



COMSOL Multiphysics

Model Manager Reference Manual

Model Manager Reference Manual

© 1998–2022 COMSOL

Protected by patents listed on www.comsol.com/patents, or see Help>About COMSOL Multiphysics on the File menu in the COMSOL Desktop for less detailed lists of U.S. Patents that may apply. Patents pending.

This Documentation and the Programs described herein are furnished under the COMSOL Software License Agreement (www.comsol.com/sla) and may be used or copied only under the terms of the license agreement.

COMSOL, the COMSOL logo, COMSOL Multiphysics, COMSOL Desktop, COMSOL Compiler, COMSOL Server, and LiveLink are either registered trademarks or trademarks of COMSOL AB. All other trademarks are the property of their respective owners, and COMSOL AB and its subsidiaries and products are not affiliated with, endorsed by, sponsored by, or supported by those trademark owners. For a list of such trademark owners, see www.comsol.com/trademarks.

Version: COMSOL 6.1

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. Other useful links include:

- Support Center: www.comsol.com/support
- Product Download: www.comsol.com/product-download
- Product Updates: www.comsol.com/support/updates
- 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

Part number: CM020017

C o n t e n t s

Chapter 1: Introduction

About the Model Manager	10
What Can You Do with the Model Manager?	10
Where Do I Access the Documentation?.	12
Overview of the Manual	14

Chapter 2: Model Manager Tools

Introduction	19
Adding Databases	21
The Add Database Window	22
New Local Database	22
Opening a Local Database	23
Connecting to a Server Database.	25
Backward Compatibility for Model Manager.	27
Databases in the COMSOL Modeling Environment	29
Opening Models from Databases.	29
Saving Models to Databases.	31
Saving Drafts of Models	36
Inserting Components, Materials, and Physics from Databases.	37
The Versions Window for the COMSOL Desktop Model	38
Comparing Models Saved in Databases	40
Selecting Files in Databases as Input Sources	42
Selecting Files in Databases as Output Targets.	43
Loading and Saving Auxiliary Data Files Stored in Databases	44
The Auxiliary Data Window for Database Input and Output	46

The Model Manager Workspace	48
Opening the Model Manager Workspace	48
The Home Toolbar	48
The Database Toolbar	50
The Model Manager Workspace Windows	51
Overview of a Model Manager Database	53
Models	53
Files	58
Tags	61
Items	62
Commits	63
Branches	64
Snapshots	66
Locations	67
Repositories	67
Users	68
Groups	69
Permission Templates	70
Browsing Databases	72
The Model Manager Window	72
The Databases Window	75
The Settings Window	77
The Commits Window	79
Activating a Database	82
The Versions Window	82
The References Window	85
Opening Models	89
Running Applications	90
Previewing Files	91
Comparing Models	91
Copying Model and File Locations	91
Organization of Models and Files	94
Tagging Models and Files	94
Organizing Items in Repositories	96

Basic Version Control	98
Saving Versions	98
Adding and Removing Tag Assignments	100
Deleting Items	101
Recording Snapshots	103
Bulk Operations	104
Importing Files	104
Exporting Items	106
User Management	108
Managing Users	108
Managing Groups	109
Access Control	111
Owners	111
Granting Permissions	112
Permission Catalog	114
Permission Levels	117
Reusing Permission Assignments Using Permission Templates	119
Maintenance	121
Built, Computed, and Plotted Data	121
Permanently Deleting Models and Files	122
The Maintenance Window	123
Database Administration	127
Database Configurations	127
Updating Search Index	128
Database Cleanup	129
Compacting Local Databases	129
Moving and Deleting Local Databases	129
Backup for Local Databases	130

Chapter 3: Searching and Filtering

Searching Versions	132
Searching in Branches	132
Searching in Snapshots and Commits	133
Full Text Search	134
Combining Full Text Search Words	134
Matched Fields	135
Item and Content Filters	137
Field Types	137
The Filter Dialog Box	139
Item Filters	140
Content Filters	143
Applied Filter Pills	146
The Model Manager Search Syntax	147
Basic Field Expressions	147
Combining Expressions	152
Searching Nodes and Settings in the Model Tree	152
Search Syntax Catalog	155

Chapter 4: Advanced Version Control

Branching	160
The Branch as a Sequence of Commits	160
Creating a New Branch	161
Merging	165
Merging Changes to a Target Branch	165
The Merge Window	166
Reverting	169
Reverting Changes on a Branch	169

The Revert Window	169
-----------------------------	-----

Chapter 5: Working with Models in Databases

Example: Modeling Using Version Control	174
Creating the Database	174
Model Wizard Setup	175
Saving a First Version	176
Saving More Versions	177
Working With a Draft of the Model	179
Comparing Versions	184
Excluding Built, Computed, and Plotted Data	185
Importing Auxiliary Data to the Database	186
The Model Manager Workspace	188

Example: Browsing, Organizing, and Searching Models and Data Files	190
Downloading the Demo Database for Model Manager.	190
Searching and Browsing the Demo Database	191
Using a Tag Tree for Organization and Retrieval in a Database	196
Creating and Assigning Tags.	201
Searching on Model Contents	206
Using the Model Manager Search Syntax	210

Example: Using Advanced Version Control Tools in the Model Manager	212
Using a Draft to Update a Single Model	212
Working with Commits	217
Using a Branch to Update Many Models	222

Chapter 6: Glossary

Glossary of Terms	232
--------------------------	------------

Introduction

Read this guide to learn how to use the *Model Manager*, a set of tools for helping you with version control of models and data files. The Model Manager is available directly from the COMSOL Desktop® and includes comprehensive functionality for saving, searching, organizing, and sharing models and data files stored in a Model Manager database. A database can either be created locally on your computer for personal use or you can set up a server database, hosted by a *Model Manager server*, for collaborative use.

In this chapter:

- [About the Model Manager](#)
- [Overview of the Manual](#)

About the Model Manager

In this section:

- [What Can You Do with the Model Manager?](#)
- [Where Do I Access the Documentation?](#)

What Can You Do with the Model Manager?

When working with a simulation model for an extended period of time, you will inevitably have a need to keep a backup of old file versions around. You may, for example, want to recover an older version in case your current modeling work goes astray, or perhaps you want to use an older version as a template for a completely new model. Your solution to this may vary from something as straightforward as saving files with different filenames on your local hard drive, saving to a file-based version control system, or by uploading files to a product lifecycle management (PLM) system provided by your organization. If you are working on models in a collaborative setting, you are also used to sharing files with your coworkers, for example via email, by placing them on a shared file system, or by allowing your coworkers to download them from a centralized version control system or PLM system.

As the amount of simulation models and data grows, you and your coworkers might find yourselves spending a large part of your time managing these models and data. This may involve working with multiple tools and software — keeping you away from your main modeling and simulation work. Some of the challenges and concerns you might face are:

- Efficient storage of models. Store only relevant data for the purpose of archiving and future retrieval, all while keeping disk space usage manageable.
- Automatic extraction of model metadata. Extract metadata when saving simulation models so that you and others may later find them, without requiring manual data entry for keywords and other search terms.
- Reuse of models. Use previously saved simulation models and data as building blocks when creating new models, perhaps generating an extensive library of reusable parts.
- Track and compare changes to models over time. Get an automatic audit trail for your simulation models to compare and restore older versions, or to reproduce modeling steps.

- Manage relationships between models and data files. Find out which simulation models use a particular data file as input or what simulation model generated a particular data file as output.
- Access control. Control who can find, open, and save simulation models and data.

The Model Manager comes with a set of tools for addressing these points — all while staying within the COMSOL Desktop modeling environment. From the COMSOL Desktop, you can create a new database on your own computer to keep track of your private models and data files, or you and your colleagues may share such models and files by uploading them to a server database accessed via a Model Manager server.

A Model Manager database is tailor-made for the storage needs of a model built in COMSOL Multiphysics®. The Model Manager makes sure to never store duplicates of simulation data when saving multiple versions of the same model. You can also avoid storing built, computed, and plotted data that may instead be reproduced from the model when needed.

The powerful Model Manager search syntax enables you to search deep into models based on their properties, features, settings, and other metadata. You may, for example, perform search queries answering:

- Which models use a Time Dependent study step?
- Which models have a Length parameter between 5 cm and 15 cm?
- Which models were last modified by me?

The Model Manager also comes with standard version control tools such as viewing the version history of models and data files, automatically detecting version conflicts when saving, and comparing versions with each other. You can, for example, open an older model version to create a completely new model with its own split-off version history, or see all the changes made to a model from one version to the next.

You can do exploratory work on an existing model in a database by creating a draft of the model. The draft is version-controlled in its own right, enabling you to experiment with various simulation ideas without polluting the version history of the original model. Once you have finished your draft work, you may choose to either keep it as a new version of the original model or discard it.

More advanced version control tools such as branching, merging, and reverting are also available. Branching enables you, for example, to work on an entire collection of models and data files in isolation, while at the same time postponing the decision whether or not your changes are worth preserving. Reverting enables you, for

example, to restore models and data files that you have previously deleted, perhaps by accident.

When using a server database accessed via a Model Manager server, you can control who can access models and data files by setting permissions. You can, for example, set which users are permitted to open or save a particular model, or set which users are permitted to search and browse a collection of models. You can also use the web-based asset management system included with a Model Manager server installation to link your models and simulation results to various documents, presentations, project notes, slides, and other supplementary files and metadata. Using the asset management system is also a simple way of sharing such files and metadata with people in your organization who may not have access to the COMSOL Multiphysics software — all while keeping everything in the same database storing your models.

Where Do 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/support/updates
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/video
Support Knowledge Base	www.comsol.com/support/knowledgebase

Overview of the Manual

This *Model Manager Reference Manual* contains information that helps you get started with the Model Manager in the COMSOL Multiphysics product. The information in this guide is specific to this functionality. Instructions on how to use COMSOL in general are included with the *COMSOL Multiphysics Reference Manual*.



As detailed in the section [Where Do I Access the Documentation?](#), this information can also be searched from the COMSOL Multiphysics software **Help** system.

Instructions on how to install, configure, and administer a Model Manager server is found in the *Model Manager Server Manual*. That manual also contains information about the asset management system included with a Model Manager server installation.

TABLE OF CONTENTS AND INDEX

To help you navigate through this guide, see the [Contents](#) section and [Index](#).

TOOLS

The [Model Manager Tools](#) chapter has an overview of the tools available in the COMSOL Desktop and includes information about [Adding Databases](#), [Databases in the COMSOL Modeling Environment](#), and [The Model Manager Workspace](#).

SEARCH

The [Searching and Filtering](#) chapter gives a detailed description of the search and filtering capabilities of a Model Manager database, including an overview of [The Model Manager Search Syntax](#).

VERSION CONTROL

The [Advanced Version Control](#) chapter introduces more advanced tools for version control management in the Model Manager, including [Branching](#), [Merging](#), and [Reverting](#).

EXAMPLE TUTORIALS

The [Working with Models in Databases](#) chapter showcases how the Model Manager tools can be integrated into your modeling workflow by way of a few example tutorials.

GLOSSARY

The [Glossary](#) chapter gives a summary of various concepts and terms specific to a Model Manager database.

Model Manager Tools

This chapter provides an overview of the tools available in the Model Manager that enables you to save, search, organize, and share models and data files stored in a Model Manager database. To quickly get started with models in databases, see the tutorial [Example: Modeling Using Version Control](#).

In this chapter:

- [Introduction](#)
- [Adding Databases](#)
- [Databases in the COMSOL Modeling Environment](#)
- [The Model Manager Workspace](#)
- [Overview of a Model Manager Database](#)
- [Browsing Databases](#)
- [Organization of Models and Files](#)
- [Basic Version Control](#)
- [Bulk Operations](#)
- [User Management](#)
- [Access Control](#)

- [Maintenance](#)
- [Database Administration](#)

Introduction

The Model Manager includes an extensive set of tools for working with version-controlled models and data files stored in databases. These tools are integrated into the COMSOL Desktop modeling environment (the Model Builder, the Application Builder, and the Physics Builder), enabling you to seamlessly switch between models and data files stored on the file system and in a Model Manager database.



In this, and in many other chapters, the term *model* will be used as a catch-all that also includes applications and physics. The term *data file* is a catch-all for other types of files that models may depend on as input or generate as output — including, for example, CAD data, interpolation functions, plots, and reports. See the [Glossary](#) for more details.

Some of the key Model Manager tools that you will encounter in your everyday modeling workflow are:

- Opening a model from a database from the **Open** window. See [Opening Models from Databases](#) to learn how you can search for models to open from databases.
- Saving a model to a database from the **Save** window. See [Saving Models to Databases](#) to learn how you can save your modeling work as a model version to a database.
- Reusing components, materials, and physics from other models via the **Select Model** window. See [Inserting Components, Materials, and Physics from Databases](#) to learn how you can copy such contents from models stored in databases into the model currently opened in the COMSOL Desktop.
- Accessing older versions of a model from the **Versions** window. See [The Versions Window for the COMSOL Desktop Model](#) to learn how you can access previously saved versions of your model, including opening a version, restoring a version, or comparing two versions with each other.
- Selecting data files stored in a database as sources for input or targets for output in the **Select File** window. See [Selecting Files in Databases as Input Sources](#) to learn how you can import data into a model from a data file stored in a database. See [Selecting Files in Databases as Output Targets](#) to learn how you can export data from a model to a data file stored in a database.

You can quickly get started with the Model Manager by creating a new database stored locally on your computer. See [Adding Databases](#) and [New Local Database](#) to learn how you can create such a database directly from within the COMSOL Desktop.



The Model Manager is supported on the same platforms as COMSOL Multiphysics®: Windows®, Linux®, and macOS.

If you have installed a Model Manager server, you can also connect to its server database from within the COMSOL Desktop — see [Connecting to a Server Database](#). Read more about installing, configuring, and administrating a Model Manager server in the *Model Manager Server Manual*.

The Model Manager comes with a dedicated workspace for database-specific tools such as browsing, organizing, and administrating your databases. See [The Model Manager Workspace](#) and further sections in this chapter to learn, for example, how you can browse and administer your configured databases in [The Databases Window](#), search for models and data files in [The Model Manager Window](#), and view settings, features, properties, and other metadata of your saved models in [The Settings Window](#).



The Model Manager tools are default enabled in the COMSOL Desktop environment. You can hide all Model Manager functionality via the **Preferences** dialog box by clearing the **Enable Model Manager** check box on the **Model Manager** page. A program restart is required.



Adding Databases

The COMSOL Desktop supports connecting to both a *local database* stored on the same computer that COMSOL Multiphysics is running on, as well as to a *server database* accessed via a Model Manager server. You can connect to as many databases as you like, and COMSOL Multiphysics will remember connected databases between program sessions.

A local database is intended for single-user use and is the recommended database when you want to keep your own models and data files under version control, without necessarily intending for others to see and work with them. It can also be useful when you just want to try out the various features that Model Manager offers. A server database is intended for multiple-user use and is the recommended database when you are working on models and data files in a collaborative setting.

Support for creating local databases is included with the COMSOL Multiphysics installation and requires no additional modules, software, or running processes. A server database requires a running Model Manager server, which can either be started on the same computer as COMSOL Multiphysics or on another computer that COMSOL Multiphysics has network access to.

To add a new database, do one of the following:




- In the **Open** window, the **Save** window, the **Select File** window, the **Select Model** window, or the **Export** window, choose **Add Database** () from the list of options.
- In the Model Manager workspace, click the **Add Database** button () in the **Database** section on the **Home** toolbar.

In this section:

- [The Add Database Window](#)
- [New Local Database](#)
- [Opening a Local Database](#)
- [Connecting to a Server Database](#)
- [Backward Compatibility for Model Manager](#)

The Add Database Window

From the **Add Database** window:

- Click the **New Local Database** button () to create a new Model Manager database that will be stored locally on your computer.
- Click the **Open Local Database** button () to add an existing Model Manager database stored locally on your computer.
- Click the **Connect to Server Database** button () to connect to a server database via a Model Manager server.



- [New Local Database](#)
 - [Opening a Local Database](#)
 - [Connecting to a Server Database](#)
-

New Local Database


To create a new local database on your computer:

- 1 Under **Database name**, write the name of the new database as stored on the file system.

The database name can only contain characters that are valid for a directory name and filename. The name will also be used as the initial label of the database in the COMSOL Desktop, although this label can later be changed — see [Database Configurations](#).

- 2 Under **Databases directory**, write the path to the parent directory that will contain the new database. You can also keep the suggested, prefilled directory path.

The database itself will be created as a subdirectory inside this directory.

- 3 Click the **Add Database** button () to create the new database.



Creating a database on a network or cloud drive is not supported and COMSOL Multiphysics will report an error if it detects such a file system path. This includes placing the database in a local folder on your computer that is mapped to an external cloud drive via two-way synchronization. For more information, see <https://www.comsol.com/support/knowledgebase/1295>.



The field under **Databases directory** contains a prefilled default directory path appropriate for the underlying operating system. You can change this default from the **Preferences** dialog box in the **Directory for local databases** field on the **Model Manager** page.

A LOCAL DATABASE ON THE FILE SYSTEM

The created database consists of a directory with the name specified by the **Database name** field, located within the parent directory specified by the field under **Databases directory** in the **New Local Database** window. This database directory contains:

- An SQLite® database file with the same name as the directory. Its file extension is `mphdb`.
- A `resources` directory containing files whose file sizes are deemed too large to store directly inside the SQLite® database file. These files can, for example, be built, computed, and plotted data generated by models, or standalone data files such as CAD data, interpolation functions, and reports.
- An `indexes` directory containing files used by Model Manager for searching and filtering. The file contents in this directory is automatically created by Model Manager using the data stored in the SQLite® database file and, as a result, may be safely skipped in backups.



The database directory also contains an `index` directory used for backwards compatibility when the database is accessed from a COMSOL Multiphysics 6.0 installation. This directory and its file contents is created automatically as needed and may be safely skipped in backups.



[Backup for Local Databases](#)

Opening a Local Database

You can add an already created database to the COMSOL Desktop. This is useful, for example, if you want to move a local database created on another computer to your current computer or if you want to restore a database whose database directory has previously been backed up by external backup software.

From the **Open** dialog box, browse to and choose the SQLite[®] database file for the database you want to add. Click the **Open** button to add the database.

Model Manager will check that it can connect to the SQLite[®] database file, as well as check that the `resources` directory and the `indexes` directory for the database, as described in [A Local Database on the File System](#), are found next to the SQLite[®] database file. If the `indexes` directory cannot be found, Model Manager will recreate the directory and its file contents using the data stored in the SQLite[®] database file. If all checks succeed, the Model Manager database is added to the COMSOL Desktop. Otherwise, the **Open Local Database** window is shown.

- 1 Write the path to the SQLite[®] database file under **Database file**.
- 2 Write the path to the `resources` directory under **Resources directory**.
- 3 Write the path to the `indexes` directory under **Search indexes directory**.
- 4 Write a label for the database under **Label**.
- 5 Click the **Add Database** button () to add the database to the COMSOL Desktop.



Adding a database stored on a network or cloud drive is not supported and COMSOL Multiphysics will report an error if it detects such a file system path. This includes placing the database in a local folder on your computer that is mapped to an external cloud drive via two-way synchronization. For more information, see <https://www.comsol.com/support/knowledgebase/1295>.

OPENING A LOCAL DATABASE FROM MULTIPLE COMSOL MULTIPHYSICS PROCESSES

You can access the same local database from multiple COMSOL Multiphysics program sessions as long as these program sessions run on the same computer that stores the database. There are, however, a few caveats to keep in mind:

- Changes saved from one COMSOL Multiphysics program session may not be immediately visible to other program sessions.
- Some advanced search and filter functionality, as described in [Searching and Filtering](#), will only be available to the first program session that connects to the local database. Other program sessions use a simplified search.

Connecting to a local database from multiple COMSOL Multiphysics program sessions running on different computers by, for example, trying to place the local

database on a network drive or a cloud drive, or use some other type of sharing or synchronization tool is not supported.



Use a server database accessed via a Model Manager server when working in a multiple-user or multiple-computer environment.

Connecting to a Server Database

You can connect to a server database via a Model Manager server running either locally on your computer or on a server computer in your organization's internal network.



See the *Model Manager Server Manual* for instructions on how to install, configure, and administer a Model Manager server.

- 1 Write the server address to the Model Manager server under **Server**.
- 2 Select the **Require secure connection** check box if the network connection is made using a secure connection, with transport layer security provided by HTTPS (as opposed to plain HTTP). A warning message is shown if the check box is cleared.
- 3 Write your user credentials used to authenticate with the Model Manager server under **User**. You can opt to remember the provided password between program sessions by selecting the **Remember password** check box. The password will be stored in an encrypted form on the local file system.
- 4 Write a label for the server database under **Label**.
- 5 Click the **Connect** button () to connect to the Model Manager server.




You are strongly recommended to connect to the Model Manager server using a secure connection, with transport layer security provided by HTTPS. Clearing the **Require secure connection** check box will send all data, including your credentials, in an unencrypted cleartext format. See the *Model Manager Server Manual* for configuring a Model Manager server to use secure connections.



To connect to a Model Manager server running on, for example, port 8181 on the same computer as COMSOL Multiphysics, write `localhost:8181` under **Server**.

CONNECTING VIA A COMSOL MULTIPHYSICS SERVER

When a COMSOL Multiphysics client is connected to a COMSOL Multiphysics server running on another host computer, the communication is typically over a nonsecure channel. This means that any credentials written in the user interface on the client computer would be sent to the COMSOL Multiphysics server computer in an unencrypted cleartext format. For this reason, Model Manager does not allow writing a password in the **Connect to Server Database** window when running in client-server mode unless it detects that the client-server connection is over a secure channel. You are instead met with a message in the **Connect to Server Database** window informing you that a secure password prompt will be shown when connecting to the database.

- 1 Fill out the connection details as you would when connecting from a standalone COMSOL Multiphysics program session, except you will not provide a password.
- 2 Click the **Connect** button ().

A **Progress** dialog box is shown informing you to provide credentials for the Model Manager server in the COMSOL Multiphysics server's console window.

- 3 In the console window for the started COMSOL Multiphysics server process, press Enter in order to provide your credentials. Write your username and password when prompted. You can also choose to remember your password for the Model Manager server, in which case the password will be stored in an encrypted form on the local file system of the COMSOL Multiphysics server computer.

When you have finished providing your credentials, the **Progress** dialog box is automatically closed and a connection to the Model Manager server is established.



To avoid having to provide your credentials in the COMSOL Multiphysics server's console window, either tunnel the connection between the COMSOL Multiphysics client and the COMSOL Multiphysics server using a secure SSH connection, or first provide the credentials with **Remember password** selected using a local COMSOL Multiphysics instance running on the COMSOL Multiphysics server computer.

TROUBLESHOOTING

The following are some common error messages when connecting to a Model Manager server and suggestions on how to address them:

The server address could not be resolved. The server address in the **Server** field does not match the address of any server in your organization's internal network. Verify that you have written the correct server address. Alternatively, if you are able to log in to the web interface of the Model Manager server using a web browser, click on the COMSOL logo in the upper-left corner to make sure you are on the start page, and then copy the web page's link address from the web browser's address field. That link address should then be pasted into the **Server** field.

Connection timed out/Connection refused. This typically happens when the server computer was found in your organization's internal network but the Model Manager server itself is currently not running on that server computer. It can also happen if you have missed providing, or provided the wrong, port number for the Model Manager server.

The current password is temporary and must be changed. You are trying to log in using the temporary password set for the default administrative user account created during installation of the Model Manager server. First log in to the Model Manager server's web interface using a web browser and change the temporary password when prompted.

Database configuration is not activated. The default server database has been deactivated by an administrator of the Model Manager server. The server database needs to be activated again via the Model Manager server's web interface before you can connect from COMSOL Multiphysics.

No default database configured. The administrator of the Model Manager server has not set a default database for the server. One of the configured server databases has to be set as the default one via the Model Manager server's web interface before you can connect from COMSOL Multiphysics.

Backward Compatibility for Model Manager

When COMSOL Multiphysics connects to a local Model Manager database created using an older COMSOL Multiphysics version, 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 versions of COMSOL Multiphysics will still be able to load models and data files from the database even after this upgrade — an important use case when you, for example, need

to recompute a database-stored model using the exact same version of COMSOL Multiphysics the model was initially solved in.



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

When you connect from a newer version of COMSOL Multiphysics to a server database hosted by an older version of Model Manager server, the Model Manager server will not be aware of any newer functionality supported by COMSOL Multiphysics. In this case, such functionality will either be hidden or disabled in the COMSOL Desktop environment.


The opposite case is similar: When you connect from an older version of COMSOL Multiphysics to a server database hosted by a newer version of Model Manager server, you will not have access to all functionality supported by the newer server from the COMSOL Desktop environment. Same as for local databases, you will still be able to load older models and data files from the server database though.



Databases in the COMSOL Modeling Environment


The Model Manager tools are integrated with the COMSOL Desktop modeling environment — wherever you interact with models and data files on the file system, there is typically equivalent functionality for interacting with such models and data files in a database. In this section, you will find details on where and how you typically encounter the Model Manager tools in the COMSOL Desktop.

- [Opening Models from Databases](#)
- [Saving Models to Databases](#)
- [Saving Drafts of Models](#)
- [Inserting Components, Materials, and Physics from Databases](#)
- [The Versions Window for the COMSOL Desktop Model](#)
- [Comparing Models Saved in Databases](#)
- [Selecting Files in Databases as Input Sources](#)
- [Selecting Files in Databases as Output Targets](#)
- [The Auxiliary Data Window for Database Input and Output](#)

Opening Models from Databases

From the **Open** window, you can find and open versions of models from one of your configured databases (see [Adding Databases](#)). Choose the database that you want to open a model from in the list of options on the left. Choose **Add Database** () if you want to add a new database.

The **Open** window is shown with a list of models saved in the database. Select a model and click the **Open** button () to open the model in the COMSOL Desktop. If a model is also an application, you can click **Run** () to launch and run the application directly.

You can also right-click a model to open it or, if the model is also an application, run it. Select **Set Tags** () to modify the tag assignment of a model — see also [Assigning Tags to Items](#).



The Open Window in the *COMSOL Multiphysics Reference Manual*.




- [Opening Models](#)
 - [Running Applications](#)
-

FINDING MODELS TO OPEN

You find models to open by writing search expressions in the search field and clicking the **Search** button. 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, description, assigned tags, and filename of a model.

You can also apply separate [Item and Content Filters](#) via [The Filter Dialog Box](#). Select a filter from the **Add Filter** menu button () in the toolbar to open the dialog box. Applied filters are shown below the search field — see [Applied Filter Pills](#).



You search for the latest versions of models by default. Each entry in the result list contains the title of the model in that version, the time when the version was saved, the name of the user that saved the version, and the tags assigned to the model. The search result is sorted on title, with a maximum of 100 models initially included in the result list. Click the **Show More** button () to append more matches to the list.



[Searching and Filtering](#)

THE OPEN WINDOW TOOLBAR

The toolbar above the search field contains the following toolbar buttons:

- Click the **Refresh** button () to refresh the search result while keeping the search expression and applied filters unchanged.
- Click the **Show More** button () to append more matching models to the search result.

- Click the **Reset** button (↶) to clear the current search expression and applied filters.
- Click the **Add Filter** menu button (➤) to apply a filter.

SELECT LOCATION IN DATABASE

You can change which model versions to search for by selecting another *location* — see also [Searching Versions](#) and [Locations](#). Click the link button above the search field to open [The Select Location Dialog Box](#) to select which location to search. The link button is hidden if there is only one location available in the database, which is the default for a new database.

Saving Models to Databases

From the **Save** window, you can save a new version of the model opened in the COMSOL Desktop to one of your configured databases. Choose the database that you want to save a model to in the list of options on the left. Choose **Add Database** (➤) if you want to add a new database.



[The Save Window](#) in the *COMSOL Multiphysics Reference Manual*.

The **Save** window can appear with four different headers depending on the model's presence in the selected database:

- **Save new:** The opened model has never before been saved to the database. A new model will be created in the database with an associated first version.
- **Save version:** The database contains previously saved versions of the opened model. A new version of an existing model will be saved to the database.
- **Save version from draft:** The opened model is a draft version. A new version of the regular model that the draft was originally created from will be saved to the database.
- **Save new from draft:** The opened model is a draft version, but the regular model that the draft was originally created from does not exist in the database. This may happen if you, or another user, permanently deletes the regular model in the database while the draft is open in the COMSOL Desktop, or if you decide to save the draft to

another database. As for **Save new**, a new model will be created in the database with an associated first version.



Saving Drafts of Models

To save a new model version:

- 1 Write the title for the model version in the **Title** field.


