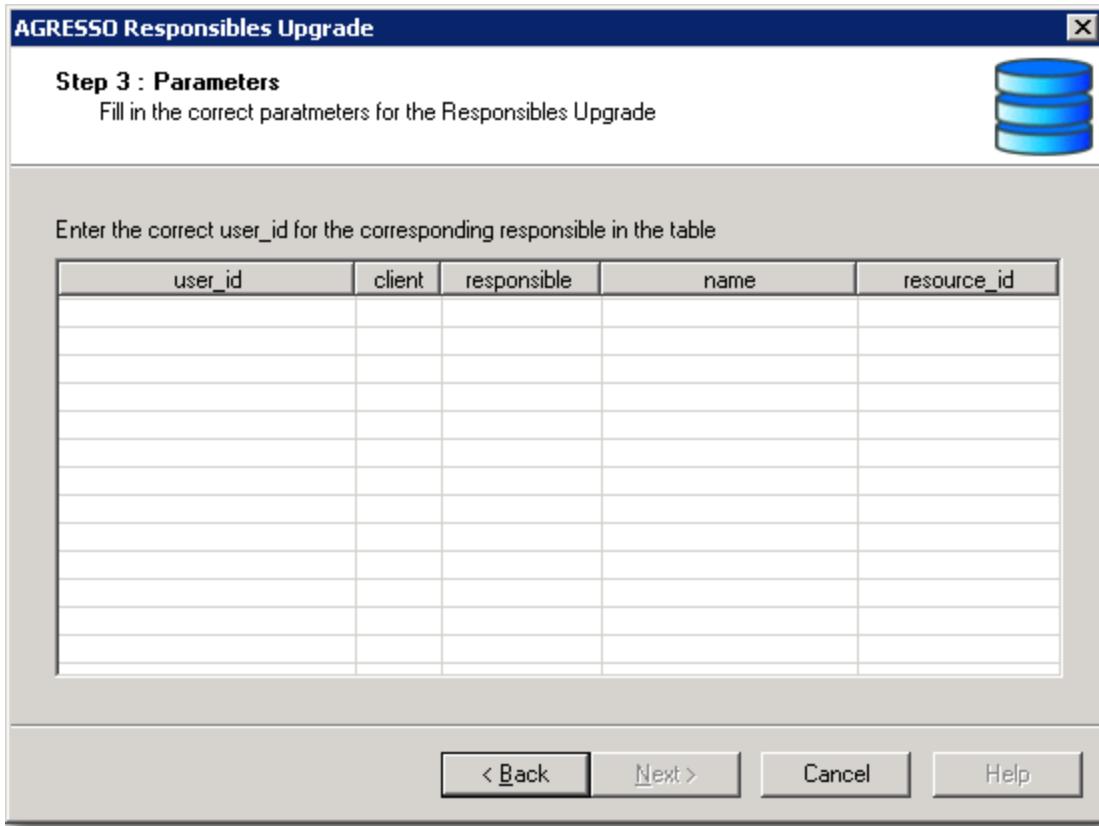


2. Click **Start** to run the script, and then **Next** (we assume that everything went OK after **Start**).

The wizard will match all identifiable resource_ids with the 5.5 user_id. When completed, the status for the *algresponsible* table is set to S.

You will now be prompted to manually enter missing user_ids:

 **Responsibles Upgrade - Step 3**

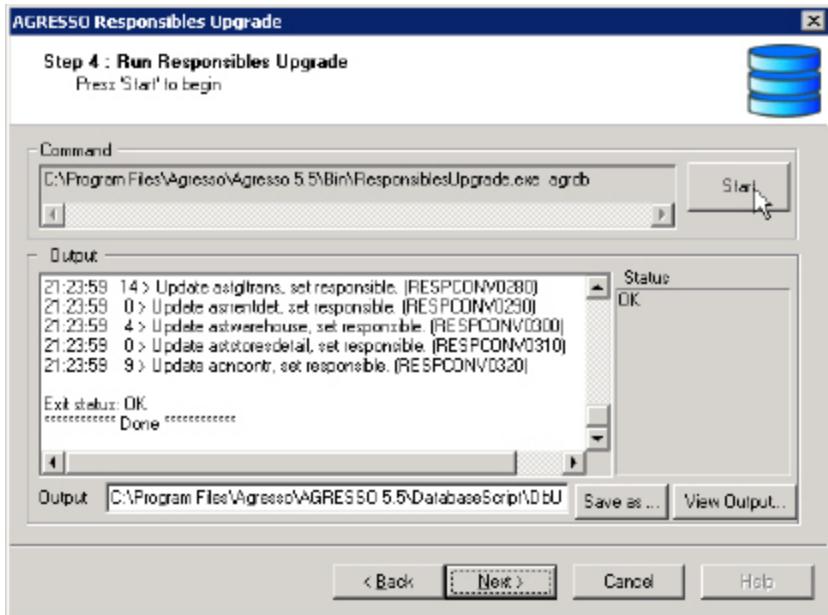


3. Repeat steps a. and b. as many times as necessary:

- a. Add missing user_ids.
- b. Click **Apply**
- c. Click **Next** when you are finished.

The next task is automatic order upgrade.

Responsibles Upgrade - Step 4



4. Click **Start**, and then **Next** to finish.

This wizard can be run several times if you would like to add user_id's at a later stage.

Complete upgrade - additional manual tasks

There will normally remain some unmatched responsible codes on the orders. If so, the responsible roles must be registered in the Agresso Smart Client, and the orders must be updated in their respective registration windows.

HRMS Upgrades and Setup

Wizards

You must run two wizards:

[Balance Upgrade](#) and

[Work Schedule and Absence Details Upgrade](#)

Manual setup

The manual setup tasks are described in the following documents:

- *Release notes, AGRESSO 5.5, HRMS/PCB integration*
- *Service Pack notes, AGRESSO Business World 5.5 SP2, Project*

Projects Upgrade and Setup

Wizards

To upgrade the Projects module, you will need to run the following wizards:

[Balance Upgrade](#) - if not already run,

[Work Schedule and Absence Details Upgrade](#) - if not already run, and

[Timesheet Upgrade](#).

Manual setup

See the documents:

- *ABW 5.5 Reference manual, PCB Setup resources, time costs & income*
- *Service Pack notes, AGRESSO Business World 5.5 SP2, Project*

Finalise Upgrade

Overview

To complete the upgrade, you should:

- Check the converted Browser templates, and correct any errors.
- Clean up duplicates and add indexes.
- Correct all user defined views due to the table changes (see Appendix, Agresso Data Dictionary)
- Remove all tables no longer in use.

Check and correct Browser templates

Tools are provided to administer and ease the upgrade process for Browser templates:

- A Browser checker utility, `BrowserTemplateChecker.exe`, introduced in 5.5 Service Pack 1 that is run after upgrade.
- Check the log file produced during the upgrade. In this log, detailed information can be found of the Browser templates which need to be adjusted to run correctly on the latest version of Agresso. To fix the problem, open the browser template, make the necessary changes and save.

Indexes and duplicates

When running the database check step in the wizard, there might be indexes missing due to duplicates.

Indexes might be very important, and missing or wrong indexes can lead to very poor and slow performance.

Check the log files if there has been any errors while creating the indexes.

See log file `.\Database Script\DbUpgrade\Upgrade\Log\logindex_<database>.log`

Use any database tool to find the duplicates (for example SQL Server Management Studio (SqlServer), SQL Developer (Oracle)).

Example on how to find the duplicates:

```
select distinct client, attribute_id from agldimension  
group by client, attribute_id  
having count(*) > 1  
order by client, attribute_id  
/
```

Delete/change so there are no duplicates in the table, and re-create the index.

Look up in the Appendix, Agresso Data Dictionary to see the correct index definition. Create the index using the preferred DBA Tool.

User defined views

The table structure is changed from release to release. Ensure all user defined objects are changed according to the new structure (See Appendix Agresso Data Dictionary)

Change the definition and recreate the views

See log file .\Database Script\DbUpgrade\Upgrade\Log\logviews_<database>.log

Amendment logging

Amendment logging triggers needs to be recreated after the upgrade. Use the Smart client to open the **AG30 Activation of logging server** from **System Administration | Data Control | Amendment logging**, and regenerate active logging triggers.

Remove tables not longer in use

The script *drop.aspx* in the *Scripts* directory will drop all old and temporary tables no more used by the application.

Note: This script must be run when the upgrade has been completely verified, and you are sure that none of the old tables are needed for backup purposes.

Upgrade overview - Production

Process description

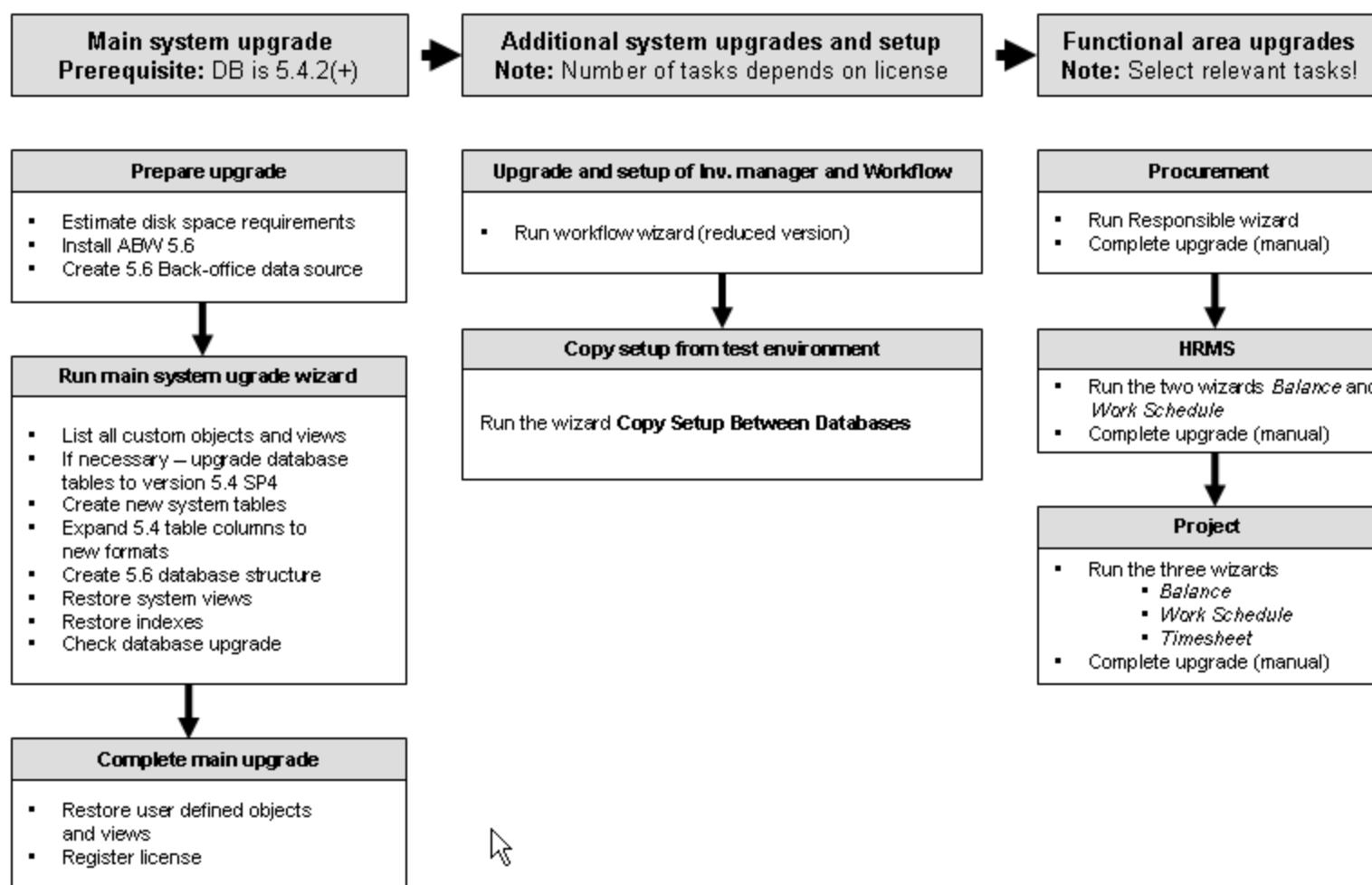
The upgrade activities presented here, starts when you have a (more or less) fully functional Agresso installation in the test environment. In addition, the Production database must have been upgraded to a 5.7.0 database. For database upgrades, see [Upgrade Process Overview](#).

As for test, the upgrade process goes through four stages:

1. The Main system upgrade wizard performs all the required, basic upgrades of your system: It will add new tables, and convert old tables and data to the new formats - as far as it goes.
2. If relevant, you will run the Workflow wizard, with reduced functionality. And instead of manually establish roles and access rights, window options, data control etc. (as you did for the test environment), you will now run the wizard *Copy Setup Between Databases*. This will ensure that the manual setup from the test environment is copied - as is - to production.
3. You must run upgrade wizards and perform manual setup for a few functional areas (or modules), where new implementations in the latest version of Agresso prevent automatic upgrading. Modules not in use can be ignored.
4. Finally, you finish the upgrade by cleaning up browser templates and remove tables not longer in use.

Overview diagram

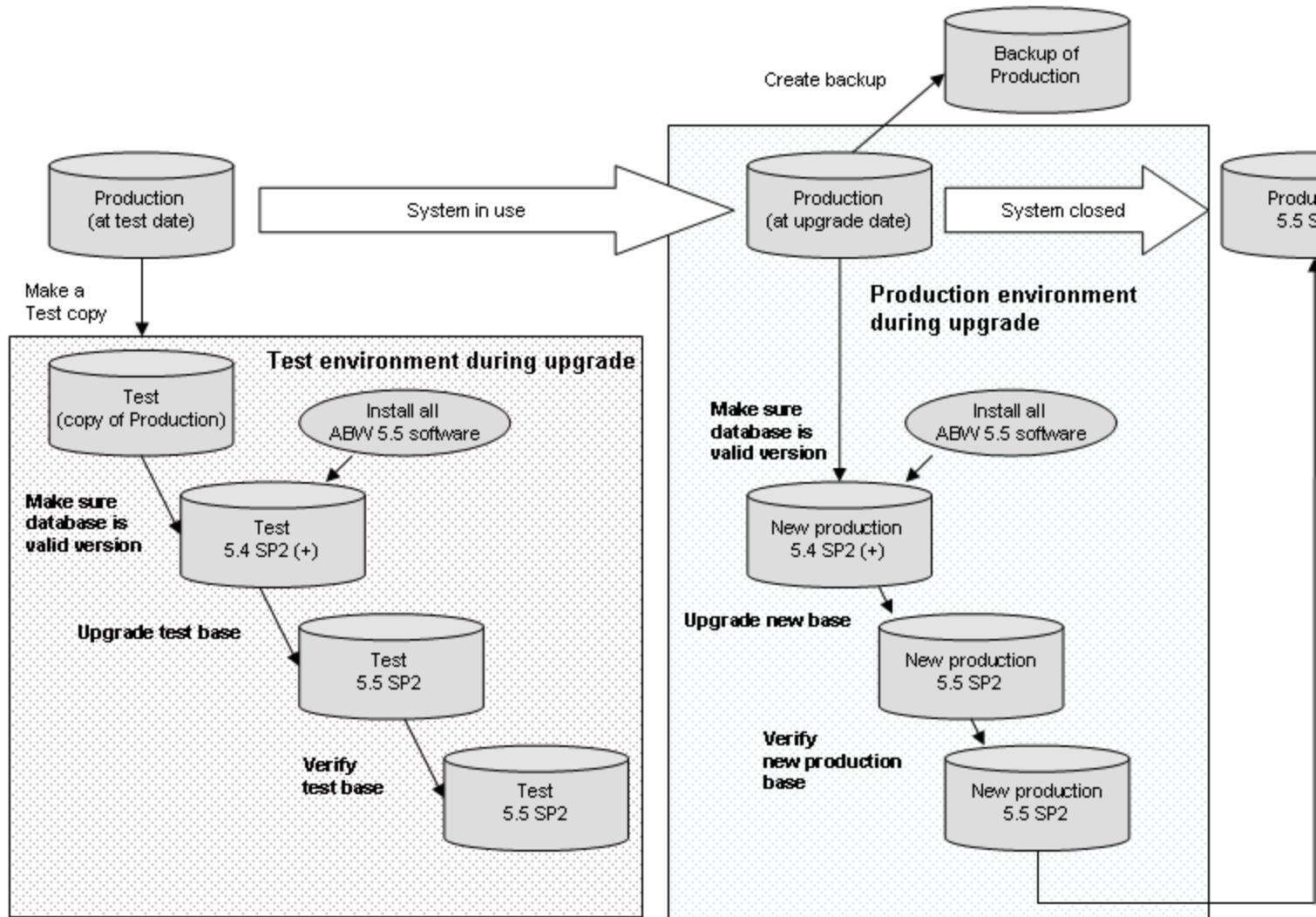
The diagram below summarises all tasks required for a complete upgrade - intended for the Production environment:



Upgrade Process Overview

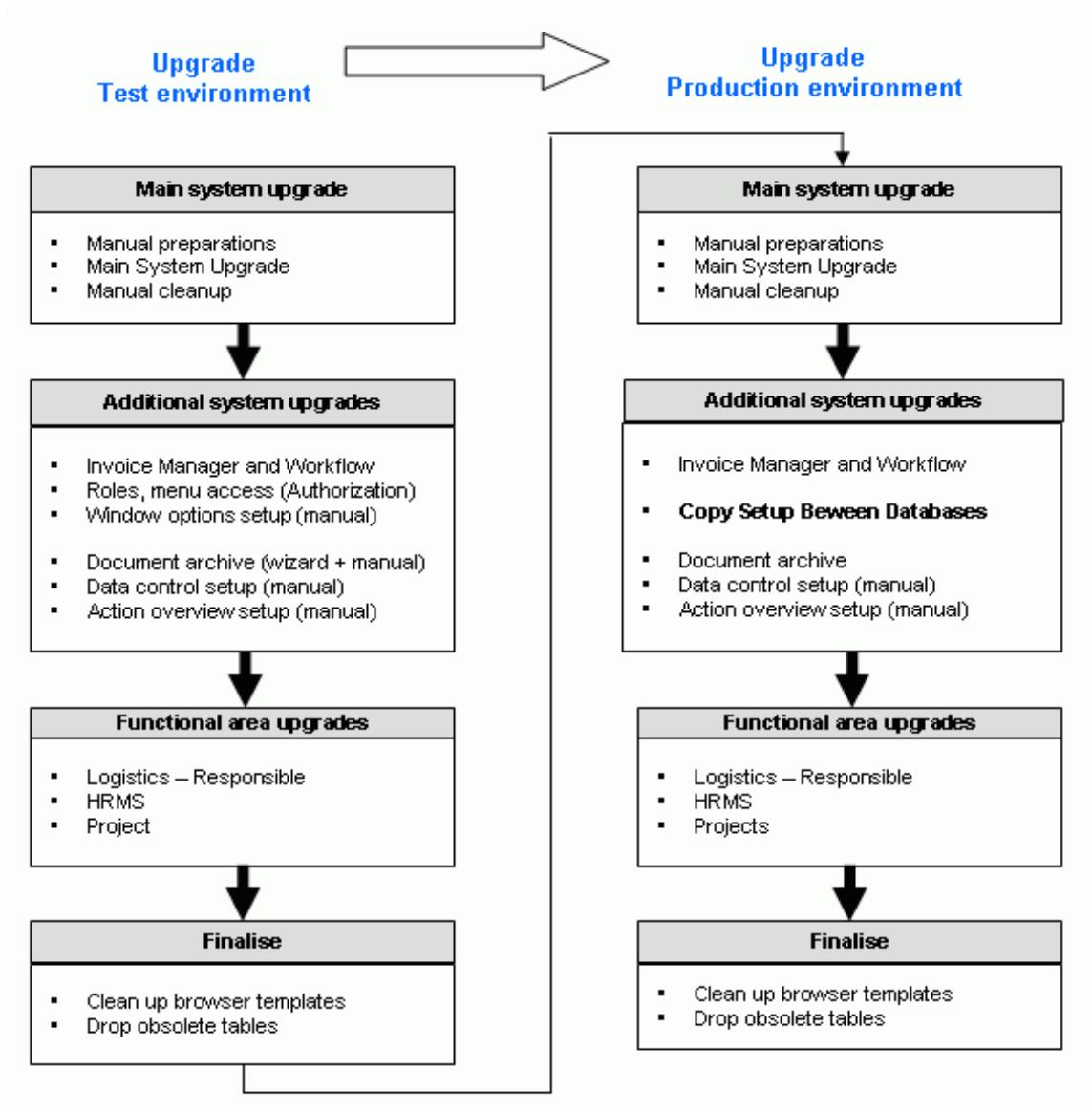
Overall upgrade process

The diagram below outlines the complete upgrade process, where you first upgrade a Test environment before you start with the Production system:



Main tasks

When we reduce the scope to the upgrade of a 5.4 SP2(+) system, the main tasks are as follows:



Prepare the Upgrade

Install UNIT4 Agresso software

When your Test database is in the correct 5.4 version, you should install all UNIT4 Agresso software.

Verify disk space

Available disk space needed for the upgrade is approximately the size of the largest table + 10%

Note: Expanding the maximum number of characters allowed in a field does not have any immediate impact on the database size. The expanded columns were previously of data type char, while they now are redefined to varchar.

Create data source and initialise Business Server environment

Use the **Agresso Management Console** to create a new data source connected to the old database.

When the connection is up and working, you must also initialise the Business Server environment (select the **Business Server** node in AMC and then **Initialise Business Server**). You do not need a database connection to initialize the Business Server Environment in AMC.

Rename or delete amendment tables

Logging

If amendment logging is turned on for an Agresso table, an amendment table (or shadow table) are created and continuously updated with all table changes.

Note: During upgrade, all amendment logging will be turned off (by the wizard), and switched on again when the convert script is complete.

Table for amendment tables

All amendment tables are defined in the table `aagamendlog`.

Naming standard: An Agresso table name is constructed from the structure `a<module><identification>` (example: `a cr client = acrc-client`) while an amendment table is extended with the letters `shd` between `<module>` and `<identification>`.

If you turn on amendment logging for `acrclient`, the amendment table `acrshdclient` will be created and added to `aagamendlog`.

Old amendment tables

During previous releases, the amendment tables have not been upgraded. When upgrading to 5.7.0, these tables will be checked, which may lead to a large number of errors.

Recommendation

To avoid problems with existing amendment tables, we recommend that you rename (or delete) all amendment tables before you start the upgrade process. Thereby, the upgrade wizard will not find them, and they will consequently not generate any errors.

If you need the amendment tables for historical reasons, they will always be available in your database copy.

To remember

Custom object definitions

The main system upgrade wizard will initially check the database for any custom objects, and write a detailed report containing all the old definitions (scripts). During the upgrade process, all custom objects will be removed.

The generated report, named `user_def_obj.txt`, will be saved to the default log directory. You will need this report, as well as a detailed knowledge of the new table definitions (see Table changes in latest version) to restore previous functionality in the latest version of the UNIT4 Agresso database.

The *custom objects* that will be removed, are:

- triggers and indexes,
- procedures,
- shadow tables,
- user defined views.

Run the Upgrade Wizard

The upgrade wizard

You find the upgrade wizard at the following location:

.\Agresso 5.7.0\DatabaseScript\DbUpgrade\UpgradeWizards.exe

Upgrade scripts and log files

Logging and error handling

The **Log File** will, when the script has run, contain a description of what happened. Each statement are listed, along with status after execution.

Note: You should - as a general rule - always look at the log file when the script is finished. If any errors occurred, they must be corrected, and the script must be run again - *before* you continue with the next upgrade step.

When we describe the actions for each step below, we *do not* ask you to view the log files. We take it for granted that you will do so!

Logging details

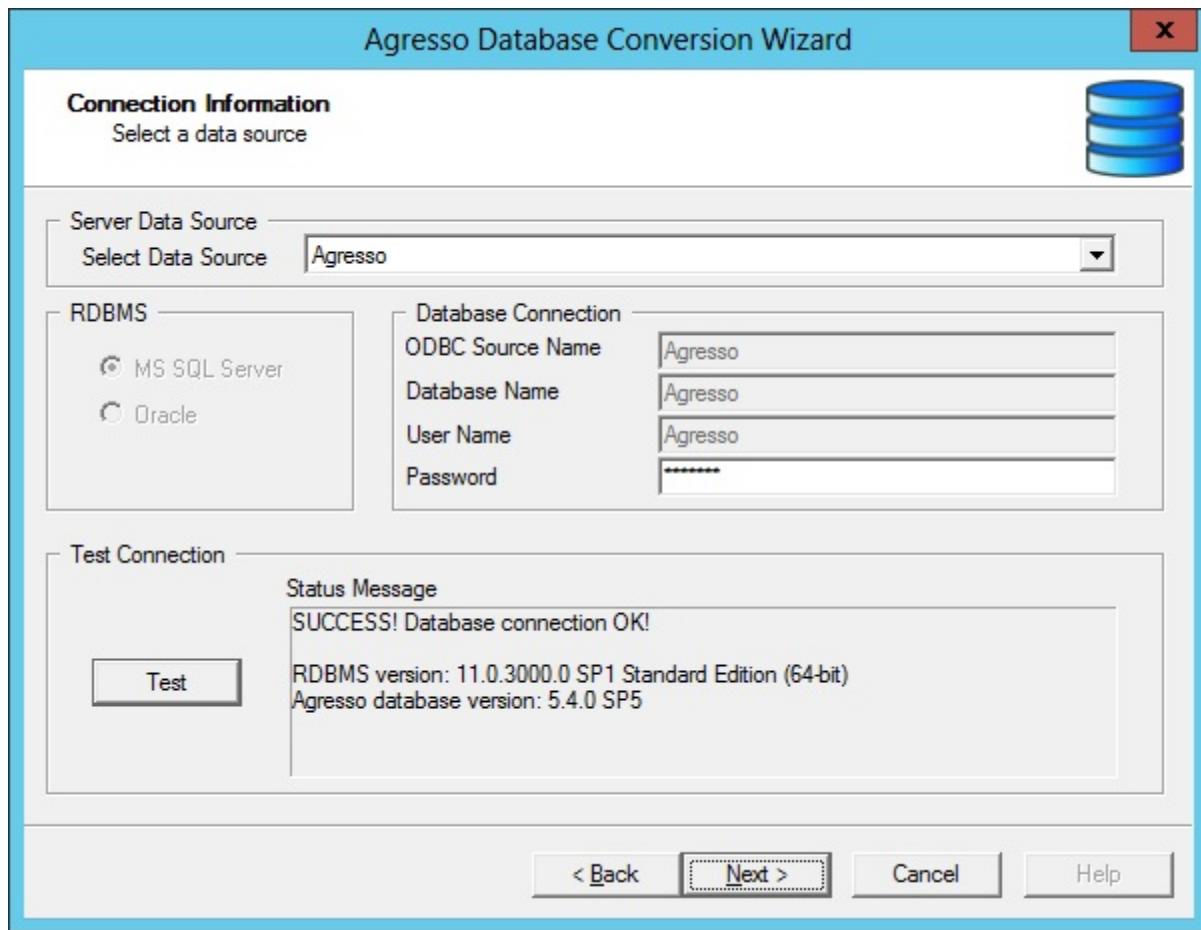
You can define the level of details to be listed in the log file by using the **Advanced** button available in the various wizard dialogs.

Run the upgrade wizard

1. When the upgrade wizard is up and running, select **Convert from 5.4**. Then select Step 1 [Main Upgrade Wizard](#). The **Connection Information** dialog will be displayed.



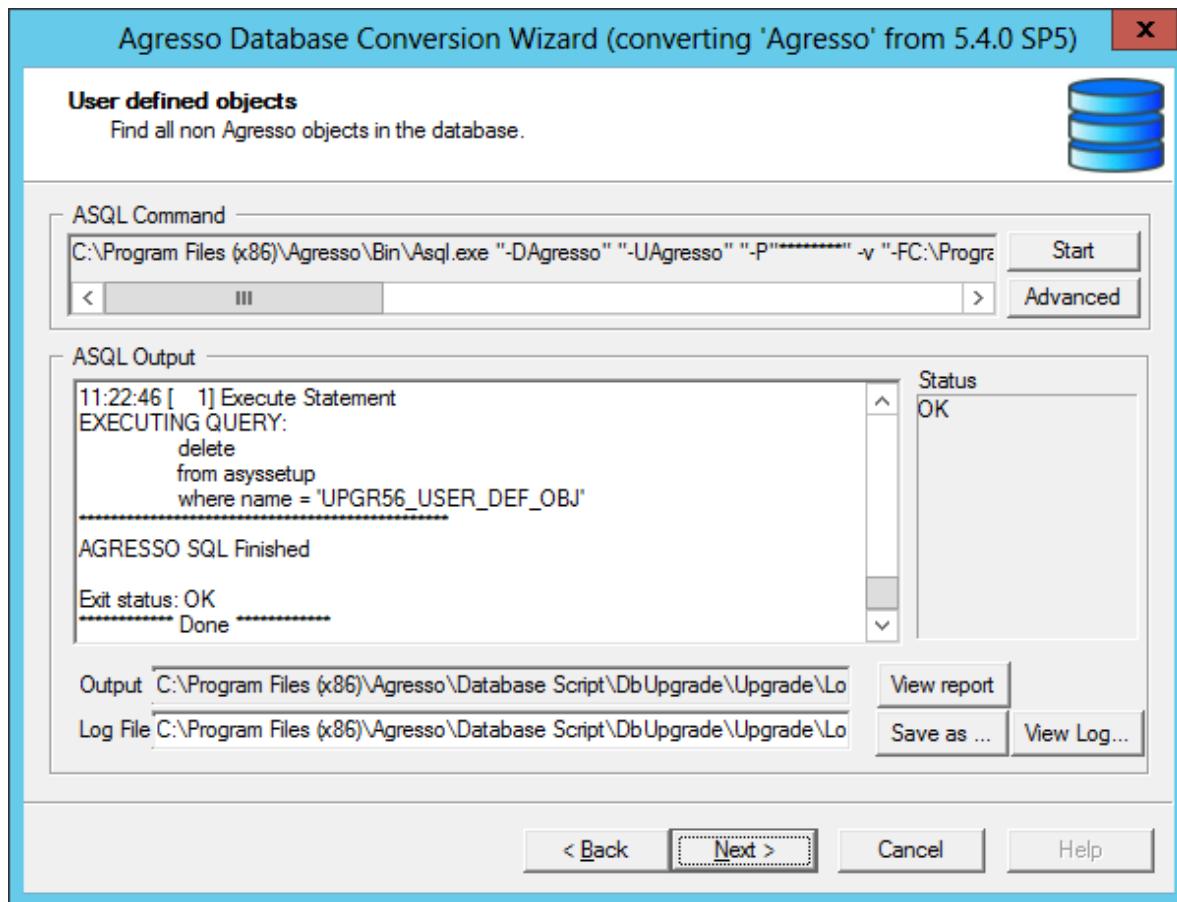
Upgrade Wizard - Connection Information



2. Select the data source, enter password, and click **Next**.

Note: If you get an error message, just accept the proposed action - and make a note of it! The error is not significant for the upgrade, but it will be when running Agresso later.

Upgrade Wizard - User defined objects



More step 3 details

Note: The report containing the custom object definitions are shown in the **ASQL Output** field. You will need this later.

Important: Do not continue if any errors are found in the database. These may be:

- **Null values** - these must be fixed before you continue. Use copyms/copyora to copy the table with the NULL values out, and then into the database again.
- **Duplicates**. Do not continue with the upgrade if there are duplicates in any of the tables
 - [aaguser](#),
 - [acruserinfo](#) or
 - [aaguserfunc](#).

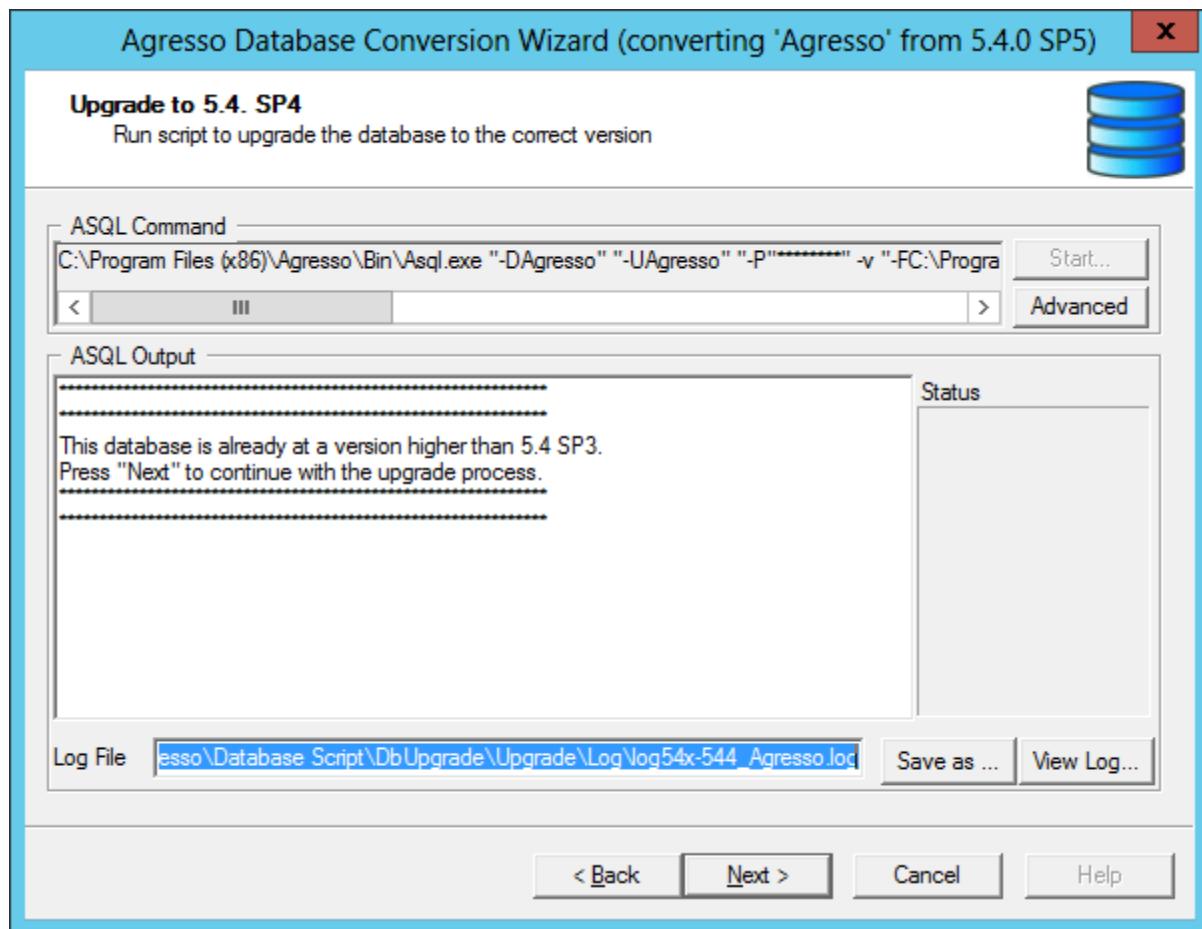
You must find the duplicates and remove them (or rename them).

Note: You must re-run the User defined objects step until there are no errors.

3. Click **Start** to run the script.

Important: You must correct all reported errors and make sure that everything is correct (re-run the script - until no errors are reported), before you can proceed to the next step.

