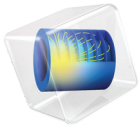

COMSOL Multiphysics

Model Manager Server Manual

6.4

COMSOL

Model Manager Server Manual

Version: COMSOL 6.4

Contact Information

Visit the Contact COMSOL page at www.comsol.com/contact to submit general inquiries or search for an address and phone number. You can also visit the Worldwide Sales Offices page at www.comsol.com/contact/offices for address and contact information.

If you need to contact Support, an online request form is located on the COMSOL Access page at www.comsol.com/support/case. Useful links:

- Support Center: www.comsol.com/support
- Product Download: www.comsol.com/product-download
- Product Updates: www.comsol.com/product-update
- COMSOL Blog: www.comsol.com/blogs
- Discussion Forum: www.comsol.com/forum
- Events: www.comsol.com/events
- COMSOL Video Gallery: www.comsol.com/videos
- Support Knowledge Base: www.comsol.com/support/knowledgebase
- Learning Center: www.comsol.com/support/learning-center

Part number: CM020018

Contents

Chapter 1: Introduction

Chapter 2: Installation

Chapter 3: Administration

Chapter 4: Asset Management

Chapter 5: Glossary

1

Introduction

Read this guide to learn how to install and administer a *Model Manager server*, a web server that hosts Model Manager databases. The Model Manager server can either be configured to use internal database components managed by the server, or external database components provided and managed by your organization. The guide also contains details on the asset management system included with a Model Manager server — a web-based tool you can use to manage simulation projects involving models built in the COMSOL Multiphysics® software. See the *Model Manager Reference Manual* for details on how to access a Model Manager server database from within the COMSOL Desktop® modeling environment.

In this chapter:

About the Model Manager Server

In this section:

What Can You Do with a Model Manager Server?

A Model Manager database can be shared between multiple users by hosting the database in a Model Manager server. You access the database by connecting to the Model Manager server from the COMSOL Multiphysics software via your organization's internal network. This enables you and your coworkers to collaborate on version-controlled simulation models and data files — all while staying within the COMSOL Desktop modeling environment. Read the *Model Manager Reference Manual* to learn more about what you can do with a Model Manager database accessed via a Model Manager server.

A Model Manager server database uses three separate server components:

- A *relational database management system* (RDBMS) for the version control management of simulation models and data files in a *SQL database*.
- A *resources directory* for storing binary and text data whose size is deemed too large to put directly inside the SQL database. This data can, for example, be built, computed, and plotted data generated by models, or auxiliary data such as CAD and interpolation data used as input by models.
- An *enterprise search platform* — the Apache Solr™ enterprise search platform — used for the search functionality of the Model Manager tool.

An installation of Model Manager server optionally includes the PostgreSQL® relational database management system and the Apache Solr™ enterprise search platform. Selecting to include these *managed server components* enables you to configure a Model Manager database without installing any additional software. Such a database can be fully managed by the Model Manager server itself, including starting and stopping component subprocesses and handling data backups. During installation, you can specify that a managed database with default configuration settings should be automatically created by the Model Manager server the first time the server starts. You are free to opt out of this automatic creation, however — a Model Manager database can always be set up at a later point once the Model Manager server is up and running.

A Model Manager server database can also use *external server components* provided and maintained by your organization. This includes using an external relational database management system for the SQL database — the supported platforms are Microsoft SQL Server®, MySQL®, Oracle® Database, and PostgreSQL®. Similarly, you can set up a Model Manager server database to use an external Apache Solr™ installation.

From the web interface of a Model Manager server, you can configure the server to use secure connections using transport layer security (TLS) provided via HTTPS, set up authentication mechanisms so that users can log in to the server via the COMSOL Multiphysics software or via a web browser, and perform other administrative tasks. You also access the web interface when you want to add or reconfigure your Model Manager server database. This includes, for example, modifying the locations on the file system where managed server components store your data. You can even configure multiple databases on the same Model Manager server using any combination of managed and external server components.

A Model Manager server also comes with a web-based *asset management system* that enables simulation engineers working in COMSOL Multiphysics to easily collaborate on simulation projects with people in your organization who may not have access to the COMSOL Multiphysics software. Via the asset management system, users can link models and data files to various documents, presentations, project notes, slides, and other supplementary files and metadata on so-called *assets* — all while keeping everything in the same database that stores the models and data files. Users can also browse and search the version-controlled models and data files themselves, as well as save new versions of them by uploading files from their local computer. This enables, for example, engineers working with other tools and software to update data via the web interface so that this data is immediately available to simulation engineers working in the COMSOL Desktop. Or, those same simulation engineers can publish results in the form of animations, images, plots, reports and other output via the asset management system simply by exporting such output to the Model Manager server from the COMSOL Desktop.

Where Can I Access the Documentation?

A number of online resources have more information about COMSOL, including licensing and technical information. The electronic documentation, topic-based (or

context-based) help, and the Application Libraries are all accessed through the COMSOL Desktop.

If you are reading the documentation as a PDF file on your computer, the blue links do not work to open an application or content referenced in a different guide. However, if you are using the Help system in COMSOL Multiphysics, these links work to open other modules, application examples, and documentation sets.

CONTACTING COMSOL BY EMAIL

For general product information, contact COMSOL at info@comsol.com.

COMSOL ACCESS AND TECHNICAL SUPPORT

To receive technical support from COMSOL for the COMSOL products, please contact your local COMSOL representative or send your questions to support@comsol.com. An automatic notification and a case number will be sent to you by email. You can also access technical support, software updates, license information, and other resources by registering for a COMSOL Access account.

COMSOL ONLINE RESOURCES

COMSOL website	www.comsol.com
Contact COMSOL	www.comsol.com/contact
COMSOL Access	www.comsol.com/access
Support Center	www.comsol.com/support
Product Download	www.comsol.com/product-download
Product Updates	www.comsol.com/product-update
COMSOL Blog	www.comsol.com/blogs
Discussion Forum	www.comsol.com/forum
Events	www.comsol.com/events
COMSOL Application Gallery	www.comsol.com/models
COMSOL Video Gallery	www.comsol.com/videos
Learning Center	www.comsol.com/support/learning-center
Support Knowledge Base	www.comsol.com/support/knowledgebase

Overview of the Manual

This *Model Manager Server Manual* contains information that helps you install, configure, and administer a Model Manager server. It also contains information about the server's asset management system — a web-based tool you can use to manage and organize simulation projects involving models built in the COMSOL Multiphysics software. The information in this guide is specific to this functionality. Instructions on how to use the Model Manager in general are included with the *Model Manager Reference Manual*.

TABLE OF CONTENTS AND INDEX

To help you navigate through this guide, see the Contents section and Index.

QUICKLY GETTING STARTED

Use the Quick Start Guide to get an overview of the necessary steps for getting started with a Model Manager server.

INSTALLING A MODEL MANAGER SERVER

The Installation chapter guides you through the installation and startup of a Model Manager server.

ADMINISTRATING A MODEL MANAGER SERVER

The Administration chapter contains details on how you configure and administer a Model Manager server via its web interface.

USING ASSETS TO MANAGE SIMULATION PROJECTS

In the Asset Management chapter, you will learn how assets can be used to connect simulation engineers working in COMSOL Multiphysics with people in your organization who may not have access to the COMSOL Multiphysics software.

Quick Start Guide

Getting started with a Model Manager server involves the following steps:

1 Install the Model Manager server software on a server computer in your organization's network — see the Installing a Model Manager Server chapter. The section Installation Planning in that chapter contains general guidelines and background information on possible server setups, data storage strategies, and handling of backups.

   During installation of the Model Manager server, you can specify that a Model Manager database using managed server components with default configuration settings should be automatically created on first launch of the server. You can safely skip this automatic creation, thereby enabling you to take full control of the database setup once the server is up and running.

2 For the highest availability, ensure that the Model Manager server software is started automatically when the server computer itself starts — see Starting a Model Manager Server.

3 Review the security guidelines for the Model Manager server software — see Security. For example, unless the network is completely trusted, it is recommended to either place the Model Manager server behind a reverse proxy with HTTPS support or add a connector with TLS enabled — see below.

4 Log in to the Model Manager server in a web browser using the temporary password provided during installation — see Accessing the Web Interface. Change your temporary password when prompted.

   If you opted out of automatic database setup, you will be redirected to the **Add Database** page in the **System** administration area for manual database setup. Continue with the next step. Otherwise, you will be redirected to the **Home** page of the server. Continue with step **6**.

5 Review the suggested data directories and backup directories of the Managed Server Components for the new Model Manager database — see also Adding the Model Manager Server Database on First Login. You are strongly recommended to at least change the backup directory locations to a different server disk to safeguard against a potential disk crash. You can also replace some, or all, of the managed server components with external server components — see Databases with External Server Components. The latter can include, for example, a SQL database administered by your organization. Click **Save** to create the new database.

6 The Model Manager server is configured with default administrative settings. To review and possibly update these settings, click the cog wheel in the top navigation bar and select **System** to open the system administration area unless the previous step already brought you here.

7 Set the authentication mechanism used when users log in. In the **System** navigation sidebar, under **Login**, you can choose between:

   a **Accounts**. Manually create accounts for the users that should have access to the Model Manager server. The account passwords are stored locally in the Model Manager server Preference Directory — see Local Authentication.

   b **External Authentication**. Delegate the account authentication to an external authentication provider such as Windows® authentication or an LDAP server — see External Authentication.

   c **Proxy Authentication**. Delegate the account authentication to a trusted reverse proxy placed in front of the Model Manager server — see Proxy Authentication.

8 Unless you have placed the Model Manager server behind a reverse proxy with HTTPS support, you are strongly recommended to configure the server to use secure connections. Under **Configuration > Connectors,** add a connector with TLS enabled — see Connectors. You will need to have a TLS certificate on hand.

9 Unless you selected during installation to manually create the database — in which case you would have been redirected to the **Add Database** page on login — decide if you want to use the default managed server components automatically set up for the Model Manager server, use custom managed server components, or use external server components:

   a If you want to use the automatically set up managed server components, review them under **Managed Components** — see Managed Server Components. Preferably update the backup directory locations of the default managed PostgreSQL® server and the default managed resources directory to locations on a different server disk than the corresponding data directories. The new backup directory locations are used the next time the Model Manager server is restarted.

   b If you want to use custom managed server components, but leave the automatically set up default components unmodified as references, add new managed server components under **Managed Components** — see Managed Server Components. Also add a new Model Manager server database using these managed server components from the **Add Database** page under **Configuration > Databases** — see Databases with Managed Server Components. Set the new

database as the current default and either deactivate or remove the automatically created database — see The Database Page.

c If you want to use external server components, add a new Model Manager server database from the **Add Database** page under **Configuration > Databases**. Configure the external SQL database, the external resources directory, and the external search indexes — see Databases with External Server Components. It is also possible to use a mixture of managed and external server components for the Model Manager database. Set the new database as the current default and either deactivate or remove the automatically created database — see The Database Page.

10 Launch COMSOL Multiphysics and verify that you can connect to the Model Manager server database from the COMSOL Desktop environment — select **File > Open From > Add Database > Connect to Server Database** and enter the address of the server and your account credentials. Remember to include the port number specified during installation in the **Server** field unless connecting on a default port — 80 for a non-secure connection; 443 for a secure connection. See Connecting to a Server Database in the *Model Manager Reference Manual*.

2

Installation

In this chapter you will learn how to install, start, and secure a Model Manager server. You are strongly recommended to first study the general guidelines and background information for the server setup in the Installation Planning section before proceeding with the Model Manager server installation itself.

In this chapter:

Installing a Model Manager Server

A Model Manager server installation is similar to a COMSOL Multiphysics® or COMSOL Server™ installation. For detailed information on installation, license files, and license management, see the *COMSOL Multiphysics Installation Guide*.

Before starting a Model Manager server you need to start a license manager. The FlexNet® license manager can be installed together with a Model Manager server. A license server is not needed when running a trial license. For more information on the FlexNet® license manager, see the *COMSOL Multiphysics Installation Guide*. You can download the document from www.comsol.com/iog.

In this section:

Installation Planning

A Model Manager server is typically installed on a server computer in your organization's internal network. Computers running the COMSOL Multiphysics software connect to the server computer over the network in order to access the Model Manager server database. Users can also access the Model Manager server web interface via a web browser — either to perform administrative tasks for the server or use the web-based asset management system.

This section contains general guidelines and background information to help you with planning your server setup. This includes a description of the various directories used by the server — the Installation Directory for the Model Manager server program, the Preference Directory for the Model Manager server settings and log files, and the Data Directories for the Model Manager server database — with particular emphasis on recommendations for access permissions, disk requirements, and backup strategies for

these directories. The section also contains details on Server Processes and Memory Footprint for the Model Manager server and some Examples of Server Setups.

INSTALLATION DIRECTORY

The Model Manager server is supported on the Windows®, Linux®, and macOS operating systems. The default installation directory for each operating system is:

- Windows®: `C:\Program Files\COMSOL\COMSOL64\ModelManagerServer`
- Linux®: `/usr/local/comsol64/modelmanagerserver`
- macOS: `/Applications/COMSOL64/ModelManagerServer`

You can choose another directory when installing the server. The installation directory may either be on a physical disk or on a network disk.

It is recommended to not give the user account running the Model Manager server process any permissions other than *read* access to the installation directory.

The installation directory contains the program files for the Model Manager server and its managed server components. It does not contain any Model Manager server databases — see Data Directories.

PREFERENCE DIRECTORY

Server settings and log files for a Model Manager server are stored in a *preference directory* with default location:

- Windows®: `<user>\.comsol\v64modelmanagerserver`
- Linux®: `<user>/.comsol/v64modelmanagerserver`
- macOS: `<user>/Library/Preferences/COMSOL/v64modelmanagerserver`

with `<user>` being the home directory for the user account running the Model Manager server process. The location when running, for example, as a Windows® service using the predefined **NetworkService** user account is:

```
C:\Windows\ServiceProfiles\NetworkService\.comsol\
v64modelmanagerserver
```

Replace `NetworkService` with `LocalService` in the directory path for the predefined **LocalService** account in Windows®.

Two subfolders of particular interest in the preference directory are:

- `db` — containing accounts, database configurations, and other server preferences.
- `logs` — log files for the Model Manager server.

It is recommended that only the user account running the Model Manager server process is granted access permissions to the preference directory.

You can specify another directory path for the preference directory using the `-prefsdir` command option — see Command Options for further details.

The Model Manager server preference directory must be located on a physical disk. Using a network disk is not supported and will result in the server being unable to start.

BACKUP OF PREFERENCE DIRECTORY

Add the subfolder `db` in the Model Manager server preference directory to your backup routines. The folder contains accounts, database configurations, and other server preferences that may be time consuming to recreate in case of a server disk crash or other failure. A periodic file system copy of the folder is sufficient.

When restoring a Model Manager server after a failure, copy the `db` folder from its backup location to its location in the preference directory before starting the server.

SERVER COMPONENTS

A Model Manager server uses three separate server components for the storage and search indexing of your data. A default installation of Model Manager server includes all three as *managed* server components, with the server handling the start and stop of component subprocesses and data backups. You also have the option of replacing some or all of these managed server components with *external* server components maintained by your organization.

The Model Manager server uses a relational database management system for storing data in a SQL database. The installation includes the PostgreSQL® relational database management system as an optional managed server component. You can also configure the server to use an external relational database management system — see External SQL Databases for a list of supported SQL database servers. The external SQL

database server can either run on the same computer as the Model Manager server or on a different computer within the same network.

For the search functionality of the Model Manager, a Model Manager server uses the Apache Solr™ enterprise search platform. The installation includes this platform as an optional managed server component. You can also configure a Model Manager server to use an external Apache Solr™ installation provided by your organization — see External Search Indexes. As for the SQL database server, the external Apache Solr™ server can either run on the same computer as the Model Manager server or on a different computer within the network.

Binary and text data that is too large to be stored inside the SQL database is stored directly on the file system inside a dedicated directory. This directory is registered with the Model Manager server as a server component. The backup of the files in the directory can either be managed by the Model Manager server or it can be handled by external backup routines maintained by your organization.

During installation, you will be given the option of letting the Model Manager server automatically set up a new Model Manager database using managed server components configured with default settings on first launch of the server. This includes, for example, data directories and backup directories located on the server computer's file system. Depending on your expected user base and IT infrastructure, you may want to modify these default settings once the Model Manager server is up and running — either by moving the default database or by adding a new database via the Model Manager server's web interface. You can also skip this automatic creation altogether, thereby enabling you to take full control of where data directories and backup directories should be located once the server is up and running.

When describing a Model Manager server and its server components there are inevitable terminology collisions for the words "database" and "server". A Model Manager server uses a SQL database server as an independent server component. A Model Manager database uses a SQL database in such a SQL database server to store a *subset* of its data. A SQL database server can contain multiple SQL databases, each used by a different Model Manager database.

- Adding the Model Manager Server Database on First Login
- Moving a Model Manager Server Database

DATA DIRECTORIES

The three server components of a Model Manager database each uses its own data directory:

- The data directory for the SQL database. This is the main storage for the version control management of models and data files in repositories and branches, the version-controlled assets in asset libraries, and all user management and access control functionality. This data will inevitably grow over time, but, with a reasonably sized disk drive, it should still be fine to keep on the same computer as the Model Manager server installation (although not necessarily the same disk drive). The size is also kept manageable by the fact that large binary and text data is stored *externally* to the SQL database, while the SQL database itself mainly stores descriptive, structural, and administrative *metadata* for this binary and text data.
- The data directory for the search index server. This data is automatically extracted from that which is stored in the SQL database and, as such, requires no backup. The size is typically much smaller than the other two data directories and there is generally no issue in placing it on the same computer as the Model Manager server installation.
- The so-called *resources directory* for storing large binary and text data associated with models, data files, and assets. Large here typically means above a few hundred kilobytes. This includes:
  - Binary data for geometries and meshes, computed solutions, and result plots associated with your models.
  - CAD data, interpolation functions, and other data files used as input for your models.
  - Documentation files, presentations, videos, and other supplementary files linked to models via the asset management system.

  The resources directory is expected to be the largest of the three and you may want to offload it to a dedicated file server whose file system is mounted by the server computer running the Model Manager server — see Examples of Server Setups.

The location of the data directories used by the default Model Manager server database optionally set up during installation depends on the computer account running the server — see Default Data Directories.

When using managed server components, it is recommended to store the data directory for the SQL database server on a physical disk for best performance. For Linux, storing the data directory on a network disk using NFS mounted in hard mode

is supported. Other network disk configurations are not supported. Placing the data directory for the search index server on a network disk is not supported for any configuration.

Support for storing the data directories of managed server components on a network disk is as follows:

- SQL database server — only supported on Linux® using NFS mounted in hard mode.
- Search index server — never supported.
- Resources directory — always supported.

If placing the data directory of the SQL database server or the search index server on the same computer as the Model Manager server is not an option, the recommended approach is to use external server components. You can install a SQL database server on another computer and then connect to that server from the Model Manager server over the network. Connecting to an external SQL database server may also prove useful if the IT infrastructure of your organization already supports managing SQL databases, including handling their backups. You can also install and connect to an Apache Solr™ enterprise search platform on another computer. Finally, the resources directory may be placed on an external file server as long as the Model Manager server can mount it as a network disk.

The computer account running the Model Manager server must have the necessary access permissions to connect to an external SQL database server, search index server, or file server. This is typically *not* the case for the predefined **LocalService** account in Windows®.

Databases with External Server Components

Files stored in the resources data directory always pass through the Model Manager server when opened or saved from a COMSOL Multiphysics session — this to ensure data integrity and that necessary permissions are verified. There is therefore little to gain from placing a file server containing the resources data directory geographically near your users if the Model Manager server itself, for example, is not.

Default Data Directories

The location of the data directories used by the default Model Manager server database optionally set up during installation depends on the computer account running the Model Manager server process. For the predefined **NetworkService** account in Windows®, for example, the directories are located inside:

```
C:\Windows\ServiceProfiles\NetworkService\AppData\Local\COMSOL\
ModelManager\ManagedDatabases\default-managed
```

with subdirectories:

- `data` — the data directory of the managed SQL database server.
- `index` — the data directory of the managed search index server.
- `resources` — the managed resources directory.

Replace `NetworkService` with `LocalService` in the directory path for the predefined **LocalService** account in Windows®.

For Linux®, the directories are located inside:

```
<user>/.comsolmodelmanagerdata/manageddatabases/default-managed
```

and for macOS:

```
<user>/Library/Application Support/COMSOL/ModelManager/
ManagedDatabases/default-managed
```

Moving a Model Manager Server Database

BACKUP OF DATA DIRECTORIES

A Model Manager server database using managed server components requires backup for the data directory of the SQL database server and for the resources data directory. The data directory of the search index server requires no backup. The default Model Manager database optionally set up during installation is already configured so that the Model Manager server handles this backup — see Managed Backups for further details.

You can also let external backup software provided by your organization handle the backup of the two data directories. This may be useful when you want to save disk space on the computer running the Model Manager server and mounting an external file server — as described for Managed Backups — is not feasible. The simplest option is to periodically shut down the Model Manager server and do a file system copy of

both directories to an external location. It is important to copy both directories as they could otherwise become inconsistent with respect to each other.

To get a consistent backup for this option, the Model Manager server must be shut down before copying the data directories. Copying the data manually via a system file explorer while the server is running — with different files captured at different points in time — will lead to data corruption.

If shutting down the server is impractical, it is also possible to do a backup while the service is running as long as the external backup software, as well as the file system volume containing the data directories, supports *frozen file system snapshots*. See the official documentation for the PostgreSQL® relational database management system for further details on file system level backups.

If your external backup software do supports frozen file system snapshots, but the data directory of SQL database server and the resources data directory reside on different disk volumes, it is recommended to order your backups so that the SQL database server's data directory is backed up first.

Backup and Restore of a Model Manager Server Database

MANAGED BACKUPS

The backup directories for the managed server components used by the default Model Manager database optionally set up during installation are located on the same disk drive as the data directories for these components — see Default Backup Directories. This backup configuration makes it possible to recover data after an accidental deletion in the database but is still vulnerable to a server disk crash. You are therefore strongly recommended to do one of the following options — or a combination thereof — when using a managed Model Manager database:

- Place the managed backup directories on another disk drive or on a mounted file server that the computer account running the Model Manager server has read and write access to.
- Keep the current location for the managed backup directories but let external backup software take periodic backups of these backup directories themselves.

The best alternative typically depends on your organization's IT infrastructure.

For the second option — taking a *secondary* backup of the managed backup directories themselves — there is no strict requirement of shutting down the Model Manager server or using frozen file system snapshots. The server writes the managed backup of the PostgreSQL® relational database management system as an ordered sequence of fixed-size files such that, once a specific file is completed, a new file is begun and the previous file will not be further modified. On the off-chance that the external backup software is running at the exact same moment that the server is writing to the most recent such file, you would only risk potentially missing data in this particular file. This could be rectified the next time the external backup software runs.

The size of the backup directory of a managed SQL database server is expected to be the same order of magnitude as the data directory itself. For a managed resources directory, the backup directory has exactly the same size unless any resources have been permanently deleted in the Model Manager database — such deletions are never propagated to the backup directory by design.

Default Backup Directories

The backup directories for the managed server components used by the default Model Manager server database optionally set up during installation are located on the same disk drive as the Default Data Directories. For the predefined **NetworkService** account in Windows®, for example, the directories are located inside:

```
C:\Windows\ServiceProfiles\NetworkService\AppData\Local\COMSOL\
ModelManager\ManagedDatabases\default-managed-backup
```

with subdirectories:

- `data_backup` — backup directory for the managed SQL database server.
- `resources_backup` — backup directory for the managed resources directory.

Replace `NetworkService` with `LocalService` in the directory path for the predefined **LocalService** account in Windows®.

For Linux®, the directories are located inside:

```
<user>/.comsolmodelmanagerdata/manageddatabases
/default-managed-backup
```

and for macOS:

```
<user>/Library/Application Support/COMSOL/ModelManager/
ManagedDatabases/default-managed-backup
```

SERVER PROCESSES AND MEMORY FOOTPRINT

The memory footprint of the main Model Manager server process is expected to be small and will typically not be a deciding factor for the overall system requirements of the server computer. When using managed server components for a Model Manager server database, there are two groups of subprocesses started on the server computer:

- Managed SQL database server processes.
- Managed search index server processes.

The memory footprint of both of these two groups will scale with the size of the database. A recommendation is to start with at least a few gigabytes of RAM and then follow up over time as your database grows.

EXAMPLES OF SERVER SETUPS

A few examples of server setups for a Model Manager server installation, in order of increasing complexity, is summarized as follows:

Example 1 — Single Server

A Model Manager server with a fully managed Model Manager database. The main Model Manager server process, as well as subprocesses for the managed SQL database server and managed search index server, all run on a single server computer. The data directories are all located on the server computer's file system. The backup directories are located on a physical disk drive different from that of the data directories.

Apart from the location of the backup directories, this is the setup used by the default managed database optionally added when installing a Model Manager server. It is also the recommended setup when getting started with a Model Manager server.

Example 2 — Backup File Server

A Model Manager server with a fully managed Model Manager database. The main Model Manager server process, as well as subprocesses for the managed SQL database server and managed search index server, all run on a single server computer. The data directories are all located on the server computer's file system. The backup directories are located on a separate file server mounted by the server computer.

Example 3 — Managed Resources File Server

A Model Manager server with a fully managed Model Manager database. The main Model Manager server process, as well as subprocesses for the managed SQL database server and managed search index server, all run on a single server computer. The data directories of the managed SQL database server and the managed search index server are located on the server computer's file system. The managed resources directory is

located on a separate file server and the backup directories are located on a second separate file server, both mounted by the server computer.

Example 4 — External SQL Database Server

A Model Manager server with a partially managed Model Manager database. The main Model Manager server process and the managed search index server subprocess run on the main server computer. The Model Manager server connects to an external SQL database server running on a second server computer. The data directory for the managed search index server is located on the main server computer, the data directory for the external SQL database server is located on the second server computer, and the managed resources directory is located on a separate file server. The backup directory for the managed resources directory is located on a second separate file server, while the backup of the external SQL database server is handled by your IT department. This option gives the most flexibility to, for example, scale up memory and disk size over time.

Installing in Windows

Start the Model Manager server installation by using the media that you have received or by using an internet download. In the installer, after selecting your preferred language, choose **New COMSOL Model Manager Server 6.4 Installation**.

COMSOL Model Manager Server uses a separate installer than the one used when installing COMSOL Multiphysics® or COMSOL Server™.

LICENSE

In the next step, **License**, select the format of your license in the **License format** list under **License information**. For the **License file** option, write the file path to the license file that you have received from your COMSOL representative in the **License file** field. Click **Browse** to browse to and choose another file path. You can also use the **Port number** and **Hostname** option or the **Three-server redundancy** option if your license administrator has given you these details.

At this point, the installer detects the type of license used. The following instructions assume that the license used during installation corresponds to a Model Manager server license.

PRODUCTS

In the next step, **Products**, select the installation location and which software products and components you want to install. All products and components are selected by default.

- **Model Manager Server** — installs the Model Manager server software components.
- **Managed PostgreSQL®** — installs software components for a PostgreSQL® database system managed by the Model Manager server. Clear the checkbox if you only want to use an external relational database management system provided by your organization.
- **Managed Apache Solr™** — installs software components for an Apache Solr™ search platform managed by the Model Manager server. Clear the checkbox if you only want to use an external Apache Solr™ installation provided by your organization.
- **License Manager** — installs the COMSOL License Manager. You only need to install this on the computer where you would like to run the license manager.

OPTIONS

In the **Options** step:

- Select the **Create COMSOL Model Manager Server 6.4 folder on Windows Start Menu** checkbox to install **Start** menu shortcuts (Windows® operating system only).
- Select the **Add Windows Firewall rules for COMSOL programs** checkbox to add Windows Firewall rules.
- Select the **Check for updates after installation** checkbox to enable checking for updates after installation.
- In the **Java runtime environment** list, select **Built-in** to use the default Java runtime included with the Model Manager server installation. Select **Custom** if you prefer to use another Java runtime that you have licensed and installed.

LICENSE MANAGER

The **License Manager** step appears if the installer installed the FlexNet® license manager and your computer has been designated to run the license server. If this step does not appear, you can manage the license server using `LMTOOLS`. This step contains the following options:

- The **Install license manager as a Windows service** checkbox is selected by default; if you clear the checkbox, the license manager will not be available as a Windows® service.

- The **Path to the debug log file** field contains a file path to the location of the license manager debug log file. The default is `C:\ProgramData\COMSOL\comsol64.log`. Click **Browse** to choose another file path.
- The **Service name** field shows the service name, `LMCOMSOL`, for information only.
- Under **Additional license manager options**, you can select any of the following checkboxes:
  - Select the **Allow the lmdown command to be executed only from this computer** checkbox to restrict the execution of the `lmdown` command, which you can use to shut down the license manager, to this computer only.
  - Select the **Disable the lmdown command** checkbox to make the `lmdown` command unavailable.
  - Select the **Disable the lmremove command** checkbox to make the `lmremove` command, which you can use to remove a user's license, unavailable.

SERVER

The **Server** step makes it possible to set up the Model Manager server as a Windows® service. You can also configure an initial administrator account, which you can use to log in to the Model Manager server web interface to continue the server setup once the installation has finished.

Basic Settings

In the **Default Model Manager server port** field, write the port number that the server will use. The default is `8181`.

Windows Service Settings

There are two ways to install the Model Manager server in Windows®. If you select the **Install the Model Manager server as a Windows service** checkbox (the default), the Model Manager server is then installed as a Windows® service. Otherwise, the Model Manager server is installed as a regular executable.

Use the **Startup** list to configure how to start the Model Manager server when installed as a Windows® service:

- **Disabled** — the service is disabled.
- **Manual** — the service will not be started after the installation. You need to start the service manually.

- **Automatic** — the service is configured to start automatically when the host computer boots or restarts. This is the default choice.
- **Automatic (Delayed Start)** — the service starts automatically but is delayed until all automatic-start threads have finished starting.

The **Automatic** setting provides the highest availability to users of the installed server.

If you choose **Disabled** or **Manual**, you can enable or start the Model Manager server service from the command line or by using the **Manage Local Services** shortcut installed on the **Start** menu under `COMSOL Launchers`. The same configuration options are also available in the snap-in **Services** in the **Microsoft Management Console** — search for `services` from the Windows® **Control Panel**.

In the **Service account** list, select the computer account that runs the service. You can choose the predefined **LocalService**, **LocalSystem**, or **NetworkService** accounts in Windows®, which have no password. When using other accounts, choose **Custom** and provide a username and password. By default, the installer selects the predefined **LocalService** account.

The **LocalService** account has limited privileges intended for running local services. It is the recommended service account if you intend to set up a Model Manager server database using a single server computer — see Examples of Server Setups. If you, for example, intend to mount a file server or connect to an external SQL database server, you must use a service account with network access privileges — for example, the predefined **NetworkService** account or a custom account.

The PostgreSQL® database system does not support running under a user account with administrative privileges on Windows® — that is, a member of the Administrators or Power Users group. If you choose to not install this software component, you are still strongly recommended to use an account with less privileges — see also Security.

Consult the documentation that came with the operating system for more information about service accounts.

Default Local Administrative User

Select the **Create a default local administrative user** checkbox to set a username and a temporary password for an administrator account. You will typically use that account to log in to the Model Manager server web interface after the installation finishes to continue the setup of the server.

Only clear the **Create a default local administrative user** checkbox if you are repairing or updating a previous installation for which an administrator account already exists.

Model Manager Server Database

Under **Add a Model Manager server database**, select **Automatic** if the Model Manager server should automatically create a new database during startup unless an existing database is found. Select **Custom** if you prefer to create the database manually via the Model Manager server web interface once the server is up and running — see Adding the Model Manager Server Database on First Login.

For the **Automatic** option, Model Manager server will create the new database in a default file system location — typically on the same disk drive as the Model Manager server installation itself. As discussed in the Installation Planning section, this may not be ideal long-term for the following two reasons:

- The disk may not meet the expected size requirements of your simulation data.
- The data directories and the backup directories of the database are located on the same disk, thereby making this setup vulnerable to a future disk failure.

You can always move the automatically created database to a new location after the server has been installed, although this may involve a fairly large number of steps — see Moving a Model Manager Server Database.

Select **Automatic** if you are mainly interested in quickly getting started with a Model Manager server database to, for example, learn in its various features. Select **Custom** if you already have an idea of how you want to set up the server database long term — see Examples of Server Setups.

	Only the **Custom** option is available if you cleared the **Managed PostgreSQL®** or the **Managed Apache Solr™** checkbox in the **Products** step.

	No new database will be created if a database from a previous Model Manager server installation is found during startup. See also Migrating From an Older Installation.

Troubleshooting

When you click **Next** from the **Server** step, the installer will verify some of the given settings, including employing a test service to check that the given service account details are valid and that the service account has the right permissions to access the installation directories. The following are some common warning and error messages that may arise and suggestions on how to address them:

The given service account does not seem able to start services due to a logon failure. Please verify that it has the right to log on as a service. The installer failed to start the test service. To verify that a custom user account has the right to log on as a service, you can check the security settings for the **Log on as a service** policy under **Control Panel** > **Administrative Tools** > **Local Security Policy** > **Local Security Settings** > **Local Policies** > **User Rights Assignments** in Windows®.

The installation directory is not accessible. Please verify that the directory is correct and that the service account has access to this path. The service launcher failed to locate the test service when starting it. The service account either does not see that location or does not have read permissions. Commonly, this is caused by using an installation location that either has security permissions that do not include the service account, or by the location being on a network share that is not mounted by the service account.

The given service account is invalid or does not exist or the password is wrong. The test service could not be installed or started due to a problem with the service account. Verify that the account details are correct.

Failed to verify the service account. An unexpected error happened. If you believe that the installation settings are correct, you can proceed with the installation. Further details are given in the `comsolsetup.log` file after the installation.

No administrative user has been defined. You did not specify an administrator account under **Default local administrative user**. In this configuration, it will not be possible to

log in to Model Manager server unless the server has already been configured with accounts after a previous installation.

INSTALL

The **Install** step shows a list of the software products and components that will be installed. Click the **Install** button to begin the installation.

FINISH

The last **Finish** step is shown when the installation has finished. You can view an installation log file in case, for example, there were warnings or errors during the installation.

If you selected to install as a Windows® service that starts automatically, you can now verify that you can access the web interface of the Model Manager server — see Accessing the Web Interface. Otherwise, see Starting a Model Manager Server.

Automated Installation in Windows

You can install a Model Manager server using an automated installation process with minimal user interaction. This method requires installation from a DVD or DVD image. An *answer file* then responds to questions while the installer is running. The answer file is a text file that contains predefined settings that the Model Manager server installer uses. A template answer file, `setupconfig.ini`, with detailed usage information is available on the DVD. Create a copy of this file and customize it with a text editor.

When the answer file is ready, start the installation by running

```
<DVD path>\setup.exe -s <answer file path>
```

where *<answer file path>* is the full path to your answer file.

AUTOMATED REMOVAL (UNINSTALLATION)

You can also use an answer file to uninstall a Model Manager server. Edit the answer file in a text editor by setting

```
installmode = uninstall
installdir = <Installation directory>
```

where *<Installation directory>* is the path to your Model Manager server installation. Run

```
<Installation directory>\bin\win64\setup.exe -s <answer file path>
```

where <*answer file path*> is the full path to your answer file. If you are running this command in a terminal window, make sure that the current working directory is outside the Model Manager server installation directory.

Installing in Linux and macOS

Installing in the Linux® and macOS operating systems is similar to Installing in Windows. The **Server** step contains Basic Settings, the option of creating a Default Local Administrative User, and a default Model Manager Server Database. See Starting a Model Manager Server for more information on how to automatically launch a Model Manager server after installation on Linux® and macOS.

For security reasons, it is not recommended to use an account with administrative privileges to run a Model Manager server. The Apache Solr™ search platform and the PostgreSQL® database system do not support running as a root user on either Linux® or macOS.

AUTOMATED INSTALLATION IN LINUX

You can install a Model Manager server in Linux® using an automated installation process similar to that on Windows® — see Automated Installation in Windows. When you have created an answer file, start the installation by running

```
<Drive path>/setup -s <answer file path>
```

where <*Drive path*> is the media mount point for the DVD or DVD image and <*answer file path*> is the full path to your answer file.

AUTOMATED INSTALLATION IN MACOS

To perform an automated installation in macOS using an answer file:

1. Open the DMG file for the Model Manager server installer in Finder.
2. Prepare an answer file by coping the template file `setupconfig.ini` to a local directory and customizing it as needed.
3. Right-click **COMSOL Model Manager Server 6.4 Installer** and select **Show Package Contents**.
4. Navigate to **Contents > Resources** and locate the setup executable.
5. Open a terminal window.

6 Click and drag the setup executable to the terminal window.

   The full path should be copied to the command prompt.

7 In the terminal, add `-s` and the full path of your `setupconfig.ini` file to the command, with spaces in between. If needed, it should be possible to click and drag the `setupconfig.ini` file to the terminal to input the path.

8 Run the command, which should now resemble something like this:

```
/Volumes/COMSOL64_mms_full_macarm64/COMSOL\ Model\ Manager\
Server\ 6.4\ Installer.app/Contents/Resources/setup -s /Users/
username/Desktop/setupconfig.ini
```

   or this:

```
/Volumes/COMSOL64_mms_full_maci64/COMSOL\ Model\ Manager\ Server\
6.4\ Installer.app/Contents/Resources/setup -s /Users/username/
Desktop/setupconfig.ini
```

   depending on the CPU architecture.

Modifying an Installation

To modify an existing Model Manager server installation — for example, to change the default port or reset the administrator password — follow these steps:

1 Stop the Model Manager server.

   If installed as a Windows® service (default for Windows®), you can stop the Model Manager server service using the **Stop Model Manager Server** shortcut installed on the **Start** menu under `COMSOL Launchers`.

2 Start the installer for the Model Manager server and select a language.

3 Choose **Add/Remove Products and Reinstall**. If a dialog is opened, browse to the installation directory.

4 Specify the new configuration in the steps that follow, similarly as for a new installation.

   You can clear the **Create default local administrative user** checkbox if you have already added an administrative user, ignoring the warning that no administrative user has been defined when clicking **Next**.