This title is kept in sync with the corresponding field in the **Presentation** section in the root node's **Settings** window. The title cannot be left empty.



- 2 Write an optional save comment in the **Comments** field.

You can later read and update this save comment in [The Versions Window](#) or the [The Commits Window](#).

- 3 Change the target branch for the save by clicking the **Location** link button and selecting another branch in [The Select Location Dialog Box](#).

The **Location** link button is hidden if the database only contains a single branch, which is the default for a new database.

- 4 Click the **Save** button () to save a new version of the model.

If you want to force the creation of a new model instead, click the expand button next to the **Save** button () and select **Save as New** () — see [Splitting a Model Version History in Two](#).





Only changing the title of a model already saved in a database does *not* mean that you will create a *new* model. While providing a descriptive title helps you with later finding it in the database, it has no bearing on the database identity of the model.

THE INFORMATION SECTION

The **Information** section in the **Save** window displays additional information concerning the save. When saving a new version of an existing model in the database, this includes both when, and by whom, various versions were saved:

- **Latest version:** The most recent version of the model being saved. The latest version depends on the target branch set in the **Location** field.



- **Current version:** The version that the current model in the COMSOL Desktop was opened from. Only shown for **Save version** if the current version is not the same as the latest version — see [Save Conflicts](#).
- **Current draft version:** The version that the current draft in the COMSOL Desktop was opened from. Only shown for **Save new from draft**.

The panel can also contain information messages () that may be of interest, as well as warning messages () signaling, for example, that the opened model is in conflict with the latest version.

Save Conflicts

When the **Save** window is shown, Model Manager will make a preemptive check that the model to be saved is not in conflict with the latest, already saved, version in the database. Such a conflict can arise, for example, if:

- The latest model version saved in the database is not the same version that the model in the COMSOL Desktop was opened from. Your coworker may, for example, have opened the same model as you, made some changes, and then saved before you now try to save it.
- The latest model version saved in the database is not the same version as a draft was originally created from. You and your coworker may, for example, have both started working on two separate drafts, and then your coworker saved their draft back to the original model before you now try to save it back to the same model. See [Saving Drafts of Models](#) for more details.
- The model in the COMSOL Desktop has been deleted on the target branch before you now try to save it.

Clicking the **Save** button () while there are save conflicts opens a dialog box asking if you want to save anyway. Click **Save** to ignore all conflicts, or **Cancel** for closing the dialog box without saving. You can also click **Compare with Latest** () to open the **Comparison Result** window to compare the opened COMSOL Desktop model with the latest version in the database. See [Comparing the Opened Model in the COMSOL Desktop With the Latest Version](#) for further details.


THE DESCRIPTION SECTION

The **Description** section in the **Save** window shows the current description of the model. As for the **Title** field, changes to the description are kept in sync with the corresponding field in the **Presentation** section in the root node's **Settings** window.



Use the **Description** field for information that should carry over between different versions of the model. Use the **Comments** field for information that only applies to the specific model version being saved.

THE TAGS SECTION


The **Tags** section shows the tags assigned to the model as a collection of *tag pills*. You can assign tags already present in the database, as well as create new tags that should be assigned. You can assign as many tags as you like. Click a tag pill and select **Remove** () to remove a tag assignment.



Any changes made to the assigned tags in the **Tags** section are saved to the database first when you save the model.




See [Tagging Models and Files](#) to learn how tags may help you in organizing your models and data files in a Model Manager database.

Win

Click the **Add Tag** button () to find an existing tag to assign to the model. Write the title of the tag in the text field to filter the popup list of available tags. The matching tags are shown with their title followed by the titles of their parent tags, if any, within parentheses. Either double-click a tag in the list or select a tag and press Enter to assign it to the model.

Click the **Set Tags** button () to open the **Set Tags** dialog box. The dialog box contains a tag tree of all available tags. Select a check box for a tag in order to assign the tag to the model. Clear a selected check box to remove a tag assignment. Click **Clear Tags** () to clear all selections. You can filter the tree of available tags by writing a tag title in the text field above the tree. Click **OK** to finish the tag assignment.

If the tag you want to assign is not present in the database, you can create a new tag from the **Tags** section. Click the **New Tag** button () to open the **New Tag** dialog box.

I In the **Title** field on the **General** tab, write the title for the new tag.

- 2 Select the tags that the new tag itself should be assigned — in other words, its *parent tags* — in the tag tree on the **Parent tags** tab. You can filter the tree of available parent tags by writing a tag title in the text field above the tree.
- 3 Write an optional save comment for the created tag in the **Comments** field.
- 4 Click **OK** to create the tag in the database.



Unlike the other tag operations in the **Tags** section, the new tag is immediately saved to the database when you click **OK** in the **New Tag** dialog box. The assignment of the new tag to the model is, however, first saved when the model is saved.



You may want to double-check that a tag with the same title is not already present in the database before creating a new tag as Model Manager will not prevent such duplicates.

THE AUXILIARY DATA SECTION

The **Auxiliary Data** section is shown in the **Save** window if the model references any auxiliary data files that are version-controlled in a Model Manager database and those data files currently have unsaved changes in the current COMSOL Multiphysics program session. Saving the model gives you the opportunity of simultaneously saving these data files to the same target branch that the model is saved to.

The section contains a table with the following columns:

- The **Save** column — a check box that, if selected, specifies that a new version of the data file should be saved when the model is saved.
- The **Title** column — the title of the data file. You can change the title by editing directly in the table cell.
- The **Files** column — the number of files to be saved. May be more than one in the case of a fileset — see [Files](#).
- An icon column — shows an information icon when a previous version of the data file is not already present in the target branch, or a warning icon if there are version conflicts for the data file. The icon's tooltip gives further details.

Select a row in the table and click **Details** () to see further details on an auxiliary data file.



- [Loading and Saving Auxiliary Data Files Stored in Databases](#)
- [The Auxiliary Data Window for Database Input and Output](#)

PERMISSION CHECK



When saving to a server database via a Model Manager server, the Model Manager server will check that you fulfill the permission requirements to save a new version of the model. This involves checking that you are permitted to:

- Save in the repository containing the target branch.
- Save in the target branch.
- Save versions of the model.


Only the first two requirements apply when creating a new model in the database. If there is any auxiliary data that will be saved with the model, a permission check will also be made for saving versions of these items.




Permission Levels

A preemptive permission check is performed when the **Save** window is opened. If the check fails, a link button () that opens a dialog box explaining why it failed is shown next to the **Save** button () — see [The Permission Requirements Dialog Box](#).

Saving Drafts of Models

To save a *draft* of the model opened in the COMSOL Desktop, go to the **File** menu and select **Save Draft** (). You can also press the keyboard shortcut Ctrl+S.








The **Save Draft** () option is only available if the model is opened from, or last saved to, a database.



The keyboard shortcut Ctrl+S either saves the current model to the file system, or as a draft in the database, depending on where the model was last saved. The file system is the default for unsaved models.

A draft model behaves in exactly the same way as a *regular* model in the database, except that a draft offers a streamlined way of saving it back as a new version of the model it originated from.

Working with a new draft model in the COMSOL Desktop typically involves these steps:


- 1 Open a regular model from a database.
- 2 From the **File** menu, select **Save Draft** () , or press Ctrl+S, to save a first version of a new draft of the opened model.
- 3 Work on the draft, intermittently selecting **Save Draft** () or pressing Ctrl+S to save additional draft versions.
- 4 From the **File** menu, select **Save as Version** (). Click **Save** () to save the opened draft as a new version of the original model. Click **Save as New** () to create a new model from the draft. See [Saving Models to Databases](#).

You can also open an existing draft model from the **Open** window — perhaps in order to continue working on a draft created in a previous COMSOL Multiphysics program session.





Drafts


Inserting Components, Materials, and Physics from Databases

From the **Select Model** window, you can find and select models stored in a database to insert one of their components, physics, or materials into the model opened in the COMSOL Desktop. Choose the database that you want to select from in the list of options on the left. Choose **Add Database** () if you want to add a new database.

The **Select Model** window is opened when you, for example, click:

- **Insert Components From** () in the **Add Component** menu in Model Builder's **Home** toolbar.



- **Insert Physics From** () in the **Insert Physics** menu in Model Builder's **Physics** toolbar.
- **Import Materials From** () in Model Builder's **Materials** toolbar.


The **Select Model** window for a database offers identical search and filter functionality as the **Open** window — see [Opening Models from Databases](#). Click the **Select** button () once you have found and selected the sought-after model to insert from.

The Versions Window for the COMSOL Desktop Model

The **Versions** window in the **Model Builder** workspace shows the history of the model opened in the COMSOL Desktop when that model is opened from a database or was last saved to a database. You can use the **Versions** window to, for example, quickly get an overview of recently saved versions, compare what was changed between two versions, open older versions, or even restore an older version as the latest version.



From the **Windows** menu () in the **Layout** section of Model Builder's **Home** toolbar, select **Versions** () to open the **Versions** window.








The window contains a table with versions of the model sorted in chronological order, most recent first. At most 100 versions are initially shown. Click the **Show More** button () to append older versions to the table. The version that is opened in the COMSOL Desktop is highlighted in bold.

The table columns are:



- The type column — the type of the model represented by an icon. See [Item Types](#) and [Save Types](#).
- The **Title** column — the title set for the model in that version.
- The **Saved** column — the time when the version was saved.
- The **Saved By** column — the name of the user that saved the version.
- The **Branch** column — the target branch the version was saved in. There is only a single branch, default named **Main**, when creating a new database.
- The **Comments** column — the optional comment provided when the version was saved.

THE VERSIONS WINDOW TOOLBAR


The toolbar above the table contains the following toolbar buttons:

- Click the **Refresh** button () to refresh the table in case a new version has been saved. The table will automatically refresh if you save a new version to the database from the COMSOL Desktop.
- Click the **Show More** button () to append older versions to the table.
- Click the **Version Details** button () to open [The Version Details Dialog Box](#) containing more information on a specific version.
- Click the **Open** button () to open a selected version in the COMSOL Desktop.
- Click the **Run** button () to launch and run a selected version in the COMSOL Desktop. Only enabled if the selected version is an application.
- Click the **Compare** button () to compare a selected version with the model opened in the COMSOL Desktop. Select two versions to compare them with each other. See [Comparing Models Saved in Databases](#).
- Click the **Restore Version** button () to save the selected version as a new latest version of the model. See [Restore Version](#) for further details.

If you right-click a model version, you can also:

- Select **Copy Location** () to copy a text string with a URI that uniquely identifies the model version in the database to the clipboard. See [Copying Model and File Locations](#).
- Select **Permanently Delete Version** () to permanently delete the model version in the database. This cannot be undone. See [Permanently Deleting a Version](#) for further details.

SPLITTING A MODEL VERSION HISTORY IN TWO

When you create a new model from an existing model, for example by saving a new draft or selecting **Save as New** () in the **Save** window, the Model Manager database stores a reference to the *origin model* from the new model (or rather, the version of the original model that was saved from). You can think of the new model as being *split off* from the original model, such that the new model receives its own identity and version history.

The **Versions** window helps you keep track of a model's potential origin by including the versions of the latter up to the source version that the new model was created from in the window's table — see [Figure 2-1](#) for a schematic representation.

You are likely to first encounter this split-off in the **Versions** window when saving a new draft from a regular model — see [Saving Drafts of Models](#). The top table rows correspond to the versions of the saved draft. The remaining table rows are the versions of the regular model that the draft originated from.

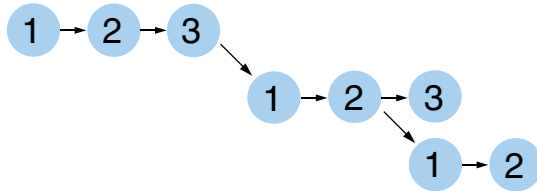


Figure 2-1: A schematic of the version history of three models, with the chronological order of versions read from left to right. The first version of the middle model was created from the third version of the top model, and the first version of the bottom model was created from the second version of the middle model. The version history of the top model includes three versions, that of the middle model includes six versions, and that of the bottom model includes seven versions. The history of the bottom model does not include the third version of the middle model as that was saved after the bottom model was split off.

Comparing Models Saved in Databases


The **Comparison Result** window enables you to compare versions of models stored in databases. In this section, you will see four different ways you can encounter comparisons involving such models:

- [Comparing Two Versions From a Model’s History](#)
- [Comparing Two Models](#)
- [Comparing a Version With the Opened Model in the COMSOL Desktop](#)
- [Comparing the Opened Model in the COMSOL Desktop With the Latest Version](#)




Comparing Models and Applications in the *COMSOL Multiphysics Reference Manual*

COMPARING TWO VERSIONS FROM A MODEL’S HISTORY


If you select two model versions in the **Versions** window and click **Compare** () , the **Comparison Result** window will open with the older version shown on the left, labeled as **Older Version**, and the newer version on the right, labeled as **Newer Version**. The versions need not belong to the same model: the older version may, for example,


belong to an origin model of the newer version's model — see [Splitting a Model Version History in Two](#). Since model versions are immutable in the database, it is not possible to merge or override any shown differences in the opened **Comparison Result** window.


COMPARING TWO MODELS

If you select two models in [The Model Manager Window](#) or [The Databases Window](#) and click **Compare** () , the **Comparison Result** window will open with one of the models labeled as **First** and the other as **Second**. The compared versions depend on the context where the models were selected. Selecting two models when [Searching in Branches](#) will compare the latest versions of each respective model. Selecting when [Searching in Snapshots and Commits](#) will compare the versions that were the latest at the time of the corresponding commit. Similar to when comparing two versions from a model's history, it is not possible to merge or override any shown differences in the opened **Comparison Result** window.

COMPARING A VERSION WITH THE OPENED MODEL IN THE COMSOL DESKTOP

If you select a single model version and click **Compare** () , the **Comparison Result** window will open with a comparison between the model opened in the COMSOL Desktop, labeled **Opened**, and the selected version, labeled **Saved**.

Right-click the top node in the **Differences** tree and select **Merge Changes to Opened** () to merge all changes marked as *incoming* (that is, changes to the model tree made in the selected model version) into the desktop model. All changes marked as *outgoing* (that is, changes made to the model tree in the desktop model) are left alone.


Right-click any node in the **Differences** tree and select **Override Difference in Opened** () to write all changes in that node to the opened model, regardless if any changes are incoming or outgoing, effectively reverting any changes you have made in the desktop model.

COMPARING THE OPENED MODEL IN THE COMSOL DESKTOP WITH THE LATEST VERSION



If you encounter a version conflict when trying to save a new version of the model opened in the COMSOL Desktop, you can open the **Comparison Result** window with a comparison between the desktop model, labeled **Opened**, and the latest version of the model being saved, labeled **Latest Version** — see also [Save Conflicts](#). Similar to [Comparing a Version With the Opened Model in the COMSOL Desktop](#), you can


merge changes and override differences into the desktop model to include those changes you want to keep from the latest version.



Selecting Files in Databases as Input Sources

Data files stored in a Model Manager database can be used as input for models in the same way you can use files stored on the file system as input. From the **Select File** window for an input setting, you can find and select files from one of your configured databases to use as input source for the model opened in the COMSOL Desktop. Choose the database that you want to select files from in the list of options on the left. Choose **Add Database** () if you want to add a new database.




The **Select File** window is, for example, opened when you click the expand button next to **Browse** () and select **Browse From** () for a model input setting in the **Settings** window in the Model Builder workspace.

The **Select File** window for a database offers similar search and filter functionality as the **Open** and **Select Model** windows, although adapted for searching data files. Click **Select** () once you have found and selected a file in the result list to use as input. If the selected data is a fileset, you will be asked to select one of the files via the **Select File from Fileset** dialog box.

You can also right-click a file in the list to select it as input. Select **Preview File** () to open the file with the default application for its file type — see also [Previewing Files](#). Select **Set Tags** () to modify the tag assignment of the file — see also [Assigning Tags to Items](#).





Selecting a data file from a Model Manager database via the **Select File** window and clicking **Select** () only specifies a source for the input. The actual loading of the input data from the database happens later — either automatically when the data is needed or manually by, for example, clicking an **Import** button. See [Loading and Saving Auxiliary Data Files Stored in Databases](#) for more details.

FINDING DATA FILES

You find data files by writing search expressions in the search field and clicking the **Search** button. 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,





description, assigned tags, and filename of a data file. The search can be restricted by an explicit file type filter set in the list next to the search field.

You can also apply separate [Item Filters](#) via [The Filter Dialog Box](#). Select a filter from the **Add Filter** menu button () in the toolbar to open the dialog box. Applied filters are shown below the search field — see [Applied Filter Pills](#).

You search for the latest versions of files by default. Each entry in the result list contains the title of the file in that version, the time when the version was saved, the name of the user that saved the version, and the tags assigned to the file. The search result is sorted on title, with a maximum of 100 files initially included in the result list. Click the **Show More** button () to append more matches to the list.

THE SELECT FILE WINDOW TOOLBAR

The toolbar above the search field contains the following toolbar buttons:


- Click the **Refresh** button () to refresh the search result while keeping the search expression and applied filters unchanged.
- Click the **Show More** button () to append more matching files to the search result.
- Click the **Reset** button () to clear the current search expression and applied filters.
- Click the **Add Filter** button () to apply a filter.

SELECT LOCATION IN DATABASE



You can change which file versions to search for by selecting another *location* — see [Searching Versions](#) and [Locations](#). Click the link button above the search field to open [The Select Location Dialog Box](#) to select which location to search. The link button is hidden if there is only one location available in the database, which is the default for a new database.

Selecting Files in Databases as Output Targets

Output data generated by a model can be stored as data files in a Model Manager database in the same way you can store such output as files on the file system. From the **Select File** window for an output setting, you can either specify a new file or select an existing file in one of your configured databases to use as output target for the model opened in the COMSOL Desktop. Choose the database that you want to select

files from in the list of options on the left. Choose **Add Database** () if you want to add a new database.



The **Select File** window is, for example, opened when you click the expand button next to **Browse** () and select **Browse From** () for a model output setting in the **Settings** window in the Model Builder workspace.

Click the **Select New** radio button in the **Select File** window for a database to specify a new file target. Write a title for the output data in the **Title** field. The **Filename** field will be automatically populated with a suggested filename for the main output file based on the title and the output's file extension.

Change the target branch for the output by clicking the **Location** link button and selecting another branch in [The Select Location Dialog Box](#). The **Location** link button is hidden if the database only contains a single branch, which is the default for a new database.

Click the **Select Existing** radio button to find and select an existing data file as output target. The window offers the same search and filter functionality as when selecting a data file from a database as input source — see [Selecting Files in Databases as Input Sources](#).

Click **Select** () once you have specified a file to use as target for the output.




Selecting a data file from a Model Manager database via the **Select File** window and clicking **Select** () only specifies a target for the output. The actual writing and saving of the output data to the database happens later — either directly by clicking, for example, an **Export** button or as a two-step procedure in which the output is first written to a temporary folder on the local file system and then manually saved to the database. See [Loading and Saving Auxiliary Data Files Stored in Databases](#) for more details.

Loading and Saving Auxiliary Data Files Stored in Databases

File versions stored in a database that are referenced as auxiliary data by a model opened in the COMSOL Desktop — either as input source or output target — are loaded on-demand from the database to a temporary *working copy* directory located


on the computer running COMSOL Multiphysics. Any input read by the model, and any output written by the model, goes via files in this directory.


There are various situations in which the model will write to the files in the working copy directory. You may, for example, have added an **Export to File** node () for a **Parametric Sweep** such that, when the corresponding study is computed, export nodes in the **Model Builder** tree run for different parameter values. When Model Manager detects that there are unsaved changes written by the model to the working copy directory, those changes can be saved as a new file version when the model itself is saved via the **Save** window — see [Saving Models to Databases](#). You can also save a new file version directly via the **Auxiliary Data** window — see [The Auxiliary Data Window for Database Input and Output](#).





Loading and saving data files stored in a database is not supported when running an external COMSOL Multiphysics batch process.

EXPORTING OUTPUT DIRECTLY TO A DATABASE

You can save output to a database from nodes in the **Model Builder** tree with export functionality. If you have specified a database as the output target via the **Select File** window, you can click an **Export** button () , or something similar, to export and save the output as a new file version.




For better control over the saved file version, you can also perform the export via the **Export** window. This enables you, for example, to write a custom save comment or change the tag assignments of the file. Choose the database that you want to export to in the list of options on the left. Choose **Add Database** () if you want to add a new database.



The **Export** window is, for example, opened when you click the expand button next to **Export** () and select **Export To** () in the **Settings** window in the Model Builder workspace.

To export output data and save it as a new file version:




- 1 Write the title for the file version in the **Title** field.
- 2 Write the filename of the main output file in the **Filename** field.
- 3 Write an optional save comment in the **Comments** field.

- 4 Click the **Save** button () to export output data and save it as a new file version. If you want to force the creation of a new data file instead, click the expand button next to the **Save** button () and select **Save as New** ()


The **Export** window is otherwise similar to the **Save** window used to save a new model version — see [Saving Models to Databases](#). You can change the target branch for the export by clicking the **Location** link button and selecting another branch in [The Select Location Dialog Box](#). The **Location** link button is hidden if the database only contains a single branch, which is the default for a new database. There is also an **Information** section displaying additional information concerning the saved file, a **Description** section for changing the file's description, and a **Tags** section for setting the assigned tags of the file.

The Auxiliary Data Window for Database Input and Output


The **Auxiliary Data** window includes data files stored in a database that are referenced in the opened model as input or output. For such files, you can:


- Show all versions of the data file in [The Versions Window](#) in the Model Manager workspace. Click the **Show Location** button () in the toolbar. This is useful, for example, if you want to see potentially newer versions for the item in the database.
- Save a new version of the file in the database using the file contents found in the working copy directory for the version. Click the **Save as Version** button () in the toolbar and save a new version in the **Save File** dialog box. This is useful, for example, if the model has written output data in the current COMSOL Multiphysics program session and that output has not yet been saved to the database. See also [Loading and Saving Auxiliary Data Files Stored in Databases](#).
- Update the file reference to the latest version of the file in the database. Click the **Update to Latest Version** button () in the toolbar.

You can also see the current status of a referenced file version:

- **Database not connected** — the status could not be determined as the database is not connected. Right-click a table row and select **Connect to database** () , or open the Model Manager workspace and activate the database from the **Database** section of the **Home** toolbar — see [Activating a Database](#).
- **Up to date** — the referenced file version is the latest version.
- **Newer versions exist** — a later version exist on the same branch as the referenced file version.

- **Not authorized to access item** — you do not meet the current permission requirements for accessing the file. See [Permission Levels](#).
- **Not available in database** — the referenced file version has been permanently deleted in the database.
- **Unsaved changes** — the model has written output to the working copy directory of the file version in the current COMSOL Multiphysics program session and that output has not yet been saved to the database.

You can import data files stored on the file system into the same database as the model, thereby placing them under version control. Select the files you want to import in the table and click **Import to Database** () in the toolbar — see [Importing Files](#).

You can also save a new version of a referenced file already present in the database by updating it from [The Settings Window](#) in the Model Manager workspace — see [File Settings](#). Select the file in the **Auxiliary Data** window's table and click the **Update to Latest Version** button () to use the new file version in the model.




[The Auxiliary Data Window](#) in the *COMSOL Multiphysics Reference Manual*

The Model Manager Workspace

Apart from the Model Manager tools integrated with the Model Builder, Application Builder, and Physics Builder workspaces — see [Databases in the COMSOL Modeling Environment](#) — the Model Manager comes with a dedicated workspace for database-specific tasks. In this section, you will find a brief overview of this workspace. Later sections will discuss each part of the workspace in more depth.

- [Opening the Model Manager Workspace](#)
- [The Home Toolbar](#)
- [The Database Toolbar](#)
- [The Model Manager Workspace Windows](#)

Opening the Model Manager Workspace

To open the Model Manager workspace, click the **Model Manager** () button in the **Workspace** section of the **Home** toolbar in either the Model Builder workspace, the Application Builder workspace (if running on Windows[®]), or the Physics Builder workspace. You can also press the keyboard shortcut Ctrl+Shift+J. The COMSOL Desktop switches to display the toolbar for the Model Manager, as well as opens windows belonging to the Model Manager workspace. A connection attempt is made to the most recently used database upon opening the workspace.




To return from the Model Manager workspace to one of the other workspaces, click on the corresponding button in the **Workspace** section of the Model Manager's **Home** toolbar. You can also press Ctrl+Shift+M for Model Builder and Ctrl+Shift+A for Application Builder.

The Home Toolbar

The **Home** toolbar contains buttons for the more commonly performed tasks in the Model Manager workspace.





THE WORKSPACE SECTION

This section contains buttons for switching to other workspaces in the COMSOL Desktop:

- The **Model Builder** button () or **Physics Builder** button (), depending on if a model or physics is opened in the COMSOL Desktop. You can also use the keyboard shortcut Ctrl+M.
- The **Application Builder** button () if running on Windows®. You can also use the keyboard shortcut Ctrl+A.





THE DATABASE SECTION




This section contains buttons for adding databases, switching the active database in the Model Manager workspace, as well as importing and exporting items in a database.

- The **Activate Database** button, to refresh the active database in the workspace. Click the lower part of the button and select one of the databases to set it as active. See [Activating a Database](#).
- The **Add Database** button (), to open the **Add Database** window for adding a new database. See [The Add Database Window](#).
- The **Import** button (), to import files from the file system into a database. See [Importing Files](#).
- The **Export** button (), to export items from a database to the file system. See [Exporting Items](#).
- The **New Tag** button (), to create a new tag in a database. See [Creating New Tags](#).

THE ITEM SECTION



This section contains buttons that target items in the database — that is, models, data files, and tags.

- Click the **Open** button () to open a model in the COMSOL Desktop. See [Opening Models](#).
- Click the **Run** button () to launch and run an application in the COMSOL Desktop. See [Running Applications](#).
- Click the **Preview File** button () to open a data file using the default application for its file type. See [Previewing Files](#).
- Click the **Compare** button () to compare models in the **Comparison Result** window. See [Comparing Models](#).

- Click the **Set Tags** button () to set the assigned tags of models, data files, or tags. See [Assigning Tags to Items](#).
- Click the **Delete** button () to delete items. This action is not permanent and can be reverted. See [Deleting Items](#).
- Click the **Delete Permanently** button () to permanently delete all versions of one or more items. This action cannot be undone. See [Permanently Deleting Models and Files](#).

THE LAYOUT SECTION


The **Layout** section contains the following functionality for opening and rearranging windows in the Model Manager workspace:

- The **Windows** menu (), for opening windows that are closed by default.
- The **Reset Desktop** button (), to reset the desktop layout to its default state.

The Database Toolbar





The **Database** toolbar contains buttons for tasks common to a database shared between multiple users — that is, a server database accessed via a Model Manager server.

THE DATABASE SECTION

This section contains the same **Activate Database** button and **Add Database** button also available in the **Database** section on [The Home Toolbar](#). The **Databases** button () toggles the visibility of [The Databases Window](#).




THE REPOSITORY SECTION

This section contains buttons for [Advanced Version Control](#) functionality:

- The **Repository** button (), to add a new repository in a database. See [Adding Repositories](#).
- The **Branch** button (), to create a new branch from an existing branch, snapshot, or commit. See [Creating a New Branch](#).
- The **Snapshot** button (), to record a point-in-time snapshot of all items with respect to a particular commit. See [Recording Snapshots](#).
- The **Merge** button (), to merge modifications of items with respect to a source location into a target branch. See [Merging](#).



THE PERMISSIONS SECTION

The **Permissions** section contains buttons for controlling access to various objects in the database, including setting ownership and permission requirements:

- Click the **Owner** button () to set the user that owns a particular database object. See [Transfer Ownership](#).
- Click the **Permissions** () button to set the permission requirements for a particular database object. See [Granting Permissions](#).
- Click the **Permission Template** () button to create a new template of predefined permission requirements that can be reused for database objects. See [Creating your own Permission Templates](#).



THE USERS SECTION

This section contains buttons for managing users and groups:

- The **User** button (), to add a new user in the database. See [Adding Users](#).
- The **Group** button (), to add a new group in the database. See [Adding Groups](#).

THE MAINTENANCE SECTION


This section contains buttons for performing maintenance operations in the database:


- The **Cleanup** button (), to optimize database storage by performing data deduplication and cleanup of unused data present after permanent deletions. See [Database Cleanup](#).
- The **Compact** button (), to optimize a local database by compacting the database file. See [Compacting Local Databases](#).

The Model Manager Workspace Windows


When you open the Model Manager workspace, you will see two windows in the COMSOL Desktop by default:

- [The Model Manager Window](#) — used primarily to search for models and data files.
- [The Settings Window](#) — used to show settings for various database objects, as well as update and save them in the database. See also [Overview of a Model Manager Database](#).