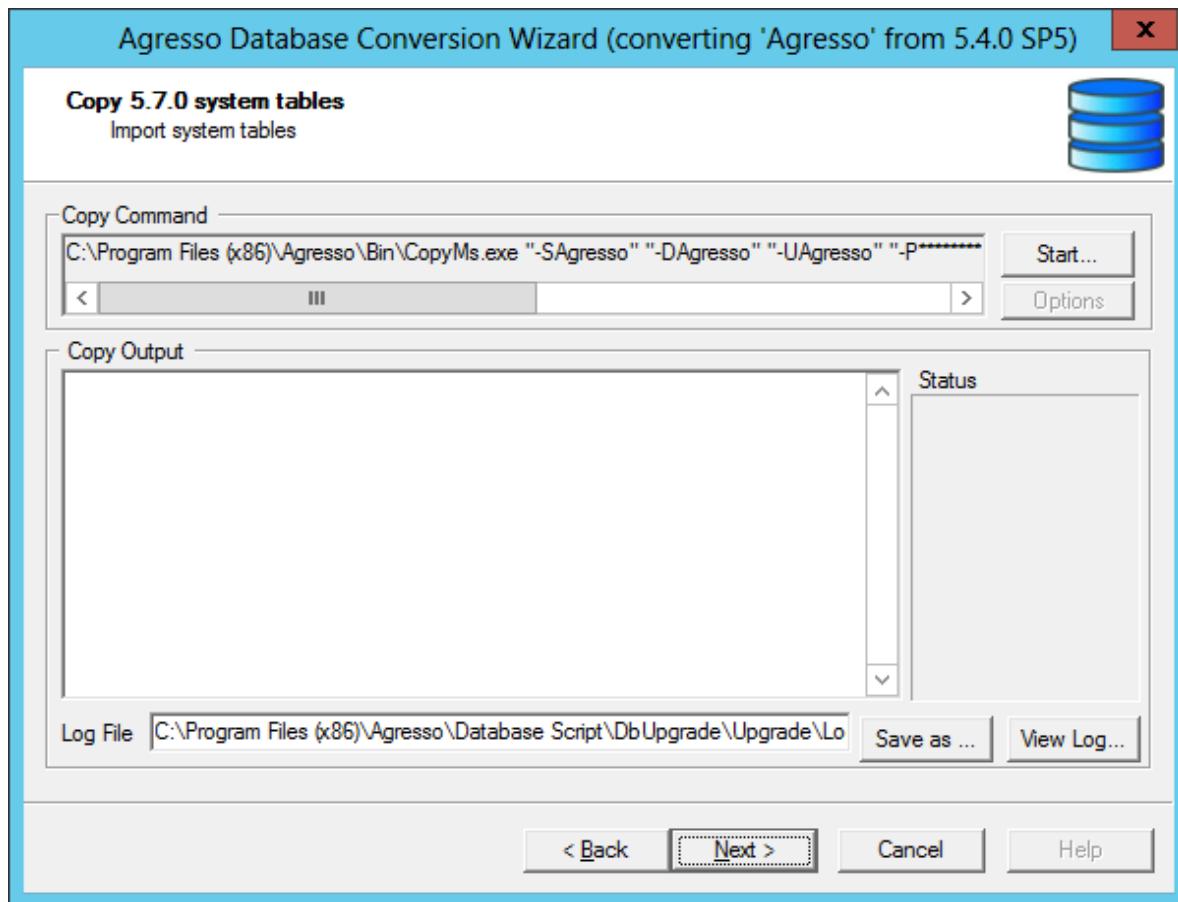
 Upgrade Wizard - Upgrade to 5.4 SP4



4. If your database needs upgrading, click **Start**. Click **Next >** to continue.

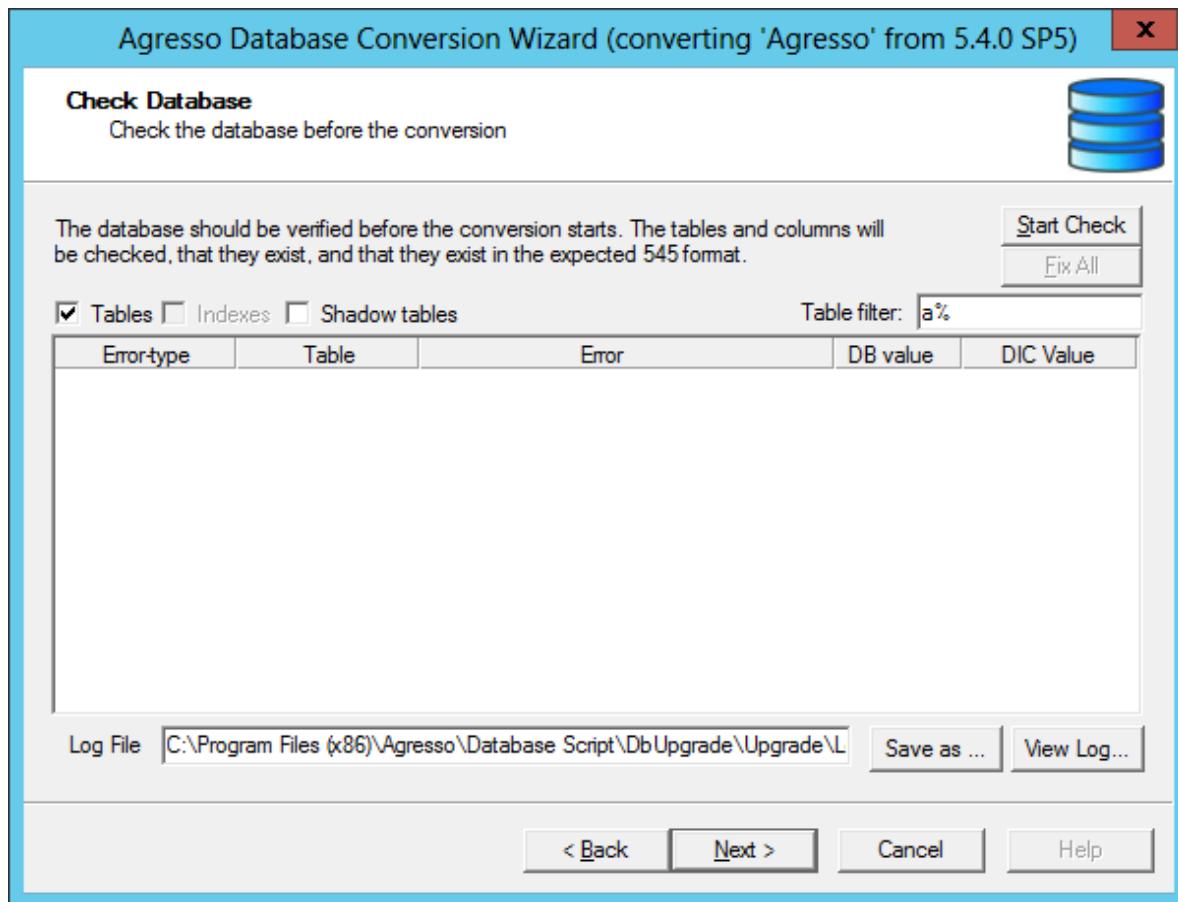
You are ready to restore the Agresso system tables, some awf* tables, and dictionary tables.

 Upgrade Wizard - Copy in System Tables



5. Click **Start** to run the script, then **Next** to continue.

Upgrade Wizard - Check database

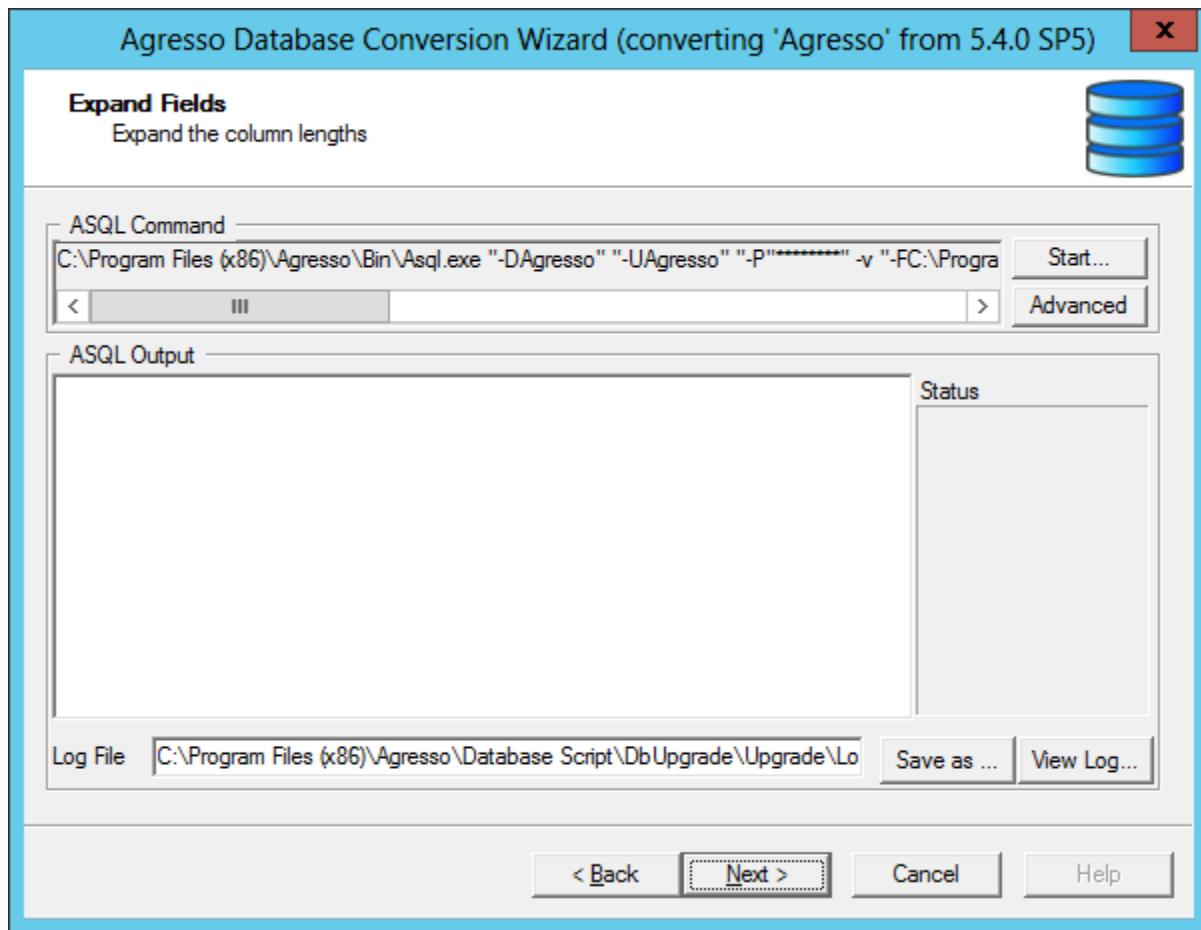


This step verifies that the database is in the expected 5.4 format. This check might take some time.

6. Click **Start Check** to check the database. Click **Next >** to continue when detected issues are verified or solved.

Next, the table columns will be expanded:

[Upgrade Wizard - Expand Fields](#)



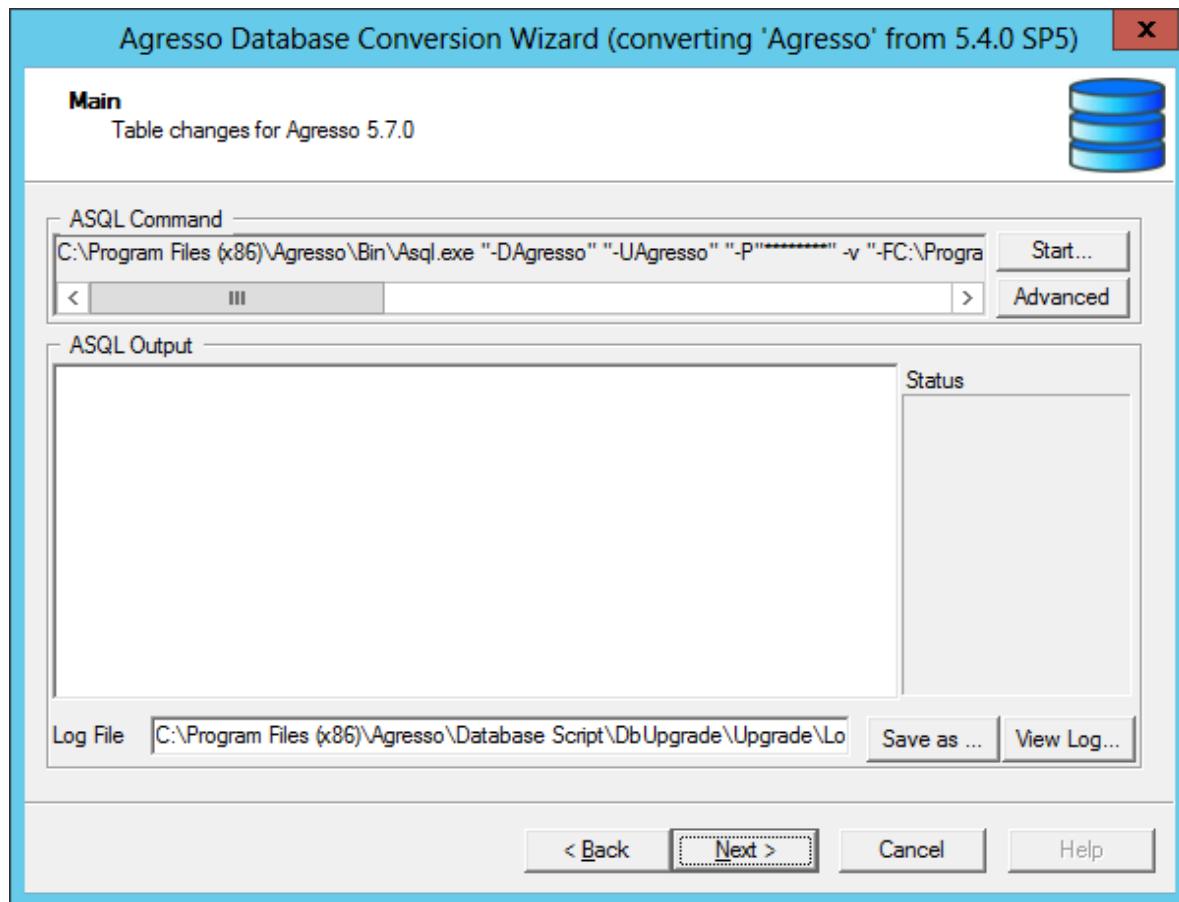
7. Click **Start** to run the script, then (when the script is finished) click **Next** to continue.

Note: This step might take some time.

Duplicates should be removed: If duplicates are found, it is recommended that the duplicates are removed before you continue. You do not have to rerun this step after the duplicates are removed.

You can now introduce the main changes in the database structure:

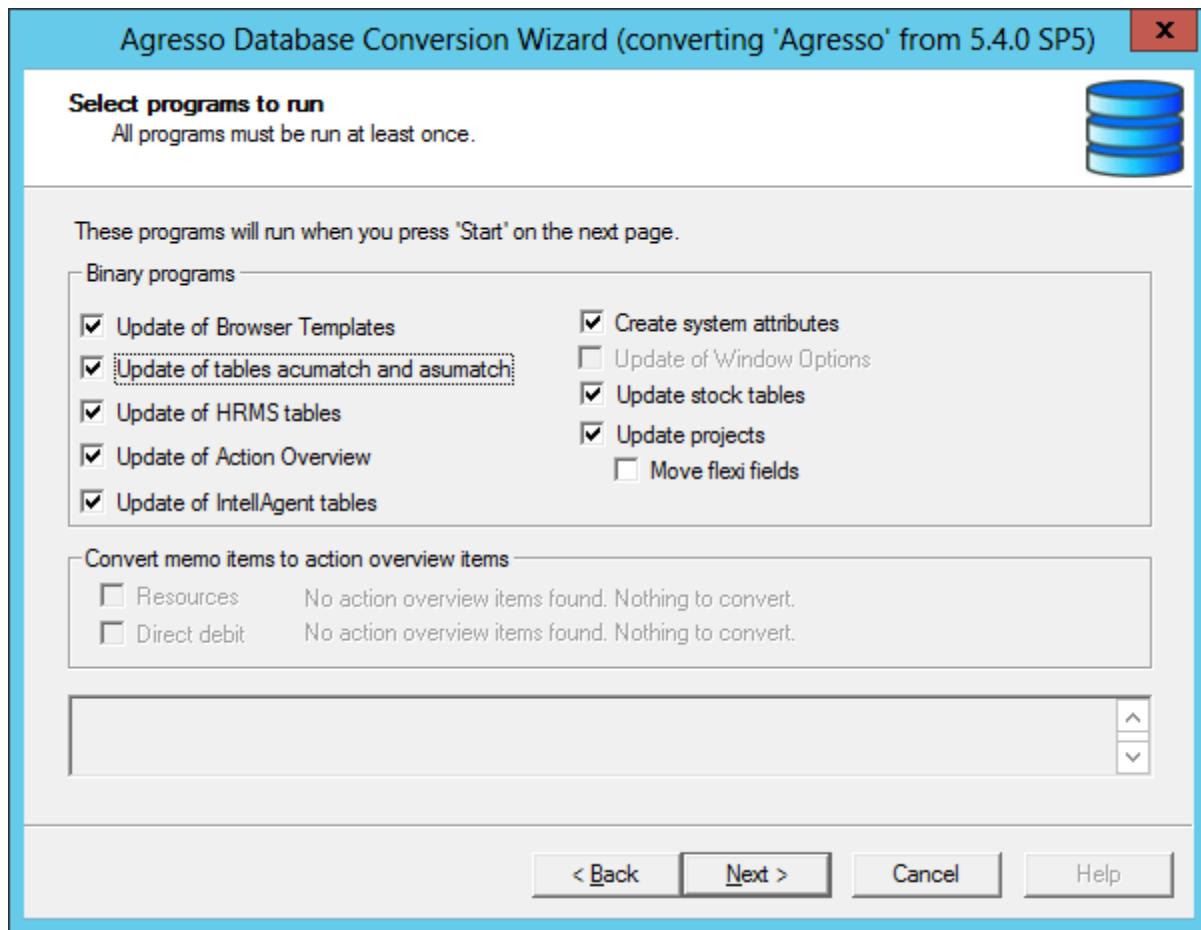
Upgrade Wizard - Main - Table changes for Agresso



8. Click **Start** to run the script, then (when the script is finished) click **Next** to continue.

The final changes to the database structure requires that you run some additional programs.

- Upgrade Wizard- Run binary programs

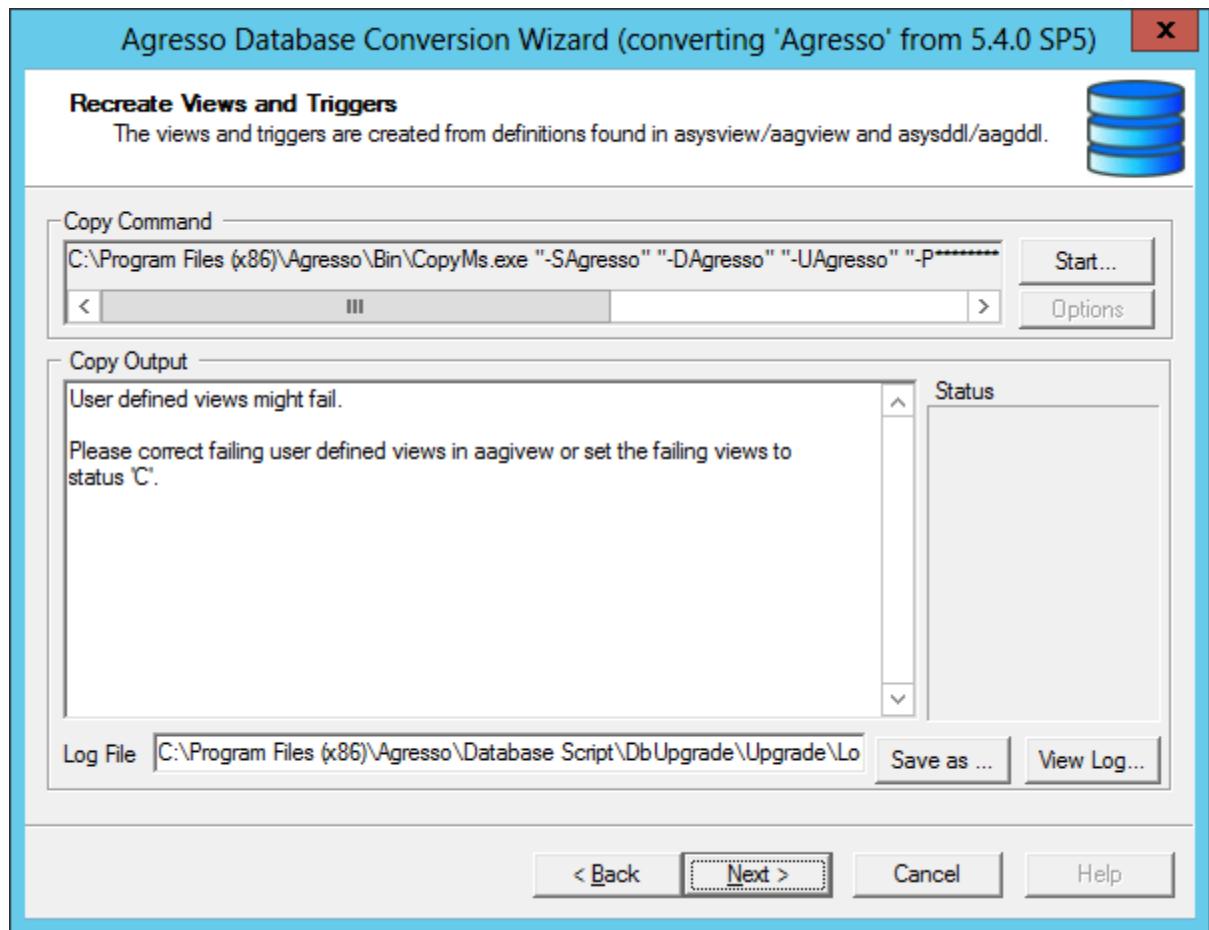


9. Select all programs listed and click **Next**.

Note: All programs must be run , but not necessarily at the same time. It will not do any harm if they are run more than once.

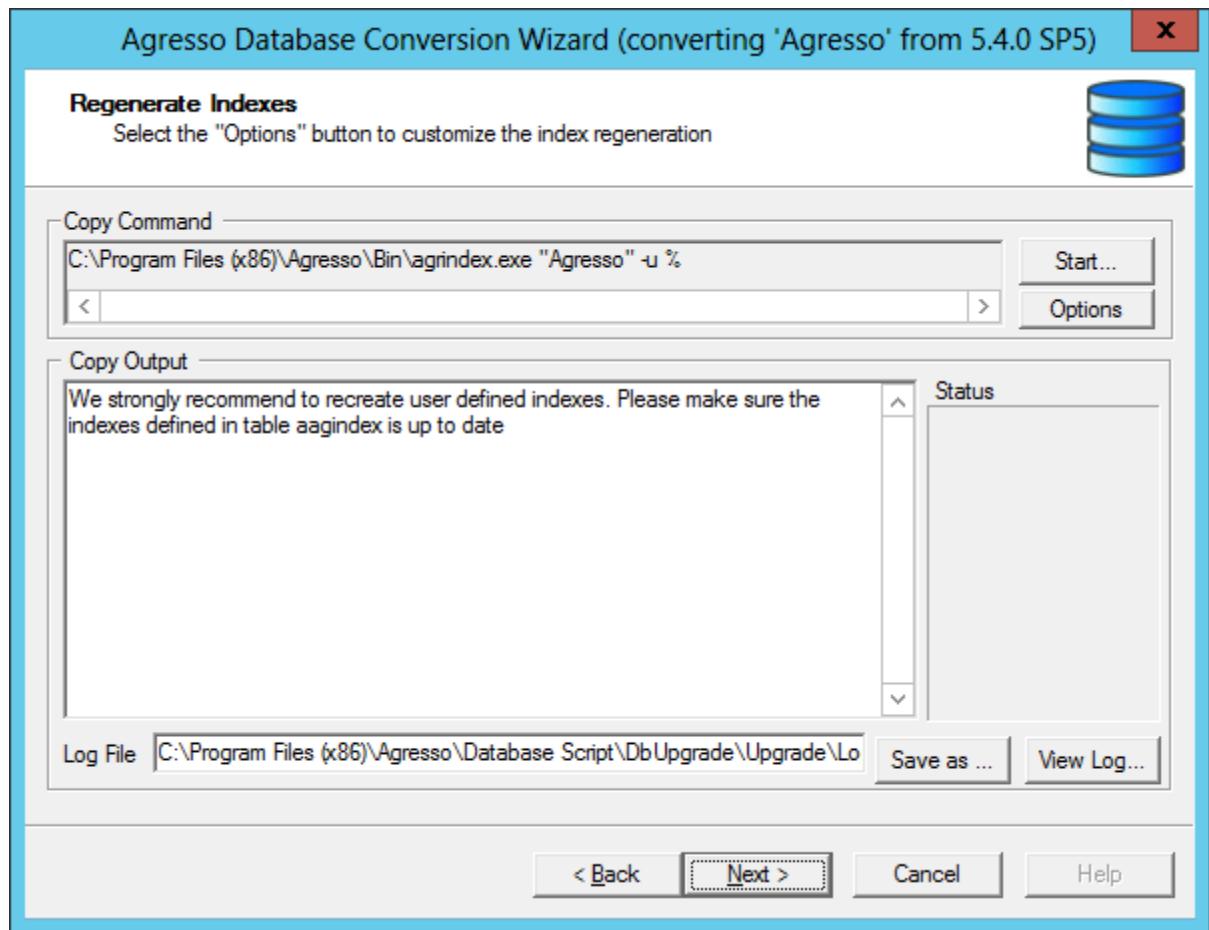
10. Click **Start** to run the selected upgrade programs, and then **Next** when the programs are completed.

 Upgrade Wizard - Recreate Views and Triggers

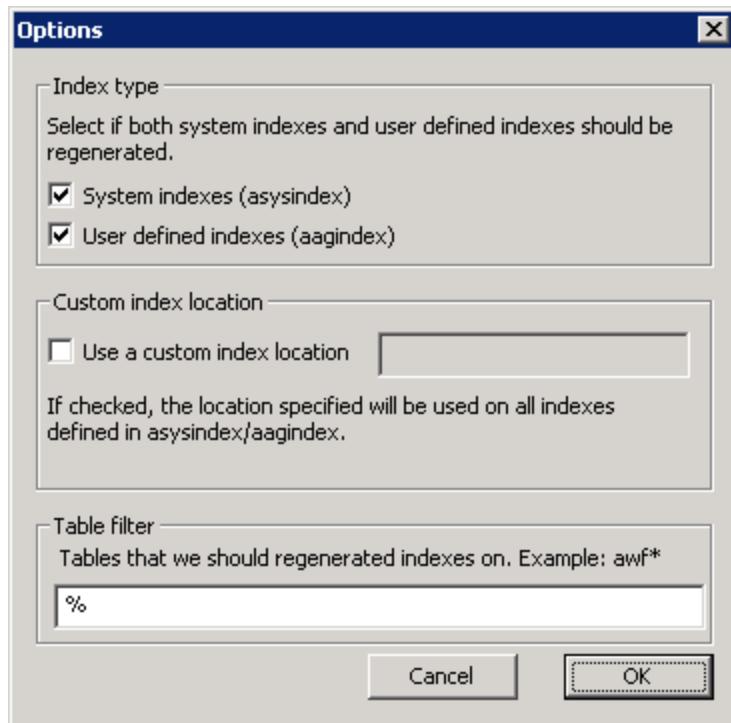


- 1.1.** Click **Start** and **Next** in the next windows, in order to restore system views and indexes. This will bring you to the final step.

Upgrade Wizard - Regenerate Indexes



12. Click the **Options** button to open the **Options** dialog:



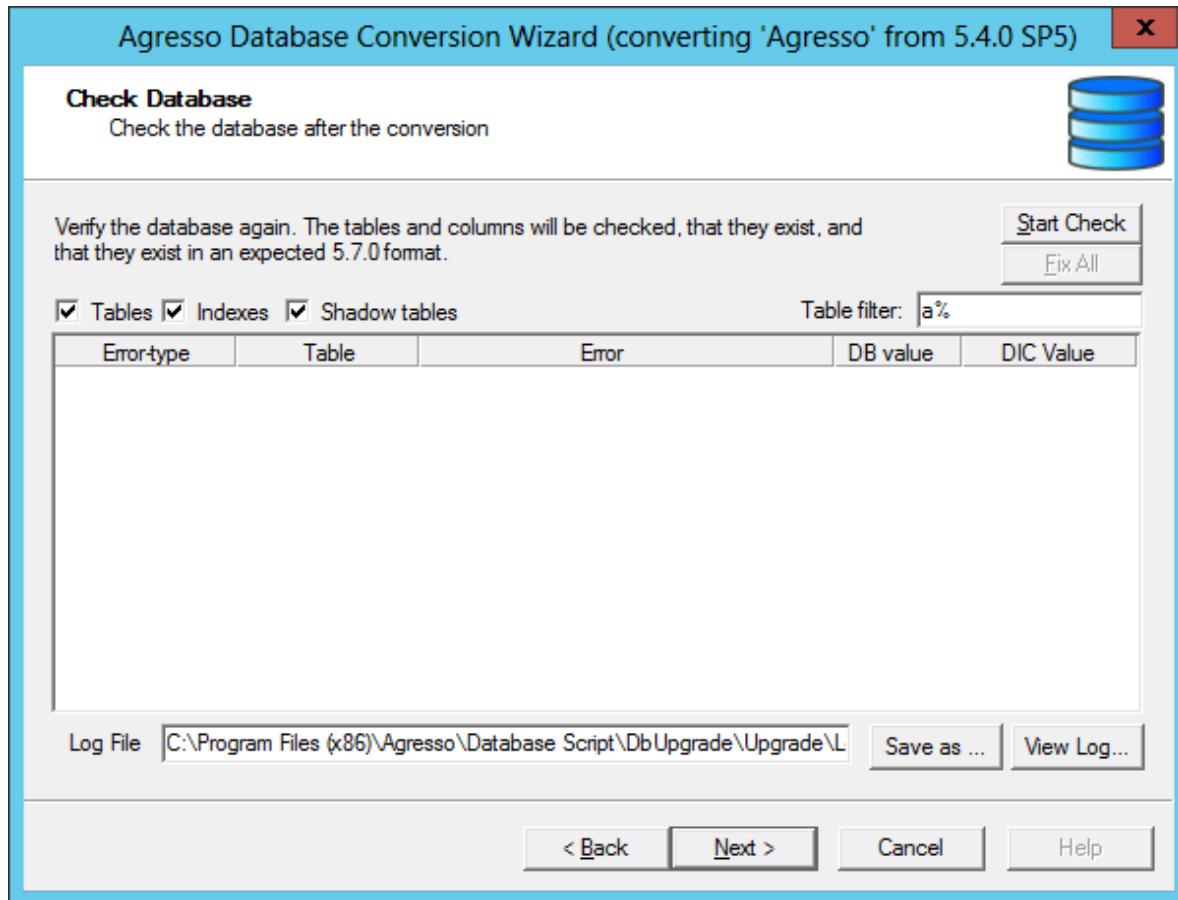
A note about indexes

User defined indexes (stored in [aagindex](#)) are recreated by default. Due to table changes, you should also change these, to avoid reduced performance. For details. see [AgrIndex](#).

13. Select options and click **OK**. Then click **Next** to continue with next step (Check Database).

This check might take some time.

 Upgrade Wizard - Check Database



14. Click **Start Check**.

This check compares the table structure, tables, columns, index with the format defined in Agresso's agrsys* tables.

If any differences occur, double-click on the error line to correct it. Or try the Fix All button. The Fix All might take some if tables with a lot of data needs to be fixed.

It is important to have the database with all the tables, columns and indexes in an expected format.

If there are tables with duplicates, clean up the duplicates and regenerate the index. A missing index might be crucial for the performance.

The following differences can be ignored:

Columns in the table are not found in the dictionary.

Columns are longer than expected.

Complete the Main Upgrade

Prerequisites

The output file *user_def_obj.txt* will contain all necessary information for restoring user defined views and objects. It will also list all duplicate user Ids (Oracle only).

In order to restore previous, custom functionality, you will require *user_def_obj.txt*, but also a detailed description of the database structure of the latest version of Agresso. See relevant Table changes in the Appendix (Agresso Data Dictionary).

Tasks

To complete the main system upgrade, you should do the following:

1. Restore user defined objects and views
2. Register license

Restore user defined objects and views

A note about amendment logging

During the upgrade, all amendment logging are stopped.

Update tasks

The table below describes how custom objects can be upgraded:

Object type	Update tasks
Triggers and indexes	Use a database tool to change the columns and data types to fit to the new database structure.
Procedures	Use a database tool to re-define the procedures according to the new database structure.
Amendment logging (shadow tables)	Amendment logging triggers needs to be recreated after the upgrade. Use the Smart client to open the AG30 Activation of logging server from System Administration Data Control Amendment logging , and regenerate active logging triggers.
User defined views	Use the Smart client to open AG17 Database view definition . Zoom into each query, make the necessary changes, and remember to save the changes to have the views recreated.

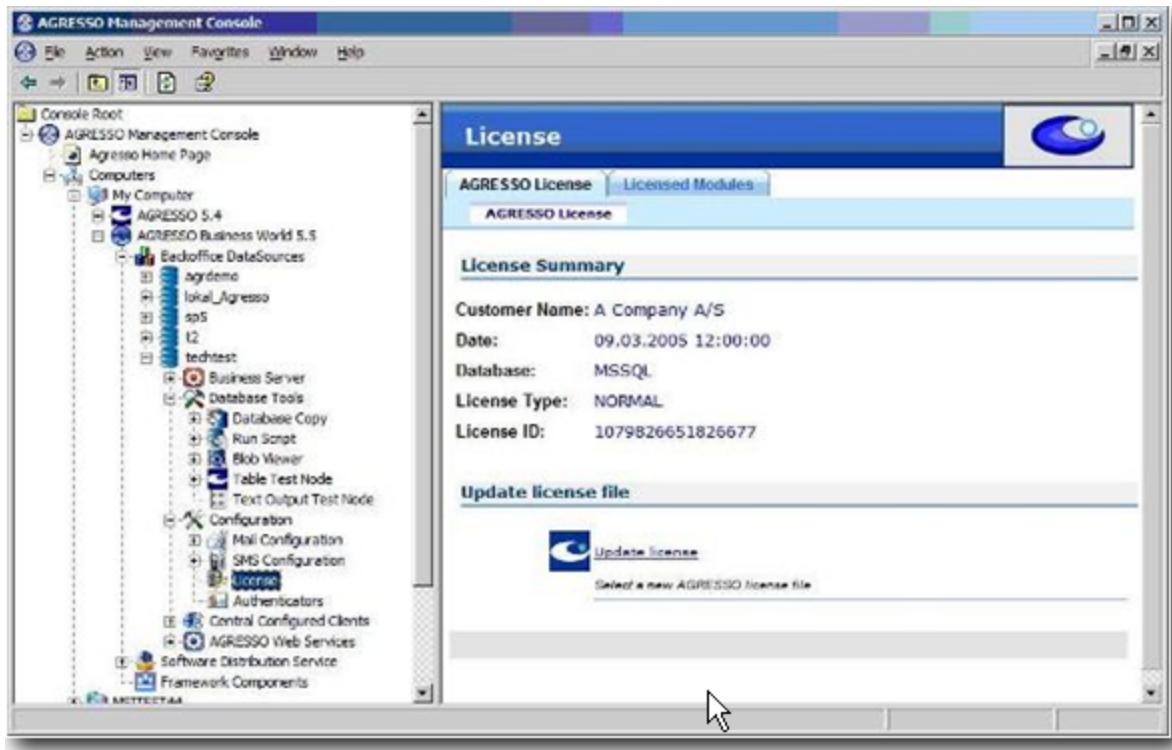
Register license

View licence

The license node in AMC allows you to add a new Agresso license to the database. It also enables you to view the licensed modules and the number of users registered for each module.