5 When the installer has finished, start the Model Manager server again.

See Starting a Model Manager Server for instructions on how to start and stop the server on different platforms.

Irrespective of what you select under **Add a Model Manager server database** in the **Server** step, no new database will be created if a database from the existing installation is found during startup.

RESETTING THE DEFAULT ADMINISTRATOR PASSWORD

You can reset the default administrator password by following the steps for modifying an existing Model Manager server installation. In the **Server** step, select the **Create default local administrative user** checkbox and write a new password to reset any existing password.

UNINSTALLING

To uninstall an existing Model Manager server installation, follow these steps:

1. Stop the Model Manager server.

   If installed as a Windows® service (default for Windows®), you can stop the Model Manager server service using the **Stop Model Manager Server** shortcut installed on the **Start** menu under `COMSOL Launchers`.
2. Start the Model Manager server installer and select a language.
3. Choose **Uninstall COMSOL Model Manager Server 6.4**. If a dialog is opened, browse to the installation directory.
4. Click the **Uninstall** button.

Uninstalling a Model Manager server will *not* delete any Model Manager databases.

Migrating From an Older Installation

To migrate from an older version to a newer version of a Model Manager server:

1. Stop the currently running Model Manager server.

   If installed as a Windows® service (default for Windows®), you can stop the Model Manager server service using the **Stop Model Manager Server** shortcut installed on the **Start** menu under `COMSOL Launchers`. Also make sure that the Windows® service is not set to be started automatically — that is, set the startup type to **Disabled**.

   For a systemd service installed on Linux®, make sure to stop and disable the old service before installing the new service.

2 Install **COMSOL Model Manager Server 6.4**.

3 Once you have verified that the **COMSOL Model Manager Server 6.4** has successfully started, uninstall the old version of the server if desired.

The Model Manager databases hosted by the old Model Manager server will now be available in the new server.

You are strongly recommended to at least stop any older installations of Model Manager server before installing a newer version to not risk any unwanted cross-talk in the data directories. When stopping an older Model Manager server installed as a Windows® service, make sure that the service is not set to be started automatically. When stopping an older systemd service installed on Linux®, make sure that the service is also disabled.

When a Model Manager server starts for the first time, it tries to locate a Model Manager server Preference Directory belonging to an older installation of the server. If such a preference directory is found, all server system settings are migrated to the new installation's preference directory. This includes, for example, any account settings and Model Manager database configurations set up for the older server.

You can also manually migrate the server system settings from the older installation. This may be necessary, for example, if the new installation runs under a different user account than the old one.

1 Stop the newly installed Model Manager server.

2 In the Preference Directory of the new installation, delete the subfolder `db`. This removes the default server system settings set for the new installation.

3 Copy the subfolder `db` from the preference directory of the old installation to the preference directory of the new installation, thereby transferring all server system settings between the installations.

4 Restart the newly installed Model Manager server.

If installed as a Windows® service (default for Windows®), you can start the Model Manager server service using the **Start Model Manager Server** shortcut installed on the **Start** menu under `COMSOL Launchers`.

5 Log in to its web interface using an administrator account configured for the old installation. Re-install the server if you do not remember the credentials for such an

account — see Resetting the Default Administrator Password. Verify that the server system settings were successfully transferred.

	See Moving an Installation if migrating from an older installation found on another computer.

Unlike the preferences directory, the data directories and backup directories of any configured databases always remain in the same file system location as they were prior to installation when installing a new version of Model Manager server. In other words, these directories are *not* copied. This means that an older installation of Model Manager server may potentially read and write in the same data directories as the newer installation unless the older server is first stopped.

	Always make sure that you do not have multiple running installations of Model Manager server accessing the same Model Manager database.

	Backward Compatibility for Server Databases

Moving an Installation

To move an existing Model Manager server installation — for example between two server computers — follow these steps:

1. Obtain the locations of the data directories and backup directories of all managed Model Manager server databases for the current Model Manager server installation.

   See Default Data Directories and Default Backup Directories for the locations of these directories for the default managed database added during installation. You can also find their locations in the **System** administration area of the Model Manager server web interface — see Managed Server Components for further details.

2. Stop the current installation of Model Manager server.

   If installed as a Windows® service (default for Windows®), you can stop the Model Manager server service using the **Stop Model Manager Server** shortcut installed on the **Start** menu under `COMSOL Launchers`.

3. Move or copy the data directories and backup directories to their new locations.

4 Install Model Manager server on the new computer. Make sure to select **Custom** under **Add a Model Manager server database** in the **Server** step of the installation to avoid ending up with a new, empty, database.

   The new installation can be a newer version of Model Manager server than the current installation.

5 Stop the newly installed Model Manager server.

6 In the Preference Directory of the new installation, delete the subfolder `db`. This removes the default server system settings automatically added during installation.

7 Copy the subfolder `db` from the preference directory of the old installation to the preference directory of the new installation, thereby transferring all server system settings between the installations.

8 Restart the newly installed Model Manager server and log in to its web interface using an administrator account configured for the old installation. Re-install the server if you do not remember the credentials for such an account — see Resetting the Default Administrator Password.

   If installed as a Windows® service (default for Windows®), you can start the Model Manager server service using the **Start Model Manager Server** shortcut installed on the **Start** menu under `COMSOL Launchers`.

9 At this point, any Model Manager server database will likely not be accessible since the Model Manager server is configured to use the locations of the old installation for the data directories and the backup directories. In the **System** administration area:

   a Click **SQL Database Servers** to open the **Managed SQL Database Servers** page. Click on a managed SQL database server to open its details page. You can ignore the expected warning message about a missing data directory. Click **Edit** and update the locations in the **Data directory** and **Backup directory** fields. Click **Save** and then **Start**.

   b Click **Resources Directories** to open the **Managed Resources Directories** page. Click on a managed resources directory to open its details page. Click **Edit** and update the locations in the **Data directory** and **Backup directory** fields. Click **Save**.

   c Click **Search Index Servers** to open the **Managed Search Index Servers** page. Click on a managed search index server to open its details page. You can ignore the expected warning message about a missing data directory. Click **Edit** and update the location in the **Data directory** field. Click **Save** and then **Start**.

10 Click the COMSOL logo in the upper left corner and verify that you can now access the database.

!	Moving a Model Manager server database using a managed SQL database server between Windows® and Linux® by copying the data directory or backup directory is not supported due to incompatible storage formats.

Starting a Model Manager Server

In this section, you will learn how to start a Model Manager server in the Windows®, Linux®, or macOS operating systems.

In this section:

Starting as a Windows Service

By default, a Model Manager server is installed as a Windows® service that is set to start automatically. In this case, a link to the Model Manager server web interface is installed on the **Start** menu as **Model Manager Server**. See Accessing the Web Interface for more information about accessing the web interface and, for example, continuing the setup of a fresh installation.

To manually start or stop the Model Manager server service, use the **Start Model Manager Server** or **Stop Model Manager Server** shortcuts installed on the **Start** menu under `COMSOL Launchers`. To configure its startup options, use the **Manage Local Services** shortcut. The same configuration options are also available in the snap-in **Services** in the **Microsoft Management Console**. Search for `services` from the Windows® **Control Panel**.

Before starting a Model Manager server, you need to start a FlexNet® license manager. The license manager can be installed together with a Model Manager server — see Installing a Model Manager Server.

Starting Manually in Windows

If you cleared the **Install Model Manager Server as a Windows service** checkbox during installation, you can start a Model Manager server manually by doing one of the following:

- Click the **COMSOL Model Manager Server 6.4** shortcut installed on the **Start** menu.
- Double-click the Windows® executable in a file browser.
- Run the Windows® executable from a command window.

The Model Manager server executable is located at:

```
<Installation directory>\bin\win64\comsolmodelmanagerserver.exe
```

An example of a typical <*Installation directory*> is:

```
C:\Program Files\COMSOL\COMSOL64\ModelManagerServer
```

For all methods of starting a Model Manager server, the command window displays a short message that the server has started and which port is being used.

Press Ctrl+C to stop the server. A short message is displayed informing that the server is stopping. Exit the command window once the window returns to accept user input again.

When a Model Manager server has been installed as a Windows® service, the shortcut for starting manually is not available on the **Start** menu. It is not recommended to start a Model Manager server manually via its executable when the service is running. Since the service typically runs under a special system user account, it does not share its configuration settings with a Model Manager server that was started manually.

Installation Directory

Starting in Linux

To start a Model Manager server manually on Linux®, type

```
<Installation directory>/bin/comsol modelmanagerserver
```

To stop the server, press Ctrl+C in the terminal window where it was started.

The Apache Solr™ search platform and the PostgreSQL® database system do not support running as a root user on Linux®.

SYSTEMD SERVICE

Use the following instructions to start a Model Manager server in a Linux® version that includes `systemd`:

1. Create a file:

   `/usr/lib/systemd/system/comsolmodelmanagerserver64.service`

   with content similar to:

```
[Unit]
Description=COMSOL Model Manager server 6.4
Wants=network.target network-online.target
After=network.target network-online.target

[Service]
Type=exec
User=comsoluser
Group=comsolgroup
TimeoutSec=300
ProtectHome=off
ExecStart=/usr/local/comsol64/modelmanagerserver/bin/comsol
modelmanagerservice

[Install]
WantedBy=multi-user.target
```

   You only need to replace the values for `User` and `Group`.

2. Activate the service via:

```
systemctl enable /usr/lib/systemd/system/
comsolmodelmanagerserver64.service
```

3. Start the service via:

```
systemctl start comsolmodelmanagerserver64
```

To stop the Model Manager server service cleanly, run:

```
systemctl stop comsolmodelmanagerserver64
```

If you want to enable transport layer security (TLS) for secure connections via HTTPS on port 443, you could add the line `AmbientCapabilities=CAP_NET_BIND_SERVICE` to the `[Service]` section of the `comsolmodelmanagerserver62.service` file. This will allow the Model Manager server to open that port in case the Linux® operating system is otherwise restricting opening of such low-numbered ports. See Adding a Connector With TLS for further details.

Starting in macOS

To start a Model Manager server manually, type

```
<Installation directory>/bin/comsol modelmanagerserver
```

To stop the server, press Ctrl+C in the terminal window where it was started.

The Apache Solr™ search platform and the PostgreSQL® database system do not support running as a root user on macOS.

Firewalls

You must open up firewalls that exist between the server and your users. Open up for incoming TCP connections to the port given during installation (by default, `8181`) or controlled by the `-port` command option — see Command Options. To improve security, you can reduce the IP address range to known potential IP addresses for your users. If you are running a Model Manager server behind a reverse proxy, only open up the port of the reverse proxy instead — see Running Behind a Reverse Proxy. If you have added a connector with TLS enabled, only open up the port of that connector instead — see Adding a Connector With TLS.

Internally, a Model Manager server may launch subprocesses and communicate with them on various ports. These additional ports do not need to be open in the firewall for users of the Model Manager server.

Setting Up the Model Manager Server for Secure Connections

You are strongly recommended to set up the Model Manager server for secure connections using one of the following two options — see also Security:

- Place the Model Manager server behind a reverse proxy with TLS encryption configured. This way, no particular configuration of the Model Manager server itself is required. The communication between the reverse proxy and the Model Manager server will not be encrypted. See also Running Behind a Reverse Proxy for more information.
- Add a TLS-enabled connector with a certificate obtained from a trusted certificate authority (CA) to the Model Manager server — see Adding a Connector With TLS.

To connect to a Model Manager server using HTTPS, select the **Require secure connection** checkbox in COMSOL Multiphysics.

Running Behind a Reverse Proxy

In some cases, it is advantageous to use a reverse proxy as an intermediary between the Model Manager server and its users. Use cases of a reverse proxy include:

- Providing many web services with the same hostname — for example, serving the Model Manager server at `example.com/modelmanagerserver` and a webmail server at `example.com/webmail`.
- Offloading TLS encryption onto the reverse proxy, either for using hardware-accelerated encryption or to simply avoid configuring encryption for each web service separately.
- Firewall features that protect the web service from attacks.
- Integration with single sign-on systems for authentication — see Proxy Authentication.

To support serving the Model Manager server at the path `/modelmanagerserver`, as in the preceding example, the reverse proxy must be configured to pass any regular HTTP requests to, for example, `example.com/modelmanagerserver/foo` on to the Model Manager server as `modelmanagerserver.com:8181/foo`. Additionally, the reverse proxy should be configured to rewrite the cookie path `/api` set by the Model Manager server to `/modelmanagerserver/api`.

Any reverse proxy that supports HTTP can be used with the Model Manager server. Two common reverse proxies are Apache `mod_proxy` and NGINX®. See the documentation of the reverse proxy software for how to set it up to forward requests.

You can use the `ProxyPassReverseCookiePath` directive in Apache `mod_proxy` and the `proxy_cookie_path` directive in NGINX® to rewrite the cookie path.

TLS ENCRYPTION

By configuring the reverse proxy to serve HTTPS, the integrity and confidentiality of the communication between the client and the reverse proxy is protected by the TLS protocol. This is the recommended way to host the Model Manager server on any network where cleartext traffic could be intercepted.

Command Options

You can specify optional command options when starting a Model Manager server using the syntax

```
comsolmodelmanagerserver.exe [<options>]
```

on Windows® and

```
comsol modelmanagerserver [<options>]
```

on Linux® and macOS platforms. See Table 2-1 for various command options available for the server.

TABLE 2-1: COMMAND OPTIONS FOR A MODEL MANAGER SERVER.

OPTION	DESCRIPTION
`-c`	Path to license file.
`-comsolinifile`	Path to `ini`-file to use when launching.
`-h, -help`	Show help message.
`-port`	Port number used by the server.
`-prefsdir`	Path to preference directory.
`-tmpdir`	Path to temporary file directory.
`-v, -version`	Show version information.

The default `ini`-file is found in the installation directory for the Model Manager server:

- Windows®: `<`*Installation directory*`>\bin\win64`
- Linux®: `<`*Installation directory*`>/bin/glnxa64`
- macOS: `<`*Installation directory*`>/bin/maci64`

The filename is `comsolmodelmanagerservice.ini` if running as a Windows® service. Otherwise, it is `comsolmodelmanagerserver.ini`.

You can add to the `ini`-file any command option in Table 2-1 that specify a configuration setting — insert `Dcs.` between the dash and the option name and separate the option value with an equals-sign. To use a custom preference directory, for example, add the line:

```
-Dcs.prefsdir=<Path to preference directory>
```

Remember to restart the Model Manager server after saving the file for any new configuration setting to apply.

Security

The following guidelines summarize the best practices for running the Model Manager server in a secure way:

- Do not give unprivileged users in the organization shell access to the system where the Model Manager server runs. Although the file system directories and processes of the Model Manager server are protected from interference by other users on the system through operating system permissions, it is general best practice to restrict shell access on production servers.
- Set up the Model Manager server to use TLS (via HTTPS) when connecting from web browsers and COMSOL Multiphysics. This increases the protection of passwords sent when logging in and reduces the risk of data leaks. The easiest way to get transport layer security is to use a reverse proxy with a certificate, which might already have been set up for other systems. Alternatively, you can add a TLS-enabled connector with a certificate to the Model Manager server — see Adding a Connector With TLS.
- Use a low-privilege account when running the Model Manager server process. This reduces the risk of privilege escalation attacks to the system. See Installing a Model Manager Server for more information.
- Configure the firewall of the computer or network segment where the Model Manager server runs to only expose the main port of the Model Manager server to the outside. If running the Model Manager server behind a reverse proxy, only expose the port of the reverse proxy to the outside instead.
- If you expose the Model Manager server to the internet, make sure to operate the Model Manager server on a network isolated from your regular corporate network. One option for implementing this is to install the Model Manager server on a computer in a so-called *DMZ network* located between the internet and your regular corporate network. Another option is to install the Model Manager server utilizing a cloud service provider.
- Ensure that the Preference Directory of the Model Manager server is not accessible by untrusted parties.
- Using permissions, you can configure access control for the content stored on the Model Manager server as needed. You can also use separate instances of the Model Manager server — with each server using their own databases — to isolate different

content and user groups. The latter is best practice when exposing the Model Manager server to users outside of your organization.

Password Security

Local passwords stored by the Model Manager server are hashed by 100,000 iterations of the PBKDF2WithHmacSHA256 algorithm. This means that an adversary that gets access to the hashed passwords will not easily be able to obtain the original passwords. However, if an adversary does obtain a local password (for example, by brute-force guessing a weak password), the adversary could log in to the Model Manager server.

By default, the Model Manager server writes hashed local passwords to the file `/db/settings/local.db` in the Preference Directory. The temporary password set for the Default Local Administrative User during installation is stored using the same hash algorithm in the file `tempadminlogin.properties` in the root of the installation directory.

As mentioned previously, it is recommended to use TLS (via HTTPS) to protect passwords sent from web browsers and COMSOL Multiphysics to the Model Manager server. A warning appears in COMSOL Multiphysics at the time of login if the connection to the Model Manager server is not secure. Web browsers will show similar warnings.

Passwords saved in COMSOL Multiphysics for connecting to the Model Manager server ("Remember password") are stored encrypted so that only the logged-in user can access them. On Windows®, the encryption is done using the Data Protection API of the operating system. On Linux® and macOS, the encryption is done using a master key stored in the preference directory, protected by file system permissions. The same encryption is used to protect passwords stored by a Model Manager server configured to connect to external SQL database servers or Apache Solr™ servers using password-based authentication methods — see also Databases with External Server Components.

When a COMSOL Multiphysics client is connected to a COMSOL Multiphysics server on another host computer, the communication is typically on a nonsecure channel. This means that Model Manager server passwords entered in the user interface on the client computer are sent to the COMSOL Multiphysics server computer in an unencrypted cleartext format. To prevent this exposure, either tunnel the connection between the COMSOL Multiphysics client and the COMSOL Multiphysics server using a secure SSH tunnel, or first enter and save the Model Manager server credentials using a local COMSOL Multiphysics instance running on the COMSOL Multiphysics

server computer. A Model Manager server password already saved on the COMSOL Multiphysics server computer is not sent to the COMSOL Multiphysics client computer when connecting to a Model Manager server.

3

Administration

In this chapter, you will learn how to configure and administer a Model Manager server via its web interface. You can log in to the web interface to, for example, set up a Model Manager server database or use the web-based asset management system — see also Asset Management. This chapter assumes that you have already installed and started a Model Manager server.

In this chapter:

Accessing the Web Interface

In a web browser, go to `http://localhost:8181`. If you are accessing it remotely, use the server name — computer name and domain, or the local IP address — of your server instead of `localhost`. If you installed with another port number than the default `8181`, use that number instead. Enter your username and password. Click the **Log In** button.

If you entered the username for the administrator account configured during installation — see Default Local Administrative User — you will be prompted to change your temporary password. Write the new password in the **Password** field and then repeat the password in the **Repeat password** field. Click **Save**.

Resetting the Default Administrator Password

See the *Model Manager Reference Manual* to learn how to connect to the Model Manager server from COMSOL Multiphysics® once you have finished configuring the server.

Adding the Model Manager Server Database on First Login

If you selected **Custom** under **Add a Model Manager server database** in the Server step of the Model Manager server installation, you will be redirected to the **Add Database** page for adding a new database. If Model Manager server finds an existing Model Manager database — perhaps created by an older installation — you are instead redirected to the **Home** page.

A suggested setup for a Model Manager database using Managed Server Components with default locations for data directories and backup directories is shown on the **Add Database** page. These are the same locations used when a database is automatically set up — see also Default Data Directories and Default Backup Directories.

You are strongly recommended to first read the section Installation Planning before adding your Model Manager server database.

Unless you are intending to use external software to manage backups of the data directories, you are strongly recommended to change the backup directory locations for the SQL database and the resources directory to a server disk different than that of their data directories — this to safeguard against a potential disk crash. You can also replace some, or all, of the managed server components with external server components — see Databases with External Server Components.

Click **Save** to create the new Model Manager server database.

Saving may take some time as Model Manager server creates and initializes the data directories on disk. Upon successful completion, you will be redirected to the **Home** page. Otherwise, you will either be returned to the **Add Database** page or be redirected to the **Database** page. Read the message at the top of the page to see why the setup did not complete.

If any existing managed server components are found by the Model Manager server, these will be made available for selection on the **Add Database** page. See Managed Server Components if you still prefer to create completely new managed server components for the database.

- Examples of Server Setups
- Adding Databases

Overview of Web Pages

The web pages in a Model Manager server can be grouped as follows:

- Web pages for the asset management system. Click the **Home** link in the top navigation bar to open the **Home** page, in which you can search for assets in the Current Database. If you have been granted access to repositories in the database, you can also search for models and data files. See Asset Management for more details.
- Web pages for system administration. Click the cog wheel in the top navigation bar and select **System** to open the administration area for configuring the Model Manager server. Only administrators can access this area.
- Web pages for database administration. Click the cog wheel in the top navigation bar and select **Database** to open the administration area for configuring the Current Database. Although all users can access this area, only those granted the necessary

database permissions can make changes. See Database Administration for more details.

- The web page for the currently logged-in account. Click the user profile in the top navigation bar and select **My Account** to open The My Account Page.

To log out from the web interface, click the user profile and select **Log Out**.

Current Database

Administrators can add multiple server databases to a Model Manager server. Select the *current database* to show in the web interface from the list of active databases in the top navigation menu. The list is only shown if there is more than one active database available. The server database set as the *default database* is initially selected in the list when you log in to the web interface — see also Configuring Databases.

See the *Model Manager Reference Manual* to learn how to connect to a database other than the default database from COMSOL Multiphysics.

Changing the Language

Administrators can change the default language used in the Model Manager server web interface:

1. In the **System** administration area, under **Configuration**, click **Language** to open the **Language** administration page.
2. Select a language in the list.
3. Click **Save**.

Users may override the default server language by specifying a custom language to be used in the web interface — see The My Account Page.

Connectors

You add a *connector* to a Model Manager server in order to configure ports that the server should listen on. This is useful if you, for example, want to enable transport layer security (TLS) for secure connections via HTTPS on port `443`.

A Model Manager server is automatically configured with a default connector listening on the port set during installation. This connector is, however, *not* configured with TLS enabled.

The **Connectors** page, opened by clicking **Connectors** in the **System** navigation sidebar, shows a table with all connectors that have been manually added to the Model Manager server. The table is empty for a new installation. Click on the label of a connector in the table to show more details for that connector. Click the **Add** button to add a new connector.

Adding a Connector With TLS

You can set up a Model Manager server to accept secure connections by adding a connector with TLS encryption enabled.

An alternative to adding a connector is using a reverse proxy with TLS configured — see Running Behind a Reverse Proxy.

To add a new connector:

1. Write a label for the connector in the **Label** field.
2. Select when and how the connector should start listening on its port in the **Start mode** list. Select **Automatic** if the connector should start listening when the Model Manager server starts. Select **Manual** if the connector should only start listening when clicking **Start** on the **Connector** page.
3. Select **Default** in the **Port** list if the connector should listen on the default port — the default port is `443` if TLS is enabled, `80` if TLS is disabled. Select **Manual** to manually write the port to listen on in the shown input field.

4 Select **All** in the **Listener address** list to let the connector listen on all local addresses. Select **Custom** and write an IP address in the shown input field to only listen on that address.

5 Select between **Enabled** or **Disabled** in the **Enable TLS** list to explicitly set whether or not TLS is enabled for the connector. Select **Automatic** if TLS should be enabled if, and only if, a TLS host configuration has been added.

6 Select between **Enabled** or **Disabled** in the **Support HTTP/2** list to explicitly set whether or not the HTTP/2 protocol is enabled for the connector. Select **Automatic** if HTTP/2 is to be enabled if, and only if, TLS is enabled.

The Linux® operating system typically restricts opening of ports below 1024, such as the default ports of a connector. The symptom is a permission-denied error when trying to start the connector. Either select **Manual** in the **Port** list and use a higher-numbered port or consult the documentation that came with your operating system to allow the Model Manager server process to open these lower-numbered ports. See also the note in the systemd Service section for a possible solution when running the Model Manager server as a `systemd` service.

To enable TLS for the new connector, you need to add a TLS host configuration with an associated server certificate for every hostname that should accept secure connections from clients. You can use a wildcard in your hostname if it is supported by the corresponding server certificate.

Click **Add TLS Host Configuration** to add a new TLS host configuration for the connector.

1 Select **Default** in the **Hostname** list to accept connections regardless of the hostname used by a connecting client. The **Default** option can be used for at most one TLS host configuration. Select **Custom** and write the name of the host associated with your certificate in the shown text field to use a custom hostname for this TLS host configuration. You can write either a fully qualified domain name, say `modelmanager.example.com`, or a wildcard domain name, say `*.example.com`.

   If no TLS host configuration uses the **Default** option for the **Hostname**, the first configuration in the list will be used for all connections for which the hostname of a connecting client does not match any **Custom** hostname — effectively changing it to use the **Default** option. Using **Custom** hostnames is therefore only useful if there is more than one TLS host configuration for a connector.

2 Select the TLS versions that are available when communicating with clients in the **Client compatibility** list. Select **Modern** for `TLSv1.3`, **Intermediate** for `TLSv1.2` and `TLSv1.3`, or **Automatic** if the Model Manager server should decide.

3 Under **Certificates**, click **Add Certificate** to add a server certificate.

   a Either write a new name, or keep the suggested name, for the certificate in the **Name** field. The name must be unique in the list of certificates for a TLS host configuration.

   b Select the type of location used to store certificates in the **Location** list. A Model Manager server supports the following locations: PEM Files, PKCS#12 Keystore, Windows Native Certificate Store, and MacOS Native Keychain.

   c Provide the settings specific for the selected location.

4 Click **Save** to save the new connector.

If you add a TLS-enabled connector and only want to allow secure connections to the server, you should add firewall rules that block external access on the port set during installation — `8181`, by default.

PEM FILES

The **PEM files** location type has the following settings:

- **Certificate file**. The path on the Model Manager server computer to the certificate file.
- **Chain file**. The path on the Model Manager server computer to the certificate chain file. Leave empty if the certificate does not have a chain file.
- **Key file**. The path on the Model Manager server computer to the file containing the certificate key. Leave empty if there is no separate file for the key.
- **Key password**. The key password for the certificate.

PKCS#12 KEYSTORE

The **PKCS#12 Keystore** location type has the following settings:

- **Keystore file**. The path on the Model Manager server computer to the keystore file.
- **Keystore password**. The password for the keystore file.
- **Key alias**. The alias of the certificate to use from the keystore. Select from the list of available aliases.
- **Key password**. The key password for the certificate.

WINDOWS NATIVE CERTIFICATE STORE

Select the alias of the certificate to use from the **Windows native certificate store** in the **Key alias** list. The alias is typically the friendly name for the certificate, as reported by the Windows® certificate manager, or the first encountered common name of the certificate if it does not have a friendly name.

A Model Manager server can only access server certificates in the **Personal** > **Certificates** directory in the **Current user** certificate store. It cannot access the **Local computer** certificate store or the per-service **Service account** certificate stores.

If the Model Manager server has been installed as a Windows® service, the **Current user** certificate store belongs to the user account the service is configured to log on as — for example **NT AUTHORITY\LocalService** for **LocalService** when running Windows® using an English system locale. This means that importing or administrating certificates must be done running as this same service account. The tool PsExec from Sysinternals can be used to launch the Windows® certificates management tool interactively as a service account, for example:

```
psexec -i -u "NT AUTHORITY\LocalService" mmc certmgr.msc
```

The service account names depend on the system locale used when running Windows®.

MACOS NATIVE KEYCHAIN

Select the alias of the certificate to use from the **macOS native keychain** in the **Key alias** list.

Common TLS Certificate Formats

A TLS certificate is typically obtained by sending a *certificate signing request* (CSR) for the website you want to host the Model Manager server to a trusted *certificate authority* (CA). In response, you may receive one of several different formats for the certificate and its chain or root certificate. This includes:

- A single `<filename>.pfx` file. This is typically a PKCS#12 Keystore file containing the certificate as well as its chain or root certificate.
- A collection of up to three files of the form: `<certificate-filename>.cer`, `<chain/root-filename>.cer`, and `<key-filename>.key`. Use these three files in the settings for PEM Files.

The Connector Page

The **Connector** page contains configuration settings and current status of a connector added to the Model Manager server. The connector fields are:

- **Label**. The label of the connector. Used for display purposes in the Model Manager server web interface.
- **Start mode**. Either **Automatic** if the connector should start listening when the Model Manager server starts or **Manual** if the connector should only start listening when clicking the **Start** button.
- **Port**. The port the connector is listening on. A **Default** value if the connector should listen on the default port — the default port is `443` if TLS is enabled, `80` if TLS is disabled. Otherwise the number of the port.
- **Listener address**. Either **All** if the connector listens on all local addresses or an IP address if the connector only listens on that address.
- **Enable TLS**. The enabled status for transport layer security — **Enabled** or **Disabled** if TLS is always enabled or disabled for the connector, **Automatic** if TLS should be enabled if, and only if, the connector has a TLS host configuration.
- **Support HTTP/2**. The enabled status for the HTTP/2 protocol — **Enabled** or **Disabled** if the HTTP/2 protocol is always enabled or disabled for the connector, **Automatic** if the HTTP/2 protocol should be enabled if, and only if, TLS is enabled.
- **Status**. The current status of the connector.

  The status shows, for example, if the connector is currently started or if the connector has failed to start due to some configuration issue. For the latter case, read the nested error information to look for hints to the underlying cause.
- **TLS Host Configurations**. A list of TLS host configurations and certificates added to the connector.

Click the **Stop** button to stop the connector from listening on its configured port. Click **Start** to start listening. Click **Reload** to force a started connector to reload based on its current configuration — this is useful, for example, if a TLS certificate has been externally updated.

Click the **Edit** button to edit the configuration of the connector. Click the **Delete Permanently** button to permanently delete the configuration.

Clicking **Delete Permanently** will only delete the configuration for the connector; it will not delete any configured keystore or certificate files.

Login Configuration

A Model Manager server can be configured to use the following authentication mechanisms when a user tries to log in:

- Local Authentication. The username and password provided on the **Log In** page are compared with the username and (hashed) password stored in a local settings database in the Model Manager server Preference Directory.
- External Authentication. The username and password provided on the **Log In** page are compared with that found in an external credentials storage using, for example, Windows® authentication or lightweight directory access protocol (LDAP).
- Proxy Authentication. The authentication is delegated to a reverse proxy placed in front of the Model Manager server. Any HTTP requests passed from the reverse proxy to the Model Manager server are trusted to have already been authenticated by the proxy.

You can configure multiple authentication mechanisms for a Model Manager server. A user will successfully authenticate with the server if at least one mechanism succeeds.

Local Authentication

An administrator can manually add an account on The Accounts Page by providing a username and password to be used by a user of Model Manager server. The password is hashed, as described in Password Security, and stored in a local settings database in the Model Manager server Preference Directory. The user may later change their password on The My Account Page.

External Authentication

A Model Manager server uses the pluggable authentication framework *Java authentication and authorization service* (JAAS) to integrate with external authentication protocols. A login configuration is specified by listing vendor-supplied *login modules* that should be applied (in the order they are listed) when the user tries to log in.

Support for Windows® authentication and LDAP authentication as login modules is included with a Model Manager server installation. You can also specify a custom login module using a *module class* obtained from a third-party vendor.

A user that has authenticated via a login module can be automatically assigned membership to one or more groups in a Model Manager server database for the duration of their login session. This is useful if an administrator has already set up group memberships in the external credentials storage and wants to use the same groups for controlling access to items in a Model Manager server database.

See Database Administration and the *Model Manager Reference Manual* for more information on user management and access control.

The **External Authentication** page, opened by clicking **External Authentication** in the **System** navigation sidebar, shows all currently configured login modules in the **Login Modules** field and all group membership mappings in the **Group mappings** field. Both lists are empty for a new installation of a Model Manager server.

Click **Edit** to edit the login modules and group membership mappings. Click **Clear** to clear all login modules and group membership mappings.

You can test the validity of the configuration by clicking the **Test** button — see Testing an External Authentication Configuration.

ADDING A LOGIN MODULE

When editing the configuration for external authentication, click **Add Login Module** to add a new login module to the overall list of login modules.

1 Select the type of login module in the **Type** list. The available options are Windows Login, LDAP, and Custom — see below for their specific configuration settings.

2 Select how a successful or unsuccessful authentication attempt by the login module should be handled in relation to other login modules in the **Control flag** list. The available options are:

   a **Requisite**. The login module is required to succeed for the overall authentication to succeed. If it succeeds, the next login module in the list of modules will be applied. If it fails, no other login module is applied.

   b **Optional**. The login module is not required to succeed for the overall authentication to succeed. Regardless if it succeeds or fails, the next login module in the list of modules will be applied.

   c **Required**. The login module is required to succeed for the overall authentication to succeed. Regardless if it succeeds or fails, the next login module in the list of modules will be applied.

   d **Sufficient**. If the login module succeeds, no other login module is applied. If it fails, the next login module in the list of modules will be applied.

3 Click the **Add Option** button under **Options** to add custom options in the **Option** and **Value** fields. The supported options are vendor specific to each login module.

The overall external authentication is successful only if all applied login modules with **Required** and **Requisite** succeeded. Note that *applied* is important here — if a login module with **Sufficient** succeeds, only login modules with **Required** and **Requisite** specified *before* that login module need to succeed. At least one login module with **Sufficient** or **Optional** must succeed if there are no login modules with **Required** or **Requisite**.

The value set for the **Control flag** is irrelevant when only a single login module has been added. The overall authentication succeeds if, and only if, the login module is successful.

Windows Login

If you have installed the Model Manager server on the Windows® operating system, users can log in using Windows® authentication by adding a **Windows login** module.

1 Write the format to identify the user being authenticated in the **Principal format** list. Select **Fully qualified names** for usernames of the form <*domain*>\<*user*>. Select **SID** to use security identifiers to identify users. Select **Both** to allow both fully qualified names and security identifiers.

2 Write the format for the returned principal role names of the authenticated user in the **Role format** list. Select **Fully qualified names** to return names of the form

<*domain*>\<*group*>. Select **SID** to return security identifiers. Select **Both** to return both fully qualified names and security identifiers. Select **None** if no principal role names should be returned.

LDAP

You can configure a login module to use LDAP to communicate with a credentials storage supporting that protocol — for example, Windows® Active Directory® (AD) or OpenLDAP.

See the Java documentation for `com.sun.security.auth.module.LdapLoginModule` to learn more about the configuration settings for the LDAP login module.

You may find the free Apache Directory Studio™ tool helpful when setting up the connection to your LDAP server. For more information, see www.comsol.com/support/knowledgebase/1306.

1. Write the URL connection string used to connect to the LDAP server and directory containing the user to authenticate in the **User Provider** field. You can write several URL connection strings separated by spaces; each URL connection string will be attempted until a successful connection to the LDAP server is established.
2. Write an optional LDAP filter string in the **User filter** field. This specifies a search filter used to locate the user in the LDAP directory by the user's *distinguished name* in LDAP. Use the special token `{USERNAME}` as a placeholder for the username provided on the **Log In** page.
3. Write an LDAP distinguished name or some other string name in the **Authentication identity** field. The name is used to locate the user in the LDAP directory. The name must contain the token `{USERNAME}`, which acts as a placeholder for the username provided on the **Log In** page. The **User filter** field must be specified if the **Authentication identity** field does not contain a distinguished name.
4. Write a string name in the **Authorization identity** field that will be used to associate a principal name with a successfully authenticated user. You can write a single token of the form `{<attribute-name>}` as a placeholder for a user attribute value in LDAP that holds a principal name.

5 Select whether or not the connection to the LDAP server should use secure sockets layer (SSL) in the **Use SSL** list.

Replace any spaces in the URL connection string with the character combination `%20` to avoid having the value interpreted as separate URLs.

You are strongly recommended to enable SSL to avoid sending credentials in cleartext to the LDAP server.

A sample configuration for Windows® Active Directory® may look like:

1 **User Provider**: `ldap://ldap.example.com:3268/DC=example,DC=com`

2 **User filter**: `(&(sAMAccountName={USERNAME})(objectclass=user))`

3 **Authentication identity**: `{USERNAME}@example.com`

4 **Authorization identity**:

with **Authorization identity** intentionally left blank.

A sample configuration for OpenLDAP may look like:

1 **User Provider**: `ldap://ldap.example.com/ou=People,dc=example`

2 **User filter**: `(&(uid={USERNAME})(objectClass=inetOrgPerson))`

3 **Authentication identity**:

4 **Authorization identity**: `{MAIL}`

with **Authentication identity** intentionally left blank.

Custom

You can configure a custom login module provided by a third-party vendor. Write the fully qualified Java class name of the login module in the **Module class** field. Consult the documentation of the login module for the custom options that need to be provided.

ADDING GROUP MAPPINGS

You can map principal names returned by login modules to corresponding group names in a Model Manager server database. A successfully authenticated user will be considered a member of mapped groups for the duration of their login session.

A *principal* is an umbrella term for a user, role, or group.

When editing the configuration for external authentication, click **Add Group Mapping** to add a new mapping.

1. Write the name of a principal returned by a login module in the **Principal name** field. This can be a username, role name, group name, or some other string name specific to the type of login module.
2. Under **Mapped groups**, click **Add Group** and write the name of a group in a Model Manager server database that the authenticated user should become a member of. The group will be automatically created in the database if it did not already exist when the user logged in. Repeat to map the principal to more groups.

Successfully mapped group names are visible in the **External Group Memberships** field on The My Account Page for the logged in account.

TESTING AN EXTERNAL AUTHENTICATION CONFIGURATION

You can test the current configuration for external authentication. Click the **Test** button and write a username and password in the opened dialog.

If the authentication succeeds, you will see the principal names returned by the login modules, including the name used to authenticate with, as well as the names of any groups these principal names would be mapped to. If the authentication fails, you will see the error messages returned by the login modules.

Proxy Authentication

You may find it useful to delegate all authentication to a trusted reverse proxy placed in front of the Model Manager server. To do this, configure the reverse proxy for authenticating clients using any authentication scheme you want. The Model Manager server itself can then identify the externally authenticated user by a HTTP header set by the reverse proxy. The reverse proxy can, for example, use the Basic HTTP

authentication scheme and set the `Authorization` HTTP header with the username of the user and an empty password.

The **Proxy Authentication** page, opened by clicking **Proxy Authentication** in the **System** navigation sidebar, shows the current configuration for such a proxy authentication scheme. A new installation of a Model Manager server is not configured to use a proxy authentication scheme, as indicated in the **Use proxy authentication** field.