You can always restore the COMSOL Desktop to this layout by clicking the **Reset Desktop** button () in the **Layout** section of [The Home Toolbar](#). From the **Windows**


menu () in the same section, you can open the following optional windows in the workspace:

- The **Comparison Result** window — used to compare model versions saved in the database. See also [Comparing Models Saved in Databases](#).
- The **Databases Window** — used to browse, organize, and administer your configured databases via [The Databases Tree](#).
- The **Commits Window** — used to view the commit history of a branch, optionally filtered to only include commits that involve a particular item.
- The **Versions Window** — used to view the version history of an item with respect to a branch.
- The **Maintenance Window** — used to perform maintenance operations for an item.
- The **References Window** — used to view relations between models and data files. See also [The Auxiliary Data Window for Database Input and Output](#).

The **Model Manager** window and the **Databases** window play a special role in the Model Manager workspace — whichever has focus determines the target for the buttons in [The Home Toolbar](#) and [The Database Toolbar](#). You may, for example, have selected one model in the **Databases** window and another model in the **Model Manager** window. The window that has focus determines which model is opened if you click the **Open** button () in the **Item** section of the **Home** toolbar.

The selection in the **Model Manager** window and the **Databases** window also determines what is shown in the **Settings**, **Commits**, **Versions**, **Maintenance**, and **References** windows, again depending on which one of these two former windows has focus.



You can turn off the automatic selection linking in the **Settings**, **Commits**, **Versions**, **Maintenance**, and **References** windows by clicking the **Link with Selection** button () in their respective toolbars. Click once more to turn on the linking.

The selection in the **Databases** window determines the searched location in the **Model Manager** window. If you select a branch tree node in [The Databases Tree](#), for example, the **Model Manager** window will automatically switch to search with respect to that branch — see also [Searching Versions](#). Similarly, if you change the searched location via [The Select Location Dialog Box](#) in the **Model Manager** window, the corresponding tree node is selected in [The Databases Tree](#).

Overview of a Model Manager Database

You will encounter several concepts and terms when working with Model Manager databases that are specific to the Model Manager tools. This section contains a guide to these concepts and terms — you may want to skip ahead to the next section, [Browsing Databases](#), and refer back to this section as needed. See also the [Glossary](#).

In this section:

- [Models](#)
- [Files](#)
- [Tags](#)
- [Items](#)
- [Commits](#)
- [Branches](#)
- [Snapshots](#)
- [Locations](#)
- [Repositories](#)
- [Users](#)
- [Groups](#)
- [Permission Templates](#)



A common feature of the objects described in this section is that they all have an underlying key that uniquely identifies them in the database. Unlike, for example, changing the filename of a file on the file system, you can safely change the label, name, and title of any database object without worrying about, for example, that other objects lose references they might have to the renamed object.

Models

You can create a *model* in a Model Manager database by either saving the model opened in the COMSOL Desktop or by importing a model directly from the file system.



-
- [Saving Models to Databases](#)
 - [Importing Files](#)
-




Every time you save a model, a new *model version* is created. This does not mean that a full copy of the model is saved anew in the database — Model Manager is able to reuse any data it finds unmodified between versions. This includes the model tree itself, any binary data used for geometries, meshes, solutions, and results, as well as any other data that the model may use. Once saved, a model version cannot be modified (except for the deletion of generated data that can be recreated from the model, that is, **Built, Computed, and Plotted Data**). You can be confident that what you save to the database will always be returned when opening the model version again, given that you open it in the same COMSOL Multiphysics version as the model version was originally saved in.



There is a subtle distinction between a model and all the versions of the model in a database. You set **Access Control** on the models themselves, while you open, save, search, and organize versions of the models. The same remark applies to any of the **Items** in a Model Manager database.

ITEM TYPES


Model versions come in three different *item types*:

- **Model** (), which is the standard type obtained, for example, when creating a new model from the Model Builder.
- **Application** (), which is a model that also has an application UI as defined in the Application Builder.
- **Physics** (), which is the type obtained when saving from the Physics Builder.

You may notice that a model can start out as the first type for its first couple of versions, but then transition into the second type once you save a version in which you have added an application UI. The opposite transition will occur if you save a version in which you have removed the application UI.

SAVE TYPES

Models can be created in the database as two different *save types*:


- **Regular** (represented by one of the icons for **Item Types**)
- **Draft** ()

Regular Models

A regular model is one that has been created in the database by saving a new model via the **Save** window or by importing a model directly from the file system. You can think

of a regular model as the main result of your modeling work. Each version represents a clear transition in which you made enough progress that it is worth keeping the version around for future reference. Perhaps you want to go back to one of the older model versions and from that create a completely new model (with its own set of model versions), or perhaps the model version that you save corresponds to a step in which your modeling work is completed, and it is time for your coworker to take over.

Drafts

A draft model is one that has been created by going to the **File** menu and selecting **Save Draft** () , or by pressing Ctrl+S, when a regular model is opened in the COMSOL Desktop.



Saving Drafts of Models

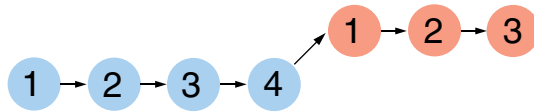
A draft model is version controlled in the same way as a regular model: The first time you select **Save Draft**, a new draft model is created in the database. Selecting **Save Draft** after that will save new versions of that *same* draft model. You may think of a draft as a transient model used for saving intermediate changes, without muddying the version history of the original model.



The draft is automatically set with a **Private** permission template when you create a new draft in a server database accessed via a Model Manager server — see [Predefined Permission Templates](#). Only you, as its owner, will be able to open or save the draft, although other users may still see it in search results.

The database stores a reference between the draft and the version of the regular model the draft originated from. This is analogous to how regular models created from existing models via **Save as New** in the **Save** window remembers their origin — see [Splitting a Model Version History in Two](#). The stored reference enables Model

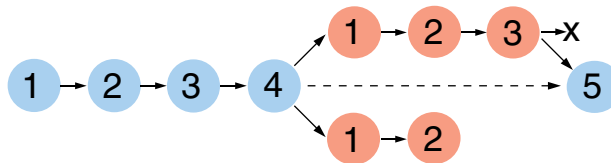
Manager, for example, to automatically discover if a newer version has been saved of the regular model *after* the draft was created — see [Save Conflicts](#).



A schematic of a possible history of a regular model and a draft model created from the former. Starting from the left, four model versions of a regular model have been saved. A draft was then created from the fourth version, after which three versions of the draft have been saved.


Once you are finished with your draft, you can open the **Save** window and save the current draft as a version of the original model. The draft will be automatically deleted in the process, although this deletion is not permanent — you can, for example, find the deleted draft via the [The Commit Details Dialog Box](#) for the corresponding commit.

You can create multiple drafts from the same regular model, each draft having its own identity and version history, by repeatedly opening a version of a regular model and pressing Ctrl+S. You and your coworker may, for example, both start working on your own drafts of the same regular model, perhaps with the intention of combining your work once you are finished.



A schematic of a possible history of a regular model and two drafts models saved in a database. The middle row starting from the left represents four model versions of a regular model. From the fourth model version, two drafts (bottom and top rows) have been independently created. Two versions of the bottom row draft, and three versions of the top row draft, have been saved. The top row draft's latest version has then been saved back to the original regular model as a fifth version, automatically deleting the top row draft in the process (represented by the x-mark). The bottom draft may have been discarded, awaiting, for example, its owner to manually delete it.

MODEL SETTINGS


The **Settings** window for a model shows settings for a specific version of the model. Update any of the settings and click the **Save** button () to save a new version of the model. You can write an optional save comment. Click **OK** to save.

The Version Section

This section displays the following fields:

- **Location.** The commit location in the database in which the model version is saved. See [Locations](#).
- **Saved.** The time when the model version was saved.
- **Saved by.** The display name of the user that saved the model version.
- **Saved in.** The COMSOL Multiphysics version that the model version was saved in.
- **Title.** The title of the model in the saved version.

The title is the same as shown in the **Title** field in the **Presentation** section in the root node's **Settings** window for a model opened in the COMSOL Desktop.



- **Filename.** The filename used by the model version if exporting it to the file system.
- **Update from.** An optional field in which the path to a file on the file system can be specified to save a new version of the model from that file.
 - a Click the **Browse** button.
 - b Select a file on the file system and click the **Open** button.
 - c Click the **Save** button (.

You can update from any model file that can be opened in COMSOL Multiphysics. The saved model version will first be converted to the current COMSOL Multiphysics version.

- **Description.** The description of the model in the saved version.

The description is the same as shown in the **Description** field in the **Presentation** section in the root node's **Settings** window for a model opened in the COMSOL Desktop

The Contents Section


This section displays the model tree as it looked when the model version was saved to the database. Use the **Collapse** button () and **Expand** button () to collapse and expand nodes in the tree.


Select a node in the tree and click **Details** () in the toolbar below the tree to open the **Details** dialog box containing node field values and setting field values for the selected model tree node.

The **Node** table in the **Details** dialog box shows the values for the properties of a node. The available properties will differ between node types. The **Settings** table shows all settings of the node.




You can find models in the database by searching on the node properties and settings shown in the **Details** dialog box for a model tree node. See [Content Filters](#) and [The Model Manager Search Syntax](#).

Select a node and click **Open Node** () to open the model in the COMSOL Desktop with the model tree node automatically selected in the user interface.

Select a component, geometry sequence, material, or physics node and click **Insert into Opened Model** () to insert the selected node into the model currently opened in the COMSOL Desktop. You will be asked to select a target parent node for the insertion if there is more than one target available in the opened model.



The **Insert into Opened Model** button () is a useful alternative to [Inserting Components, Materials, and Physics from Databases](#) via the **Select Model** window when you want to reuse multiple model content parts from the same source model. Going via the **Contents** section also has the advantage that you can right away select which component, geometry sequence, material, or physics you want to insert from a model.

The Tags Section

This section displays the [Tags](#) assigned to the model as a collection of tag pills.



See also [The Tags Section](#) for [Items](#).

Files

You can import any type of file into a Model Manager database. Any file that is not recognized as a COMSOL Multiphysics file (that is, with the file extension mph) or a

Physics Builder file (that is, with the file extension `mphpb`) is referred to as a *data file*, or *file*, in a Model Manager database.





Importing Files

Files are version controlled similarly as [Models](#) in the database. For example, when you import from the file system, a new file is created in the database with an associated first version. You can update an existing file in the database by selecting a file on the file system, and saving it as a new version from the [File Settings](#) window. You can also, for example, update a file in the database by using it as an output target of a model — see [Loading and Saving Auxiliary Data Files Stored in Databases](#).

You may wonder why COMSOL Multiphysics simulation models and data files are two different concepts in a Model Manager database, when they are all “just files” when stored on a file system. The main reason for the distinction is their respective storage characteristics and supported functionality. The known internal structure of a model enables efficient data reuse between versions, as well as searching deep within a model’s content using [The Model Manager Search Syntax](#). A data file is stored in the database as a chunk of binary or text data whose content is opaque to Model Manager.

ITEM TYPES

File versions come in two different *item types*:


- **File** (), which is the standard type obtained when the version consist of just a single file when stored on the file system. Examples include a text file containing interpolation function data or a movie file for a results animation.
- **Fileset** (), which is a version consisting of multiple files when stored on the file system. Examples include CAD data consisting of a main CAD assembly file and one or more external component files that the assembly reference, or an HTML report consisting of a main HTML document and one or more image files referenced by the document.

You may notice that a file can, for example, start out as the first type for its first couple of versions, but then transition into the second type if you save a version that contains multiple files.



Whether or not multiple data files are best stored as separate items or together as a fileset in a Model Manager database is best answered depending on if the files naturally are version-controlled as a “collective whole”. Prefer assigning [Tags](#) if all you want to accomplish is some organizational grouping in your database.

FILE SETTINGS

The **Settings** window for a file shows settings for a specific version of the file. Update any of the settings and click the **Save** button () to save a new version of the file. You can write an optional save comment. Click **OK** to save.

The Version Section




This section displays the following fields:

- **Location.** The commit location in the database in which the file version is saved. See [Locations](#).
- **Saved.** The time when the file version was saved.
- **Saved by.** The display name of the user that saved the file version.
- **Title.** The title of the file in the saved version.
- **File size.** The size of the file version when stored on the file system. For a fileset, this is the sum of the individual file sizes.
- **Description.** The description of the file in the saved version.

The title is automatically set to the filename when importing a file from the file system into a Model Manager database. You are free to change the title to something else, however — the unique identity of the file itself in the database will not be affected.

The Contents Section

This section shows a table with all files included with the version. The table has columns for the files’ filenames — possibly including a hierarchy of subfolders — and file sizes. A file version with zero or multiple table rows is a fileset; a file version with a single table row is a plain file.

Click the **Add** button () to add a file from the file system to the table. Click the **Remove** button () to remove selected files from the table. Click the **Replace** button () to replace a selected file in the table with a file from the file system.

A confirmation dialog box is shown if a file in the table is about to be replaced when adding or replacing with a file that has the same filename.



If you have a license for the CAD Import Module available when adding or replacing with a CAD assembly file, an attempt will be made to automatically resolve and include external component files that the assembly references.


The Tags Section

This section displays the [Tags](#) assigned to the file as a collection of tag pills.



See also [The Tags Section](#) for [Items](#).

Tags


A *tag* () is used to label and organize models, files, and even other tags in the database — see [Tagging Models and Files](#) to learn more. Tags can be created by importing them from folders on the file system or by manually creating new tags from within the Model Manager workspace.



- [Importing Files](#)
- [Creating New Tags](#)

Similar to [Models](#) and [Files](#), tags are version controlled in the database. Whenever you save a tag in the database — for example by giving it a new title — a new version of the tag is created. Changing the title of a tag does not mean that other items lose their tag assignment — you will still be able to find items with the tag assigned using the new title. Unlike versions of models and files, however, versions of tags do not have any underlying content associated with them.

TAG SETTINGS

The **Settings** window for a tag shows settings for a specific version of the tag. Update any of the settings and click the **Save** button () to save a new version of the tag. You can write an optional save comment. Click **OK** to save.

The Version Section

This section displays the following fields:

- **Location.** The commit location in the database in which the tag version is saved. See [Locations](#).
- **Saved.** The time when the tag version was saved.
- **Saved by.** The display name of the user that saved the tag version.
- **Title.** The title of the tag in the saved version.

The Tags Section

This section displays the parent tags assigned to the tag as a collection of tag pills.



See also [The Tags Section for Items](#).


Items

[Models](#), [Files](#), and [Tags](#) are collectively referred to as *items* in a Model Manager database. Items share many common features and functionality, including:


- Items are version controlled. Every time you save an item, a new version is created in the database.
- Items can be labeled and organized in the database by [Adding and Removing Tag Assignments](#) and by [Organizing Items in Repositories](#).
- Items can be imported and exported from and to the file system via [Bulk Operations](#).
- Item versions can be found via [Searching and Filtering](#).
- Items support [Advanced Version Control](#) such as [Branching](#), [Reverting](#), and [Merging](#).

ITEM SETTINGS


The **Settings** window for an item shows settings for a specific version of the item. Some settings are unique to a model, file, or tag, while others are shared by all items. Update



any of the settings and click the **Save** button () to save a new version of the item. You can write an optional save comment. Click **OK** to save.

The Tags Section


All items have a **Tags** section in their **Settings** window that displays the **Tags** currently assigned to the item as a collection of tag pills. Click a tag pill and select **Remove** () to remove a tag assignment.

Win


Click the **Add Tag** button () in the section's toolbar to find an existing tag to assign to the item. Write the title of the tag in the text field to filter the popup list of available tags. The matching tags are shown with their title followed by the titles of their parent tags, if any, within parentheses. Either double-click a tag in the list or select a tag and press Enter to assign it to the item.

Click the **Set Tags** button () to open the **Set Tags** dialog box. The dialog box contains a tag tree of all available tags. Select a check box for a tag in order to assign the tag to the item. Clear a selected check box to remove a tag assignment. Click **Clear Tags** () to clear all selections in the tree. You can filter the tree of available tags by writing a tag title in the text field above the tree. Click **OK** to finish the tag assignment.



Prefer the dialog box opened via the **Set Tags** button () on [The Home Toolbar](#) in the Model Manager workspace over the **Settings** window if you only want to update the assigned tags of an item without saving a new version — see also [Adding and Removing Tag Assignments](#). That dialog box also supports updating the assigned tags of multiple items at once.

Commits

A *commit* () is a set of related changes made to **Items** — that is, **Models**, **Files**, and **Tags** — within a single database save operation. This includes anything from saving item versions, changing the assigned tags of items, or deleting items, to creating a new branch, merging into a branch, and reverting a commit. The changes are saved to the database as a “unit” and, as such, can also be *reverted* as a unit.

A commit saved in the database includes:

- The branch that the commit was saved to — see [Branches](#).
- The time when the commit was saved.

- The user that saved the commit.
- An optional commit comment provided by the user.
- The set of related changes made to items in the commit.

Given a particular commit, you can browse and search the versions of items that were the *latest versions* at the time of that commit. A schematic representation of this is shown in [Figure 2-2](#): In the first commit, a first version of a model A and a model B were saved. In the second commit, a second version of model A was saved, a first version of a tag T was saved, and model B was tagged by T. In the third commit, model A was deleted and a second version of model B was saved. The database can be browsed and searched with respect to each of the three commits, with each big circle surrounding what you will find.

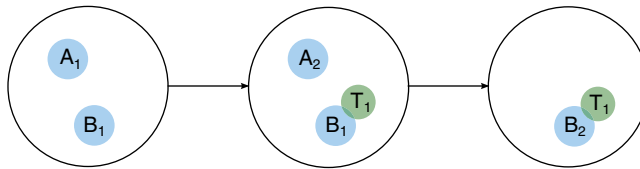


Figure 2-2: A schematic of three hypothetical commits (big circles) saved in a database. You can browse and search versions and tag assignments with respect to any one of these commits.



A synonym to commit often found in other version control systems is *revision*.



- [Locations](#)
- [Basic Version Control](#)
- [Searching Versions](#)



Branches

A *branch* (🌿) is a chronologically ordered sequence of [Commits](#) saved in the database. The sequence of commits form a history of things that has happened to [Items](#) in the database.

An important commit in a branch is the one saved most recently — that is, the *latest commit* on the sequence. When browsing and searching the database with respect to

this commit, you will find the latest versions of models and files as well as their latest assigned tags.

The branch itself is typically used as a representative for the latest item versions (that is, the version found for the most recent commit) in Model Manager:

- Expand a branch node () in [The Databases Tree](#) to browse the latest versions of all items, given that **Items** is selected in the **Show** menu () in [The Databases Window Toolbar](#).
- Select a branch in [The Select Location Dialog Box](#) to search the latest model and file versions in the **Open**, **Select File**, **Select Model**, and **Model Manager** windows. This is also the default choice in these windows.
- Select a branch in [The Select Location Dialog Box](#) to export the latest versions from the **Export** dialog box.

You can create a new branch from an existing parent branch by *branching off* from a particular source commit. This introduces an *alternative* commit history that runs in parallel with that of the parent branch — see [Branching](#).





When referring to the latest versions of items in a Model Manager database, it is always understood as being with respect to a particular branch.



[Creating a New Branch](#)

DEFAULT BRANCH

An initial, default, branch () is automatically created for a new database. This is the branch that all save actions targeting a particular repository use by default.


You can change the default branch in case you have created multiple branches in a repository. Select the corresponding branch tree node in the **Databases** tree, right-click, and select **Set Default Branch** ()

BRANCH SETTINGS


The **Settings** window for a branch shows:

- **Database**. The label of the database that the branch belongs to.
- **Repository**. The name of the repository that the branch belongs to.

- **Name.** The name of the branch.
- **Search.** The item data that can be searched on the branch. See [Searching in Branches](#).

Clicking the **Save** button () saves the **Name** and **Search** fields for the branch.



Snapshots

A *snapshot* () is a reference to a particular commit on a branch.



- [Commits](#)
- [Branches](#)

The snapshot itself is typically used as a representative, or *recording*, of the latest versions of items at the time of the commit:

- Expand a snapshot node () in [The Databases Tree](#) to browse the recorded item versions, given that **Items** is selected in the **Show** menu () in [The Databases Window Toolbar](#).
- Select a snapshot in [The Select Location Dialog Box](#) to search the recorded model and file versions in the **Open**, **Select File**, **Select Model**, and **Model Manager** windows.
- Select a snapshot in [The Select Location Dialog Box](#) to export the recorded versions from the **Export** dialog box.




Recording Snapshots

SNAPSHOT SETTINGS

The **Settings** window for a snapshot shows:

- **Database.** The label of the database that the snapshot belongs to.
- **Repository.** The name of the repository that the snapshot belongs to.
- **Date.** The timestamp of the commit that the snapshot references.
- **Name.** The name of the snapshot.

Clicking the **Save** button () saves the **Name** field for the snapshot.

Locations

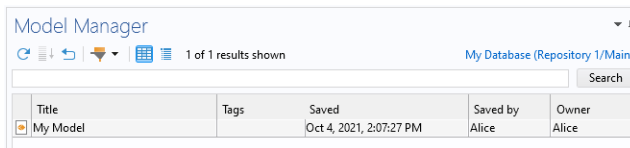
[Commits](#), [Branches](#), and [Snapshots](#) are all known as *commit locations*, or *locations*, in Model Manager. Each location is a commit in its own right or acts as a representative of a particular commit:

- **Branch.** The most recent commit saved on the chronologically ordered sequence of commits.
- **Snapshot.** The commit that the snapshot references.


A branch location is probably the most common, and useful, of the three types. By browsing and searching with respect to such a location, you are guaranteed to see the latest versions of items on that branch.

THE SELECT LOCATION DIALOG BOX

You can select a branch or, depending on context, snapshot from the **Select Location** dialog box. You open the dialog box by clicking on a *location link button*, seen here in the upper-right corner of [The Model Manager Window](#):



The text of a location link button is shown with the name of a branch or snapshot. A commit is shown with its date. Typically the label of the database and the name of the repository are also included in the text, such that the text format is Database (Repository/Location).

In the tree in the **Select Location** dialog box, expand nodes until you reach a branch or snapshot leaf node, select it, and click **OK**. If **Deleted Branches** or **Deleted Snapshots** has been selected in the **Show** menu () in [The Databases Window Toolbar](#), such branches or snapshots will also be available for selection.

Repositories

A *repository* () is a container for a collection of items and their versions in the database. When the database is created, a first repository is automatically added for you.


You would typically add more repositories to the database if you want to restrict access to a collection of items for a particular set of [Users](#) and [Groups](#). By [Granting Permissions](#) on the repository, you can, for example, restrict who is able to browse and search the item versions in the repository.


Each repository contains one or more branches. Whenever you add a new repository, an initial [Default Branch](#) in that repository is created.



Adding Repositories

DEFAULT REPOSITORY


If you have added multiple repositories to the database, Model Manager tries to remember the repository you last accessed in, for example, the **Open**, **Select File**, and **Select Model** windows. You can also set one of the repositories as the default repository () . Analogous to the [Default Branch](#), this is the repository that all save actions targeting the database use by default.

The first repository created in the database is initially set as the default one. To change the default, select the corresponding repository tree node in the **Databases** tree, right-click, and select **Set Default Repository** () .


REPOSITORY SETTINGS

The **Settings** window for a repository shows:

- **Database**. The label of the database that the repository belongs to.
- **Name**. The name of the repository.
- **Default Branch Name**. The name of the repository's [Default Branch](#).

Clicking the **Save** button () saves the **Name** field for the repository.

Users


A *user* () is someone who has connected to a Model Manager database. For a local database, the user is created with the user account name on your computer. For a server database, the user is created using the credentials you provide when connecting to the Model Manager server — see [Connecting to a Server Database](#).

A user in a Model Manager database is primarily used to identify the individual that has saved a commit, the owner of a database object, or the individuals that have been granted permissions for a database object.

USER SETTINGS

The **Settings** window for a user shows:


- **Database.** The label of the database that the user belongs to.
- **Name.** The username of the user.
- **Display Name.** An alternative name used for display purposes.
- **Group Memberships.** The groups in the database that the user is a member of.

Clicking the **Save** button () saves the display name and the group memberships for the user.

Group Memberships

Click **Add** to add the user as a member to a group in the database. In the **Search** dialog box, type a name or display name for groups and click the **Search** button. Select groups in the search result table and click **OK**.


Select groups in the **Group Memberships** list and click **Remove** to remove group memberships from the user.

Click the **Save** button () to save any changed group memberships to the database.



Managing Users

Groups


A group () is a collection of users and other groups. You typically create groups to more easily manage permissions common to several users in the database.

GROUP SETTINGS

The **Settings** window for a group shows:

- **Database.** The label of the database that the group belongs to.
- **Name.** The name of the group.


- **Display Name.** An alternative name used for display purposes.
- **Group Members.** The users, and other groups, in the database that are members of the group.

Clicking the **Save** button () saves the display name and the group members for the group.

Group Members

Click **Add** to find a user, or another group, in the database to add as a member to the current group. In the **Search** dialog box, type a name or display name for users and groups and click the **Search** button. Select users and groups in the search result table and click **OK**.

Select users and groups in the **Group Memberships** list and click **Remove** to remove group members from the group.

Click the **Save** button () to save any changed group members to the database.




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.



Managing Groups

Permission Templates

A *permission template* () is a saved list of permission assignments for a set of users and groups that you can apply to database objects. You can, for example, use a permission template to reuse the same permissions for a large collection of models, only having to update in one place — the permission template itself — in case you want to change these permissions.

Model Manager comes with three [Predefined Permission Templates](#), descriptively named Public, Protected, and Private, for each of the database objects that you can

control access to. You can also create custom permission templates for [Models](#) and [Files](#) — see [Creating your own Permission Templates](#).



Reusing Permission Assignments Using Permission Templates

PERMISSION TEMPLATE SETTINGS

The **Settings** window for a permission template shows:


- **Database.** The label of the database that the permission template belongs to.
- **Name.** The name of the permission template.
- **Type.** The object type that the permission template can be applied to. Either **Model** or **File**.
- A table containing the permission assignments of the permission template.

Clicking the **Save** button () saves the **Name** field and permission assignments for the permission template.

Permission Assignments

Click **Add** to add permissions for a user or group to the permission template. In the **Add** dialog box, type a name or display name for users and groups and click the **Search** button. Select the user or group you want to add in the search result table. You can also select the special **Everyone** or **Owner** options — see [Everyone and Owner](#). At the bottom of the dialog box, select the permissions to assign the selection. Click **OK** to add the permission assignment.

Select a row in the permission assignment table and click **Edit** to change the permissions for the selection. Click **Remove** to remove the selection from the table.

Click the **Save** button () to save any changed permission assignments to the database.

Browsing Databases

For models and data files stored on the file system, there is typically only one way to browse them — what you see is what you get. Taking the same approach in a Model Manager database could, however, quickly become confusing and hard to navigate — models with hundreds of versions may take up an entire search result, making models with only a few versions harder to find. As the title of a model rarely changes, you would also have to meticulously compare dates when opening a model version to know you are working off the most recent.

In this section, you will learn how various windows in the Model Manager workspace can be used to browse and search in useful subsets of model and file versions in your databases — including browsing and searching the latest versions of items, listing the version history of a particular item, listing the history of all changes made to items, as well as navigating the relationships between different versions.

- [The Model Manager Window](#)
- [The Databases Window](#)
- [The Settings Window](#)
- [The Commits Window](#)
- [Activating a Database](#)
- [The Versions Window](#)
- [The References Window](#)
- [Opening Models](#)
- [Running Applications](#)
- [Previewing Files](#)
- [Comparing Models](#)
- [Copying Model and File Locations](#)

The Model Manager Window

Use the **Model Manager** window to find versions of [Models](#) and [Files](#) with respect to different [Locations](#) by writing search and filter expressions in the search field and clicking the **Search** button. 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, description, assigned tags, and filename of models and files.

You can also apply separate [Item](#) and [Content Filters](#) via [The Filter Dialog Box](#). Select a filter from the **Add Filter** menu button () in the toolbar to open the dialog box. Applied filters are shown below the search field — see [Applied Filter Pills](#).



The searched location is initially set to the [Default Branch](#) in a repository, which means that the latest versions of models and files are searched. Click the top-right link button to select another location in [The Select Location Dialog Box](#).

You can switch the presentation of the search result between either a [Table View](#) or a [Tree View](#) — the former primarily used when you want to quickly find a particular item and the latter when you want to browse a larger collection of items.



Searching and Filtering



If Model Manager detects that a commit has been saved on a searched branch, an information message () is displayed at the bottom of the **Model Manager** window. This includes both commits made by you from within the COMSOL Desktop, as well as commits made by your coworkers if connected to a server database. Either click the information message text itself or the **Refresh** button () to see any updated results.