If you have a demo license installed, this node will inform you if the license will soon expire, or if it has already expired.

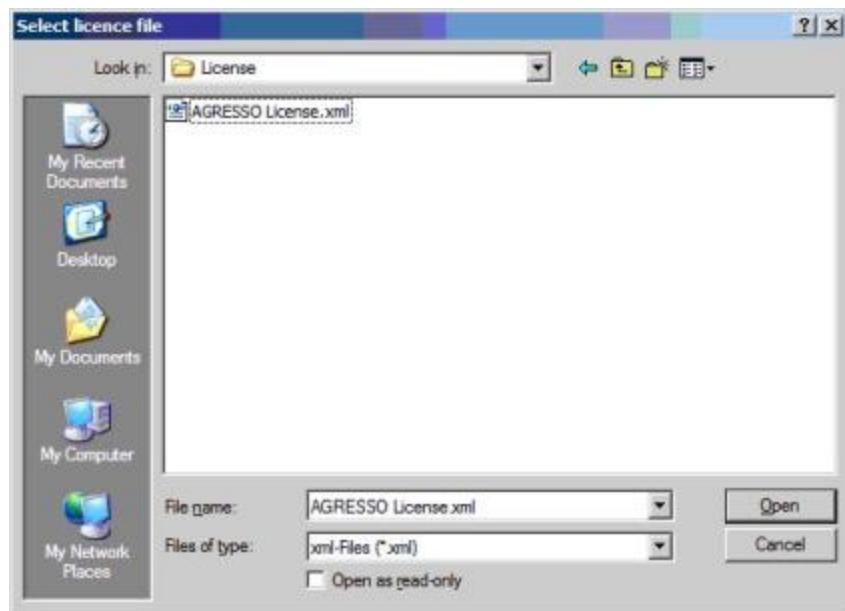
 [View License](#)



Add License

To add a license you can either right-click on the license node and select **Add License** from the **All tasks** context menu. Or you can click on the [Add License](#) or [Update License](#) link found in the right pane of the management console. Then you will be prompted to select a license file.

Add License



You can also add a license using the Agresso Client Configuration tool.

Note: You need to enter a *new license* when upgrading from 5.4. to make all parts of Agresso work. If required, contact your local Agresso support office.

Login info

Your are now ready to test your UNIT4 Agresso application with the following credentials (Note: `upgr55` is correct name after upgrading):

Username:	upgr55
Client:	<your client>
Password:	upgr55

Additional System Upgrades

You can now proceed with Additional system upgrades and setup.

Invoice Manager and Workflow

Agresso Workflow

New workflow solution

The workflow module (introduced in version 5.5 of ABW) replaces the Compello system. The new workflow module is based on an entirely new data model. Some Compello tables are kept to show historic workflow maps and are renamed to `ac<compello table name>`.

Wizard parameters

The Workflow upgrade wizard converts the data in the Compello tables to the new data model and defines processes in the new workflow system to match the existing set up in Compello.

Note: The wizard also makes it possible to upgrade rules from Compello. This is, however, NOT recommended, as the new workflow solution allows you to create far more advanced and generic rules than before.

Parameters

During the upgrade, you will be prompted to fill in some necessary parameters (see 3. below). These are explained as follows:

Parameter	Default value	Description
Client	*	The Agresso client to upgrade. Wildcards can be used (for example: *)..
Run as user		The user that will run the upgrade wizard.
Date From	19000101	Format: YYYYMMDD. Older document, transactions and images will be ignored.
Date To	20990101	Format: YYYYMMDD. Newer documents, transactions and images will be ignored.
Convert Invoice Man-	Off	Select if fixed registers from Compello shall be upgraded. If checked:

ager registers		<ul style="list-style-type: none"> Users – Users will be created and the workflow flag will be checked for them. Groups – will be converted to Roles Substitutes and supervisor definitions
Convert Invoice Manager rules	Off	<p>Only selectable if the Convert Invoice Manager registers is selected or the registers were upgraded in a previous run of the wizard.</p> <p>If selected, all Compello rules will be upgraded to the new rules tables.</p> <p>Note: This is not recommended (see The upgrade wizard above).</p>
Invoice Manager maps	Off	<p>If selected, historic transactions and Compello map data will still be available. The old maps can be viewed from the new map viewer.</p>
Create template workflow processes	Off	<p>If selected, all template workflow processes will be copied to the clients chosen for upgrade. The template processes are drafts and will need to be committed by the user chosen in Run as user (see above) to become active.</p> <p>Note: This will overwrite any current workflow processes for the clients being upgraded.</p>
Convert Invoice Manager archive	Off	<p>If selected, Compello images will be upgraded to the new solution, and UNC-path (below) will be enabled.</p>
UNC-path		<p>UNC path to locate existing Compello documents. NB! Must be UNC path!</p> <p>Example: <code>\compelloarchive\images</code></p>

Upgrade Invoice manager

Upgrade consequences

The result of the upgrade processes described below is that

- the Compello COM application is completely removed,
- users will have direct access to the file share.

Dangers: The Compello document location must be configured as a file share (UNC name) with full access for everybody using the document archive. The share can be hidden, but there is no way of stopping an advanced user from mapping the drive and accessing the documents.

In practice, no one needs more than read-only access to Compello documents, and the documents are scrambled to prevent casual browsing.

Protection: We have tried combine these requirements, and reduce the security risks, by creating a dummy Windows user, which Agresso will use to access the documents. The system parameters DS_NATIVEFILE_USER and DS_NATIVEFILE_PWD allows you to configure this user. The dummy user (and no other) needs full access to the file share. When you run the document archive system within Agresso, the rights of this dummy user are picked up. Any attempt to map the file share outside of Agresso will be denied.

New document archive and old images

With the new document archive, customers with historical Compello images can select one of the following options:

- Leave existing documents in the Compello folders as read-only, for historical access only
- Move the documents into the Agresso Document archive (and remove them from the Compello database).

Required upgrade

To be able to view Compello documents in Agresso, you must at least:

- 1.** Run the upgrade wizard to convert the database tables. (This can be done in stages by date etc.)
- 2.** Configure permissions for the Compello file share.
- 3.** Create a document type to access the Compello documents, for example Compello Invoices. This should belong to the GL Transaction key, and have the document system Compello selected.

This will change the pointers to the data only. The image files are not moved.

Complete migration

You may want to make a full migration for the following reasons:

- To have one document type only for all invoices, both old and new
- To be able to use the new Agresso document archive features

If you prefer a complete migration and remove the Compello archive, you can use the server process **DS01 Document copy** to move documents between document types. If the document types are based on different storage systems, as in this case, it automatically uses the drivers to do the conversion.

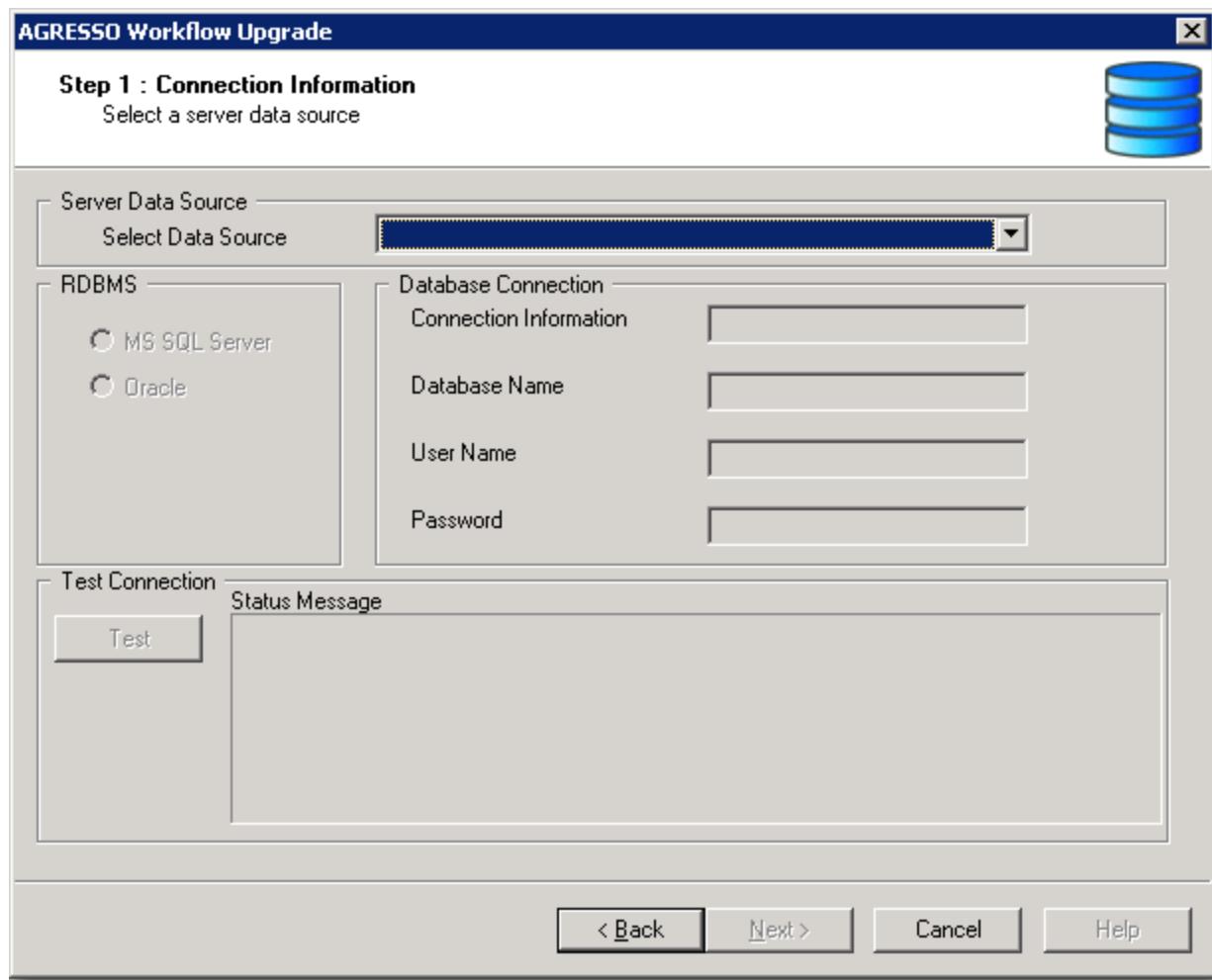
Run the wizard

Do as follows:

- 1.** Start the upgrade wizard and move to the Step 1 dialog:



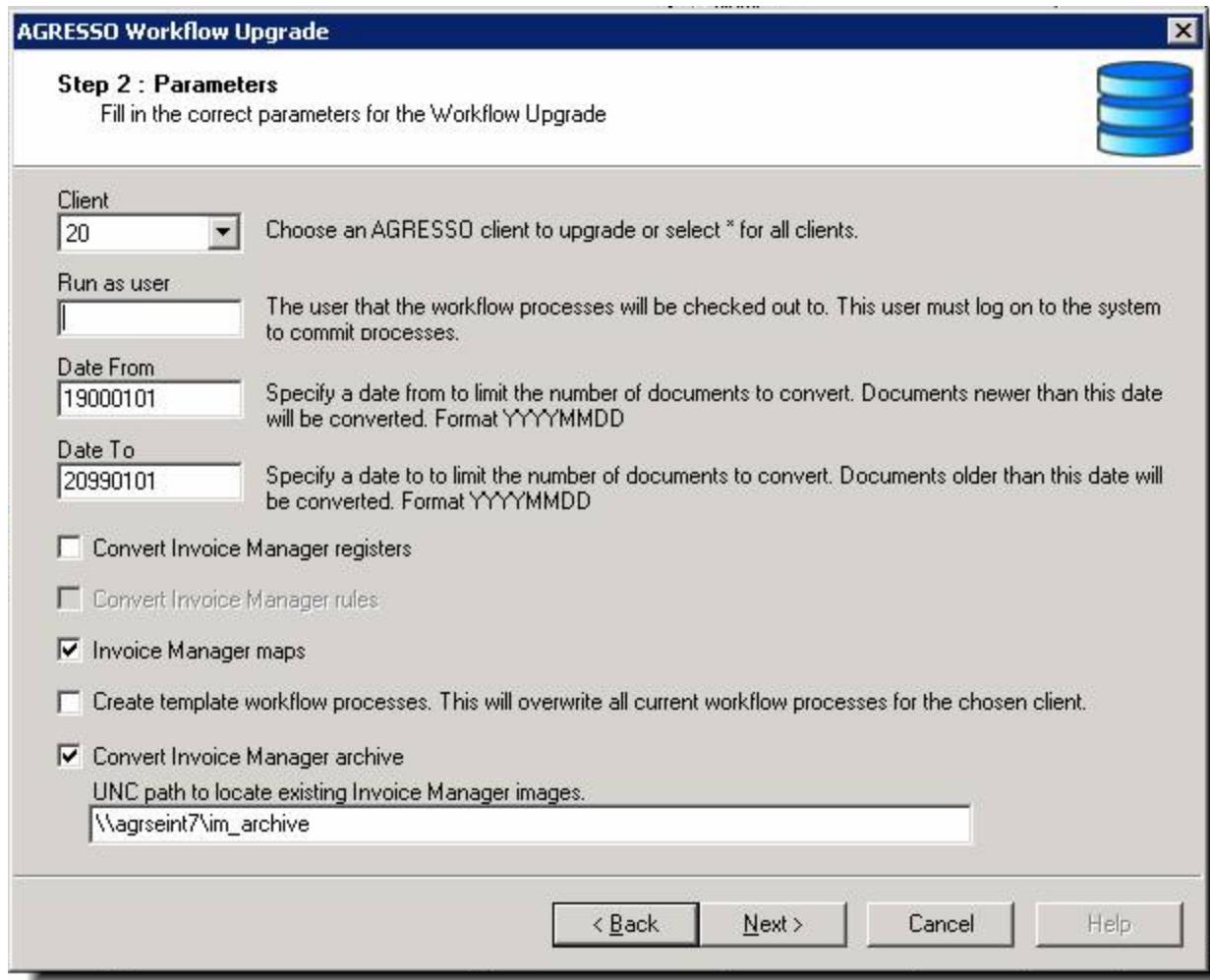
[Workflow Upgrade - Step 1](#)



2. Select correct data source from the drop-down list, fill in the correct password and click **Next**.

You are now prompted to set necessary parameter values.

Parameters - Step 2



3. Make sure the parameter settings are correct, and click **Next**.

You are ready to run the upgrade scripts.

4. Click **Start** to run the script and the **Next** to continue.

The (basic) upgrade is completed and you can close the wizard.

Document Archive Upgrade

New Document archive solution

The Agresso Document archive was completely rewritten with Agresso 5.5. The document archive is now an integrated part of a series of Agresso windows, and documents of any type (.doc, .pdf, .jpg etc.) can now be attached to most of the Agresso object's.

These profound changes require that all the previous documents must be converted.

Please refer to *Agresso 5.5 Release Notes, Document Archive* for details about technical and functional aspects of the new solution.

Invoice Manager Reference

See the topic [Workflow and Invoice Manager](#) for information on how to interface your old Compello images with the new Agresso document archive.

Running the upgrade wizard

Necessary preparations

During the upgrade, you will need to make a few selections in order to proceed (see 3. below). The parameters you must enter are explained in the following table:

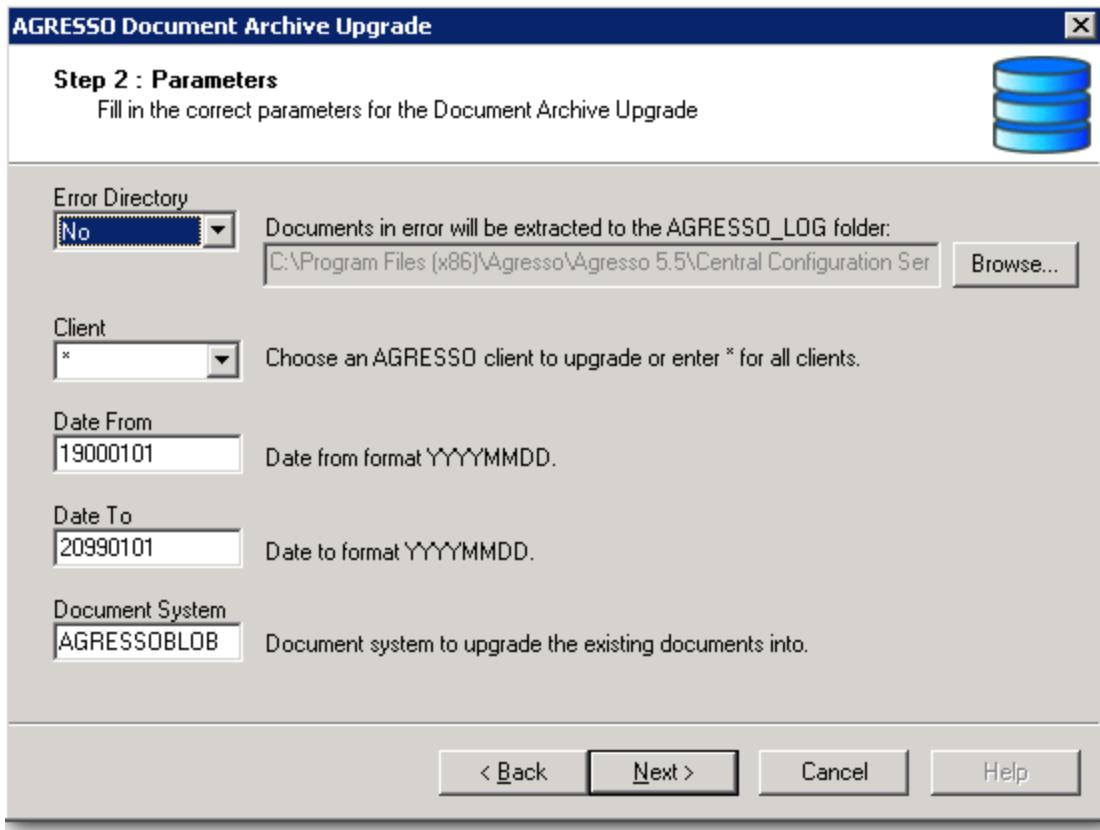
Field	Default value	Description
Error Directory	No	<p>Yes or No.</p> <p>If Yes, erroneous documents will be extracted to the server logging (AGRESSO_LOG) directory for manual fixing.</p> <p>If No erroneous documents will just be reported in the log (not extracted).</p>
Client	*	The Agresso client to upgrade. An asterisk (*) means all clients.
Date from	19000101	All documents registered at or after Date from will be included. Older documents will be ignored. Format: YYYYMMDD.
Date to	20990101	All documents registered before or at Date to will be included. Newer documents will be ignored. Format: YYYYMMDD.
Document system	AgressoBLOB	The new document archive can be connected to many document systems, including third party archives. You must choose a document system to upgrade the existing documents into. The default value is the new Agresso database archive.

Upgrade procedure

To upgrade to the new document archive solution, you will first start the Upgrade wizard. Continue as follows:

1. Double-click the [Document Archive](#) link in the Database Upgrade Wizards screen. After reading the introduction, click Next.
2. Enter connection information and click Next to prepare for step 2.

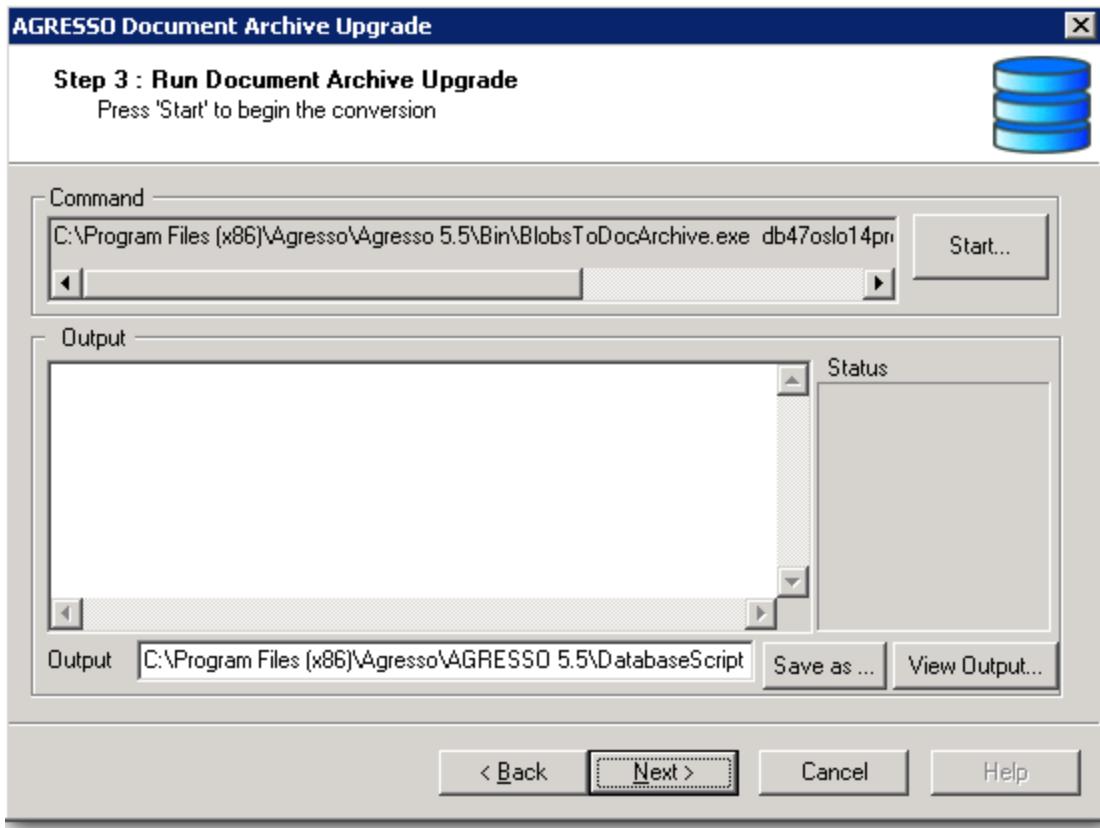




3. Fill in the required parameters and click **Next**.

You will now have to verify the path to the log file (output) made by the upgrade scripts.

 [Document Archive Upgrade - Step 3 #1](#)



4. Make sure that the output path is correct, and run the upgrade by clicking the **Start** button.

5. When the conversion is done, do as follows:

- a. Click **View Output** to study the conversion results
(we presume everything is in order)
- b. Click **Next** to bring up the final window.

6. Click **Finish**

Note: If necessary, check the log file once more and make sure everything is correct.

Manual Setup Tasks - Production

The following areas require manual setup:

- Data control - see Release Notes for Common and System Administration, ABW 5.5 and 5.5.1 combined.
- Management of open items and Action overview - see Release Notes for Action Overview, ABW 5.5 and 5.5.1 combined

Copy Setup Between Databases

General description

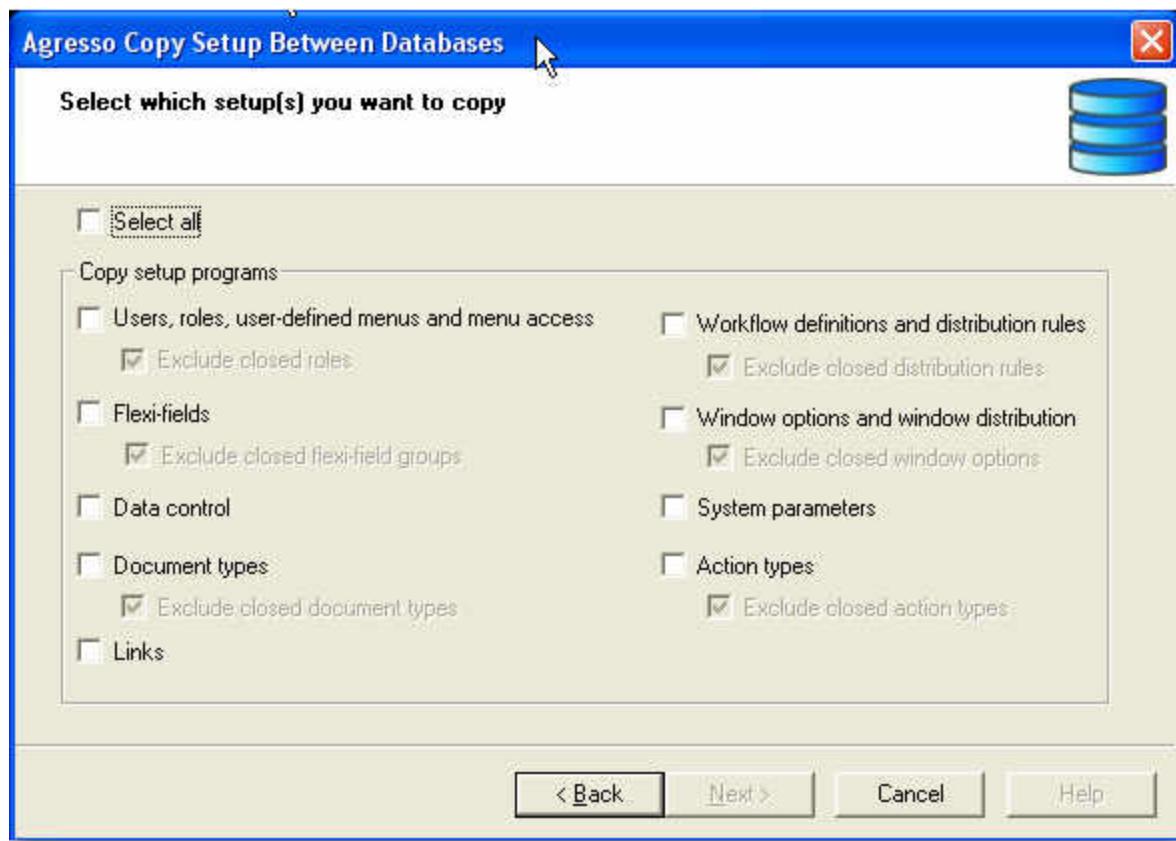
The **Copy Setup Between Databases** wizard copies the Agresso setup from one database to another. The following areas may be copied:

- Users, roles and menu access
- Workflow definitions and distribution rules
- Flexi-fields
- Window options and distribution of window options
- Data control
- System parameters
- Document types
- Action types

Note: Existing setup in the target database will be overwritten. See Dialog example below.

Dialog Example

The various options are presented in a single window:



Intended use

The wizard is created to facilitate transfer of a new and working configuration in the test environment, to the production environment.

Used in this context, it is important that all other updates are completed – on the test database – before you run the **Copy Setup Between Databases** wizard.

Run the wizard

Note: If you need more details before you run the wizard, see [Wizard details](#) and [Updated tables](#) below

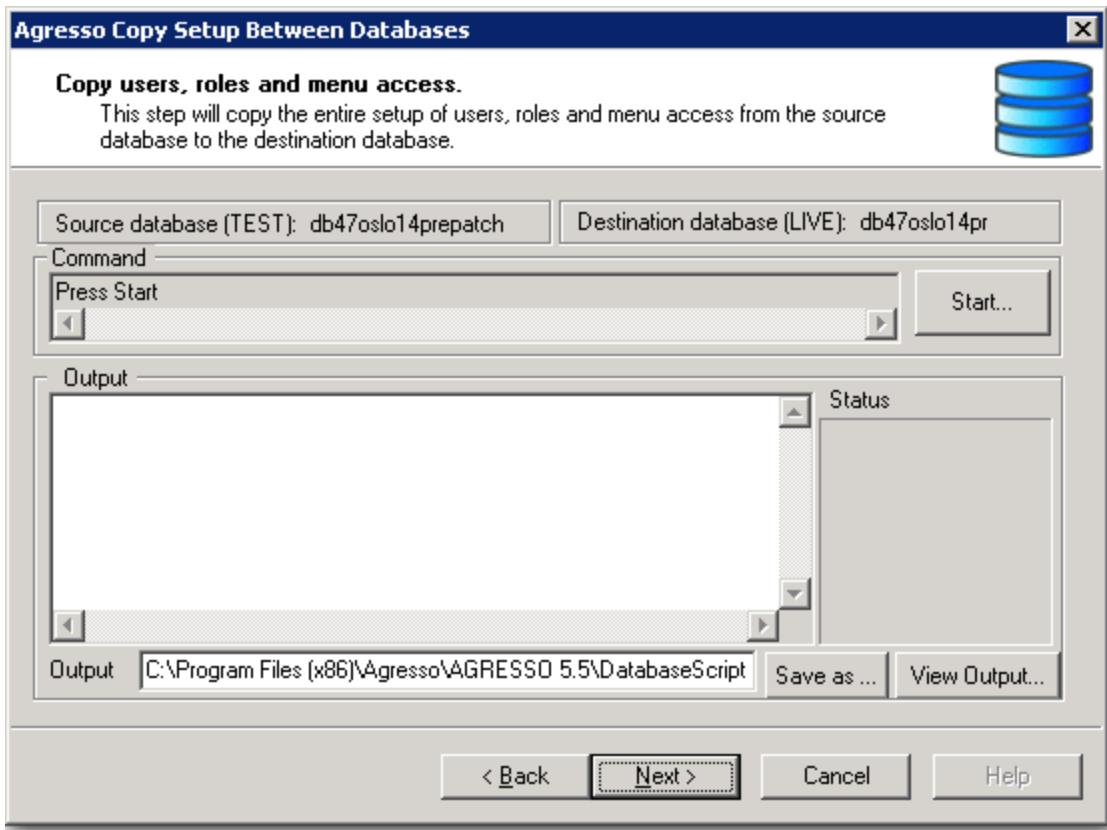
When the wizard is up and running, do as follows:

1. Enter connection information for the *Source* database and click **Next**.
2. Enter connection information for the *Destination* database and click **Next**.

You are presented with a few copy options. See Dialog example above.

3. Make your selections and click **Next**. The wizard is ready to start the Copy operation.

Example.



4. Click **Start**. You will need to confirm copying for each selected area.

Wizard details

The various copy programs (wizards) are described below:

The wizard ...	Will copy the following from source (test) database to target (production) database ...
Users, roles, user-defined menus and menu access	<ul style="list-style-type: none"> • New users (aaguser) with status Active • User information (addresses and passwords) connected to new users with status Active • All user detail information (aaguserdetail) • All roles • All menu access (aagaccess) granted to both users and roles • All user-defined menus (aagmenu) <p>You can exclude roles with status Closed from being copied to the destination database (default).</p>
Workflow definitions and distribution rules	<p>The entire setup of workflow and distribution rules. Existing setup will be replaced.</p> <p>You can exclude closed distribution (default).</p>

Flexi-fields	<p>The entire setup of Flexi-fields. Existing setup will be replaced.</p> <p>You can exclude closed Flexi-field groups (default).</p> <p>Note: No Flexi-field data will be copied!</p>
Window options and window distribution	<p>The setup of window options and distribution of window options. All existing setup will be replaced.</p> <p>You can exclude closed window options (default).</p>
Data control	<ul style="list-style-type: none"> The entire setup of data control. All existing setup will be replaced. All relations and relation values connected to attribute A11 &ndash; ROLEID All setup made in Data control management (CR49). <p>Note: The program also updates all attributes with data control activated in the <i>source</i> database.</p>
System parameters	<p>System parameters.</p> <p>All changed parameters will be listed in a report file. The path to this report file is specified in the program's log file.</p>
Document types	<p>The setup of Document archive. All existing setup will be replaced.</p> <p>Note: No documents will be copied!</p>
Action types	The setup of Action overview. All existing setup will be replaced.
Links	The setup of Links. All existing setup will be replaced.

Updated tables

Below we list the tables affected by the copy programs, and indicate the operations they are exposed to:

Users, roles, user-defined menus and menu access

The following tables are affected:

Table name	Data replaced	Rows inserted	Rows deleted
awfelemttype	X		
awfelemttypedet	X		
awfelemttypegrouping	X		
awfelemttypeitem	X		
awfelemttypemap	X		
awfprocaction	X		
awfprocdeadlines	X		

awfprocdelay	X		
awfprocdelaydet	X		
wawfprocelemtype	X		
awfprocess	X		
awfprocfunction	X		
awfprocrole	X		
awfprocsplit	X		
awfrule	X		
awfruledet	X		
awfrulegroup	X		
awfversion	X		
acrdiagramlayout	X		
awfalternate	X		
awfblmethods	X		
awfcolumns	X		
awfelemtypemenu	X		
awfmanstep	X		
awfprocrcd	X		
awfprocsin	X		
awfprocsrp		X	
awfproctin		X	
awfuserdetail	X		
aimblob		X	X

Flexi-fields

The following tables are affected:

Table name	Data replaced
aagpagesetupconnect	X
aagpagesetupdet	X
aagpagesetuphead	X

Data control

The following tables are affected:

Table name	Data replaced
aagparameter	X

Document types

The following tables are affected:

Table name	Data replaced	Rows inserted	Rows deleted
acraktionelemtyp	X		
acraktiontype	X		
agldimvalue		X	X
agldescription		X	X
acraktioncontact	X		
acraktionattval	X		
acracctionattvaldet	X		
awfelemtype	X		
awfelemtypepedet	X		
awfelemtypeitem	X		

Links

The following tables are affected:

Responsibles Upgrade

Changes in data structure

The new solution for users and roles has required a completely new implementation of the *Responsible* concept in the Logistics module. Previously, the *algresponsible* table hold information about responsible codes and the *resource_id* (not the *user_id*) that was linked to the code.

Agresso 5.5 introduced a solution where a responsible belongs to a certain role, and where role membership is based on the *user_id*.

Upgrade of orders created by Agresso 5.4x

The main purpose of this Responsibles Upgrade, is to convert active orders (created in the various logistics modules) in such a way that the responsible person still can be identified when the order is further processed.

Note

The upgrade will *not* convert the old responsible resources into new responsible *roles*.

Upgrade tasks

The main upgrade tasks are performed by the responsible upgrade wizard, and are described as follows:

Task	Description
------	-------------

Automatic code matching	Match resource_ids from the algresponsible table with the 5.7.0 users. If a user can be uniquely identified by a resource_id, the upgrade wizard will establish a new connection between the 5.4 responsible code and the 5.5 user id.
Manual matching	For all unmatched resource_ids, you are prompted to manually enter user_ids. There is no requirement that the new user really takes part in 5.5 responsible role. This task can be repeated several times, until all (old) resource_ids have a valid match.
Automatic order upgrade	The wizard will search through all relevant orders and convert all orders with a responsible codes where there has been a successful match. Log: The result will be written to the log, and can be viewed in the table <i>algprophead</i> .
Manual order update	For all the orders that were not converted, you will have to use the registration windows for the various order types, to set the correct responsible. Note: This requires that the various responsible roles are set up in the Responsible Setup window in the Smart client. See <i>Agresso 5.5 Release Notes, Logistics – Common logistics</i> .