Click **Edit** to edit the current configuration. Click **Clear** to clear the configuration in its entirety.

When editing the proxy authentication configuration:

1. Select the **Use proxy authentication** checkbox to enable proxy authentication. Clear the checkbox to disable.

   You may prefer disabling instead of clicking the **Clear** button if you later want to enable the proxy authentication again using previously set configuration settings.

2. Under **Trusted proxy connections**, select the **From loopback address** checkbox to trust proxies running on the same computer as the Model Manager server. Write any other host to be trusted under **From hosts** and click the **Add** button. Repeat this with all other trusted hosts.

   The Model Manager server will trust that connections made from these hosts have already been successfully authenticated if the hosts supply a username in the request — see below.

3. Select **Basic auth username** in the **Username source** list if the authenticated user's username is set in the `Authorization` HTTP header (using an empty password) by the reverse proxy. Select **HTTP header** if the username is set in a custom HTTP header. Write the name of the custom header in the **Username** field.

4. Write the name of an optional HTTP header containing the user's display name in the **User display name** field. Leave empty to let the display name be the same as the username.

5. Write the name of an optional HTTP header containing the name of a group that the user is a member of in the **Group name** field.

6. Write the name of an optional HTTP header containing a display name for a group that the user is a member of in the **Group display name** field. Leave empty to let the display name be the same as the group name.

7. Select whether or not the Model Manager server should write detailed log messages when a user tries to authenticate using proxy authentication in the **Detailed logging**

checkbox. This is useful during initial setup of proxy authentication but afterward should be left cleared to not fill up log files.

You can add as many group mappings as you want — group names and display names will be paired in the order they are encountered as headers in the request. An authenticated user will be set as a member of these groups for the duration of their login session. A group will be automatically created in the database if it did not already exist.

Successfully mapped group names are visible in the **External Group Memberships** field on The My Account Page for the logged in account.

Accounts

Users log in to a Model Manager server from the COMSOL Desktop environment or via a web browser using an account stored in the local settings database of the server. The account is created automatically if authenticating using External Authentication or Proxy Authentication. If using Local Authentication, an administrator first needs to create each account via the web interface. A hashed account password is stored in the local settings database in this latter case — see Password Security. No account password is stored by the Model Manager server for the other authentication mechanisms.

In this section:

The My Account Page

Click the user profile in the top navigation bar and select **My Account** to open the **My Account** page for editing your own account settings — including, for example, changing your password if using Local Authentication.

The page displays the following fields:

- **Name**. The username of your account. This is the name used, for example, when connecting to a Model Manager server database from COMSOL Multiphysics.
- **Display name**. An alternative name to your username used for display purposes.
- **Language**. The language used in the web interface for your account. A **Default** value means your account uses the Model Manager server's default language.
- **Time zone**. The time zone used to display dates and times in the web interface, as well as the time zone used to specify timestamp filters using the Model Manager search syntax. The time zone is automatically read from your browser's settings.

The Model Manager server computer's time zone is used as a fallback for timestamp filters if the local time zone is not available via your browser. An information message is displayed next to the **Time zone** field when this happens.

- **External Group Memberships**. Groups that your account have been automatically mapped as a member of via a login configuration. The field is hidden if no groups are mapped.

To set a new password for your account (if using Local Authentication):

1. Click the **Password** button.
2. Write your current password in the **Current password** field.
3. Write your new password in the **Password** field and then repeat the password in the **Repeat password** field.
4. Click **Save**.

To change your display name or preferred language:

1. Click the **Edit** button.
2. Write a new name in the **Display name** field.

   The display name defaults to the username if left empty.
3. In the **Language** menu, select the language to use in the web interface for your account. Select **Default** to use the Model Manager server's default language.
4. Click **Save**.

The Accounts Page

The **Accounts** administration page, opened by clicking **Accounts** in the **System** navigation sidebar, shows a table with all accounts created for the Model Manager server — either manually by an administrator or automatically by the server.

The table columns are:

- The **Name** column — the unique username of the account.
- The **Display Name** column — the display name of the account.
- The **Language** column — the language used in the web interface for the account.
- The **Administrator** column — whether or not an administrator account.
- The **Created** column — point in time when the account was created.
- The **Last Modified** column — point in time when the account settings was last modified.
- The **Last Login** column — point in time when the account last logged in to the Model Manager server.

Click an account in the **Name** column of the table to show more details for that account. Click the **Add** button to manually add a new account.

ADDING ACCOUNTS

To add a new account:

1. In the **Name** field, write the username that the account uses to authenticate with the Model Manager server.
2. In the **Display name** field, write an alternative name used for display purposes.

   The display name defaults to the username if left empty.
3. In the **Language** menu, select the language to use in the web interface for the account. Select **Default** to use the Model Manager server's default language.
4. Select **Yes** or **No** in the **Administrator** list to set whether or not the new account is an administrator account.
5. In the **Password** field, write the password that the account uses to authenticate with the Model Manager server. Repeat the password in the **Repeat password** field.

   The password can later be changed by the user on The My Account Page.
6. Click **Save**.

Administrators automatically pass any authorization checks in Model Manager — see also the *Model Manager Reference Manual*. You are strongly encouraged to limit the number of administrator accounts.

You only need to manually add accounts if using Local Authentication.

THE ACCOUNT PAGE

The **Account** page lets you view and edit a specific account. The account fields are:

- **Name**. The unique username of the account in the Model Manager server.
- **Display name**. The account name used for display purposes in the Model Manager server web interface.
- **Language**. The language used in the web interface for the account.

- **Administrator**. Whether or not the account is an administrator account.

  An administrator account is allowed to view and configure the Model Manager server via the **System** administration area.
- **Created**. Point in time when the account was created.
- **Last Modified**. Point in time when the account settings was last modified.
- **Last Login**. Point in time when the account last logged in to the Model Manager server.

Click **Edit** to edit the display name, language, and administrator status of the account.

Click **Change Password** to set a new password for the account if using Local Authentication. This is useful, for example, if a user forgets their account password — the user may later change the reset password on the **My Account** page.

Click **Delete Permanently** to permanently delete the account.

Deleting an account will only remove that user's ability to log in to the Model Manager server — any information related to the user in a Model Manager server database will be left unchanged. If using any authentication mechanism other than Local Authentication, a deleted account will be automatically recreated the next time the user successfully authenticates.

Managed Server Components

A Model Manager server database consists of three separate server components — a SQL database for the version control management of assets, models, and data files, a dedicated directory on the file system for storing large binary and text data, and a collection of search indexes used by the Model Manager search functionality. You can either use server components managed by the Model Manager server itself, including starting and stopping component subprocesses and handling backups, or you can use external server components provided by your organization. In the latter case, you will need to handle availability and backups for the server components outside of the Model Manager server.

In this section you will learn how to add and configure such *managed server components*, including setting up backups for their data directories. You will also learn how to restore these components from a backup. See the next section to learn how you combine managed server components into a Model Manager server database, optionally swapping out all, or some, of the components with your own external server components.

To configure a database that is entirely managed by the Model Manager server itself, select the **Managed PostgreSQL®** and **Managed Apache Solr™** checkboxes in the Products step of the Model Manager server installation.

In this section:

Managed SQL Database Servers

A *managed SQL database server* is a server instance of the PostgreSQL® relational database management system controlled by the Model Manager server itself. The **Managed SQL Database Servers** page, opened by clicking **SQL Database Servers** in the **System** navigation sidebar, shows a table with the configurations of all such managed SQL database servers. If you selected during installation to let Model Manager server automatically set up a new database, or if you kept the suggested configuration settings

for the SQL database when adding the database on first launch, a single configuration — **Default managed PostgreSQL® server** — is initially shown in the table. See also The Default Managed Database.

Links to the **Managed SQL Database Servers** page are hidden in the web interface if you cleared the **Managed PostgreSQL®** checkbox in the Products step of the Model Manager server installation.

Click on the label of a managed SQL database server in the table to show more details for that managed server. Click the **Add** button to add a managed SQL database server.

ADDING MANAGED SQL DATABASE SERVERS

You can add a new managed SQL database server to Model Manager server from the **Add Managed SQL Database Server** page. This is useful, for example, if you want to use different options than the default ones for the automatically added managed SQL database server but, at the same time, want to keep the default one around as a reference. You would also need to add the managed SQL database server if you want to connect to one that already exists on the file system, possibly created from another Model Manager server installation.

To add a managed SQL database server:

1. Write a label for the managed SQL database server in the **Label** field.

   The label is only used for display purposes in the Model Manager server web interface.

2. Write the path on the file system where the data directory for the managed SQL database server is located in the **Data directory** field. The data directory will be automatically created if it does not already exist on the file system.

   The data directory must be located on a physical disk unless running on Linux® using NFS mounted in hard mode. Other network disk configurations are not supported.

3. Select when and how the managed SQL database server is started in the **Start mode** list. Select **Automatic** if the server should start when the Model Manager server is started or when the SQL database server is first used by a Model Manager database. Select **Manual** if the server must be manually started on The Managed SQL Database Server Page.

4. Select the **Enable backup** checkbox to enable backups of the data directory. Write the path on the file system where backups will be written to in the **Backup directory** field.

The backup directory will be automatically created if it does not already exist on the file system. See also Backup of a Managed SQL Database Server.

5 In the **Trigger mode** list under **Restore**, select **Automatic (most recent)** if the data directory should be automatically restored from the most recent backup if it is missing on disk, or if a previous restore has not yet finished, when the managed SQL database server starts. Select **Manual** if a restore requires manual triggering on The Managed SQL Database Server Page. See also Restore of a Managed SQL Database Server.

6 Click **Save** to add the new configuration. The data directory and the backup directory will be automatically created as needed the next time the managed SQL database server is started.

The specified data directory must either not exist at all on the file system or, if it exists, be empty if you want to create a new managed SQL database server. A nonempty data directory is assumed to belong to an existing managed SQL database server that you want to add a configuration for in the Model Manager server.

When the **Start Mode** is set to **Manual**, the parent directory to the data directory must already exist on the file system for the managed SQL database server to successfully start. If it does not exist, you can manually create it from The Managed SQL Database Server Page. Parent directories are created automatically for **Start Mode** set to **Automatic**.

Placing the data directory of a managed SQL database server on a network file system is not supported unless running on Linux® using NFS mounted in hard mode — see also Installation Planning.

Prefer placing the backup directory on a physical disk different than the disk containing the data directory. Otherwise, you might risk complete data loss in case of a server disk crash.

At least one SQL database needs to be added to the managed SQL database server to use it for a Model Manager server database — see SQL Databases.

THE MANAGED SQL DATABASE SERVER PAGE

The **Managed SQL Database Server** page contains configuration details for a managed SQL database server that is added to the Model Manager server — see Adding Managed SQL Database Servers. The configuration fields are:

- **Platform**. The database system platform used for the managed SQL database server.
- **Label**. The label of the managed SQL database server. Used for display purposes in the Model Manager server web interface.
- **Data directory**. The file system path to the data directory of the managed SQL database server.

  The currently available disk space on the file storage containing the data directory is shown below the path.

  You can see the current status of the data directory in its **Status** field. If the data directory does not exist on the file system, click the **Create** button to create it.
- **Start mode**. The start mode of the managed SQL database server — **Automatic** if the server starts when the Model Manager server is started or when the SQL database server is first used by a Model Manager database, **Manual** if the server must be manually started.

  The adjacent **Status** field shows, for example, if the server is currently running together with its current uptime or if the server has failed to start due to some configuration issue. For the latter case, read the nested error information to look for hints to the underlying cause.

Click the **Stop** button to stop a running managed SQL database server. Click **Start** to start it.

To edit the configuration for a managed SQL database server, proceed as follows:

1. Deactivate all Model Manager server databases using SQL Databases in the managed SQL database server. See also The Database Page.
2. Click the **Stop** button to stop the managed SQL database server.
3. Click the **Edit** button.

4 Enter the new configuration settings. If you change any directory paths, manually move the corresponding directories on the file system via, for example, a system file explorer *before* saving.

5 Click the **Save** button.

6 Start the managed SQL database server if not automatically started.

7 Activate all databases deactivated in the first step.

Deleting a Managed SQL Database Server Configuration

Click the **Delete Permanently** button to permanently delete the configuration.

	Clicking **Delete Permanently** will only delete the configuration for the managed SQL database server; it will not delete its data directory or backup directory. You can add an existing SQL database server again by adding a new configuration that points to its data directory and backup directory.

SQL DATABASES

The version control management system for simulation models and other data are stored in SQL databases in the managed SQL database server. A single Model Manager server database uses exactly one SQL database. The SQL databases are stored inside the data directory for the managed SQL database server.

	A SQL database cannot be shared between Model Manager databases. Multiple Model Manager databases may, however, share the same managed SQL database server.

The **Databases** section shows a table with all SQL databases in the SQL database server. The table columns are:

- The **Name** column — the name of the SQL database.
- The **Used by** column — the Model Manager database using the SQL database, if any.
- The permanently delete column — click the **Delete Permanently** button to permanently delete the SQL database.

	Permanently deleting a SQL database cannot be undone unless you restore the managed SQL database server from a backup.

To add a new SQL database, write the name of the database and click the **Add** button under the **Databases** section on the **Managed SQL Database Server** page. The name may only contain the letters a–z and digits 0–9 and must start with a letter. Some specific names are explicitly forbidden — either because these databases already exist internally in the database server or because they are *reserved names*. The Model Manager server will show an error message if such a forbidden name is entered. A suggested name is `modelmanager` — the same name used by The Default Managed Database.

The managed SQL database server must be running in order to add new databases.

BACKUP OF A MANAGED SQL DATABASE SERVER

When you enable backup for a managed SQL database server, it will continuously back up any data written to its data directory to a subdirectory inside the backup directory. This backup is *incremental* in the sense that it records all changes made in the server's SQL databases as a transactional log. When you restore a managed SQL database server, including all its SQL databases, this transactional log is replayed.

Since the time it takes to replay the transactional log scales with the running time of the managed SQL database server, it is wise to periodically create a new *base backup*. This is a complete point-in-time snapshot of the managed SQL database server. Once you have created such a base backup, the incremental backup will instead record changes using that backup as a new baseline — thereby speeding up any future restore.

On the **Managed SQL Database Server** page, click the **Trigger Base Backup** button to create a new base backup. The **Backup directory** field shows the directory path in which backups are stored.

A new base backup is automatically created when you enable backup for a managed SQL database server.

RESTORE OF A MANAGED SQL DATABASE SERVER

If you have configured a managed SQL database server with **Trigger mode** for restoring set to **Automatic (most recent)**, a missing data directory will be automatically restored using the most recently backed-up state in the backup directory when the managed SQL database server is started.

You can also manually restore the data directory of a managed SQL database server by clicking the **Trigger Restore** button on the **Managed SQL Database Server** page. A confirmation dialog is shown, informing you that the following steps will be taken by the Model Manager server:

1. The SQL database server is stopped if currently running.
2. Any existing data directory is renamed to avoid overwriting it with the restored data.
3. The most recent base backup is copied from the backup directory to the data directory.
4. The SQL database server is launched in recovery mode and automatically restores itself using the most recently found backed-up state — see Backup of a Managed SQL Database Server. Once finished, the SQL database server is made available again.

You are strongly recommended to also recreate any search indexes whose indexed data is populated from SQL databases in the restored SQL database server — see the manual steps in Restore of a Managed Search Index Server. Otherwise, search results in the Model Manager may not reflect what is stored in the SQL database.

You are recommended to deactivate any Model Manager server database using SQL Databases in the managed SQL database server before triggering a restore. See The Database Page.

Managed Resources Directories

A *managed resources directory* is a data directory controlled by the Model Manager server itself. The directory contains binary and text data that are deemed too large to store directly inside a SQL database. The **Managed Resources Directories** page, opened by clicking **Resources Directories** in the **System** navigation sidebar, shows a table with the configurations of all such managed resources directories. If you selected during installation to let Model Manager server automatically set up a new database, or if you kept the suggested configuration settings for the resources directory when adding the database on first launch, a single configuration — **Default managed resources directory** — is initially shown in the table. See also The Default Managed Database.

Click on the label of a managed resources directory in the table to show more details for that directory. Click the **Add** button to add a managed resources directory.

Unlike Managed SQL Database Servers and Managed Search Index Servers, a managed resources directory cannot be shared between Model Manager databases. You need to create a new managed resources directory for every Model Manager database you add to the Model Manager server.

ADDING MANAGED RESOURCES DIRECTORIES

You can add a managed resources directory to Model Manager server from the **Add Managed Resources Directory** page. This is useful, for example, if you want to use different options than the default ones for the automatically added managed resources directory but at the same time want to keep it around as a reference. You would also need to add the managed resources directory if you want to connect to one that already exists on the file system, possibly created from another Model Manager server installation.

To add a managed resources directory:

1. Write a label for the resources directory in the **Label** field.

   The label is only used for display purposes in the Model Manager server web interface.

2. Write the path on the file system where the directory is located in the **Data directory** field. The data directory will be automatically created if it does not already exist on the file system.

   The data directory may be placed on either a physical disk or a network disk.

3. Select the **Enable backup** checkbox to enable backups of the data directory. Write the path on the file system where the backup will be written to in the **Backup directory** field. The backup directory will be automatically created if it does not already exist on the file system. See also Backup of a Managed Resources Directory

4. In the **Trigger mode** list under **Restore**, select **Automatic** if the data directory should be automatically restored from backup if it is missing on disk, or a previous restore has not yet finished, when activating the Model Manager server database using the directory — see The Database Page. Select **Manual** if a restore requires manual triggering on The Managed Resources Directory Page. See also Restore of a Managed Resources Directory.

5 Click **Save** to add the new configuration.

The specified data directory must be empty if you want to initialize a new managed resources directory. A nonempty data directory is assumed to belong to an existing managed resources directory that you want to add a configuration for in the Model Manager server.

Place the backup directory on a physical disk different than the disk containing the data directory. Otherwise, you might risk complete data loss in case of a server disk crash.

You may want to consider placing the data directory for managed resources on a dedicated file server whose file system is mounted by the server computer running the Model Manager server. The same goes for the backup directory using a second mounted file server. See Installation Planning.

THE MANAGED RESOURCES DIRECTORY PAGE

The **Managed Resources Directory** page contains details on the configuration of a managed resources directory added to the Model Manager server — see Adding Managed Resources Directories. The configuration fields are:

- **Label**. The label of the managed resources directory. Used for display purposes in the Model Manager server web interface.
- **Data directory**. The file system path to the managed resources directory.

  The currently available disk space on the file storage containing the directory is shown below the path.
- **Used by**. The Model Manager database using the managed resources directory, if any.

To edit the configuration for a managed resources directory, proceed as follows:

1 Deactivate any Model Manager server database using the resources directory, if any — see also The Database Page.

2 Click the **Edit** button.

3 Enter the new configuration settings. If you change any directory paths, manually move the corresponding directories on the file system via, for example, a system file explorer *before* saving.

4 Click the **Save** button.

5 Activate the database using the resources directory, if any.

Deleting a Managed Resources Directory Configuration

Click the **Delete Permanently** button to permanently delete the configuration.

Clicking **Delete Permanently** will only delete the configuration for the managed resources directory; it will not delete the directory itself. You can add the managed resources directory again using a new configuration that points to the directory and its backup directory.

BACKUP OF A MANAGED RESOURCES DIRECTORY

If you enable backup for a managed resources directory, the Model Manager server will continuously back up any files written to the resources directory to the backup directory. The **Backup directory** field on the **Managed Resources Directory** page shows the directory path for this backup.

You can also trigger a complete backup of the whole resources directory — this will back up *all* files in the resources directory to the backup directory. Click the **Trigger Backup** button on the **Managed Resources Directory** page. A backup will then be scheduled to begin if and when the Model Manager server database using the resources directory is activated.

RESTORE OF A MANAGED RESOURCES DIRECTORY

If you have configured a managed resources directory with **Trigger mode** set to **Automatic**, a missing directory will be automatically restored from the backup directory when a Model Manager server database using the managed resources directory is activated.

You can also manually restore a managed resources directory by clicking the **Trigger Restore** button on the **Managed Resources Directory** page. A restore will then be scheduled to begin when the Model Manager server database using the resources directory is activated. A confirmation dialog is shown, informing you that the following steps will be taken once the restore has begun:

1 The SQL database component of the Model Manager server database will be queried for all files expected to be present in the resources directory.

2 All expected files that are missing from the resources directory are copied from the backup directory.

A triggered restore of a managed resources directory only begins once a Model Manager server database using the resources directory is activated — see The Database Page.

Managed Search Index Servers

A *managed search index server* is a server instance of the Apache Solr™ enterprise search platform controlled by the Model Manager server itself. The **Managed Search Index Servers** page, opened by clicking **Search Index Servers** in the **System** navigation sidebar, shows a table with the configurations of all such managed search index servers. If you selected during installation to let Model Manager server automatically set up a new database, or if you kept the suggested configuration settings for the search indexes when adding the database on first launch, a single configuration — **Default managed Apache Solr™ server** — is initially shown in the table. See also The Default Managed Database.

Links to the **Managed Search Index Servers** page are hidden in the web interface if you cleared the **Managed Apache Solr™** checkbox in the Products step of the Model Manager server installation.

Click on the label of a managed search index server in the table to show more details for that managed server. Click the **Add** button to add a managed search index server.

ADDING MANAGED SEARCH INDEX SERVERS

You can add a managed search index server to the Model Manager server from the **Add Managed Search Index Server** page. This is useful, for example, if you want to use different options than the default ones for the automatically added managed search index server but at the same time want to keep it around as a reference. You would also need to add the managed search index server if you want to connect to one that already exists on the file system, possibly created from another Model Manager server installation.

To add a managed search index server:

1 Write a label for the managed search index server in the **Label** field.

   The label is only used for display purposes in the Model Manager server web interface.

2 Write the path on the file system where the data directory for the managed search index server is located in the **Data directory** field. The data directory will be automatically created if it does not already exist on the file system.

   The data directory must be located on a physical disk. Placing the directory on a network disk is not supported.

3 Specify the maximum heap memory that will be allocated to the Java virtual machine (JVM) of the managed search index server in the **Heap memory** list. Select **Default** to keep the default settings for the server. Select **Custom** and specify an integer value in either megabytes (**MB**) or gigabytes (**GB**).

   The **Default** setting is recommended as a starting point. As your Model Manager server database grows in size, you could consider increasing the value if the current heap memory usage — as displayed on The Managed Search Index Server Page — stays close to the maximum value for an extended period of time.

4 Select when and how the managed search index server is started in the **Start mode** list. Select **Automatic** if the server should start when the Model Manager server is started or when the search index server is first used by a Model Manager database. Select **Manual** if the server must be manually started on The Managed Search Index Server Page.

5 Click **Save** to add the new configuration. The data directory will be automatically created as needed the next time the managed search index server is started.

The specified data directory must be empty if you want to create a new managed search index server. A nonempty data directory is assumed to belong to an existing managed search index server that you want to add a configuration for in the Model Manager server.

When the **Start Mode** is set to **Manual**, the parent directory to the data directory must already exist on the file system for the managed search index server to successfully start. If it does not exist, you can manually create it from The Managed Search Index Server Page. Parent directories are created automatically for **Start Mode** set to **Automatic**.

Placing the data directory of a managed search index server on a network file system is not supported — see also Installation Planning.

At least two search indexes need to be added to the managed search index server in order to use it for a Model Manager server database — see Search Indexes.

Unlike Managed SQL Database Servers and Managed Resources Directories, a managed search index server requires no backup.

THE MANAGED SEARCH INDEX SERVER PAGE

The **Managed Search Index Server** page contains details on the configuration of a managed search index server added to the Model Manager server — see Adding Managed Search Index Servers. The configuration fields are:

- **Platform**. The search platform used for the managed search index server.
- **Label**. The label of the managed search index server. Used for display purposes in the Model Manager server web interface.
- **Heap memory**. The maximum heap memory that will be allocated to the Java virtual machine (JVM) of the managed search index server — **Default** if the default setting is used for the server, otherwise a custom size in either megabytes or gigabytes.

  The current heap memory usage is displayed below the maximum heap memory setting. You should consider increasing the setting if the current usage remains close to the maximum value for an extended period of time.

The JVM on some platforms may not respect a too low of a value for the **Heap memory** field and instead end up using a larger value than specified.

- **Data directory**. The file system path to the data directory of the managed search index server.

  The currently available disk space on the file storage containing the data directory is shown below the path.

  You can see the current status of the data directory in its **Status** field. If the data directory does not exist on the file system, click the **Create** button to create it.

- **Start mode**. The start mode of the managed search index server — **Automatic** if the server starts when the Model Manager server is started or when the search index server is used by a Model Manager database, **Manual** if the server must be manually started.

  The adjacent **Status** field shows, for example, if the server is currently running together with its current uptime or if the server has failed to start due to some configuration issue. For the latter case, read the nested error information to look for hints to the underlying cause.

Click the **Stop** button to stop a running managed search index server. Click **Start** to start it.

To edit the configuration for a managed search index server, proceed as follows:

1. Deactivate all Model Manager server databases using Search Indexes in the managed search index server. See also The Database Page.
2. Click the **Stop** button to stop the managed search index server.
3. Click the **Edit** button.
4. Enter the new configuration settings. If you change any directory paths, manually move the corresponding directories on the file system via, for example, a system file explorer *before* saving.
5. Click the **Save** button.
6. Start the managed search index server if not automatically started.
7. Activate all databases deactivated in the first step.

Deleting a Managed Search Index Server Configuration

Click the **Delete Permanently** button to permanently delete the configuration.

Clicking **Delete Permanently** will only delete the configuration for the managed search index server; it will not delete its data directory. You can add an existing search index server again by adding a new configuration that points to its data directory.

SEARCH INDEXES

The search data used by the Model Manager search functionality is populated in *search indexes* in the managed search index server. A Model Manager server database requires

two search indexes. The search indexes are stored inside the data directory for the managed search index server.

A search index cannot be shared between Model Manager databases. Multiple Model Manager databases may, however, share the same managed search index server.

The **Search indexes** section shows a table with all search indexes in the search index server. The table columns are:

- The **Name** column — the name of the search index.
- The **Used by** column — the Model Manager database using the search index, if any.
- The permanently delete column — click the **Delete Permanently** button to permanently delete the search index.

A permanently deleted search index can be recreated by first adding back a new, empty, search index. The empty search index is then automatically populated with search data when the Model Manager server database using the search index is activated.

To add a new search index, write the name of the search index and click the **Add** button under the **Search Indexes** section on the **Managed Search Index Server** page. The name may only contain the letters a–z and digits 0–9 and must start with a letter. Suggested names are `assets` and `items` — the same names used by The Default Managed Database.

The search index server must be running in order to add new search indexes.

Unlike the data directories of Managed SQL Database Servers and Managed Resources Directories, the data directory and its search indexes do not require backup because the contents can be recreated from that of the SQL database — see Restore of a Managed Search Index Server.

RESTORE OF A MANAGED SEARCH INDEX SERVER

A managed search index server is automatically restored if its data directory is missing when the server is started, although all of its search indexes are then empty. The search

indexes will be automatically populated with search data when the Model Manager server database using the search indexes is activated.

You can also manually restore the data directory as follows:

1. Deactivate all Model Manager server databases using Search Indexes in the managed search index server. See The Database Page.
2. Stop the managed search index server.
3. Move any existing data directory to a new location.
4. Start the managed search index server.

   At this point, the data directory and its search indexes will be recreated based on the configuration settings of the managed search index server. The search indexes will, however, be empty of search data.
5. Activate all databases deactivated in the first step.

   At this point, the empty search indexes will be automatically repopulated with search data loaded from the corresponding SQL database as needed.

Populating the search indexes after a restore may take some time depending on the size of the SQL database.

Model Manager Server Databases

A Model Manager server database consists of the following components:

- A SQL database — either added to a SQL database server managed by the Model Manager server or added to an external SQL database server provided by your organization.
- A resources directory — either a directory managed by the Model Manager server or an external directory.
- Two search indexes — either added to a search index server managed by the Model Manager server or added to an external search index server provided by your organization.

In this section, you will learn how to configure a Model Manager server database using these components. You will see that you can use any combination of Managed Server Components and external server components for the database. You will also learn how to safely move a Model Manager server database on the file system, as well as how to restore your database from a backup in case of failure.

- Configuring Databases
- Databases with Managed Server Components
- Databases with External Server Components
- Moving a Model Manager Server Database
- Backup and Restore of a Model Manager Server Database
- Backward Compatibility for Server Databases

Read Installation Planning for general guidelines and background information to help you with planning your Model Manager server database setup.

Configuring Databases

A Model Manager server database can be set as either *active* or *inactive* depending on whether or not it should accept connections from COMSOL Multiphysics or the Model Manager server web interface. You would typically deactivate a Model Manager server database when you want to change its configuration settings or perform a

restore from backup. One of the databases can also be set as the *default* one, which is the default server database that users will connect to when accessing the Model Manager server from COMSOL Multiphysics. It is also the database initially selected as the Current Database in the web-based asset management system.

The **Databases** page, opened by clicking **Databases** in the **System** navigation sidebar, shows a table with all configured Model Manager server databases. If you selected during installation to let Model Manager server automatically set up a new database, or if you kept the suggested label when adding the database on first launch, the configuration of a single database — **Default managed database** — is initially shown in the table.

The table columns are:

- The **Label** column — the label of the database. Used for display purposes in the Model Manager server web interface.
- The **Status** column — the active and default status of a database.

Click on the label of a Model Manager server database on the **Databases** page to show more details for that database. Click the **Add** button to add a Model Manager server database.

THE DEFAULT MANAGED DATABASE

The Model Manager server database that is created if you selected **Automatic** under **Add a Model Manager server database** in the Server step of the installation uses three managed server components:

- A managed SQL database server with label **Default managed PostgreSQL® server** — see Managed SQL Database Servers.
- A managed resources directory with label **Default managed resources directory** — see Managed Resources Directories.
- A managed search index server with label **Default managed Apache Solr™ server** — see Managed Search Index Servers.

The components are created with default locations for their data directories and backup directories based on the user account running the Model Manager server process — typically on the same disk as the installation itself. These locations are not ideal as, for example, that disk may have limited storage capacity and, moreover,

backups are placed on the same physical disk as the data itself. You are strongly recommended to at least change the backup locations to a separate disk.

Place the backup directories for the managed SQL database server and the managed resources directory on a physical disk different than that of their corresponding data directories if you decide to use the default managed database. This will protect you from complete data loss in case of a server disk crash. See also Examples of Server Setups and Moving a Model Manager Server Database.

The **Automatic** option is only available if you select both the **Managed PostgreSQL®** and **Managed Apache Solr™** checkboxes in the Products step of the Model Manager server installation. If you select just one of them, you must set up the Model Manager server database manually from the web interface — see Adding the Model Manager Server Database on First Login.

- Default Data Directories
- Default Backup Directories

ADDING DATABASES

You can add a Model Manager database to the Model Manager server from the **Add Database** page. This is useful, for example, if you want to use different options than The Default Managed Database but at the same time want to keep the default database around as a reference. You would also need to add the database if you want to connect to one that already exists on the file system, possibly created from another Model Manager server installation, if you opted out from automatically creating a database in the Server step of the Model Manager server installation, or if you want to restore from backup after a server failure that also included loss of the Preference Directory — see Backup and Restore of a Model Manager Server Database.

To add a Model Manager server database:

1 Write a label for the database in the **Label** field.

The label is only used for display purposes in the Model Manager server web interface.

2 Write an optional alias for the database in the **Alias** field. When set, the alias must only contain lowercase Latin alphabet characters, numerical characters, underscores, and dashes. The alias can contain, at most, 100 characters.

The alias field is required if you want to connect to a Model Manager server database other than the default server database from a COMSOL Multiphysics installation.

3 In the **SQL Database** > **Type** list, select among the options:

- **Managed** — to use a SQL database in one of the existing managed SQL database servers or to create a new managed SQL database if none is available. See Managed SQL Databases for details.
- **External** — to use a SQL database in an external SQL database server. See External SQL Databases for details.

4 In the **Resources Directory** > **Type** list, select among the options:

- **Managed** — to use one of the existing managed resources directories or to create a new managed resources directory if none is available. See Managed Resources Directories for details.
- **External** — to use an external directory on the file system. See External Resources Directories for details.

5 In the **Search Indexes** > **Type** list, select among the options:

- **Managed** — to use search indexes in one of the existing managed search index servers or to create new managed search indexes if none are available. See Managed Search Indexes for details.
- **External** — to use search indexes in an external search index server. See External Search Indexes for details.

6 Optionally click **Test Connection** in the upper right corner to verify that the Model Manager server can successfully connect to all components using the specified configuration settings. The button is disabled when new managed server components will be created as part of the setup.

7 Click **Save** to add the database.

Saving may take some time in case new data directories are created and initialized. Upon completion, you will be redirected to the **Home** page. Otherwise, you will either be returned to the **Add Database** page or be redirected to The Database Page. Read the message at the top of the page to see why the setup did not complete.

While the Model Manager server performs the database setup, the list of steps and their current status are shown in a **Progress** dialog. Click the **Cancel** button if you want to cancel the database setup. The step currently being processed will first finish

before you are returned to the **Add Database** page. Make the necessary changes on the page and click **Save** to resume the setup.

The database must be activated before users can connect to it from COMSOL Multiphysics or the Model Manager server web interface. An attempt is made by Model Manager server to automatically activate the database when you click **Save**. See also The Database Page.

Once a resources directory or a search index has been used in combination with a SQL database for an activated Model Manager database, these components can no longer be used in combination with any other SQL database.

THE DATABASE PAGE

The **Database** page contains details on the configuration of a Model Manager database added to the Model Manager server. The configuration fields are:

- **Label**. The label of the Model Manager server database. Used for display purposes in the Model Manager server web interface.
- **Alias**. An optional alias for the Model Manager server database.

  The alias is used as an identifier to, for example, connect to the Model Manager server database from a COMSOL Multiphysics installation even when the database is not set as the default one.
- **SQL Database**. The SQL database used by the Model Manager database.
- **Resources Directory**. The resources directory used by the Model Manager database.
- **Search Indexes**. The two search indexes used by the Model Manager database.

Click the **Edit** button to edit the configuration for the database.

Click the **Test Connection** button to test if the configuration can be used to connect to the database.

Click the **Deactivate** button to deactivate the database. Users will not be able to connect to the database from COMSOL Multiphysics or the Model Manager server web interface. Click **Activate** to make the database available again.

!	Always deactivate a database before editing its configuration.

If you have configured multiple databases, you can click the **Set as Default** button to set the database as the current default. Click **Remove as Default** to remove it as the current default.

Click the **Delete Permanently** button to permanently delete the database configuration.

!	Clicking **Delete Permanently** will only delete the configuration for the Model Manager server database; it will not delete any data. You can add the database again using a new configuration.

Databases with Managed Server Components

You can configure a Model Manager database to use components managed by the Model Manager server itself.

MANAGED SQL DATABASES

On the pages for adding or editing a Model Manager database:

1. Select **Managed** in the **SQL Database** > **Type** list to use a SQL database in a managed SQL database server.
2. Select one of the managed SQL database servers in the **Server** list. If no managed SQL database server is available, you can create a new one as part of the setup:
   - **a** Write the path on the file system where the data directory for the managed SQL database server is located in the **Data directory** field. The data directory will be automatically created if it does not already exist on the file system.
   - **b** Optionally write the path on the file system where backups will be written to in the **Backup directory** field. The backup directory will be automatically created if it does not already exist on the file system.

   See also Adding Managed SQL Database Servers.

3 For an existing managed SQL database server, select a SQL database in the **Database** list. If no SQL database is available, you can create a new one as part of the setup: Write the name of the SQL database in the **Database** field. See also SQL Databases.

A SQL database in a managed SQL database server can be used by, at most, one Model Manager server database.

Managed SQL Database Servers

MANAGED RESOURCES DIRECTORIES

On the pages for adding or editing a Model Manager database:

1 Select **Managed** in the **Resources Directory** > **Type** list to use a managed resources directory.

2 Select one of the managed resources directories in the **Directory** list. If no managed resources directory is available, you can create a new one as part of the setup:

   a Write a label for the new managed resources directory in the **Label** field.

   b Write the path on the file system where the directory is located in the **Data directory** field. The data directory will be automatically created if it does not already exist on the file system.

   c Optionally write the path on the file system where the backup will be written to in the **Backup directory** field. The backup directory will be automatically created if it does not already exist on the file system.

   See also Adding Managed Resources Directories.

A managed resources directory can be used by, at most, one Model Manager server database.

Managed Resources Directories

MANAGED SEARCH INDEXES

On the pages for adding or editing a Model Manager database:

1 Select **Managed** in the **Search Indexes** > **Type** list to use search indexes in a managed search index server.

2 Select one of the managed search index servers in the **Server** list. If no managed search index server is available, you can create a new one as part of the setup: Write the path on the file system where the data directory for the managed search index server is located in the **Data directory** field. The data directory will be automatically created if it does not already exist on the file system. See also Adding Managed Search Index Servers.

3 For an existing managed search index server, select two different search indexes to use in the **Assets** and **Items** lists. If no search indexes are available, you can create new ones as part of the setup: Write the names of the search indexes in the **Assets** and **Items** fields. See also Search Indexes.

A search index in a managed search index server can be used by, at most, one Model Manager server database.

Managed Search Index Servers

Databases with External Server Components

You can configure a Model Manager database to use components maintained externally by your organization. This is useful, for example, if you already have a SQL database server installation with associated backup routines set up.

EXTERNAL SQL DATABASES

On the pages for adding or editing a Model Manager database:

1 Select **External** in the **SQL Database** > **Type** list to use a SQL database in an external SQL database server.

2 Select the type of SQL database platform in the **Platform** list. The supported platforms are:

- PostgreSQL® Database
- Microsoft SQL Server® Database
- MySQL® Database
- Oracle® Database

Set up a backup routine for the SQL database server that includes the SQL database used by the Model Manager server.

PostgreSQL® Database

1 In the **Server name** field, write the name of the computer that the PostgreSQL® database server process runs on. Defaults to `localhost`.

2 In the **Port** field, write the number of the port that the PostgreSQL® database server listens on. Defaults to `5432`.