THE MODEL MANAGER WINDOW TOOLBAR

The toolbar in the **Model Manager** window contains the following toolbar buttons:








- Click the **Refresh** button () to refresh the search result while keeping the search expression and applied filters unchanged.
- Click the **Show More** button () to include more matching models and files in the search result. The button is disabled when the [Tree View](#) is shown for a branch.
- Click the **Reset** button () to clear the current search expression and applied filters.
- Click the **Add Filter** button () to apply a filter.
- Click the **Table** button () to view the search result in a table.
- Click the **Tree** button () to view the search result in a tree.

TABLE VIEW

The *table view* of the search result shows matching model and file versions in a table with columns:

- The type column — the type of the model or file represented by an icon. See [Item Types](#) and [Save Types](#).

- The **Title** column — the title set for the model or file in that version.
- The **Tags** column — the tags set for the model or file with respect to the searched location.
- The **Saved** column — the time when the version was saved.
- The **Saved By** column — the name of the user that saved the version.
- The **Owner** column — the name of the user that owns the model or file.

The search result is sorted on title, with a maximum of 100 models and files initially included in the result. You can append more search results at the bottom of the table until all matching models and files have been fetched by repeatedly clicking the **Show More** button () in the toolbar.



Using the [Table View](#) is recommended when you expect to find a small number of models and data files in your search.

TREE VIEW

The *tree view* of the search result shows matching model and file versions under their assigned tags in [The Tag Tree](#). Given that an item may be assigned multiple tags, you can encounter the same model or file in multiple positions in the tree. Each model or file tree node shows the title set for the version, the time when the version was saved, and the name of the user that saved the version. The appearance and behavior of the tree differ somewhat depending on if you are searching item versions with respect to [Branches](#) or with respect to [Snapshots](#) or [Commits](#), with the former typically offering a richer experience — see also [Searching Versions](#).




Using the [Tree View](#) is recommended when you expect to browse through a large search result of potentially thousands of items.

Searching in Branches


When searching in a branch, the tree view shows a count of the number of matching models and files found under each tag node within parentheses next to the tag's title. If you have created a large number of tags in your database, the count may be shown first when you expand a tag node (with ellipses shown within parentheses until then).

If you have not written a search expression in the search field or added any filters, tags that are not assigned to any items are also included in the tree. This enables you to browse through all items in [The Tag Tree](#) — tags included. If there is a search


expression or applied filter, tags are excluded from the tree if the count is known to be zero.

The model and file versions under a tag tree node are sorted on title, with a maximum of 100 versions initially included. When more results are available under a tag node, a **Show More** tree node () can be expanded to reveal the items that comes next.

Searching in Snapshots and Commits






When searching item versions in a snapshot or commit, the tree view shows the first 100 matching model and file versions (sorted on title), irrespective of their assigned tags. You can append more search results to the tag tree until all matching models and files have been fetched by repeatedly clicking the **Show More** button () in the toolbar.



There is a subtle difference between the tag tree shown in the tree view of the **Model Manager** window when searching in [Snapshots](#) and the corresponding tag tree shown in [The Databases Tree](#). For the former tree, you only see the tags assigned to models and files that are included in the search result. This means that more and more tags may show up when you repeatedly click the **Show More** button () in the toolbar. Tags that are not assigned to any models or files will never show up. For the latter tree, you always see all tags at a fixed level when you expand a tree node.

The Databases Window

The **Databases** window shows a tree with all databases you have added in the COMSOL Desktop. You typically use the tree to browse and administer the content of databases, including, for example, edit [Database Configurations](#), add [Repositories](#), create [Branches](#), record [Snapshots](#), and, for server databases, manage [Users](#) and [Groups](#).







Click the **Step In** button () to show a subtree of the tree in the window. You can continue stepping into nodes until only leaf nodes are visible. Click **Step Back** () to return one step, or **Step Home** () to show the whole tree. You can toggle the visibility and appearance of various nodes in the **Show** menu () and the **Item Tree Node Text** menu () .

TOGGLING THE DATABASES WINDOW

Click the **Databases** button () on the **Database** toolbar to toggle the visibility of the **Databases** window.


THE DATABASES WINDOW TOOLBAR



The toolbar in the **Databases** window contains the following toolbar buttons:

- Click the **Step Home** button () to show all nodes in the **Databases** window's tree; that is, to return to the default tree view after stepping into nodes.
- Click the **Step Back** button () to show the nodes previously shown before stepping into a node.
- Click the **Step In** button () to only show the child nodes, and their descendants, of a selected node in the **Databases** window's tree.
- Click the **Refresh** button () to refresh the child nodes of the selected node.
- Click the **Show** menu () to set visibility options for nodes.
- Click the **Item Tree Node Text** menu () to set text options for nodes corresponding to models and files.

THE DATABASES TREE

The top nodes in the **Databases** window's tree are the **Database Configurations** of all databases that have been added to the COMSOL Desktop: local databases and server databases accessed via a Model Manager server.

The child nodes to the databases configurations consist of all **Repositories** you are authorized to see, as well as a **Security** node () containing child nodes related to user management. The **Security** node is hidden for a local database.


Each repository node contains a **Branches** node () and a **Snapshots** node (). The **Branches** node and the **Snapshots** node contain all **Branches** and **Snapshots**, respectively, you are authorized to see.




Each of the branch and snapshot nodes contain the latest versions of **Items** — that is, **Models**, **Files**, and **Tags** — with respect to the corresponding commit. For a branch node, these are the latest item versions at present, while for a snapshot, these are the item versions that were the most recent ones at the snapshot's timestamp — see also **Locations**.

The items are further organized into a subtree based on their assigned tags with respect to the corresponding commit — see also **The Tag Tree**. You may think of the branch node or snapshot node as representing a root tag in this subtree.




Select **Items** in the **Show** menu button () to see item versions under branch and snapshot nodes.


The items under a tag node are sorted on title, with a maximum of 100 models and files initially shown. When more items are available under a tag node, a **Show More** tree node () can be expanded to reveal the items that comes next.



The **Security** node contains a **Users** node () , a **Groups** node () , and a **Permission Templates** node () . Each one contains, respectively, all **Users**, **Groups**, and custom **Permission Templates** in the database.




Right-click a node and select **Refresh** () to refresh all child nodes of that node. You may find this useful, for example, to see changes made in a server database from another COMSOL Multiphysics program session, perhaps run by your coworker.



Deleting and Restoring Unversioned Database Objects


You can delete **Repositories**, **Branches**, **Snapshots**, **Users**, **Groups**, and **Permission Templates** by right-clicking their corresponding tree nodes and selecting **Delete** () . This deletion is not permanent but rather marks them as hidden in the COMSOL Desktop by default.

Clicking **Show** () enables you to show deleted (hidden) nodes in the tree. You can right-click such nodes and select **Restore** () to remove their hide-marker, making them visible again everywhere in the COMSOL Desktop. The options set via the **Show** toolbar button also determines the visibility of deleted repositories, branches, and snapshots in [The Select Location Dialog Box](#).

Permission templates can also be permanently deleted. Right-click and select **Delete Permanently** () . This requires that the permission template is not assigned to any items.

The Settings Window

The **Settings** window shows settings for the database object selected in [The Databases Window](#) or [The Model Manager Window](#) (depending on which window has focus). Click **Link with Selection** () to disable this automatic linking. You can still update the **Settings** window with a new database object by right-clicking the object and selecting **Settings** () .

You use the **Settings** window to view and update an object stored in the database. To update the object, change any of its values in the **Settings** window's fields and click the **Save** button () . In contrast to the **Settings** windows in the Model Builder, the

Application Builder, or the Physics Builder workspaces, changes made in the window are not automatically saved to the database. The **Reset** button (↶) will be enabled when there are unsaved changes to an object shown in the **Settings** window.



Remember to click **Save** (💾) to save any changes made to an object in the **Settings** window.

Model Manager will try to remember unsaved changes for an object during the program session: If you change some fields for a database object, select another database object in the **Databases** window or the **Model Manager** window (so that the **Settings** window is updated), and then return to the first database object, your unsaved changes for the object will be reapplied. This enables you to save the changes to the database at your convenience. Model Manager will also show a warning dialog box if you have unsaved changes when the **Settings** window is updated with a new database object.



You can disable the warning for unsaved changes: clear the **Warn if the Settings window contains unsaved changes** check box on the **Model Manager** page in the **Preferences** dialog box.




When you click the **Save** button (💾) for a model, file, or tag, a dialog box is shown in which you can provide an optional commit comment. Click **OK** to save a new version of the item. If you have made any changes to the tag pills in the **Tags** section, the assigned tags of the item are also updated. Clicking **Save** for any other database object immediately saves the changes to the database.



- [Model Settings](#)
 - [File Settings](#)
 - [Tag Settings](#)
 - [Branch Settings](#)
 - [Snapshot Settings](#)
 - [Repository Settings](#)
 - [User Settings](#)
 - [Group Settings](#)
 - [Permission Template Settings](#)
-


THE SETTINGS WINDOW TOOLBAR

The toolbar in the **Settings** window contains the following toolbar buttons:



- Click the **Link with Selection** button () to enable or disable whether to automatically update the **Settings** window based on the current selection in the **Databases** window or the **Model Manager** window (which ever had focus last).
- Click the **Save** button () to write any changes made in the **Settings** window to the database.
- Click the **Reset** button () to reload the shown object in the **Settings** window from the database, thereby discarding any pending changes you have in the window. The button is disabled if no changes have been detected.

The Commits Window

You use the **Commits** window to view the history of all **Commits** belonging to a branch. Click the link button to select another branch in [The Select Location Dialog Box](#).









The commits are shown in a table with the 100 most recent commits initially retrieved from the database. Click **Show More** () to append older commits to the table. If the branch was created from another branch, older commits on the parent branch are also included in the table. The table columns are:

- The **Date** column — the time when the commit was saved.
- The **User** column — the name of the user that saved the commit.
- The **Branch** column — the name of the branch that the commit belongs to.
- The **Comments** column — the optionally provided comments when the commit was saved.



The commit history shown in the table follows the current selection in [The Databases Window](#) or [The Model Manager Window](#) (depending on which window has focus). If you select a particular item, only commits involving that item are shown in the table; otherwise, all commits on the branch are shown. Click **Link with Selection** () to disable the automatic linking. You can still update the **Commits** window with a new database object by right-clicking the object and selecting **Commits** () when available.

THE COMMITS WINDOW TOOLBAR

The toolbar in the **Commits** window contains the following toolbar buttons:

- Click the **Refresh** button () to refresh the table to see any new commits that have been saved on the branch. The table will automatically refresh if you save a new commit to the branch from the COMSOL Desktop.
- Click the **Show More** button () to append older commits to the table.
- Click the **Link with Selection** button () to enable or disable whether to automatically update the **Commits** window based on the current selection in the **Databases** window or the **Model Manager** window (whichever had focus last).
- Click the **Commit Details** button () to open [The Commit Details Dialog Box](#).
- Click the **Branch** button () to open [The Create Branch Dialog Box](#) to branch off from the selected commit.
- Click the **Snapshot** button () to record a snapshot for the selected commit.
- Click the **Merge** button () to open [The Merge Window](#) to merge all changes made up to the selected commit to a target branch.
- Click the **Revert** button () to open [The Revert Window](#) for the selected commit.

If you right-click a commit in the table, you can also

- Select **Export** () to export all versions of items that were the latest versions at the time of the commit to the file system — see [Exporting Items](#).
- Select **Search in Commit** () to set the selected commit as the location in [The Model Manager Window](#). You can then search for all versions of items that were the latest versions at the time of the commit. See also [Searching in Snapshots and Commits](#).


THE COMMIT DETAILS DIALOG BOX


The **Commit Details** dialog box summarizes the set of related item changes saved in a commit.

- The **Location** shows the database, repository, and branch that the commit was saved to.
- The **Date** shows the time when the commit was saved.
- The **User** shows the name of the user that saved the commit.

- The **Comments** shows the optionally provided comments when the commit was saved.
- The **Changes** table shows a table with the set of related item changes made in the commit.






When you click **Revert** () for a commit, the inverse of the set of changes shown in the **Changes** table can be applied from [The Revert Window](#).

Click the **Edit Comments** button () to update the comments for the commit. Click **Save** to save the updated comments to the database. Click **Cancel** to revert back to the original comments.



For a Model Manager server database, only a root administrator or the user that saved the commit can update the comments.

From the toolbar for the **Changes** table, you can:

- Click the **Open** button () to open the latest version of a model at the time of the commit. Only enabled for a change that involved a model.
- Click the **Preview File** button () to open the latest version of a file using the default application for its file type. Only enabled for a change that involved a file.
- Click the **Version Details** button () to open [The Version Details Dialog Box](#) containing more information on a version.

See [Figure 2-3](#) for an example of commit details taken from the tutorial [Example: Modeling Using Version Control](#) in which a draft has been saved back as a new version of its original model.

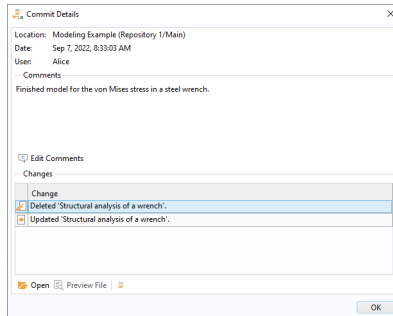


Figure 2-3: The Commit Details dialog box for a commit in which a new version has been saved of a model, and a draft has been deleted.



Activating a Database

You can select one of your configured databases from the menu in the **Database** section on the **Home** toolbar to *activate* it in the Model Manager workspace. This will:

- Automatically connect to the database if it was not already connected. You may be prompted for credentials if connecting to a server database.
- Set the location in [The Model Manager Window](#) and [The Commits Window](#) to a default branch from a repository in the database.
- Select the default branch in a repository in [The Databases Tree](#).

The Versions Window

The **Versions** window in the Model Manager workspace shows the version history of an item — that, is a model, file, or tag — with respect to a branch. These correspond to a subset of [Commits](#) on the branch in which the item was either added or updated. Click the link button to select another branch in [The Select Location Dialog Box](#).

The version history shown in the window follows the current item selection in [The Databases Window](#) or [The Model Manager Window](#) (depending on which window has focus). Click **Link with Selection** () to disable this automatic linking. You can still update the **Versions** window with a new item by right-clicking and selecting **Versions** ()


The versions are shown in a table sorted in chronological order. At most 100 versions are initially shown. Click the **Show More** button () to append older versions to the








table. If the branch was created from another branch, older versions saved on the parent branch are also included in the table. If the viewed model is opened in the COMSOL Desktop, the corresponding model version row is highlighted in bold in the table. As for [The Versions Window for the COMSOL Desktop Model](#), when viewing the version history of a model, the history of any origin model is also appended to the table — see [Splitting a Model Version History in Two](#).




The table columns are:

- The type column — the type of the item represented by an icon.
- The **Title** column — the title set for the item in that version.
- The **Saved** column — the time when the version was saved.
- The **Saved By** column — the name of the user that saved the version.
- The **Branch** column — the target branch of the commit in which the version was saved.
- The **Comments** column — the optional comment provided when the version was saved.








THE VERSIONS WINDOW TOOLBAR

The toolbar in the **Versions** window contains the following toolbar buttons:

- Click the **Refresh** button () to refresh the table in case any new versions have been saved. The table will automatically refresh if you save a new version on the branch from the COMSOL Desktop.
- Click the **Show More** button () to append older versions to the table.
- Click the **Link with Selection** button () to enable or disable whether to automatically update the **Versions** window based on the current selection in the **Databases** window or the **Model Manager** window (which ever had focus last).
- Click the **Version Details** button () to open [The Version Details Dialog Box](#) containing more information on the version.
- Click the **Open** button () to open a selected model version in the COMSOL Desktop.
- Click the **Run** button () to launch and run a selected version in the COMSOL Desktop. Only enabled if the selected version is an application.
- Click the **Preview File** button () to open a selected file version using the default application for its file type. See [Previewing Files](#).

- Click the **Compare** button () to compare a selected model version with the model opened in the COMSOL Desktop. Select two model versions to compare them with each other.
- Click the **References** button () to open [The References Window](#) to view all references between the selected version and other item versions.
- Click the **Restore Version** button () to save the selected version as a new latest version of the model. See [Restore Version](#).

If you right-click a version in the table, you can also

- Select **Export** () to export the version to the file system — see [Exporting Items](#).
- Select **Export to New Local Database** () to export the version to a new local database — see [Export to New Local Database](#).
- Select **Branch** () to create a new branch from the commit that the version was saved in.
- Select **Snapshot** () to record a snapshot of the commit that the version was saved in.
- Select **Copy Location** () to copy a text string with a URI that uniquely identifies a model or file version in the database to the clipboard. See [Copying Model and File Locations](#) for what you can do with this text string.
- Select **Permanently Delete Version** () to permanently delete a model or file version in the database. This cannot be undone. See also [Permanently Deleting a Version](#).
- Select **Settings** () to update the **Settings** window with the version. You may find this useful to see settings for an older version not available when searching with respect to a branch or snapshot.

THE VERSION DETAILS DIALOG BOX

The **Version Details** dialog box shows further details for an item version. The following is shown for all items:

- **Location** — the database, repository, and branch that the version was saved to.
- **Saved** — the time when the version was saved.
- **Saved By** — the name of the user that saved the version.
- **Title** — the title set on the item in the saved version.
- **Comments** — the optional comments provided when the version was saved.

For a model version:

- **Saved in** — the COMSOL Multiphysics versions that the model version was saved in.
- **Save Type** — if the model is a regular model or a draft model.
- **Item Type** — if it is a model, application, or physics.
- **Filename.** The filename used by the model version when exporting it to the file system.
- **Description.** The description of the model in the saved version.

For a file version:

- **Item Type** — if it is a file or fileset.
- **File size.** The size of the file version when stored on the file system. For a fileset, this is the total size of all files.
- **Description.** The description of the file in the saved version.

Click the **Edit Comments** button () to update the comments for the commit associated with the version — see also [The Commit Details Dialog Box](#). Click **Save** to save the updated comments to the database. Click **Cancel** to revert back to the original comments.




For a Model Manager server database, only a root administrator or the user that saved the version can update the comments.


The References Window



You use the **References** window to view and browse file versions that are *referenced* from a particular model version or, conversely, model versions that are *referencing* a particular file version. This enables you, for example, to discover model-file relationships without first having to open a model in the COMSOL Desktop.




The versions listed in the **References** window for a model version correspond to the subset of entries in the **Auxiliary Data** window whose **Location** column point to the database.


The references shown in the window follows the current model or file selection in [The Databases Window](#) or [The Model Manager Window](#) (depending on which window is focused). Click **Link with Selection** () to disable this automatic linking. You can still

update the **References** window with a new model or file by right-clicking and selecting **References** ()

You can view either referenced versions or referencing versions by clicking **Show Referenced Versions** () or **Show Referencing Versions** (), respectively. If you open the window for a selected model version, referenced versions are shown by default. For a selected file version, referencing versions are shown. The title of the selected version is shown at the top of the window.

The referenced or referencing versions are shown in a table sorted in chronological order. At most 100 versions are initially shown. Click the **Show More** button () to append older versions to the table. If a model version is opened in the COMSOL Desktop, the corresponding model version row is highlighted in bold in the table. The table columns are:

- The **type** column — the type of the item represented by an icon.
- The **Title** column — the title set for the item in that version.
- The **Reference Type** column — the type of reference between a model version and a file version describing in what way they are related.
- The **Saved** column — the time in which the version was saved.
- The **Saved By** column — the name of the user that saved the version.
- The **Repository** column — the repository that the version belongs to.
- The **Branch** column — the target branch in which the version was saved.
- The **Comments** column — the optional comment provided when the version was saved.














You can iteratively explore how different item versions depend on each other by repeatedly selecting a row in the table and clicking **References** ()




The database protects the *referential integrity* between item versions: a file version referenced by a model version, for example, cannot be permanently deleted, as long as both versions are saved in the same database.







THE REFERENCES WINDOW TOOLBAR

The toolbar in the **References** window contains the following toolbar buttons:

- Click the **Refresh** button () to refresh the table in case any new version references have been created. The table will automatically refresh if you save a new version to the database from the COMSOL Desktop.
- Click the **Show More** button () to append references to older versions to the table.
- Click the **Link with Selection** button () to enable or disable whether to automatically update the **References** window based on the current selection in the **Databases** window or the **Model Manager** window (which ever had focus last).
- Click the **Version Details** button () to open [The Version Details Dialog Box](#) containing more information on the version.
- Click the **Open** button () to open a selected model version in the COMSOL Desktop.
- Click the **Run** button () to launch and run a selected version in the COMSOL Desktop. Only enabled if the selected version is an application.
- Click the **Preview File** button () to open a selected file version using the default application for its file type. See [Previewing Files](#).
- Click the **Compare** button () to compare a selected model version with the model opened in the COMSOL Desktop. Select two model versions to compare them with each other.
- Click the **References** button () to show version references for the selected version.
- Click the **Show Drafts** button () to enable or disable whether to include version references that are drafts.
- Click the **Show Referenced Versions** button () to show versions that are referenced by the viewed version (as indicated by the title next to the toolbar). This is the default choice when viewing a model version.
- Click the **Show Referencing Versions** button () to show versions that are referencing the viewed version (as indicated by the title next to the toolbar). This is the default choice when viewing a file version.
- Select from the **Reference Types** menu () the type of references to include in the table. The available options are **Input File** and **Output File**.

If you right-click a version in the table, you can also


- Select **Export** () to export the version to the file system — see [Exporting Items](#).

- Select **Export to New Local Database** () to export the version to a new local database — see [Export to New Local Database](#).
- Select **Branch** () to create a new branch from the commit that the version was saved in.
- Select **Snapshot** () to record a snapshot of the commit that the version was saved in.
- Select **Copy Location** () to copy a text string with a URI that uniquely identifies a model or file version in the database to the clipboard. See [Copying Model and File Locations](#) for what you can do with this text string.
- Select **Permanently Delete Version** () to permanently delete a version in the database. This cannot be undone. See also [Permanently Deleting a Version](#).
You will only be able to perform this deletion for versions that are not referenced by other versions.
- Select **Settings** () to update the **Settings** window with the version.

REFERENCE-TRACKING IN MODEL MANAGER


Model Manager keeps explicit track of references between item versions when those versions are all stored in the same database. To see versions from other databases that are referenced by a model, open the model in the COMSOL Desktop and scan the **Auxiliary Data** window for input or output with a database set in the **Location** column.

References between a model version and other item versions are automatically discovered and tracked when a model is saved to a database. Model Manager looks for item versions referenced in the model tree of the saved model and matches them with item versions stored in the target database. This reference-tracking mechanism may lead to some surprising results that are good to be aware of:

- If you export output from the model opened in the COMSOL Desktop directly to a database, the resulting saved file version will not be referenced in the database *until* you also save a new version of the model in that database. See also [Loading and Saving Auxiliary Data Files Stored in Databases](#).
- If you update a file from the Model Manager workspace — for example via the **Settings** window — the saved file version will not be referenced by any model version, even if the previous file version was referenced. To see the references for the older file version, open [The Versions Window](#) for the file item, select the older version, and click the **References** button ()
- If you open the **References** window for a file version, you are likely to discover that it is referenced by a whole sequence of model versions. For a file version with

reference type **Output file**, the oldest model version in that sequence is likely the best candidate for reproducing the output.

Opening Models

You can open a model version in the COMSOL Desktop from [The Model Manager Window](#) or [The Databases Tree](#) by double-clicking a model version or by selecting a model version and clicking **Open** () in the **Item** section of the **Home** toolbar. You can also double-click or click **Open** from the toolbars of other windows in the Model Manager workspace that show model versions.

When you try to open a model version from the **Model Manager** window and the searched location is set to a branch, or from the **Databases tree** under a branch node, Model Manager first checks if any newer versions of drafts of the model exist on the same branch. If so, [The Open Dialog Box](#) is shown in which you can choose to open one of these newer draft versions instead. This helps you, for example, from accidentally creating a new draft when one is already ongoing — perhaps created by one of your coworkers.

THE OPEN DIALOG BOX

The **Open** dialog box shows a tree of model versions that Model Manager assumes may be of particular interest when opening a model version. The root node is the latest version of the model initially selected. Child nodes are the latest versions of drafts created from that model. Select one of the nodes and click **Open** to open a model version in the COMSOL Desktop.

PREVENTING SIMULTANEOUS ACCESS

While Model Manager has no concept of automatically *locking* a model for exclusive access by a single user, you can achieve the same effect via permissions — see [Granting Permissions](#). You can, for example, assign the **Open model** permission or **Save model** permission exclusively to yourself so that only you can open or save versions of a model.



If a simulation model is being developed in a collaborative setting, and you want to guarantee that only one person is modifying the model at a time, assign a **Protected** permission template and hand over the model by changing its owner — see [Owners](#).

OPENING A MODEL ON A CLUSTER

You can open a model stored in a database on a cluster as long as the cluster's root node can connect to the database. In the **Preferences** dialog box, select **From intermediate file** in the **Load model on nonroot nodes** list on the **Cluster** page under the **Model Manager** page to only require that the root node can connect. Select **Direct from database** in case also nonroot nodes can connect.





When **From intermediate file** is selected, the model is first saved as a temporary file on the root node's file system and then transferred to all nonroot nodes. When **Direct from database** is selected, all nodes load the model directly from the database without any intermediate step.

NETWORK CONNECTION ISSUES

When you open a model from a database, binary data such as meshes, geometries, solutions, and result plots are only loaded on-demand. This reduces, for example, unnecessary bandwidth usage when connected to a server database via a Model Manager server.

If you experience network connection issues when COMSOL Multiphysics needs to load binary data from the database, you will be asked to save any ongoing modeling work as a recovery on the file system. The model can be opened from recovery once the network connection is reestablished.

Running Applications

Click **Run** () in the **Item** section of the **Home** toolbar to launch and run an application version () from [The Model Manager Window](#) or [The Databases Tree](#). You can also click **Run** from the toolbars of other windows in the Model Manager workspace that show application versions.


Similar to when [Opening Models](#), Model Manager first checks if any newer versions of drafts of the application exist on the same branch. If so, [The Run Dialog Box](#) is shown in which you can choose to run one of these newer draft versions instead.

THE RUN DIALOG BOX

The **Run** dialog box shows a tree of versions that Model Manager assumes may be of particular interest when running an application version. The root node is the latest version of the application initially selected. Child nodes are the latest versions of drafts



created from that application. Select one of the nodes and click **Run** to launch and run an application version in the COMSOL Desktop.

Previewing Files

You can preview a data file by opening it with the default application for its file type. Double-click a file version, or select a file version in [The Model Manager Window](#) or [The Databases Tree](#) and click **Preview File** () in the **Item** section of the **Home** toolbar. You can also click **Preview File** from the toolbars of other windows that show file versions in the Model Manager workspace.

If the file version is a fileset, you will be asked to select which file to preview from the **Select File from Fileset** dialog box. Select a file from the table and click **OK**.


Comparing Models

You can compare the model opened in the COMSOL Desktop with a model version from the [The Model Manager Window](#) or [The Databases Tree](#) by selecting the version and clicking **Compare** () in the **Item** section of the **Home** toolbar. You can compare two model versions with each other by selecting two versions and clicking **Compare** (). You can also click **Compare** from the toolbars of other windows that show model versions in the Model Manager workspace.



Comparing Models Saved in Databases


Copying Model and File Locations




Model versions and file versions stored in a Model Manager database can be uniquely identified using an *item version location*. This is a text string URI that you can copy to your computer's clipboard by right-clicking an item version in the Model Manager workspace and selecting **Copy Location** ()



An item version location should not be confused with a commit location. The former uniquely identifies an item version stored in a Model Manager database; the latter uniquely identifies a commit in a Model Manager database — see also [Locations](#).


USING COPY LOCATION WITH MODELS



You can use **Copy Location** () to open a specific model version in another COMSOL Multiphysics program session:

- 1 Right-click a model version in the Model Manager workspace and select **Copy Location** ().
A text string URI that uniquely identifies the model version is copied to your computer's clipboard.
- 2 From the **File** menu in another COMSOL Multiphysics program session, select **Open From** (- 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** (

An identical **Clipboard** menu option also appears in the **Select Model** window when you have copied a model version location — see [Inserting Components, Materials, and Physics from Databases](#).