Run the upgrade wizard

When the **Responsibles Upgrade wizard** is up and running, navigate to the Step 1 window, and do as follows:

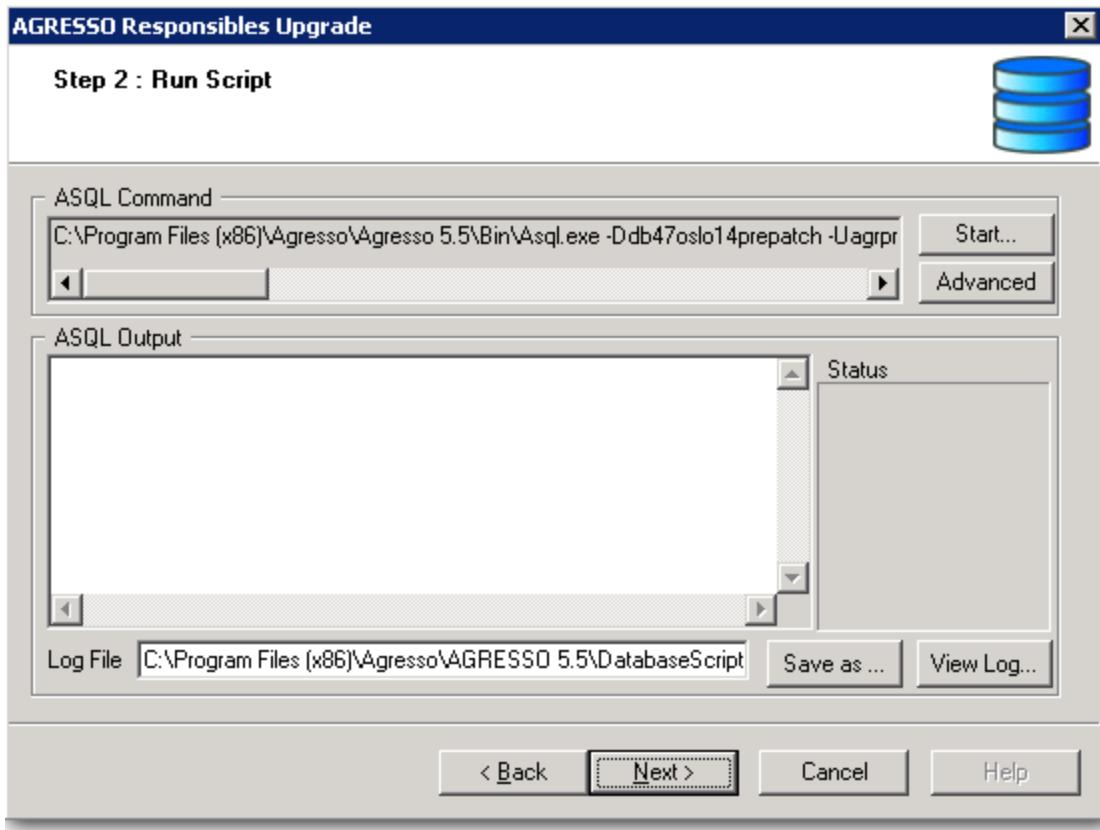
1. Select correct data source from the drop-down list, fill in the correct password and click **Next**.

You are now ready to run the upgrade scripts.

Note: If you already have run the upgrade scripts once, and in addition performed some manual matching (3. below), a re-run of the wizard will remove all manual updates!



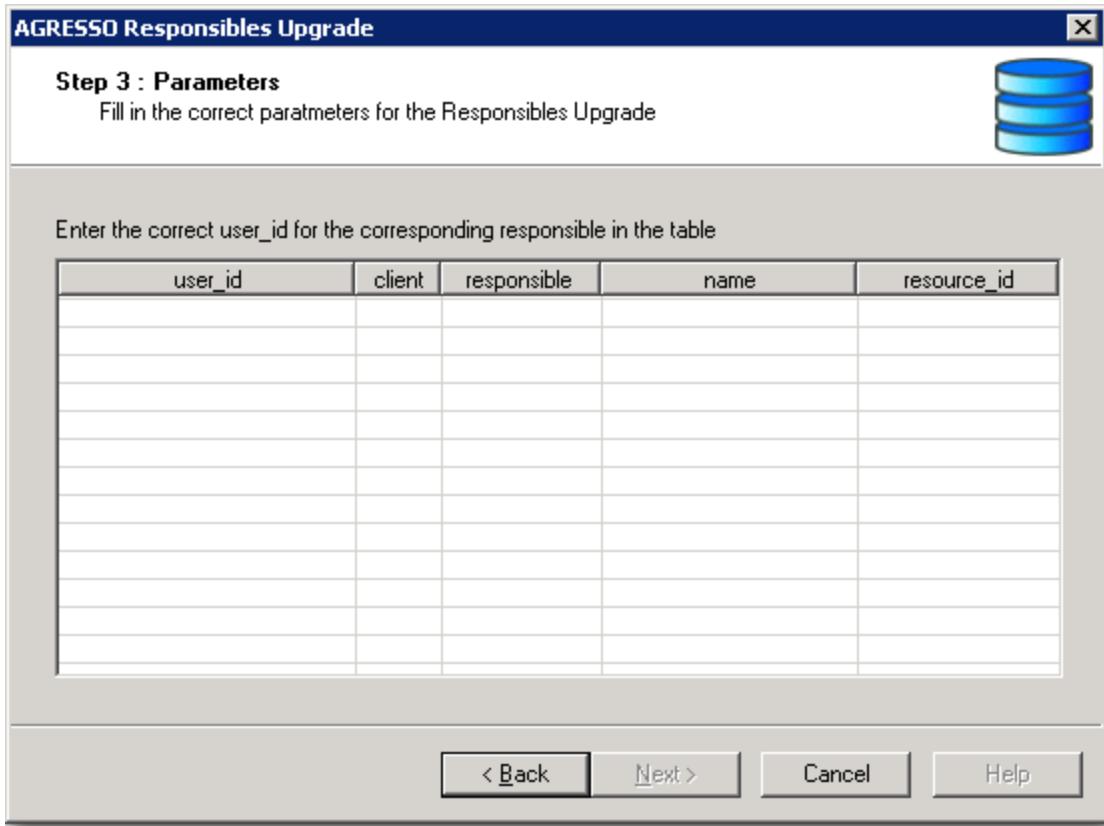
Responsibles Upgrade - Step 2



2. Click **Start** to run the script, and then **Next** (we assume that everything went OK after **Start**).
The wizard will match all identifiable resource_ids with the 5.5 user_id. When completed, the status for the *algresponsible* table is set to S.

You will now be prompted to manually enter missing user_ids:

 **Responsibles Upgrade - Step 3**

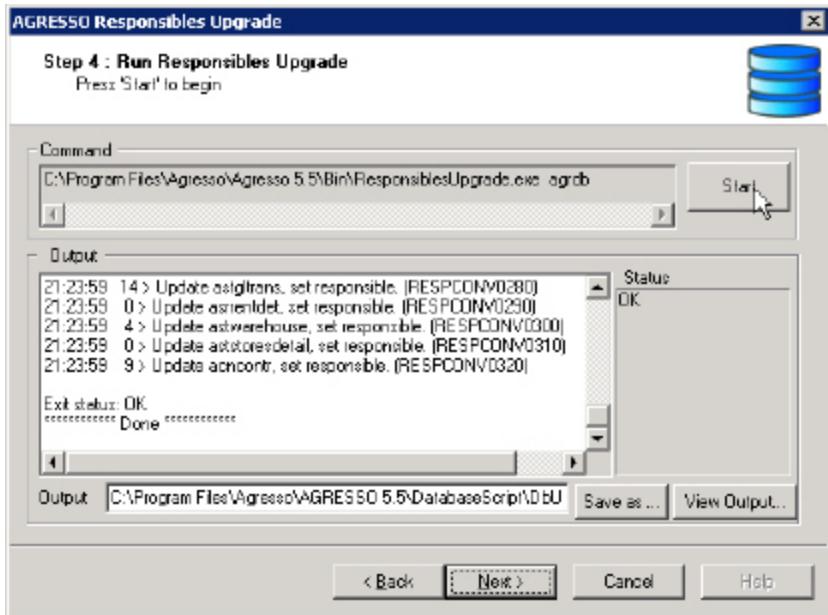


3. Repeat steps a. and b. as many times as necessary:

- a. Add missing user_ids.
- b. Click **Apply**
- c. Click **Next** when you are finished.

The next task is automatic order upgrade.

 **Responsibles Upgrade - Step 4**



4. Click **Start**, and then **Next** to finish.

This wizard can be run several times if you would like to add user_id's at a later stage.

Complete upgrade - additional manual tasks

There will normally remain some unmatched responsible codes on the orders. If so, the responsible roles must be registered in the Agresso Smart Client, and the orders must be updated in their respective registration windows.

HRMS Upgrades and Setup

Wizards

You must run two wizards:

[Balance Upgrade](#) and

[Work Schedule and Absence Details Upgrade](#)

Manual setup

The manual setup tasks are described in the following documents:

- *Release notes, AGRESSO 5.5, HRMS/PCB integration*
- *Service Pack notes, AGRESSO Business World 5.5 SP2, Project*

Projects Upgrade and Setup

Wizards

To upgrade the Projects module, you will need to run the following wizards:

[Balance Upgrade](#) - if not already run,

[Work Schedule and Absence Details Upgrade](#) - if not already run, and

[Timesheet Upgrade](#).

Manual setup

See the documents:

- *ABW 5.5 Reference manual, PCB Setup resources, time costs & income*
- *Service Pack notes, AGRESSO Business World 5.5 SP2, Project*

Finalise Upgrade

Overview

To complete the upgrade, you should:

- Check the converted Browser templates, and correct any errors.
- Clean up duplicates and add indexes.
- Correct all user defined views due to the table changes (see Appendix, Agresso Data Dictionary)
- Remove all tables no longer in use.

Check and correct Browser templates

Tools are provided to administer and ease the upgrade process for Browser templates:

- A Browser checker utility, `BrowserTemplateChecker.exe`, introduced in 5.5 Service Pack 1 that is run after upgrade.
- Check the log file produced during the upgrade. In this log, detailed information can be found of the Browser templates which need to be adjusted to run correctly on the latest version of Agresso. To fix the problem, open the browser template, make the necessary changes and save.

Indexes and duplicates

When running the database check step in the wizard, there might be indexes missing due to duplicates.

Indexes might be very important, and missing or wrong indexes can lead to very poor and slow performance.

Check the log files if there has been any errors while creating the indexes.

See log file `.\Database Script\DbUpgrade\Upgrade\Log\logindex_<database>.log`

Use any database tool to find the duplicates (for example SQL Server Management Studio (SqlServer), SQL Developer (Oracle)).

Example on how to find the duplicates:

```
select distinct client, attribute_id from agldimension  
group by client, attribute_id  
having count(*) > 1  
order by client, attribute_id  
/
```

Delete/change so there are no duplicates in the table, and re-create the index.

Look up in the Appendix, Agresso Data Dictionary to see the correct index definition. Create the index using the preferred DBA Tool.

User defined views

The table structure is changed from release to release. Ensure all user defined objects are changed according to the new structure (See Appendix Agresso Data Dictionary)

Change the definition and recreate the views

See log file .\Database Script\DbUpgrade\Upgrade\Log\logviews_<database>.log

Amendment logging

Amendment logging triggers needs to be recreated after the upgrade. Use the Smart client to open the **AG30 Activation of logging server** from **System Administration | Data Control | Amendment logging**, and regenerate active logging triggers.

Remove tables not longer in use

The script *drop.aspx* in the *Scripts* directory will drop all old and temporary tables no more used by the application.

Note: This script must be run when the upgrade has been completely verified, and you are sure that none of the old tables are needed for backup purposes.

Document Archive Upgrade

New Document archive solution

The Agresso Document archive was completely rewritten with Agresso 5.5. The document archive is now an integrated part of a series of Agresso windows, and documents of any type (.doc, .pdf, .jpg etc.) can now be attached to most of the Agresso object's.

These profound changes require that all the previous documents must be converted.

Please refer to *Agresso 5.5 Release Notes, Document Archive* for details about technical and functional aspects of the new solution.

Invoice Manager Reference

See the topic [Workflow and Invoice Manager](#) for information on how to interface your old Compello images with the new Agresso document archive.

Running the upgrade wizard

Necessary preparations

During the upgrade, you will need to make a few selections in order to proceed (see 3. below). The parameters you must enter are explained in the following table:

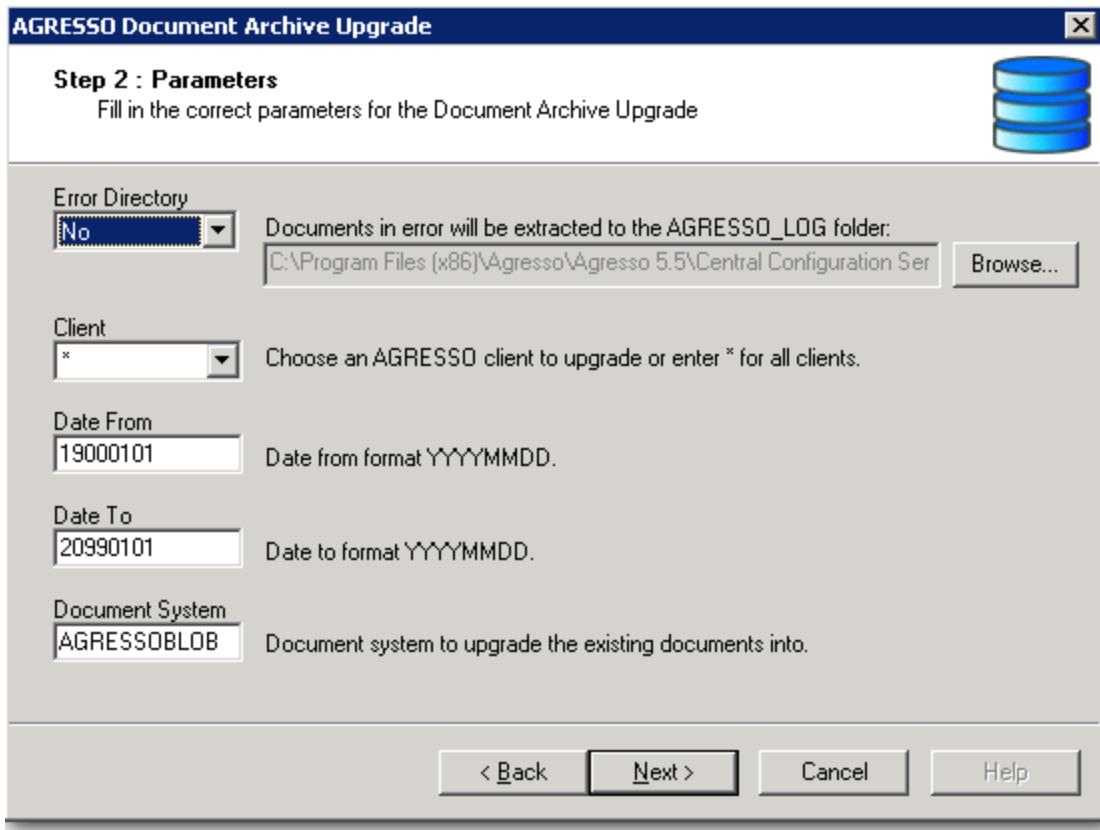
Field	Default value	Description
Error Directory	No	<p>Yes or No.</p> <p>If Yes, erroneous documents will be extracted to the server logging (AGRESSO_LOG) directory for manual fixing.</p> <p>If No erroneous documents will just be reported in the log (not extracted).</p>
Client	*	The Agresso client to upgrade. An asterisk (*) means all clients.
Date from	19000101	All documents registered at or after Date from will be included. Older documents will be ignored. Format: YYYYMMDD.
Date to	20990101	All documents registered before or at Date to will be included. Newer documents will be ignored. Format: YYYYMMDD.
Document system	AgressoBLOB	The new document archive can be connected to many document systems, including third party archives. You must choose a document system to upgrade the existing documents into. The default value is the new Agresso database archive.

Upgrade procedure

To upgrade to the new document archive solution, you will first start the Upgrade wizard. Continue as follows:

1. Double-click the [Document Archive](#) link in the Database Upgrade Wizards screen. After reading the introduction, click Next.
2. Enter connection information and click Next to prepare for step 2.

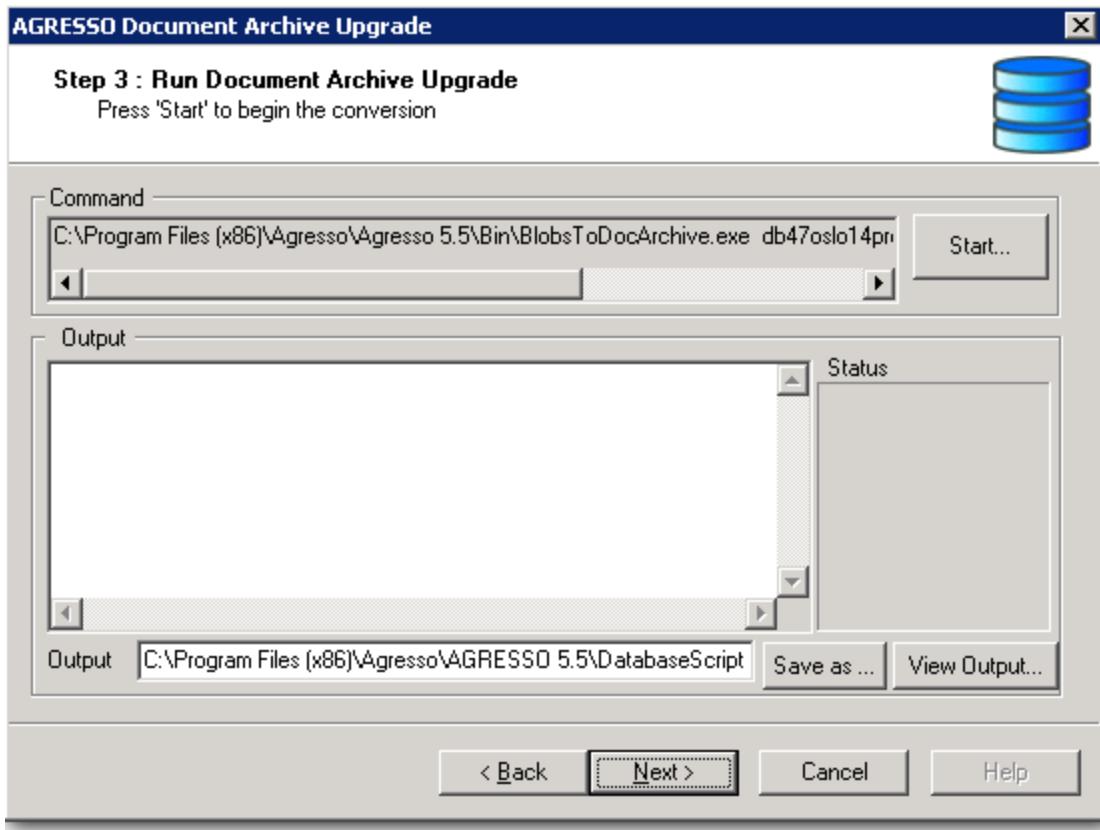




3. Fill in the required parameters and click **Next**.

You will now have to verify the path to the log file (output) made by the upgrade scripts.

 [Document Archive Upgrade - Step 3 #1](#)



4. Make sure that the output path is correct, and run the upgrade by clicking the **Start** button.

5. When the conversion is done, do as follows:

- a. Click **View Output** to study the conversion results
(we presume everything is in order)
- b. Click **Next** to bring up the final window.

6. Click **Finish**

Note: If necessary, check the log file once more and make sure everything is correct.

Create System Attributes

Description

The wizard **Create System Attributes** will generate all the fixed attributes currently missing in your UNIT4 Agresso installation.

The wizard is part of Step 6. **Run binary programs** in the main system upgrade wizard, but is also available in a stand alone version, available in the file [CreateAttributesWizard.exe](#).

Standard (fixed) attributes only

The wizard updates only attributes delivered as part of the standard **UNIT4 Agresso** installation (with attribute id from **A0** to **LZ**).

Other attribute types, for example localisation specific attributes or custom attributes, will not be updated by the wizard.

All clients will be updated

Attributes are client dependent and the wizard will generate one attribute instance per client. Attribute names and descriptions will comply with the selected language (Company Information screen, CR01) for the various clients.

Run the wizard

The wizard is very straightforward, with no complex options. Just follow the instructions on screen.

Responsibles Upgrade

Changes in data structure

The new solution for users and roles has required a completely new implementation of the *Responsible* concept in the Logistics module. Previously, the *algresponsible* table hold information about responsible codes and the *resource_id* (not the *user_id*) that was linked to the code.

Agresso 5.5 introduced a solution where a responsible belongs to a certain role, and where role membership is based on the *user_id*.

Upgrade of orders created by Agresso 5.4x

The main purpose of this Responsibles Upgrade, is to convert active orders (created in the various logistics modules) in such a way that the responsible person still can be identified when the order is further processed.

Note

The upgrade will *not* convert the old responsible resources into new responsible *roles*.

Upgrade tasks

The main upgrade tasks are performed by the responsible upgrade wizard, and are described as follows:

Task	Description
Automatic code matching	Match resource_ids from the <i>algresponsible</i> table with the 5.7.0 users. If a user can be uniquely identified by a <i>resource_id</i> , the upgrade wizard will establish a new connection between the 5.4 responsible code and the 5.5 user id.
Manual matching	For all unmatched <i>resource_ids</i> , you are prompted to manually enter <i>user_ids</i> . There is no requirement that the new user really takes part i 5.5 responsible role. This task can be repeated several times, until all (old) <i>resource_ids</i> have a valid match.
Automatic order upgrade	The wizard will search through all relevant orders and convert all orders with a responsible codes where there has been a sucessful match. Log: The result will be written to the log, and can be viewed in the table <i>algprophead</i> .
Manual order update	For all the orders that were not converted, you will have to use the registration windows for the various order types, to set the correct responsible. Note: This requires that the various responsible roles are set up in the Responsible Setup window in the Smart client. See <i>Agresso 5.5 Release Notes, Logistics – Common logistics</i> .

Run the upgrade wizard

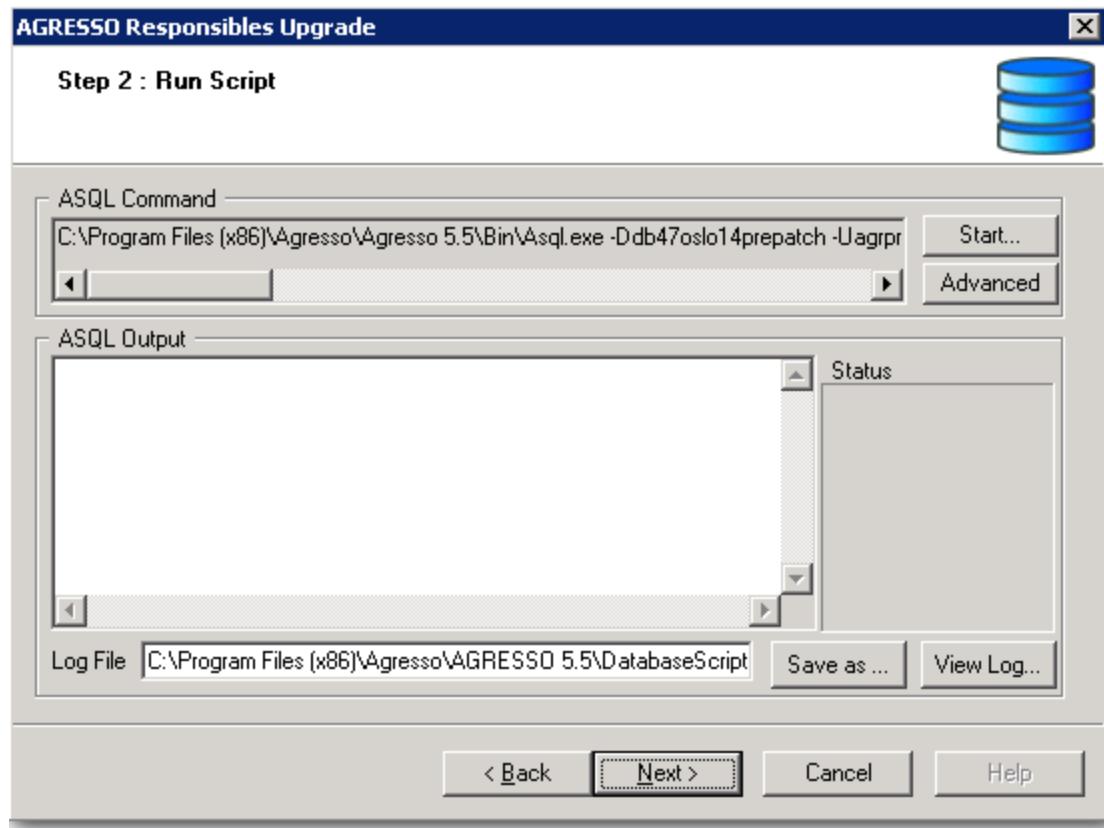
When the **Responsibles Upgrade wizard** is up and running, navigate to the Step 1 window, and do as follows:

1. Select correct data source from the drop-down list, fill in the correct password and click **Next**.

You are now ready to run the upgrade scripts.

Note: If you already have run the upgrade scripts once, and in addition performed some manual matching (3. below), a re-run of the wizard will remove all manual updates!

 **Responsibles Upgrade - Step 2**

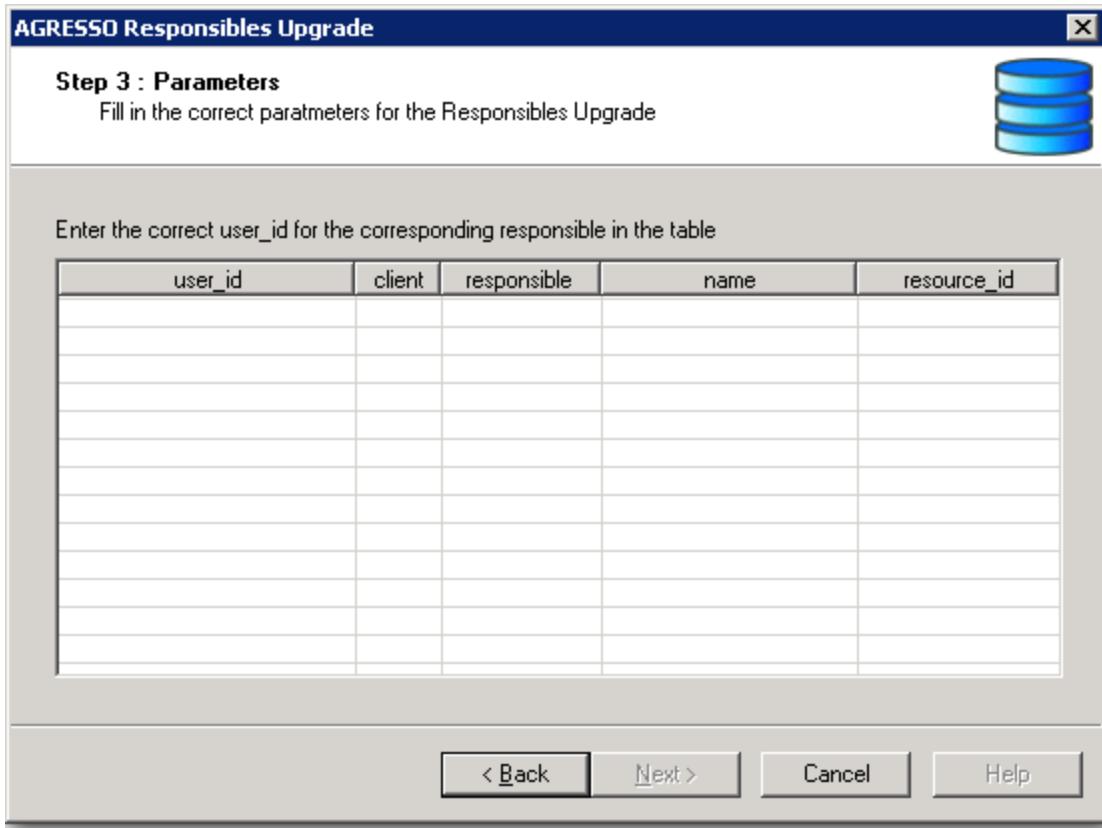


2. Click **Start** to run the script, and then **Next** (we assume that everything went OK after **Start**).

The wizard will match all identifiable resource_ids with the 5.5 user_id. When completed, the status for the *algresponsible* table is set to S.

You will now be prompted to manually enter missing user_ids:

 **Responsibles Upgrade - Step 3**

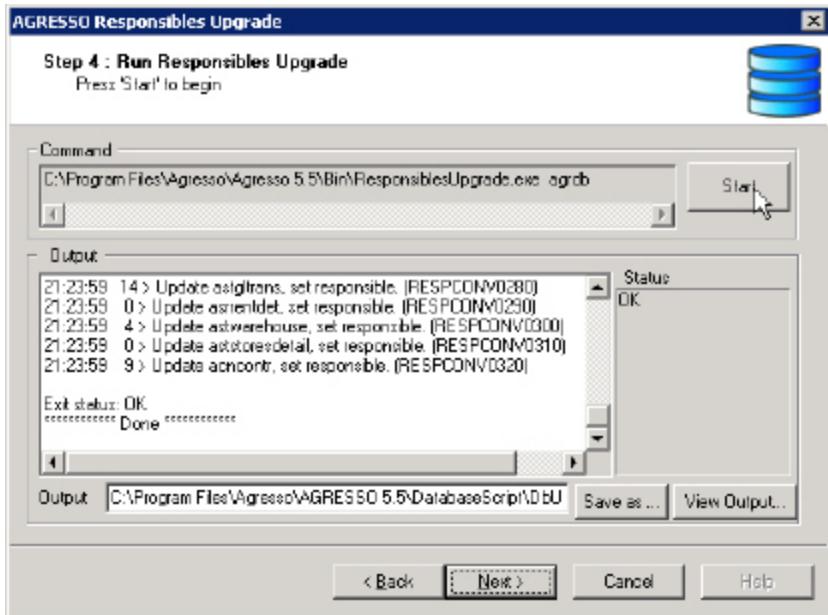


3. Repeat steps a. and b. as many times as necessary:

- a. Add missing user_ids.
- b. Click **Apply**
- c. Click **Next** when you are finished.

The next task is automatic order upgrade.

Responsibles Upgrade - Step 4



4. Click **Start**, and then **Next** to finish.

This wizard can be run several times if you would like to add user_id's at a later stage.

Complete upgrade - additional manual tasks

There will normally remain some unmatched responsible codes on the orders. If so, the responsible roles must be registered in the Agresso Smart Client, and the orders must be updated in their respective registration windows.

Balance Upgrade

New solution for balance handling

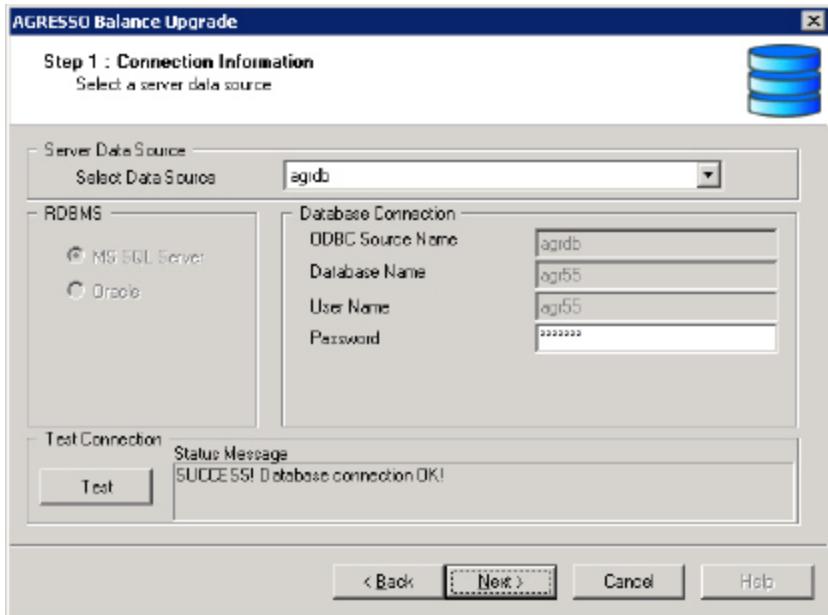
Prior to Agresso 5.5, the modules Project and Payroll had different solutions for balance handling. From Agresso 5.5 a new, common solution was introduced, and old Balance data must therefore be converted.

Procedure

When the wizatrd is up and running, move to Step 1.



Balance Upgrade - Step 1

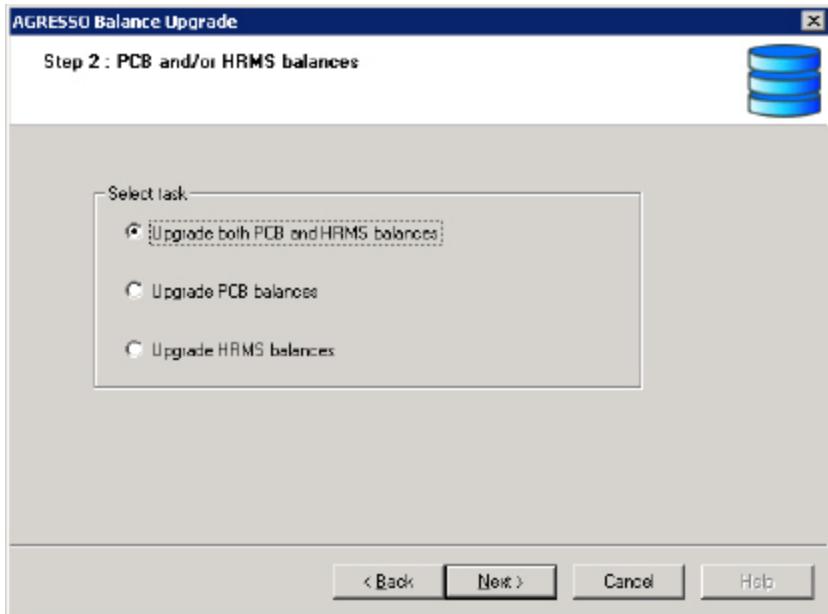


Do as follows:

1. Select correct data source from the drop-down list, fill in the correct password and click **Next**.

You will be prompted to select upgrade option:

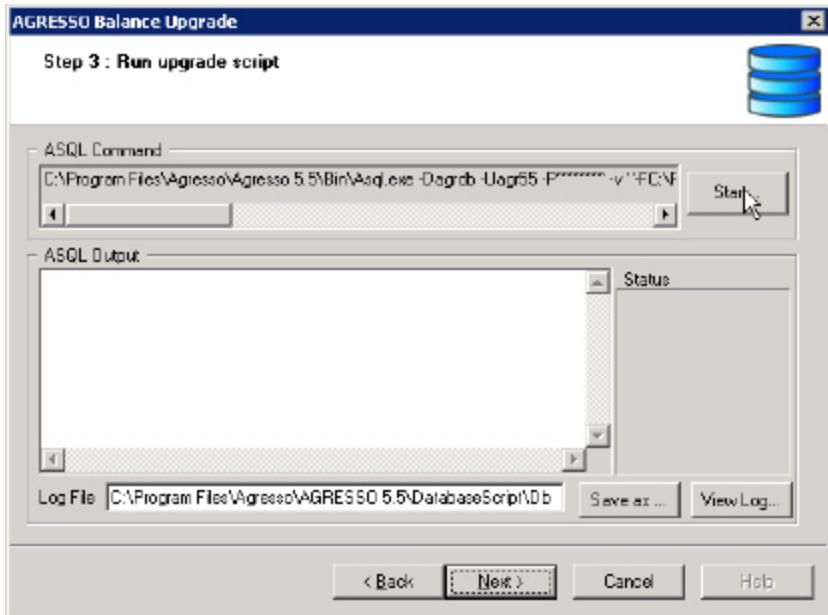
Balance Upgrade - Step 2



2. Make your selection and click **Next**.

You are ready to run the selected upgrade script which will move date to the new *ahsbalance* table.