3 In the **Database name** field, write the name of a SQL database that you want the Model Manager to use.

4 In the **Method** list, under **Authentication**, select the method of authentication. Select **Password** to authenticate using password-based authentication methods. Select **Passwordless** to authenticate using passwordless authentication methods (for example, SSPI on Windows®).

5 In the **Username** field, write the username used to authenticate with the PostgreSQL® database server.

6 In the **Password** field, write the password used to authenticate with the PostgreSQL® database server. This is only available when using password-based authentication methods.

The password will be encrypted by the Model Manager server and stored in a local SQLite® database in the Model Manager server Preference Directory— see also Password Security.

The Model Manager supports PostgreSQL® version 10.0 and newer.

Microsoft SQL Server® Database

1. In the **Server name** field, write the name of the computer that the Microsoft SQL Server® database server process runs on. Defaults to `localhost`.
2. In the **Port** field, write the number of the port that the Microsoft SQL Server® database server listens on. Defaults to `1433`.
3. In the **Database name** field, write the name of a SQL database that you want the Model Manager to use.
4. In the **Instance name** field, write the instance name of the Microsoft SQL Server® database server.
5. In the **Application name** field, write an application name to use when connecting to the database.
6. In the **TLS encryption** menu, select the TLS encryption mode used for the network communication between the Model Manager server and the Microsoft SQL Server® database server. Select **Default** to use the default mode of the database server, **Disabled** to not require TLS encryption, **Strict** to require strict TLS encryption, and **Enabled** to require non-strict TLS encryption. The strict TLS encryption mode uses version 8.0 of the Tabular Data Stream (TDS) protocol — supported in Microsoft SQL Server® 2022 and newer — allowing for a more secure handshake and supporting TLS 1.3.
7. In the **Trust server certificate** menu, select the trust mode for the certificate of the database server when TLS encryption is enabled. Select **Yes** to always trust the certificate. Select **No** if the Model Manager server must first validate the server certificate for the database server connection to succeed. Select **Default** to use the default trust mode — currently the same as **No**. If **TLS encryption** is set to **Strict**, the server certificate is always validated regardless of this setting.
8. In the **Custom hostname in certificate** field, write the hostname used when validating the certificate of the database server when TLS encryption is enabled. Leave empty to use the value of the **Server name** field.
9. Select the **Use integrated security** checkbox to authenticate using the Windows account running the Model Manager server process.
10. In the **Username** field, write the username used to authenticate with the Microsoft SQL Server® database server. Only available if the **Use integrated security** checkbox is cleared.

11 In the **Password** field, write the password used to authenticate with the Microsoft SQL Server® database server. This is only available if the **Use integrated security** checkbox is cleared.

The password will be encrypted by the Model Manager server and stored in a local SQLite® database in the Model Manager server Preference Directory— see also Password Security.

	The Model Manager supports Microsoft SQL Server® 2014 and newer.

	Microsoft SQL Server® must be installed with the Full Text Search component to use it with a Model Manager server database.

	You are strongly recommended to enable row versioning via the READ_COMMITTED_SNAPSHOT isolation level for the Microsoft SQL Server® database you intend to use with the Model Manager server database.

	For further details on the TLS encryption settings, consult Microsoft's official documentation on Microsoft JDBC Driver for SQL Server.

MySQL® Database

1 In the **Server name** field, write the name of the computer that the MySQL® database server process runs on. Defaults to `localhost`.

2 In the **Port** field, write the number of the port that the MySQL® database server listens on. Defaults to `3306`.

3 In the **Database name** field, write the name of a SQL database that you want the Model Manager to use.

4 Select the **Use SSL** checkbox to require secure connections.

5 Select the **Verify server certificate** checkbox to require that the MySQL® database server's certificate is verified.

6 In the **Username** field, write the username used to authenticate with the MySQL® database server.

7 In the **Password** field, write the password used to authenticate with the MySQL® database server.

The password will be encrypted by the Model Manager server and stored in a local SQLite® database in the Model Manager server Preference Directory— see also Password Security.

Connecting with the Model Manager server to MySQL® requires a JDBC driver that is licensed by your organization. In the file `comsolmodelmanagerserver.ini`, found inside the installation directory for the Model Manager server:

- Windows®: `<Installation directory>\bin\win64`
- Linux®: `<Installation directory>/bin/glnxa64`
- macOS: `<Installation directory>/bin/maci64`

add the line

```
-Dcs.modelmanager.mysql.jdbcdriver=<Path to JDBC driver>
```

with `<Path to JDBC driver>` being the path to the JDBC driver's JAR file on the file system.

If running as a Windows® service, the file is `comsolmodelmanagerservice.ini`.

!	The Model Manager supports MySQL® version 8.0.20 and newer.

!	A Model Manager server database requires `utf8mb4` character set encoding and that the `foreign_key_checks` system variable is enabled.

!	Some search functionality in Model Manager relies on InnoDB full-text indexes. You are strongly recommended to set
	`innodb_ft_min_token_size=1`
	`innodb_ft_enable_stopword=OFF`
	in the MySQL® configuration file.

Oracle® Database

1 In the **Server name** field, write the name of the computer that the Oracle® Database server process runs on. Defaults to `localhost`.

2 In the **Port** field, write the number of the port that the Oracle® Database server listens on. Defaults to `1521`.

3 In the **System Identifier** field, write the system identifier (SID) of the Oracle® Database instance.

Leave this field empty if you want to connect using a service name instead.

4 In the **Service name** field, write the service name for the Oracle® Database instance.

Leave this field empty if you want to connect using the system identifier instead.

5 In the **Username** field, write the username used to authenticate with the Oracle® Database server.

6 In the **Password** field, write the password used to authenticate with the Oracle® Database server.

The password will be encrypted by the Model Manager server and stored in a local SQLite® database in the Model Manager server Preference Directory— see also Password Security.

Connecting with the Model Manager server to Oracle® Database requires a JDBC driver that is licensed by your organization. In the file `comsolmodelmanagerserver.ini`, found inside the installation directory for the Model Manager server:

- Windows®: `<`*`Installation directory`*`>\bin\win64`
- Linux®: `<`*`Installation directory`*`>/bin/glnxa64`
- macOS: `<`*`Installation directory`*`>/bin/maci64`

add the line

```
-Dcs.modelmanager.oracle.jdbcdriver=<Path to JDBC driver>
```

with `<`*`Path to JDBC driver`*`>` being the path to the JDBC driver's JAR file on the file system.

If running as a Windows® service, the file is `comsolmodelmanagerservice.ini`.

The Model Manager supports Oracle® Database version 12.2.0.1 and newer.

EXTERNAL RESOURCES DIRECTORIES

On the pages for adding or editing a Model Manager database:

1 Select **External** in the **Resources Directory** > **Type** list to use an external resources directory.

2 In the **Directory** field, write the path on the file system to an existing directory.

	Set up a backup routine that backs up the resources directory — including all its subfolders and files — to a physical disk separate from that of the resources directory itself.

EXTERNAL SEARCH INDEXES

On the pages for adding or editing a Model Manager database:

1 Select **External** in the **Search Indexes** > **Type** list to use search indexes stored in an external search index server.

2 In the **Platform** list, select **Apache Solr™ server**.

3 In the **Base URL** field, write the base URL used to connect to the Apache Solr™ server. Defaults to `http://localhost:8983/solr`.

4 In the **Method** list, select the method of authentication. Select **Password** to authenticate using a username and password. Select **None** if the Apache Solr™ server does not require authentication.

5 In the **Username** field, write the username used to authenticate with the Apache Solr™ server.

6 In the **Password** field, write the password used to authenticate with the Apache Solr™ server.

The password will be encrypted by the Model Manager server and stored in a local SQLite® database in the Model Manager server Preference Directory— see also Password Security.

7 In the **Assets** and **Items** fields, write the names of Apache Solr™ cores to use as search indexes for the Model Manager server database.

	The Model Manager supports Apache Solr™ version 8.7.0 and newer.

The Model Manager search functionality uses a block join query parser to search nested documents. Make sure that the `managed-schema` files for the Apache Solr™ cores includes the element `<fieldType name="_nest_path_" class="solr.NestPathField"/>`.

A Model Manager server uses an exponential backoff strategy (up to a few minutes) if any circuit breakers configured for the Apache Solr™ server trip during indexing of documents. Trying to use the Model Manager search functionality at this point will also fail with an error message informing of insufficient resources for the current load.

You can optionally set up a backup routine that includes the Apache Solr™ cores used by the Model Manager server database. This will make the restore process run faster because there will be less search data to transfer from the SQL database to the cores.

Moving a Model Manager Server Database

You can move the data directories and backup directories of a Model Manager server database using Managed Server Components to another file system location. This is necessary, for example, if you want to keep using the **Default managed database** automatically set up during installation of a Model Manager server, but the available space on the disk containing the server installation itself is not sufficient.

The steps outlined in this section are not suitable when you want to move an entire Model Manager server installation as any server system settings, including those for accounts and authentication setups, will not be included. See Moving an Installation instead.

Moving a Model Manager server database using a managed SQL database server between Windows® and Linux® by copying the data directory or backup directory is not supported due to incompatible storage formats.

NEW INSTALLATION

You can use an existing managed Model Manager server database with a new installation of Model Manager server as follows:

1. Obtain the locations of the data directories and backup directories of the database you want to move.

   See Default Data Directories and Default Backup Directories for the locations of these directories for the default managed database added during installation. You can also find their locations in the **System** administration area of the Model Manager server web interface — see Managed Server Components for further details.

2. Stop any existing installation of Model Manager server that uses the database you want to move.

   If installed as a Windows® service (default for Windows®), you can stop the Model Manager server service using the **Stop Model Manager Server** shortcut installed on the **Start** menu under `COMSOL Launchers`.

3. Place the data directories and backup directories at their new locations by moving or copying them on the file system.

   You can choose to only include the data directories, in which case the backup directories will be automatically created when the database is added to the new installation. You can also choose to only include the backup directories, in which case the data directories will be automatically restored when the database is added. The latter could also be forced upon you if you, for example, need to restore your database from backup after a complete server disk failure — see also Restore of a Model Manager Server Database.

4. Install Model Manager server. Make sure to select **Custom** under **Add a Model Manager server database** in the **Server** step of the installation to avoid ending up with a new, empty, database.

5. Log in to the Model Manager server web interface using the username and temporary password for the administrator account specified during installation. Change your temporary password when prompted. See also Accessing the Web Interface.

   The web interface automatically navigates to the **Add Database** page as Model Manager server detects that there is no database available.

6. On the **Add Database** page, write the locations for the data directories and backup directories as specified in step **3**. Either of these directories may be empty depending on what you selected to include in step **3**. Click **Save**.

It may take the Model Manager server some time to restore the data directories if only the backup directories were included in step **3**. This will be evident by a lack of complete search results on the **Home** page until the restoration has completed.

EXISTING INSTALLATION

This section contains the necessary steps to move the data directories and backup directories for the **Default managed database** of an existing Model Manager server installation when that server is run as a Windows® service using the predefined **LocalService** user account. The new data directories are inside a hypothetical `D:\ModelManagerServer` root directory and the new backup directories are inside a `E:\ModelManagerServerBackup` root directory. You can adapt the example by skipping certain steps if you only want to move some of the directories. You can also adapt it by modifying paths if you are running with a different user account, if running on another operating system, or if moving some other Model Manager server database.

Start by creating the *parent* directories to the new data directories and backup directories via, for example, a system file explorer on the server computer:

1 Create the `ModelManagerServer` directory on the `D` drive and the `ModelManagerServerBackup` directory on the `E` drive.

Next, deactivate the Model Manager server database via the Model Manager server web interface:

2 On the **Database Configurations** page in the **System** administration area, click **Default managed database**.

3 On The Database Page for the **Default managed database**, click the **Deactivate** button.

By deactivating the database, you prevent users from accessing the database via COMSOL Multiphysics or via the Model Manager server web interface while its data directories are being moved.

Configuring Databases

You will move the directories for each of the three managed server components used by the **Default managed database** by first stopping the component's subprocess (if any), moving its directories, updating its configuration, and then starting the subprocess again. For the managed SQL database server:

4 On the **Managed SQL Database Servers** page in the **System** administration area, click **Default managed PostgreSQL® server**.

5 On The Managed SQL Database Server Page for the **Default managed PostgreSQL® server**, click the **Stop** button.

The data directory of the managed SQL database server is located inside

```
C:\Windows\ServiceProfiles\LocalService\AppData\Local\COMSOL\
ModelManager\ManagedDatabases\default-managed
```

as the `data` subdirectory. Its backup directory is located inside

```
C:\Windows\ServiceProfiles\LocalService\AppData\Local\COMSOL\
ModelManager\ManagedDatabases\default-managed-backup
```

as the `data_backup` subdirectory.

6 Move the `data` subdirectory to the `ModelManagerServer` directory on the `D` drive so that its new path becomes `D:\ModelManagerServer\data`.

7 Move the `data_backup` subdirectory to the `ModelManagerServerBackup` directory on the `E` drive so that its new path becomes `E:\ModelManagerServerBackup\data_backup`.

8 On The Managed SQL Database Server Page for the **Default managed PostgreSQL® server**, click the **Edit** button.

9 Write `D:\ModelManagerServer\data` in the **Data directory** field.

10 Write `E:\ModelManagerServerBackup\data_backup` in the **Backup directory** field.

11 Click the **Save** button.

12 Click the **Start** button to start the managed SQL database server.

Managed SQL Database Servers

The data directory and backup directory for the managed resources directory is found next to those of the managed SQL database server as the `resources` subdirectory and the `resources_backup` subdirectory respectively.

13 Move the `resources` subdirectory to the `ModelManagerServer` directory on the `D` drive so that its new path becomes `D:\ModelManagerServer\resources`.

14 Move the `resources_backup` subdirectory to the `ModelManagerServerBackup` directory on the `E` drive so that its new path becomes `E:\ModelManagerServerBackup\resources_backup`.

15 On the **Managed Resources Directories** page in the **System** administration area, click **Default managed resources directory**.

16 On The Managed Resources Directory Page for the **Default managed resources directory**, click the **Edit** button.

17 Write `D:\ModelManagerServer\resources` in the **Data directory** field.

18 Write `E:\ModelManagerServerBackup\resources_backup` in the **Backup directory** field.

19 Click the **Save** button.

Managed Resources Directories

For the managed search index server:

20 On the **Managed Search Index Servers** page in the **System** administration area, click **Default managed Apache Solr™ server**.

21 On The Managed Search Index Server Page for the **Default managed Apache Solr™ server**, click the **Stop** button.

The data directory for the managed search index server is found next to that of the managed SQL database server as the `index` subdirectory. It has no backup directory.

22 Move the `index` subdirectory to the `ModelManagerServer` directory on the `D` drive so that its new path becomes `D:\ModelManagerServer\index`.

23 On The Managed Search Index Server Page for the **Default managed Apache Solr™ server**, click the **Edit** button.

24 Write `D:\ModelManagerServer\index` in the **Data directory** field.

25 Click the **Save** button.

26 Click the **Start** button to start the managed search index server.

Managed Search Index Servers

Finally, activate the Model Manager server database:

27 On The Database Page for the **Default managed database**, click the **Activate** button.

The Model Manager server database is now accessible again.

Backup and Restore of a Model Manager Server Database

You should configure backup of your Model Manager server database to protect from data loss in case of, for example, a server disk crash. Backup for a database using Managed Server Components can be configured from the web interface and is handled automatically by the Model Manager server itself. Backup for a database using external components needs to be configured and maintained outside of the Model Manager server by your organization.

BACKUP OF A MODEL MANAGER SERVER DATABASE

Backup for a Model Manager server database involves configuring separate backups for its SQL database and its resources directory. The search indexes of a Model Manager server database do not require backup since they can be recreated from the data in the SQL database.

For a Model Manager server database using Managed Server Components with backup enabled, Model Manager server will continuously back up any data written to the managed SQL database server and to the managed resources directory to their respective backup directories — see Backup of a Managed SQL Database Server and Backup of a Managed Resources Directory. These two backup directories together form the complete backup of the Model Manager server database itself.

For a Model Manager server database using external server components, you must manually set up backup routines of their data using external backup software. For External SQL Databases, this involves a backup of the SQL database used by the Model Manager server database. For External Resources Directories, this involves a backup of the directory itself, including all its subfolders and files.

Read Installation Planning for general guidelines on server setups and where to place backup directories.

RESTORE OF A MODEL MANAGER SERVER DATABASE

The steps required to restore a Model Manager server database from backup depend on how much has been lost in terms of configuration and if the database uses managed server components, external server components, or possibly a mixture thereof.

Configuration Settings Unavailable

If the Model Manager server Preference Directory has been lost, and there is no backup of that directory available, you will first need to recreate the configuration for the Model Manager server database. Proceed as follows:

1. Reinstall the Model Manager server — see Modifying an Installation. Make sure to select **Custom** under **Add a Model Manager server database** in the **Server** step of the installation to avoid ending up with a new, empty, database.
2. Log in to the Model Manager server web interface using the username and temporary password for the administrator account specified during installation. Change your temporary password when prompted. See also Accessing the Web Interface.
3. On the **Add Database** page — which you will be redirected to after saving your new password — configure the server components for the Model Manager database:
   - For a managed SQL database server or a managed resources directory, write the location for a directory that is either empty or missing from the file system in the **Data directory** field. Write the location for your existing backup directory in the **Backup directory** field.
   - For an external SQL database server or an external resources directory, make sure that the component has already been restored from its backup. Write the configuration settings used to access the component.
   - For a managed or external search index server, write the configuration settings for the component. In this case, there is no need to restore anything from backup.

   See also Adding Databases.
4. Click the **Save** button. Model Manager server proceeds by adding a Model Manager database with all its data restored from the backup locations.

   It may take Model Manager some time to restore the data directories. This will be evident by a lack of complete search results on the **Home** page until the restoration has completed.

As an alternative to immediately proceeding with step **3**, you can also manually restore each managed server component separately before adding the Model Manager database. This is useful, for example, when the backup directory of the managed SQL database server contains multiple SQL databases, and you want to take control over which one is used for the restored Model Manager database. Begin by recreating the configuration of each managed server component:

- Adding Managed SQL Database Servers

- Adding Managed Resources Directories
- Adding Managed Search Index Servers

Remember to specify the path for the backup directory of the managed SQL database server, as well as for the backup directory of the managed resources directory. If you set the **Trigger mode** for restoring to **Automatic (most recent)** for the managed SQL database server, and its data directory is missing on disk, the data directory will be automatically restored from backup when the server starts. Similarly, if the **Trigger mode** for restoring is set to **Automatic**, a managed resources directory is automatically restored if the directory is missing on disk.

Proceed with step **3** to recreate a configuration for the Model Manager database — optionally replacing managed server components with external server components.

Configuration Settings Available

When a configuration for the Model Manager server database is available:

1. If currently activated, deactivate the Model Manager server database — see The Database Page for instructions.
2. Restore the SQL database:
   - For an external SQL database server, use the restore functionality specific to the platform.
   - For a managed SQL database server, click the **Trigger Restore** button on the **Managed SQL Database Server** page unless a restore has already been triggered automatically. Wait until the restore finishes. See also Restore of a Managed SQL Database Server.
3. Restore the resources directory:
   - For an external resources directory, copy files from the backup directory to their corresponding location in the resources directory — optionally skipping files already present in the resources directory.
   - For a managed resources directory, click the **Trigger Restore** button on the **Managed Resources Directory** page unless a restore will trigger automatically. The restore will begin once the database is activated again; see below. See also Restore of a Managed Resources Directory.

4 Restore the search indexes:

- For an external search index server, recreate the search indexes used by the Model Manager server database or, if you have one available, restore the search index server from backup.
- For a managed search index server, follow the steps in Restore of a Managed Search Index Server.

5 Activate the database if not already activated by a previous step.

At this point, if the Model Manager server database uses a managed resources directory, all missing files will be restored from the backup directory to the data directory. All search indexes will also be populated with search data loaded from the SQL database as needed. The time for both of these steps to complete depends on the size of the database.

When the Model Manager server database has been activated, it is possible to connect to the database from COMSOL Multiphysics. Search results may be incomplete, however, until all search indexes have been repopulated with search data. Some simulation models and auxiliary data files may also fail to load if they depend on data in the resources directory that have yet to be copied from the backup directory by the ongoing restore.

Backward Compatibility for Server Databases

When a newer version of a Model Manager server connects to a server database created using an older Model Manager server, the database will be automatically upgraded to support any functionality added in the newer version. This upgrade does not modify the storage format of models and data files, however, so older COMSOL Multiphysics versions connecting to the Model Manager server will still be able to open models and data files in the database even after this upgrade.

Models saved in a Model Manager server database have the same version requirement constraint as models stored as MPH-files on the file system: a model saved to a database from a newer version of COMSOL Multiphysics cannot be opened from an older version of COMSOL Multiphysics.

Model Manager Server Log Files

Administrators can download log files generated by the Model Manager server directly from the web interface. This is useful, for example, if you do not have easy access to the file system containing the Model Manager server's Preference Directory.

1. In the **System** administration area, under **Monitor**, click **Logs** to open the **Logs** administration page.
2. In the **Files** menu, select **All** to download all available log files. Select **Last Modified** to only download log files that were last modified between the start of the day specified in the **From** field and the end of the day specified in the **To** field.
3. Click **Download**.

   The logs are downloaded as a compressed zip archive to your computer.

Database Administration

You can administer the Current Database from the database administration area of the web interface. Click the cog wheel in the top navigation bar and select **Database** to open this area. Although all users may access the database administration area, only users granted the appropriate database permissions are authorized to save any changes.

The **Database** navigation sidebar is divided into four sections: **Security**, **Assets**, **Attributes**, and **Workflows**. The **Security** section is similar to the **Security** node in the **Databases** tree in the Model Manager workspace of COMSOL Multiphysics. You can, for example, manage local group memberships of users or grant general database permissions to users and groups.

The **Assets**, **Attributes**, and **Workflows** sections contains pages for administrating and extending the asset management system included with a Model Manager server. You may, for example, want to add new asset libraries with custom permissions or add new types of assets with their own sets of attributes.

In this section:

- Users
- Groups
- Database Permissions
- Permission Templates
- Asset Libraries
- Asset Types
- Primitive Attributes
- Composite Attributes
- Primitive Attribute Workflows
- Composite Attribute Workflows

See the *Model Manager Reference Manual* for more information on user management and access control in a Model Manager database.

Read the Asset Management chapter to learn more about how the web-based asset management system can be used to organize and share simulation projects within an organization.

Users

A *user* in a Model Manager database is primarily used to identify the individual that has saved a database object, the owner of a database object, or the individuals that have been granted permissions for a database object. A new user is automatically created based on your account the first time you connect to a Model Manager server database.

See Users in the *Model Manager Reference Manual*.

You can manage users in the database from the **Users** page, opened by clicking **Users** in the **Database** navigation sidebar. You can, for example, see all users that have been active in the database or save group memberships in the database. The latter is useful, for example, if you are unable to provide group memberships via an external credentials storage — see External Authentication.

The page shows a table containing:

- The **Name** column — the unique username of a user. This will be matched with the username of a corresponding account during authentication with the Model Manager server.
- The **Display Name** column — an alternative display name of a user to be used for display purposes.

Select the **Show deleted** checkbox to include deleted users in the table.

Click on a user in the **Name** column to show more details for that user. Click the **Add** button to add a new user.

ADDING USERS

While a user is automatically created as needed when connecting to a Model Manager server database, there may be situations when you want to manually add a new user to the database before that user has connected. One example is when you preemptively want to grant them permissions to various database objects.

To add a new user:

1. Write the username of the user in the **Name** field. This username must match that of the account used when authenticating with the Model Manager server — see Accounts.
2. Write an alternative display name for the new user in the **Display Name** field. This is the name that will primarily be shown in the user interface.

   The display name defaults to the username if left empty.
3. All groups that the new user will be a member of are shown as a list under **Group memberships**. Select a group from the **Add group membership** list to add the user as a member. The group membership list is hidden if there are no groups in the database.
4. Click **Save** to add the new user.

You do not need to specify any group memberships if you are using External Authentication and you have set up group mappings — see Adding Group Mappings. The same goes if using Proxy Authentication with a group name header.

THE USER PAGE

The **User** page contains details on a user in the database, including the display name and all group memberships that are stored explicitly in the database. The fields are:

- **Name**. The unique username of the user in the database. This is mapped to a corresponding account via the account's username — see also The Account Page.
- **Display name**. The name used for display purposes in the Model Manager server web interface.
- **Group memberships**. The groups that the user is a member of. This does not include groups mapped to the corresponding account by a Login Configuration — see The My Account Page.

Group membership is transitive: if a user is a member of a group, which in turn is a member of another group, the user is considered a member of the latter group as well.

Click **Edit** to change the display name or group memberships of the user. Click **Delete** to delete the user in the database. The deletion is not permanent — you can restore the deleted user by clicking **Restore**.

Groups

A *group* in a Model Manager database is a collection of users and other groups. Groups are used to more easily manage permissions common to several users in the database. You can either create groups manually in the database or, by Adding Group Mappings to an External Authentication mechanism, let them be automatically created based on the principal names returned by the external credentials storage. Groups will also be created automatically if using Proxy Authentication with a group name header.

See Groups in the *Model Manager Reference Manual*.

The **Groups** page, opened by clicking **Groups** in the **Database** navigation sidebar, shows a table with all groups in the database. The table columns are:

- The **Name** column — the unique name of a group.
- The **Display Name** column — an alternative display name of a group to be used for display purposes.

Select the **Show deleted** checkbox to include deleted groups in the table.

Click on a group in the **Name** column of the table to show more details for that group. Click the **Add** button to add a new group.

ADDING GROUPS

To add a new group:

1. Write the name of the group in the **Name** field. This name must match the principal name returned by the external credentials storage if you want the added group to be associated with a principal in the storage.
2. Write an alternative display name for the new group in the **Display Name** field. This is the name that will primarily be shown in the user interface.

   The display name defaults to the group name if left out.
3. All members of the new group are shown as a list under **Group members**. Select a user or group from the **Add group member** list to add them as a member.

4 Click **Save** to add the new group.

You do not need to specify any group members for the group if you are using External Authentication and you have set up explicit group mappings — see Adding Group Mappings. The same goes if you are using Proxy Authentication with a group name header.

THE GROUP PAGE

The **Group** page contains details on a group in the database, including the display name and all group members that are stored explicitly in the database. The fields are:

- **Name**. The unique name of the group in the database.
- **Display name**. The name used for display purposes in the Model Manager server web interface.
- **Group members**. The members of the group. This does not include members whose group membership is mapped by a Login Configuration — see The My Account Page.

Click **Edit** to change the display name or group members of the group. Click **Delete** to delete the group in the database. The deletion is not permanent — you can restore the deleted group by clicking **Restore**.

Database Permissions

Administrators can delegate a few administrative tasks to users by granting them permissions for actions that target the database itself. The **Database permissions** page, opened by clicking **Database Permissions** in the **Database** navigation sidebar, shows a table with users and groups mapped to their granted permissions. The table columns are:

- The **Name** column — the name of a user or group.
- The **Permissions** column — the database permissions granted to the user or group.

The table is initially empty, which means that only administrators can perform these administrative tasks.

See Access Control in the *Model Manager Reference Manual*.

Click the **Edit** button to edit the table of granted database permissions.

To grant permissions to users or groups:

1. Select a user or group from the **Add** list.
2. In the added table row, select the permissions to grant from the **Permissions** list.
3. Repeat for other users or groups.
4. Click the **Save** button to save the table.

Most of the available permissions are listed in the Permission Catalog in the *Model Manager Reference Manual*. The remaining ones apply to the asset management system and are listed in Table 3-1.

TABLE 3-1: AVAILABLE ASSET MANAGEMENT SYSTEM PERMISSIONS FOR THE DATABASE ITSELF.

PERMISSION	DESCRIPTION
Add asset libraries	Allowed to create a new asset library.
Manage asset types	Allowed to create, edit, and delete asset types and attributes.
Permanently delete assets	Allowed to permanently delete assets.

Permission Templates

You can create *permission templates* in a Model Manager database as a way of reusing permission assignments for different assets, models, or data files. This saves you the work of manually granting the same set of permissions to users and groups for multiple database object. Creating permission templates also allows you to propagate permission requirement changes to multiple database objects by only updating the permissions in one place — the permission assignments of the permission template itself.

See Permission Templates in the *Model Manager Reference Manual*.

The **Permission Templates** page, opened by clicking **Permission Templates** in the **Database** navigation sidebar, shows a table with all user-defined permission templates in the database. The table columns are:

- The **Name** column — the name of a permission template.
- The **Type** column — the type of database objects that the permission template applies to.

Select the **Show deleted** checkbox to include deleted permission templates in the table. You can also filter the table on permission template types from the **Type** list.

Click on the name of a permission template in the table to show more details for that permission template. Click the **Add** button to add a new permission template.

ADDING PERMISSION TEMPLATES

To add a new user-defined permission template:

1. Write a name for the permission template in the **Name** field.
2. Select the type of permission template to create in the **Type** list. You can select between **Asset**, **File**, or **Model**.
3. Select the user or group you want to grant permissions for in the **Add** list. Added users or groups shows up as table rows. Select each user's or group's granted permissions in the **Permissions** list.
4. Click the **Save** button to add the new permission template.

THE PERMISSION TEMPLATE PAGE

The **Permission Template** page contains details on a user-defined permission template in the database. The fields are:

- **Name**. The name of the permission template.
- **Type**. The type of database objects that the permission template applies to.

The permissions that are granted by the permission template are shown in a table.

Click **Edit** to edit the permission template.

Updating the permission assignments for a permission template affects not only future applications of the template but also the permissions of any database objects to which the permission template has already been applied.

Click **Delete** to delete the permission template in the database. This deletion is not permanent — you can restore the deleted template by clicking **Restore**.

Click the expander button next to the **Delete** button and select **Delete Permanently** if you *do* want to permanently delete the permission template in the database. Permanent deletion will only succeed if the permission template is not assigned to any database objects.

Asset Libraries

An *asset library* is a container for a collection of assets and their versions in a Model Manager database. When the database is created, a first asset library — **Asset library 1** — is automatically added for you. You would typically add more libraries to the database if you want to restrict access to a collection of assets for a particular set of Users and Groups. By granting permissions on the asset library, you can, for example, restrict who is able to browse and search the assets in the library.

See Controlling Access to Assets Using Libraries.

The **Asset Libraries** page, opened by clicking **Asset Libraries** in the **Database** navigation sidebar, shows a table with all asset libraries in the database. The table columns are:

- The **Name** column — the name of the asset library.
- The **Owner** column — the name of the user set as the current owner of the asset library.

Select the **Show deleted** checkbox to include deleted asset libraries in the table.

Click on an asset library in the **Name** column of the table to show more details for that asset library. Click the **Add** button to add a new asset library. Write a name for the new asset library in the **Name** field. Click **Save**.

THE ASSET LIBRARY PAGE

The **Asset Library** page contains details on an asset library in the database. Click **Rename** to change the name of the asset library. Click **Delete** to delete the asset library in the database. The deletion is not permanent — you can restore the deleted asset library by clicking **Restore**.

Click **Owner** to transfer ownership of the asset library to another user. Only the current owner or an administrator can transfer ownership of the asset library. The user that creates the asset library is automatically set as its initial owner.

See Owners in the *Model Manager Reference Manual*.

Click **Permissions** to grant permissions for the asset library to users or groups. Only the current owner or an administrator can grant permissions for the asset library. See Table 3-2 for the permissions available for an asset library.

TABLE 3-2: ALL AVAILABLE PERMISSIONS FOR ASSET LIBRARIES.

PERMISSION	DESCRIPTION
Delete asset library	Allowed to delete the asset library.
Restore asset library	Allowed to restore the asset library when deleted.
Save in asset library	Allowed to save assets as well as perform any other save operation inside the asset library.
Save asset library	Allowed to save the asset library itself to, for example, rename it.
See asset library	Allowed to see the asset library in the user interface. This is a necessary permission for browsing assets inside the asset library.

See Granting Permissions in the *Model Manager Reference Manual*.

When the Model Manager authorizes a database action that targets an asset in the asset management system, it consults up to three levels of protection: the asset library that the asset belongs to, the workflow for the asset's asset type (if any), and, possibly, the asset itself. See Table 3-5 for all permissions available for asset type workflows. See Table 4-1 for all permissions available for assets.

The necessary permission combinations for possible database actions targeting assets are summarized as follows:

TABLE 3-3: NECESSARY PERMISSION COMBINATIONS FOR PERFORMING DATABASE ACTIONS THAT TARGET ASSETS OF A SPECIFIC ASSET TYPE (WITH AN ASSOCIATED WORKFLOW), AND BELONGING TO A SPECIFIC ASSET LIBRARY.

ACTION	ASSET LIBRARY	ASSET TYPE	ASSET
Browse and search assets	See asset library	Open asset	Open asset
Create a new asset	Save in asset library	Create asset	N/A
Open an asset page	See asset library	Open asset	Open asset
Save a new asset version	Save in asset library	Save asset	Save asset
Delete an asset	Save in asset library	Delete asset	Delete asset
Restore an asset	Save in asset library	Restore asset	Restore asset

Asset Types

You use an *asset type* to define the collection of data fields, or *attributes*, that can be edited on assets having that particular asset type. When the database is created, a first asset type — simply named **Asset** — using a few predefined attributes is automatically created for you. You can modify this asset type by adding new attributes to its definition, or you can create entirely new asset types using a combination of predefined and new attributes.

An asset type has a list of *sections*, with each section having a title and a subset of the asset type's attributes. The order of the attributes in each section, as well as the overall order of the sections themselves, determines how the corresponding **Asset** page for an asset having that particular asset type is displayed.

An asset type may also be associated with *workflows*. Workflows enables you to set up organizational processes and rules to help you manage the lifecycle of assets. This includes, for example, defining various stages that an asset can go through and the available transitions between those stages.

The **Asset Types** page, opened by clicking **Asset Types** in the **Database** navigation sidebar, shows a table with all asset types in the database. The table columns are:

- The **Name** column — the name of the asset type.
- The **Alias** column — the unique shorthand alias for the asset type.

Select the **Show deleted** checkbox to include deleted asset types in the table.

Click on an asset type in the **Name** column of the table to show more details for that asset type. Click the **Add** button to add a new asset type.

See also Adding New Types of Assets and Using Workflows to Set Up Organizational Processes.

ADDING ASSET TYPES

To add a new asset type:

1. Write a name for the asset type in the **Name** field.
2. Write a shorthand alias for the asset type in the **Alias** field. The alias must be unique among the set of all aliases and start with an uppercase Latin alphabet character followed by zero or more such characters, numbers, or underscores. The alias can contain, at most, 16 characters.

   The alias is used in web links for assets of this asset type, as seen, for example, in the web browser's address bar on the **Asset** page. Once set for an asset type, it cannot be changed.
3. Attributes defined for the asset type are added via sections. Under **Sections**:
   a. Write a title for the section in the **Title** field.
   b. Add attributes to the section in the **Add attribute** list — either by creating a new attribute in the database or by selecting an existing one. Added attributes appear in the **Attributes** table. You can rearrange the attributes in the order they should appear in the section using drag and drop — see also Rearranging Table Rows.

      Select **New primitive attribute** in the **Add attribute** list to both create a new primitive attribute in the database and add it to the section — see Adding Primitive Attributes for the available fields. Select **New composite attribute** to do the same for a new composite attribute — see Adding Composite Attributes. A created primitive attribute or composite attribute may later be reused for other asset types as well.

      Click **Edit** (the pen icon) next to an attribute added via the **New primitive attribute** option or the **New composite attribute** option to edit its fields. Click **Delete Permanently** (the cross icon) to permanently delete the attribute from the database.
4. Click the up and down arrows to rearrange the sections in the order you want them to appear on an asset page.

5 Click the **Workflow** menu option and select the **Use workflows** checkbox to enable the use of workflows with the asset type — see Using Workflows with Asset Types for details. Click **Preview** to see a preview presentation of the corresponding **Asset** page populated with some placeholder data for an hypothetical asset having the new type. Click **Sections** to return to the initial section presentation.

6 Click **Save** to add the new asset type.

Different asset types can reuse the same attributes. A single attribute can, however, only appear once as a top-level member on an asset type. An option is to add different composite attributes containing the same primitive attribute — see Primitive Attributes and Composite Attributes.

Any new primitive attribute or composite attribute created via the **Add Asset Type** page will remain in the database even if you cancel the asset type creation. You can delete the attribute via The Primitive Attribute Page or The Composite Attribute Page, respectively.

USING WORKFLOWS WITH ASSET TYPES

You apply Primitive Attribute Workflows and Composite Attribute Workflows to assets by adding them to the assets' corresponding asset types. The workflows for attributes may be shared between different asset types analogous to how attributes themselves may be shared.

The workflow management system in Model Manager server also enables you to apply workflow rules for the asset type itself via an *asset type workflow*. You can, for example, set up Transitions with associated Activities for this workflow so that updating the value of one attribute automatically sets the value for another attribute. You can also grant permissions to assets based on their current attribute values, including ones for opening their **Asset** pages or saving new asset versions.

See also Using Workflows to Set Up Organizational Processes.

The workflow state for an asset type workflow is given by the current values of its attributes — both primitive and composite — on the **Asset** page. Workflow transitions involve modifying these attribute values by saving a new asset version. You can define at most one such workflow for each asset type.

You enable workflows for asset types when adding or editing the asset type:

1. Click the **Workflow** button.
2. Select the **Use workflows** checkbox.
3. Write a label for the workflow of the asset type in the **Label** field. The field is automatically populated with the name of the asset type.
4. Write an optional description in the **Description** field.

An asset type workflow contains lists of Member Workflows, State Conditions, Transitions, Activities, and Permissions. A list can be left empty.

Member Workflows

To apply a primitive or composite attribute workflow on the **Asset** page, you must first add it as a member workflow to the corresponding asset type workflow. Select the workflow from the **Add workflow** list under the **Member workflows** table. Only workflows for attributes that are added to the asset type's sections are available for selection — workflows for member attributes of composite attributes must be added via composite attribute workflows. Multiple member workflows may be added for the same attribute. You can rearrange the member workflows using drag and drop — see also Rearranging Table Rows. The top-to-bottom sort order reflects the order in which the workflows are evaluated when saving an asset.

You introduce default values for new assets by adding member workflows with state conditions set as default state conditions — see Primitive Attribute Workflows.