USING COPY LOCATION WITH FILES

For a file version, you can use **Copy Location** () to reference the file version from an input or output setting in the model opened in the COMSOL Desktop:

- 1 Right-click a file version in the Model Manager workspace and select **Copy Location** (). If the file version is a fileset, you will be asked to select which file location to copy in the **Select File from Fileset** dialog box. Select a file from the table and click **OK**.
A text string URI that uniquely identifies the file is copied to your computer's clipboard.
- 2 Open the Model Builder or Application Builder workspace and select a model tree node that has a **Filename** field with a **Location** menu() in the **Settings** window.
- 3 Paste the copied location into the **Filename** field.
The filename of the file, and the database the file belongs to, is displayed as a label in the **Filename** field. Depending on the input or output nature of the setting, you can now import or export data from or to the file in the database.



The **Copy Location** action is a good alternative to the **Select File** window when you want to reference an older version of a data file from a model.



- [Selecting Files in Databases as Input Sources](#)
 - [Selecting Files in Databases as Output Targets](#)
 - [Loading and Saving Auxiliary Data Files Stored in Databases](#)
-

Organization of Models and Files

Model Manager supports two means of organizing models and files in a database — assigning [Tags](#) to them and placing them in [Repositories](#). The first primarily helps you with search and workflow management, the second with access control.

In this section:


- [Tagging Models and Files](#)
- [Organizing Items in Repositories](#)

Tagging Models and Files


You can assign [Tags](#) to your models and data files to help you with organizing them in the database. You can even assign tags to the tags themselves, thereby building a tree structure of tags — see [The Tag Tree](#). In this regard, tags may remind you of folders on the file system. But tags go beyond that:

- You can find models and files by searching on their assigned tags. You can even find such items by searching on *ancestor* tags — that is, tags assigned to the item’s tags themselves. For example, if you create a tag `Interpolation Functions` and add it under the tag `Project A`, you may assign the former tag to data files and then find those data files by searching on the latter tag.
- You can assign multiple tags to a model, file, or even to a tag itself.
- The tag assignments are version controlled independently. A new commit is saved on the branch when you add or remove assigned tags to an item — see [Adding and Removing Tag Assignments](#). This has the benefit that:
 - You can track changes to the assigned tags of items over time, giving you an organizational history. Perhaps you introduce tags for different stages in your modeling workflow, say `In Progress`, `To Review`, `Approved`, and `Needs More Work`.
 - You can revert any changed tag assignments from [The Revert Window](#).
 - You can search items based on their tag assignments at a particular time by [Searching in Snapshots and Commits](#). Perhaps you want to find all items that had

a particular tag, say **In Progress**, a month ago, or merely see what the tag tree looked like. When [Searching in Branches](#), you search the present tag assignments.

- The settings for tags are version controlled. If you change the title of a tag and click **Save** () in the **Settings** window, a new version of the tag is created. Any tag assignments involving the saved tag will, however, be left unchanged.


CREATING NEW TAGS


Click **New Tag** () in the **Database** section of the **Home** toolbar to open the **New Tag** dialog box. The **Location** field shows the database, repository, and branch that the new tag will be created in.

- 1 In the **General** tab, write the title for the new tag in the **Title** field.
- 2 Select the **Add to selected models and files** check box if you want the new tag to be automatically assigned to the current selection of model and files in [The Model Manager Window](#) or [The Databases Window](#) (depending on which window has focus). The check box is disabled if no models or files are selected.
- 3 In the **Parent tags** tab, select tags in [The Tag Tree](#) that will be assigned to the new tag — in other words, its parent tags. Leaving all nodes in the tree cleared will create the new tag under the root. You can filter the tree of available parent tags by writing a tag title in the text field above the tree.

The tree is empty when you create your first tag.

- 4 In the **Comments** field, write an optional comment for the associated commit.
- 5 Click **OK** to save the first version of the new tag in the database.

If a tag version is selected in [The Model Manager Window](#) or [The Databases Window](#) when you click **New Tag** () , that tag will be initially selected as a parent tag in [The Tag Tree](#) in the **New Tag** dialog box.

You can see your new tag under the branch node in [The Databases Tree](#) if you have selected **Items** when clicking **Show** () in the **Databases** window's toolbar. You can also see it in the [Tree View](#) in [The Model Manager Window](#) as long as you have not written any search expressions in the search field or applied any filters.

THE TAG TREE

Since tags can be assigned to other tags, items in a Model Manager database can be represented as nodes in a tree structure. Items assigned a particular tag appear as child nodes to the tag node. You can find a particular item in multiple positions in the tree as items can be assigned more than one tag. You may think of items not assigned any



tags at all as placed under a hidden *root tag*. In [The Databases Tree](#), such items appear as children under the nodes of [Branches](#) and [Snapshots](#).



To help you with identifying the regular model that a draft originated from, the draft node is placed as a child node to the regular model node in the tag tree, whenever both the model and draft share the same assigned tag. See also [Save Types](#).

ASSIGNING TAGS TO ITEMS

To assign already created tags to an item, do one of the following:

- Select the item in [The Model Manager Window](#) or [The Databases Window](#) and click the **Set Tags** button () in the **Item** section of the **Home** toolbar. You can select multiple models and files if you want to set the tag assignments of more than one item at once — see also [Adding and Removing Tag Assignments](#).
- Set assigned tags from the **Tags** section in the **Settings** window of **Items**. Do not forget to click **Save** () in the **Settings** window to save any changed tag assignments.
- Set assigned tags from the **Tags** section in the **Save** window when saving a model version or in the **Export** window when exporting a file version — see also [Saving Models to Databases](#) and [Exporting Output Directly to a Database](#).
- Select to import folders as tags when bulk importing files from the file system — see also [Importing Files](#).

All four options sets the tag assignments via a commit. The second, third, and fourth option also save a new version of the item. The first option does not, which may be preferable as to not create unnecessary [Save Conflicts](#) if, for example, a model item happens to be simultaneously open in the COMSOL Desktop.

Organizing Items in Repositories


You can create multiple [Repositories](#) if you want to group your models and files into separate containers in the database. You may, for example, want to set up permissions

that differ between various collections of models — perhaps hiding sensitive models from all but a select group of users.



Keep in mind that you cannot simultaneously search models and files placed in different repositories. This may be good from an access control viewpoint but bad for discoverability.

ADDING REPOSITORIES

Click **Repository** () in the **Repository** section of the **Home** toolbar to open the **Add Repository** dialog box. The **Database** field shows the label of the database in which the repository will be created.

- 1 Write the name of the new repository in the **Name** field.
- 2 Write the name of the initial, default, branch in the **Default branch name** field — see [Default Branch](#).
- 3 You can provide an optional comment for the initial commit that will be saved when the branch is created in the **Comments** field.
- 4 You can set up permissions for the new repository in the **Permissions** field. This field is only shown if connected to a server database via a Model Manager server. See [Granting Permissions](#).
- 5 Click **OK** to add the repository.

The added repository appears as a new child node to the corresponding database node in [The Databases Tree](#).

Basic Version Control

You modify items — that is, models, files, and tags — in the database by saving a *commit* on a branch. The set of changes in such a commit can be categorized as:

- Added, updated, and restored items — these are changes that save new versions of items.
- Added and removed tags — these are changes that modify the assigned tags of an item.
- Deleted items — these are changes that effectively hide items on a branch.

Commits involving the first category of changes have a direct correspondence with the table entries in [The Versions Window](#) — each version is saved in a commit. The latter two categories, which do not save new versions, only appear in [The Commits Window](#).

You can provide an optional *commit comment* when saving a new commit. Oftentimes a comment describing the changes is already suggested. The comments can later be read and modified from [The Commits Window](#) or from [The Versions Window](#).




In this section:


- [Saving Versions](#)
- [Adding and Removing Tag Assignments](#)
- [Deleting Items](#)
- [Recording Snapshots](#)



Advanced Version Control

Saving Versions

There are various ways in which you can save a new version of an item, both within the Model Manager workspace and from one of the other workspaces. You can, for example, save a new version of the model opened in the COMSOL Desktop from the **File** menu by clicking **Save Draft** () , **Save as Version** () , or **Save To** () — see [Saving Models to Databases](#).

In the Model Manager workspace, you can save a new version by clicking **Save** () in the **Settings** window. For a model this will, for example, update the model's title, description, and filename, but leave the model tree, as well as any binary data like meshes, geometries, and solutions, unchanged. Other examples of saving versions include:

- Exporting output from a model directly to the database — see [Exporting Output Directly to a Database](#).
- Importing files on the file system into the database — see [Importing Files](#).
- Creating a new tag — see [Creating New Tags](#)
- Renaming an item — see [Rename Item](#)
- Restoring an older item version as a new latest version — see [Restore Version](#).
- Merging updated items from a source branch to a target branch — see [Merging](#).
- Reverting a commit that included added, updated, or deleted items — see [Reverting](#).




You generally want to avoid saving a new version of the model opened in the COMSOL Desktop from the Model Manager workspace. Otherwise you will get [Save Conflicts](#) when trying to save a new version of the desktop model.




[Creating a New Branch](#) does not save new versions of the items included on the new branch — each item version on the new branch is initially the same as that on the parent branch.


RENAME ITEM

From [The Model Manager Window](#) or the [The Databases Tree](#), right-click and select **Rename** () , or press the keyboard shortcut F2, to change the title of an item via the **Rename Item** dialog box. The **Location** field shows the database, repository, and branch that the item will be renamed on.

- 1 Write the new title for the item in the **Title** field.
- 2 Write an optional comment in the **Comments** field.
- 3 Click **OK** to save the renamed item as a new version in the database.

RESTORE VERSION

You can restore an older version in the version history of an item as the new latest version. This enables you to, for example, recover an older model version without first having to open it in the COMSOL Desktop and manually save it as a new version via **Save as Version** ()

From [The Versions Window](#), click **Restore Version** () to open the **Restore Item Version** dialog box. The **Location** field shows the database, repository, and branch that the item version will be restored to. The title of the restored version is shown in the **Title** field.

- 1 Write an optional comment in the **Comments** field.
- 2 Click **OK** to perform the restore.



Reverting Changes on a Branch

Restoring Versus Reverting


The different use cases for restoring a version and reverting a commit in which a version was saved are best exemplified via two examples:

- You have created a draft from a model and then saved ten versions of that draft. By the tenth version, you realize that the modeling setup has gone wrong for the draft, and that things started to go bad by version four. Solved by restoring version three from [The Versions Window](#) such that a version eleven, identical to version three, is saved directly in the database.
- You have decided to save what you, initially, think is a finished draft back to its original model. Such an action simultaneously saves a new version of the original model and deletes the draft. You then realize that you did this too soon — there were more things you wanted to first change in the draft. Solved by reverting the commit in which the draft was saved back to the original model from [The Commits Window](#): the draft is added back to the branch and the model is updated to its previous version before it was overwritten by the draft.

Adding and Removing Tag Assignments

You can modify the assigned tags of an item without saving a new version. This is useful if you, for example, want to change the tags of the model that is opened in the COMSOL Desktop without introducing [Save Conflicts](#). You can also modify the tag

assignments of multiple items at once — for example by assigning all of them with the same tag.

From [The Model Manager Window](#) or [The Databases Window](#), select one or more items and click the **Set Tags** button () in the **Item** section of the **Home** toolbar to open the **Set Tags** dialog box. The **Location** field shows the database, repository, and branch in which the assigned tags will be modified. The **Item** field indicates the item, or items, whose tag assignments will be modified.

- 1 Under **Tags**, select and clear check boxes for tags that will be added and removed, respectively. Leaving all nodes in the tree cleared will remove all tags from the item and place the item under the root tag — see also [The Tag Tree](#). You can filter the tree of available tags by writing a tag title in the text field above the tree.

If multiple items were selected when you opened the **Set Tags** dialog box, some check boxes may be shown in an *indeterminate state*. These correspond to tags that some, but not all, of the items were assigned prior to opening the dialog box.

Leaving the check boxes in their indeterminate state means that the corresponding tag assignments will *not* be modified — an item that already had the tag assigned will keep that tag, an item that did not have the tag will not gain it.


- 2 Write an optional save comment in the **Comments** field.
- 3 Click **OK** to save the new tag assignments to the database.

You can create a new tag from the **Set Tags** dialog box. Click **New Tag** () to open the **New Tag** dialog box — see [Creating New Tags](#). Once created, the new tag is added to the tag tree in the **Set Tags** dialog box in an initially selected state.

Deleting Items

You can delete items in the database without permanently losing them. In this way you can still access these items via older [Commits](#) and [Snapshots](#), without having to see them when you search the latest versions of items.

DELETING MODELS AND FILES

Select one or more models and files and click **Delete** () in the **Item** section of the **Home** toolbar to open the **Delete Item** dialog box. You can also press the Del key. The **Location** field shows the database, repository, and branch that the items will be deleted on. All items that will be deleted are shown in a table under **Items**.

- 1 Write an optional comment in the **Comments** field.

- 2 Click **OK** to delete the items in the database.



As with all commits, the deletion targets a specific branch. If you have created other branches that include the selected items, the items will remain visible on those branches.



To find a previously deleted model or file, either set the searched location to a snapshot created before the item was deleted in [The Model Manager Window](#), or find a commit in [The Commits Window](#) saved before the item was deleted, right-click it, and select **Search in Commit** (🔍). In both cases you can find the item by searching on its title in the **Model Manager** window.

DELETING TAGS

Select a tag and click **Delete** (🗑️) in the **Item** section of the **Home** toolbar to open the **Delete Tag** dialog box. The **Location** field shows the database, repository, and branch that the tag will be deleted on. The **Title** field shows the title of the latest version of the title.

- 1 In the **Tagged Items** field:

- Select **Remove Tag** to remove the tag from any items that the tag is assigned to. Items that only had the deleted tag will be placed under the root node in [The Tag Tree](#).
- Select **Add nearest ancestor tag** to remove the tag from any items that the tag is assigned to, and assign the parent tags to the deleted tag to these items instead.


- 2 Write an optional comment in the **Comments** field.


- 3 Click **OK** to delete the tag and assign new tags to tagged items as needed.

Recording Snapshots

You can save a reference to a particular commit in a repository by *recording* a snapshot. You may, for example, find snapshots useful when:

- You have reached a stage in your modeling workflow where a collection of model versions and file versions on a branch have reached a finished state. Via the snapshot, you can easily find all these versions in the future.
- You repeatedly want to browse in the database based on how it looked at the time of the commit. See [Searching in Snapshots and Commits](#).

Recorded snapshots are shown under the **Snapshots** node in [The Databases Tree](#). Click **Show** () in the **Databases** windows toolbar and select **Items** to see the correspondingly recorded item versions.

Click **Snapshot** () in the **Repository** section of the **Home** toolbar to open [The Record Snapshot Dialog Box](#). When [The Model Manager Window](#) is focused and the searched location is a branch, the latest versions will be recorded. The same is true when [The Databases Window](#) has focus and a node in a branch subtree is selected.



Snapshots

THE RECORD SNAPSHOT DIALOG BOX

The **Location** field shows the database, repository, and branch where the snapshot is recorded. The **Date** field shows the time of the corresponding commit that will be referenced by the snapshot.

- 1 Write a name for the new snapshot in the **Name** field.
- 2 You can set up permissions for the new snapshot in the **Permissions** field. This field is only shown if connected to a server database via a Model Manager server. See [Granting Permissions](#).
- 3 Click **OK** to record the new snapshot.

The recorded snapshot appears as a new child node to the **Snapshots** node in [The Databases Tree](#).

Bulk Operations

Model Manager supports bulk import and bulk export of models and data files to and from the file system.

In this section:


- [Importing Files](#)
- [Exporting Items](#)




Importing Files

You can import models and data files from the file system directly into a database without, for example, having to open each model in the COMSOL Desktop and manually save it. You can also choose to preserve any folder organization you made on the file system by importing folders as [Tags](#). Files inside a folder will then be assigned the corresponding tag.






Imported models will be converted to the current COMSOL Multiphysics version.

Click **Import** () in the **Database** section of the **Home** toolbar to open the **Import** dialog box. A **Target Location** for the imported files will be automatically suggested based on the current selection in [The Databases Window](#) or [The Model Manager Window](#) (depending on which window is focused). You can change the target by clicking the link button and choose another branch in [The Select Location Dialog Box](#).

- Click **Add Folder** () to add a folder containing files to import. Files in subfolders will also be included.
- Click **Add File** () to add a file to import.
- Click **Exclude File** () to exclude selected files from being imported.

The files to import are shown in a table with columns:

- The type column — the type of file to import based on its file extension represented by an icon. The types are () for mph, () for mphphb, and () for any other file extension.

- The **File** column — the filename of the file to import.
- The **Source** column — the directory path to the folder that was selected if added via **Add Folder** or the file path if added via **Add File**.

Click **OK** to begin the import. Clicking **Stop** in the progress window stops the import, but already imported files are not deleted from the database. You can see a summary of the import in the **Messages** window in the Model Builder workspace once the import finishes.



If you have a license for the CAD Import Module available when importing a CAD assembly file, an attempt will be made to automatically resolve and include external component files that the assembly references.

IMPORT OPTIONS

The import procedure can be configured under **Options** in the **Import** dialog box.

Select the **Include auxiliary files found in imported models** check box to automatically import referenced files located on the file system that are found inside an imported model. The model will be automatically updated so that it references the corresponding file version in the database.

In the **Tags** field:

- Select **None** to not assign any tags to imported files.
- Select **Import subfolders as tags** to also import all subfolders found under a directory path in the **Source** column as **Tags**. Imported models and files found in these subfolders will be tagged by the corresponding tags.
- Select **Use existing tags** to select tags to assign to imported models and files among tags already present in the database.




Importing a large number of files with a complicated folder structure and auxiliary file dependencies may require a bit of planning to get the desired result. You are encouraged to first create a throwaway local database in which you can experiment with your import — perhaps clicking **Stop** when only a subset of your files have been imported.



Model Manager will detect and skip *duplicate* files, that is files with the same filename and file content. This includes both files that you have explicitly added to the table in the **Import** dialog box, as well as referenced auxiliary files if you have selected **Include auxiliary files found in imported models**. If duplicate files are located in different subfolders, and you have selected **Import subfolders as tags**, the imported files will still be assigned tags corresponding to all those subfolders.


IMPORT TO TARGET TAG

You can import models and data files into the database so that all items are placed under a specific tag in [The Tag Tree](#).

Right-click a tag and select **Import** (). The **Import** dialog box opens with the title of the selected tag shown in a **Target tag** field. You add folders and files to import in exactly the same way as before — the only difference is that **Use existing tags** is not available under **Tags**.

Exporting Items


You can export models and data files from the database to the file system. Any tag organization you have made can be preserved on the file system by exporting tags as subfolders. Models and data files tagged by a specific tag will be exported to the corresponding subfolder.

Click **Export** () in the **Database** section of the **Home** toolbar to open the **Export** dialog box. The **Source Location** field shows the location from which versions of items will be exported. Click the link button or **Browse** button in the **Target folder** field to select a folder on the file system to which the items will be exported.

The models and files that will be exported are shown in a table with a type column and an **Item** column with the title of the item. The items in the table depend on the current selection in [The Databases Window](#) or [The Model Manager Window](#) depending on which window that is focused:

- Selecting a location — that is, a branch, snapshot, or commit — gives the latest versions of all items with respect to the corresponding commit of the location — see also [Locations](#).

- Selecting tags gives the latest versions of all items that are assigned those tags or are assigned a tag with a selected tag as an ancestor in [The Tag Tree](#).
- Selecting models or files gives their latest version with respect to the corresponding commit of the browsed location.

Select table rows and click **Exclude** () to exclude items in the table from being exported. Click **OK** to begin the export. You can see a summary of the export in the **Messages** window in the Model Builder workspace once the export finishes.



The files of a fileset will be exported to a directory whose directory name is the title of the fileset.

EXPORT OPTIONS

The export procedure can be configured under **Options** in the **Export** dialog box.

Select the **Include auxiliary files found in exported models** check box to automatically export any referenced files located in the database that are found inside an exported model. The model will be automatically updated so that it references the corresponding file on the file system.

Select the **Export tags as subfolders** check box to export tags assigned to exported items as subfolders on the file system, with exported files placed inside these subfolders.



If you select **Export tags as subfolders**, and an exported item version has multiple tags, only one of these tags will be exported as a subfolder (the first one when sorting tag titles alphabetically). This includes items that you have explicitly added to the table in the **Export** dialog box and any referenced auxiliary files if you have selected **Include auxiliary files found in exported models**.

User Management

Model Manager comes with tools for managing users, as well as groups of users, in a server database accessed via a Model Manager server.

In this section:

- [Managing Users](#)
- [Managing Groups](#)



All user management functionality is either hidden or disabled for a local database in the Model Manager workspace.

Managing Users

You can manage [Users](#) in a Model Manager server database from the Model Manager workspace. You can, for example, see all users that have been active in the database or add users as members to [Groups](#). The latter is useful, for example, if the Model Manager server has been set up with an authentication scheme that does not support providing such group memberships from an external credentials storage when a user logs in.


When you connect and authenticate with a Model Manager server, a new user will be automatically created in the database.



See also *Users* in the *Model Manager Server Manual*.

ADDING USERS

There may be situations where you want to add a new user to the server database before the user has connected for the first time. One example is when you preemptively want to grant them permissions to various database objects.

Click **User** () in the **Users** section on the **Database** toolbar to open the **Add User** dialog box. The **Database** field shows the label of the database in which the user will be added.

- 1 Write the username of the user in the **Name** field. This username should match the one used when authenticating with the Model Manager server.
- 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.
- 3 All groups that the new user will be set as a member of is shown as a list under **Group memberships**.
 - a Click **Add** to open the **Search** dialog box in which you can search for groups via their names and display names.
 - b Select groups in the table and click **OK** to add them to the list.Select groups in the list and click **Remove** to remove them from the list.
- 4 Click **OK** in the **Add User** dialog box to add the user to the database.

The added user appears as a new child node to the **Users** node under **Security** in [The Databases Tree](#).



The username in the **Name** field is the same one used to authenticate with the Model Manager server.


Managing Groups

You can manage [Groups](#) of users in a Model Manager server database from the Model Manager workspace. You can, for example, see all groups in the database or add a new group to the database. The latter is useful, for example, if the Model Manager server has been set up with an authentication scheme that does not support providing groups from an external credentials storage when a user logs in to the server.



See also *Groups* in the *Model Manager Server Manual*.

ADDING GROUPS

Click **Group** () in the **Users** section on the **Database** toolbar to open the **Add Group** dialog box. The **Database** field shows the label of the database in which the group will be added.

- 1 Write the name of the group in the **Name** field.

- 2 Write a display name for the new group in the **Display Name** field. This is the name that will primarily be shown in the user interface.
- 3 You can add existing users or other groups as members to the new group under **Group members**.
 - a Click **Add** to open the **Search** dialog box in which you can search for users and groups via their names and display names.
 - b Select users and groups in the table and click **OK** to add them to the list.
Select users and groups in the list and click **Remove** to remove them from the list.
- 4 Click **OK** in the **Add Group** dialog box to add the group to the database.

The added group appears as a new child node to the **Groups** node under **Security** in [The Databases Tree](#).

Access Control

You can set up permissions to control access to items stored in a server database accessed via a Model Manager server. You can define these at various levels in [The Databases Tree](#) — from coarse-grained permissions set on [Repositories](#) to fine-grained permissions set on individual [Models](#) and [Files](#).

When you authenticate with a Model Manager server, the server matches the provided username with a user stored in the database. Your user, as well as any groups that you are a member of, are checked whenever you perform a database action that requires authorization. The one exception is if you authenticate using a special *root administrator* account, in which case authorization checks automatically pass.

In this section, you will find answers to:

- Who can set permissions? See [Owners](#).
- How are permissions set? See [Granting Permissions](#)
- What permissions can be set for database objects? See [Permission Catalog](#).
- How do permissions combine to control access to models and files? See [Permission Levels](#).
- How can the same permission assignments be reused between different database objects? See [Reusing Permission Assignments Using Permission Templates](#).




All access control functionality is either hidden or disabled for a local database in the Model Manager workspace.

Owners

You control who can set permissions on database objects via *ownerships* — every object has an owner, which is one of the [Users](#) in the database. Except for root administrators, only the owner can change a database object's permissions. The user that creates an object is automatically set as its initial owner.

TRANSFER OWNERSHIP

You can transfer the ownership of a database object to another user if you own the object, or if you are authenticated as a root administrator.

Click **Owner** () in the **Permissions** section of the **Database** toolbar. The **Owner** dialog box is opened for the current selection in [The Databases Window](#) or [The Model Manager Window](#) (depending on which window is focused).

The **Database** field shows the label of the database that the object belongs to, the **Name** field shows the name of the object, and the **Current owner** field shows the username of the user that owns the object. Under **New Owner**, write the name or display name of the user to transfer ownership to in the search field. Click **Search**. Select one of the users in the table and click **OK** to save the new ownership.


Granting Permissions

You can grant permissions to the following database objects:

- [Repositories](#)
- [Branches](#)
- [Snapshots](#)
- [Models](#)
- [Files](#)
- [Tags](#)

There is also a set of global permissions for the database itself — see [Granting Database Permissions](#).

You can change the permissions for a database object if you own the object, or if you are authenticated as a root administrator.

Click **Permissions** () in the **Permissions** section of the **Database** toolbar. The **Permissions** dialog box is opened for the current selection in [The Databases Window](#) or [The Model Manager Window](#) (depending on which window is focused).

The **Database** field shows the label of the database that the object belongs to, the **Name** field shows the name of the object, and the **Current owner** field shows the username of the user that owns the object.

- 1 In the **Permissions** list:
 - a Select **None** to not set any permission requirements on the database object. See also [Permission Levels](#) to learn how an object may still be protected by a parent object in [The Databases Tree](#).
 - b Select **Public**, **Protected**, or **Private** to set one of the [Predefined Permission Templates](#).
 - c Select one of the previously created permission templates — see [Creating your own Permission Templates](#).
 - d Select **Custom** to set up [Custom Permissions](#) for the database object.
- 2 Click **OK** to save the permissions for the database object.

CUSTOM PERMISSIONS

Selecting **Custom** in the **Permissions** field enables you to customize the permissions for a database object. Granted permissions and their grantees — that is, users or groups — are shown in a table with the following columns:

- The type column — the type of the grantee represented by an icon.
- The **Name** column — the name of the grantee.
- The **Permissions** column — all granted permissions.

Click the **Add** button to add another user or group with a set of permissions.

- 1 In the opened **Add** dialog box, write the names or display names of users and groups in the search field. Click **Search**.
- 2 Select a user or group in the table.
- 3 Under **Permissions**, select the permissions to grant the user or group. Use **Select all** to grant all permissions.
- 4 Click **OK** to add the granted permissions and the grantee to the table.

Select a row in the table and click the **Edit** button to modify already granted permissions. In the opened **Edit** dialog box, select and clear the **Permissions** check boxes accordingly.

Click the **Remove** button to remove a grantee from the table.



The set of grantees and corresponding granted permissions is known as an *access-control list* (ACL).


EVERYONE AND OWNER

You can grant permissions to a special group, **Everyone**, that automatically includes all users. Use this when you, for example, want to grant a subset of permissions to all users but restrict another subset of permissions to only some users and groups.

The user that is the owner of a database object can be implicitly granted permissions by adding an **Owner** from the **Add** dialog box. Use this when you, for example, want to make sure that the permissions are transferred to the correct user when ownership is changed — see [Transfer Ownership](#).

GRANTING DATABASE PERMISSIONS

You can grant permissions for actions that target the database itself. You can think of this as delegating a few administrative tasks that would otherwise be restricted to root administrators. Only root administrators can grant database permissions.

Right-click a database node in [The Databases Tree](#) and select **Database Permissions** () to open the **Database Permissions** dialog box. The **Database** field shows the label of the database.

You add, edit, and remove granted permissions and their grantees in a table in the same way as when selecting **Custom** in the **Permissions** field for a database object, except that [Everyone and Owner](#) are not available — see [Custom Permissions](#).

Permission Catalog

This section contains a catalog of all permissions available for database objects.

DATABASE

TABLE 2-1: ALL AVAILABLE PERMISSIONS FOR THE DATABASE ITSELF.

PERMISSION	DESCRIPTION
Add repositories	Allowed to create a new repository.
Cleanup	Allowed to perform maintenance operations that involves various cleanup jobs (for example, data deduplication).
Delete computed data	Allowed to delete built, computed, and plotted data of models.
Index	Allowed to perform maintenance operations on search indexes.
Manage security	Allowed to create, delete, edit, and restore users, groups, and permission templates.
Permanently delete items	Allowed to permanently delete versions of models and files.
See disk space usage	Allowed to see the disk space usage of models.

REPOSITORY

TABLE 2-2: ALL AVAILABLE PERMISSIONS FOR REPOSITORIES.