Balance Upgrade - Step 3



3. Click **Start** to run script, and then **Next** to continue.

The upgrade is completed and you can close the wizard.

Work Schedule and Absence Details Upgrade

Changes from 5.5

Previously both TR and HRMS had separate setups for work schedule. From version 5.5, these are merged into one common setup. Moreover, to use absence, the employees now need to be connected to a work schedule (through relation, system parameter or position). Previously this was optional.

The data model has changed. Absence is now stored per day in a new absence details table, which is not the case for already registered absences. To support reversing, reporting and enquiry purposes, the old absences must be converted.

Converting absence entries

To convert old absence entries, you need to decide whether you want to create work schedule entries for the employees' old overrides of personal work schedules (previously stored in apscalendar). Previously you could override the employees personal work schedules upon absence entry. If you need the history of this to be incorporated in the employees work schedules, you must specifically select to do that (see step 8 below).

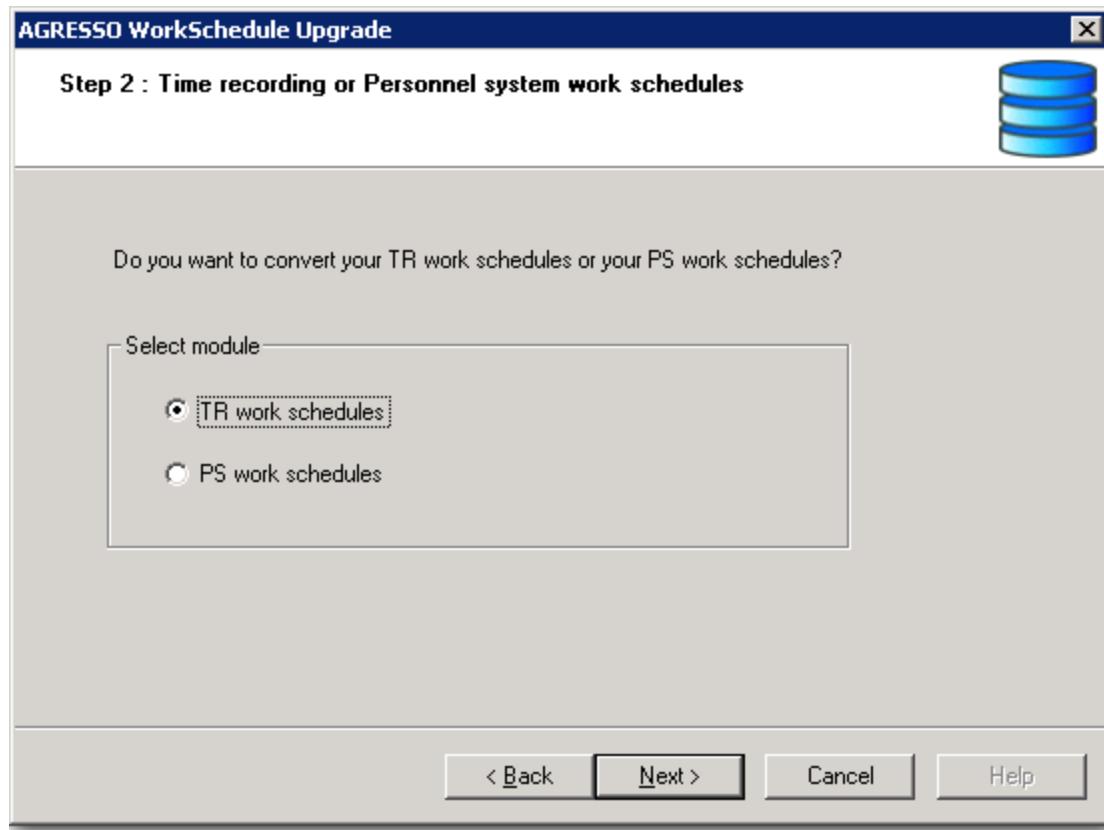
Note: Beware that if you have many and old overrides, this will create many rows in the work schedule tables. It might be wise to clean out very old data before converting.

Upgrade

You use the Agresso Work Schedule Upgrade wizard on the Upgrade DVD to run the upgrade scripts. This is not a straightforward process, however, since the wizard requires that a system parameter is correctly set, and that the server processes **HS05** and **HS04** are run between two wizard steps.

- 1 Start the Agresso Work Schedule Upgrade and click **Next** to proceed to Step 1.
2. Enter connection information and click **Next**.

 Work Schedule Upgrade - Step 2

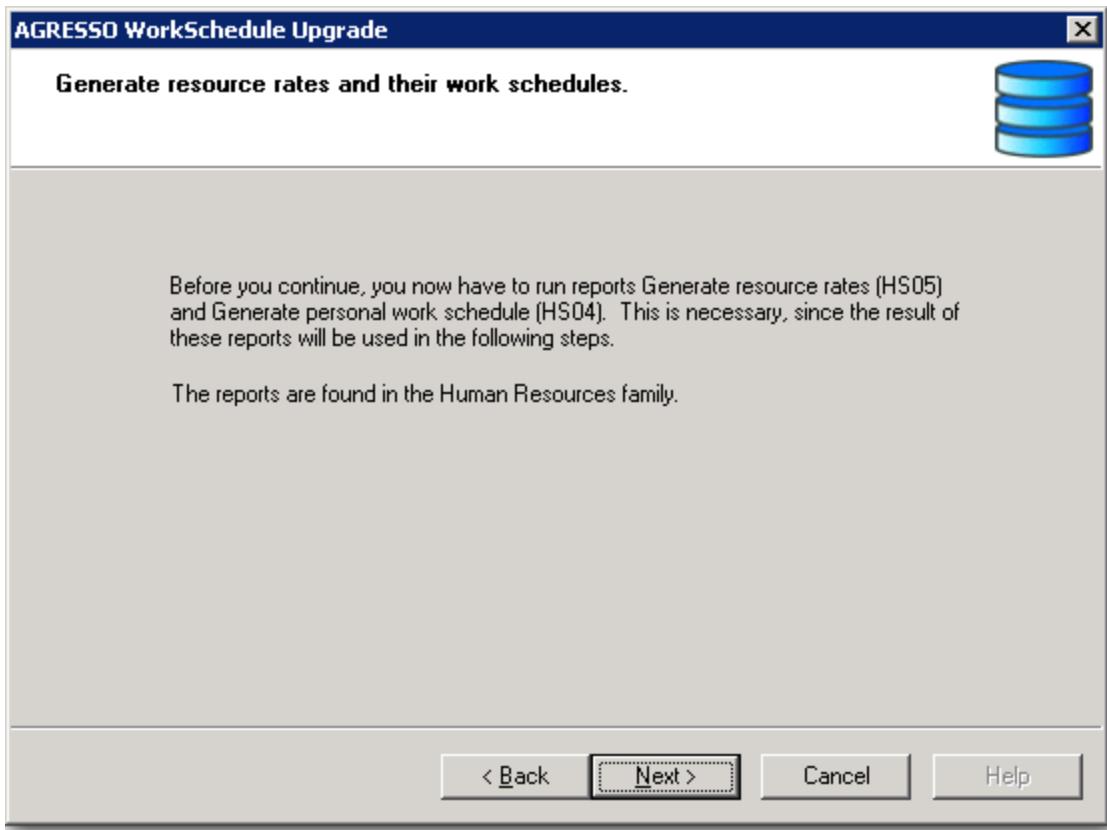


3. Select the schedules to upgrade and click **Next**.

You will now be ready to run the upgrade script for the selected schedules.

4. Click **Start** to run the script and then **Next** (if errors, see the log file in the ASQL Output window).

 Example: Update Agresso



You now need to complete the upgrade from the Agresso Smart client.

5. Start the Agresso Smart client and make sure that the system parameter HS_PARTTIME_PCT is set to the value reference you use for the employees part time percentage.

System Parameters

EN System parameters - Payroll/Personnel/Expense								
		Setup Template Reset parameter to system setup value Reset parameter to default value			Add Link Organise Links			
Sys.setup	EN	Client	EN					
?								
2	ABS_DAYS_BETWEEN	HS	8		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
4	HS_CO_STATUS	HS	60	status	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
6	HS_DAY_BREAK_TIME	HS	4	0000	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
7	HS_DEF_DAY_VAR	HS	2	N	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
8	HS_DEF_WORKSCH	HS	25		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
15	HS_MARK_DAY	HS	99	Sunday	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
18	HS_PARTTIME_PCT	HS	25	I001	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>

6. Run **HS05 Generate resource rates** with the following parameter settings:

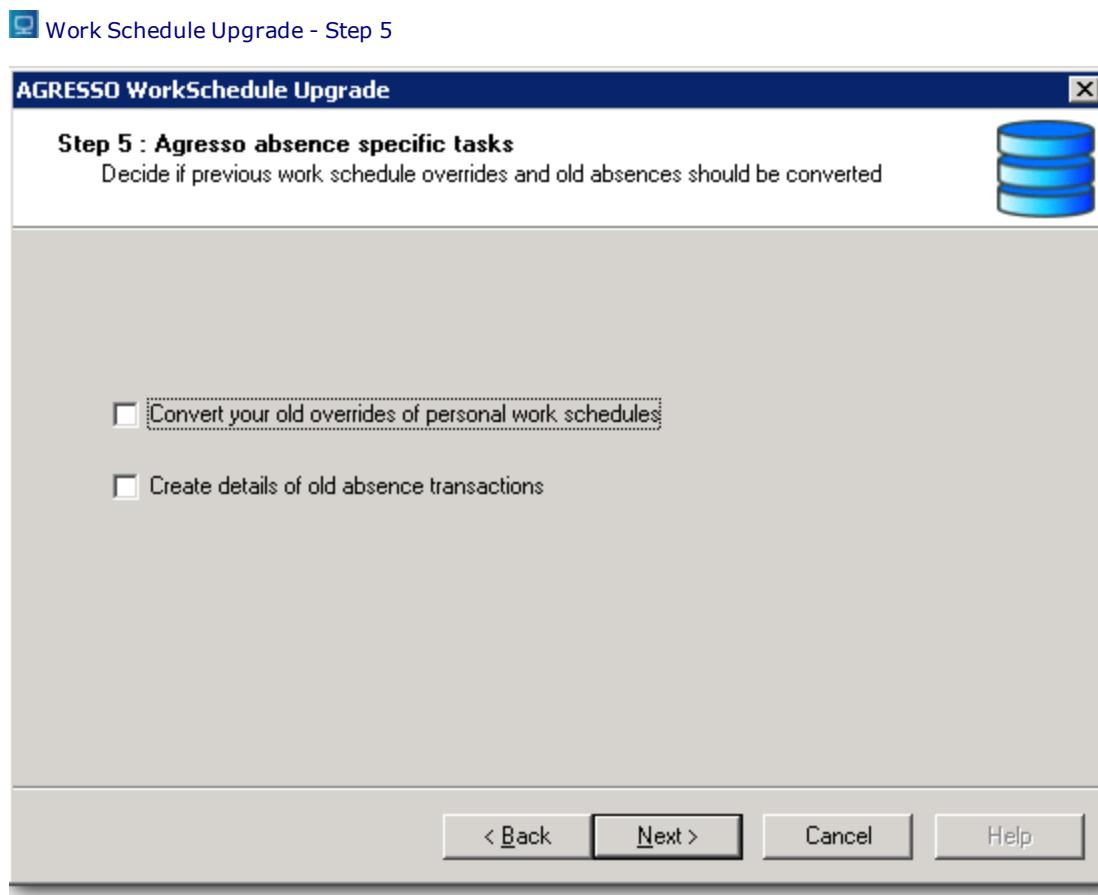
- Set the **Date from** as far back in time as you need to create rate history for the employees.
- Leave the **Relation** parameter blank and set **Relational value** to *****.

All rates will be converted to the new system of resource rates. This is essential to make Agresso HRMS run like before, regardless of the work schedule upgrade.

Note: Before you proceed with the next step, be aware of the following: If you want to use a holiday scheme to be included in the creation of the different work schedules, this must be set up in the **Holidays** window, before you run **HS04 Generate personal work schedule**.

7. Run **HS04 Generate personal work schedule** for all work schedules.

When step 7 is completed, you must return to the upgrade wizard and make a decision on absence specific tasks. See Converting absence entries above.



8. Make your selection and click **Next**.

You are ready to run the final scripts.

9. Click **Start** to run the scripts, and then **Next**. (If errors, see the log file in the **ASQL Output** window).

The upgrade is completed and you can close the wizard.

Timesheet Upgrade

Introduction

The Timesheet upgrade wizard will upgrade the time transactions from both *group timesheets* and *personal timesheets*, based on the parameters you choose. The wizard is available on the Upgrade DVD. All time transactions will be stored in new, common tables, and before they are converted to the new data structure, you will not be able to view them in the timesheet windows.

Note: Time transactions that have *not* been updated can still be viewed using a browser enquiry against the 5.4 tables.

Parameter information

The wizard will upgrade timesheets for selected periods. You have to enter period from, posting period, and for the Group timesheets you will also enter last complete period and attribute id. The period from needs no further explanation, but the other three require a few words:

Posting period: The period will be used for selection voucher numbers for the timesheets. The voucher type used in the upgrade is the value of the system parameter VOUCHER_TYPE.

Last complete period: For this period and all previous periods, the timesheets status will be set to Terminated if all rows are terminated. If not, the timesheet status will be set to Draft. The rows which are not terminated will also be set to Draft. For timesheets after this period the status will be set to Draft for both the timesheet header and rows.

Run the Upgrade wizard

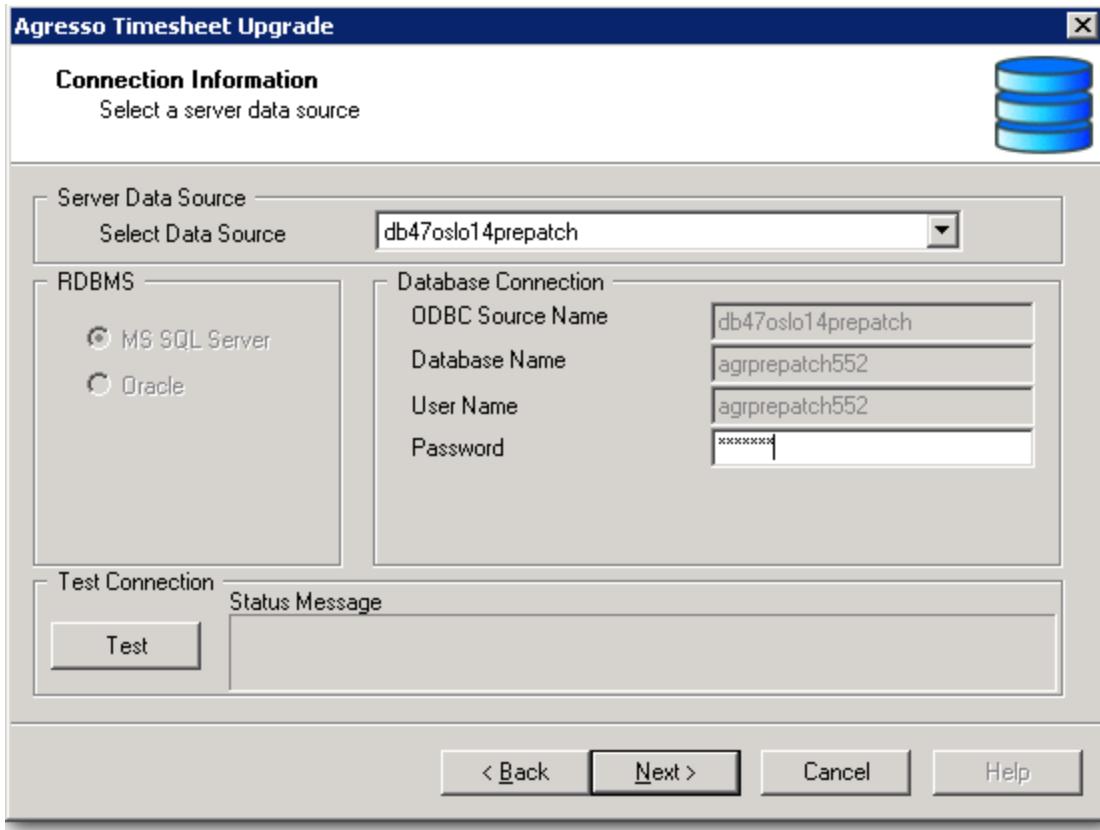
Before you continue, you must have the **Timesheet Upgrade wizard** up and running

Connect to database

1. Navigate to the Step 1 window.

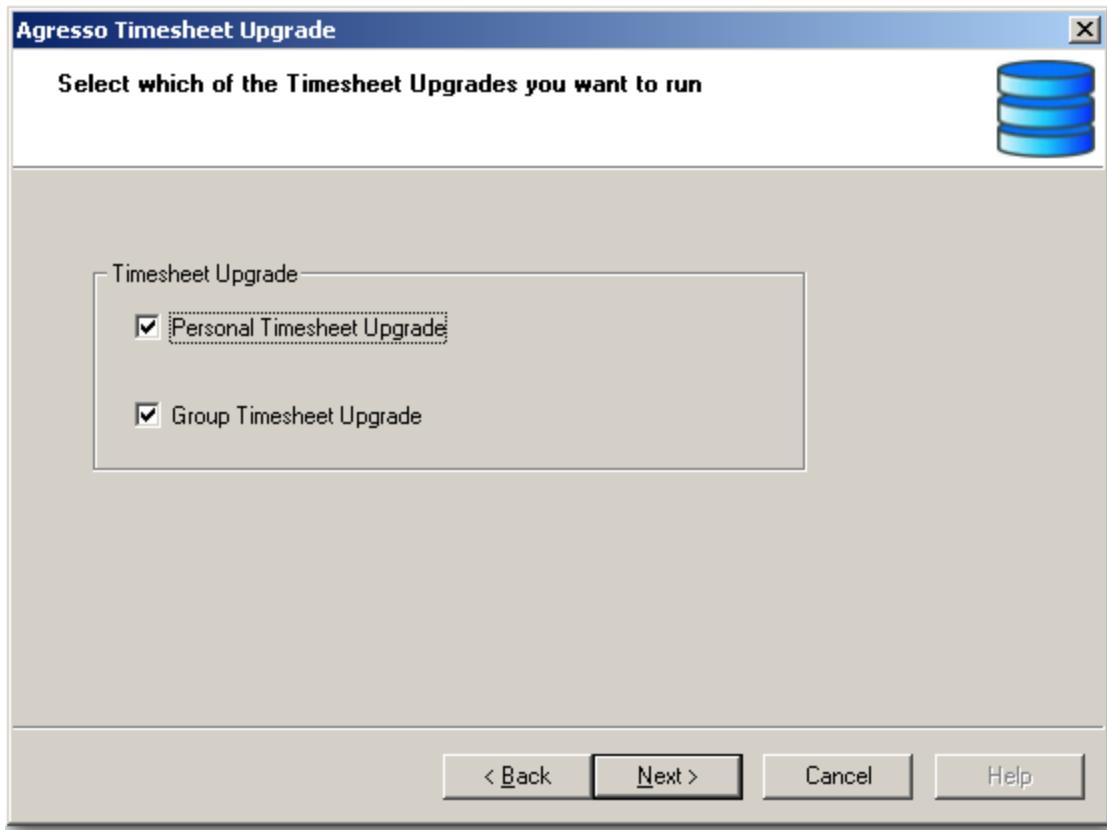


Example



- 2.** Select the correct data source, enter correct password, and click **Next**.
You are ready to select what to upgrade.

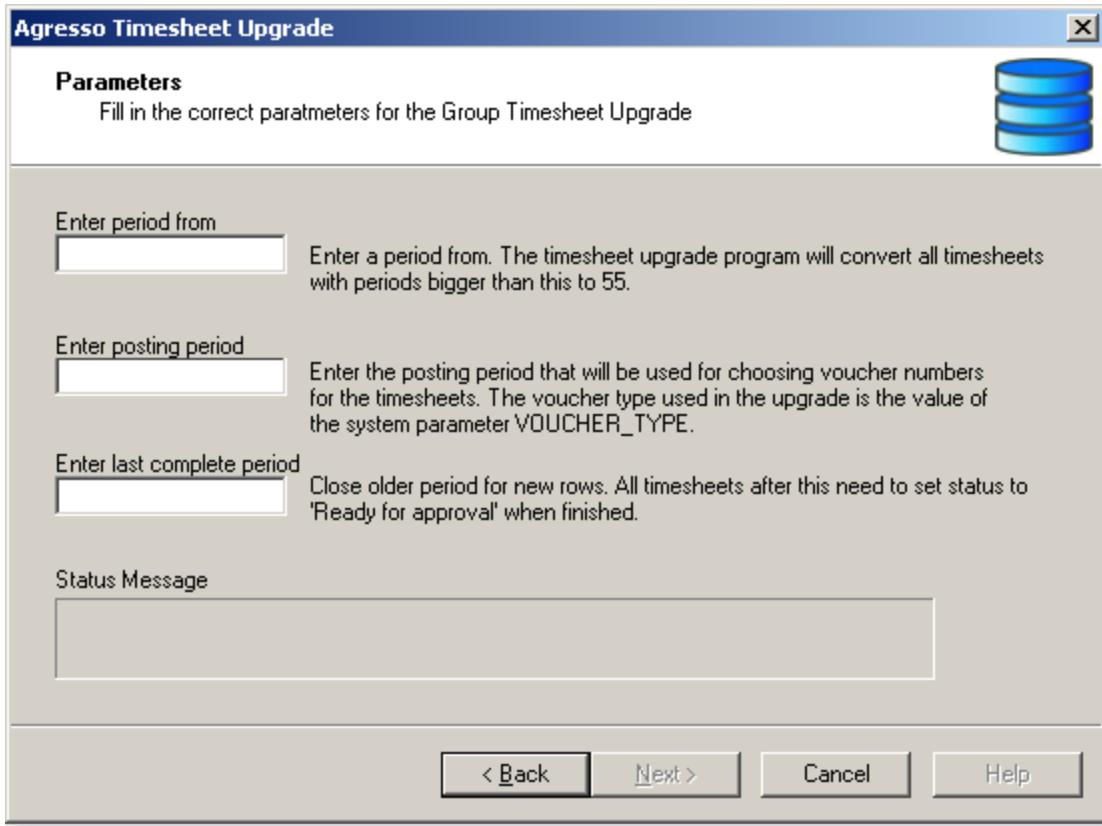
Example



Select items to upgrade and run scripts

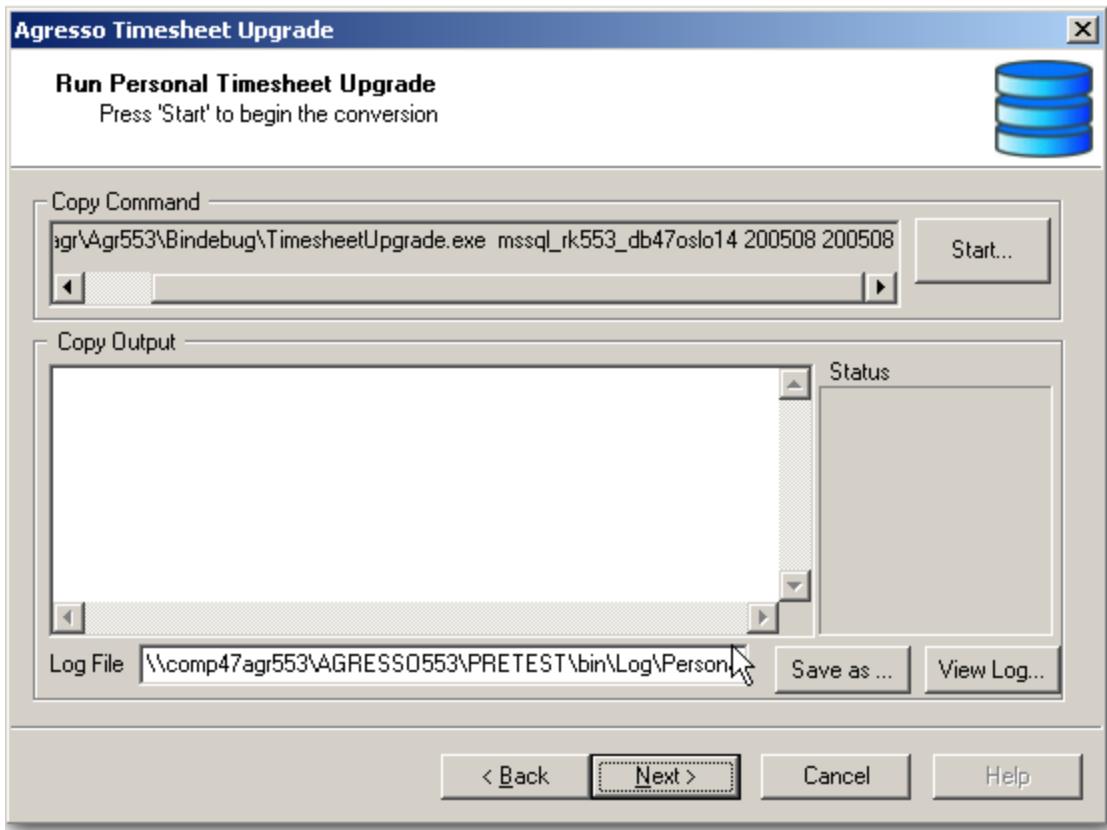
3. Make your selections and click **Next**. You are prompted to enter periods.

Note: The number of fields in the dialog depends on what you selected: Example



4. Fill in correct periods and click **Next**

Example



5. Click **Start** to start upgrading.

6. Click **Finish** to complete the timesheet upgrade.

Copy Setup Between Databases

General description

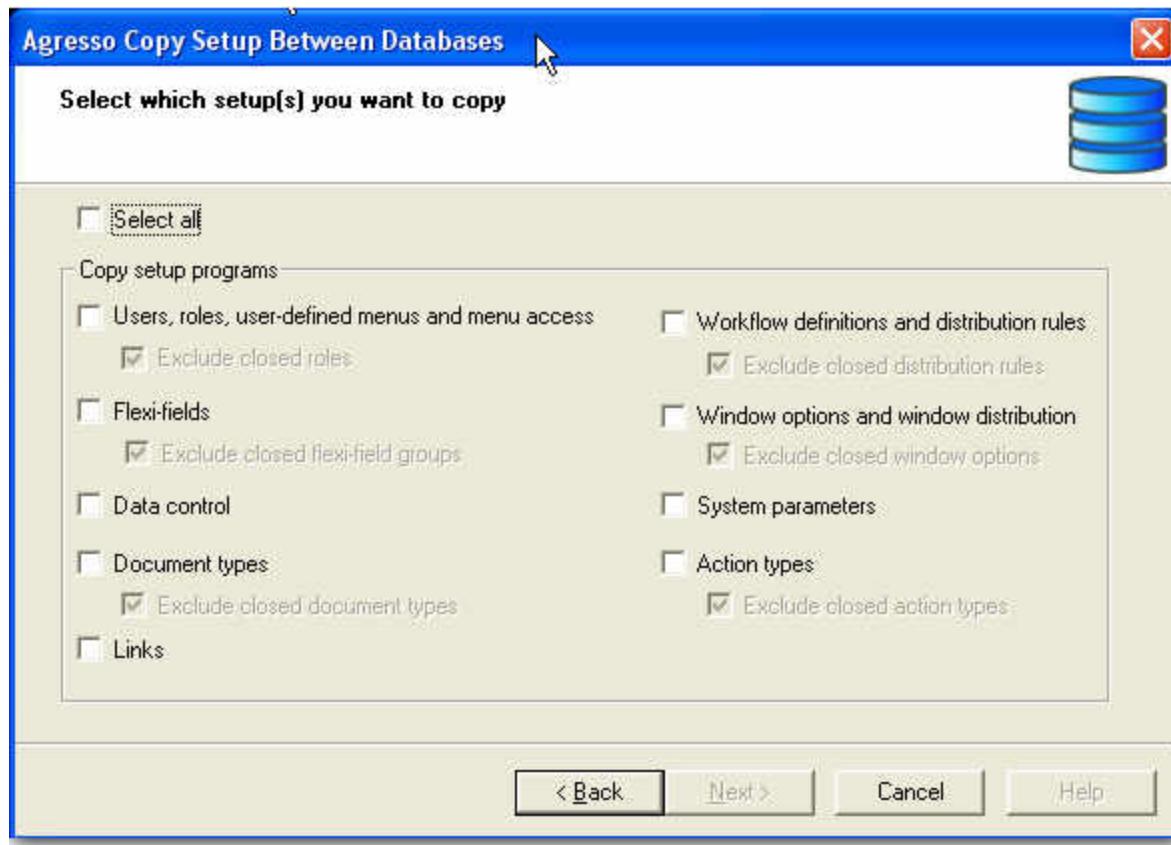
The **Copy Setup Between Databases** wizard copies the Agresso setup from one database to another. The following areas may be copied:

- Users, roles and menu access
- Workflow definitions and distribution rules
- Flexi-fields
- Window options and distribution of window options
- Data control
- System parameters
- Document types
- Action types

Note: Existing setup in the target database will be overwritten. See Dialog example below.

Dialog Example

The various options are presented in a single window:



Intended use

The wizard is created to facilitate transfer of a new and working configuration in the test environment, to the production environment.

Used in this context, it is important that all other updates are completed – on the test database – before you run the **Copy Setup Between Databases** wizard.

Run the wizard

Note: If you need more details before you run the wizard, see [Wizard details](#) and [Updated tables](#) below

When the wizard is up and running, do as follows:

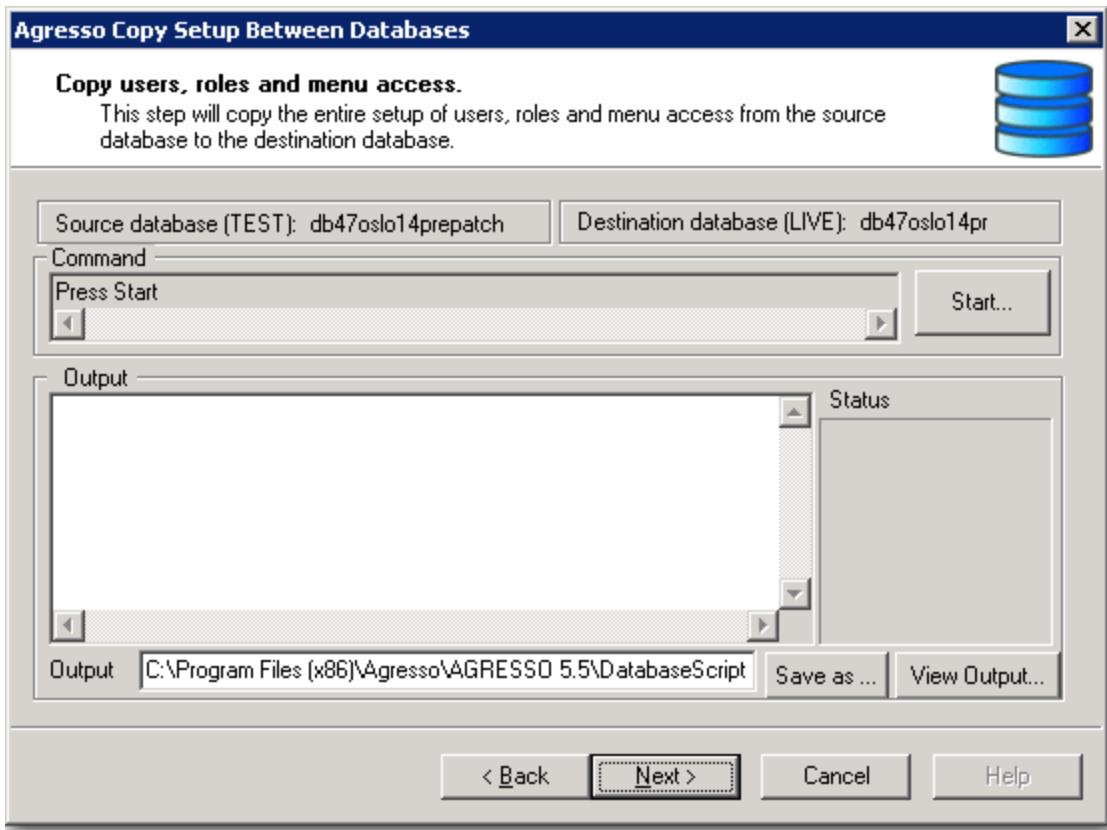
1. Enter connection information for the *Source* database and click **Next**.
2. Enter connection information for the *Destination* database and click **Next**.

You are presented with a few copy options. See Dialog example above.

3. Make your selections and click **Next**. The wizard is ready to start the Copy operation.



Example.



4. Click **Start**. You will need to confirm copying for each selected area.

Wizard details

The various copy programs (wizards) are described below:

The wizard ...	Will copy the following from source (test) database to target (production) database ...
Users, roles, user-defined menus and menu access	<ul style="list-style-type: none"> • New users (aaguser) with status Active • User information (addresses and passwords) connected to new users with status Active • All user detail information (aaguserdetail) • All roles • All menu access (aagaccess) granted to both users and roles • All user-defined menus (aagmenu) <p>You can exclude roles with status Closed from being copied to the destination database (default).</p>
Workflow definitions and distribution rules	<p>The entire setup of workflow and distribution rules. Existing setup will be replaced.</p> <p>You can exclude closed distribution (default).</p>

Flexi-fields	<p>The entire setup of Flexi-fields. Existing setup will be replaced.</p> <p>You can exclude closed Flexi-field groups (default).</p> <p>Note: No Flexi-field data will be copied!</p>
Window options and window distribution	<p>The setup of window options and distribution of window options. All existing setup will be replaced.</p> <p>You can exclude closed window options (default).</p>
Data control	<ul style="list-style-type: none"> The entire setup of data control. All existing setup will be replaced. All relations and relation values connected to attribute A11 &ndash; ROLEID All setup made in Data control management (CR49). <p>Note: The program also updates all attributes with data control activated in the <i>source</i> database.</p>
System parameters	<p>System parameters.</p> <p>All changed parameters will be listed in a report file. The path to this report file is specified in the program's log file.</p>
Document types	<p>The setup of Document archive. All existing setup will be replaced.</p> <p>Note: No documents will be copied!</p>
Action types	The setup of Action overview. All existing setup will be replaced.
Links	The setup of Links. All existing setup will be replaced.

Updated tables

Below we list the tables affected by the copy programs, and indicate the operations they are exposed to:

Users, roles, user-defined menus and menu access

The following tables are affected:

Table name	Data replaced	Rows inserted	Rows deleted
awfelemtype	X		
awfelemtypedet	X		
awfelemtypegrouping	X		
awfelemtypeitem	X		
awfelemtypemap	X		
awfprocaction	X		
awfprocdeadlines	X		

awfprocdelay	X		
awfprocdelaydet	X		
wawfprocelemtype	X		
awfprocess	X		
awfprocfunction	X		
awfprocrole	X		
awfprocsplit	X		
awfrule	X		
awfruledet	X		
awfrulegroup	X		
awfversion	X		
acrdiagramlayout	X		
awfalternate	X		
awfblmethods	X		
awfcolumns	X		
awfelemtypemenu	X		
awfmanstep	X		
awfprocrcd	X		
awfprocsin	X		
awfprocsrp		X	
awfproctin		X	
awfuserdetail	X		
aimblob		X	X

Flexi-fields

The following tables are affected:

Table name	Data replaced
aagpagesetupconnect	X
aagpagesetupdet	X
aagpagesetuphead	X

Data control

The following tables are affected:

Table name	Data replaced
aagparameter	X

Document types

The following tables are affected:

Table name	Data replaced	Rows inserted	Rows deleted
acraktionelemttype	X		
acraktiontype	X		
agldimvalue		X	X
agldescription		X	X
acraktioncontact	X		
acraktionattval	X		
acracctionattvaldet	X		
awfelemtype	X		
awfelemtypedet	X		
awfelemtypeitem	X		

Links

The following tables are affected:

Database Maintenance

Common maintenance

The most important considerations involves:

- Indexes
- Database backup
- Space allocation and Database surveillance
- Scheduled jobs
- Report files

Indexes

Add customized indexes

You can add indexes to your system to increase the performance of the database. Agresso delivers some necessary indexes for maintaining row uniqueness and for performance purposes. When adding indexes in the database, you should use the **Indexes** maintenance window (AG18) in the Smart client. The customized indexes will then be stored in the table *aagindex*.