State Conditions

State conditions for asset type workflows are defined similar to those of Composite Attribute Workflows. Asset type workflows supports state conditions of type **State expression** and the built-in state conditions listed in Table 3-4. State values are not supported.

To add a state condition to the workflow:

1. Select the type of state condition to add in the **Add State Condition** menu.
2. For a **State expression**, write its label and optional description in the **Label** and **Description** field, respectively.
3. For a **State expression**, specify the expression statement in the **Expression** field. You write the statement by combining Primitive Attribute Field Expressions and

Composite Attribute Field Expressions using boolean `AND` and `OR` operators, `NOT` and `ANY` operators, and by grouping them with parentheses. Unlike composite attribute workflows, expressions involving member attributes of a composite attribute must include the identifier of the composite attribute itself.

Only primitive attributes of value type **Boolean**, **Integer**, **Keyword**, **Keyword array**, **User**, and **User array** can be used in a state expression. There are no restrictions on composite attributes.

TABLE 3-4: ALL AVAILABLE STATE CONDITIONS OF BUILT-IN TYPE FOR AN ASSET TYPE WORKFLOW..

STATE CONDITION	DESCRIPTION
Always	A condition that is always satisfied. Combines the **Any state value** and **No state value** built-in state conditions.
Any state value	A condition that is satisfied if at least one attribute on the asset has a non-empty value.
Never	A condition that is never satisfied.
No state value	A condition that is satisfied if all attributes on the asset have an empty value.

A state condition of type **Never** is primarily useful to temporarily disable a transition by adding it as a from-state condition or as a to-state condition.

Transitions

Transitions for an asset type workflow correspond to changes made to the overall attribute values when saving an asset. For the from-state and to- state conditions, you can use any state condition belonging to the asset type workflow itself, as well as any state condition from a member workflow. Member state conditions are evaluated using the same grouping-logic as applied for the transitions of Composite Attribute Workflows. Action transitions are not supported for an asset type workflow.

To add a transition to the workflow:

1. Click **Add Transition**.
2. Write the label of the transition in the **Label** field. Write an optional description in the **Description** field.
3. In the **From** and **To** field, select state conditions to add as from-state and to-state conditions for the transition, respectively.

   You can select any number of state conditions belonging to the workflow itself and to its member workflows.

4 In the **Permissions** table for the transition, add users and groups that are granted permission to perform the transition. Leave the table empty to grant everyone permission.

   In the **Permissions with State Conditions** table cell, next to the **Perform transition** permission type, add state conditions from the **Add State Condition** menu to conditionally grant the permission. The permission is granted to the user or group if at least one state condition is satisfied before saving the asset. The permission is always granted if the collection of state conditions is empty.

   See also Permissions for permission types granted to the assets themselves.

Activities

The same activities available for the transitions of Composite Attribute Workflows are also available for the workflow of an asset type.

To add an activity:

1 Select either **Perform transition** or **Transition cancellation** in the **Add Activity** menu below a transition.

2 Write the label of the activity in the **Label** field. Write an optional description in the **Description** field.

3 For a **Perform transition** activity, select transitions from the **Add Transition** menu.

   For a **Transition cancellation** activity, select state conditions from the **Add State Condition** menu.

Permissions

You can grant shared permissions to all assets of a specific asset type by adding permissions to the workflow of the asset type. The available permission types are listed in Table 3-5.

To add permissions:

1 Add a user or group to the **Permissions** table at the bottom of the page.

2 In the **Permissions with State Conditions** table cell, select checkboxes for the permission types to grant for the user or group.

3 For each permission type, add state conditions from the **Add State Condition** menu to conditionally grant the permission. A state condition can either belong to the asset type workflow itself or to one of its member workflows.

A permission is granted to the user or group if at least one state condition is satisfied by the attribute values before saving the asset. The permission is always granted if the collection of state conditions is empty.

Adding the built-in state conditions **Current user** or **Not current user** of a member workflow to an **Open asset** permission entry is not recommended. The first condition is *never* satisfied, and the second condition is *always* satisfied, when applying permissions to a search result for assets — regardless if the current user is assigned to the relevant **User picker** attribute or not.

A state condition from a member workflow for a composite attribute in **Table** mode is satisfied if at least one table row satisfies the condition. The state condition is always satisfied if the table is empty.

When non-empty, the permissions in the **Permissions** table adds an additional permission layer to the layers defined by the permissions for the asset itself and the asset library that the asset belongs to — see Granting Permissions to an Asset and Asset Libraries. Unlike the latter two layers, however, the access control imposed by the asset type workflow may depend on the specific attribute values for an asset (via the use of state conditions). A user may be granted permissions for some versions of an asset, while denied for other versions of the same asset.

Opening the **Asset Versions** page for an asset requires that the user is granted the **Open asset** permission for the *latest* version of the asset. Users may thus see, for example, the title of older asset versions even if they are not permitted to open their corresponding **Asset** page.

TABLE 3-5: ALL AVAILABLE PERMISSIONS FOR ASSET TYPE WORKFLOWS.

PERMISSION	DESCRIPTION
Create asset	Allowed to create an asset with the asset type.
Open asset	Allowed to see and open an asset with the asset type.
Save asset	Allowed to save new versions of an asset with the asset type.

PERMISSION	DESCRIPTION
Delete asset	Allowed to delete an asset with the asset type.
Restore asset	Allowed to restore an asset with the asset type when the asset is deleted.

THE ASSET TYPE PAGE

The **Asset Type** page contains details on an asset type in the database. The fields are:

- **Name**. The name of the asset type.
- **Alias**. The alias of the asset type.

Click the **Sections** menu option (default selected) to see the list of all sections and their corresponding attributes for the asset type. Click **Workflow** to see the workflow defined for the asset type. The **Workflow** option is disabled if the asset type does not use workflows — see Using Workflows with Asset Types for details. Click **Preview** to see a preview presentation of the corresponding **Asset** page populated with some placeholder data for an hypothetical asset having the current type.

Click **Edit** to edit the asset type, including adding, rearranging, and removing its sections or attributes, as well as enabling the use of workflows.

Click **Duplicate** to open the **Add Asset Type** page filled in advance with the current asset type's sections and attributes as a starting point.

Click **Delete** to delete the asset type in the database. The deletion is not permanent — you can restore the deleted asset type by clicking **Restore**.

Click the expander button next to the **Delete** button and select **Delete Permanently** if you *do* want to permanently delete the asset type in the database. Permanent deletion will only succeed if the asset type is not assigned to any assets.

EXAMPLES OF WORKFLOWS FOR ASSET TYPES

This section lists a few applications of workflows for asset types. See also Examples of Workflows for Primitive Attributes and Examples of Workflows for Composite Attributes.

Started Timestamp

Add a `Started` primitive attribute of widget type **Datetime** picker to an asset type. Define a **Free workflow** for the attribute with an action transition:

```
Label: Set current time
From: No state value
To: Current timestamp
```

Add this workflow, together with the `Status` workflow from Examples of Workflows for Primitive Attributes, to the workflow of the asset type. Define a transition for the asset type workflow:

```
Label: Set started
From: Status / Approved
To: Status / Implementing
Perform transition activity: Started / Set current time
```

The transition fires when the `Status` attribute value changes from `Approved` to `Implementing`. As a side-effect, the `Started` attribute value is set to the current timestamp.

A **Change value** permission could be added to the `Started` workflow that prevents users from manually changing the timestamp.

Automatic User Assignment

Add the `Responsible` and `Project Proposal` workflows from Examples of Workflows for Primitive Attributes to the workflow of an asset type. Define a transition for the asset type workflow:

```
Label: Assign responsible
From: Project Proposal / Draft
To: [Project Proposal / Pending Approval by Engineering,
     Project Proposal / Pending Approval by Sales]
Perform transition activity: Responsible / Assign to Me
```

The transition fires when the `Project Proposal` attribute value changes from `[Draft]` to `[Pending Approval by Engineering, Pending Approval by Sales]`, which happens, for example, when a user clicks the **Request Approval** button on the **Asset** page. As a side-effect, the user requesting approval is assigned to the `Responsible` attribute.

Deny Approval by Responsible

Continuing with the previous example, add a transition:

```
Label: Deny responsible
From: [Project Proposal / Approved by Engineering,
       Project Proposal / Approved by Sales]
To: Project Proposal / Approved by Management
Transition cancellation activity: Responsible / Current user
```

The transition fires when the `Project Proposal` attribute value changes from `[Approved by Engineering, Approved by Sales]` to `[Approved by Management]`, which happens, for example, when a user clicks the **Approve**

(Management) button on the **Asset** page. As a side-effect, the save is canceled if the manager trying to approve is currently assigned to the `Responsible` attribute.

The same business rule could have been accomplished by granting a **Perform transition** permission for managers to the transition with a state condition **Not current user** from the `Responsible` member workflow.

Restricted Asset Version

Add the `Status` workflow from Examples of Workflows for Primitive Attributes to the workflow of an asset type. Define two **State expression** conditions for the asset type workflow:

```
Label: Not suggested
Expression: NOT @status:suggested

Label: Approved/Implementing
Expression: @status:(approved OR implementing)
```

Conditionally grant the **Open asset** and **Save asset** permissions to a `Project Manager` group and a `Simulation Engineer` group using these state conditions:

```
Project Manager          Open asset
                         Save asset
Simulation Engineer      Open asset    Not suggested
                         Save asset    Approved/Implementing
```

Only the `Project Manager` group can open assets in the `Suggested` stage and save new versions of assets in the `Suggested`, `Completed`, and `Canceled` stages.

The same permissions for the `Simulation Engineer` group could have been accomplished by explicitly listing those state conditions from the `Status` workflow that do allow access.

Hide Modeling Tasks

Add the `Modeling Tasks` workflow from Examples of Workflows for Composite Attributes to the workflow of an asset type. Define a **State expression** condition for the asset type workflow:

```
Label: CAD provided
Expression: NOT @modeling_task:CAD
```

Conditionally grant the **Open asset** permission to a `Simulation Engineer` group using this state condition:

```
Simulation Engineer      Open asset    CAD provided
```

Assets for which all models are still in the `CAD` step will now be hidden from users in the `Simulation Engineer` group; visible assets will either have no model versions at all or at least one that has moved beyond the `CAD` step.

Primitive Attributes

You define the data fields available for a particular asset type using *primitive attributes*. A primitive attribute added to an asset type will appear as an editable data field on all assets having that asset type.

See Adding New Types of Data Using Attributes.

Each primitive attribute has a *value type* defining the kind of data it can store and how you can search on that data. This can be a simple type, such as a text or a number, or an advanced type, such as a link to a model version stored in the database. All available value types are listed in Table 3-7.

A primitive attribute also has a *widget type* defining how the data is edited in the web interface. This can, for example, be an input field, a list of options in a combo box, or a picker for selecting a model version. All available widget types are listed in Table 3-6.

Given a particular widget type, only a subset of all value types are available to combine the widget type with. Some widget types also require a set of *allowed values* for the data, which becomes an additional constraint on the data that a primitive attribute can store.

When a Model Manager server database is created, two primitive attributes — **Attachment** and **Model Version** — are automatically created for you. They correspond, respectively, to an uploaded file attachment and a link to a model version in a repository.

The **Primitive Attributes** page, opened by clicking **Primitive Attributes** in the **Database** navigation sidebar, shows a table with all primitive attributes in the database. The table columns are:

- The **Label** column — the label of the primitive attribute.
- The **Identifier** column — the unique identifier of the primitive attribute.
- The **Widget** column — the type of widget used when editing the primitive attribute.

- The **Value** column — the type of value that can be set for the primitive attribute.
- The **Description** column — the optional description of the primitive attribute.

Click on the label of a primitive attribute in the table to show more details for that primitive attribute. Click the **Add** button to add a new primitive attribute.

ADDING PRIMITIVE ATTRIBUTES

To add a new primitive attribute:

1. Write a label for the primitive attribute in the **Label** field. The label is shown on an asset page next to the asset's attribute value.
2. Write an identifier for the primitive attribute in the **Identifier** field. This identifier must be unique among the set of all attributes — primitive and composite — and start with a Latin alphabet character followed by zero or more additional such characters, numbers, or underscores.

   The identifier of a primitive attribute is, for example, used in the Model Manager search syntax when filtering on attribute values — see Searching on Asset Attributes.

   An identifier will be automatically generated as soon as you starting typing in the **Label** field. Feel free to change the generated identifier as you see fit.
3. Write a description of your attribute in the **Description** field.

   The description is shown as a tooltip to the attribute field label on The Asset Page and in the **Filters** menu on The Home Page.
4. Select the type of widget used to edit the primitive attribute in the **Widget** list. See also Widget Types.
5. For widget types that support multiple value types, select the type of value that can be stored in the primitive attribute in the **Value** list. See also Value Types.
6. For applicable widget types, add allowed values for the primitive attribute to the **Allowed values** table. The allowed values are shown as selectable options when editing the attribute on the **Asset** page, when filtering table rows for composite

attributes on the **Asset** page, and when filtering assets on the **Home** page. At least one allowed value must be added to the table.

a Write a display label for an allowed value in the **Label** field. The label is used for the corresponding selectable option.

b Write the underlying value in the **Value** field.

c Click **Add Allowed Value** to add to the table.

You can rearrange the allowed values using drag and drop — see also Rearranging Table Rows. Click **Sort alphabetically** to enable automatic sorting on label. Click again to disable. The sort order of the allowed values determines the order for the corresponding selectable options on the **Asset** page and the **Home** page.

Click **Delete** to remove an allowed value from the table.

7 Click **Save** to add the new primitive attribute.

Widget Types

The widget type of a primitive attribute determines how the corresponding attribute value is displayed and edited on an asset page. Each widget type supports a subset of all Value Types. The following widget types are available:

TABLE 3-6: ALL AVAILABLE WIDGET TYPES FOR PRIMITIVE ATTRIBUTES.

WIDGET TYPE	SUPPORTED VALUE TYPES	DESCRIPTION
Checkbox	Boolean	A checkbox that can be selected or cleared.
Checkbox list	Keyword array	A set of options in which a subset is chosen by selecting checkboxes.
Combo box	Integer or Keyword	A set of options in which a single option can be selected from a combo box.
Date picker	Date	A calendar in which a date can be selected.
Datetime picker	Timestamp	A widget in which a calendar date and a time can be selected.
File upload	Attachment	A file browser for uploading a file as an attachment.
File version picker	File version	A search widget for finding and selecting a data file version in the database.

WIDGET TYPE	SUPPORTED VALUE TYPES	DESCRIPTION
Hyperlink	Link	Input fields for editing an address and a link text for a web resource.
Input field	Keyword	A text input field for editing a short, name-like, keyword string.
Input fields	Keyword array	Multiple text input fields for editing short, name-like, keyword strings. The keywords can be sorted via drag-and-drop or by enabling a **Sort alphabetically** control.
List box	Keyword array	A set of options in which a subset is chosen by selecting rows from a list.
Model version picker	Model version	A search widget for finding and selecting a model version in the database.
Number	Integer	A number input field for writing an integer.
Radio button	Integer or Keyword	A set of options in which a single option can be selected from a row of radio buttons.
Text area	Text	A text area for writing longer texts that, for example, can contain several paragraphs.
User picker	User	A search widget for finding and selecting a user in the database.
Users picker	User array	A search widget for finding and selecting multiple users in the database.

The **Checkbox list** and **List box** widget types are functionally equivalent, with the main difference that the latter always shows all available options on screen when editing (and thus suitable when the number of options are relatively few). The analogous holds true for the **Combo box** and **Radio button** widget types.

The difference between the **File upload** and the **File version picker** widget types may not be obvious. The former is used to attach files to assets such that these attachments are version controlled together with the asset itself. The latter is used to link to existing

data files that are version controlled as separate items in a repository of the Model Manager database.

Value Types

The value type of a primitive attribute determines how the corresponding attribute data is validated, as well as how that data is made available for searching and filtering. The following value types are available:

TABLE 3-7: ALL AVAILABLE VALUE TYPES FOR PRIMITIVE ATTRIBUTES.

VALUE TYPE	DESCRIPTION
Attachment	A file uploaded as an attachment on an asset. The attribute value is a unique identifier for the attachment.
Boolean	A `true` or `false` value.
Date	A date and time value in the ISO-8601 format with the time set to zero and without any timezone interpretation.
File version	A link to a data file version in the database. The attribute value is a unique identifier for the data file version.
Integer	An integer value.
Keyword	A keyword value. Typically used for short, name-like, search strings, such as an author, a project identifier, or a status value.
Keyword array	Multiple keyword values.
Link	A link to a web resource. The attribute value contains both a link address and a link text.
Model version	A link to a model version in the database. The attribute value is a unique identifier for the model version.
Text	A value that contains text, such as for an abstract or for a summary.
Timestamp	An UTC date and time value in the ISO-8601 format.
User	A link to a user in the database. The attribute value is a unique identifier for the user.
User array	Links to multiple users in the database. The attribute value is an array of unique identifiers for the users.

The **Date** value type is used to store a whole day without any explicit reference to a specific timezone. You can use it, for example, to store an event-like date for which the location may be inferred elsewhere.

THE PRIMITIVE ATTRIBUTE PAGE

The **Primitive Attribute** page contains details on a primitive attribute in the database. The fields are:

- **Identifier**. The unique identifier of the primitive attribute.
- **Label**. The label of the primitive attribute.

  The label is used as the field text of the attribute on The Asset Page.
- **Description**. The description of the primitive attribute.

  The description is shown as a tooltip on the field label of the attribute on The Asset Page.
- **Widget**. The type of widget used when editing the primitive attribute.
- **Value**. The type of value that can be set for the primitive attribute.
- **Allowed values**. The set of allowed values for the primitive attribute. This field is only shown for Widget Types that support a set of allowed values.
- **Workflows**. Links to workflows that are based on the primitive attribute — see Primitive Attribute Workflows.

Click **Edit** to edit the primitive attribute, including possibly changing its widget type and value type or adding allowed values.

Changing a primitive attribute from one widget type to another, while leaving the value type and, when applicable, any allowed values unchanged, is always harmless — assets that use the primitive attribute will simply change how their data is displayed and edited. The data itself is left untouched even when saving a new version of the asset.

Changing the value type of a primitive attribute should be done with care. If you open the **Asset** page for an asset that uses a primitive attribute whose value type has been changed, Model Manager will try to automatically convert the data value to the new type using natural conversion rules. If a conversion is known to be impossible — for example, changing the value type from **Date** to **Attachment** — Model Manager will simply remove the value from being displayed on the page. The value is lost altogether for the new version if the asset is subsequently saved, although it remains on the old version.

The same is true if you remove an allowed value from a primitive attribute that has previously been saved on older assets — the removed value becomes effectively hidden on the **Asset** page for those older assets and it will not be included in the new version if one of those assets is resaved. If you add the allowed value back, however, it reappears on the **Asset** page for the older assets.

Click **Duplicate** to open the **Add Primitive Attribute** page filled in advance with the current primitive attribute's widget type, value type, and allowed values.

Click **Delete Permanently** if you want to permanently delete the primitive attribute in the database. The attribute will be automatically removed from any asset type using it, or from any composite attribute it is a member of. Any assets with data for the primitive attribute will, however, be left untouched until you save a new version of the asset (in which case the attribute data will simply be excluded from the new version).

If you permanently delete a primitive attribute by accident, and do not want to go to the trouble of restoring your database from backup, simply create the attribute anew and add it to the applicable asset types. Any assets that have been updated in the interim will need to have their lost attribute data manually recovered from older versions — see also Asset Version History.

Permanently deleting a primitive attribute will also delete all workflows for the attribute.

Composite Attributes

You can combine primitive attributes as members of a *composite attribute*. This combination can either be done in *simple mode* or in *table mode*. In simple mode, you can store a single composite data entry in the composite attribute, such that the data entry is a combination of one value for each primitive attribute member. In table mode, you can store multiple such data entries in the composite attribute — when displayed and edited, each primitive attribute becomes a table column and each data entry a table row.

See Adding New Types of Data Using Attributes.

When a Model Manager server database is created, two composite attributes in table mode — **Attachments** and **Model Versions** — are automatically created for you. They correspond, respectively, to a list of uploaded file attachments and a list of links to model versions in repositories. The single members of both composite attributes are set as primary members — see Adding Composite Attributes for definition.

The **Composite Attributes** page, opened by clicking **Composite Attributes** in the **Database** navigation sidebar, shows a table with all composite attributes in the database. The table columns are:

- The **Label** column — the label of the composite attribute.
- The **Identifier** column — the unique identifier of the composite attribute.
- The **Description** column — the optional description of the composite attribute.
- The **Mode** column — the mode of the composite attribute, which could be either **Table** or **Simple**.

Click on the label of a composite attribute in the table to show more details for that composite attribute. Click the **Add** button to add a new composite attribute.

ADDING COMPOSITE ATTRIBUTES

To add a new composite attribute:

1. Write a label for the composite attribute in the **Label** field. The label is shown on an asset page as a header to the primitive attribute members.
2. Write an identifier for the composite attribute in the **Identifier** field. This identifier must be unique among the set of all attributes — primitive and composite — and start with a Latin alphabet character followed by zero or more additional such characters, numbers, or underscores.

   The identifier of a composite attribute is used in the Model Manager search syntax when filtering on attribute values — see Searching on Asset Attributes.

   An identifier will be automatically generated as soon as you starting typing in the **Label** field. Feel free to change the generated identifier as you see fit.
3. Write a description of your attribute in the **Description** field.

   The description is shown as a tooltip to the attribute field label on The Asset Page.
4. Select between **Simple** and **Table** mode in the **Mode** list.
5. Add member primitive attributes in the **Add primitive attribute** list. Added primitive attributes are shown in a table under **Member attributes** in the order they will appear on the asset page — top-to-bottom for **Simple** mode and left-to-right for **Table** mode. You can rearrange the member attributes using drag and drop — see also Rearranging Table Rows.

   Select **New primitive attribute** in the **Add primitive attribute** list to both create a new primitive attribute in the database and add it as a member to the composite attribute

— see Adding Primitive Attributes for the available fields. The created primitive attribute may later be reused for other composite attributes as well.

Click **Edit** (the pen icon) next to an attribute added via the **New primitive attribute** option to edit its fields. Click **Delete Permanently** (the cross icon) to permanently delete the attribute from the database.

Select the checkbox of those member attributes that should be set as *primary* members in the **Primary** column. Being a primary member imposes the following constraints:

- A value must always be set for the attribute when saving an asset. This means, for example, that the corresponding table cell for a composite attribute in **Table** mode cannot be left empty.
- Duplicates are not allowed when combining the primary members' values for a single data entry. When there is only a single primary member, this means, for example, that the corresponding table column for a composite attribute in **Table** mode must contain unique values.

Only primitive attributes of value type **Attachment**, **Boolean**, **Date**, **File version**, **Integer**, **Keyword**, **Model version**, and **User** support being set as primary members.

6 Click **Save** to add the new composite attribute.

Any new primitive attribute created via the **Add Composite Attribute** page will remain in the database even if you cancel the composite attribute creation. You can delete it via The Primitive Attribute Page.

Defining a composite attribute in **Table** mode with a single primitive attribute is how you get a data field that can contain a list of values. Examples include a list of supplementary files or a list of linked model versions. The only exceptions are a list of string values or a list of users for which you can use a primitive attribute with the **Keyword array** or **User array** value type, respectively— see Table 3-7.

You typically set a primitive attribute as a primary member of a composite attribute if you want to use its value as a "unique identifier" for the entries in a table. This enables, for example, transitions and permissions for composite attribute workflows to apply to individual table rows when saving an asset — see also Composite Attribute Workflows.

THE COMPOSITE ATTRIBUTE PAGE

The **Composite Attribute** page contains details on a composite attribute in the database. The fields are:

- **Identifier**. The unique identifier of the composite attribute.
- **Label**. The label of the composite attribute.

  The label is used as the field text of the attribute on The Asset Page.
- **Description**. The description of the composite attribute.

  The description is shown as a tooltip on the field label of the attribute on The Asset Page.
- **Mode**. The mode of the composite attribute — either **Table** or **Simple**.
- **Member attributes**. The list of primitive attributes that are members of the composite attribute. The primary members have a checked marker in the **Primary** column.
- **Workflows**. Links to workflows that are based on the composite attribute — see Composite Attribute Workflows.

Click **Edit** to edit the composite attribute, including possibly changing or rearranging its primitive attribute members.

Removing primitive attribute members from a composite attribute should be done with care. If you open and subsequently save a new version of an asset that uses a composite attribute for which a member has been removed, any data for that primitive attribute member will be excluded from the new version. Similarly, if you change from **Table** mode to **Simple** mode, only the data in the first table row will be kept in a new version of an asset.

Click **Duplicate** to open the **Add Composite Attribute** page filled in advance with the current composite attribute's mode and member attributes.

Click **Delete Permanently** if you want to permanently delete the composite attribute in the database. The attribute will be automatically removed from any asset type using it. Any assets with data for the composite attribute will, however, be left untouched until

you save a new version of the asset (in which case the data will simply be excluded from the new version).

If you permanently delete a composite attribute by accident, and do not want to go to the trouble of restoring your database from backup, simply create the attribute anew and add it to the applicable asset types. Any assets that have been updated in the interim will need to have their lost attribute data manually recovered from older versions — see also Asset Version History.

Permanently deleting a composite attribute will also delete all workflows for the attribute.

Primitive Attribute Workflows

You can incorporate basic rules and dynamic behavior for the values of primitive attributes by defining *primitive attribute workflows*. You apply a primitive attribute workflow by adding the workflow to an asset type containing the workflow's primitive attribute.

See also Using Workflows to Set Up Organizational Processes.

The workflow state for the workflow of a primitive attribute is determined by the current value of the attribute on the **Asset** page. The workflow transitions for the workflow describe updates to this value. Such transitions may optionally be represented on the **Asset** page as buttons, with the transition's label used for the button text. Clicking a button saves a new version of the asset with the updated attribute value. You can define any number of workflows for a specific primitive attribute.

The **Primitive Attribute Workflows** page, opened by clicking **Primitive Workflows** in the **Database** navigation sidebar, shows a table with all primitive attribute workflows in the database. The table columns are:

- The **Label** column — the label of the primitive attribute workflow.

- The **Mode** column — the mode for the workflow. Either **Process workflow** or **Free workflow**.
- The **Description** column — the optional description of the primitive attribute workflow.

Click on the label of a primitive attribute workflow in the table to show more details for that primitive attribute workflow. Click the **Add** button to add a new primitive attribute workflow.

See also Examples of Workflows for Primitive Attributes.

ADDING PRIMITIVE ATTRIBUTE WORKFLOWS

To add a new primitive attribute workflow:

1. Select the primitive attribute for the workflow from the **Attribute** list.

   Workflows can be defined for all types of primitive attributes, although some workflow functionality is only available for a subset of types — see also Value Types.

2. Select the mode for the workflow from the **Mode** list — either **Process workflow** or **Free workflow**.

   A process workflow puts restrictions on the possible values for the primitive attribute, as well as how these values can be updated from one value to the next. It may be used, for example, to define various stages of a project- or task-like asset. A free workflow imposes no such restrictions. See also Process Workflows and Free Workflows.

3. Write a label for the primitive attribute workflow in the **Label** field.
4. Click **Continue** to continue to the next page.

   The **Add Primitive Attribute Workflow** page shows the primitive attribute, workflow mode, and workflow label provided on the previous page in the **Attribute**, **Mode**, and **Label** fields, respectively.

5. Write an optional description for the workflow in the **Description** field.

A primitive attribute workflow contains lists of State Conditions, Transitions, and Permissions. A list can be left empty. Click **Save** to add the new primitive attribute workflow once you are finished setting these lists up.

To apply the primitive attribute workflow to your assets, you must also add the workflow as a member workflow to an asset type or to a composite attribute workflow — see Using Workflows with Asset Types and Composite Attribute Workflows.

State Conditions

The *state conditions* for a primitive attribute workflow are conditions posed on the attribute's values. Given an attribute value, a state condition is either satisfied or not satisfied. The simplest type of state condition is a *state value*. A state value is satisfied as a state condition if its corresponding value is either equal to that of the attribute or, for an attribute of value type **Keyword array** or **User array**, is an element of the attribute's array. Given this correspondence between state values and attribute values, a workflow state for a primitive attribute workflow may also be defined as a collection of state values.

A state condition can be formulated as a *state expression* via a filter expression written using a subset of the Model Manager search syntax. The state condition is satisfied if it matches state values as a logical expression. The workflow management system also includes *built-in state conditions* that can be used as dynamic shorthands for state values and state expressions.

A state value, as well as the built-in state conditions **Current timestamp** and **Current user**, may be set as a *default state condition*. The corresponding value of a default state condition is automatically set for a primitive attribute if the current attribute value is empty.

For a primitive attribute with a list of allowed values, the **Add Primitive Attribute Workflow** page is automatically populated with state conditions of type **State value** corresponding to each allowed value. The sort order for the state conditions is the same as defined for the allowed values. Click the **Delete** button (trashcan) to remove an unwanted state condition. Click the up and down arrow buttons to change the sort position of a state condition.

The allowed values of a primitive attribute are only used when adding a new primitive attribute workflow. If you change the allowed values of a primitive attribute and want those changes to be reflected in workflows, you must manually update the workflows' state values.

The sort order for state conditions is only used when displaying these on the **Primitive Attribute Workflow** page.

To add a state condition to the workflow:

1 Select the type of state condition to add in the **Add State Condition** menu. Select from **State value**, **State expression**, or one of the built-in state conditions — see also Table 3-8.

   A **State value** or **State expression** condition is only available for primitive attributes of value type **Boolean**, **Integer**, **Keyword**, **Keyword array**, **User**, and **User array**.

   The available built-in state conditions depend on the attribute type. A **Current timestamp** condition may be used for an attribute of type **Timestamp**; a **Current user** or **Not current user** condition may be used for an attribute of type **User** or **User array**. A built-in state condition can only be added once to the list of state conditions.

2 For a **State value** or **State expression**, write the label of the state condition in the **Label** field. Write an optional description in the **Description** field.

3 For a **State value**, specify the corresponding value for the state condition in the **Value** field. For **Integer**, **Keyword**, and **Keyword array**, write the state value in the input field. For **Boolean**, **User**, and **User array**, select the state value from the list.

   For a **State expression**, specify the expression statement in the **Expression** field. Write the statement as `@<attribute-identifier>:<state-value-expression>` using the identifier of the primitive attribute and an expression involving state values. Remember to escape spaces in state values with backslashes.

4 Select the **Default** checkbox to enable a state condition as a default state condition.

   The **Default** field is only available for a **State value** or a built-in state condition of type **Current user** and **Current timestamp**. Only a single state condition can have a selected checkbox if the value type is **Boolean**, **Integer**, **Keyword**, or **User**. Selecting a second checkbox automatically clears the currently selected checkbox. Multiple checkboxes can be selected if the value type is **Keyword array** or **User array**.

State expressions support a subset of the Model Manager search syntax. You can combine state values with boolean `AND` and `OR` operators, use `NOT` and `ANY` operators, as well as group state values with parentheses. Use of wildcard characters (except `ANY`) or ranges is *not* supported. See also Primitive Attribute Field Expressions.

You introduce a default value for a primitive attribute by defining a workflow for the attribute having a **State value** condition set as a default state condition. The workflow must also be added to relevant asset types.

While primitive attributes of value types other than **Boolean**, **Integer**, **Keyword**, **Keyword array**, **User**, and **User array** cannot have state conditions of type **State value** or **State expression**, they can use built-in state conditions. Such workflows may be useful as member workflows of composite attribute workflows or asset type workflows when defining activities for transitions — see Composite Attribute Workflows and Using Workflows with Asset Types. You can also control which users can update the primitive attributes by adding permissions to their workflows.

TABLE 3-8: ALL AVAILABLE STATE CONDITIONS OF BUILT-IN TYPE FOR A PRIMITIVE ATTRIBUTE WORKFLOW..

STATE CONDITION	DESCRIPTION
Always	A condition that is always satisfied. Combines the **Any state value** and **No state value** built-in state conditions.
Any state value	A condition that is satisfied if the primitive attribute has a non-empty value.
Current timestamp	A condition that is satisfied when equal to the Model Manager server's current timestamp. May be used as a default state condition. May not be used as a state condition for a permission.
Current user	A condition that is satisfied for the user currently saving an asset. May be used as a default state condition.
Never	A condition that is never satisfied.
No state value	A condition that is satisfied if the primitive attribute has an empty value.
Not current user	A condition that is satisfied for all users except the user currently saving an asset.

A state condition of type **Never** is primarily useful to temporarily disable a transition by adding it as a from-state condition or as a to-state condition — see Transitions.

Transitions

The *transitions* for a primitive attribute workflow correspond to changes made to the attribute value when saving an asset. Transitions are expressed as a collection of state conditions that match attribute values before the save, known as the *from-state conditions*, and a collection of state conditions that match attribute values after the save, known as the *to-state conditions*. A transition is *enabled* ("can occur") if the from-state conditions are all satisfied for the asset; an enabled transition has *fired*

("occurred") if the to-state conditions are all satisfied for the saved asset. The state conditions need not all be state values; using a state expression or a built-in state condition for a from- or to-state condition is also possible. The collection of from-state conditions may be empty, in which case the transition is enabled when the value is empty. The analogous holds true for empty to-state conditions.

An *action transition* is a transition that is actively initiated by a user. When the transition is enabled — that is, when its from-state conditions are all satisfied — a corresponding button is displayed on the **Asset** page. The label of the transition is used for the button text; the description is used for the tooltip. Clicking the button removes all state values matched by the from-state conditions and adds all state values identified by the to-state conditions. If more than one action transition is enabled, the remaining ones are available as menu options on the **Asset** page.

A *general transition* is used to automate behavior that should happen when a user updates an asset according to the pattern prescribed by the transition's from- and to-state conditions. There are no restrictions on the types of state conditions that may be used for such transitions.

For a process workflow defined for a primitive attribute with allowed values, the **Add Primitive Attribute Workflow** page is populated with action transitions between the allowed values, with one transition per adjacent pair of values. Click the **Delete** button (trashcan) to remove an unwanted transition. Click the up and down arrow buttons to change the sort position of a transition.

The sort order for transitions is used when displaying these on the **Primitive Attribute Workflow** page. It is also used to determine the order of the button and menu options on the **Asset** page when a transition is enabled: the first enabled transition is shown for the button, the other as menu options via an expander button.

To add a transition to the workflow:

1. Click **Add Transition** to add a general transition. Click **Add Action Transition** to add an action transition.
2. Write the label of the transition in the **Label** field. Write an optional description in the **Description** field.

3 In the **From** field, select state conditions to add as from-state conditions for the transition.

   Any number of state conditions can be added if the primitive attribute has value type **Keyword array** or a **User array**. Otherwise, zero or one condition can be added. For an action transition, only a **State value** condition or a built-in condition of type **Always**, **Any state value**, **Current timestamp**, **Current user**, **Never**, or **No state value** can be added. A **State expression** or **Not current user** cannot be used as a from-state condition for action transitions. A general transition has no restrictions on the supported state condition types.

4 In the **To** field, select state conditions to add as to-state conditions for the transition.

   The requirements for the to-state conditions are similar to the **From** field except that **Always**, **Any state value**, **Never**, and **No state value** are disallowed for action transitions — only **State value**, **Current user**, and **Current timestamp** are allowed.

5 In the **Permissions** table for the transition, add users and groups that are granted permission to perform the transition. Leave the table empty to grant everyone permission. See also Permissions for changing the primitive attribute value itself.

For a **Process** workflow, any update to a primitive attribute value must happen through the firing of an action transition. State values corresponding to the from-state conditions of the action transition are removed, and state values corresponding to the to-state conditions are added.

If multiple transitions fire for the workflow when saving an asset, the overall authorization succeeds if the current user is granted permission to perform *all* general transitions that fired and, if any action transitions fired, *at least one* of those action transitions.

Action transitions for a primitive attribute of value type **Timestamp** do not appear as buttons on the **Asset** page.

	A workflow having multiple from-state and to-state conditions in its transitions can be defined using a **Checkbox list** or a **List box** attribute. It could be used for a process in which an asset can be in "multiple stages" at the same time. An example would be a status-like attribute for which separate tasks need to be finished before the asset can move to the next stage, and these tasks are worked on in parallel.

	An empty collection of from-state conditions corresponds to a transition enabled when the attribute value is empty before saving. An empty collection of to-state conditions corresponds to a transition that fires when the attribute value is empty after saving.

	You can also cause an action transition to fire by explicitly updating the attribute value by editing and then saving an asset. This is useful, for example, if you want to include other changes on the asset when saving the new version.

Permissions

To grant permissions to users and groups that can update the value of the primitive attribute:

1. Add a user or group to the **Permissions** table at the bottom of the page.
2. In the **Permissions with State Conditions** table cell, next to the **Change value** permission type, add state conditions from the **Add State Condition** menu to conditionally grant the permission.

   The **Change value** permission is granted to the user or group if at least one state condition is satisfied by the primitive attribute value before saving the asset. The permission is always granted if the collection of state conditions is empty.

THE PRIMITIVE ATTRIBUTE WORKFLOW PAGE

The **Primitive Attribute Workflow** page contains details on a primitive attribute workflow in the database. The fields are:

- **Attribute**. The primitive attribute for the workflow.
- **Mode**. The mode for the workflow — either **Process workflow** or **Free workflow**.
- **Label**. The label for the workflow.
- **Description**. The optional description for the workflow.

- **State conditions**. The list of state conditions for the workflow. Hidden if the workflow does not have any state conditions.

  The state conditions are listed as three groups: all conditions of type **State value**, all state conditions of type **State expression**, and finally all built-in state conditions.
- **Transitions**. The list of transitions for the workflow, including their from-state condition, to-state conditions, and granted permissions. Hidden if the workflow does not have any transitions.
- **Permissions**. The list of granted permissions for changing the value of the primitive attribute. Hidden if no permissions have been explicitly granted, which means that all users are allowed to change the attribute value.

Click **Edit** to edit the primitive attribute workflow.

Click **Delete Permanently** if you want to permanently delete the primitive attribute workflow in the database. The workflow will be automatically removed from any asset type using it, as well as from any composite attribute workflow it is a member of.

EXAMPLES OF WORKFLOWS FOR PRIMITIVE ATTRIBUTES

This section lists a few applications of workflows for primitive attributes.