PERMISSION	DESCRIPTION
Create branch	Allowed to create a new branch inside the repository.
Delete repository	Allowed to delete the repository.
Record snapshot	Allowed to record a new snapshot inside the repository.
Restore repository	Allowed to restore the repository when deleted.
Save in repository	Allowed to save commits as well as perform any other save operation inside the repository. This is, for example, a necessary permission for saving branches, snapshots, and item versions inside the repository.
Save repository	Allowed to save the repository itself to, for example, rename it.
See repository	Allowed to see the repository in the user interface. This is, for example, a necessary permission for browsing items inside the repository.
Set default branch	Allowed to set the default branch inside the repository.

BRANCH

TABLE 2-3: ALL AVAILABLE PERMISSIONS FOR BRANCHES.

PERMISSION	DESCRIPTION
Delete branch	Allowed to delete the branch.
Restore branch	Allowed to restore the branch when deleted.
Save in branch	Allowed to save commits as well as perform any other save operation inside the branch. This is, for example, a necessary permission for saving item versions on the branch.
Save branch	Allowed to save the branch itself to, for example, rename it or change search capabilities.
See branch	Allowed to see the branch in the user interface. This is, for example, a necessary permission for browsing all item versions on the branch.

SNAPSHOT

TABLE 2-4: ALL AVAILABLE PERMISSIONS FOR SNAPSHOTS.

PERMISSION	DESCRIPTION
Delete snapshot	Allowed to delete the snapshot.
Restore snapshot	Allowed to restore the snapshot when deleted.
Save snapshot	Allowed to save the snapshot itself to, for example, rename it.
See snapshot	Allowed to see the snapshot in the user interface. This is, for example, a necessary permission for browsing the latest item versions with respect to the snapshot's referenced commit.

MODEL

TABLE 2-5: ALL AVAILABLE PERMISSIONS FOR MODELS.

PERMISSION	DESCRIPTION
Delete model	Allowed to delete the model.
Open model	Allowed to open the model's versions.
Save model	Allowed to save versions of the model.

FILE

TABLE 2-6: ALL AVAILABLE PERMISSIONS FOR FILES.

PERMISSION	DESCRIPTION
Delete file	Allowed to delete the file.
Download file	Allowed to download the file's versions.
Save file	Allowed to save versions of the file.

TAG

TABLE 2-7: ALL AVAILABLE PERMISSIONS FOR TAGS.

PERMISSION	DESCRIPTION
Delete tag	Allowed to delete the tag.
Save tag	Allowed to save versions of the tag.

Permission Levels

When Model Manager authorizes a database action that targets item versions on a branch, it consults up to three levels of protection — the repository that the items belong to, the branch the item versions belong to, and, possibly, the items themselves. An analogous level consultation is done for item versions recorded by a snapshot.

PERMISSION COMBINATIONS FOR BRANCHES

The necessary permission combinations for possible database actions that target item versions on a branch are summarized as follows:

TABLE 2-8: NECESSARY PERMISSION COMBINATIONS FOR PERFORMING POSSIBLE DATABASE ACTIONS THAT TARGET ITEMS ON A BRANCH.

ACTION	REPOSITORY	BRANCH	ITEM
Browse and search item versions	See repository	See branch	N/A
Open a model in the COMSOL Desktop	See repository	See branch	Open model
Download data files	See repository	See branch	Download file
Create a new item	Save in repository	Save in branch	N/A
Assign tags to items	Save in repository	Save in branch	N/A
Save a new model version	Save in repository	Save in branch	Save model
Save a new file version	Save in repository	Save in branch	Save file
Save a new tag version	Save in repository	Save in branch	Save tag
Delete a model	Save in repository	Save in branch	Delete model

TABLE 2-8: NECESSARY PERMISSION COMBINATIONS FOR PERFORMING POSSIBLE DATABASE ACTIONS THAT TARGET ITEMS ON A BRANCH.

ACTION	REPOSITORY	BRANCH	ITEM
Delete a file	Save in repository	Save in branch	Delete file
Delete a tag	Save in repository	Save in branch	Delete tag



An important takeaway from [Table 2-8](#) is that if you grant a **See repository** permission and a **See branch** permission to users, they will be able to see settings of all models and files with versions on the branch via, for example, [The Model Manager Window](#), [The Databases Window](#), or the [The Versions Window](#). If some models contain sensitive information exposed through, for example, the **Contents** section in the [Model Settings](#), you must restrict access to the repository or branch to protect it. Limiting who can open an individual model should not be relied upon to keep its inner workings secret.

You may wonder why access to models and data files cannot be controlled via their assigned tags? After all, tags have many similarities with folders on the file system, and access to files on the file system can typically be controlled via folder permissions. The motivation is twofold:

- Models and files can have several tags, that is, they can appear in multiple places in [The Tag Tree](#).
- Tag assignments are version controlled via commits — see [Adding and Removing Tag Assignments](#).

This makes it hard to reason about the access granted to a model or file. An item could perhaps be protected under one tag's permissions but exposed under another tag's. An item could be protected under a tag at the present time, but older versions may be exposed in the commit history if the item was previously tagged by another tag with less restrictive permissions.


PERMISSION COMBINATIONS FOR SNAPSHOTS

The necessary permission combinations for possible database actions that target the item versions recorded by a snapshot are summarized as follows (there are no database actions that save to the database):

TABLE 2-9: NECESSARY PERMISSION COMBINATIONS FOR PERFORMING POSSIBLE DATABASE ACTIONS THAT TARGET THE ITEM VERSION RECORDED BY A SNAPSHOT.

ACTION	REPOSITORY	SNAPSHOT	ITEM
Browse and search item versions	See repository	See snapshot	N/A
Open a model in the COMSOL Desktop	See repository	See snapshot	Open model
Download data files	See repository	See snapshot	Download file

THE PERMISSION REQUIREMENTS DIALOG BOX

Model Manager performs a preemptive authorization check whenever you open a window or dialog box that is intended for saving to the database. If the check fails, a link button () whose button text summarizes why the check failed is shown.

Click the link button to open a dialog box with the necessary permission requirements shown in a table. Select the **Show only nongranted required permissions** check box to only see permission requirements that you lack.

Reusing Permission Assignments Using Permission Templates

You can create your own [Permission Templates](#) to reuse permission assignments for different database objects. This saves you the tedious work of manually granting the same set of permissions to users and groups for multiple database object. Creating your own permission templates also enables 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.



Updating the permission assignments for a permission template affects not only future applications of the template but also the permissions of database objects to which the template has already been applied.

PREDEFINED PERMISSION TEMPLATES


Model Manager comes with three predefined permission templates for each of the database object types — **Public**, **Protected**, and **Private**. Dividing the permissions in the [Permission Catalog](#) into *read* permissions and *write* permissions:

TABLE 2-10: THE PERMISSIONS GRANTED FOR THE PREDEFINED PERMISSION TEMPLATES.

PERMISSION TEMPLATE NAME	GRANTED READ PERMISSIONS	GRANTED WRITE PERMISSIONS
Public	Everyone	Everyone
Protected	Everyone	Owner
Private	Owner	Owner

CREATING YOUR OWN PERMISSION TEMPLATES

You can create new permission templates for models or files in the Model Manager workspace.

Click **Permission Template** () in the **Permissions** section of the **Database** toolbar to open the **Add Permission Template** dialog box. The **Database** field shows the label of the database in which the template is added.

- 1 Write the name of the permission template in the **Name** field.
- 2 Select whether to add a **Model** or **File** permission template in the **Type** field.
- 3 You add, edit, and remove granted permissions and their grantees in a table in the same way as when selecting **Custom** in the **Permissions** field for a database object — see [Custom Permissions](#).
- 4 Click **OK** to add the new permission template to the database.

The added permission template appears as a new child node to the **Permission Templates** node under **Security** in [The Databases Tree](#).

Maintenance

One of the biggest challenges with archiving simulation data is the inevitable growth of disk space usage, regardless if simulation data is stored in a database or on the file system. Oftentimes you will not be able to persist some, or all, of the [Built, Computed, and Plotted Data](#) generated by your models due to the sheer amount of storage capacity that would be needed.

In this section, you will find various ways in which Model Manager can help you keep the size of your databases under control via maintenance operations. Performing these maintenance operations for a server database accessed via a Model Manager server typically requires that you have authenticated using a root administrator account or have been granted the relevant database permissions — see [Granting Database Permissions](#).

- [Built, Computed, and Plotted Data](#)
- [Permanently Deleting Models and Files](#)
- [The Maintenance Window](#)

Built, Computed, and Plotted Data

One of the most significant contributions to the overall disk space required for storing your models comes from generated simulation data. This includes built geometries and meshes, computed solutions, and results plots — *built, computed, and plotted data* for short. For large models, it is often unfeasible to keep this data around for longer periods of time, preferring instead to regenerate the data by opening and recomputing a model in COMSOL Multiphysics.


There are two ways in which you can avoid storing built, computed, and plotted data in your databases:

- Exclude the data altogether when saving a version of the model to the database. See the [Save](#) section of [The Root Settings Windows](#) in the *COMSOL Multiphysics Reference Manual*.
- Delete the data from all versions of a model in the **Estimated Disk Space Usage** dialog box. See [Estimated Disk Space Usage](#).




You can see the COMSOL Multiphysics version that a model version was saved in from the **Saved in** field in [Model Settings](#). This is useful when you want to regenerate built, computed, or plotted data for a model version whose data is no longer present in the database.


Permanently Deleting Models and Files

You can permanently delete all versions of models and files in a Model Manager database by clicking the **Delete Permanently** button () in the **Item** section of the **Home** toolbar. You will be asked to confirm that you want to permanently delete all versions of the current selection of items in the [The Databases Window](#) or [The Model Manager Window](#) (depending on which window is focused). Any drafts that were created from selected models will also be permanently deleted.



Beware that this action cannot be undone. If you only intend to permanently delete all drafts of a model, and not the model itself, use the **Permanently Delete All Drafts** button () in [The Maintenance Window](#).

PERMANENTLY DELETING A VERSION



You can permanently delete a single version of a model or file by right-clicking the version in [The Versions Window](#) or [The Maintenance Window](#) and selecting **Permanently Delete Version** (). This is useful if a particular version is unsuitable to keep in the database.




Since Model Manager does not duplicate unchanged data when saving versions of a model, you may often find that permanently deleting a single version does not reclaim much disk space.

The Maintenance Window

The **Maintenance** window in the Model Manager workspace shows all versions in all repositories and branches that an item has been saved to and that you are authorized to see. You can use the window to get an overview of the total footprint that a model makes in a database.

The set of versions shown in the window follows the current item selection in [The Databases Window](#) or [The Model Manager Window](#) (depending on which window is focused). Click **Link with Selection** () to disable this automatic linking. You can still update the **Maintenance** window with a new item by right-clicking and selecting **Maintenance** ()

The versions are shown in a table sorted in chronological order. At most 100 versions are initially shown. Click the **Show More** button () to include older versions in the table. If the viewed model is opened in the COMSOL Desktop, the corresponding model version row is highlighted in bold in the table. The table columns are:

- The **Type** column — the type of the item represented by an icon.
- The **Title** column — the title set for the item in that version.
- The **Saved** column — the time when the version was saved.
- The **Saved By** column — the name of the user that saved the version.
- The **Repository** column — the repository that the version was saved in.
- The **Branch** column — the target branch of the commit in which the version was saved.
- The **Comments** column — the optional comment provided when the version was saved.
















When viewing the **Maintenance** window for a model, all versions of all drafts of the model are also included.









Viewing the table in the Maintenance window as “the version history” of a model can sometimes be misleading. Since versions of a model’s drafts are included in the table, and these drafts may have been worked on in parallel by multiple users, going from one version to the next in the table may not represent how the model has evolved over time. A similar observation holds when a model has been saved on multiple branches. The true version history of a model is best viewed in [The Versions Window](#).

THE MAINTENANCE WINDOW TOOLBAR


The toolbar in the **Maintenance** window contains the following toolbar buttons:

- Click the **Refresh** button () to refresh the table in case any new versions have been saved. The table will automatically refresh if you save a new version to the database from the COMSOL Desktop.
- Click the **Show More** button () to include older versions in the table.
- Click the **Link with Selection** button () to enable or disable whether to automatically update the **Maintenance** window based on the current selection in the **Databases** window or the **Model Manager** window (which ever had focus last).
- Click the **Version Details** button () to open [The Version Details Dialog Box](#) containing more information on the version.
- Click the **Estimated Disk Space Usage** button () to open the **Estimated Disk Space Usage** dialog box — see [Estimated Disk Space Usage](#).
- Click the **Open** button () to open a selected model version in the COMSOL Desktop.
- Click the **Run** button () to launch and run a selected version in the COMSOL Desktop. Only enabled if the selected version is an application.
- Click the **Preview File** button () to open a selected file version using the default application for its file type. See [Previewing Files](#).
- Click the **Compare** button () to compare a selected model version with the model opened in the COMSOL Desktop. Select two model versions to compare them with each other.
- Click the **References** button () to show version references for the selected version.
- Click the **Export to New Local Database** button () to export the version to a new local database — see [Export to New Local Database](#).
- Click the **Show Drafts** button () to include the versions of all drafts of a model in the table.
- Click the **Permanently Delete All Drafts** button () to permanently delete all drafts created from a model.
- Click the **Permanently Delete Selected Version** button () to permanently delete a selected version in the table.
- Click the **Permanently Delete All Versions** button () to permanently delete all versions of a model or file. For a model, all versions of all drafts created from the model will also be permanently deleted.

If you right-click a version in the table, you can also

- Select **Export** () to export the version to the file system — see [Exporting Items](#).
- Select **Branch** () to create a new branch from the commit that the version was saved in.
- Select **Snapshot** () to record a snapshot of the commit that the version was saved in.
- Select **Copy Location** () to copy a text string with a URI that uniquely identifies a model or file version in the database to the clipboard. See [Copying Model and File Locations](#) for what you can do with this text string.
- Select **Restore Version** () to save the selected version as a new latest version of the model. See [Restore Version](#).
- Select **Settings** () to update the **Settings** window with the version.

EXPORT TO NEW LOCAL DATABASE

You can export a single model version to a new local database by clicking the **Export to New Local Database** button () in the **Maintenance** window's toolbar. Select a name for the database directory in the **New Local Database** dialog box and click **Save**. You can then open the created database in the COMSOL Desktop — see [Opening a Local Database](#).

ESTIMATED DISK SPACE USAGE

The **Estimated Disk Space Usage** dialog box contains an estimate on the total disk space used by all versions of a model, including all versions of drafts created from the model.

The upper part shows statistics on the model when including all its drafts. The bottom part shows statistics for only the drafts. The **Versions** field shows the total number of versions on all branches. The **Data** field is the estimated disk space usage for all versions, and the **Computed Data** field is the estimated disk space usage for [Built, Computed, and Plotted Data](#).



Summing the estimated disk space usage of several models may result in an overestimate as identical files used by different models are never duplicated in the database, but will still be included in the usage estimate for each model.

- Click the **Permanently Delete All Versions** button to permanently delete all versions of the model, including all versions of all drafts created from the model.

- Click the **Permanently Delete Drafts** button to permanently delete all drafts created from the model.
- Click one of the **Delete Computed Data** buttons to permanently delete all **Built, Computed, and Plotted Data** of either all versions of the model, including all versions of all drafts created from the model, or only from all drafts created from the model.

Database Administration


In this section, you will find details on how you can administer your configured databases within the Model Manager workspace. You will also learn how to move, delete, and perform backups of databases stored locally on your computer.

- [Database Configurations](#)
- [Updating Search Index](#)
- [Database Cleanup](#)
- [Compacting Local Databases](#)
- [Moving and Deleting Local Databases](#)
- [Backup for Local Databases](#)

Database Configurations

You can manage added databases in the COMSOL Desktop from the top nodes in [The Databases Tree](#). Right-click and select **Delete Configuration** (✘) to remove an added database from the COMSOL Desktop. This will only delete the configuration for the database, not the database itself.

DATABASE SETTINGS

The **Settings** window for a database shows its configuration in the COMSOL Desktop. Click the **Save** button () to update the current configuration.

The General Section

This section shows the label of the database in the **Label** field.



The Storage Section

This section is only shown for a local database. The fields are:

- **Database file.** The path to the SQLite[®] database file on the file system.
- **Resources directory.** The path to the resources directory on the file system.
- **Search indexes directory.** The path to the search indexes directory on the file system.



[A Local Database on the File System](#)

Click the **Show in System Explorer** button () to open the database directory in the system explorer native to the operating system. Click the **Compact** button () to compact the database file — see [Compacting Local Databases](#).

The Connection Section

This section is only shown for a server database accessed via a Model Manager server. The fields are:

- **Server.** The server address to the Model Manager server.
- **Require secure connection.** Selected if the network connection is made using a secure connection, with transport layer security provided by HTTPS (as opposed to plain HTTP). A warning message is shown if the check box is cleared.
- **Username.** The username used when connecting to the server.
- **Password.** The password used when connecting to the server. The current password is not accessible.


The **Password** field is hidden when running COMSOL Multiphysics in client–server mode — see also [Connecting Via a COMSOL Multiphysics Server](#).


- **Remember password.** Selected if the password is remembered between program sessions. The password is stored in an encrypted form on the local file system.



Connecting to a Server Database




COMSOL Desktop disconnects from a connected Model Manager server when you click the **Save** button () in the **Settings** window. You can reestablish the connection by, for example, activating the database in the Model Manager workspace — see [Activating a Database](#).

Click the **Test Connection** button () to test if the filled-in values in the **Connection** section can be used to successfully connect to the Model Manager server.


Updating Search Index

Model Manager uses a specialized *search index* when searching and filtering items with respect to a branch. Under normal program execution, Model Manager automatically pushes any updates made to items in the database to this search index, both for a local database and for a server database. This push may fail, however — for a Model


Manager server, for example, due to intermittent network issues when the Model Manager server process tries to communicate with the process managing the search index.

You can manually trigger the push of changes to the search index, thereby letting the search index catch up with any changes made in the database. Right-click a database node in [The Databases Tree](#) and select **Index** () to push changes made on all branches. Do the same for a branch node to only push changes made on that branch.

Database Cleanup

When you permanently delete models or files in the database, some unused data may remain in the database. Click **Cleanup** () in the **Maintenance** section of the **Home** toolbar to instruct Model Manager to perform a cleanup of such data. This may take a few seconds up to several minutes depending on the size of the database.

Compacting Local Databases


You can optimize a local database by clicking **Compact** () in the **Maintenance** section of the **Home** toolbar to open the **Compact** dialog box. The **File size** field shows the current size of the SQLite[®] database file, and the **Estimated file size after compacting** field shows the estimated size after the file has been compacted. Click **OK** to compact the file. This may take a few seconds up to several minutes depending on the size of the database file.






[A Local Database on the File System](#)

Moving and Deleting Local Databases

You can move a Model Manager database stored locally on your computer to another file system location:

- 1 Open the [The Databases Window](#) in the Model Manager workspace and select the top database tree node for the database you want to move. In [The Settings Window](#), under **Storage**, note the file system location of your database. You can also click the **Show in System Explorer** button () to open the database directory in the system explorer native to the operating system.

- 2 Right-click the database tree node and select **Delete Configuration** () to delete the configuration for the database.
- 3 Move the directory containing your database to the new location on your computer's file system — see also [A Local Database on the File System](#).
- 4 Click the **Add Database** button () in the **Database** section on the **Home** toolbar.
- 5 Click the **Open Local Database** button () and select the SQLite® database file in the moved database directory — see also [Opening a Local Database](#). Model Manager will automatically connect to the local database in the new location.

To permanently delete a local Model Manager database, follow the previous steps but instead delete the database directory on the file system after you have deleted the configuration for the database.

Backup for Local Databases

Local Model Manager databases can be backed up using any regular backup software used for backing up the workstation. Make sure to include all files and subdirectories in the database directory as described in [A Local Database on the File System](#): the SQLite® database file, the `resources` directory, and the `indexes` directory. You may reduce the total disk space usage of your backups by optionally skipping the `indexes` directory as its file contents can always be recreated by Model Manager.



It may be tempting to place the database directory in a local folder on your computer that is automatically synchronized using cloud storage software with a remote cloud drive. This is not supported, however, and COMSOL Multiphysics will report an error if it detects such a file system path. For more information, see <https://www.comsol.com/support/knowledgebase/1295>.

To restore a previously backed-up database directory, copy the database directory and all its file contents from your backup location to your computer and follow the steps in [Opening a Local Database](#).

Searching and Filtering

In this chapter you will learn how to search for models and data files in a Model Manager database. You will see how you can match such items by performing both full text searches, as well as by applying various filter criteria. You will also learn how you can search “deep” within the model tree, matching models on their node properties, parameters, features, and other settings.

In this chapter:

- [Searching Versions](#)
- [Full Text Search](#)
- [Item and Content Filters](#)
- [The Model Manager Search Syntax](#)

Searching Versions

Whenever you search for models and files in a Model Manager database, you search within a select subset of versions. There are two possibilities:

- You search for the latest versions at the present time.
- You search for the versions that were the latest at some particular time in the past.

These two correspond to searching with respect to the following [Locations](#) in a database:

- [Searching in Branches](#)
- [Searching in Snapshots and Commits](#)

In this section you will learn how the search capabilities of Model Manager differ when searching these locations.

Searching in Branches

Select a branch in [The Select Location Dialog Box](#) to search for the latest versions of models and files in that branch. This is also the default location when you choose a database from the list on the left in the **Open**, **Select File**, and **Select Model** windows, or when [Activating a Database](#) in the Model Manager workspace.



The link button to open [The Select Location Dialog Box](#) is hidden in the **Open**, **Select File**, and **Select Model** windows when a database only contains a single branch and no snapshots.

Which search tools are available when searching for the latest versions of models and files is determined by the value selected in the **Search** list found in the **Settings** window for a branch — see [Branch Settings](#). Select **Item fields and content** to enable the full Model Manager search functionality for the branch. Select **Only text and tags** to limit the search to the same capabilities as when [Searching in Snapshots and Commits](#).

The **Item fields and content** value is the preferred and default selection for the main branch in a repository. You can then use all the search tools available in Model Manager, including applying [Item and Content Filters](#) from [The Filter Dialog Box](#) and writing custom filter queries using the [The Model Manager Search Syntax](#). There is,


however, a cost in terms of disk space usage by the corresponding search index — see also [Updating Search Index](#). You may want to select **Only text and tags** when:

- You create a new branch off the main branch that contains only a few models and files. Finding a particular model or file is then a matter of simply browsing a short list of items — that is, no advanced searching is required.
- You create a new branch off the main branch on which models and files will rarely be added or updated. Advanced searching can then be performed on the main branch — selecting the corresponding version on the new branch may then be done, for example, from [The Versions Window](#) or from [The Maintenance Window](#).




When [Opening a Local Database from Multiple COMSOL Multiphysics Processes](#), search will be limited to that used when [Searching in Snapshots and Commits](#) for all processes except the first one to connect.

Searching in Snapshots and Commits

Select a snapshot in [The Select Location Dialog Box](#), or right-click a commit and select **Search in Commit** () in [The Commits Window](#), to search for the versions of models and files that were the latest at the commit date. You may find this useful if, for example, you want to find a version of an old model or file that has been deleted, or if you perhaps want to see how the branch was organized with tags at the time of the commit.

Searching versions in a snapshot or commit is limited as compared to [Searching in Branches](#). You can perform full text searches by selecting **Text** or search on the titles of assigned tags by selecting **Tags** from the list next to the **Search** button in [The Model Manager Window](#). Other search windows offer only full text search. You can also apply a separate filter on file type extensions from a list in the **Select File** window. Other [Item and Content Filters](#) or the [The Model Manager Search Syntax](#) are not available.



The **Add Filter** button () is disabled in the **Open, Select File, Select Model, and Model Manager** windows' toolbars when you search with respect to a snapshot or commit.

Full Text Search

A fast way for you to find models and files in a Model Manager database is to write individual search words, separated by spaces, in the input fields of the **Open**, **Select File**, **Select Model**, or **Model Manager** windows and click the **Search** button. In this section, you will learn how Model Manager performs a *full text search* to match these search words against fields shown in [The Settings Window](#) for models and files.

In this section:

- [Combining Full Text Search Words](#)
- [Matched Fields](#)



Search words match in a case-insensitive way when searching in Model Manager.

Combining Full Text Search Words

Model Manager combines all search words that you write with AND-logic. In practical terms this means that the models and files included in a returned search result are such that all search words matched in the searched fields.

An example from the **Application Libraries** window: If you import `busbar.mph` found under **COMSOL Multiphysics>Multiphysics** into a database, it may, for example, have the following fields:

- **Title:** Electrical Heating in a Busbar
- **Description:** This example analyzes the resistive heating of a busbar designed to conduct direct current.
- **Tags:** Multiphysics

You can find this model by writing some or all of the search words: `busbar example multiphysics`. The first word, `busbar`, matches on a word in the title and the description, the second word, `example`, matches a word in the description, and the last word, `multiphysics`, matches an assigned tag.

You can use a *wildcard* asterisk character in a search word to match zero or more arbitrary characters when searching in a branch with **Item fields and content** configured — see [Searching in Branches](#). Appending, for example, a wildcard to a search word will

match that word against the beginning of a word or the whole word in the searched text. See also [Wildcard Matching](#).

You can also match on phrases — that is, multiple words in a sequence — by enclosing search words in quotation marks. Searching on "conduct direct current" will match in the description in the example, while other permutations of these search words will not match. See also [Phrase Matching](#).



Leading and trailing punctuation markers — for example, dots and hyphens — are ignored when searching words in a field of type [Text Field](#). That is why the trailing dot in `current.` can be omitted.

Including a wildcard asterisk character or using phrase quotation marks have no special meaning when [Searching in Branches](#) with **Only text and tags** configured or when [Searching in Snapshots and Commits](#). Model Manager will, however, automatically append an *implicit* wildcard to each search word in those cases.

Matched Fields

The fields matched against in a full text search differ slightly depending on the search capabilities of the searched location.



- [Searching in Branches](#)
 - [Searching in Snapshots and Commits](#)
-

MATCHING ITEM FIELDS AND CONTENT

Full text search words match the following [Model Settings](#) and [File Settings](#) when **Item fields and content** is configured for a branch:

- The **Title** field of a model or file.
- The **Description** field of a model or file.

- The **Filename** field of a model or file.
- The title of assigned **Tags**. Either tags assigned directly to an item or an ancestor to such a tag in [The Tag Tree](#).



Unlike when [Matching Only Text and Tags](#), the full text search does not match against the file type extension of a file. You can, however, search on such extensions using a [File Type](#) filter.

MATCHING ONLY TEXT AND TAGS

Full text search words match the following [Model Settings](#) and [File Settings](#) when **Only text and tags** is configured for a branch or when searching in a snapshot or in a commit:

- The **Title** field of a model or file.
- The **Description** field of a model or file.
- The **Filename** field of a model or file.
- The file type extension of a file as defined by its **Filename** field.



Selecting **Tags** instead of the default **Text** in the list next to the **Search** button in [The Model Manager Window](#) matches search words on the title of assigned tags — either tags assigned directly to an item or an ancestor to such a tag in [The Tag Tree](#).

Item and Content Filters

In this section you will learn how to apply the predefined filters available from the **Filter** dialog box. To learn how to write your own custom filters, see [The Model Manager Search Syntax](#).

- [Field Types](#)
- [The Filter Dialog Box](#)
- [Item Filters](#)
- [Content Filters](#)
- [Applied Filter Pills](#)



Filters match in a case-insensitive way in Model Manager.

Field Types

The [Item Filters](#) and [Content Filters](#) that are available in the Model Manager search tools can be categorized based on the types of fields that they match on. These *field types* affect, for example, how the field values specified in filters are interpreted when searching the database.

- [Text Field](#)
- [Keyword Field](#)
- [Date Field](#)
- [Numeric Field](#)
- [Boolean Field](#)
- [Selection Field](#)

TEXT FIELD

A filter on a *text field* matches search words in a text. Individual “words” in the text are obtained by splitting on spaces, punctuation markers, and other delimiter characters. Examples of text fields are the title and description of a model or file.

KEYWORD FIELD

A filter on a *keyword field* matches a search word against a string value. Keyword fields are typically found for short, name-like, search data, such as the filename of a model or file.

To include spaces in a keyword field's value, precede each space using a backslash character — see [Escaping Reserved Characters](#).



If you omit the backslash character before a space in the value for a keyword field, the value will be interpreted as two words combined with Boolean AND-logic — see [Basic Field Expressions](#). The search word `electrical heating busbar.mph` for a filename field is equivalent to `electrical AND heating AND busbar.mph` — an expression that can never match the filename of a model (a model can have at most one filename). Write `electrical\ heating\ busbar.mph` instead.