! Agresso will not save any indexes except those you have captured within Agresso under an upgrade or patching.

Recreation of Indexes

To maximize the performance and speed, it is important that the indexes are correct. The report **CR29 Rebuild database indexes** (Agresso Smart client) or the [AgrIndex](#) program can be used to recreate all the indexes delivered at installation time.

Database Backup

Basic rule

You should regularly take a complete backup of your database.

When errors occur

The Database administrator or System administrator is usually responsible for the Agresso data, including backup. Erroneous production data must be handled locally. Your support representative will help you whenever possible. An important precaution is to establish - and maintain - good procedures for data validation and consistency checks. Without such procedures, errors may be impossible to correct.

Prevent loss of data

There are at least three methods to prevent loss of data:

- Use mirroring of all disks. Make sure that all data is stored on two identical disks. As long as one of them is available, you are safe.
- Use extra disks for control information (RAID). This makes it possible to continue without data loss if one disk in a disk set should crash.
- Make a complete copy of all your data at fixed intervals.

Some RDBMS's can handle both on-line and off-line backup. Unless you need 24-hour access to your data, you should use offline backup.

Offline Backup

This means that you shut down your database system at some time during the night or weekend, make a copy of all database files, and finally restart the database. These operations should be put together in a script and executed by your OS scheduling system.

In addition to complete backups, it is possible to back up only incremental changes to the database. Then you do not have to run the complete backup too often. The incremental backup is often put on a disk drive, whereas the complete backup is sent to a tape. Some RDBMS's can handle the incremental backup automatically.

The tapes from your complete backup must be handled safely. They should be kept in a safe place, and not where your server and disk drives are located.

Online Backup

This means that your database do backups when the database is running. To run online backup your database have to be in archivelog mode.

We recommend using Recovery Manager (RMAN) to take care of the online backups.

Space allocation and database surveillance

Most RDBMS's use pre-allocation of space for database objects. When your tables and indexes occupy all the disk space allocated at installation time, you will need to allocate more space.

It is also important to keep track of database log usage. Some RDBMS's require manual action to make space available for log information. You may also need to change the size of your log area, if it is not sufficient to handle the load of your application.

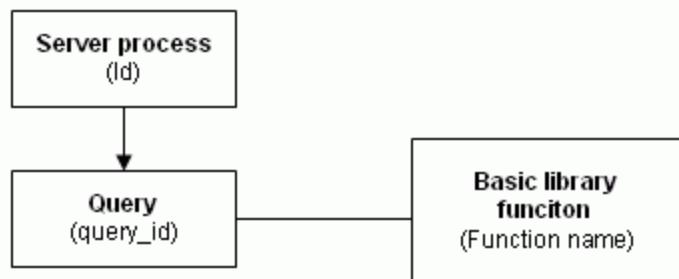
Tuning Overview

Running server processes

Queries and database operations

When you run a server process, a set of queries associated with the server process is executed. Each of the queries is identified by a unique query id (query_id), and each query will call a basic library function, identified by an Agresso function name.

This is illustrated in the diagram below:



The arrowhead illustrates that a server process executes one or more associated queries (the least complex process uses actually two queries) while the line between the query and basic database operation indicates that one query triggers one database operation.

The basic database operations

Agresso utilises six basic database operations, explained in the table below:

Agresso function name	Description
AGRCreateTableId	Creates a new database table. CreateTableId can also fill the new table with data defined by a select statement given as a parameter to the function (optional).
AGRExecSqlId	Executes any sql statement.
AGRCreateUpdateId	Creates a temporary table and uses it to do an update of another table.
AGRUpdTableId	Updates an existing table with data from (an)other table(s).
AGRSelIntoId	Selects a record from one or more tables and returns the values into a program variable.
AGRSelLoopId	Executes a cursor loop given by a select (parameter). For each record of the cursor loop one or two statements are executed. These are also given as parameters. SelLoopId may typically be used to do an insert or update, depending on whether the record already exists or not.

Tuning options

A tuning option in our context is the possibility to force an extra sql statement to be executed in addition to the original (basic) query. The intention is to speed up processing. A basic query can have numerous tuning options (extra sql statements).

Example: You can insert an sql statement before the existing query is executed to create a new index and another to make sure that the index shall be used in the basic database operation. In addition you can insert an option after the original query has finished, to ensure that the tuned query is written to the log file.

Available tuning options

Reference: The descriptions in this section relates to options you have in the [Server Process Optimization window](#).

Agresso provides a set of tuning options, described in the table below:

Tuning option	Description
NO_OPT	Used to overrule any other tuning options.
LOGQUERY	Writes the query to the log file.
INDEX	Creates a new index for a given table, based on selected columns.
HINT	Allows you to add hints to the query optimiser. Most native hints from Oracle and Sqlserver are supported.
STAT	Updates available statistics tables.
PLAN (Oracle only!)	The query plan will be printed to the log file.
ABORT_JOB	Aborts the job before or after (dependent on value of Position) the query has run.
SNAPSHOT	Makes a copy of a temporary table created by the query. The table sequence number must be entered in the Table column. Default=0 (first)
STRATEGY	Update strategy for query. Currently only relevant for Oracle 11.2 and later, when updating one table from another. You select the strategy in the Argument column: <ul style="list-style-type: none">• MERGE_ROWID - will use MERGE INTO with JOIN on ROWID• WHERE_EXISTS - will use JOIN in WHERE EXISTS• ROWID_IN - will JOIN on ROWID

Tuning option execution rule

The tuning option execution rule tells Agresso when a specific tuning shall be executed in relation to the original query. There are three basic rules of execution, which will behave slightly differently depending on the basic database operation that is performed when the query is executed.

- | | | |
|---------------|---|--|
| Init | — | Option is to be executed as early as possible, before the actual execution of the query. |
| Before | — | For multi step operations, the option will be executed after the first step. |
| After | — | After the query is finished. |

Time of execution and database operation

The execution rules forces the following execution of the tuning option for each of the available basic database operations:

Database operation	Execution rule	Actual execution point
--------------------	----------------	------------------------

AGRCreTableId	Init	Before table is created
	Before	Before data is inserted into the new table
	After	After table is created and data (if any) is inserted
AGRExecSqlId	Init	Before query is executed
	Before	Before query is executed
	After	After query is executed
AGRCreUpdateId	Init	Before temporary table is created
	Before	Before update
	After	After update
AGRUpTableId	Init	Not relevant. Shall not be used.
	Before	Before update
	After	After update
AGRSelIntoId	Init	Before query is executed
	Before	Before query is executed
	After	After query is executed
AGRSelLoopId	Init	Before the main select statement
	Before	Before the first query inside cursor is executed
	After	Before the second query inside cursor is executed
AGROpenCur	Init	Option is to be executed as early as possible, before the actual execution of the query.
	Before	For multi step operations, the option will be executed after the first step.
	After	After AGRCloseCur is called. I.e. LOGQUERY option will only log "Close Cursor", not the AGROpenCur query.

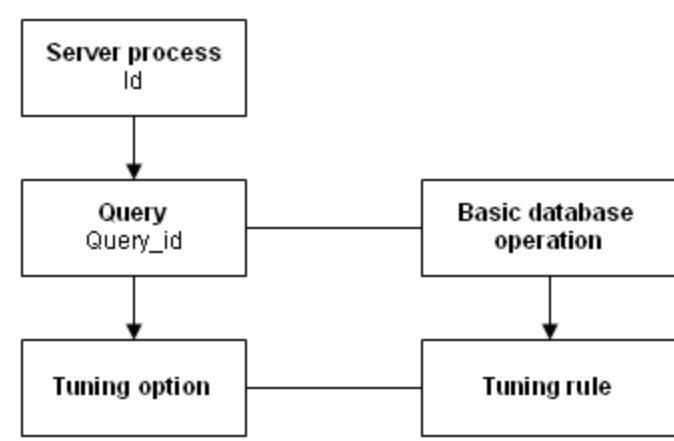
Tables used for tuning setup and modifications

The standard setup for the Agresso server processes is stored in the table asysqueryopt, delivered as part of the Agresso installation package. This table contains a limited number of tuning actions.

All tuning actions created for a specific customer are stored in the table aagqueryopt. Customer specific tuning will usually survive Agresso upgrades.

Tuning data model

The following diagram summarises the elements so far identified as part of the tuning process:



The Server Process Optimization window

Window description

The Server process optimisation window is the primary tool to use for server process tuning. An example of the window is shown below, with default tuning for the LG04 server process, variant 0:

Window example

An example of the Server process optimization window is shown below:

(AGRESSO DEMO) Server process optimisation									
	Report	LG04	Variant						
#	Default	Query	SeqNo	Position	Description	Options	Argument	Table	Data
?	<input type="checkbox"/>								
1	<input checked="" type="checkbox"/>	LG04_HV0010	0	After		INDEX	order_id	0	All datab
2	<input checked="" type="checkbox"/>	LG04_HV0010	1	After		STAT	compute statistics	0	Oracle
3	<input checked="" type="checkbox"/>	LG040030	0	After		INDEX	order_id	0	All datab
4	<input checked="" type="checkbox"/>	LG040030	1	After		INDEX	apar_id,order_id	0	All datab
5	<input checked="" type="checkbox"/>	LG040030	2	After		STAT	compute statistics	0	Oracle
6	<input checked="" type="checkbox"/>	LG040040	0	After		INDEX	article_id,order_id,client	0	All datab
7	<input checked="" type="checkbox"/>	LG040040	1	After		INDEX	order_id, line_no	0	All datab
8	<input checked="" type="checkbox"/>	LG040040	2	After		INDEX	apar_id,order_id	0	All datab
9	<input checked="" type="checkbox"/>	LG040040	3	After		STAT	compute statistics	0	Oracle
10	<input checked="" type="checkbox"/>	LG040070	0	After		INDEX	order_id	0	All datab
11	<input checked="" type="checkbox"/>	LG040070	1	After		STAT	compute statistics	0	Oracle

Explanation of fields

Top row: The top row contains drop-down lists for the server process (labelled Report) and variant number you want to optimise.

Table columns: The table columns contain tuning details for the various queries. These are explained in the following table:

Column name	Description
Default	A flag indicating whether this row is part of default tuning or not.
Query	The query id identifying a specific query associated with the server process.
SeqNo	<p>The sequence number of the tuning option. When a specific query id is repeated in several rows, the options are executed in the following order:</p> <ol style="list-style-type: none"> 1. All options with Init as execution rule (in the Position column), then ordered by their sequence numbers. 2. All options with Before in the position column, ordered by sequence numbers. 3. All options with After in the position column, ordered by their sequence numbers.
Position	<p>The execution rule to apply. See previous pages. Note that rules are implemented according to the basic database operation the query performs.</p>
Description	A description of the tuning option.
Options	The selected tuning option. See previous pages.
Argument	The argument needed for the selected option. Descriptions are given below.
Table	A number identifying the table in the query.
Database	The selected database to which the tuning entry applies.

Agresso in a Terminal Server (Citrix) Environment

Introduction

The basic functionality of the Agresso Smart Client has been tested on Microsoft Windows 2008 R2 Remote Desktop Services with Citrix XenApp 6.

We have found no particular problems specific to Agresso, but, since not everything has been tested thoroughly, there might be issues that require special attention (for example: use of local resources such as printers and scanners).

Installation options

Server recommendation

The requirements for running the Smart Client from a Terminal Server, are the same as for running the Smart Client on a PC.

Alternatives

You can either install the Agresso Smart Client locally on the server or use an existing centrally configured client.

Installing Smart Client (locally) from the DVD

Recommendation

We recommend that you install Agresso Smart Client from the console while the server is in [Install mode](#) and no remote users are logged in.

When the client is installed and the data source has been configured using [Agresso Client Configuration](#), you should start up the Smart Client and check that it performs properly. If no problems occur, set the server back to [Execute mode](#).

Installing by using Centrally Configured Client

If you already have the Agresso components installed on a server, you can use a centrally configured client for running Agresso on the Terminal Server. A few issues must be considered, though:

- The Agresso client packages that are required on local PC's will also be required on the Terminal Server.
- We recommend that the centrally configured client files are copied from your Agresso server and placed on a local drive on the Terminal Server before executed by users. This will ensure optimal performance. Running the centrally configured client from a shared folder on the network is also possible.

Agresso and Citrix Presentation Server

If your Terminal Server runs Citrix Presentation Server, you will have additional options for distributing the installed application to end users. Agresso supports Citrix Published Applications, and has been tested with the Access Suit WEB Interface (previously called Citrix NFuse) and Citrix Secure Gateway for access over Internet.

Oracle syntax for correlated updates

5.5 improvement

The lack of join syntax in traditional Oracle updates has been the cause of much tuning effort. Agresso 5.5 is using a new syntax for updates with join that seems to solve this problem. Testing has shown a significant speed improvement for many server jobs.

With this syntax Oracle is performing updates similar to SQL Server both when it comes to performance and function.

It is possible to have an ambiguous replace. If the join gives two or more different records that match the one to be updated, you will get no error and thus not know which value was used for the update. This is consistent with the behavior of SQL Server.

However, the normal business logic in Agresso prevents this kind of duplicates.

Sample Code

Version 5.4:

```
UPDATE HGL1933

SET ( HGL1933.description1 ) = (
    SELECT a.description
    FROM agldescription a
    WHERE a.attribute_id = 'A0' AND a.dim_value = HGL1933.dim1 AND
        a.client = 'EN' AND a.language = 'EN' )

WHERE EXISTS (
    SELECT a.description
    FROM agldescription a
    WHERE a.attribute_id = 'A0' AND a.dim_value = HGL1933.dim1 AND
        a.client = 'EN' AND a.language = 'EN' )
```

Version 5.5:

```
UPDATE /*+ bypass_ujvc*/
(SELECT a.description adescription , h.description hdescription
FROM Hagr56_2_GL120003 h , agldescription a
WHERE a.attribute_id = h.attribute_id
AND a.dim_value = h.dim_value
AND a.client = 'EN'
AND a.language= 'EN' )
SET hdescription = adescription
```

The hint used in the new syntax /*+ bypass_ujvc*/) is needed.

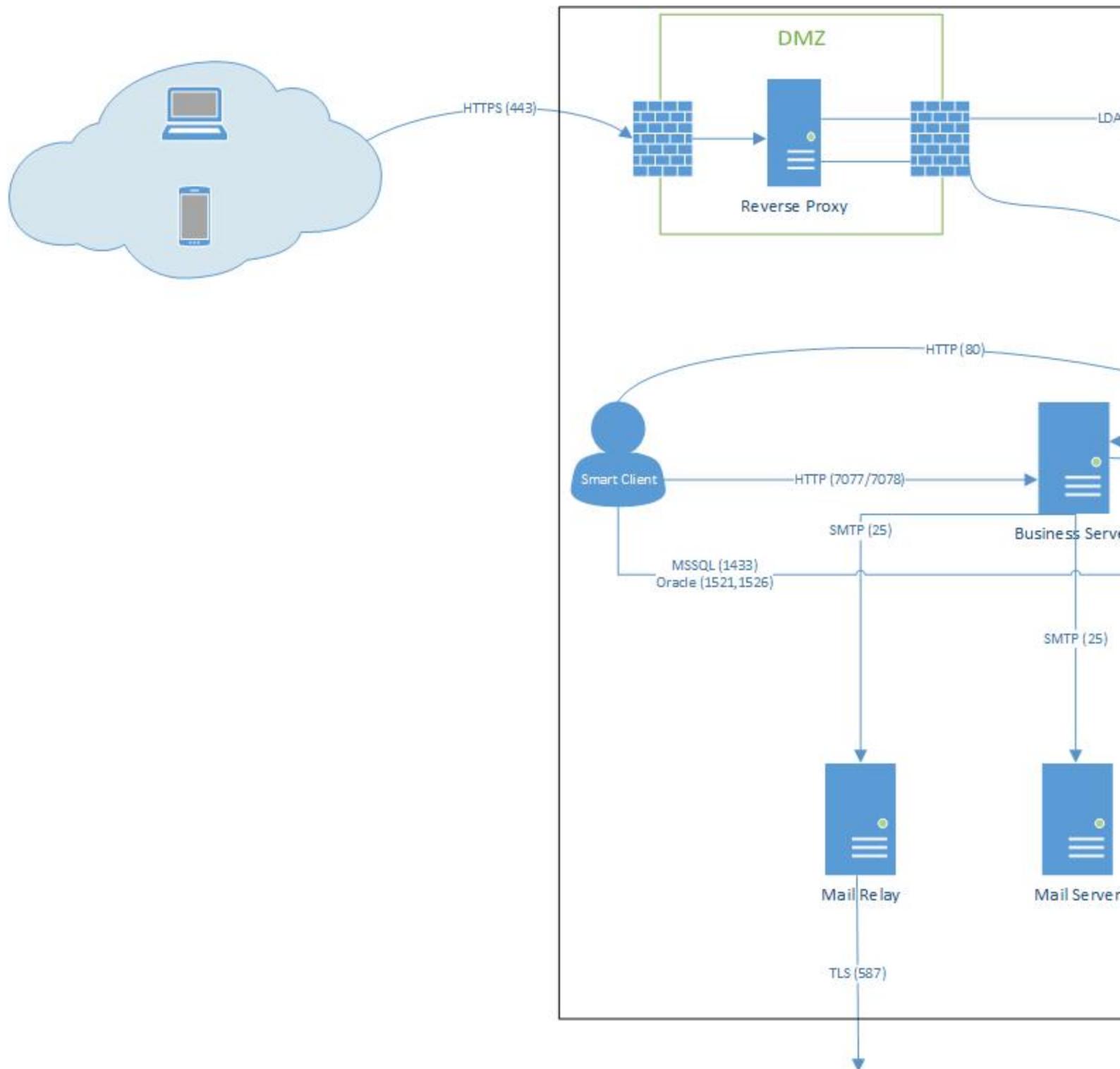
Oracle 11.2 (The bypass_ujvc hint is not supported in Oracle 11.2) :

```
MERGE INTO Hagr56_GL120003 h
USING (SELECT a.description AS description, h.rowid xzfd_rid
FROM agldescription a, Hagr56_GL120003 h
WHERE a.attribute_id = h.attribute_id
AND a.dim_value = h.dim_value
AND a.client = 'EN'
AND a.language = 'EN' ) xzfd_t
ON (xzfd_t.xzfd_rid = h.rowid)
WHEN MATCHED THEN
UPDATE SET h.description = xzfd_t.description
```

Typical Agresso network topology

Overview

The diagram outlines the communication between server and client components in a typical Agresso installation:



Tools Overview

Three tools

There are three main tools supplied by Agresso to manipulate the database during an installation, and later during maintenance and database upgrades:

- Agresso Copy
- Agresso SQL
- AgrIndex

Agresso Copy

[Agresso Copy](#) is used to copy table definitions and data between databases and database management systems (RDBMSs).

There are an Agresso Copy program for each supported database. If using Oracle database, the name is `copyora.exe`, while the MS SQL alternative is named `copyms.exe`.

! In this Appendix we are discussing the copy command in detail and how it used from the command line. For information on how to use the copy feature from the Management Console, please refer to the [Configuration](#) section.

Agresso SQL

[Agresso SQL](#) is a general purpose SQL interpreter, which is available for all of Agresso's platforms and databases. The Agresso SQL program is called `asql.exe` for all databases.

AgrIndex

[AgrIndex](#) (`AgrIndex.exe`) is a console application used to re-generate indexes for Agresso tables.

Agresso Copy

Functionality

Specialised tools

The Agresso database copy programs, `copyora` and `copyms` and are standalone programs used to load tables into a new Agresso database and to move tables from one database to another (for example during an upgrade).

To copy data between two Oracle databases, you may also use the Oracle exp/imp utilities. However, in order to move data between different database management systems, you need to use the Agresso copy programs.

Furthermore, in some cases these programs are easier to use than the native tools. For instance, the Agresso copy programs will delete all tables by default before they are re-created and the data is loaded. Using native Oracle import, you must delete the tables manually before running import.

`files.lst`

The copy programs are used directly from the command prompt. They will copy table definitions and indexes to and from ASCII files. A file named `files.lst` controls which tables are copied. Each table copied out of a database has its table definition, index definitions and data placed in a separate file. The `files.lst` file contains the names of all these files.

By default, the copy programs drops and re-creates all tables in the control file list. Make absolutely sure that your *files.lst* is correct before issuing a **copy in** command. The *files.lst* in the current directory is always used, unless **Agresso Copy** is told otherwise through command line parameters.

Views

The copy programs can also create views in the database. The view definitions must be in the table *aagview* in the same database. Use the **-v** flag during **copy in** to create views.

Note: During **copy out** the **-v** flag has no relevant effect.

Examples

Task: You have copied out a complete database, but want to copy only the tables relevant to the Agresso Fixed Assets module back in (afa* tables).

Solution: You edit *files.lst*, and removes all lines not starting with *afa*.

Task: Copy in all the tables for the Contract module (acn* tables).

Solution: If you have saved the original *files.lst*, you can edit this file and make a new *files.lst* as described above.

You can also type `DIR /B acn*asq > files.lst`. This command will create a new *files.lst* containing only acn* entries.

Syntax

General

The most common syntax for **Agresso Copy** for the different databases is:

MS SQL: `copyms.exe -d<direction> -U<user> -P<passwd> -S<odbc datasource> [-z -v] [<tables>]`

Oracle: `copyora.exe -d<direction> -U<user> -P<passwd> -D<database> -S<tnsname> -z -v [<tables>]`

Example: If you shall copy all Agresso tables from MSSQL, database *agresso*, to Oracle, you write:

```
copyms.exe -dout -Uagresso -Pagresso -S<odbc datasource> a%
```

```
copyora.exe -din -Uagresso -Pagresso -Dagresso -S<tnsname> -z -v -a1000
```

For **copy in**, no table names should be specified. The tables (files) listed in *files.lst* will be copied.

List all options

In order to get an overview of all available copy options, enter the following at the command prompt:

```
copyms -? | more
```

The -a parameter (Oracle)

The **-a** parameter is very important for Oracle. If this parameter is set to a large number, e.g. 5000, the program will copy 5000 rows at a time, consuming lots of memory, but speeding up the copy process. The **-a** option is only relevant for Oracle databases.

The -z parameter

The **-z** parameter tells the database to create procedures and functions from the *asysddl* and *aagddl* tables.

Null values

In general, Agresso tables are defined with the `not null` and `default value` options for every column. Allowing null values in a table is not acceptable in Agresso, and the application will fail.

Agresso Copy Syntax

There is one copy program for each database. If using Oracle database, the name is `copyora.exe`, and if using the MS SQL database, the program is called `copyms.exe`. These programs have the same functionality. Every table copied in or out has its own data file associated with it (990+ files when copying a full Agresso database). The data file is an ASCII file and can be edited by the user (e.g. change table name/column name/location, remove/change index, etc.). Tables with blob will create one directory for each table as the name of the table, and all blob data will be in that directory.

Agresso Copy is a part of the application server software in Agresso Business World. The program is started from AGRESSO_EXE (\agresso\server), in command prompt window.

The copy programs have a lot of parameters, but the most common syntax for Oracle is:

```
copyora -dout -U<user> -P<password> -S<dbserver> <table names>
copyora -din -U<user> -P<password> -S<dbserver> [-A] [-v] [-z]
```

The `-D` parameter is not necessary for Oracle. For Oracle the `-S` is used for TNS-names. SQL Server use the `-S` parameter for an ODBC data source.

The parameters can be set in any order, but the table names must be entered after the last parameter. Notice that no table names are usually entered when data is copied into a database (table names are retrieved from the `files.lst` file). It is possible to supply a list of table names for "copy in" as well, in which case only these tables will be loaded and not all the tables listed in `files.lst`.

You can list as many table names as you want (restricted only by OS). Table names are separated by white space. Valid wild cards are (1) underscore (match exactly one arbitrary character), and (2) percentage sign (match any number of arbitrary characters).

! In UNIT4 Agresso there is a new set of tables used by Agresso Invoice Manager. These tables use different database settings, and must be copied separately. These tables start with a "c".

Data files

Regardless of which database the data was copied out from, the data files all look the same. They contain the name and location of the table, column definitions, etc., all divided into sections.

The name of the data file consists of the first eight characters of the table name, a punctuation mark and a three digit number. If `agltrans` is the 32nd table to be copied out, the data file will be called `agltrans.031` (starts from 000).

Table section

[TABLE]

name=acsgroup

location=agrstatic

This first section has two parameters; name and location.

The location parameter is only used if the `-T` flag is set (see description of parameters to copy programs). If `location=default` the parameter is also ignored.

Columns section

[COLUMNS]

```
account,char,,8,,0,1,,  
apar_gr_id,char,,2,,0,1,,  
....
```

Each line in the columns section consists of five values:

- name (beware of case sensitive databases)
- type (char, varchar, date, bool, int1, int2, int, float, money, raw)
- length
- null values
- default value

The length value is only used for char or varchar columns. The other lengths can be ignored. When copying data from an Informix table, a money column may have a length of 8000+. This is normal.

If the length is larger than 255 (or = 16), it will be ignored for the following columns:

- authorize (which will be given length 1500)
- col_09 (which will be given length 700)
- col_10 (which will be given length 500)
- col_11 (which will be given length 500)
- date_string (which will be given length 366)
- info (which will be given length 366)
- q_param (which will be given length 1500)
- q_where (which will be given length 400)
- query (which will be given length 1600)
- text (which will be given length 720)

If columns are *not null* and all columns have default values (apart from special tables in Agresso Invoice Manager):

- char, varchar: default '';
- bool, int1, int2, int: default 0
- money, float: default 0.0
- date: default Jan 1 1900, 00:00:00

Index section

[INDEXES]

```
iacsgroup1, agrindex,1,apar_gr_id, apar_type, client
```

The index section consists of one line for each index. The first value is the index name and must be less than 18 characters. The next value is the physical location of the index. The third value is a unique flag, and the rest of the values are the names of the columns included in the index.

Parameter section

[PARAMETERS]

```
col_sep=,
```

```
line_sep=[EOLN]
```

```
string_quote="
```

This section informs the copy program how the data is stored. Do not change anything here.

Data section

[DATA]

```
"2010","3","P",0,"NO",0.000000,"NOK",0,"Random suppliers",0,  
" ",1,"NO",19980312 01:07:00," ",0,"1580"," ",0," ",0,"30",0,"sysno",
```

If deleting everything in front of this section, you will have a clean data file which can be used in other programs (e.g. Excel). If you change something in the data section, make sure that the changes do not conflict with the column definitions. Note that there is a comma at the end of each line of data.

files.lst

There is a special file created by Copy out, called *files.lst*. This is the first file read by Copy in. It contains a list of all the data files in the directory.

If you have a directory containing data files for all the tables in an Agresso database, but only need three tables, you can change *files.lst* so only the data files for these three tables are listed here. The copy program will then ignore the rest of the data files in the directory. This can also be done by using a "dir" command to create a new *files.lst*:

```
dir /b a*.* > files.lst
```

Make sure that there are no empty lines or special characters in *files.lst*. This may stop the program prematurely.

Agresso Copy Parameters

Mandatory parameters

Parameter descriptions

When starting the program with no parameters, you will get an overview of the syntax for Agresso Copy. Remember that there should be no space between a parameter identifier and its value.

The following table gives complete list of the mandatory parameters. Note that the parameters must come in the sequence listed below.

Parameter	Description	Example
-d	States copy direction, in to database or out.	<i>-din</i>

		-dout
-U	Login name. Specifies the login name for the user. When using MS SQL as the database, make sure that the login name is using SQL Server Authentication.	-Uadminuser
-P	The password for the user logging in. Make sure the login specified by the -U parameter is using SQL Server authentication, and that the password is correct.	-PmyPassw1
-D	MS SQL only: The login name specified with the -U parameter always have a default database it connects to. The -D parameter is only used if the login name has access to more than one database, and you want to copy from/to a different database than the default.	-Dagr55
-S	States logical server name Oracle: The database logical server name. Copyora needs a valid net service name to connect to the database. MS SQL: The ODBC data source.	-Sagr55
<tables>	List of tables to be copied. You can list as many tables as you want (limited by the OS). You can also use % (any strings) and _ (any character).	a% (will copy all tables beginning with a)

Example

The following command will

Optional parameters

The following are a complete list of the optional parameters with syntax and description:

» Append

Syntax: **-A**

This parameter instructs the copy program not to re-create tables before data is read into it. Make sure that the table definition in the database corresponds to the data file. If not, it results in an error message for each row.

» Array size (batch size)

Syntax: **-a<size>**

This parameter is only used by Oracle, not MS SQL. The purpose of the "-a" parameter is to tell the copy program how many lines it should insert at a time. The larger the value, the faster the program works. However, if this value is too large, you will eventually run out of log or memory. Default is 100. When this value is set to zero, all rows are inserted one at a time (slow).

» Maximum number of blob files per directory

Syntax: **-B<size>**

Blobs in a table are copied into separate files called i000000000000n.raw, where "n" is a row counter. Depending on the disk and the OS, there is a limit to how many files a directory can hold. The copy program will spread the files into separate directories if there are more than 32000 rows in the table. 32000 is the default, but can be changed using this variable.

Where flag

Syntax: `-b<0|1>`

This flag is used in conjunction with the "-q" flag. If a "where" statement is used when copying out data, you can decide whether or not you want it stored in the data file with the data. It can then be used to replace data when copying into a database. Valid values are:

- 0: Do not add the "where" statement to the data file when copying out, or, do not use the "where" statement when copying in. This is the same as not setting it at all.
- 1: When copying out, the "where" statement is added to the data file. When copying in, data in the table where the "where" statement is true, is deleted. The data in the data file is then added. The table is not re-created, so datatypes and lengths of columns must match the datafile.

Column separator

Syntax: `-c<char>`

This parameter is used when you want to change the column separator. It is only used when copying data out. If you are copying Agresso tables, using this parameter is not recommended. In addition to a character, [TAB] and [EOLN] are valid values.

Default values

Syntax: `-e0`

All the columns are created without default values. If this parameter is not set, character columns will have an empty string as default, numeric columns will get "0" as default, and date columns will get "Jan 1 1900" as default.

File directory

Syntax: `-f<path>`

Unless this parameter is used, all the data files are created/read from the current directory (the directory you are calling the program from). If the directory does not exist, the program will try to create it. A path can be given relative to the current directory.

Display only

Syntax: `-g`

Only used with parameter "-din". The create and drop commands are echoed to "stdout". No tables are dropped or created and no data is read. If the result from the copy command is redirected to a file, the file can later be run from within interactive SQL. Views are created regardless of this parameter.

Name of list file

Syntax: `-h<file name>`

Unless this parameter is used, a file called files.lst will be created in the same directory where your data files are. This file is a listing of all files which are going to be read. Notice that the file cannot contain path names or any information other than file names (e.g. comments, etc.).

Index flag

Syntax: `-i[0|1]`

You can ignore the index section by using "-i0". If you only want to ignore the unique flag, use "-i1".

Index location

Syntax: `-I[<location>]`

If this parameter is used without any value, the index location parameter in the data file is used. If you supply a location, all indexes are created in this location (dbspace, tablespace or segment).

Progress indicator

Syntax: `-j<rows>`

The program will print a dot to stout for every `<rows>` row copied.

Line separator

Syntax: `-l<char>`

This parameter should not be used. The default line separator is newline and should not be changed.

Identity column

Syntax: `-m`

Used by MS SQL only. When this parameter is set, the identity column (agrtid) is not added when the table is created.

Null values

Syntax: `-n`

Use this flag if you want to allow "null values" in the tables. Agresso standard tables should not be allowed to have "null values". However, special tables for Agresso Invoice Manager must allow "null values".

Column names

Syntax: `-o<col_names>`

Only tables containing a column name in the list of column names are retrieved.

Original datatype

Syntax: `-O`

Datatype text in MS SQL Server will be converted to varchar2(4000) in Oracle. This should only be used when copying special tables for Agresso Invoice Manager.

Where statement

Syntax: `-q<string>`

You can limit the amount of data copied out by using this parameter. Supply an SQL "where" clause enclosed in double quotes.

Automatic ANSI to OEM conversion check

Syntax: `-Q`

If you run the SQL Server Client Network Utility, you will find a setting called "Automatic ANSI to OEM conversion" under the "DB-Library information" tab. If this is checked (turned ON), it may cause problems when copying text containing characters that are not part of the English alphabet (like some Scandinavian characters). The copy program will automatically check this setting to make sure it is OFF. Use "`-Q`" if you do not want to perform this check.

Trigger warnings

Syntax: `-r`

If the table being copied contains a trigger, the definition is printed to stdout. When copying into a database, the trigger is not re-created.

Raw (blob) data file name

Syntax: `-R<column_name>`

When tables containing blobs are copied, every blob is stored in separate files called `i00xxxx.raw`. If your table contains `file_name` information for every blob, you can use these when the blob is copied out. Give the column name containing the file information as parameter `-R`. If the `file_name` is blank, or there are duplicates, the default `i00xxxx.xxx` is used (it will only keep the file extension).

String quote

Syntax: `-s<char>`

In the data file, all strings are enclosed in double quotes. If you want to use single quotes or something else (maybe no quotes at all), use this parameter.

Table location

Syntax: `-T[<location>]`

If this parameter is used without any value, the location parameter in the data file is used. If you supply a location, all tables are read into this location (dbspace, tablespace or segment).

Unicode

Syntax: `-u`

Tables is by default created using "char" and "varchar". The parameter "`-u`" results in tables created with types "nchar" and "nvchar".

View

Syntax: `-v`

Creates database views. Only used with "copy in". View definitions are read from the table `aagview`.

Extended copy

Syntax: `-x`

Invoice Manager tables must be copied with special parameters (see section [Parameters used with Invoice Manager tables](#)) that are not used for Agresso core tables. Tables starting with "c" is treated as Invoice Manager tables. If for some reason you do not want to treat tables starting with "c" as special tables, use "`-x0`".

Native Copy Programs

Native programs

All the database servers can copy data to and from text files, but there is no standard way of doing this, and there is no standard format for the files. Usually you have to do the copy operation in several steps.

For the database servers we are using, the native programs listed below can be used. These programs have many parameters and some need different types of parameter files.

In order to copy data between Agresso databases more easily, Agresso Copy was made.

Oracle

Programs: exp, imp, sqlloader.

Advantages: Fast, can handle one or more tables, very powerful ('r;everything' is possible).

Disadvantages: Does not drop tables, Difficult to read/write 'r;foreign files'

MS SQL Server

Program: bcp.

Advantages: Fast, easy to use.

Disadvantages: Does not drop tables. One table at a time. A special database option must be set.

Agresso SQL Program Overview

Agresso SQL Program (asql.exe)

This program executes SQL statements. You can use either Agresso SQL syntax or native database syntax. The statements are stored in an ASCII file with **.asq** file extension. This file is referred to by the parameter (-F) at the command line when running the program.