Default Option

Given an `Approval` attribute of widget type **Radio button** with allowed values `Approved` and `Not approved`, you may want `Not approved` selected by default when adding a new asset. Define a **Free workflow** for `Approval` with a **State value** condition having **Value** set to `Not approved` and for which the **Default** checkbox is selected.

Assign to Me

Given a `Responsible` attribute of widget type **User picker**, you may want the value to initialize to the user adding a new asset. You also want to enable other users to quickly assign themselves as the responsible. Define a **Free workflow** for `Responsible` with a built-in state condition **Current user** for which the **Default** checkbox is selected. Define an action transition labeled `Assign to Me` with from-state condition **Always** and a to-state condition **Current user**. An **Assign to Me** button will be present next to the `Responsible` attribute on the **Asset** page.

Restricted Attribute

Given an attribute value that only a restricted group of users should be able to update, define a **Free workflow** for the attribute with a **Change value** permission granted to that group.

Status Workflow

Given a `Status` attribute of widget type **Combo box** with allowed values `Suggested`, `Approved`, `Implementing`, `Completed`, and `Canceled`, define a **Process workflow** with one **State value** condition for each allowed value. Also add a built-in **Any state value** condition. Set the `Suggested` state value to be a default state condition so that the workflow initializes in the "suggested stage".

Define five action transitions with the from- and to-state condition mappings: `Suggested` to `Approved`, `Approved` to `Suggested`, `Approved` to `Implementing`, `Implementing` to `Approved`, and `Implementing` to `Completed`. Also add an action transition with the from- and to-state condition mapping `Any state value` to `Canceled`. This workflow describes a simple work process of reversible stages that can be canceled at any stage.

Multiple-Approval Workflow

Given a `Project Proposal` attribute of widget type **Checkbox list** with allowed values `Draft`, `Pending Approval by Engineering`, `Pending Approval by Sales`, `Approved by Engineering`, `Approved by Sales`, and `Approved by Management`, define a **Process workflow** with one **State value** condition for each allowed value. Set `Draft` to be a default state condition so that the workflow initializes in the "draft stage".

Define action transitions that enable a process in which the proposal is reviewed and approved in parallel by the engineering and sales departments, and then when both have approved, the proposal is reviewed and approved by management:

```
Label: Request Approval
From: Draft
To: [Pending Approval by Engineering, Pending Approval by Sales]
Permission: (Everyone)

Label: Approve (Engineering)
From: Pending Approval by Engineering
To: Approved by Engineering
Permission: Engineering Group

Label: Approve (Sales)
From: Pending Approval by Sales
To: Approved by Sales
Permission: Sales Group

Label: Approve (Management)
From: [Approved by Engineering, Approved by Sales]
To: Approved by Management
Permission: Management Group
```

When the attribute value is `Draft`, the Asset page displays the **Approve (Engineering)** and the **Approve (Sales)** action transitions as button and menu options. Either option may be triggered first. After both options have been triggered (irrespective of order), the attribute value is `[Approved by Engineering, Approved by Sales]` and an **Approve (Management)** button is shown. Only users from each respective group are granted permission to perform their transitions for approval.

Composite Attribute Workflows

You add *composite attribute workflows* to set up dependencies between the workflows of primitive attributes when the latter are added as members to a composite attribute. You can, for example, set up general transitions for the composite attribute workflow so that updating the value of one primitive attribute automatically sets the value for another attribute. A composite attribute workflow is also used to enable workflows for primitive attributes *per-table-row* of a composite attribute in **Table** mode. You apply a composite attribute workflow by adding the workflow to an asset type containing the workflow's composite attribute.

See also Using Workflows to Set Up Organizational Processes.

For a composite attribute in **Simple** mode, the workflow state is given by the aggregation of the member attributes' values and workflow transitions involve updating these values. For a composite attribute in **Table** mode, each table row is associated with its own workflow state. Workflow transitions involve adding, updating, or removing a table row. You can define any number of workflows for a specific composite attribute.

The **Composite Attribute Workflows** page, opened by clicking **Composite Workflows** in the **Database** navigation sidebar, shows a table with all composite attribute workflows in the database. The table columns are:

- The **Label** column — the label of the composite attribute workflow.
- The **Description** column — the optional description of the composite attribute workflow.

Click on the label of a composite attribute workflow in the table to show more details for that composite attribute workflow. Click the **Add** button to add a new composite attribute workflow.

See also Examples of Workflows for Composite Attributes.

ADDING COMPOSITE ATTRIBUTE WORKFLOWS

To add a new composite attribute workflow:

1. Select the composite attribute for the workflow from the **Attribute** list.

   Workflows can be defined for composite attributes in both **Simple** mode and **Table** mode.
2. Write a label for the composite attribute workflow in the **Label** field.
3. Click **Continue** to continue to the next page.

   The **Add Composite Attribute Workflow** page shows the composite attribute and the workflow label provided on the previous page in the **Attribute** and **Label** fields, respectively.
4. Write an optional description for the workflow in the **Description** field.

A composite attribute workflow contains lists of Member Workflows, State Conditions, Transitions, Activities, and Permissions. A list can be left empty. Click **Save** to add the new composite attribute workflow once you are finished setting these lists up.

To apply the composite attribute workflow to your assets, you must also add the workflow as a member workflow to an asset type — see Using Workflows with Asset Types.

Select a composite attribute with at least one primary member if you want to set up transitions for your workflow that should fire when a table row is updated — see also Transitions.

Member Workflows

To enable a workflow for a primitive attribute found inside a composite attribute, you must first add it to a composite attribute workflow as a member. To add a member workflow, select a primitive attribute workflow from the **Add workflow** list under the **Member workflows** table. Only workflows for primitive attributes that are members of

the composite attribute are available for selection. Multiple member workflows may be added for the same attribute. You can rearrange the member workflows using drag and drop — see also Rearranging Table Rows. The top-to-bottom sort order reflects the order in which the workflows are evaluated when saving an asset.

You introduce default values for new table rows of a composite attribute in **Table** mode by defining a workflow for the composite attribute that has member workflows with **State value** conditions set as default state conditions. The workflow must also be added to relevant asset types.

State Conditions

State conditions for composite attribute workflows are defined similar to those of Primitive Attribute Workflows. A state condition for a composite attribute in **Table** mode is satisfied if it is satisfied for *at least one* table row. Composite attribute workflows support state conditions of type **State expression** and the built-in state conditions listed in Table 3-9. State values are not supported.

To add a state condition to the workflow:

1. Select the type of state condition to add in the **Add State Condition** menu.
2. For a **State expression**, write its label and optional description in the **Label** and **Description** field, respectively.
3. For a **State expression**, specify the expression statement in the **Expression** field. You write the statement by combining primitive attribute field expressions for the member attributes using boolean `AND` and `OR` operators, `NOT` and `ANY` operators, and by grouping them with parentheses. Unlike a primitive attribute workflow, you do not include the attribute identifier of the composite attribute itself in the expression. As an example,

   `@<member-attribute-identifier-1>:<state-value-expression-1> OR @<member-attribute-identifier-2>:<state-value-expression-2>`

   Only member attributes of value type **Boolean**, **Integer**, **Keyword**, **Keyword array**, **User**, and **User array** can be used in a state expression.

For a composite attribute in **Simple** mode, the state conditions apply on the aggregated values of the member attributes. For a composite attribute in **Table** mode, the state conditions apply on a per-table-row basis.

See Adding Primitive Attribute Workflows for how to write the primitive attribute field expressions themselves.

TABLE 3-9: ALL AVAILABLE STATE CONDITIONS OF BUILT-IN TYPE FOR A COMPOSITE ATTRIBUTE WORKFLOW..

STATE CONDITION	DESCRIPTION
Always	A condition that is always satisfied. Combines the **Any state value** and **No state value** built-in state conditions.
Any state value	For a composite attribute in **Simple** mode, a state condition that is satisfied if at least one member attribute has a non-empty value. For the table row of a composite attribute in **Table** mode, a condition that is satisfied if at least one table cell in the row has a non-empty value.
Never	A condition that is never satisfied.
No state value	For a composite attribute in **Simple** mode, a state condition that is satisfied if all member attributes have an empty value. For the table row of a composite attribute in **Table** mode, a condition that is satisfied if all table cells in the row have an empty value.

A state condition of type **Never** is primarily useful to temporarily disable a transition by adding it as a from-state condition or as a to-state condition. The **No state value** condition may also be used to match updates to a table that involves adding or removing table rows.

Transitions

As for Primitive Attribute Workflows, transitions for a composite attribute workflow correspond to changes made to the attribute values of the composite attribute when saving an asset. For the from-state and to- state conditions, you can use any state condition belonging to the composite attribute workflow itself, as well as any state condition from a member workflow. Action transitions are not supported for a composite attribute workflow.

A transition for a composite attribute workflow is enabled before saving an asset if the following criteria are fulfilled:

- The from-state conditions belonging to the workflow itself are all satisfied.

- The from-state conditions when grouped by member workflow are all satisfied for the primitive attribute value of the corresponding member attribute.
- If a member workflow is present among the to-state conditions, but not the from-state conditions, the member attribute value is empty before the save.

An enabled transition for a composite attribute workflow has fired after saving an asset if the following criteria are fulfilled:

- The to-state conditions belonging to the workflow itself are all satisfied.
- The to-state conditions when grouped by member workflow are all satisfied for the primitive attribute value of the corresponding member attribute.
- If a member workflow is present among the from-state conditions, but not the to-state conditions, the member attribute value is empty after the save.

You can think of the member workflow's from-state and to-state conditions as defining "member transitions" — these member transitions must all fire for the overall transition to fire.

For a composite attribute with primary members, a table row is considered updated when saving an asset if its corresponding values for the primary members are the same before and after the save. For composite attributes without primary members, all rows before the save are considered removed and all rows after the save are considered added.

Moving a table row for a composite attribute with primary members is not considered a change so no transition will fire.

A transition that is enabled when *adding* a table row has from-state conditions that match a "default" table row populated with the values corresponding to the default state conditions of member workflows. If no member workflow has default state conditions, a **No state value** condition from the composite attribute workflow itself is enough as a from-state condition. Otherwise, at least one default state condition from a member workflow must be used. An **Any state value** condition can be used as a to-state condition for the transition. If the composite attribute has a primary member, you can also use the state expression conditions `NOT @<`*`primary-identifier`*`>:ANY` and `@<`*`primary-identifier`*`>:ANY` as an alternative from-state and to-state condition, respectively. Finally, if you define a workflow for the primary member and add it as a member workflow, you can use **No state value** and **Any state value** conditions from the member workflow as from-state and to-state condition, respectively.

A transition will not fire if a user adds an empty table row or a table row that only has default values in its table cells.

A transition that fires when *removing* a table row has to-state conditions that match an empty table row. A **No state value** condition from the composite attribute workflow itself is enough as a to-state condition. An **Any state value** condition can be used as a from-state condition for the transition to be enabled.

A transition will not fire if a user removes an empty table row.

To add a transition to the workflow:

1. Click **Add Transition**.
2. Write the label of the transition in the **Label** field. Write an optional description in the **Description** field.

3 In the **From** and **To** field, select state conditions to add as from-state and to-state conditions for the transition, respectively.

   You can select any number of state conditions belonging to the workflow itself and to its member workflows.

4 In the **Permissions** table for the transition, add users and groups that are granted permission to perform the transition. Leave the table empty to grant everyone permission.

   In the **Permissions with State Conditions** table cell, next to the **Perform transition** permission type, add state conditions from the **Add State Condition** menu to conditionally grant the permission. The permission is granted to the user or group if at least one state condition is satisfied before saving the asset. The permission is always granted if the collection of state conditions is empty.

   See also Permissions for changing the composite attribute value itself.

Activities

You can define *activities* for a transition that should *run* whenever the transition fires.

A *perform transition activity* consists of a collection of action transitions that will be fired by Model Manager server if they were enabled *before* saving the asset. Any number of action transitions belonging to member workflows of the composite attribute workflow can be added to the activity.

A *transition cancellation activity* associated with a transition consists of a collection of state conditions such that, if at least one condition in the collection is satisfied *before* the save, the save of an asset is canceled. A state condition can either belong to the composite attribute workflow itself or to one of its member workflows. The save is always canceled if the collection is empty.

To add an activity to a transition:

1 Select either **Perform transition** or **Transition cancellation**. in the **Add Activity** menu below a transition.

2 Write the label of the activity in the **Label** field. Write an optional description in the **Description** field.

3 For a **Perform transition** activity, select transitions from the **Add Transition** menu.

   For a **Transition cancellation** activity, select state conditions from the **Add State Condition** menu.

Multiple activities of the same type run from top-to-bottom when the transition fires. Click the up and down arrow buttons to change their run order. Click the **Delete** button (trashcan) to remove an activity.

	Transitions are evaluated in the top-to-bottom order defined on the workflow. This determines, for example, the order in which activities from different transitions run when saving an asset.

	The state conditions of all transitions are evaluated *before* any **Perform transition** activities have been run. This prevents any cascading effects where running an activity inadvertently fires unrelated transitions.

	The action transitions performed by **Perform transition** activities are *not* subject to any access control checks. Only access control for the fired transition, that the activities themselves belong to, is checked.

Permissions

As for primitive attribute workflows, you can grant permissions to users and groups that can update the attribute values of the composite attribute overall:

1. Add a user or group to the **Permissions** table at the bottom of the page.
2. In the **Permissions with State Conditions** table cell, next to the **Change values** permission type, add state conditions from the **Add State Condition** menu to conditionally grant the permission. A state condition can either belong to the composite attribute workflow itself or to one of its member workflows.

   The **Change values** permission is granted to the user or group if at least one state condition is satisfied by the composite attribute values before saving the asset. The permission is always granted if the collection of state conditions is empty.

THE COMPOSITE ATTRIBUTE WORKFLOW PAGE

The **Composite Attribute Workflow** page contains details on a composite attribute workflow in the database. The fields are:

- **Attribute**. The composite attribute for the workflow.
- **Label**. The label for the workflow.
- **Description**. The optional description for the workflow.

- **Member workflows**. The primitive attribute workflows that are members of the workflow.
- **State conditions**. The list of state conditions for the workflow, with all **State expression** conditions shown first followed by all built-in state conditions. Hidden if the workflow does not have any state conditions.
- **Transitions**. The list of transitions for the workflow, including their from-state condition, to-state conditions, activities, and granted permissions. Hidden if the workflow does not have any transitions.
- **Permissions**. The list of granted permissions for changing the values of the composite attribute. Hidden if no permissions have been explicitly granted, which means that all users are allowed to change the values.

Click **Edit** to edit the composite attribute workflow.

Click **Delete Permanently** if you want to permanently delete the composite attribute workflow in the database. The workflow will be automatically removed from any asset type using it.

EXAMPLES OF WORKFLOWS FOR COMPOSITE ATTRIBUTES

This section lists a few applications of workflows for composite attributes. See also Examples of Workflows for Primitive Attributes.

Default Table Cell

Given a table cell that you want to initialize with a default value when adding a new table row, define a **Free workflow** for the corresponding primitive attribute with a default state condition. Add the primitive attribute workflow as a member workflow to a workflow of the composite attribute.

Required Table Cell

Given a table column that you want to ensure never has empty table cells, define a `Missing value` **State expression** for a composite attribute workflow with expression statement:

```
NOT @<attribute-identifier>:ANY
```

Define a transition for the workflow that fires when the value is cleared:

```
Label: Clearing value
From: Always
TO: Missing value
```

The from-state condition uses the **Always** built-in state condition. Finally, add a **Transition cancellation** activity without any state conditions to this transition. Its label could be `Required value`.

	You cannot use a **No state value** condition belonging to a member workflow for the required table column as such a transition would not fire when adding a new table row (as the empty table cell would be considered unmodified).

Restricted Table

Given a table that only a restricted group of users should be able to add, update, or remove table rows of, define a workflow for the composite attribute with a **Change values** permission granted to that group.

Modeling Tasks

Add a `Modeling Task` primitive attribute with allowed values `CAD`, `Physics`, `Simulation`, and `Results`. Create a **Process workflow** for the primitive attribute that describes the allowed values as steps for building and solving a model using COMSOL Multiphysics together with other tools. Each allowed value is represented by a **State value** condition. An action transition is defined for each adjacent-pair of allowed values.

Add a composite attribute in **Table** mode with a **Model version picker** attribute as a primary member. Also add the `Modeling Task` primitive attribute as a member. Finally, add a workflow for the composite attribute with the step-workflow as a member workflow.

The composite attribute workflow can be used for a list of models that people from different teams are currently adding features to based on their domain of expertise (CAD geometries, physics setups, solvers, and so on). Different models in the list may be in different stages at any given time. The **Asset** page for an asset type containing the composite attribute and its workflow shows a transition button per-table-row for the next step of each model.

4

Asset Management

Read this chapter to learn how the web-based asset management system included with a Model Manager server installation helps simulation engineers to collaborate on simulation projects with people in your organization who may not have access to the COMSOL Multiphysics software.

In this chapter:

Managing Simulation Projects Using Assets

A Model Manager server includes a web-based asset management system that enables users to access models and data files in Model Manager without requiring a COMSOL Multiphysics installation. From the Model Manager server web interface, users can search for versions using the Model Manager search tool, browse the model tree contents of individual versions, download versions as MPH files, and even upload MPH files to save new versions. From COMSOL Multiphysics, simulation engineers can share results from simulation runs by exporting animations, images, plots, reports, and other output files to the Model Manager server, while other engineers can upload new versions of data files via the web interface — versions which are then immediately available as input files to the simulation engineer from the COMSOL Desktop. The asset management system also enables you to version control various documents, presentations, project notes, slides, and other supplementary files and metadata related to models and data files— all while keeping everything in the same Model Manager server database.

In this section, you will learn what is included with the asset management system by default with a new Model Manager server installation. See the next section to learn how the system can be extended in various ways to suit the needs of your organization.

In this section:

Introduction to Asset Management

Simulation engineers working in the COMSOL Desktop modeling environment often do so in the context of some broader objective. This can, for example, be the development of a new product at a company or a project at a research institute. In these contexts, the simulation work will inevitably involve people in your organization who uses other tools and software, and who may not have access to a COMSOL Multiphysics installation. An experimentalist may need to provide the simulation engineer with new experimental data or a CAD engineer may have updated a CAD geometry. The simulation engineer, in turn, may want to share simulation results in the form of animations, images, and plots. There will also be a need to manage various documents, presentations, notes, slides, and other supplementary files and metadata related to the simulation project.

Managing and sharing all this data can be done using a variety of tools: network file systems together with shared spreadsheets, custom intranet applications, commercial document management systems, project tracking software, or product lifecycle management (PLM) software. Depending on sophistication, these tools may support access control, auditing (who updated what and when), process and workflow management (state transitions), searching and filtering, and version control management.

A Model Manager server installation includes a web-based system for managing simulation projects via *assets*. This *asset management system* forms a layer on top of the existing version control system for models and data files. You can think of an asset as a container for links to your model versions, data file versions, attached supplementary files, and various custom metadata fields. You can use the asset management system as either a standalone tool or as a subcomponent to existing systems within your organization — the existing systems could then, for example, contain web links to assets in the Model Manager server.

Assets are version controlled in their own right — every time you save changes to an asset, a new asset version is stored in the database. This enables you to both track changes made to assets over time, as well as revert such changes if need be. Storing links to your models together with supplementary files and metadata on assets also gives you a searchable archive of past and present projects — all while keeping everything in the same database as your models.

You can search assets on their linked model and file versions, attachments, and any metadata fields — either via full text search or by applying filters. This is true both for

predefined fields common to all assets in a Model Manager server database, as well as for fields that you have added via customization.

The asset management system uses the same user management functionality available for models and data files. Each asset has an associated owner; each asset version has a record of the user that saved it. You grant permissions to users in order to control who has access to an asset. Assets can also be organized into libraries with individual access control settings. Processes and other business rules relevant to your organization may be described by adding workflows for assets.

The asset management system also gives you direct access to the version-controlled models and data files themselves directly from the web interface. You can search for versions using the same search functionality available in the COMSOL Desktop. You can also view the version history of these items, browse their contents and their relationships, and download them as files on your computer. Finally, new versions of models and data files can be saved by uploading files directly from the web interface.

The combination of assets together with direct access to models and data files results in a powerful tool for collaboration. At the start of a simulation project, a new asset is created. By adding links to models, users can quickly navigate both to the models themselves, as well as to all input files that these models depend on, or all output files these models have generated. New input files for a simulation can be uploaded via the web interface; animations, images, plots, reports and other output files can be downloaded. Once the project is completed, the version-controlled asset gives you a valuable historical record of how the project evolved from start to finish.

Overview of the Asset Management Web Pages

The main pages for the asset management part of the Model Manager server web interface consist of:

- The Asset Page. The page for an individual asset. The **Asset** page enables you, for example, to view and edit the attribute fields of the asset, browse the version history to access older versions of the asset, compare two versions to see what changes were made in-between these versions, or navigate to models and data files linked by the asset.
- The Model Page. The page for an individual model. From the **Model** page, you can view and edit metadata fields for the model, including its title and description fields. You can also save a new version of the model by uploading an MPH file from your computer, or download the version by exporting it as an MPH file to your

computer. Built, computed, and plotted data may be optionally excluded in the downloaded file. You can also browse the model tree contents of the model version, as well as browse its relationships to other assets and data files in the system.

- The File Page. The page for an individual data file. Similar to the Model page, the **File** page enables you to view and edit metadata fields for the file, update its file contents by uploading a new file from your computer, as well as download its file contents as a file to your computer. You can also browse its relationships to other assets and models in the system.
- The Home Page. The start page of the asset management system from which you can either search for assets in your asset libraries, or for models and data files in your repositories. The web browser automatically navigates to this page after you log in to the web interface — see Accessing the Web Interface. You can access the **Home** page from other pages by clicking either the COMSOL logo or the **Home** link in the top navigation bar.

For a new Model Manager server database, there are no assets, models, or data files in the database so the **Home** page shows an empty search result.

See Database Administration for the administration of the asset management system.

Adding a New Asset

To add an asset to the asset management system of a new installation of Model Manager server:

1. Click **Add** in the top navigation bar and select **Asset**. This starts a two-page wizard for adding a new asset.
2. On the first page of the wizard, write a title for the new asset in the **Asset title** field. Unlike all other fields on an asset, the title field is mandatory. Click **Continue**.
3. On the second page of the wizard:
   a. Write a description for the asset in the **Description** field.
   b. Click **Choose thumbnail** to browse for a thumbnail image. Write a caption for the image in the **Caption** field.
   c. To add a link to a model version in the database, click **Add Row** under **Model versions**. In the **Search or paste** field for the added table row, write search terms to match models in the database. The search result automatically updates — with

a slight delay — as you type in the input field. You can also add a wildcard asterisk (*) to match, for example, the beginning of titles. Click a model version in the search result to select it for the link.

See Searching for Model Versions to Add to learn more about finding model versions to add as links on the **Asset** page. See Adding a New Model to learn how to save a model to the database by uploading an MPH file.

d To upload a file from your file system, click **Add Row** under **Attachments**. In the added table row, either drag and drop a file to the dotted rectangular area or click the text **Click here to upload** to browse for a file on your computer.

4 Once you have finished adding model versions and file attachments, click **Save** to add the new asset to the database.

You can link to an arbitrary selection of model versions on an asset — both to versions of the same model, as well as versions of different models. The model versions can be selected from any repository and branch you are permitted to browse.

Which version of a model you choose to link to depends on the modeling workflow that best suits your needs. You can, for example, create an asset at the beginning of a new project and link to the first version of a model, or set of models, relevant to the project. As the work progresses and you reach important milestones, you might add or replace certain model versions on the asset. Perhaps when the project is finished, you link to the final version of a model. There is, however, no urgency to continually update links on the asset whenever you save new versions of your models in the database — from the **Model** page of a linked model version, you can quickly navigate to the latest version of the model.

There are no restrictions on the types of supplementary files you can upload on the asset. Examples include notes, word-processing documents, presentations, reports, slides, images, and videos. Data files used as auxiliary data by models — for example, CAD data or interpolation functions — are best version controlled inside the repositories, though, side by side with the models.

If the asset management system has been extended with new asset libraries and asset types, the first page of the wizard lets you select the library and asset type for the new asset via a **Choose library** and **Choose asset type** list, respectively. On the second page, you will likely find many more data fields to edit for the asset depending on what fields

have been added to the asset management system via customization — see also Customizing the Asset Management System.

See also Example: A Project Asset Type for a tutorial that includes steps for adding and editing assets.

The Asset Page

The **Asset** page shows the latest version of an asset by default. The buttons and menus in the toolbar at the top of the page contains:

- **Versions**. View older versions of the asset — see Asset Version History.
- **Compare with Previous**. Compare the current version with the previous one — see Comparing Asset Versions.

  The **Compare with Previous** menu option is hidden for the first version of an asset.
- **Delete**. Delete the asset — see Deleting an Asset.
- **Delete Permanently**. Permanently delete the asset — see Deleting an Asset.
- **Edit**. Edit the asset and save the changes as a new version — see Editing an Asset.
- **Permissions**. Set permissions for the asset — see Granting Permissions to an Asset.
- **Owner**. Set the owner of the asset — see Transfer Ownership of an Asset.
- **Related**. Add links to other assets — Relating Assets.

The asset page has a top section containing the title, description, and thumbnail image of the asset. The upper right corner shows:

- **Identifier**. The unique identifier of the asset.
- **Library**. The asset library that the asset belongs to.

  A new Model Manager server installation comes predefined with a single asset library with the name **Asset library 1**. See also Asset Libraries.
- **Asset Type**. The type of the asset. Defines the available data fields on the asset — see also Asset Types.

  A new Model Manager server installation comes predefined with a single asset type with the plain name **Asset**. The **Asset Type** field is only visible once you have added more types to the asset management system.
- **Owner**. The user currently set as the owner of the asset.

- **Saved**. The point in time when the version was saved.
- **Saved by**. The user that saved the version.

Links to other, related, assets in the asset management system are shown in the **Related Assets** section. Click on a link to open the corresponding **Asset** page for the related asset. The section is hidden if there are no related assets.

Other data fields for the asset are shown in collapsible sections further down on the page. The default asset page has two such sections:

- **Model versions**. A table with model versions in the database that are linked by the asset. Click on the title to open The Model Page for the model version.

  Opening the **Model** page for a model version requires that you have been granted read permissions to the corresponding repository and branch of the model version.
- **Attachments**. A table with supplementary file attachments uploaded on the asset. Click on the filename to download the file to your computer.

See Customizing the Asset Management System to learn how you can extend the **Asset** page by adding other data fields tailored to the needs of your projects. See Searching on Asset Attributes to learn how you can search assets by matching on those data fields.

EDITING AN ASSET

Click **Edit** to make changes to the asset. The **Asset** page is shown with various input fields for updating its data, similar to when adding a new asset. You can change the title, description, and thumbnail image and caption of the asset, as well as add or remove linked model versions and attached files. None of the changes made on the page are stored in the database until you click **Save**, in which case all changes are saved collectively as a new asset version. Click **Cancel** to return the page to its read mode with all data restored as it was before the edit began.

Searching for Model Versions to Add

Click **Add Row** below the **Model versions** table to add a new table row for a linked model version. You search for model versions in the database by writing search terms in the **Search or paste** field of the added table row — you can write both plain search words as well as any number of filter expressions using the Model Manager search syntax. Plain search words will match on the titles, descriptions, tags, and filenames of models. You can also add a wildcard asterisk (`*`) to match, for example, the beginning of titles.

The search result automatically updates — with a slight delay — as you type in the input field. Click a model version in the search result to select it for the link. You can also add a link by pasting the location string identifier for a model version in the **Search or paste** field from your computer's clipboard — see the **Copy Location to Clipboard** button on The Model Page.

Pasting a location string in the **Search or paste** field as obtained via the **Copy Location to Clipboard** button on The Model Page or the **Copy Location** () context menu option in the Model Manager workspace is by far the quickest way to add a model version as a link to an asset.

You can select between two search modes from the menu in the lower right corner of the search result:

- **Latest Versions for Location**. Search the latest versions of models in a branch. Expand the accompanying location selector to select another branch to search in.
- **All Versions in Database**. Search all versions of models in the database. Expand the accompanying location selector to apply one or more repositories and branches as filters. Only versions saved in selected repositories and branches are returned in the search result.

Click **Advanced Search** in the lower left corner of the search result to open the **Add to Asset** dialog. The dialog shows the search result in a paginated table. Unlike the **Search or paste** field inside a table row, you must press Enter or click the magnifying glass in the **Search** field to trigger a new search.

Click on a table row to show more details about that model version. You can also click **View Model Version** to open the **Model** page for the model version in a new browser window. Click an **Add** button to close the dialog and add the corresponding model version as a link on the asset. Click **Cancel** to close the dialog without selecting any model version.

To replace an existing link on the asset, click the **Replace** button next to the title of the model version. To remove the link altogether, click the **Clear** button.

Users will be able to see some of the asset's metadata in the **Assets** table on the **Model** page of the linked model version even if they are not permitted to open the **Asset** page for the asset. See Asset Versions Linking to the Model Version.

See Searching and Filtering in the *Model Manager Reference Manual*.

Uploading Files on Assets as Attachments

You can attach files to the asset by uploading them from your computer. Click the **Add Row** button under the **Attachments** table. In the added table row, either drag and drop a file to the dotted rectangular area or click the text **Click here to upload** to browse for a file. You can also drag and drop multiple files, or select multiple files in the browse dialog. One file will be added to the current table row while additional rows will be appended to the table for the other files.

Examples of files that are suitable as attachments are word documents, presentations, notes, slides, and other supplementary files that are related to your simulation project, but are not directly used for simulations. MPH files and data files are better stored separately as version-controlled items in repositories. Only items stored in repositories may be accessed from the COMSOL Desktop. You can add links to versions of such items on the asset via a **Model version picker** attribute and a **File version picker** attribute.

A **Model version picker** attribute is included on the default **Asset** page for a new Model Manager server installation — see also Searching for Model Versions to Add. You can add a new **File version picker** attribute to the **Asset** page via the administration pages for asset types — see Asset Types.

A confirmation dialog is shown if you try to upload an MPH file to the **Attachments** table. Click **Upload** if you still prefer attaching the MPH file rather than saving it to a repository.

Rearranging Table Rows

You can rearrange the rows in a table by moving the mouse pointer over the crossed arrows icon in a row, pressing and holding down the mouse button to “grab” the row, moving the mouse pointer to “drag” the row to its desired location in the table, and then releasing the mouse button to “drop” the row in its new location.

RELATING ASSETS

You can add links to other assets in the asset management system as a way of *relating* them to the current asset. To edit the current collection of relations, select **Related** in the list opened via the **Edit** button on the **Asset** page to open the **Related Assets** page.

The **Related Assets** page shows the currently related assets in the top table, with columns for the identifier and the title of each asset. Click **Delete** to delete a relation.

To add more relations:

1. In the **Search for assets to add** field, write search and filter expressions and press Enter, or click the magnifying glass, to trigger a new search. The matching assets are shown in the bottom table.

   You can write plain search words and any number of filter expressions using the Model Manager search syntax. Plain search words will match on the title and description of assets. See also Searching Assets on The Home Page.
2. Click the **Add** button (the plus sign) in a table row to add a relation to that asset.

   The selected asset is added to the top table.
3. Repeat the first two steps until you have added all relations.
4. Click **Save** to save the collection of relations.

The related assets appear as links in the **Related Assets** section. Click a link to navigate to the corresponding **Asset** page for that asset.

Users will be able to see the identifier and title of a related asset on the **Asset** page even if they are not granted permission to open its corresponding page.

TRANSFER OWNERSHIP OF AN ASSET

You control who can set permissions on assets via ownerships — every asset has an owner, which is one of the Users in the database. Except for administrators, only the owner can change an asset's permissions. The user that creates an asset is automatically set as its initial owner.

See Owners in the *Model Manager Reference Manual.*

You can transfer the ownership of an asset to another user if you own the asset or are an administrator. Select **Owner** in the list opened via the **Edit** button. In the opened dialog, the **Current owner** field shows the username of the user that currently owns the asset. Under **New Owner**, select the user to transfer ownership to. Click **Save**.

GRANTING PERMISSIONS TO AN ASSET

You can change permissions for an asset if you own the asset or are an administrator. Granting permissions to users and groups enables you to control, for example, who can view the asset or make changes to the asset. All available permissions for an asset are listed in Table 4-1.

See Granting Permissions in the *Model Manager Reference Manual*.

TABLE 4-1: ALL AVAILABLE PERMISSIONS FOR ASSETS.

PERMISSION	DESCRIPTION
Open asset	Allowed to see and open an asset.
Save asset	Allowed to save versions of the asset.
Delete asset	Allowed to delete the asset.
Restore asset	Allowed to restore the asset when deleted.

To set permissions, select **Permissions** in the list opened via the **Edit** button. In the **Permissions** list in the opened dialog:

- Select **None** to not set any permission requirements on the asset.
- Select **Public**, **Protected**, or **Private** to set a predefined permission template — see Predefined Permission Templates in the *Model Manager Reference Manual*.
- Select one of the user-defined permission templates for assets — see Permission Templates.
- Select **Custom** to set up custom permissions for the asset — see Custom Permissions in the *Model Manager Reference Manual*.

Click **Save** to save the new permissions for the asset.

You can also grant permissions to assets based on their asset type — see Asset Types — as well as the asset library they belong to — see Asset Libraries.

ASSET VERSION HISTORY

You can access older versions of an asset via the **Asset Versions** page. Click **Versions** on the **Asset** page to show a table with all versions listed in chronological order, with the latest version shown at the top.

The table columns are:

- The **Title** column — the title of the saved version.
- The **Saved** column — the time when the version was saved.
- The **Saved By** column — the name of the user that saved the version.

Click on a version in the **Title** column to see how the asset looked in that version. To restore the asset from an older version, click **Restore Version**. The older version will be automatically resaved as a new latest version.

You can compare two versions of an asset by selecting their checkboxes in the table and clicking the **Compare** button. The **Asset Comparison** page is opened with a comparison between the versions' data fields — see Comparing Asset Versions.

COMPARING ASSET VERSIONS

You can see the result of comparing two versions of an asset on the **Asset Comparison** page. You can reach this page via one of the following:

- Select **Compare with Previous** in the list opened via the toggle button next to the **Versions** button on the **Asset** page. The **Asset Comparison** page shows the result of comparing the current version on the **Asset** page with the previously saved version.

  The **Compare with Previous** menu option is hidden for the first version of an asset.
- Select **Compare with Latest** in the list opened via the toggle button next to the **Versions** button on the **Asset** page. The **Asset Comparison** page shows the result of comparing the current version on the **Asset** page with the latest saved version.

  The **Compare with Latest** menu option is hidden for the latest version of an asset.
- Select the checkboxes for two versions on the **Asset Versions** page and click **Compare**. The **Asset Comparison** page shows the result of comparing the two versions.

The **Asset Comparison** page shows the before and after values of all fields that have changed between the two versions, with the before value indicated by a minus sign and a red color highlighting, and the after value by a plus sign and a green color highlighting. Some field types also use strikethroughs and underlines to indicate before and after, respectively. Fields that have the same value in both versions are hidden on the page.

Select the **Show detailed text changes** checkbox for text fields to show individual changes made to words and sentences inside paragraphs — removed words have red strikethroughs and added words have green underlines. Entire paragraphs deemed either completely removed or completely added are indicated by a left-aligned red or

green horizontal line with a minus or plus sign, respectively. Clear the checkbox to see the before and after texts in their entirety.

Select the **Only show modified rows** checkbox above table-valued fields to hide those table rows that are the same in the two versions — both when it comes to the row's position and the row's cell values. Clear the checkbox to see all table rows.

DELETING AN ASSET

Click **Delete** on the **Asset** page to delete the asset. The asset will still be present in the database but no longer show up in search results by default. To restore a previously deleted asset, click **Restore**.

Select **Delete Permanently** in the list opened via the toggle button next to the **Delete** button to permanently delete the asset and all its versions.

Permanently deleting an asset cannot be undone unless you restore the Model Manager server database from a backup.

Adding a New Model

You can add a new model to a repository in the Model Manager server database by uploading an MPH file via the web interface. Any MPH file saved in COMSOL Multiphysics version 5.3 or later may be uploaded in this way.

A Model Manager server preserves the format of the MPH file as originally saved in COMSOL Multiphysics when uploading the MPH file via the web interface. The same is also true when downloading a model version as an MPH file. This enables you to add models saved in older versions of COMSOL Multiphysics to a repository in the Model Manager server database so that these models may also be opened in those older COMSOL Multiphysics versions in the future.

While MPH files saved in COMSOL Multiphysics version 5.2a or older cannot be stored in a repository, they can be uploaded as attached files on an asset. Such file attachments, however, cannot be opened directly from the database in COMSOL Multiphysics.

To add a new model from an MPH file:

1 Click **Add** in the top navigation bar and select **Model**. This starts a two-page wizard for adding a new model.

2 On the first page of the wizard, under **Upload Model**, either drag and drop an MPH file to the dotted rectangular area or click the text **Click here to upload** to browse for an MPH file on your computer.

3 On the second page of the wizard:

   a Select the repository and branch that the model will be added to in the location selector in the upper right corner.

   b Model Manager automatically extracts the title, description, and filename from the uploaded MPH file. Either keep these values unchanged or update with new values in the **Title**, **Description**, and **Filename** fields.

   c Optionally assign tags to the new model in the **Tags** list.

4 Click **Save** to open the **Save Model** dialog. Write an optional commit comment in the **Comments** field. Click **Save** to save the model to the database.

The **Model** page is automatically opened with the saved version of the new model.

Model Manager will automatically detect if the MPH file reference any model versions or file versions in the database as auxiliary data. You can view these referenced versions in the **References** table on the **Model** page for the saved model. Unlike saving from the COMSOL Desktop, however, discovery of a reference to a version that is missing from the database will *not* cancel the save.

The Model Page

The **Model** page shows the latest version of a model with respect to a branch in a repository. The toolbar at the top of the page contains the buttons and menu options:

- **Versions**. View older versions of the model — see Model Version History.
- **Copy Location to Clipboard**. Copy a location string identifier to your computer's clipboard. This identifier can be used to quickly add a link to the model version when editing an asset on the **Asset** page — see Searching for Model Versions to Add.
- **Open in COMSOL Desktop**. Open the model version in a COMSOL Multiphysics program session — see Opening Model Versions in the COMSOL Desktop.