DATE FIELD

A filter on a *date field* matches against a date and time value. You can specify the date and time format according to the localization rules for the language set in the COMSOL Desktop or according to the ISO-8601 standard format. An example of a date field is the last modified timestamp of a model or file.



The following are all valid date and time values for a date filter matching August 24, 2021, 2:30 p.m. when running the COMSOL Desktop with language set to English:

- 8/24/21
- 8/24/21, 2:30:00 PM
- 2021-08-24
- 2021-08-24T14:30:00

Dates and times can either be matched exactly or as an interval. The intervals can be open on one or both sides. See also [Range Matching](#). You can also specify intervals from a select set of *date shorthands*. This includes, for example, `TODAY` for the current day and `CURRENTWEEK` for a date interval covering the current week. See [Table 3-6](#) for the complete list.

NUMERIC FIELD

A filter on a *numeric field* matches on a real or complex scalar value. An example of a numeric field is the last computation time of a study, or a numerical setting or parameter value. The unit to use for dimensioned scalar values depends on the context.

Numeric fields can either be matched exactly or by specifying intervals for the real and imaginary parts separately. The intervals can be open on one or both sides. See also [Range Matching](#).


BOOLEAN FIELD

A filter on a *Boolean field* matches on either `true` or `false`. An example is whether or not a 1D or 2D component is axisymmetric.

SELECTION FIELD

A filter on a *selection field* is a specialized filter that involves selecting values from a predetermined set. Examples include [Item Types](#) and [Save Types](#) of models or the user that last modified a model or file.

The Filter Dialog Box

You can apply filters to a model and file search from the **Filter** dialog box. Click the **Add Filter** button () from the toolbar of the **Open**, **Select File**, **Select Model**, or **Model Manager** windows. Select one of the [Item Filters](#) or [Content Filters](#) in the menu. The **Filter** dialog box is opened with fields and options appropriate for the selected filter.

The **Filter** field shows the name of the selected filter. The other input fields depend on which of the filters you selected.


Select any of the [Filter Options](#) under **Options** to quickly modify how the filter's field values will be combined and interpreted. You can see a preview of the corresponding filter query expression written using [The Model Manager Search Syntax](#) under **Filter query preview**.



Spaces entered in an input field for a [Text Field](#) in the **Filter** dialog box will be automatically replaced by Boolean OR operators by default — note the generated filter query expression under **Filter query preview**. This is in contrast to when writing a custom filter query using [The Model Manager Search Syntax](#) in which spaces are interpreted as Boolean AND if no Boolean operator is explicitly written.



Spaces entered in an input field for a [Keyword Field](#) in the **Filter** dialog box will be automatically escaped — see [Escaping Reserved Characters](#).

Click the **Customize filter query** button () to replace the filter query with your own customized query.

Click **OK** to apply the field values as a new filter.

FILTER OPTIONS

You can modify how filters are applied in the **Filter** dialog box by selecting one of the filter options under **Options**.

Select **Prefix match** to automatically append a wildcard to each search word in a [Text Field](#) or at the end of a [Keyword Field](#). See also [Wildcard Matching](#).



You include a wildcard by writing an asterisk character in a search word. The wildcard will match any number of arbitrary characters.

Select **Match all terms** to replace the OR-logic used between search words in an input field for a [Text Field](#) or selected check boxes in a [Selection Field](#) with AND-logic.

Select **Match whole phrase** to require that the search words match a sequence of words (that is, a *phrase*) in a [Text Field](#) or [Keyword Field](#). See also [Phrase Matching](#).



You match search words as a phrase by enclosing them in quotation marks.


Select **Negate match** to require that the specified field values do *not* match any models or files in the search result. See also [Negated Matching](#).

Item Filters


Item filters match on field values typically found in the **Version** section and **Tags** section for [Model Settings](#) and [File Settings](#).

- [Item Title](#)
- [Item Description](#)
- [Item Tag](#)
- [Item Type](#)
- [Save Type](#)
- [Last Modified](#)
- [Last Modified By](#)
- [Filename](#)
- [File Type](#)
- [Owner](#)

ITEM TITLE

The **Item Title** filter () is a [Text Field](#) filter that matches on the title of a model or file. You may find this filter useful when the searched title is a word commonly found also in descriptions and tags, such that performing a [Full Text Search](#) would yield too many results.

ITEM DESCRIPTION

The **Item Description** filter () is a [Text Field](#) filter that matches on the description of a model or file. Similar to an [Item Title](#) filter, you may find this useful when the searched description is also a common title or tag.


ITEM TAG

The **Item Tag** filter () is a [Selection Field](#) filter that matches on the assigned tags of a model or file. Select one or more tags in [The Tag Tree](#) shown in the **Filter** dialog box. Click **Clear Tags** () to clear all selections — the filter will then match on all items.




You will notice that the generated filter query under **Filter query preview** does not show the titles of the tags, but rather their unique keys in the database. This is to ensure that you find exactly the items tagged by the selected tag, and not items tagged by a tag that happen to share the same title.


ITEM TYPE

The **Item Type** filter () is a [Selection Field](#) filter on the [Item Types](#) of a model or file version. Select from **Model**, **Application**, **Physics**, **File**, and **Fileset**.


SAVE TYPE

The **Save Type** filter () is a [Selection Field](#) filter on the [Save Types](#) of a model or file (the latter only has regular as save type). Select from **Regular** and **Draft**.

LAST MODIFIED

The **Last Modified** filter () is a [Date Field](#) filter for when the latest version was saved. Select one of the date shorthands in the **Range** list to use a preset date interval. Select **Manual** and write a start date in the **From** field and an end date in the **To** field. Leave one or both input fields empty to not set a corresponding bound.


LAST MODIFIED BY

The **Last Modified By** filter () is a [Selection Field](#) filter on the user that saved the latest version. Write the name or display name of a user in the **Name** field. Spaces in names are automatically escaped with backslashes.

FILENAME

The **Filename** filter () is a [Keyword Field](#) filter on the filename used by a model or file when exported to the file system. The filename includes the file extension as a suffix.


FILE TYPE

The **File Type** filter () is a combined [Selection Field](#) and [Keyword Field](#) filter on the file extension of a file. Select a file extension from the **Predefined** field, or manually type file extensions, separated by spaces, in the **Manual** keyword field.



Unlike other [Keyword Field](#) filters in the **Filter** dialog box, in which spaces are escaped with backslash, spaces between file extensions will instead be automatically replaced with Boolean OR operators.

OWNER

The **Owner** filter () is a [Selection Field](#) filter on the current owner of a model or file. Write the name or display name of a user in the **Name** field. Spaces in names are automatically escaped with backslashes.

Content Filters

Content filters match on values in the [The Contents Section of Model Settings](#) — that is, on node properties, parameters, features, and other settings in the model tree of a model.

- [Parameter](#)
- [Setting](#)
- [Space Dimension](#)
- [Physics](#)
- [Study Step](#)
- [Node Type](#)
- [Label](#)
- [Name](#)
- [Tag](#)
- [Created](#)
- [Author](#)
- [Version](#)
- [Comment](#)
- [Last Modified](#)
- [Last Modified By](#)



[Searching Nodes and Settings in the Model Tree](#)

PARAMETER

The **Parameter** filter (P_1) is a combined [Keyword Field](#), [Text Field](#), and [Numeric Field](#) filter on parameters in a **Parameters** node. You can create a filter by combining:

- The parameter name as a [Keyword Field](#) in the **Name** field.
- The parameter description as a [Text Field](#) in the **Description** field.
- The parameter expression, including units, as a [Keyword Field](#) in the **Value** field.
- The parameter value, excluding units, as a real scalar [Numeric Field](#) in the **From** and **To** range fields. Leave one or both input fields empty to not set a corresponding bound.




The parameter value uses the unit system as set in the model when it was saved to the database.




[Parameters](#) in the *COMSOL Multiphysics Reference Manual*

SETTING


The **Setting** filter () is a combined [Keyword Field](#) and [Text Field](#) filter on general settings found in the **Settings** window for a node. You can create a filter by combining:

- The name of a setting as a [Keyword Field](#) in the **Name** field. This is an identifier of the setting that is not visible in the Model Builder, Application Builder, or Physics Builder **Settings** windows. You can find the name of a particular setting in the **Name** column in the **Details** dialog box for a node in [The Contents Section](#).
- The description of a setting as a [Text Field](#) in the **Description** field.
- The value of a setting as a [Keyword Field](#) in the **Value** field.
- The value of a setting when accessed via the COMSOL API as a [Keyword Field](#) in the **API value** field.



Not all node settings can be searched in the database. You can see which settings are available by, for example, finding a model that has the node type you want to search, select the node in [The Contents Section](#), click the **Details** button () , and examine the rows in the **Settings** table in the opened **Details** dialog box.


SPACE DIMENSION

The **Space Dimension** filter () is a [Selection Field](#) filter on the space dimension of a **Component** node. Select from **3D, Axial Symmetry (2D)**, **2D, Axial Symmetry (1D)**, **1D**, and **0D**.



[The Component Node](#) in the *COMSOL Multiphysics Reference Manual*

PHYSICS

The **Physics** filter () is a [Selection Field](#) filter on physics interfaces used by a model. You select physics interfaces you want to filter on from a tree. Select a space dimension in the **Show available physics for** field to filter the tree on physics interfaces available for that dimension. You can also filter the tree by writing the name of a physics interface and clicking the **Search** button.



[The Physics Nodes](#) in the *COMSOL Multiphysics Reference Manual*

STUDY STEP

The **Study Step** filter () is a [Selection Field](#) filter on study steps used by a model. Select the study steps you want to filter on from the shown list.




[Study and Study Step Types](#) in the *COMSOL Multiphysics Reference Manual*

NODE TYPE

The **Node Type** filter () is a combined [Text Field](#) and [Keyword Field](#) filter for finding models with nodes of a particular type. You can create a filter by combining:

- The node type as shown in the user interface as a [Text Field](#) in the **Type** field.
- The node type as used in the COMSOL API as a [Keyword Field](#) in the **API type** field.
- The node class as used in the COMSOL API as a [Keyword Field](#) in the **API class** field.



Click the **Model Tree Node Text** button () and select **Type** to see a node's type in the Model Builder tree.

You typically include a filter on the **API type** or **API class** fields to narrow the search when the **Type** field alone matches too wide.

NODE PROPERTIES

You can filter on node properties as shown in the **Properties** window for a node.

Label

The **Label** filter () is a [Text Field](#) filter on a node's label.

Name

The **Name** filter () is a [Keyword Field](#) filter on a node's name.

Tag

The **Tag** filter () is a [Keyword Field](#) filter on a node's tag.

Created

The **Created** filter () is a [Date Field](#) filter for when the node was created.

Author

The **Author** filter () is a **Keyword Field** filter on the node's **Author** field in the **Properties** window.



The value in a node's **Author** field is written as a general string value in the **Properties** window and need not correspond to any **Users** in a database.

Version

The **Version** filter () is a **Keyword Field** filter on the node's **Version** field in the **Properties** window.




The value in a node's **Version** field is written as a general string in the **Properties** window and does not correspond to any particular model version in a database.

Comment

The **Comment** filter () is a **Text Field** filter on the node's **Comments** field in the **Properties** window.

Last Modified



The **Last Modified** filter () is a **Date Field** filter for when the node was last modified. Except for the root node, this field is only available from the model object via the COMSOL API. Only a subset of all node types automatically update this value.

Last Modified By

The **Last Modified By** filter () is a **Keyword Field** filter on the last modified by value accessible from the model object via the COMSOL API. Only a subset of all node types automatically update this value.

Applied Filter Pills

All applied filters are shown above the search result in the **Open**, **Select File**, **Select Model**, and **Model Manager** windows, with each filter represented by a *filter pill*. The pill's text is a short summary of the filter.

Click a pill and select **Edit** () to edit an existing filter via **The Filter Dialog Box**. Select **Remove** () to remove a filter.

The Model Manager Search Syntax

The filter functionality in Model Manager is based on a tailor-made search syntax adapted for searching deep within a COMSOL Multiphysics simulation model. You may have seen examples of this syntax already in the **Filter query preview** field in [The Filter Dialog Box](#). The syntax enables you to write filter queries that find models whose node properties, parameters, features, and other settings satisfy arbitrarily complex constraints. As an example, you can write a custom filter query that matches all models that have an axisymmetric 2D component, for which the component uses a particular physics interface and a particular material.

In this section, you will learn how you can formulate such custom filter queries. Via examples of gradually increasing complexity, you will see how to write simple expressions for filters on single fields, how you can combine such field expressions using Boolean logic, and finally how you can nest field expressions based on the nested node hierarchy in the model tree. The section ends with a complete listing of all available field expressions in the Model Manager search syntax.

- [Basic Field Expressions](#)
- [Combining Expressions](#)
- [Searching Nodes and Settings in the Model Tree](#)
- [Search Syntax Catalog](#)



The Model Manager search syntax is only available when [Searching in Branches](#) and with **Item fields and content** configured.



The examples in this section will exclusively refer to [Models](#). You can also write analogous custom filter queries for [Files](#) using the item field expressions in the [Search Syntax Catalog](#).

Basic Field Expressions

A filter consists of one more *field expressions* combined with Boolean operators — for example, AND, OR, and NOT — and other grouping operators. Each field expression specifies which *field* is searched and the *field value* being searched on. The field also

has a particular type which dictates how the field value will be interpreted by Model Manager — see [Field Types](#).

You write a 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 fields in [Table 3-1](#), and *<field-value>* the value being filtered on. Write, for example,

```
@title:busbar
```

to find all models whose title contain the word `busbar`. To match on several search words, enclose the words with parentheses. Write

```
@title:(electrical busbar)
```

to find all models whose title contain both the words `electrical` and `busbar`.

A space between two search words is automatically interpreted as a Boolean AND in the Model Manager search syntax. The previous expression is thus equivalent to:

```
@title:(electrical AND busbar)
```

Write

```
@title:(electrical OR busbar)
```

if you want to find all models whose title contain either `electrical` *or* `busbar`.



You may have noticed that spaces in the input field for a [Text Field](#) in [The Filter Dialog Box](#) are automatically replaced by a Boolean OR in the corresponding **Filter query preview**. Select **Match all terms** under **Options** to change this to AND instead.

Field expressions can be written for other [Field Types](#) than a simple [Text Field](#). Write

```
@lastModified:9/17/21
```

to find all models last modified on September 17, 2021 using a [Date Field](#).



You can write the field name in a case-insensitive way. Thus `@lastmodified:9/17/21` works fine.

You can enter a custom filter query either directly in the search field in the **Open**, **Select File**, **Select Model**, and **Model Manager** windows, or apply it as a separate filter in [Applied](#)

[Filter Pills](#) by first clicking the **Customize filter query** button () from [The Filter Dialog Box](#).

If you want to combine full text search with a custom filter query, write the former first. The following is valid:

```
electrical busbar @description:example
```

and matches on all models with a title, description, filename, or assigned tags containing the words `electrical` and `busbar`, and that has a description containing the word `example`. The following is not valid:

```
electrical @description:example busbar
```

and results in an error message.



Write all plain search words first, followed by zero or more custom filter queries.

CONTROLLING PRECEDENCE USING PARENTHESES

A Boolean **AND** takes precedence over a Boolean **OR** in the Model Manager search syntax. The filter

```
@title:(electrical AND busbar OR tuning AND fork)
```

matches models whose title either contains both `electrical` and `busbar`, or whose title contain both `tuning` and `fork`. You can override this operator precedence with parentheses. Write

```
@title:(electrical AND (busbar OR tuning AND fork))
```

to match all models whose title contain `electrical`, and either the single word `busbar` or the two words `tuning` and `fork`.

WILDCARD MATCHING

You can use an asterisk character as a *wildcard* symbol that matches on zero or more arbitrary characters. The filter:

```
@title:electric*
```

matches on all models whose title begin with `electric`.

You can match on all models that have *any* value set for a field by using a single wildcard symbol. Write

```
@description:*
```

to find all models with a nonempty description. This can also be written using the special ANY symbol:

```
@description:ANY
```

which may feel more intuitive.



You can use a wildcard symbol anywhere in a search word. Placing it at the start (that is, a *postfix* search) may, however, result in slow query times. The exception is when the field value only contains a single wildcard symbol, and nothing else.



Wildcard matching on search words that include punctuation markers — for example, periods, commas, colons, and hyphens — may lead to surprising results when used in a full text search or in a [Text Field](#) filter due to how the search splits text into word tokens. You are recommended to avoid using wildcards for such search words. A [Keyword Field](#) filter does not have this limitation.

PHRASE MATCHING

You can match on phrases — that is, multiple words in a sequence — by enclosing them in quotation marks. Write

```
@title:"electrical heating"
```

to match on models whose title contains `electrical` followed by `heating`. You can also combine phrase search with ordinary search. Write

```
@title:("electrical heating" busbar)
```

to match on models whose title contains `electrical` followed by `heating`, as well as the word `busbar`, for example, `Electrical Heating in a Busbar`.



[Wildcard Matching](#) inside a phrase is not supported.

NEGATED MATCHING

You can reverse the match logic using the special NOT symbol. Write

```
@title:(NOT busbar)
```

to find all models whose title does *not* contain the word `busbar`. The `NOT` symbol takes precedence over both `AND` and `OR`, although you can override this precedence with parentheses. Write

```
@title:(NOT (electrical busbar))
```

to find all models whose title does not contain the words `electrical` and `busbar`.



You can exclude the parentheses if the `NOT` is followed by a single word. Thus `@title:NOT busbar` is equivalent to the first example.

RANGE MATCHING

A filter on a [Date Field](#) and a [Numeric Field](#) can be written as inclusive ranges. Write

```
@lastModified:[9/1/21 TO 9/30/21]
```

to find all models last modified for the month of September in 2021. Use a wildcard symbol for unbounded ranges. The filter

```
@lastModified:[* TO 8/31/21]
```

matches on all models that have not been modified after August 2021.

ESCAPING RESERVED CHARACTERS

Some characters serve special purposes in the Model Manager search syntax and are considered *reserved characters*. If you want to search on words that contain these characters, precede them by a backslash. Write

```
@tag:(\[In Progress\])
```

to match all models assigned the tag with title `[In Progress]`. The enclosing parentheses are necessary as there are two search words, `[In` and `Progress]`.

The ten reserved characters are:

```
{ } ( ) [ ] " : \ SPACE
```

The last one is a common pitfall when writing a filter on a [Keyword Field](#) that matches a string containing a space character. Write

```
@filename:electrical\ heating\ busbar.mph
```

to match a model with filename `electrical heating busbar.mph`. Parentheses are not necessary here because the value is considered as one search word when escaping the two space characters.

Combining Expressions

You can combine any number of field expressions to form more complicated filters. Write

```
@title:busbar @description:example
```

to find all models whose title contain the word `busbar` and whose description contain the word `example`. As for search words in the field value of a single field expression, a space character between two field expressions is interpreted as a Boolean AND. Write

```
@title:busbar OR @description:example
```

to combine the two field expressions with a Boolean OR.

You can also use parentheses to control operator precedence. Write

```
@saveType:draft AND (@owner:Alice OR @lastModifiedBy:Alice)
```

to find all drafts that are either owned *or* were last modified by user Alice.

Searching Nodes and Settings in the Model Tree

You can find models by matching on their node properties, features, and settings in the model tree. You use a similar search syntax for field expressions involving these nodes as when searching on general item fields — see [Basic Field Expressions](#).

In this section you will learn how to write basic filter queries that match on models with a particular node or setting, as well as advanced filter queries that match on models in which a particular node or setting is nested within other nodes.

BASIC NODE FIELD EXPRESSIONS

You write a basic *node field expression* filter using an @-notation of the general form:

```
@node{@<field-name>:<field-value>}
```

with *<field-name>* equal to the name of one of the available fields in [Table •](#) and *<field-value>* the value being filtered on. Write, for example,

```
@node{@type:rotor}
```

to find all models with nodes of a type that contains the word `rotor`. With this, you can, for example, find models that have a Frozen Rotor study step, or models that have a Beam Rotor interface. Write

```
@node{@type:(beam AND rotor)}
```


to narrow the search to the latter models.



You can leave out the Boolean AND as Model Manager automatically adds this whenever there is a space between two search words or two field expressions.

You can match a node on several properties, features, and settings by combining several expressions within `@node{...}`. Write

```
@node{@type:(beam rotor) @comment:"Use axial vibration"}
```

to find a model with a Beam Rotor interface for which a certain comment has been written in the **Comments** field in the **Properties** window for the node — see also [Phrase Matching](#).

A node field expression can also be combined with item field expressions, as well as with other node field expressions. Write

```
@lastModifiedBy:Alice @node{@type:(beam rotor)}
```

to find models with a Beam Rotor interface that were last modified by user Alice.

USING NAMED NODES

Oftentimes you want to find models with a node of a particular type. One challenge that you then face is that the **Type** field for a [Node Type](#) may match wider than you perhaps anticipated. Writing

```
@node{@type:(time dependent)}
```

will match on Time Dependent study steps. But it will also match on Time-Dependent Solvers, which may not be what you wanted. Write

```
@node{@apiClass:StudyFeature @type:(time dependent)}
```

to only match on the study step.

You can find the **API class**, as well as all other node fields, of a particular node in the **Node** table in the **Details** dialog box opened from [The Contents Section](#). These often have a technical name that may be challenging to remember. To help you with this, Model Manager comes with a set of predefined *named nodes* that work as aliases. You will match on only a Time Dependent study step by writing

```
@studyStep{@type:(time dependent)}
```

A complete listing of all named nodes is given in [Table 3-4](#).

You use an **API type** field expression when a named node is not enough to distinguish types. Writing

```
@physics{@type:(electric currents)}
```

will match on models with an Electric Currents interface. But it will also match on models with an Electric Currents in Layered Shells interface. Write

```
@physics{@apiType:ConductiveMedia}
```

to only match the former physics interface.

BASIC SETTING FIELD EXPRESSIONS

You write a basic *setting field expression* filter using an @-notation of the general form:

```
@setting{@<field-name>:<field-value>}
```

with *<field-name>* equal to the name of one of the available fields in [Table 3-3](#) and *<field-value>* the value being filtered on. As for node field expressions, you can include several expressions within @setting{...}. Write, for example,

```
@setting{@description:Length @value:9\cm\}}
```

to find all models with a setting **Length** having value 9[cm] — see also [Escaping Reserved Characters](#). When the setting is a scalar, you can also match on a range of values:

```
@setting{@description:Length @scalarReal:[0.05 TO 0.15]}
```



The scalar length values are written in the unit system of the model, unlike the textual length value in the previous example that matched on the exact string including the unit name.

You can find the available settings of a particular node under **Setting** in the **Details** dialog box opened from [The Contents Section](#). You may find it useful, for example, to include a filter on the **Name** setting field whenever the **Description** setting field matches too wide.

NESTED NODE AND SETTING MATCHING

You can write custom filter queries that match on settings for a particular node by nesting @setting{...} within @node{...}, or within any of the names nodes. Write

```
@parameters{@setting{@description:Length @scalarReal:0.09}}
```

to match a parameter setting on a **Parameters** node.

There is no limit on the number of setting field expressions you can include. Write, for example,

```
@parameters{@created:[9/1/21 TO 9/31/21]
@setting{@description:Length @scalarReal:0.09}
@setting{@description:Width @scalarReal:0.05}}
```

to find models with a **Parameters** node created in September 2021, such that the node has a **Length** parameter with value 9[cm] and a **Width** parameter with value 5[cm].

You can also nest two or more node field expressions inside each other. Write, for example,

```
@component{@spaceDimension:2 @physics{@apiType:ConductiveMedia}}
```

to find all models with a **2D** component that contains an Electric Currents interface.



The Model Manager search syntax supports nesting node field expressions five levels deep in the model tree.

Search Syntax Catalog

In this section, you will find a summary of all field expressions and symbols available in the Model Manager search syntax:

- [Table 3-1](#) contains item field expressions that match on basic fields of models and files.
- [Table 3-2](#) contains node field expressions that match on node types and properties.
- [Table 3-3](#) contains setting field expressions that match on node settings.
- [Table 3-4](#) contains the various expressions used to write plain and nested node and setting filters.
- [Table 3-5](#) contains symbols available in the search syntax.



If you are uncertain of what field values are available for a [Selection Field](#), write any value and press Enter. Model Manager will show an error dialog containing supported values.

TABLE 3-1: FIELD EXPRESSIONS FOR ITEM FIELDS.

SYNTAX	TYPE	DESCRIPTION
@title:...	Text	Matches on the title of an item.
@description:...	Text	Matches on the description of an item.
@tag:...	Text	Matches on the titles of tags assigned to an item.
@tagKey:...	Selection	Matches on the unique keys of tags assigned to an item.
@itemType:...	Selection	Matches on the item type of an item.
@saveType:...	Selection	Matches on the save type of an item.
@lastModified:...	Date	Matches on the last modified date of an item,
@lastModifiedBy:...	Selection	Matches on the name or display name of the user that last modified an item. Escape spaces in names with backslash.
@filename:...	Keyword	Matches on the filename used when exporting an item to the file system.
@fileType:...	Keyword	Matches on the file type extension of a file.
@owner:...	Selection	Matches on the name or display name of the user that owns the item. Escape spaces in names with backslash.

TABLE 3-2: FIELD EXPRESSIONS FOR NODE FIELDS.

SYNTAX	TYPE	DESCRIPTION
@label:...	Text	Matches on the label of a node.
@name:...	Keyword	Matches on the name of a node.
@tag:...	Keyword	Matches on the tag of a node.
@type:...	Text	Matches on the node type as shown in the user interface.
@apiClass:...	Keyword	Matches on the class of a node in the COMSOL API.
@apiType:...	Keyword	Matches on the node type as represented in the COMSOL API.
@created:...	Date	Matches on the date when the node was created.
@author:...	Text	Matches on the author property of a node.
@lastModified:...	Date	Matches on the last modified date of a node.
@lastModifiedBy:...	Text	Matches on the last modified by property of a node.
@version:...	Text	Matches on the version property of a node.

TABLE 3-2: FIELD EXPRESSIONS FOR NODE FIELDS.

SYNTAX	TYPE	DESCRIPTION
@comment:...	Text	Matches on the comment property of a node.
@lastComputationTime:...	Numeric	Matches on the last computation time of a Study node in seconds.
@spaceDimension:...	Numeric	Matches on the spatial dimension of a Component or Geometry node.
@axisymmetric:...	Boolean	Matches on the axisymmetric property of a Component or Geometry node.

TABLE 3-3: FIELD EXPRESSIONS FOR SETTING FIELDS.

SYNTAX	TYPE	DESCRIPTION
@name:...	Keyword	Matches on the identifier name of a setting.
@description:...	Text	Matches on the description of a setting.
@value:...	Keyword	Matches on the value of a setting.
@apiValue:...	Keyword	Matches on the value of a setting as represented in the COMSOL API.
@scalarReal:...	Numeric	Matches on the real part of a scalar setting value.
@scalarImag:...	Numeric	Matches on the imaginary part of a scalar setting value.

TABLE 3-4: NODE AND SETTING EXPRESSIONS.

SYNTAX	API CLASS	DESCRIPTION
@model{...}	Model	Matches on the root node of a model.
@parameters{...}	ModelParamGroup	Matches on a Parameters node.
@component{...}	ModelNode	Matches on a Component node.
@geometry{...}	GeomSequence	Matches on a Geometry node.
@material{...}	Material	Matches on a Material node.
@commonDefinition{...}	CommonFeature	Matches on a common definition node.
@physics{...}	Physics	Matches on a Physics node.
@multiphysicsCoupling{...}	MultiphysicsCoupling	Matches on a Multiphysics coupling node.
@mesh{...}	MeshSequence	Matches on a Mesh node.
@study{...}	Study	Matches on a Study node.
@studyStep{...}	StudyFeature	Matches on a Study Step node.

TABLE 3-4: NODE AND SETTING EXPRESSIONS.

SYNTAX	API CLASS	DESCRIPTION
@results{...}	Results	Matches on the Results node.
@nodeGroup{...}	NodeGroup	Matches on a Node Group .
@node{...}	N/A	Matches on an arbitrary node.
@setting{...}	N/A	Matches on a node setting.

TABLE 3-5: SYMBOLS IN THE MODEL MANAGER SEARCH SYNTAX.

SYNTAX	DESCRIPTION
*	A wildcard in a search word.
AND	Combines two field values or field expressions with a Boolean AND.
OR	Combines two field values or field expressions with a Boolean OR.
NOT	Negates a field value or field expression.
ANY	An alias for a field value containing a single wildcard.
(...)	Enclosing field values or field expressions.
{...}	Enclosing expressions inside a node or setting filter.
[... TO ...]	A field range value.
"..."	Quotation marks enclosing a sequence of search words as a phrase.