! This section is discussing detailed information on ASQL program and syntax. For information on how to use Agresso SQL Script from the Management Console, please refer to the document [Agresso SQL Scripts](#) in the Configuration section.

Connection parameters

The following is a description of the input parameters and the Agresso SQL syntax. Some of the input parameters, called the *connection parameters*, vary from platform to platform. Parameters common to all platforms are called *input parameters*.

-D<Agresso Data Source>

The -D parameter refers to an Agresso data source. Prior to running asql.exe, a data source must be set up. This can be done using the Agresso Management Console. The -D parameter should not be used if you use the parameters below for alternative connection.

Alternative connection parameters

-U<Database User>, -P<User Password> and -D<Database Name>

The parameters below should only be used in special circumstances where you are unable to use an Agresso Data Source.

! Agresso environment variables cannot be used unless you connect with an Agresso Data Source.

Under special circumstances asql is used when there is no Agresso Data Source, e.g. Agresso Budget (Budwin). If you are not using an Agresso Data Source, there are several parameters you can use instead of -D:

ODBC/MS Sql Server: These parameters are optional. If not given, database will be the default database connected to the login name.

! If you change default database on the odbc data source (-S below), this will have no effect here. It will still be the default database from the login name that is used.

Oracle: These parameters are not in use in an Oracle environment.

-S<Native Source>

ODBC/MS Sql Server: ODBC data source (-D<database name> is optional).

Oracle: Oracle Net Service Name (-D<database name> is not used).

»-I<Interface>

Specify RDBMS type (default value is ODBC):

- ODBC
- ORACLE
- NATIVE (If you select NATIVE, you must specify Agresso database driver with d<agrdb.dll>)

»-d<Db driver>

Only used in conjunction with the parameter INATIVE. In that case, you must specify and Agresso database driver, e.g. agrodbc.dll or agora.dll.

General input parameters

The following general input parameters are available:

»Path to asq files, -f

This parameter is not required. If the asq files are not in the current directory, you can use this parameter to tell asql.exe where the files are stored. Normally this parameter is used in conjunction with the -h parameter.

This parameter will override the environment variable AGRESSO_SCRIPT.

If the -h parameter is used and the list file contains path names, this parameter is ignored.

»Name of list file, -h

A list file is a file containing the names of one or more asq files. When using the -h parameter, asql.exe will run all the asq files listed in the list file. The files can be entered in the list file with either full or relative path. If the path is given, it will override the -f parameter and AGRESSO_SCRIPT.

This parameter will override the -F parameter (name of asq file).

»Name of asq file, -F

The name of the file containing the SQL commands to be executed. This file will normally have the file extension asq. You can enter the full path name. If no path name is entered, asql.exe will first look in the directory specified in the environment variable AGRESSO_SCRIPT. If this environment variable is not defined, it will look in the current directory.

This parameter is overridden by the -h parameter giving the name of a list file.

»Exit flag, -x

This parameter is only used when running more than one asq file (i.e. -h parameter is used). It is used in conjunction with the "on error exit" command. The "on error exit" command's scope is only the current file. Thus, if an error occurs after this command, asql.exe will skip the rest of the current file, but continue with the next one. However, if you want asql.exe to cancel all files when an error occur, this parameter is used.

»Verbose flag, -v[+]

All commands and information are printed to a log file. This file is called asql.log, and is found in the directory specified in the AGRESSO_LOG environment variable. If this environment variable is not defined, the log file will be created in the current directory.

Every time you run asql.exe with the parameter -v, the current log file is replaced with the new one. If you don't want to delete the old file, but append to it, use the + switch.

Environment variables

The following environment variables are used by asql.exe:

» AGRESSO_IMPORT

This environment variable is used by the copy statement and points to the directory where the data file is:

» AGRESSO_LOG

The directory (full path) to where the log file is created. If not set, current directory is used.

» AGRESSO_SCRIPT

The directory (full path) to where the asq file(s) are found. If not set, current directory is used.

Agresso SQL Syntax Overview

ASQL

Agresso SQL is an SQL variant tailored to facilitate Agresso specific database operations. ASQL is translated into the native SQL syntax of the host database when executing.

Native Database SQL can be enabled by prefixing the SQL with the [DATABASE keyword](#) or by using [database blocks](#).

ASQL features include:

- Rich set of functions
- Database independent data types
- CREATE TABLE FROM SELECT with any host database
- Importing data from text files

ASQL does not support:

- Outer joins
- NULL values

Data definition language (DDL)

The following commands are used to create and remove tables and views:

Command	Description
CREATE TABLE	Creates a new table in the database
CREATE TABLE AS	Creates a new table in the database based on succeeding query
DROP TABLE	Removes a table from the database. The table definition will disappear
CREATE VIEW AS	Creates a view from parts of one or more tables
DROP VIEW	Removes a view from the database.

The table definition will disappear

Data manipulation language (DML)

The following commands are used to retrieve, insert, delete and update rows in a table.

Command	Description
SELECT	Retrieves data (rows) from a table
INSERT	Inserts new data (rows) into a table
UPDATE	Changes existing data (rows) in a table
DELETE	Deletes data (rows) from a table
MERGE [INTO]	Merges content from two tables
COPY	Copies rows between a table and text file

Index commands

An index is a table of pointers. Each row in the indexes points to a corresponding row in a data table. Unique indexes are used to ensure that a table contains no duplicate row.

Command	Description
CREATE UNIQUE INDEX	A column or combination of columns in a table that uniquely identifies each row in the table. (primary index)
CREATE INDEX	Defines an index on a table (secondary index)
DROP INDEX	Removes an index

Comments

Comments can be added anywhere in an asq-file, even inside statements. However, they can not be used inside strings. Comments are in the /* */ form and will be ignored when reading the file.

Sample code

```
UPDATE mydescription m
  FROM description d
    SET m.description = d.description
  /*
   , m.amount = fmulf(m.amount, abs(d.mult))*1.1
  */
  WHERE m.account = d.account
  AND m.client = d.client
```

The amount column will not be updated.

Log messages

Writelog

You can use the WRITELOG keyword to improve the messages in the log file.

Sample code

Example A:

```
insert into mytable (col1, col2, col3)
select col1, col2, 0 from myothertable
/
```

Example B:

```
writelog Inserting into :tablename
insert into :tablename (col1, col2, col3)
select col1, col2, 0 from myothertable
/
```

In the example A, you get the following in the log file:

```
13:02:01 2 > Execute statement
```

In example B, you get:

```
13:02:01 2 > Inserting into mytable
```

Notice that you can use variables directly in your log message.

Print

The PRINT statement can be used to put text into your log file. The string can contain variables.

Sample code

```
print :count items found when looking
/
```

Data Types and Functions

Data types

The following data types can be used to define columns in the tables that you are going to create in the database. The CONVERT function only recognises da of the types CHAR | DATE | FLOAT | INT | MONEY.

Type	Length	Description
char	1-255	Fixed length character string. Length normally <= 16
vchar	1-4000	Variable length character string
bool	N/A	Normally used as a flag. Valid values are 0 or 1
int1	N/A	Small integers. Valid values from -255 to 255
int2	N/A	Medium size integers. Valid values from -32768 to 32768
int8	N/A	Large integer.
int	N/A	Normal integers. Valid values from -2147483648 to 2147483648
float	N/A	A float value. Valid values depend on database server
money	N/A	A money value. Legal values depend on database server. Do not expect a precision of more than 16
date	N/A	A date/date time value
guid	N/A	A global unique identifier. A new identifier can be created with GETGUID()
longtext	N/A	long text columns. varchar(max) on mssql, clob on oracle. Max length: 32000 chars.
raw	N/A	Used to store binary large objects (BLOBS)

Aggregate functions

The arithmetic operators + , - , * , \ can be used.

Function	Description
COUNT(*)	Returns the number of rows in the specified table
COUNT(all distinct expr)	Returns the number of (distinct) non-null values in a column
SUM(all distinct expr)	Adds up the values in a specified column
AVG	Returns the average of all the values in the specified column
MAX	Returns the maximum value that occurs in the specified column
MIN	Returns the minimum value that occurs in the specified column

Numeric functions

Function	Description
ABS(n)	Returns the absolute value of a numeric expression (n)
ROUND(n,x)	n rounded to x dp
MOD(n,b)	n modulus b
SQRT(n)	square root of n
fdiv(expr1,expr2)	Converts expr1 and expr2 to float and divides. Type of the return value is float
fdivm(expr1,expr2)	Same as fdiv, but type of the return value is money
fmul(expr1,expr2)	Converts expr1 and expr2 to float and multiplies. Type of the return value is float
fmulm(expr1,expr2)	Same as fmultiplication, but type of the return value is money

<code>mod(expr1,expr2)</code>	Returns the integer result of expr1 modulo expr2
<code>min(col)</code>	Returns the lowest value
<code>max(col)</code>	Returns the highest value
<code>power(expr1, expr2)</code>	Returns $\text{expr1}^{\text{expr2}}$
<code>sum(col)</code>	Returns the sum of column values
<code>sqrt(expr)</code>	Returns square root of expr

sample

```
PRINT '*** ASQL script sample - math functions ***'
```

```
/
```

```
ON ERROR EXIT
```

```
/
```

```
IF EXISTS asqsamples_math
```

```
/
```

```
DROP TABLE asqsamples_math
```

```
/
```

```
END
```

```
/
```

```
CREATE TABLE asqsamples_math
```

```
(
```

```
    description varchar(256),
```

```
    abs_number int1,
```

```
    round_number int1,
```

```
    mod_number int1,
```

```
    sqrt_number int1,
```

```
    fdiv_number float,
```

```
    fdivm_money money
```

```
)
```

```
/
```

```
INSERT INTO asqsamples_math
```

```
(
```

```
    description,
```

```
    abs_number,
```

```

round_number,
mod_number,
sqrt_number,
fdiv_number,
fdivm_money

)

VALUES
(
'Test sys functions',
abs(-1.23),
round(123.569932, 1),
mod(5,2),
sqrt(8),
fdiv(8,3),
fdivm(8,3)
)
/

```

Conversion

Function	Description
convert(from,to,expr)	Converts a value from one datatype to another. The first two parameters can have one of the following values: char, int, float, money, or date. (In Oracle, money, int and float are the same type.)
int2str(integervalue)	Returns <i>integervalue</i> as a string
to_counter(expression)	Returns <i>expression</i> as an int8.
to_float(expression)	Returns <i>expression</i> as a float.
to_int(expression)	Returns <i>expression</i> as an int.
to_money(expression)	Returns <i>expression</i> as money.

String functions

Function	Description
bloblength(col_expr)	Returns the number of bytes in a blob column.
charindex(search-string, searchfor[, startIndex])	Returns index of first character of <i>searchfor</i> in <i>searchstring</i> . -1 if not found. Optionally, you can start the search at <i>startIndex</i> .

concat(expr1,expr2)	Concatenates one string to another
length(expr)	Returns length of expression
left(expr, n)	Returns the leftmost n characters of a string
lpad(c,length,expr)	Pads the leftmost string expr with c characters. The length of result string is length Note: Padding with blanks will not work since all result strings are trimmed.
lshift(expr,n)	Shifts value of the string expr n characters to the left
lower(expr)	Returns the input string (expr) in lowercase
replace(originalstring, stringtoreplace, replacewith)	Replaces all occurrences of <i>stringtoreplace</i> in <i>originalstring</i> with <i>replacewith</i> .
right(expr,n)	Returns the rightmost n characters of a string
rpad(c,length,expr)	Pads the string expr with c characters. The length of result string is length. Note: Padding with blanks will not work since all result strings are trimmed.
rtrim(expr)	Returns the string expr without trailing blanks
space(n)	Returns a string containing n spaces (max 1800)
squeeze(expr)	Returns the string expr without leading and trailing blanks
substr(expr, start-index, numchars)	Returns a substring of <i>expr</i> starting at <i>startindex</i> and includes <i>numchars</i> characters.
to_char(expression)	Returns the expression as a string. Ref. <i>ToString()</i> i C#.
upper(expr)	Returns the input string (expr) in uppercase

» sample

```
PRINT '*** ASQL script sample - string functions ***'
```

```
/
ON ERROR EXIT
/
IF EXISTS asqsamples_string
/
DROP TABLE asqsamples_string
/
END
/
CREATE TABLE asqsamples_string (
description varchar(256),
concat_string varchar(25),
length_int int,
left_string varchar(25),
```

```
lpad_string varchar(25),
lshift_string varchar(25),
lower_string varchar(25),
max_string varchar(25),
right_string varchar(25),
space_string varchar(25),
squeeze_string varchar(25),
substr_string varchar(25)

)
/
INSERT INTO asqsamples_string
(
description,
concat_string,
length_int,
left_string,
lpad_string,
lshift_string,
lower_string,
right_string,
space_string,
squeeze_string,
substr_string
)
VALUES
(
'Test sys functions',
concat('text1','text2'),
length('length of this string'),
left('this is returned. This is not', 16),
lpad('a', 10, '0123'),
lshift('0123456789', 4),
lower('MAKE ME LOWER'),
right('return to forever', 4),

```

```

space(10),
squeeze(' do not fill in the blanks '),
substr('0123456789', 2, 4)
)
/

```

Misc functions

Function	Description
convert(from, to,expr)	Converts a value from one datatype to another. The first two parameters can have one of the following values: CHAR, INT, FLOAT, MONEY or DATE (In Oracle, MONEY, INT and FLOAT are the same type) The following conversions gives unpredictable results: DATE ?? ? INT, FLOAT or MONEY
ifnull(expr, expr)	Returns the second expression if the first returns null
ascii(character)	Returns the ascii code for a character. ascii('A') returns 65

» sample

```
PRINT '*** ASQL script sample - sys functions ***'
```

```

/
ON ERROR EXIT
/
IF EXISTS asqsamples_sys
/
DROP TABLE asqsamples_sys
/
END
/
CREATE TABLE asqsamples_sys (
description varchar(256),
convert_char varchar(15),
convert_float float,
ifnull_char varchar(15),
int2str_char varchar(15)
)
```

```

/
INSERT INTO asqsamples_sys
(
description,
convert_char,
convert_float,
ifnull_char,
int2str_char
)
VALUES
(
'Test sys functions',
convert(float, char, 12.3456),
convert(char, float, '12.3456'),
ifnull(NULL, 'was null'),
int2str(15)
)
/

```

Date functions

Function	Description
AGRDBTODAY()	Returns date for current day
AGRDBNOW()	Returns the current date and time
cts2day(string)	Converts a string to date and truncates to day (time part is set to 00:00:00)
date2iso(date)	Returns a string 'yymmdd' (substracted from date)
datediff(date1,date2)	Returns a floating point number (date1 - date2). The integer portion is number of days
datepart(unit,date)	Returns an integer containing the specified part of the date. Unit can have one of the following values (date parts): DAY, HOUR, MIN, MONTH, QUARTER, SEC, WEEK or YEAR
datetime2str(adate)	Returns the date value <i>adate</i> to a string ("yymmdd hh:mi:ss")
datetrunc(unit,date)	Returns a date value that represents the input date truncated to the level of granularity expressed in parameter unit. Unit can have one of the following values (date parts):DAY, MONTH or YEAR
dayadd(n,date)	Adds n days to input date and returns the new date. Use negative number of days (n) to subtract

getdate()	Returns current date
secadd()	Adds n seconds to input date and returns the new date.
monthadd(n,date)	Adds n months to input date and returns the new date. Use negative number of months (n) to subtract
date2str(date)	Converts a date to string
iso2date('yymmdd')	Converts a string 'yymmdd' to a date
str2date(stringdate)	Returns a date from a string in the format 'yyyymmdd hh:mi:ss' (as the next)
to_date((stringdate)	Returns a date from a string in the format 'yyyymmdd hh:mi:ss' (as above)
ts2day(date)	Truncates date to day (time part is set to 00:00:00)

» sample

```

PRINT '*** ASQL script sample - date functions ***'

/
ON ERROR EXIT
/
IF EXISTS asqsamples_date
/
DROP TABLE asqsamples_date
/
END
/
CREATE TABLE asqsamples_date (
description varchar(256),
agrdbnow_date date,
getdate_date date,
now_date date,
cts2day_date date,
date2iso_date date,
datediff_float float,
datepart_int int2,
datepart_date date,
dayadd_date date,
monthadd_date date,
ts2day_date date,
agrdbtoday_date date,

```

```

date2str_str varchar(25)
)
/
INSERT INTO asqsamples_date
(
description,
agrdbnow_date,
getdate_date,
now_date,
cts2day_date,
date2iso_date,
datediff_float,
datepart_int,
datepart_date,
dayadd_date,
monthadd_date,
ts2day_date,
agrdbtoday_date,
date2str_str
)
VALUES
(
'Test date functions',
AGRDBNOW(),
GETDATE(),
NOW,
CTS2DAY('12/12/79'),
date2iso(to_date('20080731 01:02:03')),
datediff(to_date('20080731 01:02:03'), to_date('20080730 01:02:03')), /* will return 1 - one day*/
datepart(sec, to_date('20080731 01:02:03')), /* will return 3 - three seconds */
datetrunc(day, to_date('20080731 01:02:03')), /* will return 2008, rest of the day will be */
dayadd(2, to_date('20081231 01:02:03')),
monthadd(2, to_date('20081231 01:02:03')),
ts2day(to_date('20080731 01:02:03')),

```

```

AGRDBTODAY(),
date2str(to_date('20080731 01:02:03'))
)
/

```

Guid functions

Function	Description
getguid()	Returns a new guid.
guid2str()	Returns a string representation of a guid.
to_guid(guidasstring)	Returns a guid based on a formatted string value.

sample

```

/* ASQL script sample - Test guid functions*/

on error exit
/

if exists asqsamples_guids
/

DROP TABLE asqsamples_guids
/
end
/
CREATE TABLE asqsamples_guids (
description varchar(256),
my_guid guid )
/
INSERT INTO asqsamples_guids (description,my_guid) VALUES ('Test getguid function', {fn GETGUID()})
/

```

Ranking functions

Asql supports two ranking functions:

```
DENSE_RANK() OVER([< partition_by_clause >] <order_by_clause >)
```

Returns the rank of rows within the partition of a result set, without any gaps in the ranking. The rank of a row is one plus the number of distinct ranks that come before the row in question.

```
ROW_NUMBER() OVER ([<partition_by_clause>] <order_by_clause> )
```

Returns the sequential number of a row within a partition of a result set, starting at 1 for the first row in each partition

Macros

Function	Description
MAX_DATE	Highest acceptable date (20991231 23:59:59)
MIN_DATE	Lowest acceptable date (19000101 00:00:01)
NO	0 (FALSE)
NOW	Current date and time (date type - not char)
TODAY	Current date (at 00:00:00) (date type - not char)
YES	1 (TRUE)

Native Database SQL

Options

If you want to use native SQL syntax within ASQL, you have two options:

- use the keyword `database` before the actual SQL statement. The Agresso parser will keep the statement as it is, and not try to translate it from ASQL to native code.
- use database blocks to delimit a set of SQL statements which the Agresso parser shall ignore. A database block is identified by the keywords begin and end:

DATABASE keyword syntax example

If you need to create a sequence in ASQL, this is not supported by ASQL. You write the following instead:

```
DATABASE CREATE SEQUENCE mysec
```

where the CREATE SEQUENCE statement will be resulting (parsed) SQL statement.

Database Block syntax example

Database server type

A database block is delimited by the keywords BEGIN ... END. Immediately after BEGIN, you must identify the database server type. This type must correspond to the actual server(!) and can be one of the following:

- ORACLE
- SQLSERVER
- SQLSERVER7

Example

The following examples illustrates the use of database blocks, without actually displaying any native code:

Oracle:

```
begin oracle  
  <native SQL>  
end
```

MS SQL:

```
begin sqlserver  
  <native SQLSERVER SQL>  
end
```

If ... Blocks

IF [NOT] ... END IF

IF blocks are typically used to create tables that don't exist, drop them if they do exist, or alter them if a column is missing or defined wrong. But you can also use an IF block in a general way.

Syntax

There are two main syntax variants:

Table manipulation syntax:

```
if [not] exists <table> [<column>] [constraint]
```

```
<statements>
```

```
end if
```

General syntax:

```
if [not] SELECT < ... >
```

```
<statements>
```

```
end if
```

Table manipulation

When using EXISTS, The IF statement will check if the table or column exists, and run the statements if the condition is true. You can also use a select statement, instead of an object.

! Checking on constraints should only be used for SQL Server. You can use the key word "constraint" if you want to enter the "if" clause if a constraint exists on the column, but you don't know the name.

Examples

```
if not exists testtab
/
create table testtab (col1 int, col2 char(10))
/
end if
/
...
if exists testtab col1 constraint
/
alter table testtab drop col1 constraint
/
end if
/
...
if exists testtab col1
/
database alter table testtab drop col1
/
end if
/
```

General use of IF ... blocks

When used without EXISTS and <table>, you use the IF to determine actions on basis of what the SELECT returns (null or zero rows = false)

Example

```
IF NOT SELECT id from tab where id = 3  
/  
INSERT INTO tab (id, col) VALUES (3, 'value')  
/  
END  
/
```

If there is no rows with id = 3, the statement will insert a new row into the table tab.

Conditional functions: CASE

CASE

CASE returns a value which can be used in ASQL statements.

Syntax

The syntax is as follows:

```
CASE [ expression ]  
WHEN condition_1 THEN value_1  
WHEN condition_2 THEN value_2  
...  
WHEN condition_n THEN value_n  
ELSE value_default  
END
```

Example

The following example shows how you can use CASE to simplify your SQL. The original statements were as follows:

```
UPDATE acrtrans SET status = 'N' WHERE status= 'Y' AND voucher_no=12345 AND client='EN' AND trans_type='AP'  
UPDATE acrtrans SET status = 'N' WHERE status= 'Y' AND voucher_no=12345 AND client='EN' AND trans_type='AR'  
UPDATE acrtrans SET status = '' WHERE status= 'Y' AND voucher_no=12345 AND client='EN' AND trans_type='GL'
```

```

UPDATE acrtrans SET status = 'X' WHERE status= 'Y' AND voucher_no=12345 AND client='EN' AND trans_
type='TX'

UPDATE acrtrans SET status = 'X' WHERE status= 'Y' AND voucher_no=12345 AND client='EN' AND trans_
type='TE'

```

Using CASE, we can write it like this:

```

UPDATE acrtrans SET status =
CASE trans_type
WHEN 'AP' THEN 'N'
WHEN 'AR' THEN 'N'
WHEN 'GL' THEN ''
WHEN 'TX' THEN 'X'
WHEN 'TE' THEN 'X'
ELSE 'Y'
END
WHERE status='Y' AND voucher_no=12345 AND client='EN'

```

Variables

ASQL variables

Scope

Agresso SQL allows you to use variables. These variables must be defined outside database blocks, but can be used inside the same blocks.

When a variable is defined, it will remain defined throughout the program. This means that you can not define the same variable in more than one file, if you want to run these files together (using the -h parameter).

Define variables

Variables can be used almost anywhere in a statement. You define them using the DEFINE statement and assign values using a SELECT statement. You refer to the variables using a colon in front of the name (eg. :myvar).

You can give your variable any name except those that are Agresso SQL keywords. The name should not be in quotations.

Data types

Valid data types are explained below:

Type	Description
char	Normal string with max length 255.
ident	ASQL-specific string with max length 40. An ident variable will always be assigned a string value. When referred to, however, it will return the object (table, column etc) identified by the assigned string value. See example below.
int	Standard int.

float	Standard float.
date	Standard date.

Example

In the following example, we define 3 variables and copies values from 3 columns in the last row in `org_table` into them. The column `testtab` refers to a table, and we put the value (table name) into the ident variable `mytable`.

The final update statement then updates the table referred to by `mytable`.

```
define char description(60)
/
define ident mytable(32)
/
define date last_update
/
select testtab, description, last_update
into :mytable, :description, :last_update
from org_table
/
update :mytable
set description = :description
where last_update < :last_update
/
```

A note on the INTO clause: You assign variable values in the INTO clause. The first element in the SELECT list corresponds to the first variable in the INTO clause, the second element corresponds to the second variable, etc. It does not affect the value assignments if there are more SELECTed elements than variables.

Transaction and Error Handling

Transactions

You can group INSERT and UPDATE statements together. This is done by transaction statements. Statements inside a transaction can be viewed as one single statement. It either succeeds or fails.

Make sure you do not have Data Definition Statements (create or drop statements) inside a transaction.

BEGIN

The BEGIN statement marks the beginning of a transaction.

ABORT

The ABORT statements abort a transaction. All statements executed after the last BEGIN transaction statement are undone. You cannot call ABORT transaction without having called BEGIN transaction.

END

The END statement marks the end of a transaction. All the statements called since the BEGIN transaction statement are now committed to the database.

 Try to keep transactions short. As long as you are inside a transaction, you keep locks on the tables you are manipulating. This leads to low concurrence and can also lead to deadlocks.

Error handling

When a statement does not execute because of an error, the default behaviour is to report the error message and then continue with the next statement. This behaviour is controlled by the ON ERROR statement (see also the input parameter exit flag).

There are two options:

- **On error exit:** When an error occurs, the program will either exit or skip to the next .asq file (see input parameter exit flag). If you are inside a transaction, all statements (since begin transaction) are rolled back.
- **On error cont:** This is the default behaviour (if no EN ERROR statement is used). When an error occurs, you continue with the next statement.

Data Definition Language (DDL)

DDL

A data definition language (DDL) supports the definition or declaration of database objects. Data definition statements are used to define and maintain database objects. In Agresso SQL these statements are introduced by one of the following:

CREATE (table, index, view),
ALTER,
DROP.

Note: Data definition statements should NOT be inside transactions. If the database server allow this, make sure the transaction is inside a database block.

Create table

You create a table by specifying a table name and column names with data types.

Note: In an ASQL Query series the table will automatically be dropped before it is created. Also all columns will be created to not allow NULL values, and with default values

General syntax

The general syntax is as follows:

```
create table <tab_name>
{as <select statement> |
(<col_name> <data_type>[<(length)>], ...)}
[tbspace = <location>]
```

Syntax elements

The various syntax elements are explained below:

Element	Description
<tab_name>	Name of the table you are creating.
<select statement>	A normal select statement in Agresso SQL syntax.
<col_name>	The name of a column.
<data_type>	The data type of the column. See Data Types and Functions for an overview of valid values for type and length.
<location>	The tabspace clause is used when you want to put the table on a location (tablespace, segment, dbspace) other than the default

Example - Create table

```
CREATE TABLE department
(depname CHAR(25),
depno INT,
amount MONEY,
responsible INT,
date_from DATE,
date_to DATE,
flag INT,
company CHAR(2),
hours FLOAT)
```

Example - Create table from another table

Two examples:

```
CREATE TABLE testdepartment AS
SELECT depno, depname, company
FROM department
WHERE type = 'test'
```

...

```
CREATE TABLE testdepartment AS
SELECT depno, depname, company,
SPACE(50) AS deptmanager,
CONVERT(INT,FLOAT,0) AS hours
FROM department
WHERE type = 'test'
```

Restrictions

Number of columns: The maximum number of columns is 250.

Bytes in a row: Some database servers restrict the total number of bytes for a row to less than 2KB.

Long columns: There should never be more than one raw column in a table, or more than one char/vchar column longer than 255 bytes.

ORDER BY not allowed: When creating tables with a SELECT statement, the ORDER BY clause can not be used.

Create index

An index is a table of pointers. Each row in the index points to a corresponding row in the table. You gain speed by accessing records through an index. Although there is a specific screen for maintaining indexes in Agresso it does not include temporary tables.

General syntax

The general syntax is as follows:

```
create [unique] index <ind_name>
on <tab_name> (<col_name>, ...)
[[tabspace = <location>]]
```

Syntax elements

The various syntax elements are explained below:

Element	Description
unique	Adds a unique constraint to the table. Two rows where all the columns listed in the index have the same value, is not allowed
<ind_name>	Specifies the name of the index
<tab_name>	Specifies the name of the table
<col_name>	Specifies a column name. You should try and limit the number of columns in the index. Agresso does not allow more than 10.
<location>	The tabspace clause is used when you want to put the index on a location (tablespace, segment, dbspace) other than the default

Example

```
CREATE UNIQUE INDEX aidepartment1 ON department (depno, company)
CREATE INDEX aidepartment2 ON department (responsible, client)
```

Create view

A view is a virtual table that does not have any existence in its own right, but is derived from one or more underlying base tables. Views can be created at any time.

General syntax

```
create view <view_name> [(<col_name>, ...)]
as <select_statement>
```

Syntax elements

The various syntax elements are explained below:

Element	Description
---------	-------------

<view_name>	Specifies the view name.
<col_name>	Specifies a column name. The view should not have more than 250 columns.
<select_statement>	A select statement with some limitations. You can not use an order by clause and you can not use set operators (like union).

Modify table

Formal syntax

The general syntax for MODIFY TABLE is as follows:

```
MODIFY TABLE <table>
{
  ADD [WITH CHECK] <column> <type_expression> [default {<str>|<num>}]
  [, <column> <type_expression>...] |
  DROP <column> {CONSTRAINT | <constraint>} |
  MODIFY <column> <type_expression> [default <expression>]
}
```

Syntax elements

The various syntax elements are explained below:

Element	Description
<table>	The name of the table.
<column>	Specifies the name of the column.
<constraint>	The name of the constraint. If you do not know the name of the constraint, but want to drop it anyway, use the keyword "constraint".

General example

```
MODIFY TABLE MyTestTable
ADD WITH CHECK newcol50 varchar(50), newcol100 varchar(100)
```

Note: using WITH CHECK: When using the WITH CHECK option, the command will check against the system tables – for all columns listed in the command – whether the column exists or not. This can slow down performance.

If a column already exists, a message is written to the log, and the column is ignored.

Restrictions: MODIFY TABLE in ASQL allows you to write identical MODIFY <column name> statements for both Oracle and MS SQL. It is important, however, that you are aware of the following restrictions:

- You cannot modify a column in MS SQL if it is part of an index (you can in Oracle).
- You cannot modify a column in Oracle, if the column has any values (you can in MS SQL).
- The column must already have a NOT NULL constraint.

Drop tables, view and indices

Drop table or view

The general syntax for deleting a table or a view, is as follows::

```
drop {table | view} <name>
```

where <name> identifies the table or view to drop.

Example: `DROP testdepartment`

Note: The `DROP` table <name> command will destroy both the content of the table and the table definition.

Temporary tables: It is not necessary to run `DROP` table <name> for temporary tables. Agresso will by default drop them at the end of the process.

Drop index

The syntax for deleting an index, is as follows::

```
drop <index_name> on <table_name>
```

Note: The 'on <table_name>' part is not mandatory. If you omit it, however, performance can be real slow!.

Data Manipulation Language (DML)

DML

A data manipulation language (DML) supports the manipulation and processing of database objects. The following commands are described:

- `SELECT`,
- `INSERT`,
- `DELETE`,
- `UPDATE`.
- `MERGE`.

Select

Syntax

The general syntax for `SELECT` is as follows:

```
SELECT [DISTINCT] column1, column2 ...
FROM table
WHERE <condition>
[GROUP BY column1, column2, ...]
[HAVING <condition>]
[ORDER BY column1, column2, ...]
```

Asterisk: Instead of specifying columns, you can use an asterisk (*), meaning all columns:

```
SELECT * FROM table ...
```

The DISTINCT keyword

When the DISTINCT keyword is used, duplicate rows are ignored. Only distinct (different) rows, compared to all previously retrieved, will be added to the result table.

The WHERE <condition> clause

The WHERE <condition> clause restricts the returned rows to those matching the <condition>, and where the condition in general corresponds to a specific value in one or more columns.

Examples: Typical examples of the WHERE <condition> as given below:

```
WHERE cust_name <> 'agresso'
```

...

```
WHERE cust_name <> 'visvas' and income > 10000
```

Comparison operators: The following comparison operators ar used in ASQL:

Operator	Description
=	equal to
!=	not equal to
>	greater than
<	less than
>=	greater than or equal to
<=	less than or equal to

Logical operators: The following logical operators ar used in ASQL:

Operator	Description and examples
[NOT] operator [ALL ANY]	???
IS [NOT] NULL	Example: <pre>WHERE tab_name IS NOT NULL</pre>
[NOT] EXISTS	Example: <pre>WHERE exists (SELECT * from agltransact WHERE <voucher_date> TODAY</pre>
BETWEEN	Example: <pre>WHERE date BETWEEN date_from AND date_to</pre>
[NOT] IN	Example: <pre>WHERE name IN ('per''kari,')</pre>
[NOT] LIKE	Use to compare two strings for a partial match: Example: <pre>WHERE name LIKE 'A%'</pre>
AND	Validates two conditions, and both must be true.

	<p>Example:</p> <pre>WHERE cust_name = 'MyCust%' AND income >= 500000</pre>
OR	<p>Validates two conditions, and one must be true.</p> <p>Example:</p> <pre>WHERE cust_name = 'MyCust%' OR income >= 500000</pre>
NOT	<p>Negates a condition value. The expression is true if the condition is false.</p> <p>Example:</p> <pre>WHERE NOT (cust_name = 'MyCust%')</pre>

ORDER BY

ORDER BY is used to sort the query result in either ascending (default) or descending order. The columns in the ORDER BY-statement must appear in the SELECT list.

Example - order result in ascending order:

```
SELECT depno, depname, responsible
FROM department
ORDER BY responsible, depno
```

Example - order result in descending order

To order the result in descending order, you use the keyword DESC:

```
SELECT depno, depname, responsible
FROM department
ORDER BY depno DESC
```

GROUP BY

GROUP BY gathers rows into groups and sorts the groups. With aggregate functions, for example SUM, you can get a nice result that is almost ready for printing.

The columns in the ORDER BY statement must appear in the SELECT list.

Example - order by employee group:

```
SELECT empl_group, empl_no, description, SUM(amount)
FROM payroll
GROUP BY empl_group
```

HAVING

HAVING is used to eliminate groups that do not meet the search criteria.????

Example: In the following example ????

```
SELECT empl_no, name, AVG(salary)
FROM employees
GROUP BY empl_no, name
HAVING AVG(salary) > 180000
```

UNION

UNION (with additional keywords DISTINCT or ALL) is used if you have two tables and you need to include data from both tables.

Example: In the example below, we will get a result consisting of

```
SELECT second_name,  
       first_name  
  FROM prospect  
 UNION  
SELECT second_name,  
       first_name  
  FROM customer
```

Using aliases

When writing join queries, aliases are very useful. The amount of typing can be cut down by using aliases (or correlation names). An alias is a short name that stands for a table name.

Example: The query:

```
SELECT employees.name, department.depname  
      FROM employees, department  
     WHERE employees.depno = department.depno  
       AND employees.client = department.client
```

can be simplified to:

```
SELECT a.name, b.depname  
      FROM employees a, department b  
     WHERE a.depno = b.depno  
       AND a.client = b.client
```

Cartesian Product

The simplest join is a two table SELECT in which there are no WHERE statement. Every row of the first table is joined with every row of the second table. The result is the Cartesian product of the two source tables.

Example:

```
SELECT a.emplno, a.name, a.depno, b.depname  
      FROM employees a, department b
```

Equal Join

By applying constraints to the join with a WHERE clause, unwanted rows can be filtered out. The WHERE clause should contain conditions specifying that the value in the first table must be equal to the value of the corresponding column in the second table. To avoid Cartesian product all the columns in the unique index should be included in the WHERE clause.