- **Run in COMSOL Desktop**. Run the model version in a COMSOL Multiphysics program session as an application — see Running Application Versions in the COMSOL Desktop. Only available for model versions of type **Application**.
- **Download**. Download the model version as an MPH file — see Downloading a Model Version.
- **Customized Download**. Download the model version as an MPH file using custom download settings.
- **Edit**. Edit the model and save the changes as a new version — see Editing a Model.

The model page shows the title, description, assigned tags, filename, model tree, and thumbnail image of the model. A placeholder image is shown for models lacking a thumbnail image. The list of metadata fields on the left are:

- **Location**. The repository and branch that the model version is saved in.
- **Owner**. The user currently set as the owner of the model.
- **Saved**. The point in time when the version was saved.
- **Saved by**. The user that saved the version.
- **Saved in**. The COMSOL Multiphysics version that the model was saved from.
- **Product requirements**. The add-on products required to build the model in COMSOL Multiphysics. At least one product from each item in the bulleted list must be available for license checkout.

The model tree of the model version is shown in the **Contents** field. Click on a tree node to expand or collapse its children.

The model tree is only available for a model saved in COMSOL Multiphysics version 6.2 or later if the model was saved to the database by uploading an MPH file.

Models in the *Model Manager Reference Manual*.

MODEL VERSION REFERENCES

You can see references between the current model version and other versions in the **References** table on the **Model** page. Select **Show Referenced Versions** to list all versions *referenced* by the current model version; select **Show Referencing Versions** to list all versions *referencing* the current model version.

The references are categorized by the following types:

- **Input file**. A data file version used as input by a model version. Examples include CAD data, interpolation data, and mesh data.
- **Output file**. A data file version generated as output from a model version. Examples include animations, images, plots, and reports.
- **Geometry part**. A geometry part loaded from another model version.

You can filter the table by selecting from these types in the **Reference Type** list.

The table columns are:

- The type column — an icon used to represent of the type of the version.
- The **Title** column — the title of the version.
- The **Reference Type** column — the type of version reference.
- The **Saved** column — the point in time when the version was saved.
- The **Saved by** column — the user that saved the version.
- The **Repository** column — the repository that the version was saved in.
- The **Branch** column — the branch that the version was saved in.
- The **Comments** column — the commit comment written when the version was saved.

Click on the title of a version in the table to open its **Model** or **File** page.

The versions shown in the **References** table — with **Show Referenced Versions** selected — are so-called *auxiliary data* used by the model as either an input source or an output target.

The References Window in the *Model Manager Reference Manual.*

ASSET VERSIONS LINKING TO THE MODEL VERSION

You can see asset versions that link to the current model version in the **Assets** table on the **Model** page. If more than one version of an asset link to the model version, only the most recently-saved asset version is included in the table.

The table columns are:

- The **Identifier** column — the unique identifier of the asset.

- The **Title** column — the title of the asset version.
- The **Saved** column — the point in time when the asset version was saved.
- The **Saved by** column — the user that saved the asset version.
- The **Owner** column — the user that owns the asset.

Click on the title of an asset version in the table to open its **Asset** page.

Users that are granted permission to access the **Model** page of a model version will be able to see the information in the **Assets** table even if they are not permitted to open the corresponding **Asset** page.

EDITING A MODEL

Click **Edit** to save a new version of the model. The **Model** page is shown with input fields for updating the model's title, description, assigned tags, and filename. A modified input field can be returned to its original value by clicking the **Reset** button next to the field.

You can update the model from an MPH file. In **Update from** field, either drag and drop an MPH file to the dotted rectangular area or click the text **Click here to upload** to browse for an MPH file on your computer. Click **Clear** next to the filename of the uploaded MPH file to clear the field. Click **Upload New File** to replace with another MPH file. Only the title, description, assigned tags, and filename of the model will be updated if you leave the **Update from** field empty when saving.

Unless you have already modified the corresponding field on the page, Model Manager server will automatically update the values for the **Title**, **Description**, and **Filename** fields from values found in the uploaded MPH file. Click the **Reset** button next to a field if you would rather keep the value of the saved version.

None of the changes made on the page are stored in the database until you click **Save**, in which case all changes are saved collectively as a new model version. Click **Cancel** to return the page to its read mode with all data restored as it was before the edit began.

To save your changes as a new model version:

1. Click the **Save** button at the bottom of the page.
2. Write a commit comment in the opened **Save Model** dialog.

3 Click **Save** in the dialog.

	Any version conflicts detected between the latest model version and a potential model version identified by the uploaded MPH file will be ignored when saving. Such a conflict may occur, for example, if the MPH file was previously obtained by downloading an older version of the model from the database.

	Identical to Adding a New Model, a Model Manager server will preserve the format of the MPH file as originally saved in COMSOL Multiphysics when updating from an uploaded MPH file. You can update from MPH files saved in COMSOL Multiphysics version 5.3 or later.

	Model Settings in the *Model Manager Reference Manual*.

MODEL VERSION HISTORY

You can view the model's version history with respect to the latest version on a particular branch on the **Model Versions** page. Click **Versions** on the **Model** page to show a table with versions saved on the branch sorted chronologically with most recent first. To view the version history with respect to another branch, select it from the location selector in the upper right corner.

The table columns are:

- The type column — an icon used to represent of the type of the version.
- The **Title** column — the title of the version.
- The **Saved** column — the point in time when the version was saved.
- The **Saved by** column — the user that saved the version.
- The **Repository** column — the repository that the version belongs to.
- The **Branch** column — the branch that the version belongs to.
- The **Comments** column — the commit comment written when the version was saved.

Click on a title in the **Title** column to open the **Model** page for that specific version.

The Versions Window in the *Model Manager Reference Manual*.

DOWNLOADING A MODEL VERSION

Click the **Download** button on the **Model** page to download the model version as an MPH file to your computer. For models with large solution data, the download may take some to begin as Model Manager server first needs to build the MPH file on the server computer.

The Model Manager server preserves the format of the model version as originally saved in COMSOL Multiphysics in the exported MPH file.

Built, computed, and plotted data is included in the downloaded MPH file based on the selected value in the **On file** list in the **Save** section of the **Settings** window for the root node of the model tree. To override this, click the expander button next to the **Download** button and select **Customized Download**. In the **Customized Download** dialog, under **Computed data**, select among the options:

- **Include or exclude based on application settings**. Same behavior as when clicking the **Download** button. Default selected.
- **Include**. Download with built, computed, and plotted data included.
- **Exclude**. Download with built, computed, and plotted data excluded.

Click **Download** in the dialog to download the model version using the selected option for built, computed, and plotted data.

OPENING MODEL VERSIONS IN THE COMSOL DESKTOP

Users with a COMSOL Multiphysics installation on Windows® can open a model version in the COMSOL Desktop by clicking the **Open in COMSOL Desktop** button on the **Model** page. The **Opening Model in the COMSOL Desktop** dialog is shown in the web interface while an attempt is made to open the model version in a suitable COMSOL Multiphysics program session. If no program session is currently running, or if models are being solved in all available program sessions, a new program session will be

launched. Otherwise, a dialog is shown in one of the program sessions asking how to open the model version:

- Click **Open in This Window** to open in the current program session.
- Click **Open in New Window** to launch a new program session.
- Click **Cancel** to not open at all.

	Opening a model version via **Open in COMSOL Desktop** is only supported for COMSOL Multiphysics version 6.4 or later on the Windows® operating system. You must have also selected the **Associate links in the Model Manager server web interface with this installation** checkbox in the **Options** step of the COMSOL Multiphysics installer.

	You will get an error message in the COMSOL Desktop if there is no database configuration for the Model Manager server database in the Model Manager workspace. Select **File>Open From>Add Database>Connect to Server Database** and fill in the connection details for the Model Manager server. See also Connecting to a Server Database in the *Model Manager Reference Manual*. You can proceed by clicking **Open in COMSOL Desktop** once more from the web interface.

Running Application Versions in the COMSOL Desktop

For model versions of type **Application**, you can also click the expander button next to the **Open in COMSOL Desktop** button and select **Run in COMSOL Desktop**. The **Running Application in the COMSOL Desktop** dialog is shown while an attempt is made to run the model version as an application in COMSOL Multiphysics.

Using Copy Location to Clipboard to Open the Model Version

If you have an older installation of COMSOL Multiphysics (6.0–6.3), or if you are running on Linux® or macOS, you can use the location string identifier of the model version to open it:

1. Click the **Copy Location to Clipboard** button on the **Model** page.

   The location string identifier for the model version has now been copied to the computer's clipboard.
2. Launch the COMSOL Multiphysics software if not already running. Select **File** > **Open From** in the COMSOL Desktop.

3 Select the **Clipboard** menu option in the **Open** window. The model version is shown as a single option in the list on the right. Click **Open**.

You may be asked to fill in your username and password for the Model Manager server.

Copying to clipboard via the **Copy Location to Clipboard** button is only available when accessing the Model Manager server web interface via `localhost` or through a secure connection using HTTPS.

Adding A New File

You can add a new data file to a repository in the Model Manager server database by uploading it via the web interface. This is useful, for example, when you want to distribute a new input file to simulation engineers working in the COMSOL Multiphysics software.

To save a new file:

1 Click **Add** in the top navigation bar and select **File**.

2 Select the repository and branch that the file will be added to in the location selector in the upper right corner.

3 Add one or more file resources for the new file version to the **Contents** table by uploading them from your computer — see also Editing a File.

The filename of the first file resource added to the table will be automatically used in the **Title** field.

4 Either keep the suggested title in the **Title** field or write a custom title.

5 Optionally write a description of the file in the **Description** field. You can also select tags to assign the file in the **Tags** list.

6 Write a commit comment in the opened **Save File** dialog.

7 Click **Save**.

The **File** page is automatically opened with the saved version of the new file.

An example of a file version containing multiple file resources — also known as a *fileset* — is CAD data consisting of a main CAD assembly file and one or more external component files that the assembly references.

The File Page

The **File** page shows the latest version of a file with respect to a branch in a repository. The toolbar at the top of the page contains the buttons:

- **Edit**. Edit the file and save the changes as a new version — see Editing a File.
- **Versions**. View older versions of the file — see File Version History.
- **Download**. Download the file contents to your computer's file system. File contents for a fileset is downloaded as a compressed archive file.

The file page shows the title, description, and assigned tags of the file. The list of metadata fields on the left are:

- **Location**. The repository and branch that the file version is saved in.
- **Owner**. The user currently set as the owner of the file.
- **Saved**. The point in time when the version was saved.
- **Saved by**. The user that saved the version.
- **File size**. The total size of the file version in bytes.

Files in the *Model Manager Reference Manual.*

FILE VERSION CONTENTS

Binary and text data for the file version are shown as *file resources* in the **Contents** table. A file version containing a single file resource is referred to as a plain *file*. Examples include an interpolation text file or an animation file. A file version containing multiple file resources is referred to as a *fileset*. Examples include a CAD assembly with external component files, an image sequence, or an HTML report.

The table columns are:

- The **Filename** column — the filename of the file resource.
- The **File Size** column — the size of the file resource in bytes.
- The **Last Modified** column — the date and time when the contents of the file resource was last modified.

File resources in a file version may be organized into a hierarchy of directories. A directory is shown with a triangle symbol next to its name in the **Filename** column. Click the triangle to list the file resources and, possibly, subdirectories found inside that

directory. File resources found on the top level in the table are thought of as belonging to an implicit *root directory*.

FILE VERSION REFERENCES

You can see references between the current file version and other versions in the **References** table on the **File** page similar to how such references are shown on the **Model** page — see also Model Version References.

ASSET VERSIONS LINKING TO THE FILE VERSION

You can see asset versions that link to the current file version in the **Assets** table on the **File** page similar to how such links are shown on the **Model** page — see also Asset Versions Linking to the Model Version.

EDITING A FILE

Click **Edit** to edit the file. The **File** page is shown with input fields for updating its title, description, and assigned tags. You can also add, replace, or remove file resources for the file.

The **Contents** table is shown with an additional top table row representing the root directory when the **File** page is in edit mode. Click the **Add File** button to upload a file resource to the root directory. In the opened dialog, either drag and drop a file to the dotted rectangular area or click **Click here to upload** to browse for a file on your computer.

Click the **Replace** button next to an existing file resource to replace its binary or text data with that of a file on your computer. The existing file resource will keep its current filename.

Click **Delete** to remove a file resource from the table.

You can also create a new folder under the root directory. Click **Add Folder** to open the **Add Folder** dialog. Write the name of the folder in the **Title** field. Click **Add**. With the folder added to the table, you can continue by adding new files and other folders under that folder.

None of the changes made on the page are stored in the database until you click **Save**, in which case all changes are saved collectively as a new file version. Click **Cancel** to return the page to its read mode with all data restored as it was before the edit began.

To save a new file version:

1 Click the **Save** button at the bottom of the page.

2 Write a commit comment in the opened **Save File** dialog.

3 Click **Save** in the dialog.

A directory hierarchy of a file version is only stored implicitly in the database via the relative paths, with respect to the implicit root directory, of the file resources. No information is stored about the directories themselves. As a consequence, empty folders are ignored by Model Manager when saving.

A fileset with a deeply nested hierarchy is expected to be rare. The prime example of a fileset with a hierarchy is an HTML report with a main HTML file at the top and associated images inside a folder.

File Settings in the *Model Manager Reference Manual*.

FILE VERSION HISTORY

You can view the file's version history with respect to the latest version on a particular branch on the **File Versions** page similar to how this history is shown for models — see also Model Version History.

The Home Page

You reach the **Home** page of the asset management system by clicking on the COMSOL logo or the **Home** link in the top navigation bar. The **Home** page is the default landing page when you log in to the Model Manager server web interface. Click **Assets** in the upper left corner to search for assets in the database. Click **Models and Files** to search for models and data files in repositories.

- Searching Assets
- Searching Models and Files

SEARCHING ASSETS

You can search for the latest versions of assets from the **Home** page of the asset management system. Write search and filter expressions in the **Search** field and press

Enter or click the magnifying glass to trigger a new search. You can write plain search words and any number of filter expressions using the Model Manager search syntax — see Searching on Asset Attributes. Plain search words will match on the identifier, title, and description of assets.

Matching assets are shown in a paginated table below the **Search** field. Select **Rank**, **Title**, or **Last Modified** to change the field being sorted on — sorting by rank means that those assets that best match the search expression are shown first. Toggle between ascending and descending order by clicking the vertical arrow button next to the sort field list. Select **Table View** or **Compact View** to show or hide the assets' thumbnail images in the table.

The default table columns are:

- The **Title** column — the current title of the asset.
- The **Description** column — the current description of the asset.
- The **Last Modified** column — the point in time when the asset was last modified.

You can change the number of assets shown on a search result page by selecting from the menu next to the sort field. Use the arrow buttons to navigate between search result pages.

Place the mouse cursor over a table cell to see a tooltip with its contents. This helps, for example, when reading long description texts in **Compact View**.

Click on a row to open the corresponding **Asset** page for an asset in the table.

The Model Manager server hides asset versions whose **Asset** page you are not permitted to open from the search result — see also Table 4-1.

Asset Filters

You can apply asset filters from the **Filters** menu — click the vertical expander bar on the left side of the **Home** page if the menu is not already visible. The available filter options for a new Model Manager server installation are:

- **Identifier**. The unique identifier of an asset.
- **Last Modified**. A date range for when the latest version of an asset was saved. Leave **From** or **To** empty to not specify a lower or upper bound for the date.
- **Last Modified By**. The user that saved the latest version of an asset. A multiple selection of users is combined with `OR`-logic when filtering.

- **Owner**. The user that owns an asset. A multiple selection of users is combined with `OR`-logic when filtering.
- **Deleted**. Optionally show assets that have been deleted. The default is to hide such assets.
- **Attachments**. The filename, file type, last modified date, and file size of uploaded file attachments.
- **Model versions**. The title, item version type, item save type, saved date, saved by-user, and owner of linked model versions.

The list of available filter options will grow as you extend the asset management system via customization — see Customizing the Asset Management System.

The search result is automatically updated as you specify the value of each filter. Click **Clear Filters** in the **Filters** menu to clear all specified filter values.

Currently active filters are shown in the **Active filters** list under the **Search** field. Click the cross button to remove a filter. Click **Clear All Filters** to remove all filters.

Asset filters specified via input fields support being written as the field value part of a field expression — see also Asset Search Syntax. You can, for example, surround words with quote characters (`"..."`) to match them as phrases or use an asterisk symbol (`*`) for wildcard matching. A warning banner is shown below the input field if there is a problem with the search syntax — see Syntax Errors. The corresponding filter in the **Active filters** list also shows a warning icon. Click the warning icon to focus the problematic filter input.

Click the eye icon next to a filter option to show or hide the corresponding table column in the search result. You can also show or hide the **Title**, **Description**, **Thumbnail**, **Library**, and **Asset Type** fields as table columns.

SEARCHING MODELS AND FILES

Click **Models and Files** on the **Home** page of the asset management system to search for versions of *items* — that is, models and data files — in the Model Manager server database. Write search and filter expressions in the **Search** field and press Enter or click the magnifying glass to trigger a new search. You can write plain search words and any number of filter expressions using the Model Manager search syntax.

You can select between two search modes from the list in the upper right corner:

- **Latest Versions for Location**. Search the latest versions of items in a branch. Expand the accompanying location selector to select another branch to search in.
- **All Versions in Database**. Search all versions of items in the database. Expand the accompanying location selector to apply one or more repositories and branches as filters. Only versions saved in selected repositories and branches are returned in the search result.

Searching Versions in the *Model Manager Reference Manual*.

Matching versions are shown in a paginated table below the **Search** field. Select **Rank**, **Title**, **Saved**, **Saved In**, **Size**, or **Computed Data** to change the field being sorted on — sorting by rank means that those versions that best match the search expression are shown first. Toggle between ascending and descending order by clicking the vertical arrow button next to the sort field list.

Sorting Search Results in the *Model Manager Reference Manual*.

The default table columns are:

- The type column — an icon used to represent of the type of the version.
- The **Title** column — the title of the version.
- The **Tags** column — the tags assigned to the item that the version belongs to.

  The **Tags** column is only shown for the **Latest Versions for Location** search mode.
- The **Saved** column — the point in time when the version was saved.
- The **Saved by** column — the user that saved the version.
- The **Owner** column — the user that owns the item that the version belongs to.

You can change the number of items shown on a search result page by selecting from the menu next to the sort field. Use the arrow buttons to navigate between search result pages.

Place the mouse cursor over a table cell to see a tooltip with its contents.

Click on a row to open the corresponding **Model** or **File** page for a model or file version in the table.

The Model Manager server hides model and file versions whose **Model** and **File** pages you are not permitted to open from the search result.

Item Filters

You can filter the search result by applying item filters from the **Filters** menu — see also Asset Filters. The available filter options are:

- **Title**. The title of the version.
- **Description**. The description of the version.
- **Tags**. The tags assigned to the item that the version belongs to.

  The **Tags** filter is only available for the **Latest Versions for Location** search mode.
- **Item Version Type**. The type of the item version.
- **Item Save Type**. The save type of the item that version belongs to.
- **Saved**. A date range for when the version was saved. Leave **From** or **To** empty to not specify a lower or upper bound for the date.
- **Saved By**. The user that saved the version. A multiple selection of users is combined with `OR`-logic when filtering.
- **Saved In**. A range for the COMSOL Multiphysics version number that a model version was saved in. Leave **From** or **To** empty to not specify a lower or upper bound for the version number.

  Examples of supported formats: `5.3a`, `6.1`, `5.3.1.348`, and `6.1.0.346`.
- **Commit Comment**. The commit comment written when the version was saved.
- **Filename**. The filename of a model version and the filenames of the file resources of a file version.
- **File Type**. The file type extensions of the file resources of a file version.
- **Size**. The estimated disk space usage of a version when stored on the file system.
- **Computed Data**. The estimated disk space usage of the built, computed, and plotted data of a model version when stored on the file system.
- **Owner**. The user that owns the item that the version belongs to. A multiple selection of users is combined with `OR`-logic when filtering.

The search result is automatically updated as you specify the value of each filter. Click **Clear Filters** in the **Filters** menu to clear all specified filter values.

Currently active filters are shown in the **Active filters** list under the **Search** field. Click the cross button to remove a filter. Click **Clear All Filters** to remove all filters.

Item filters specified via input fields support being written as the field value part of a field expression — see also The Model Manager Search Syntax in the *Model Manager Reference Manual*. You can, for example, surround words with quote characters (`"..."`) to match them as phrases or use an asterisk symbol (`*`) for wildcard matching. A warning banner is shown below the input field if there is a problem with the search syntax — see Syntax Errors. The corresponding filter in the **Active filters** list also shows a warning icon. Click the warning icon to focus the problematic filter input.

Click the eye icon next to a filter option to show or hide the corresponding table column in the search result. You can also show or hide the **Repository** and **Branch** fields as table columns.

Item Filters in the *Model Manager Reference Manual*.

Customizing the Asset Management System

As you may have noticed, the default **Asset** page provided with a new Model Manager server database is somewhat rigid when it comes to adding data. You can, for example, add text to the **Description** field, add links to model versions in the **Model versions** section, and upload supplementary files in the **Attachments** section. Other than that, there is not much flexibility.

In this section, you will learn how the asset management system can be customized so that more types of data can be added to an asset — either by extending the asset page predefined for a Model Manager server database or by defining new types of assets. You will also learn how assets can be organized into asset libraries to control access, and how internal processes and business rules in your organization can be described by workflows.

- Adding New Types of Data Using Attributes
- Adding New Types of Assets
- Using Workflows to Set Up Organizational Processes
- Controlling Access to Assets Using Libraries

See also Example: A Project Asset Type for an introductory tutorial that includes customization of the asset management system.

Adding New Types of Data Using Attributes

You can extend the available fields on an asset page by defining new *primitive attributes*. A primitive attribute is a data field that can hold a specific *value type*. Examples include (see Table 3-7 for the complete list):

- A **Keyword** value. A project identifier, author, search label, project status, or some other name-like metadata.
- A **Text** value. An abstract, project notes, or some other multiple-paragraph text.
- A **Date** value. A publication date, review date, or some other date.
- A **Boolean** value. An approved flag or some other true or false choice.

- A **Link** value. A link to a web page or some other web resource.
- An **Attachment**. Documentation files, presentations, slides, videos, or any other type of supplementary file.
- A **Model version**. A link to a version-controlled model stored in a repository in the Model Manager server database.
- A **File version**. A link to a version-controlled data file stored in a repository in the Model Manager server database.
- A **User**. A link to a user in the Model Manager server database.

The new attributes are added to sections on the **Asset** page, either to one of the existing sections or by defining new sections for the page. Each primitive attribute has an *attribute label* which is shown next to the attribute value when viewing or editing the asset. Both the sections and the primitive attributes inside each section can be rearranged as you see fit.

You define how the values of a primitive attribute are edited by selecting a *widget type*. Examples include (see Table 3-6 for the complete list):

- An **Input field** for writing a keyword value in a single-line input field.
- A **Text area** for writing text, possibly split into several paragraphs, in a rectangular text area.
- A **Date picker** for selecting a date from a calendar.
- A **Checkbox** for selecting a Boolean true or false value.
- A **Combo box** or **Radio button** for selecting a single value among a set of allowed values. This could be a status flag or some other predetermined set of options.
- A **Checkbox list** or **List box** for selecting multiple values among a set of allowed values. This could be product labels or some other predetermined set of options.
- A **Hyperlink** for specifying a web page or some other web resource.
- A **File upload** for browsing and selecting a file on the file system to be uploaded as an attachment.
- A **Model version picker** for searching and selecting a version-controlled model in a repository to add as a link on an asset.
- A **File version picker** for searching and selecting a version-controlled data file in a repository to add as a link on an asset.
- A **User picker** for searching and selecting a user in the Model Manager server database.

Given a particular widget type for a primitive attribute, only a subset of all value types are available to select from. A **Date picker**, for example, can only store a **Date** value and a **File upload** can only store an **Attachment**.

The primitive attributes also show up in the Asset Filters menu on the **Home** page. You can apply these attributes as search filters via filter widgets analogous to the ones used when editing. The attributes can also be shown as columns in the search result table.

A Model Manager server database comes predefined with two primitive attributes, **Model version** and **Attachment**, whose attribute labels have been set to be identical to their corresponding value type. They appear as table columns in the two sections, **Model versions** and **Attachments**, on the predefined **Asset** page.

See Primitive Attributes to learn how administrators can define new primitive attributes via the **Administration > Database** area. See Asset Types to learn how to add these primitive attributes to sections on the **Asset** page.

See Searching on Asset Attributes to learn how you can search and filter on your custom data fields.

PRIMITIVE ATTRIBUTES OF VERSION TYPE

Primitive attributes of value type **Model version** or **File version** are special in that they only store a link to a fixed version of a model or data file in a repository. These attributes are displayed on the **Asset** page with the version's title, the point in time when the version was saved, the user that saved the version, and the item version type of the version. You can click on the title to open the corresponding **Model** or **File** page for the version.

You edit these primitive attributes by searching and selecting a model or data file version in the database. See Searching for Model Versions to Add for editing an attribute of **Model version** value type. Editing an attribute of **File version** value type works identically.

GROUPING PRIMITIVE ATTRIBUTES INTO COMPOSITE ATTRIBUTES

You can group one or more primitive attributes together by adding a *composite attribute*. The composite attribute can be defined using one of two *modes*:

- A **Simple** mode. This is useful if a collection of primitive attributes "logically" belong together. The primitive attributes appear under a common header given by the label of the composite attribute. An example of such a composite attribute is a journal reference consisting of primitive attributes for the author, publication, URL, and year.
- A **Table** mode. This is useful if the values of a collection of primitive attributes belong together *and* you want to add multiple data entries for the primitive attributes when editing assets. The primitive attributes appear as columns in a table. Each data entry is then a row in the table. An example of such a composite attribute is a table of journal references with the listed primitive attributes as columns.

Composite attributes also appear in the Asset Filters menu on the **Home** page with their primitive attribute members shown as groups of subfilters. You can apply these member attributes as search filters via filter widgets analogous to the ones used when editing. The member attributes of composite attributes in **Simple** mode can be shown as columns in the search result table.

A Model Manager server database comes predefined with two composite attributes, **Model versions** and **Attachments**, whose labels are the pluralized forms of the (single) primitive attributes they contain. Both use the **Table** mode.

See Composite Attributes to learn how administrators can define new composite attributes via the **Administration > Database** area. See Asset Types to learn how to add these composite attributes to sections on the **Asset** page.

Table Filters

If you add a primitive attribute with a set of allowed values — for example, a primitive attribute with a **Radio button** for editing — to a composite attribute using **Table** mode, that attribute will also be available as a table filter on the **Asset** page. Select checkboxes in the list to only show table rows with the corresponding attribute values. Clear all checkboxes to not apply a filter on the attribute.

Adding New Types of Assets

You can customize which attributes are available on an asset page by defining new *asset types*, each asset type having its own list of sections and attributes. This is useful if you want to use the asset management system to store different types of content. You could, for example, add a *Simulation Project* asset type or an *Internal Training Course* asset type — each with their own collection of attributes. Teams sharing the same Model Manager server may also use different processes for managing their simulation projects. Setting up separate asset types with their own distinct workflows is then a good solution — see Using Workflows to Set Up Organizational Processes.

When there is more than one asset type in the database, a list of available asset types appear on the first page of the **New Asset** wizard — see Adding a New Asset. There is also an **Asset Type** filter option in the Asset Filters menu on the **Home** page.

You cannot change the assigned asset type after you have created an asset.

A Model Manager server database comes predefined with a single asset type, simply named **Asset**. It has two sections, **Model versions** and **Attachments**, whose labels are the same as the predefined composite attributes they contain — see Grouping Primitive Attributes into Composite Attributes.

See Asset Types to learn how administrators can define new asset types via the **Administration > Database** area.

Using Workflows to Set Up Organizational Processes

You can incorporate processes and other business rules relevant to your organization by adding *workflows* to the asset management system of a Model Manager server. While assets — as described so far — primarily act as containers for project data, workflows enables you to extend assets as to capture the project process itself. This includes, for example, defining the various stages of a project, the group of people involved in each project stage, and how a project may evolve between different stages.

A workflow is typically defined as a collection of *workflow states* and a collection of *workflow transitions* in-between those states. The workflow management system of a Model Manager server builds upon the asset types and attributes you have added to

the asset management system — see Adding New Types of Data Using Attributes and Adding New Types of Assets. Workflow states are determined by the attribute values of assets. Workflow transitions happen when those attribute values are modified by saving new versions of assets. The workflow state history is directly reachable through the version history of the asset.

By adding workflows to the asset management system, you can:

- Designate a primitive attribute as representing the various stages of a process, with the current value of the primitive attribute corresponding to a specific stage. A process could be defined for the asset itself, or for individual table rows of a composite attribute. Any number of primitive attributes may be designated in this way.
- Let a specific value for a primitive attribute be its default workflow state. When adding a new asset, or when adding a new table row to a composite attribute having the primitive attribute as a column, the field is automatically populated with this value.
- Only allow the values for a primitive attribute to change according to predefined transition rules. The transitions are shown on the **Asset** page as buttons that, when clicked, updates the asset with the new value.
- Grant permissions that controls which users can perform specific transitions.
- Grant permissions to primitive and composite attributes. You can restrict all but a select group of users from updating the value of a specific attribute.
- Grant permissions to assets based on their primitive and composite attribute values. You can, for example, let the value of an attribute determine which users can open the **Asset** page for an asset version or save new versions of an asset.
- Conditionally set a new value for a primitive attribute when some transition occurs.
- Conditionally cancel the save of an asset when some transition occurs.

- Workflows for Primitive Attributes
- Workflows for Composite Attributes
- Workflows for Asset Types

WORKFLOWS FOR PRIMITIVE ATTRIBUTES

For a *primitive attribute workflow*, the workflow state is the current value of a primitive attribute on an asset. A workflow transition corresponds to changing that

value by saving a new asset version. The workflows themselves are applied in the asset management system by adding them to asset types.

See Primitive Attribute Workflows to learn how administrators can define workflows for primitive attributes via the **Administration > Database** area. See Using Workflows with Asset Types to learn how to add workflows for primitive attributes to asset types.

The workflow state for a primitive attribute is defined by the current *state values* for the attribute. For a **Combo box**, **Radio button**, **User picker**, or any other widget in which users specify a single value, the state value is simply the attribute value itself. For widgets in which multiple values may be specified — for example, **Checkbox list**, **List box**, or **Users picker** — the workflow state is a collection of state values.

In the workflow management system, state values are examples of *state conditions*. A state condition is a condition posed on the current state values. It is either satisfied or not satisfied. For a state condition of **State value** type, the condition is satisfied if its defined value is one of the current state values.

A basic example is a `Status` **Radio button** attribute with allowed values `Planned`, `Ongoing`, and `Completed`. A workflow for the `Status` attribute could consist of three **State value** conditions, one for each allowed value.

A *state expression* is a state condition written as a filter expression using the Model Manager search syntax. The state condition is satisfied for a collection of state values if the filter expression would match assets with corresponding attribute values when searching.

An example of a state expression for the `Status` workflow could be `@status:(planned OR ongoing)`. The condition is satisfied if the attribute value is either `Planned` or `Ongoing`.

Searching on Asset Attributes

A Model Manager server also has *built-in state conditions* that can be used as dynamic shorthands for state values and state expressions. The **Any state value** condition, for

example, is satisfied for an attribute with any non-empty value, the **No state value** condition is satisfied for an empty value, and the **Current user** condition is satisfied for the user currently saving an asset. See Table 3-8 for a complete list of all built-in state conditions for primitive attribute workflows.

You can define a default value for a primitive attribute by specifying that a state value is a *default state condition*. The corresponding value of a default state condition is automatically set for a primitive attribute if the current attribute value is empty — either when adding a new asset or when updating an existing asset with an empty value. You can also specify some of the built-in state conditions as default.

The `Planned` state value is a natural default for the `Status` workflow. The corresponding option will be automatically selected on the **Asset** page when adding a new asset.

Transitions occur if the state values for the primitive attribute are modified when saving an asset. Some state values may be removed; others may be added. A transition is defined as a collection of state conditions that are satisfied *before* saving an asset — the *from-state conditions* — and a collection of state conditions that are satisfied *after* saving an asset — the *to-state conditions*. A transition is *enabled* if all its from-state conditions are satisfied. An enabled transition has *fired* if all its to-state conditions are satisfied.

An *action transition* is a transition that is represented on the **Asset** page as a button. When a user clicks the button, a new version of the asset is saved with all state values matched by the from-state conditions removed and all state values identified by the to-state conditions added.

The `Status` workflow could have a `Start` action transition such that clicking the corresponding **Start** button on the **Asset** page changes the attribute value from `Planned` to `Ongoing`.

A transition can have a collection of users and groups that are granted permission to perform the transition. You can also grant permissions to users and groups that can

modify the primitive attribute overall. A collection can be empty, in which case all users are granted permission.

Examples of Workflows for Primitive Attributes

Process Workflows and Free Workflows

A primitive attribute workflow may be in one of two modes. A *process workflow* restricts the possible values of a primitive attribute to the state conditions of type **State value** explicitly defined for the workflow. Changes to the attribute can only be performed via action transitions — modifying an attribute value in some other way is prevented by Model Manager. As the name suggests, you typically use this type of workflow to set up some process in which assets move through various "stages" as described by the workflow's state values. More than one state value may be satisfied at any given time for attributes that can store multiple values in an array. Transitions of type **Action transition** for a process workflow typically correspond to tasks that should be performed to reach the next stage.

A *free workflow* does not impose any restrictions on the values for the primitive attribute, nor how the values may change via transitions. Workflows of this type are typically used to define default values or to grant permissions for primitive attributes. They are also useful as secondary workflows to run some type of side-effect when a transition for some other workflow fires — see Workflows for Composite Attributes and Workflows for Asset Types.

WORKFLOWS FOR COMPOSITE ATTRIBUTES

You apply workflows for primitive attributes present as members of a composite attribute by adding them as *member workflows* to a *composite attribute workflow*. This enables you, for example, to set up a workflow process for the collection of data found in each table row on the **Asset** page. Composite attribute workflows are applied in the asset management system by adding them to asset types.

Continuing with the `Status` workflow introduced in the previous section, a `Tasks` composite attribute could have the `Status` **Combo box** attribute as a table column. The table rows are tasks in various stages of completion.

See Composite Attribute Workflows to learn how administrators can define workflows for composite attributes via the **Administration > Database** area. See Using Workflows with Asset Types to learn how to add workflows for composite attributes to asset types.

A composite attribute workflow can also be used to set up dynamic dependencies between primitive attributes by defining state conditions and transitions for the composite attribute workflow itself. A transition may have one or more *activities* that conditionally *runs* when a transition fires. A *perform transition activity* can be used to automatically perform an action transition for a primitive attribute workflow when the transition for the composite attribute workflow fires. A *transition cancellation activity* can be used to prevent a save if some state conditions are satisfied when the transition fires.

The `Tasks` composite attribute could have a `Responsible` **User picker** attribute as another table column. The action transition **No state value** to **Current user** could be performed via an activity whenever the `Status` attribute changes from `Planned` to `Ongoing` — the user that clicks the **Start** button for a task is automatically set as "responsible" for the task.

Examples of Workflows for Composite Attributes

WORKFLOWS FOR ASSET TYPES

You apply workflows for primitive and composite attributes by adding them as *member workflows* to an *asset type workflow* — the overall workflow defined for the asset type itself. Similar to Workflows for Composite Attributes, you can add state conditions, transitions, and activities for the asset type workflow, thereby enabling you to set up dependencies between different primitive and composite attributes on the **Asset** page.

See Using Workflows with Asset Types to learn how administrators can define workflows for asset types via the **Administration > Database** area.

By adding permissions to the workflow of an asset type, you can define access control rules that apply to *all* assets with that asset type. You can, for example, grant permissions to users and groups that are allowed to add new assets, open the **Asset** page

for assets, or save new versions of existing assets. The permissions can even be granted conditionally based on the current values for primitive and composite attributes.

Examples of Workflows for Asset Types

Controlling Access to Assets Using Libraries

You can organize your assets into different *asset libraries*. This is useful, for example, if you want to set up permissions that differ between various collections of assets — perhaps assigning dedicated asset libraries to different departments in your organization, or simply hiding sensitive assets from all but a select group of users. Grant the **See asset library** permission for groups that may see the assets in the library; grant the **Save in asset library** permission to groups that may save assets in the library.

Assets in different asset libraries can use the same asset type.

You can also grant permissions to assets based on their asset type — see Asset Types — as well as to the individual assets themselves — see Granting Permissions to an Asset.

When there is more than one asset library in the database, a list of available libraries appear on the first page of the **New Asset** wizard — see Adding a New Asset. There is also a **Library** filter option in the Asset Filters menu on the **Home** page.

You cannot move an asset from one asset library to another.

A Model Manager server database comes predefined with one asset library, **Asset library 1**. You can rename the library in the **Administration** > **Database** area — see The Asset Library Page.

See Asset Libraries to learn how administrators can add new asset libraries, as well as grant permissions to these libraries, via the **Administration > Database** area.

Searching on Asset Attributes

In this section, you will learn how to search for assets in a Model Manager server database. You will see how you can match assets by performing full text searches and by applying various filter criteria. Filters can target the predefined fields that are common to all assets as well as any attributes that have been added by Customizing the Asset Management System.

In this section:

Full Text Search

A fast way for you to find assets in a Model Manager server database is to write individual search words, separated by spaces, in the **Search** field on either the **Home** page or, for all other pages, in the top navigation bar. The search words are matched with the identifier, title, and description fields of an asset.

The Model Manager combines all search words that you write with `AND`-logic. In practical terms, this means that the assets in the search result are such that *all* search words are found somewhere in the title or description of the asset.

You can use a wildcard asterisk in a search word to match zero or more arbitrary characters. Appending, for example, a wildcard at the end of a search word will match that word against the beginning of words in the searched text. You can also match on phrases — that is, multiple words in a sequence — by enclosing the search words in quotation marks.

See Wildcard Matching and Phrase Matching in the *Model Manager Reference Manual*.

Filters

You can narrow down a search result by applying filters, either by specifying filter options in the Asset Filters menu or by writing filter expressions in the **Search** field using the Model Manager search syntax — see also Asset Search Syntax.

Filters match in a case-insensitive way in the Model Manager.

The available filters can be categorized based on the types of fields that they match on. These field types affect how the field values specified in the filters are interpreted when searching the database. Text fields and keyword fields, for example, are used for text and name-like search data, respectively. A date or numeric field type enables you, for example, to match either exactly or as a range, while a selection field enables you to match values in a predetermined set.

See Field Types in the *Model Manager Reference Manual*.

To include spaces in a keyword field's value, precede each space using a backslash — see Keyword Field and Escaping Reserved Characters in the *Model Manager Reference Manual*. No such escaping is necessary when searching a field of text type.

ASSET FIELD FILTERS

The field filters common to all assets are:

- Asset Identifier
- Title
- Description
- Library
- Asset Type
- Last Modified
- Last Modified By
- Owner
- Deleted

Asset Identifier

The **Identifier** filter is a selection filter that matches on the unique identifier of an asset.

Title

The **Title** filter is a text field filter that matches on the title of an asset. You may find this filter useful when the searched title contains a word that is also commonly found in descriptions, such that performing a Full Text Search would yield too many results. The filter is only available using the Model Manager search syntax — see Table 4-3.

Description

The **Description** filter is a text field filter that matches on the description of an asset. Similar to a Title filter, you may find this useful when the searched description is also a common title. The filter is only available using the Model Manager search syntax — see Table 4-3.

Library

The **Library** filter is a selection filter that matches on the name of the asset library that an asset belongs to.

Asset Type

The **Asset Type** filter is a selection filter that matches on the name of the asset type assigned to an asset.

Last Modified

The **Last Modified** filter is a timestamp field filter for when the latest asset version was saved.

Last Modified By

The **Last Modified By** filter is a selection field filter on the user that saved the latest asset version. Write the name or display name of the user. A space in a searched name must be escaped by preceding the space with a backslash.

Owner

The **Owner** filter is a selection field filter on the current owner of an asset. Write the name or display name of the user. A space in a search name must be escaped by preceding the space with a backslash.

Deleted

The **Deleted** filter is a Boolean field filter on whether or not an asset is deleted.

ATTRIBUTE FIELD FILTERS

Filter options for attributes appear in the lower part of the Asset Filters menu on the **Home** page. Similar to Asset Field Filters, each attribute filter option has an associated field type that affects how the values are interpreted when searching the database. The assigned field type depends on the value type of the primitive attribute, see Table 4-2.

Only primitive attributes used by at least one asset type — either directly as a top-level attribute added to one of the asset type's sections or indirectly as a member of an added composite attribute — are shown in the menu. If you apply a filter on asset types themselves via the **Asset Type** filter, the list of shown attribute filters is further reduced based on the selected asset types.

Filters on primitive attributes of a **Date** or **Number** value type are specified as a range in the Filters section. Primitive attributes with a set of allowed values are filtered by selecting a subset of values from a list.

A primitive attribute with the **Link** value type can be matched on either the link URL address or the link text.

A primitive attribute with the **Attachment** value type can be matched on the filename, file type, last modified date, and file size of the uploaded attachment.

A primitive attribute with the **File version** or **Model version** value types can be matched on the title, item version type, item save type, saved date, saved by-user, and owner for the linked item version.

A primitive attribute with the **User** or **User array** value type can be matched on either the identifying key or the name and display name of the user.

TABLE 4-2: THE FIELD TYPE ASSIGNED TO EACH PRIMITIVE ATTRIBUTE VALUE TYPE.

PRIMITIVE ATTRIBUTE VALUE TYPE	FIELD TYPE
Attachment	Multiple field types (Table 4-5)
Boolean	Boolean
Date	Date
File version	Multiple field types (Table 4-6)
Integer	Numeric
Keyword	Keyword
Keyword array	Keyword
Link	Multiple field types (Table 4-4)
Model version	Multiple field types (Table 4-6)
Text	Text
User	Selection

Composite Attribute Filters

Composite attributes also appear in the Asset Filters menu, with each of their primitive attribute members available as subfilter options. For table-valued composite attributes, the primitive attribute filters combine with Boolean `AND`-logic *on a per table row basis*.

Primitive attributes shown as top-level filters in the **Filters** menu only match the values of those primitive attributes added to the top-level of asset type sections, not those that are members of composite attributes. Conversely, primitive attributes shown as subfilters to composite attributes only match values inside those composite attributes.

Asset Search Syntax

The filter functionality in the asset management system of a Model Manager server is based on the same tailor-made search syntax used when searching models and their contents. The syntax enables you to write filter queries that find assets whose built-in fields and attribute fields satisfy arbitrarily complex constraints.

In this section, you will learn how you can formulate such custom filter queries. You will see how to write simple expressions for filters on single asset fields and primitive attribute fields as well as how to nest field expressions using the primitive attribute members of a composite attribute. For more details on the search syntax, see The Model Manager Search Syntax in the *Model Manager Reference Manual*.

BASIC ASSET FIELD EXPRESSIONS

An asset filter consists of one or more *field expressions* combined with Boolean operators — for example, `AND`, `OR`, and `NOT` — and other grouping operators. Each field expression specifies which *field* and what *field value* is being filtered on.

You write an asset field expression using an `@`-notation of the general form:

```
@<field-name>:<field-value>
```

with *`<field-name>`* equal to the name of one of the available asset fields in Table 4-3, and *`<field-value>`* the value being filtered on. Write, for example,

```
@title:crane
```

to find assets whose title contains the word `crane`. To match on several search words, enclose the words with parentheses. Write

```
@title:(mounted crane)
```

to find assets whose title contains the words `mounted` and `crane`.

A space between two search words is automatically interpreted as a Boolean `AND`. The previous expression is thus equivalent to:

```
@title:(mounted AND crane)
```

Write

```
@title:(mounted OR crane)
```

if you want to find assets whose title contain `mounted` *or* `crane`.

If you want to combine a full text search with a custom filter query, write the former first. The following is valid:

```
mounted crane @assetType:project @lastModifiedBy:alice
```

and matches on assets whose title or description contains the words `mounted` and `crane`, whose asset type is `Project`, and that was last modified by user `Alice`. The following is not valid:

```
@assetType:project @lastModifiedBy:alice mounted crane
```

and results in an error message.

See Basic Field Expressions in the *Model Manager Reference Manual*.

TABLE 4-3: FIELD EXPRESSIONS FOR ASSET FIELDS.

SYNTAX	TYPE	DESCRIPTION
@assetIdentifier:...	Selection	The unique identifier of an asset.
@assetKey:...	Selection	The unique key of an asset.
@assetLibrary:…	Selection	The name of the asset library that an asset belongs to.
@assetType:…	Selection	The name of the asset type of an asset.
@assetVersionKey:...	Selection	The unique key of an asset version.
@description:…	Text	The description of an asset.
@isDeleted:…	Boolean	The deleted state of an asset.
@lastModified:…	Timestamp	The instant in time when an asset was last modified. May be used in place of @saved:...

SYNTAX	TYPE	DESCRIPTION
@lastModifiedBy:…	Selection	The name or display name of the user that last modified an asset. Escape spaces in names with a backslash. May be used in place of @savedBy:...
@owner:…	Selection	The name or display name of the user that owns the asset. Escape spaces in names with a backslash.
@saved:...	Timestamp	The instant in time when an asset version was saved.
@savedBy:...	Selection	The name or display name of the user that saved an asset version. Escape spaces in names with a backslash.
@title:…	Text	The title of an asset.

PRIMITIVE ATTRIBUTE FIELD EXPRESSIONS

You can filter on primitive attribute fields using a search syntax similar to that of the predefined asset fields:

```
@<attribute-identifier>:<field-value>
```

with *<attribute-identifier>* equal to the identifier of a primitive attribute and *<field-value>* the value being filtered on. Write, for example,

```
@status:planned
```

to find all assets with the value `planned` for a hypothetical primitive attribute `status` of **Keyword** value type. Similarly, the expression

```
@week_estimate:[5 TO 10]
```

matches all assets with a range of values for a hypothetical primitive attribute `week_estimate` of **Integer** value type.

A primitive attribute field expression only matches on the values of attributes added as *top-level attributes* to the sections of an asset type. See Composite Attribute Field Expressions for the search syntax used to filter on primitive attributes that are members of composite attributes.

If the attribute identifier is the same as one of predefined field names in Table 4-3, precede the identifier with a backslash. Thus, the field expression `@title:<field-value>` matches on the title of an asset; the field expression `@\title:<field-value>` matches on the attribute with identifier `title`.

For primitive attributes with value types that support filtering on multiple fields, the above syntax automatically targets a default field. For an **Attachment** it is the filename, for a **File version** or **Model version** it is the item version's title, and for a **Link** it is the link text. Write, for example,

```
@product_manual:truck\ mounted\ crane.pdf
```

to find all assets with an uploaded file `truck mounted crane.pdf` for a hypothetical **Attachment** primitive attribute with identifier `product_manual`.

The filename field of an attachment has a keyword field type, which means that spaces must be escaped by a backslash.

To match on one of the other fields, write

```
@<attribute-identifier>{@<field-name>:<field-value>}
```

with *<attribute-identifier>* equal to the identifier of the attribute and *<field-name>* equal to the name of one of the available fields in Table 4-4, Table 4-5, and Table 4-6. Write, for example,

```
@product_manual{@fileType:pdf}
```

to find all assets with product manuals of PDF type. You can also combine multiple fields within the curly braces. Write,

```
@product_manual{@fileType:pdf @lastModified:[* TO 12/31/20]}
```

to find product manuals of PDF type that were created before 2021.

A similar syntax is used when searching on the fields of **File Version**, **Model Version**, or **Link** attributes. Write, for example,

```
@auxiliary_data{@itemVersionType:fileset}
```

to find all assets linking to fileset versions via a hypothetical **File Version** primitive attribute with identifier `auxiliary_data`.

TABLE 4-4: FIELD EXPRESSIONS FOR PRIMITIVE ATTRIBUTES OF LINK VALUE TYPE.

SYNTAX	FIELD TYPE	DESCRIPTION
@text:...	Text	The link text of the link.
@url:…	Keyword	The URL address of the link.

TABLE 4-5: FIELD EXPRESSIONS FOR PRIMITIVE ATTRIBUTES OF ATTACHMENT VALUE TYPE.

SYNTAX	FIELD TYPE	DESCRIPTION
@filename:…	Keyword	The filename of the attachment.
@fileType:...	Keyword	The file type extension of the attachment.
@lastModified:…	Timestamp	The last modified timestamp of the attachment,
@size:…	Numeric	The size in bytes of the attachment.

TABLE 4-6: FIELD EXPRESSIONS FOR PRIMITIVE ATTRIBUTES OF LINKED ITEM VERSION VALUE TYPES.

SYNTAX	FIELD TYPE	DESCRIPTION
@itemKey:...	Selection	The unique key of the item that the item version belongs to.
@itemSaveType:...	Selection	The item save type of the item that the item version belongs to.
@itemType:…	Selection	The item type of the item that the item version belongs to.
@itemVersionKey:...	Selection	The unique key of the item version.
@itemVersionType:...	Selection	The item version type of an item version.
@originItemKey:...	Selection	The unique key of the origin item to the item that the item version belongs to.
@owner:…	Selection	The name or display name of the user that owns the item. Escape spaces or other reserved characters in names with backslash.
@saved:...	Timestamp	The instant in time when the item version was saved.
@savedBy:...	Selection	The name or display name of the user that saved the item version. Escape spaces or other reserved characters in names with backslash.
@title:…	Text	The title of an item version.

COMPOSITE ATTRIBUTE FIELD EXPRESSIONS

You can filter on primitive attributes that are members of composite attributes using a syntax of the form:

`@<composite-identifier>{@<primitive-identifier>:<field-value>}`

with *`<composite-identifier>`* equal to the identifier of a composite attribute, *`<primitive-identifier>`* equal to the identifier of a primitive attribute that is a member of the composite attribute, and *`<field-value>`* equal to the value being filtered on. Write, for example,

```
@journal_reference{@conference:paris @year:2020}
```

to find all assets with the values `Paris` and `2020` for the two primitive attributes, `conference` and `year`, belonging to the composite attribute `journal_reference`. This syntax is especially useful when you want to match on multiple cells in a table row corresponding to a composite attribute in **Table** mode. Write, for example,

```
@models{@model:(mounted crane) @product_version:11.2}
```

to find all assets with a table row in a hypothetical `models` composite attribute table in which the `model` column has a model version title containing `mounted crane`, and the `product_version` column has the value `11.2`.

SYNTAX ERRORS

The Model Manager server web interface automatically detects if your write an expression in a search or filter input field using a malformed search syntax. A warning banner with a message describing the problem is shown below the input field. The warning banner stays visible until you either correct the problem or click the **Close** button on the banner.

A common source of error is forgetting to escape reserved characters by preceding them with a backslash (\) when writing search words in the **Search** field or the **Filters** menu on the **Home** page. This includes, for example, parentheses and quote characters. See also Escaping Reserved Characters in the *Model Manager Reference Manual*.

Example: A Project Asset Type

This section showcases some of the tools available in the asset management system of a Model Manager server using the example of a hypothetical *Project* asset type. You will learn how to customize the asset management system by defining new attributes that may, for example, be relevant to a project in a product development context. These attributes are added to a new asset type to be used by project-like assets. You will also learn how to create, edit, and search such assets, including working with linked model versions, supplementary files, and other metadata added to the assets.

Some of the steps in this tutorial involves the use of the Model Manager tools in the COMSOL Multiphysics software — you can skip these steps if you do not have access to a COMSOL Multiphysics installation.

- Defining the Project Asset Type
- Adding Projects
- Linking to Simulation Models in the Database
- Viewing Older Versions
- Finding Projects Using Attribute Filters
- Opening Linked Simulation Models in the COMSOL Desktop

See also Working with Models in Databases in the *Model Manager Reference Manual*.

Defining the Project Asset Type

The projects saved in our customized asset management system will be based off a new asset type that has fields for storing the following data:

- **Status**. One of `Planned`, `Ongoing`, or `Completed` for the status of a project.
- **Work group**. A label for the group within the organization working with a project.
- **Documentation**. An uploaded file that serves as documentation for a project.
- **Presentation**. An uploaded file used during a presentation related to a project.
- **Presented**. The date when a presentation was held.
- **Model version**. A simulation model being used in a project.

Except for the last field, these fields are not available in the default asset management system included with a new Model Manager server database. Part of our task when creating the new project asset type will therefore be to extend the asset management system with new attributes that can hold the corresponding data.

You need to either be an administrator or have been granted the **Manage asset types** database permission to customize the asset management system — see Database Permissions.

The new asset type is added from the **Database** administration area in the Model Manager server web interface:

1. Click the cog wheel in the top navigation bar and select **Database** in the opened **Administration** menu.
2. Click **Asset Types** in the **Database** navigation sidebar.
3. Click **Add** to open the **Add Asset Type** page for adding a new asset type to the asset management system.
4. Write `Project` in the **Name** field.
5. Write `PROJ` in the **Alias** field.

The alias `PROJ`, combined with a unique integer, is used in web links for project assets — the first project you create will get the identifier `PROJ-1`. Once the new asset type has been saved to the database, this alias cannot be changed.

Attributes are organized into collapsible sections on an asset page. You will add three sections for the project asset type: **Information**, **Files**, and **Simulation Models**.

1. On the **Sections** card, write `Information` in the **Title** field.
2. Click the **Add Section** button to add a second section. Write `Files` in its **Title** field.
3. Click the **Add Section** button again to add a third and final section. Write `Simulation models` in its **Title** field.

While defining a new asset type, you may find it useful to see a preview of how the corresponding **Asset** page will look like. Click the **Preview** menu option located above the **Sections** card. The page switches into a preview mode for a hypothetical project

asset. It shows the three sections — all currently void of any attributes. Click the **Sections** menu option to return to the list mode used to edit the sections.

There will already be a default asset type on the **Asset Types** page. This asset type is automatically created for a new Model Manager server database. It uses the two predefined composite attributes — **Attachments** and **Model versions**.

ADDING YOUR FIRST PRIMITIVE ATTRIBUTE

We will add the **Status** field as a new primitive attribute placed first in the **Information** section.

1. In the **Add attribute** list in the **Information** section, select **New primitive attribute**.
2. Write `Status` in the **Label** field.

   The **Identifier** field is automatically populated with a lowercase `status` for the unique identifier of the new attribute. This identifier is used, for example, when Searching on Asset Attributes. Once the attribute has been created, this identifier cannot be modified.
3. Write `The current status of a project` in the **Description** field.
4. Select **Combo box** in the **Widget** list.

   The **Value** list is automatically set to the default value type for the widget, which in the case of a **Combo box** is **Keyword**.
5. In the **Allowed Values** table, write `Planned` in both the **Label** input field and the **Value** input field. Click **Add Allowed Value**. Repeat this with `Ongoing` and `Completed`.
6. Finish by clicking the **Add Primitive Attribute** button.

You have created your first primitive attribute. As its label and description indicates, it can be used to track the current status of a project.

There will already be two primitive attributes in the **Add Attribute** list — **Attachment** and **Model version**. These are automatically created for a new Model Manager server database.

Primitive Attributes

ADDING MORE ATTRIBUTES

You will use a simple **Input field** for storing the work group within the organization that is involved with the project. Another reasonable widget for such a field would be a **Combo box** with a predefined list of available work groups.

1. In the **Add attribute** list in the **Information** section, select **New primitive attribute**.
2. Write `Work group` in the **Label** field.
3. Write `Organizational unit` in the **Description** field.
4. Select **Input field** in the **Widget** field.
5. Click **Add Primitive Attribute**.

Open the preview mode again. As expected, the **Information** section now displays a **Status** field and a **Work group** field with some placeholder data in both fields. Return to the sections list mode.

Users of the asset management system would likely want to upload multiple documentation and presentation files on a single project. You can accomplish this by adding the new **Documentation** and **Presentation** fields as columns in two separate tables. For the former:

1. In the **Add attribute** list in the **Files** section, select **New composite attribute**.
2. Write `Documentation files` in the **Label** field.

   The **Description** field can be empty. Keep the default **Table** value in the **Mode** list.
3. In the **Add primitive attribute** list under **Member attributes**, select **New primitive attribute**.
4. Write `Documentation` in the **Label** field.
5. Write `Documentation file` in the **Description** field.
6. Select **File Upload** in the **Widget** list.

   The **Value** field automatically defaults to **Attachment**.
7. Click **Add Primitive Attribute** to add the new primitive attribute as a member of the composite attribute.
8. Click **Add Composite Attribute** to add the composite attribute itself to the **Files** section.

A composite attribute is an aggregation of one or more primitive attributes. As such, it may be used to store a *composite* value. Moreover, a composite attribute can be defined in one of two modes — **Simple** or **Table**. The former is used when there is a single composite value. As such, it is displayed as a simple grouping of fields on the

Asset page. The latter is used when there are multiple such composite values — the composite attribute is displayed as a table with the individual composite values as table rows and the primitive attributes as columns.

The **Documentation files** composite attribute is a special case where the composite value is just a single value — the uploaded file. It is displayed as a table having a single column, with each table row containing a single documentation file.

There will already be two composite attributes in the **Add Attribute** list — **Attachments** and **Model versions**. These are automatically created for a new Model Manager server database.

Composite Attributes

Any presentation held for a project has a corresponding presentation file and a date when the presentation was held. To store this data, add a table with two columns to the **Files** section:

1. In the **Add attribute** list in the **Files** section, select **New composite attribute**.
2. Write `Presentations` in the **Label** field.
3. Write `Presentations held` in the **Description** field.
4. In the **Add primitive attribute** list under **Member attributes**, select **New primitive attribute**.
5. Write `Presentation` in the **Label** field.
6. Write `Presentation file` in the **Description** field.
7. Select **File Upload** in the **Widget** list.
8. Click **Add Primitive Attribute**.
9. Select **New primitive attribute** again for an additional **Presented** column in the table. Use `Presented`, `Date of presentation`, and **Date picker** for the **Label**, **Description**, and **Widget** fields, respectively. Finish with **Add Primitive Attribute**.
10. Click **Add Composite Attribute** to add the composite attribute as a second attribute to the **Files** section.

Now could be a good point to preview the **Asset** page once more. Click **Preview**. You will see the two **Documentation files** and **Presentation** tables in the **Files** section. Click **Sections** to return to the sections list mode.

FINISHING THE ASSET TYPE DEFINITION

The last field, **Model version**, already exists as a primitive attribute. It is used by the default asset type that is automatically created for a new Model Manager server database. The primitive attribute itself is a member of an automatically created composite attribute, **Model versions**. You will reuse this latter attribute to link multiple model versions on a project: in the **Add attribute** list in the **Simulation Models** section, select **Model versions**.

With all attributes added to their respective sections, click **Save** at the bottom of the **Add Asset Type** page to save the new project asset type.

You should see three sections on the **Asset Type** page containing five attributes in total — two primitive attributes and three composite attributes. Click on the **Presentations (presentation)** link to open its details page. You should see the primitive attributes **Presentation** and **Presented**, the two members of the **Presentations** composite attribute.

Asset Types

UPDATING AN ATTRIBUTE DEFINITION

As currently defined, the **Status** primitive attribute is edited on an **Asset** page by selecting a value in a **Combo box** widget — a list of options that you can show or hide by clicking on the widget. This is useful when there are many allowed values to select among. In our case, though, the **Radio button** widget may be more suitable:

1. Click **Primitive Attributes** in the **Database** navigation sidebar.
2. On the **Primitive Attributes** page, click **Status** to open its details page.
3. Click **Edit**.
4. In the **Widget** list, select **Radio button**.
5. Click **Save**.

The **Radio button** widget always shows all options when editing — useful in this case, as there are only three options to choose from.

Adding Projects

With the new asset type in place, it is time to start adding projects to the database. You will add two projects that correspond to the development of two hypothetical products

— a wrench and a busbar. These examples will be used to demonstrate how you can add, edit, and search for assets in your customized asset management system.

See also *Example 1: Structural Analysis of a Wrench* and *Example 2: The Busbar — A Multiphysics Model* in *Introduction to COMSOL Multiphysics*.

1. Click **Add** in the top navigation bar and select **New Asset**.
2. Write `Combination wrench` in the **Asset title** field.
3. Select **Project** in the list.
4. Click **Continue**.
5. In the **Description** field, write `Improving the structural integrity of the combination wrench product line`.
6. In the **Status** field, select **Ongoing**.
7. In the **Work group** field, write `Research and Development`.
8. Click **Save**.

You have now created a first asset in the database of the project asset type. Click the COMSOL logo or the **Home** link in the top navigation bar to open the **Home** page. Write `Combination wrench` in the **Search** field and press Enter. The project asset is shown in the search result table. Click on its table row to return to the **Asset** page for the project.

Adding a New Asset

EDITING THE PROJECT

At the moment, the project does not contain much data other than its title, description, status, and work group value. There are, for example, no supplementary files uploaded on the project.

1. Click **Edit** in the upper-right corner.

   The **Asset** page for the project is shown in its editing state.
2. Click the **Add Row** button under the **Documentation files** table in the **Files** section.

3 Click the **Click here to upload** link and browse for a file on your computer. For the purpose of this tutorial, you can select any file on your file system to masquerade as some documentation manual.

   The selected file is added as a row in the table. You can add more documentation files by repeatedly clicking the **Add Row** button under the **Documentation files** table and selecting a file via the **Click here to upload** link.

4 Click the **Add Row** button under the **Presentations** table.

5 Upload any file in the **Presentation** column much like you did for the documentation files.

6 In the **Presented** column, click the calendar and select an imagined date the presentation was held.

7 Click **Save**.

The **Files** section is now filled with a few uploaded files associated with the project. You can click the filename of a file to download it to your computer.

Every time you click **Save**, a new version of the asset is stored in the database. To see all changes made to the asset when the current version was saved, click the expander button next to the **Versions** button and select **Compare with Previous**. The **Asset Comparison** page opens with a comparison between the second, latest, version and the first version. You will see the documentation files and presentations that you added in the previous steps indicated by plus signs and green color. Click the arrow button in the upper left corner to return to the **Asset** page.

The Asset Page

EDITING THE ASSET TYPE

You can modify and extend an existing asset type even after you have created your first asset of that type. You may, for example, want to rearrange the layout of sections and attributes on the project asset type, or perhaps add new types of attributes to your projects.

See Table 3-6 for a complete list of attribute widgets that can be added to an asset page, including the types of values that can be edited using these widgets.

As an example, you are going to add the predefined **Attachments** composite attribute to the project asset type. You may, for example, decide to use this for miscellaneous files that are neither documentation files nor presentations.

1. Click the cog wheel in the top navigation bar and select **Database** in the opened **Administration** menu.
2. Click **Asset Types** in the **Database** navigation sidebar.
3. Click the **Project** asset type on the **Asset Types** page.

   The details page for the asset type is opened. You can see the current layout of sections and the attributes they contain.
4. Click **Edit**.
5. In the **Files** section, select **Attachments** in the **Add attribute** list.

   The **Attachments** composite attribute is added last to the section. You can rearrange the attributes by moving the mouse pointer over the crossed arrows icon in a table row, pressing and holding down the mouse button to "grab" the table row, moving the mouse pointer to "drag" the table row to its desired location in the table, and then releasing the mouse button to "drop" the table row in its new location.
6. Click **Save**.

Open the **Asset** page for your combination wrench project. An empty **Attachments** table is now visible in the **Files** section.

There are no restrictions on the types of supplementary files you decide to upload on your projects. Examples include project notes, word processing documents, reports, slides, images, and videos. Files used as auxiliary data by simulation models — for example, CAD data or interpolation functions — are best version-controlled inside repositories, though, side by side with the models.

You can add a new **File version** primitive attribute to an asset type if you want to link to data files version-controlled in your repositories on the **Asset** page.

ADDING MORE PROJECTS

Add a second asset of the project asset type to the database:

1. Click **Add** in the top navigation bar and select **New Asset** in the opened menu.
2. Write `Busbar redesign` in the **Asset title** field.
3. Select **Project** in the list.

4 Click **Continue**.

5 In the **Description** field, write `New design for the busbar component`.

6 In the **Status** field, select **Planned**. Leave **Work group** empty.

7 Click **Save**.

A second project asset is created in the database. The same set of attributes available for editing on the first asset are also present on this new asset.

Linking to Simulation Models in the Database

You can add links to an arbitrary selection of model versions on your project assets — both to versions of the same model, as well as versions of different models. Which version of a particular model you choose to add depends on the process that best suits your needs. You can add a first version of a model already when the model is created. As you reach important milestones in your simulation work, you can add, or replace with, newer model versions by editing the asset. Perhaps when the **Status** attribute is set to **Completed**, you add the final version of a model.

To demonstrate this linkage, you will save two models to the Model Manager server database by connecting to the database from the COMSOL Desktop modeling environment. For the combination wrench project:

1 Launch the COMSOL Multiphysics software.

2 In the **File** menu, select **Application Libraries** ().

3 Select **COMSOL Multiphysics** > **Structural Mechanics** > **wrench**.

4 Click **Open** ().

5 In the **File** menu, select **Save To** ().

6 In the **Save** window, select **Add Database** ( ) in the list of options.

7 Click the **Connect to Server Database** button ().

8 Follow the steps in Connecting to a Server Database in the *Model Manager Reference Manual* to connect to the Model Manager server database.

After COMSOL Multiphysics has finished connecting to the Model Manager server, the server database is shown selected in the **Save** window.

9 Click the **Save** button () to save a first version of the model in the database.

Similarly for the busbar redesign project:

1 Select and open **COMSOL Multiphysics** > **Multiphysics** > **busbar** in the **Application Libraries** window.

2 Select **Save To** ( ) in the **File** menu.

3 In the **Save** window, select the Model Manager server database in the list of options.

4 Click **Save** ( ).

You can also save a model to a Model Manager server database by uploading an MPH file via the web interface — see Adding a New Model.

Return to the **Asset** page for the busbar redesign project in the asset management system.

1 Click **Edit**.

2 Under the **Model versions** table, click the **Add Row** button.

3 In the added table row, write `busbar` in the **Search or paste** field.

4 Select **Electrical Heating in a Busbar** in the search result to add the model version to the asset.

5 Click **Save**.

Open the **Asset** page for the combination wrench project and repeat the previous steps for the wrench model.

Opening Linked Simulation Models in the COMSOL Desktop

Once you have found a model version of interest on an asset, you can quickly open that model version in the COMSOL Desktop. On the **Asset** page for the combination wrench project:

1 Click the title of the wrench model in the **Model version** column of the **Model versions** table.

The **Model** page is opened for the linked model version. On the page, you can, for example, browse the model tree of the version, download the version as an MPH file, or save a new version by updating from an MPH file.

2 Click the **Open in COMSOL Desktop** button in the toolbar.

The **Opening Model in the COMSOL Desktop** dialog is shown in the web interface while an attempt is made to open the model version in the COMSOL Multiphysics program session running on your desktop.

3 In COMSOL Multiphysics, click **Open in This Window** in the shown dialog to open the wrench model in the COMSOL Desktop.

	The **Open in COMSOL Desktop** functionality is only available for COMSOL Multiphysics version 6.4 or later on the Windows® operating system. You must have also selected the **Associate links in the Model Manager server web interface with this installation** checkbox in the **Options** step of the COMSOL Multiphysics installer.

As an alternative to **Open in COMSOL Desktop**:

1 Click the **Copy Location to Clipboard** button on the **Model** page.

A location string identifier for the model version is automatically copied to your computer's clipboard.

2 In the **File** menu in COMSOL Multiphysics, select **Open From** ().

3 In the **Open** window, the **Clipboard** () option is automatically selected in the list.

4 Click the **Open** button () to open the wrench model in the COMSOL Desktop.

	Copying to clipboard via the **Copy Location to Clipboard** button is only available when accessing the Model Manager server web interface via `localhost` or through a secure connection using HTTPS.

	Opening Model Versions in the COMSOL Desktop

Viewing Older Versions

You can access older versions of an asset via the **Asset Versions** page. You can also restore an older version as a new latest version — useful, for example, if you make unwanted changes to an asset by mistake.

On the **Asset** page for the combination wrench project:

1 Click the **Versions** button in the upper-right corner.

The **Asset Versions** page shows a table with the three saved versions of the combination wrench project.

2 Click the middle row in the table.

   The **Asset** page is opened for that version. Notice that there is no linked model version in the **Model versions** table.

3 Click the **Versions** button once more and then click the bottom row in the table.

   The **Asset** page is opened for the first version of the project. There are no uploaded files on the project.

4 Click the **View Latest Version** link at the top of the page.

   The **Asset** page is opened for the latest version. There are both uploaded files and a linked model version present on the page, as expected.

From the **Asset Versions** page, you can also compare all changes made when going from an older version to a newer version. Select the checkboxes for the corresponding two versions in the table and click the **Compare** button. The **Asset Comparison** page opens with all changes indicated as either additions or removals.

Click the COMSOL logo or the **Home** link in the top navigation bar to return to the **Home** page.

Finding Projects Using Attribute Filters

As your collection of assets in the database grows in size, it will become important to be able to search and filter on the assets. On the **Home** page:

1 If not already expanded, click the vertical bar on the left side of the page to expand the **Filters** menu.

2 Under **Asset Type**, select **Project** in the list.

   The search result is updated to show the two assets with the project asset type.

3 Under **Status**, select **Ongoing** in the list.

   The search result shows the ongoing combination wrench project.

4 Click the **Clear Filters** button to clear the two applied filters.

5 In the **Title** field under **Model versions > Model version**, write `electrical heating busbar`.

   The **Title** field of the **Model version** attribute matches on the title of the linked model version. In this case, the search result shows the busbar redesign project. You can also try a wildcard search. Test with, for example, `electric*` in the **Title** field.

6 Click the **Clear Filters** button to clear the **Model versions > Model version** filter.

7 Under **Work group**, write `research and development`.

   You may be surprised to find that no assets match the filter. The reason is that spaces must be escaped with a backslash when filtering on attributes of keyword value type. Alternatively, you can filter using a wildcard or using phrase matching.

8 Replace the filter value with `research\ and\ development`. Also test with an appended wildcard, `research*`, and as a phrase, `"research and development"`.

   In all three cases, you will match on the combination wrench project.

9 Click **Clear Filters** to clear the **Work group** filter.

You can also filter by writing a filter expression directly in the **Search** field. Write, for example, `@status:planned` and press Enter to match on the busbar redesign project.

See Searching on Asset Attributes to learn more on how you can search and filter assets on their attributes.

Adding a Status Workflow for Projects

The **Status** attribute for your project assets can be used to describe the "stages" of a project in which the initial stage is always **Planned**, followed by **Ongoing**, and then finally **Completed**. You can incorporate these rules by adding a *workflow* for the **Status** attribute to the asset management system.

1 Click the cog wheel in the top navigation bar and select **Database** in the opened **Administration** menu.

2 Click **Primitive Workflows** in the **Database** navigation sidebar.

3 Click **Add** to open the **Add Primitive Attribute Workflow** page for adding a new workflow for a primitive attribute.

4 In the **Attribute** list, select **Status**. Keep the preselected **Process workflow** option in the **Mode** list.

5 Write `Status Workflow` in the **Label** field.

6 Click **Continue** to continue to the next page

The **Add Primitive Attribute Workflow** page shows three state values and two transitions automatically generated based on the three allowed values for the **Status** attribute. The two transitions are **Planned** to **Ongoing**, and **Ongoing** to **Completed**. For the purpose of this tutorial, these transitions will do just fine.

1. For the **Planned** state value, under **State conditions**, select the **Default** checkbox to enable the value as the initial stage in the workflow.
2. In the first empty **Label** field, under **Transitions**, write **Start Project**. In the second empty **Label** field, write **Complete Project**.
3. Click **Save**.

With the workflow created, you need to add it to the **Project** asset type for it to apply to your assets.

1. Click **Asset Types** in the **Database** navigation sidebar.
2. Click **Project** to open the **Asset Type** page. Click **Edit**.
3. Click the **Workflow** menu option and select the **Use workflows** checkbox to enable workflows for the asset type.
4. In the **Add workflow** list, under **Member workflows**, select **Status Workflow**.
5. Click **Save**.

Open the **Asset** page for the busbar redesign project. The page now displays a **Start Project** button next to the **Planned** value in the **Status** field. The button corresponds to the transition from the "planned stage" to the "ongoing stage".

1. Click **Start Project**.

   A new version is saved with the **Status** field updated to **Ongoing**. The **Start Project** button has also been replaced with a **Complete Project** button on the **Asset** page.
2. Click **Complete Project**.

   An additional version is saved with the **Status** field now updated to **Completed**. No button is shown as we have reached the last stage.

You can also try out the workflow rules by creating a new project. As you will notice, the **Planned** option is automatically preselected as the initial value for the **Status** field. Select the **Completed** option and click **Save**. As expected, Model Manager shows an error informing you that the transition from **Planned** to **Completed** is not allowed for the **Status Workflow**. Select the **Ongoing** option instead and click **Save**. A new project already in the "ongoing stage" is successfully added to the database.

See Using Workflows to Set Up Organizational Processes to learn more about the workflow management system of a Model Manager server.

This concludes this introductory tutorial on the asset management system.

5

Glossary

This Glossary of Terms contains terms related to system and database administration as they relate to a Model Manager server. For references to further information about a term, see the index.

Glossary of Terms

administrator A privileged account with a Model Manager server that always passes permission requirement checks.

account Settings for a user that can log in to a Model Manager server by supplying their username and password.

action transition A workflow transition that can be initiated by clicking a button, or selecting a menu option, on the **Asset** page.

activity An operation that is performed by Model Manager when a workflow transition fires.

asset A container for data associated with a simulation project.

asset library A collection of assets.

asset type A collection of attribute and workflow definitions that together specify a type of asset.

asset type workflow The overall workflow defined for an asset type. May contain primitive and composite attribute workflows as member workflows.

built-in state condition A state condition that can be used as a dynamic shorthand for a state value or state expression.

composite attribute An attribute that contains a collection of primitive attributes as members.

composite attribute workflow A workflow for a composite attribute. May contains primitive attribute workflows as member workflows.

connector Configuration settings for ports that a Model Manager server listens on. May be used enable transport layer security (TLS) for secure connections via HTTPS.

external server component A server component for a Model Manager server database provided externally from Model Manager server.

free workflow A workflow that does not impose any restrictions on the workflow's state values, nor how the state values may change via workflow transitions.

from-state condition A state condition that must be satisfied if a workflow transition should be enabled.

managed SQL database server A SQL database server whose process and backup routines are managed by the Model Manager server itself. A Model Manager server database uses a SQL database in the SQL database server for version control metadata.

managed resources directory A data directory whose backup routines are managed by the Model Manager server itself. Used by a Model Manager server database to store large binary and text data.

managed search index server A search index server whose process is managed by the Model Manager server itself. A Model Manager server database uses two indices in the search index server for the Model Manager search functionality.

managed server component A server component for a Model Manager server database whose process and backup routines are managed by the Model Manager server itself.

member workflow A workflow added as a member to another workflow.

perform transition activity A workflow activity that fires a collection of enabled action transitions when it runs.

primitive attribute A field for storing data on an asset.

primitive attribute workflow A workflow for a primitive attribute. The workflow state is determined by the current value of the attribute on an asset. The workflow transitions describe updates to this value.

process workflow A workflow that restricts the possible values of a primitive attribute to the state conditions of type **State value** explicitly defined for the workflow. Changes to the attribute can only be performed via action transitions.

state condition A condition posed on the state values for a workflow. The condition is either satisfied or not satisfied.

state expression A state condition written as filter expression using a the Model Manager search syntax. The state condition is satisfied if it matches state values as a logical expression.

state value A state condition that matches the state values of a workflow by equality.

to-state condition A state condition that must be satisfied for an enable workflow transition to fire.

transition A transformation of state values for a workflow when saving an asset. A transition is enabled if its from-state conditions are all satisfied. An enabled transition fires if its to-state conditions are all satisfied.

transition cancellation activity A workflow activity that cancels a save if a collection of state conditions are satisfied when it runs.

value type The data type for the values that can be stored in a primitive attribute.

widget type The field widget used to update the value stored in a primitive attribute.

workflow A set of rules for how the values of an attribute, or attributes, are updated on an asset.

See also Glossary of Terms in the *Model Manager Reference Manual*.

Index