Example:

```
SELECT a.emplno, a.name, a.depno, b.depname  
      FROM employees a, department b  
     WHERE a.depno = b.depno
```

Outer Joins

The result set from an Outer join includes all the records in which the linked field value in both tables is an exact match. It also includes a row for every record in the primary table for which the linked field value has no match in the lookup table. For instance, you can use a Left Outer join to view all customers and the orders they have placed, but you also get a row for every customer who has not placed any orders. These customers appear at the end of the list with blanks in the fields that would otherwise hold order information.

Insert

With the INSERT statement you can enter new rows with data into a table.

Syntax

1) - get values from another table:

```
INSERT INTO <table name1> (<column1>, <column2>, ...)  
SELECT <column3>, <column4>,...  
FROM <table name2>
```

2) Insert values directly:

```
INSERT INTO <table name> (<column1>, <column2>, ...)  
VALUES (<value1>, <value2>)
```

Examples

Copy data from one table to another:

```
INSERT INTO testdep (depno, depname, responsible, floor, client)  
SELECT depno, depname, responsible, floor, client  
FROM department  
WHERE depno = 200
```

Insert values:

```
INSERT INTO testdep (depno, depname, responsible, floor, client)  
VALUES (200, 'Administration', 2, 5, 'NO')
```

Update

The UPDATE command will change data in the table rows specified in a WHERE statement.

Syntax

```
UPDATE <table name>  
SET <column name>= <value>, ...  
WHERE <condition>
```

Example

```
UPDATE testdep  
SET depname = 'Administration',  
responsible = 3  
WHERE depno = 200
```

Delete

The DELETE command removes rows identified by a WHERE <condition> clause.

Note: If the WHERE statement is omitted, all rows will be deleted.

Syntax

```
DELETE FROM <table name> [WHERE <condition>]
```

Example

```
DELETE FROM testdep WHERE floor = 3
```

Merge into

The MERGE command is an official SQL standard (introduced in SQL: 2003 - the 5th version of the SQL standard), and it is also included into the Agresso SQL standard.

Syntax

The formal syntax is as follows:

```
MERGE INTO <table> <alias> USING <table> <alias>
ON <join condition>
WHEN MATCHED THEN
UPDATE <set clause>
WHEN NOT MATCHED THEN
INSERT (<column list>) VALUES (<values list>);
```

Reference

See Release notes for Agresso Platform 2.2 for more details and examples.

Copy Statement

COPY options

Overview

The COPY statement in ASQL is used to copy data between an Agresso table and a text (ASCII) file. The following options are available:

- COPY IN - copies data from a line separated text file to an Agresso table. Data for the various table columns are either in fixed positions in the text file, or tab separated.
- COPY TO - copies data from an Agresso table to a line separated text file. The data copied to the text file can either be given fixed positions or be tab separated.

- COPY FROM - copies data from a line separated text file to an Agresso table. The data in the text file are found in defined positions, but these positions may vary throughout the text ifle, depending on the value of a defined key.

Not to be used inside a transaction

Please note the none of the COPY options described here can be called inside a transaction.

COPY IN

The COPY IN statement copies data from an ASCII file and into an Agresso table.

Syntax

The general syntax is as follows:

```
copy in [import] file = <file name>,
colsep = <column separator>, table = <table_name>,
<col_name>[ = [s]<number>], ...
```

Parameters and keywords

The various syntax elements are described below:

Element	Description
IMPORT	If the IMPORT keyword is used, it tells the COPY statement to look for the file in the directory pointed to by the AGRESSO_IMPORT environment variable. In this case, no path is needed.
<file_name>	The name of the file you want to import from. If no path is given, it will look either in the current folder or where AGRESSO_IMPORT points to.
COLSEP	Column separator keyword. Must be set to a valid <column separator> value (F or T).
<column separator>	Indicates how data in the text file shall be interpreted. Valid values are: <ul style="list-style-type: none"> • F - Fixed positions, meaning that the data for a column occupies a fixed number of positions. The start position is always the next position after the previous column, or 1 if it is the first column. When Fixed positions are used, <col_name> parameter must always be followed by the (text) length to copy. • T - Tab separated.
<table_name>	Name of table to copy data into.
<col_name>	Name of column to copy data into.
S	Used for a specific column. The S switch can only be used for one column, and indicates that the column shall contain a sequence number, and not data from the text file. If the S is followed by a number, this number will be the first number in the sequence (for the first row). For subsequent rows, the sequence number will increase with 1. If the S switch is used, it must always be followed by the first sequence number. Note: The column to use for sequence numbers, must have been defined as an int column!
<number>	Used in conjunction with the COLSEP value F. When the file has a fixed format, the value for every column takes up the same number of bytes in every row. This number specifies the number of characters (text length).

An integer value can have up to 18 digits, and a float or money value up to 25 digits, and they are both stored as character strings in the file.

Remember: When the S switch is used, it does not refer to the length, but to the first sequence number.

Example

In the following example, we copy data from *myDataFile.txt*, located in the current folder - or in the import folder. The data have fixed positions, and we copy data from position 1 to 33. The third column in the table will hold a sequence number, starting with 1 for the first row:

```
copy in file = 'myDataFile.txt', colset = F, table = hlptab,  
account=8,  
amount=25,  
sequence_no=s1  
/
```

COPY TO

The COPY TO statement copies data from a table and into an ASCII file.

Syntax

The general syntax is as follows:

```
copy to [import] file = <file_name>,  
colsep = <column separator>,  
select <col_name>[ = <number>], ...  
from <table name>[, <table name>]  
[where <condition>]
```

Parameters and keywords

The various syntax elements are described below:

Element	Description
IMPORT	If the IMPORT keyword is used, it tells the COPY statement to create the file in the directory pointed to by the AGRESSO_IMPORT environment variable. In this case, no path is needed.
<file_name>	The name of the file you want to copy to. If no path is given, it will either be created in the current folder or in the folder pointed to by the AGRESSO_IMPORT environment variable.
COLSEP	Column separator keyword. Must be set to a valid <column separator> value (F or T).
<column separator>	Indicates how data in the text file shall be separated. Valid values are: <ul style="list-style-type: none">• F - Fixed positions.• T - Tab separated.

SELECT	SELECT statement used to identify the table(s) and columns to copy from. Apart from special syntax related to the copy operation, this is a standard SELECT, allowing you to use any type of conditions to filter out unwanted rows.
<col_name>	Name of column to copy from. If the <column separator> is set to F, you must also specify the number of characters to write to the text file.
<number>	Used when the colsep value is F. This number specifies the number of characters (text length) to use for the column value in the text file. This means that shorter values will be padded with blanks.
<table name>	Name of table(s) to copy from.

Example

In the following example, we copy data from two columns in the table hlptab to the file C:\AGRCOPY.txt - but only rows where the value of the amount column is higher than 80000.

```
copy to import file = 'c:\AGRCOPY.txt',
colsep = F,
select account=8,
amount=25
from hlptab
where amount > 80000
/
```

COPY FROM

The COPY FROM statement copies data - on variable format - from a text file to a table.

Syntax

```
copy from [import] = <file_name>,
table = <table_name>,
key = <key_startpos> - <key_endpos>,
{<key_value> = {<col_name> <type>(<pos>), ...} ...}
```

Parameters and keywords

The various syntax elements are described below:

Element	Description
IMPORT	If used, it tells the COPY statement to create the file in the directory pointed to by the AGRESSO_IMPORT environment variable.
<file_name>	The name of the file you want to import from. If no path is given, COPY FROM will look either in the current folder or where AGRESSO_IMPORT points to.
<tab_name>	Specifies the name of the table you want the data to be copied into.
KEY	The KEY keyword is used to identify the start and end position of the row identifier (key!). Although

	other data fields may be located on different positions throughout the file, the key must always be available within the same start and end position.																										
<key_startpos>	Start position for the key value. The first character on a row is in position 1																										
<key_endpos>	End position for the key value.																										
<key value>	A specific key value which introduces a new interpretation of the data. When used, the <key value> must be followed by one or more column specifications. See <col_name>, <type> and <pos> below.																										
<col_name>	Name of column to copy data into. Must be followed by the column data type (tells how to interpret the text data) and the start and end position for the column data in the text file.																										
<type>	Specifies the data type of the column. Valid type values are: <table border="1"> <thead> <tr> <th>Type</th><th>Description</th></tr> </thead> <tbody> <tr> <td>c</td><td>Character</td></tr> <tr> <td>i</td><td>Integer</td></tr> <tr> <td>f<no></td><td>Float with <no> decimals, no decimal separator</td></tr> <tr> <td>f.<no></td><td>Float with <no> decimals and a decimal separator</td></tr> <tr> <td>m<no></td><td>Money with <no> decimals, no decimal separator</td></tr> <tr> <td>m.<no></td><td>Money with <no> decimals and a decimal separator</td></tr> <tr> <td>d1</td><td>Date on the format MMDD.</td></tr> <tr> <td>d2</td><td>Date on the format YYMMDD.</td></tr> <tr> <td>d3</td><td>Date on the format YYYYMMDD.</td></tr> <tr> <td>d</td><td>Date on the format YYYYMMDD hh:mm:ss.</td></tr> <tr> <td>p1</td><td>Period on the format YYPP.</td></tr> <tr> <td>p2</td><td>Period on the format YYYYPP.</td></tr> </tbody> </table>	Type	Description	c	Character	i	Integer	f<no>	Float with <no> decimals, no decimal separator	f.<no>	Float with <no> decimals and a decimal separator	m<no>	Money with <no> decimals, no decimal separator	m.<no>	Money with <no> decimals and a decimal separator	d1	Date on the format MMDD.	d2	Date on the format YYMMDD.	d3	Date on the format YYYYMMDD.	d	Date on the format YYYYMMDD hh:mm:ss.	p1	Period on the format YYPP.	p2	Period on the format YYYYPP.
Type	Description																										
c	Character																										
i	Integer																										
f<no>	Float with <no> decimals, no decimal separator																										
f.<no>	Float with <no> decimals and a decimal separator																										
m<no>	Money with <no> decimals, no decimal separator																										
m.<no>	Money with <no> decimals and a decimal separator																										
d1	Date on the format MMDD.																										
d2	Date on the format YYMMDD.																										
d3	Date on the format YYYYMMDD.																										
d	Date on the format YYYYMMDD hh:mm:ss.																										
p1	Period on the format YYPP.																										
p2	Period on the format YYYYPP.																										
<pos>	The position is in the <start_pos> - <end_pos> form, for example 12-23 . The first position in a row is 1. Remember that mycol (10-15) means that the mycol column in the table have length 6, starting in position ten, including position 15.																										

Example

In the following example we copy from the file Tbrk0606.dat located in the current folder or in the import folder. The KEY is located in first 8 positions, and we can handle three key values (meaning that other key values will be ignored):

```
COPY FROM IMPORT = 'Tbrk0606.dat',
TABLE = 'hlptbl',
KEY = 1-8,
'940SWI01' = foreign_acc c(42-76),
statement_ob c(110-134)
'940SWI02' = foreign_acc c(42-76),
trans_date d2(91-96),
voucher_date d1(97-100),
```

```
dc_cflag c(101-101),  
cur_amount m2(104-118),  
description c(141-156),  
ext_inv_ref c(157-162)  
'940SWI05' = foreign_acc c(42-76),  
statement_cb c(91-115)
```

! Note that there should be no comma between each new key value.

Sample

The script should be placed in a text file with asq-extension.

```
/* ASQL script sample */  
PRINT '*** ASQL script sample ***'  
  
/* abort script on the first error */  
ON ERROR EXIT  
  
/* drop the table if it exist */  
IF EXISTS asqsample  
  
DROP TABLE asqsample  
  
END  
  
/* Create a table */  
CREATE TABLE asqsample  
(  
    varchar_col varchar(256),
```

```
number_col int1,
money_col money,
last_update date
)
/
/* CREATE unique index */
CREATE UNIQUE INDEX aiasqsample on asqsample (vchar_col,number_col)
/
/* Insert a row into the table */
INSERT INTO asqsample
(
    vchar_col,
    number_col,
    money_col,
    last_update
)
VALUES
(
    'test row 1',
    12,
    12.12,
    NOW
)
/
/*
define a variable that will contain the agresso version */
DEFINE agresso_version char(50)
/
/*
set a initial value for the variable */
SELECT 'Agresso version not found' INTO :agresso_version
/
```

```

/* If the table asysetup with column text2 exist, write the text2 column value into the variable agresso_version*/
IF EXISTS asysetup text2 CONSTRAINT
/
SELECT concat('Agresso version: ',text2) INTO :agresso_version
FROM asysetup
WHERE name = 'BASE_VERSION'
/
END IF
/

/* Write agresso version to output*/
PRINT :agresso_version
/

/* define a variable*/
DEFINE agrdata char(50)
/
/* Run a MSSQL spesific query and insert tablespace for the table we created into variable agrdata*/
BEGIN sqlserver
/
select distinct s.groupname into :agrdata
from sysobjects t, sysindexes i, sysfilegroups s, syscolumns c
where t.type = 'U'
and t.id = c.id
and t.id = i.id
and (i.indid=0 OR i.indid=1)
and i.groupid = s.groupid
and t.name = 'asqsample'
/
END
/

```

```

/* Script will continue if something fails in the rest of the script */

ON ERROR CONTINUE
/

/* Run a oracle spesific query and insert tablespace for the table we created into variable agrdata */

BEGIN oracle
/
    select tablespace_name into :agrdata
    from user_tables
    where table_name = UPPER('asqsample')
/
END
/
/* Write tablespace to output */
PRINT :agrdata
/
PRINT '***** SCRIPT COMPLETE *****'
/

```

The script can be run with the command:

```
ASQL.exe -D<datasource> -F"<path to script>" -v -l"c:\asqllogfile.log"
```

56 News

Syntax

- LONGTEXT: the content of a LONGTEXT column can be maximum 32000 characters
- BLOBLENGTH(col_expr)
- CHARINDEX(string_expr1, string_expr2 [,start_location]) Searches expression2 for expression1 and returns its starting position if found. The search starts at start_location (optional).
- DATETIME2STR(date_expr) Converts a data value to a string on the format 'yyyymmdd hh:mi:ss'

- INT2STR(int_expr)
- REPLACE(string_expr, string_pattern, string_replacement) Replaces all occurrences of a specified string value with another string value
- STR2DATE(string_expr)
- SUBSTR(string_expr, start_expr, length_expr)
- TO_CHAR(expr) Converts an expression to string
- TO_COUNTER(expr) Converts an expression to int8
- TO_DATE(string_Expr) Converts a string of format 'yyyymmdd hh:mi:ss' to date (same as datetime2str)
- TO_GUID(string_expr)
- TO_INT(expr)
- TO_FLOAT(expr)
- TO_MONEY(expr)

And also the ranking functions

- DENSE_RANK () OVER ([< partition_by_clause >] < order_by_clause >)
Returns the rank of rows within the partition of a result set, without any gaps in the ranking. The rank of a row is one plus the number of distinct ranks that come before the row in question.
- ROW_NUMBER () OVER ([<partition_by_clause>] <order_by_clause>)
Returns the sequential number of a row within a partition of a result set, starting at 1 for the first row in each partition.

AgrIndex

Re-generation of indexes

General purpose

AgrIndex.exe is a console application for re-generation of indexes for Agresso tables. When run, AgrIndex will collect the index definitions from the tables `asysindex` and `aagindex` and verify that all indexes in the database exist according to the definitions.

If a defined index is missing, AgrIndex will re-create it. If the index exist, but the index column definition is wrong, the column will be re-created correctly.

Index definition tables

Agresso index definitions are found in the table `asysindex`, which contains the indexes distributed by the Agresso installation package. User defined index defintions should be placed in the `aagindex` table.

Application location

A standard Agresso installation will place *AgrIndex.exe* in the bin directory (AGRESSO_EXE).

Using AgrIndex

Basic syntax

The syntax for running AgrIndex is as follows:

```
Agrindex.exe <DataSource> [-s | -a | -u] [-t<location> | -T<location>] [-n] [<table filter>]
```

where `<DataSource>` and at least one parameter (`-s` | `-a` | `-u`) are mandatory.

Parameters

The table below explains the parameters (or switches) you can use:

Parameter	Description
<code>-s</code>	Indexes found in <code>asysindex</code> are regenerated
<code>-a</code>	Indexes found in <code>aagindex</code> are regenerated
<code>-u</code>	Indexes found in both <code>asysindex</code> and <code>aagindex</code> are regenerated (union)
<code>-t<location></code>	Specify database location. This will be used for locations not found on existing indexes or if the index is missing.
<code>-T<l-ocation></code>	Specify a database location that will be used for all re-generated indexes.
<code>-n</code>	Don't use <code>acrclient</code> location for system tables (<code>acrclient</code> 's location is used for system indexes by default).
<code><table filter></code>	A table filter, in the form of a table name (wildcards allowed) can be added to the end of the parameter list. Example: <code>acr%</code> limits the selection to tables starting with <code>acr</code> .

Examples of use

The following examples shows AgrIndex in use:

- To regenerate all Agresso indexes for data source MyServerDataSource:
`Agrindex.exe MyServerDataSource -s`
- To regenerate indexes for all tables starting with `aim` with location `MY_LOCATION`:
`Agrindex.exe MyServerDataSource -u -TMY_LOCATION aim%`
- To regenerate all user defined indexes (indexes defined in `aagindex`):
`Agrindex.exe MyServerDataSource -a`
- To regenerate indexes for the table `aagmenu`
`Agrindex.exe MyServerDataSource -s aagmenu`

Related topics

[Agresso Data Segments](#)

Security Areas

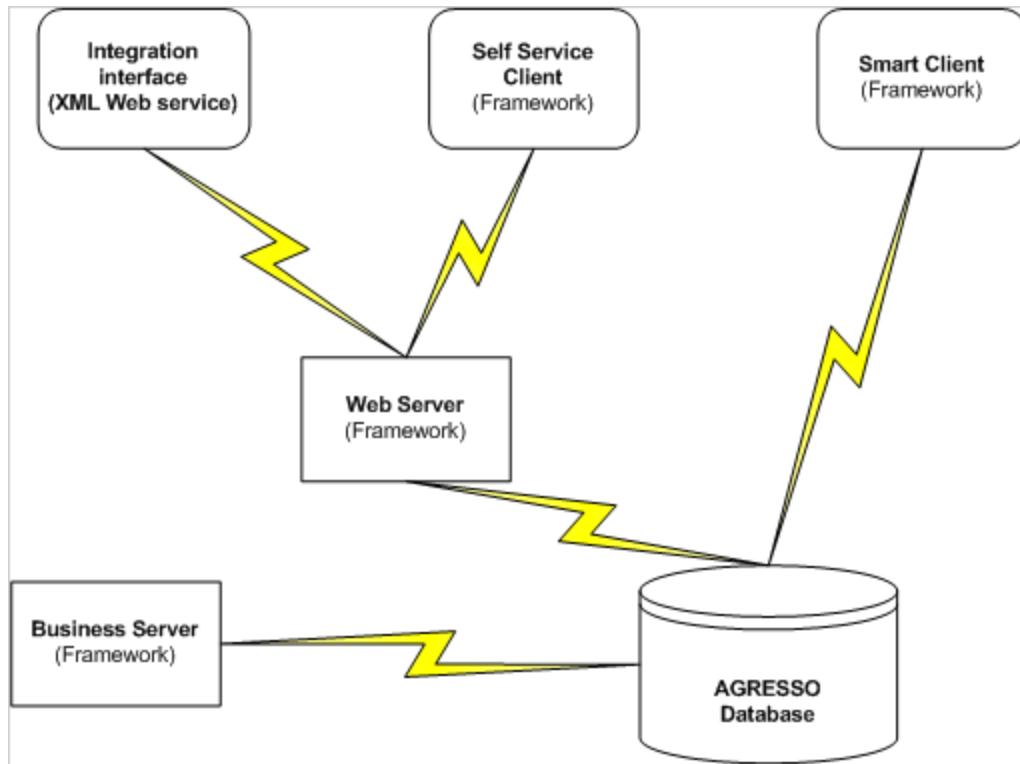
Basic architecture

From a security point of view, we can identify the following basic elements in a typical Agresso configuration:

- A set of client types:
 - Integration interface (a customised browser client)
 - Agresso Self service client
 - Agresso Smart client
- The Web server,
- The Business server,
- The Agresso database.

Communication diagram

The communication between the various elements are illustrated below:



Customer responsibility

Before implementing Agresso in a production environment, it is important that the installation owner has performed a thorough risk analysis, and has a high degree of security awareness.

Security areas

Agresso explicitly addresses two main security areas:

- Access control
- Program or module authentication.

These areas are managed by standard Agresso functionality and the Agresso Framework components, which makes sure that neither users nor program modules get access to the Agresso database unless properly authenticated.

In order to set up a secure Agresso system, it is also necessary to protect communication between the various elements. Agresso has no built-in functionality for protection of communication through insecure lines, but supports all the generally available tools for hiding and protecting communication over the network(s).

General recommendations

The following general rules should be applied:

- Run the system with the lowest privileges possible.
It is important to configure the system to run with the lowest privileges possible to reduce the damage if someone executes non-trusted code within the Agresso context.
- Configure your operating system as recommended by the vendor.
Please follow the recommended security guidelines from the operating system or dependent system vendors. Always install the latest patches and security updates.
- Assure that you have tightened your NTFS permissions and configure the code groups and permission sets at the most detailed level possible. Configure software restriction policies on your domain and assure that you have an appropriate physical protection for the system.
- Change the Agresso schema owner password at regular intervals.
- Use Centrally Configured Clients for Smart Client users; avoid stand alone client installations! For high security environments, we recommend Citrix / Terminal Server, with Smart Client as Published application.
- Use traffic encryption between clients and the application and database server.

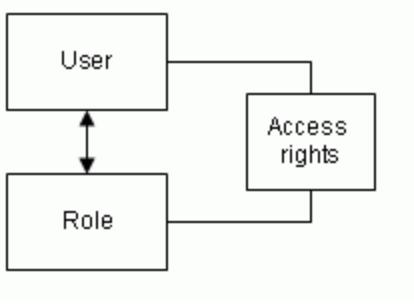
Access Control

Password protection

Users and roles

The basic access rights in Agresso are based on registered users and defined roles. A user may inhabit, or belong to, many roles, and a role will normally encompass many users. The access rights for a specific user are made up of the sum of access rights for the user as such, and for the roles the user belongs to. Both users and roles have a defined life-span, identified by a from-date and a to-date.

The relations between users, roles and access rights are outlined in the following model:



User authentication models

Agresso supports authentication models for both Active Directory (AD) and LDAP users, and has default solutions for log-on through specific Agresso log-on windows, as well as for single sign-on (SSO).

Password protection

When a person tries to connect to Agresso, Agresso will always check against registered user names and passwords.

Passwords are encrypted according to the company's selected encryption algorithm. Agresso is delivered with five predefined encryption variants (numbered 0 to 4, where the highest number points to the most secure algorithm). The company is free to introduce new ones, which will then be added to the list.

The PWD_VARIANT system parameter

The system parameter PWD_VARIANT holds a value that points to the companies default encryption algorithm. Upon delivery of Agresso, PWD_VARIANT will have the default value 4, pointing to the an algorithm using SHA-1 and salt encryption.

When new users are created, their password will be encrypted according to the value of PWD_VARIANT.

A note on ABW 5.4: In Agresso 5.4, the default variant value was 3. The new default variant value will not be used until the user changes his/her password.

Other parameters: See overview of password related system parameters [below](#).

Individual adjustments

The encrypted password is stored together with a pointer to the encryption variant used. An administrator can change the encryption variant for individual users, and thus disregard the default value of PWD_VARIANT.

Default password encryption

Salt and SHA-1

When using the (default) PWD_VARIANT 4, the user's password is *salted* and encrypted according to the SHA-1 algorithm before it is stored in the database.

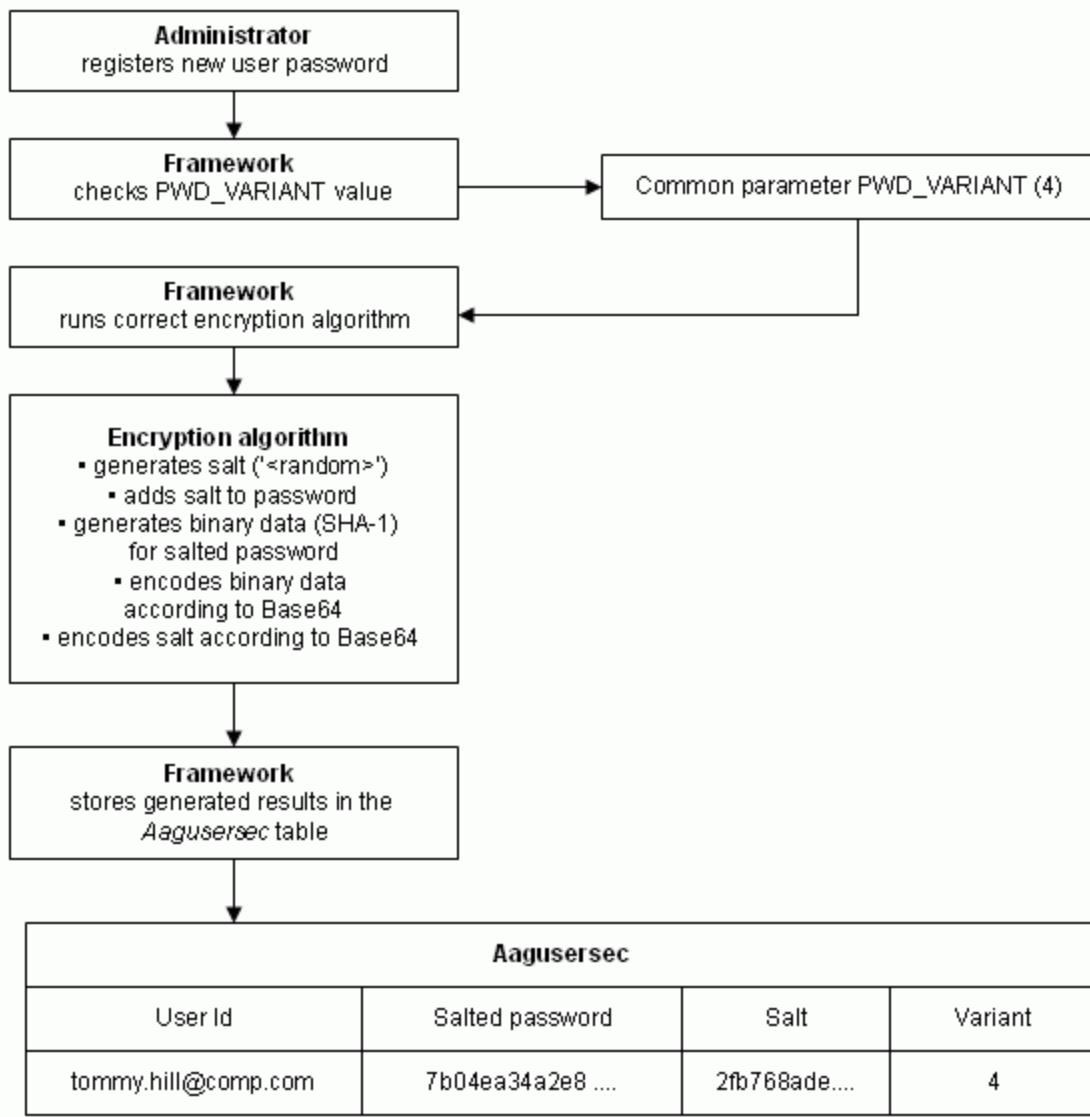
Encryption and storage

Salt is a random length string with random characters, generated when a new user is created or when the user changes the password.

This salt is added to the password – before encryption – and stored with the salted password and the PWD_VARIANT in the table Agusersec.

Encryption example

The diagram below illustrates how a (new) password is encoded and stored according to the default value of PWD_VARIANT:



Management of roles and users

Reference

Management of users and roles in Agresso are handled by the User master file and Role master file respectively. In addition, we can use the Data control management master file to further detail access rights for roles and users.

Please refer to the Agresso Data control and user access reference manual for details. In this document we will just give a general overview of user and role management, in order to present a complete picture of security related issues.

ACCESS model

A user – or a role – is given access to a set of menu items, and if desired, to all underlying menu items in the menu tree. If necessary, an administrator can restrict access to specific menu items on lower levels.

Data control

Although a user (or role) may have full access to a certain master file, data control allows the administrator to restrict access to specific instances of the data, based on the data value.

Attribute based control: Data control is based on the functionality for attributes, attribute values and relations, and is implemented for all master files.

Password settings by system parameters

Parameter name	Description
PWD_CODE	If turned on, the password must contain both characters and digits.
PWD_DURATION	Sets the valid period &endash; in days &endash; for a new password.
PWD_EXPIRATION_WARNING	States &endash; in number of days before the password expiration date &endash; when a user shall be notified.
PWD_GENERATION	Sets the number of historic passwords that will be remembered when a new password is entered and validated. The new password cannot be any of the stored, historic passwords.
PWD_LOCKOUT_DURATION	Specifies - in minutes &endash; for how long you will be locked out from Agresso after a defined number of invalid logon attempts. The number of invalid attempts is defined in the parameter PWD_TRIES.
PWD_LOG	Specifies whether a record of the accesses to Agresso is maintained. If the function is used the following will be logged: 1. Logons. 2. Normal logging on by use of Agresso's Exit function. 3. Lockouts. The report System access log (AG11) shows logged accesses and exits.
PWD_NOCHARS	Sets the minimum number of characters required for a valid password. If set to 0, there will be no check.
PWD_NOT_USERNAME	If turned on, the password cannot be the same as the user name.
PWD_RULE_DIGITSCOUNT	If turned on, the password must at least contain the number of digits defined by the parameter value.
PWD_RULE_SPECIALCOUNT	If turned on, the password must at least contain the number of special characters defined by the parameter value.
PWD_RULE_LOWERCASECOUNT	If turned on, the password must at least contain the number of lowercase characters defined by the parameter value.
PWD_RULE_UPPERCASECOUNT	If turned on, the password must at least contain the number of uppercase characters defined by the parameter value.
PWD_TRIES	Sets the number of accepted, invalid logon attempts before the user is locked out. The same limitation is valid for all users.
PWD_VARIANT	A pointer to the security architecture where the encryption algorithm or hash is stored.

	When changing this variant, all new passwords (for new users and when users change their password) will be affected.
--	--

Integrated Database Authentication

[Open Reference manual](#)

The link above will open a PDF-version of the Reference manual for Integrated Database Authentication.

Authentication and Authenticators

Authenticators controls how users authenticate when they logon to Agresso. By default users login with Agresso user and password (Agresso Authentication).

Another options is Windows Authentication (Referred to as Single-Sign-On in previous Agresso releases).

Configure Windows Authentication

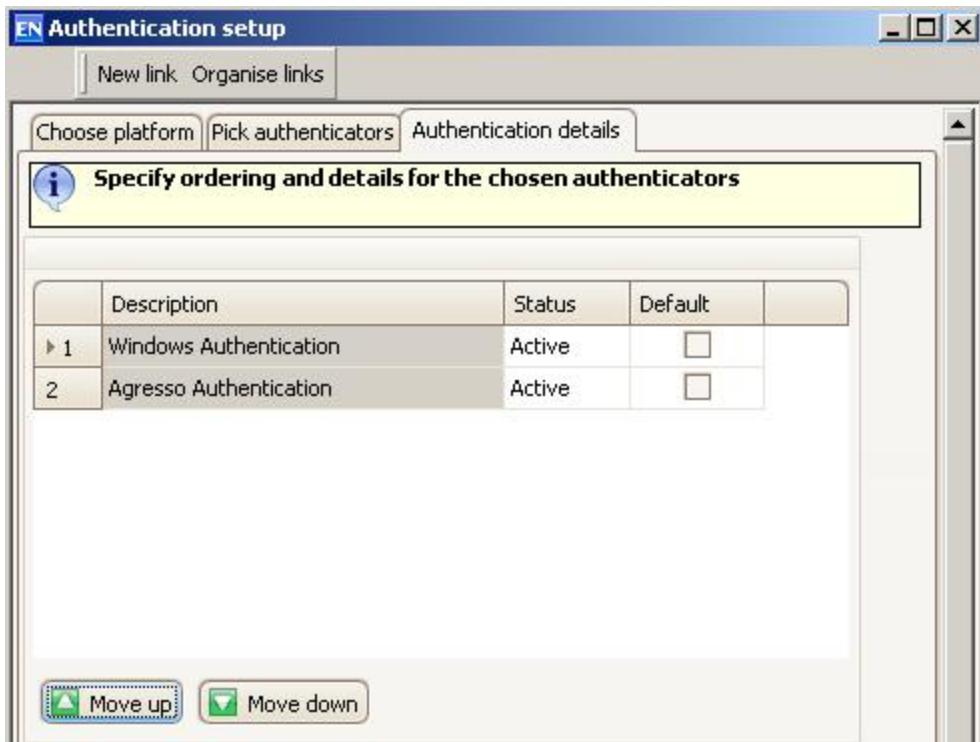
To enable windows authentication, the Windows Authentication authenticator must be configured and moved to first priority.

Existing authenticators are configured from the authentication setup page. The configuration is done in three steps:

1. Select the platform: Self Service, Smart Client or Web Services.
2. Select the authenticators that should be made available
3. Set the priority of the authenticators

To enable integrated windows authentication, select the windows authentication authenticator and move it to first priority.

Agresso users needs to be associated with a domain users to use windows authentication, this can be configured from **user master file** in the smart client.



Self Service and Web Services

For Self Service and web services an additional step is needed to enable or disable windows authentication. Use the Authentication node under the web application in AMC to configure this.

For details on the Authentication setup configuration, please refer to the Reference manual for authentication setup that can be found on the DVD.

Supported platforms

Reference

[See latest supported platforms list document](#)

Tables used by AMC

The following tables are used by AMC:

Table name	Description
aagserverqueue	Holds the configured server queues.
aagserviceuri	Contains the published web applications.

Table name	Description
asysservice	Contains the available web services. The services appears in the "Add Web Service" wizard.
aagservcmdhead, aagservcmddetail	Contains commands posted to AgrBusinessServer service. For instance when a server queue is added from AMC, the change is added to this table, the AgrBusinessServer service polls these tables for changes, and executes the command, without having to restart the service.
aagsystemconfig	Contains Mail, SMS and printer configuration.
aaguserconfig	Contains mail configuration for Agresso users.
aagalerthead ,aagalertdetail	Contains status of jobs used by Process View in AMC and the Alert Server Web Service. The table accreport , is used for monitoring the report queue in the Process View. By default these tables are polled every third second when the process view is open. The poll frequency can be configured from the properties dialog on the "UNIT4 Agresso node" in AMC.
aagitwsservice, aagitwsversion, aagitwsextension	Tables that contains web services configured on the Agresso Web Service Host.
acrupgradehead, acrupgrade(detail, acrupgradeblob	Used by the Upgrade Manager.
acrmalqueueserv	Contains SMTP profiles configured for AgrMailQueue and the profile settings for the queue.

Agresso.Management.PowerShell Reference

[View reference manual](#)

Use this link to open the html-version of the Reference manual for Agresso.Management.PowerShell.

Migration from Oracle to MS SQL Server

If switching RDBMS from Oracle to MS SQL Server and using copyora/copym to recreate the tables, all expected char columns will be of the type of varchar. This will cause problems in the payment routines. To fix this for the column that can not be of type of varchar, run the script [oracle_to_mssql.asp](#) located in the directory ..\Database Script\DbUpgrade\Upgrade\Scripts. Do this AFTER the migration from Oracle to MsSql.

The script can be run from a command window using [asql.exe](#), or from [Agresso Management Console](#) under the .\Backoffice Datasources\<DataSource>\Database Tools\Run Script node.