TABLE 3-6: DATE SHORTHANDS IN THE MODEL MANAGER SEARCH SYNTAX.

SYNTAX	DESCRIPTION
TODAY	Shorthand for an interval covering the current day.
YESTERDAY	Shorthand for an interval covering yesterday.
CURRENTWEEK	Shorthand for an interval covering the current week.
CURRENTMONTH	Shorthand for an interval covering the current month.
CURRENTYEAR	Shorthand for an interval covering the current year.

Advanced Version Control

In this chapter, you will learn how you can experiment with a collection of models and data files in a Model Manager database by creating an alternative, perhaps private, commit history. If your experiments are successful, you can then incorporate the updated models and files as new versions in the main commit history. You will also learn how to undo changes in your commit history.

In this chapter:

- [Branching](#)
- [Merging](#)
- [Reverting](#)

Branching

In this section you will see how you can create an alternative commit history in a repository by creating a new branch from an existing branch, also known as *branching off* from the existing branch. You may, for example, find branching useful when:

- You and your team of coworkers save your ongoing work on models and data files on a branch private to the team, and then publish the finished work on a branch shared with your organization. Perhaps there are multiple teams publishing to this latter branch.
- You start a project involving one or more models and data files already present in the database, but you are uncertain what the end result is going to be. Perhaps the modifications you plan to make will result in new versions of the existing models, or perhaps you want to create entirely new models. Perhaps you decide to discard the modifications altogether. By branching, you can postpone these decisions until the project is finished.
- You want to experiment with a single model in a way that is hidden from other users. You therefore forgo the more lightweight, and recommended, way of [Saving Drafts of Models](#) — opting to create a branch that uses a **Private** permission template,



Prefer [Repositories](#) over [Branches](#) if you want to create independent silos of models and data files.

In this section:

- [The Branch as a Sequence of Commits](#)
- [Creating a New Branch](#)


The Branch as a Sequence of Commits

An initial branch is automatically created when you add a new repository in a database. This is especially true for the initial repository automatically added when you create a new database. Any set of changes to items — that is, model, files, and tags — are saved in commits on this branch. See [Basic Version Control](#) for more details.

You may think of a branch as a sequence, or history, of such commits. Each commit identifies a collection of versions that were the most recently saved, or latest, at the

time of the commit. Any particular version could have been saved in that particular commit, or in a previous commit. Each commit also identifies the tag assignments to items at the time of the commit. By comparing the collections of item versions, as well as the assigned tags of items, present in two different commits, Model Manager can infer all item changes done in-between the first commit and the second commit. See [Figure 2-2](#) for a schematic representation of a branch containing three commits.



Use [The Select Location Dialog Box](#), or click branch nodes () in [The Databases Tree](#), to switch between branches in [The Model Manager Workspace Windows](#).


Creating a New Branch

You can select any commit on a branch to create a new branch from that *source commit*. This starts a new sequence of commits that runs in parallel with the first sequence. When the branch is created, the collections of versions and assigned tags will be identical in the source commit and on the new branch. But as soon as you start saving new versions and reassigning tags, the branches will diverge. See [Figure 4-1](#), which is a continuation of [Figure 2-2](#), for a schematic representation.



The Branch as a Sequence of Commits



- Prefer saving a draft over creating a new branch if all you want to do is experiment with a model, without necessarily affecting its version history. See [Saving Drafts of Models](#).
- Prefer using **Save as New** () if you want to create a copy of a model. Changes to the copy can later be merged to the original model via the **Comparison Result** window. See [Saving Models to Databases](#) and [Comparing a Version With the Opened Model in the COMSOL Desktop](#).

A good mental picture is to think of a tree in which the initial branch is the trunk of the tree, with an initial commit saved at the base of the trunk (at the ground), and successive commits stacked on top of each other. Other branches created off the main

branch correspond to tree branches shooting out from the trunk. A repository can be thought of as the tree itself.

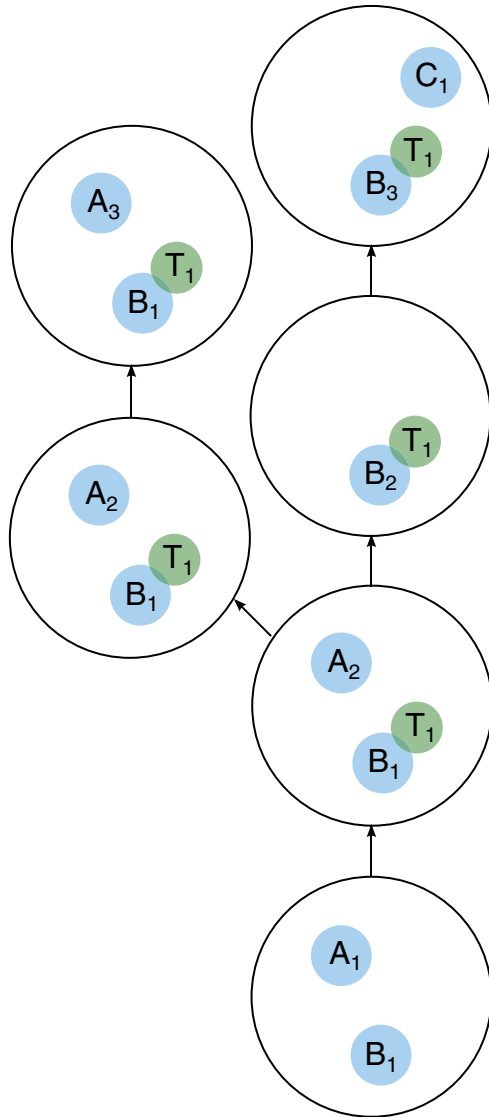







Figure 4-1: A schematic representation of a repository containing two branches. A second branch has been created off the main branch's second commit. In the third commit on the main branch, the model A was deleted. On the new branch's second commit, a new version of model A was instead saved. Browsing the latest versions on the main branch will return model versions B₃ and C₁. Browsing on the new branch will return A₃ and B₁.

To create a branch from a particular source commit, do one of the following:

- Select a branch node () in the [The Databases Tree](#) and click the **Branch** button () in the **Repository** section of the **Database** toolbar. The source commit is the latest commit on the branch.
- Select a snapshot node () in the [The Databases Tree](#) and click the **Branch** button () in the **Repository** section of the **Database** toolbar. The source commit is the commit that the snapshot references.
- Select a commit table row in the [The Commits Window](#) and click the **Branch** button () in the window's toolbar.

In all cases, [The Create Branch Dialog Box](#) is opened.



Creating a new branch is a fairly cheap operation in a Model Manager database in terms of actual data storage. No data is copied except for the small amount of metadata necessary for determining which items are initially present on the branch. There is, however, a cost in terms of disk space usage if a search index is created for the branch — see also [Searching in Branches](#).


PARTIAL BRANCHES

You can create a *partial branch* that contains a subset of all items that were present in the source commit. Select models and files in [The Databases Tree](#) or [The Model Manager Window](#) to include those items in the new branch. Select tags to include all items that are tagged by the selected tag. All tag assignments present in the source commit will be mirrored on the new branch as well.

THE CREATE BRANCH DIALOG BOX

You create the new branch from the **Create Branch** dialog box. The **Database** field shows the database in which the branch is created, and the **Repository** field shows the repository that the source commit and new branch both belong to.

- 1 Write the name of the new branch in the **Name** field.
- 2 Write an optional comment for the initial commit that will be made for the new branch in the **Comments** field.
- 3 Select **Item fields and content** in the **Search** list for complete search and filter support on the new branch; otherwise, select **Only text and tags** — see also [Searching in Branches](#).

- 4 In the **Selection** list:
 - Select **All** to include all items from the source commit.
 - Select **Current selection** to only include the items whose versions were selected when the dialog box was opened. The selected item versions are displayed in a table under the **Selection** field. Select a table row and click the **Exclude** button () to exclude the item from the new branch.
 - Select **Empty** to create a branch that does not include any initial items at all.
- 5 You can set up permissions for the new branch in the **Permissions** field. This field is only shown if connected to a server database via a Model Manager server. See [Granting Permissions](#).
- 6 Click **OK** to create the new branch in the database.

The created branch appears as a new child node to the **Branches** node in [The Databases Tree](#).



An empty branch is, for example, useful if you only intend to create new models on the branch. You can later merge these models to the source branch.



An initial commit is always made on a new branch. If you open [The Commit Details Dialog Box](#) for this commit you will see that the **Changes** table is, however, empty — no new versions or tag assignments are saved in the database in that initial commit.

Merging

After you have created a new branch and made changes to items on that branch, you will eventually want to transfer these changes back to its parent branch. In this section, you will learn how you can *merge* such changes to a branch.

- [Merging Changes to a Target Branch](#)
- [The Merge Window](#)






Merging Changes to a Target Branch

You merge item changes to a *target branch* by first selecting a source commit on a *source branch*. Changes made on the source branch up to and including the source commit will be made available for merging into the target branch. The merge itself will create a new commit on the target branch that includes all item changes you decided to merge.



To make two branches “equal” to each other in terms of their latest item versions, you first need to merge all changes from the latest commit on the first branch to the second branch, and then repeat this in the opposite direction.

To merge from a particular source commit, do one of the following:

- Select a branch node () in the [The Databases Tree](#) and click the **Merge** button () in the **Repository** section of the **Database** toolbar. The source commit is the latest commit on the branch.
- Select a snapshot node () in the [The Databases Tree](#) and click the **Merge** button () in the **Repository** section of the **Database** toolbar. The source commit is the commit that the snapshot references.
- Select a commit table row in the [The Commits Window](#) and click the **Merge** button () in the window’s toolbar.

In all cases, [The Merge Window](#) is opened with a suggested target branch.

The Merge Window




You use the **Merge** window to select and merge item changes made on a source branch, up to a particular source commit, to a target branch.




Merging Changes to a Target Branch

The **Source** field shows the location corresponding to the source commit for which the window was opened. Click the link button to select a new branch or snapshot as source in [The Select Location Dialog Box](#). The **Target** field shows the location corresponding to the target branch for the merge. Click the link button to select another branch.

The window contains a table with item changes made on the source branch that are not present in the target branch. The table columns are:

- The type column — the type of the changed item represented by an icon.
- The **Source Change** column — a description of the item change.
- The **Conflicting Target Changes** column — one or more changes on the target branch that are incompatible, or *in conflict*, with the source change.
- The **Selection** column — an icon representing whether to include the source change (), ignore the source change (), or if there is a conflict ().



Click the **Merge Changes** button () to open [The Merge Dialog Box](#) after you have decided which changes to merge and resolved any merge conflicts.





Resolving Merge Conflicts

THE MERGE WINDOW TOOLBAR

The toolbar in the **Merge** window contains the following toolbar buttons:




- Click the **Refresh** button () to refresh the table in case any new commits have been saved on the target branch.
- Click the **Take Source** button () to include a source change in the merge. This is the default choice.

- Click the **Keep Target** button () to ignore a source change, keeping the target as is.
- Click the **Merge Changes** button () to open the [The Merge Dialog Box](#).

RESOLVING MERGE CONFLICTS


When you work with a collection of items with versions on multiple branches, you will inevitably encounter conflicting changes when merging from a source branch to a target branch. Such merge conflicts can arise, for example, when:

- Versions of the same item have been saved on both branches.
- An item has been saved on one branch but has been deleted on the other branch.
- An item has been assigned a tag on one branch, but that tag has been deleted on the other branch.


Merge conflicts are indicated in the **Selection** icon column by (). Select the table row and click the **Take Source** button () to overwrite all conflicting changes on the target branch with the corresponding source change. Click the **Keep Target** button () to keep the target unchanged by skipping the source change.



Merging Conflicting Model Updates


The all or nothing choice for including a source change may be too coarse when a model has been updated on both branches. The model version on the source branch and the model version on the target branch can contain independent updates to the model tree, and it would then make sense to incorporate both updates in a *merged model version* on the target branch. You can proceed as follows:

- 1 Open the model version on the target branch in the COMSOL Desktop.
- 2 Select the model version on the source branch in [The Model Manager Window](#) and click the **Compare** button () in the **Item** section of the **Home** toolbar.

The **Comparison Result** window is opened with a comparison between the model in the COMSOL Desktop — that is, the target model version — and the selected source model version.

- 3 Merge the changes you want to keep from the source model version into the opened model using the merge functionality in the **Comparison Result** window.
- 4 From the **File** menu, select **Save as Version** ().
- 5 Save a new version of the model to the target branch from the **Save** window. This becomes the merged model version.

- 6 Click the **Refresh** button () in the **Merge** window's toolbar to recompute source and target branch changes.
- 7 If there is still a conflict between the source branch and target branch for the model update, which is expected, select the row and click the **Keep Target** button () to keep the merged model version on the target.

If there is more than one update conflict between a model version on the source branch and the target branch, repeat these steps for each one. Once finished, finish the merge by clicking **Merge Changes** ().

THE MERGE DIALOG BOX

The **Merge** dialog box gives you a final chance to either go through with the merge or cancel it (except for any manually merged model versions already saved on the target branch — see [Merging Conflicting Model Updates](#)). The **Source location** field shows the location of the source commit, and the **Target location** field shows the target branch. You can write an optional comment in the **Comments** field for the commit created by the merge.

The table shows all changes that will be applied to the target branch by the merge commit. These changes may differ from the original source changes shown in the **Merge** window depending on which changes were included, which were skipped, and potential merge conflict resolutions.

Click **OK** to merge the changes in the database.



Once a source commit has been merged into a target branch, any changes on the source branch that were skipped will not show up in the **Merge** window the next time you open the window for a newer source commit. To include such older source changes, manually perform them on the target branch.



You may find the number of source changes in the **Merge** window overwhelming if there has been many commits on the source branch. One solution is to first merge from an older commit on the source branch, and then progressively work yourself up to the latest commit. One drawback is that this requires more than one merge commit on the target branch, which may unnecessarily pollute the commit history on that branch.

Reverting

In this section, you will learn how to undo some or all of the changes saved in a commit. This is valuable, for example, when you want restore a previously deleted model or file, revert to the version of an item that existed before a commit, or even revert a merge.


- [Reverting Changes on a Branch](#)
- [The Revert Window](#)

Reverting Changes on a Branch

When you revert a commit on a branch, Model Manager takes the set of changes done in the commit and computes the corresponding reverse changes. These changes are then saved as a new commit on the branch. A created item will be deleted, a deleted item will be recreated, and an updated item will be replaced by its previous version. A tag added to an item will be removed, and a tag removed from an item will be added.



There is a subtle difference between restoring a version — see [Restore Version](#) — and reverting a commit in which a version was saved. When selecting and restoring a version, you save *that* version as the new latest version. When selecting and reverting a commit, you save the version that *preceded* the commit's version as the new latest version.

Select a commit table row in [The Commits Window](#) and click the **Revert** button () in the toolbar to open [The Revert Window](#) for the selected commit.




The Revert Window


You use the **Revert** window to select and apply changes to a branch such that they revert changes made in a commit.



[Reverting Changes on a Branch](#)

The window contains a table with the reverting item changes. The table columns are:

- The type column — the type of the item to change represented by an icon.
- The **Change to Apply** column — a description of the reverting item change.
- The **Conflicting Latest Changes** column — one or more later changes on the branch that are incompatible, or *in conflict*, with a reverting change.
- The **Selection** column — an icon representing whether to apply the reverting change () , ignore the change () , or if there is a conflict () .





Click the **Apply Revert** button () to open [The Apply Revert Dialog Box](#) after you have decided which changes to apply and resolved any revert conflicts.



Resolving Revert Conflicts

THE REVERT WINDOW TOOLBAR




The toolbar in the **Revert** window contains the following toolbar buttons:

- Click the **Refresh** button () to refresh the table in case any new commits have been saved on the branch.
- Click the **Take Change to Apply** button () to apply a reverting change to the branch.
- Click the **Keep Latest Change** button () to skip a reverting change, keeping the latest change as is.
- Click the **Apply Revert** button () to open the [The Apply Revert Dialog Box](#).

RESOLVING REVERT CONFLICTS





When you revert a commit on a branch, you may discover that the reverse changes are in conflict with the current item versions on the branch. Such revert conflicts can arise, for example, when:


- Reverting a commit in which an item version was saved, but a newer version of the item exists on the branch.
- Reverting a commit in which an item version was saved, but the item has been deleted on the branch in a newer commit.
- Reverting a commit in which an item was assigned a tag, but the tag has itself been deleted on the branch in a newer commit.

Revert conflicts are indicated in the **Selection** icon column by (). Select the table row and click the **Take Change to Apply** button () to overwrite all conflicting latest changes on the branch with the corresponding reverting change. Click the **Keep Latest Change** button () to keep the target unchanged by skipping the reverting change.

Resolving Conflicting Model Versions

Unlike merging from a source branch to a target branch, you may see less need to manually merge the model trees of conflicting model versions when reverting a commit on a branch — see [Merging Conflicting Model Updates](#). Nevertheless, you can proceed as follows:

- 1 Open the older model version you want to revert to in the COMSOL Desktop.
- 2 Select the latest model version on the branch in [The Model Manager Window](#) and click the **Compare** button () in the toolbar.
The **Comparison Result** window is opened with a comparison between the model in the COMSOL Desktop — that is, the older model version to revert to — and the latest model version.
- 3 Merge the changes you want to keep from the latest model version into the opened model using the merge functionality in the **Comparison Result** window.
- 4 From the **File** menu, select **Save as Version** ().
- 5 Save a new version of the model to the branch from the **Save** window — this becomes the reverted model version. There will inevitably be [Save Conflicts](#) with the latest model version. Choose to ignore them.
- 6 Click the **Refresh** button () in the **Revert** window's toolbar to recompute changes.
- 7 If there is still a conflict between the version being reverted to and the latest version for the model, which is expected, select the row and click the **Keep Latest Change** button () to keep the latest model version.

If there is more than one conflict between an older model version being reverted to and a latest model version, repeat these steps for each one. Once finished, finish the revert by clicking **Apply Revert** ().

THE APPLY REVERT DIALOG BOX

The **Apply Revert** dialog box gives you a final chance to either go through with the revert or cancel it (except for any manually reverted model versions already saved on the target branch — see [Resolving Conflicting Model Versions](#)). The **Location** field

shows the branch on which the revert is made. You can write an optional comment in the **Comments** field for the commit created by the revert.

The table shows all changes that will be applied to the branch by the revert commit. These changes may differ from the original changes shown in the **Revert** window depending on which changes were included, which were skipped, and potential revert conflict resolutions.

Click **OK** to apply the changes in the database.

Working with Models in Databases

This chapter showcases the Model Manager tools available in the COMSOL Desktop modeling environment by way of a few tutorials. You will, for example, learn how to open and save versions of models in a database, how to browse, organize, and search models in a database, and how to use advanced version control tools such as reverting commits and creating branches.

In this chapter:

- [Example: Modeling Using Version Control](#)
- [Example: Browsing, Organizing, and Searching Models and Data Files](#)
- [Example: Using Advanced Version Control Tools in the Model Manager](#)




Example: Modeling Using Version Control

This section introduces some of the Model Manager tools you will typically encounter when working with models and data files stored in databases. The section uses a streamlined version of the tutorial *Example 1: Structural Analysis of a Wrench* found in *Introduction to COMSOL Multiphysics*. You are encouraged to first work through that tutorial if you are new to the COMSOL Multiphysics software.

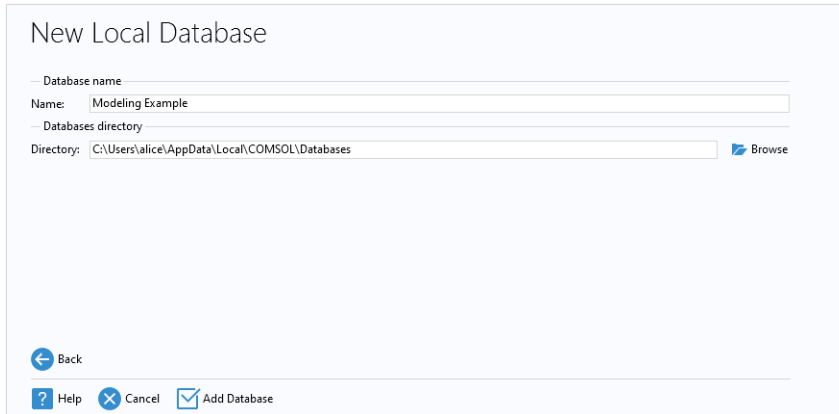
- [Creating the Database](#)
- [Model Wizard Setup](#)
- [Saving a First Version](#)
- [Saving More Versions](#)
- [Working With a Draft of the Model](#)
- [Comparing Versions](#)
- [Excluding Built, Computed, and Plotted Data](#)
- [Importing Auxiliary Data to the Database](#)
- [The Model Manager Workspace](#)

Creating the Database

You are strongly recommended to create a new local database when working through the steps of this tutorial.

- 1 From the **File** menu, select **Open From** ()
- 2 In the **Open** window, choose **Add Database** () in the list of options.
- 3 In the **Add Database** window, choose **New Local Database** ()

- 4 In the **New Local Database** window, write **Modeling Example** as the name for the new database in the **Name** field.





New Local Database




— Database name

Name: Modeling Example

— Databases directory


Directory: C:\Users\alice\AppData\Local\COMSOL\Databases 

 Back

 Help  Cancel  Add Database

- 5 Click **Add Database** ().

A progress window is displayed informing you that the database is being created on the file system. Once finished, the **Open** window is shown with the newly created database selected in the list.





From the **Open** window, you can search the database for models to open. At the moment your database is empty. Click **Cancel** ().



Adding Databases

Model Wizard Setup


You will use the **Model Wizard** to quickly get started with a new model for the structural integrity of a wrench.

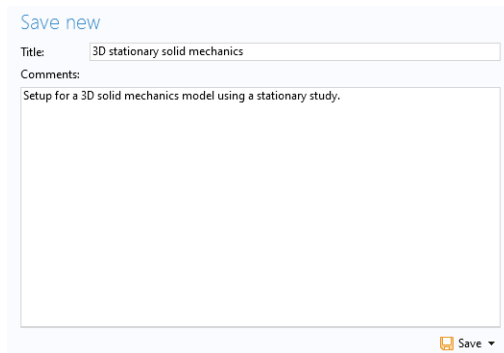
- 1 From the **File** menu, select **New**. Choose **Model Wizard** in the **New** window.
- 2 In the **Select Space Dimension** window, choose **3D**.
- 3 In **Select Physics**, select **Structural Mechanics>Solid Mechanics (solid)** (). Click **Add**. Click **Study** () to continue.
- 4 In **Select Study**, click **Stationary** () under **General Studies**. Click **Done** ().

A new 3D solid mechanics model using a stationary study is initialized in the COMSOL Desktop.

Saving a First Version

Now is a good time to place your new model under *version control* by saving it to your database.

- 1 From the **File** menu, select **Save To** ()
- 2 In the **Save** window, choose your newly created database, **Modeling Example**, in the list of options.
The **Save** window shows the selected database set as the target for the save. The header reads **Save new** as the model is not yet present in the database.
- 3 Write **3D stationary solid mechanics** in the empty **Title** field.
- 4 You can write an optional comment describing what you are saving in the **Comments** field. Write **Setup for a 3D solid mechanics model using a stationary study**.



- 5 Click the **Save** button ()

A first version of the model is now saved in the database.









Saving Models to Databases


Click the root node in the **Model Builder** window. The **Title** field in the **Presentation** section of the **Settings** window has been updated with the title you gave when saving.

Saving More Versions

The model uses a geometry that was previously created and stored in the COMSOL native CAD format `mphbin`.

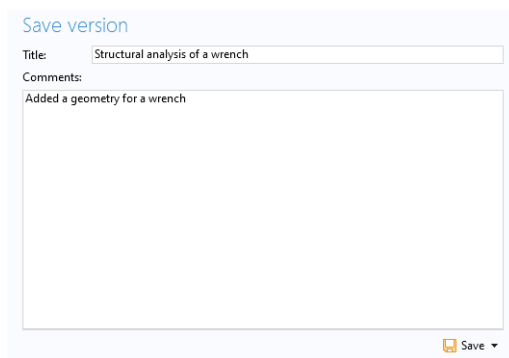
- 1 In the **Model Builder** window, right-click **Geometry 1** () and select **Import** ()
- 2 In the **Settings** window for **Import**, from the **Source list**, select **COMSOL Multiphysics file**.
- 3 Click **Browse** () and locate the file `wrench.mphbin` in the application library folder of the COMSOL installation folder. Its default location in Windows[®] is
`C:\Program Files\COMSOL\COMSOL61\Multiphysics\applications\COMSOL_Multiphysics\Structural_Mechanics\wrench.mphbin`
Double-click to add, or click **Open**.
- 4 Click **Import** () to display the geometry in the **Graphics** window.
- 5 Select **Geometry 1** () and click the **Build All** button () in the **Settings** window.

With a geometry added to the model, save a second version:

- 1 From the **File** menu, select **Save as Version** ()

The **Save** window opens with your database preselected in the list of options. The header reads **Save version** as the model already exists in the database. The same title you gave when saving the first version is suggested also for this second version.



- 2 In the **Title** field, change the title to `Structural analysis of a wrench`.
- 3 Write `Added a geometry for a wrench` in the **Comments** field.



- 4 Click the **Save** button ()



You now have two versions of the model saved in your database.

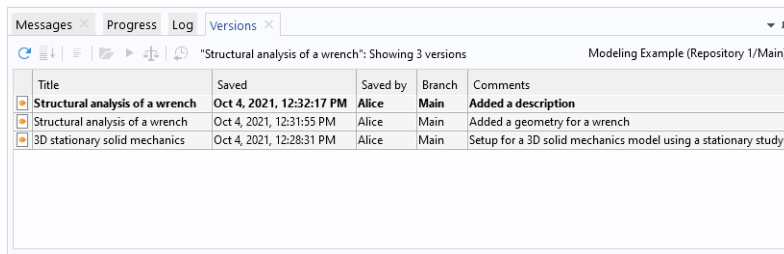
Add a description to your model and save a third version:

- 1 Click the root node in the Model Builder window.
- 2 In the **Description** field under the **Presentation** section, write **Analysis of the mechanical stress level in a wrench**.
- 3 From the **File** menu, select **Save as Version** ()
- 4 Expand the **Description** section on the right in **Save** window. The text area is prefilled with the same description you wrote in the **Presentation** section. You could have also written your new description directly here.
- 5 Write **Added a description** in the **Comments** field. Click **Save** ()

THE VERSIONS WINDOW

You have, up to this point, saved three versions of your model in the database.





From the **Windows** menu () in the **Layout** section of Model Builder's **Home** toolbar, select **Versions** () to open the **Versions** window. You will see your three versions in a table. The top table row is highlighted in bold as it is opened in the COMSOL Desktop.




The screenshot shows the 'Versions' window for a model titled 'Structural analysis of a wrench'. The window title bar includes 'Messages', 'Progress', 'Log', and 'Versions'. The main area contains a table with the following data:

Title	Saved	Saved by	Branch	Comments
Structural analysis of a wrench	Oct 4, 2021, 12:32:17 PM	Alice	Main	Added a description
Structural analysis of a wrench	Oct 4, 2021, 12:31:55 PM	Alice	Main	Added a geometry for a wrench
3D stationary solid mechanics	Oct 4, 2021, 12:28:31 PM	Alice	Main	Setup for a 3D solid mechanics model using a stationary study

You can open an older version from the **Versions** window:

- 1 Select the bottom row in the table and click the **Open** button () in the toolbar. You can also double-click the row. Select **No** if you are asked to save any unsaved changes.
The first version is opened in the COMSOL Desktop. There is no **Import** () node under the **Geometry 1** () node, and the **Description** field is empty.
- 2 Select the middle row in the table and click **Open** ()
The second version is opened in the COMSOL Desktop — the import node is now present under the geometry node, but the **Description** field is still empty.


- 3 Select the top row in the table and click **Open** ()

The third, and latest, version is opened in the COMSOL Desktop.



The Versions Window for the COMSOL Desktop Model

Working With a Draft of the Model




It may have crossed your mind that saving a new model version requires several steps — especially as compared to just pressing Ctrl+S for a model opened from the file system. You need to open the **Save** window, perhaps think of a comment describing your changes (although the comment is not required), and then click the **Save** button (). You might even realize after saving multiple versions that your modeling work has gone in the wrong direction. You would then have a *version history* cluttered with unwanted versions.



A more lightweight option when working on a model is to save a *draft* of the model. You can save versions of this draft without affecting the original model. Once you are happy with your draft, you can save it back as a new version of the original model. You may of course choose to discard your draft altogether — instead opening the original model and, perhaps, starting a new draft.

STARTING A DRAFT

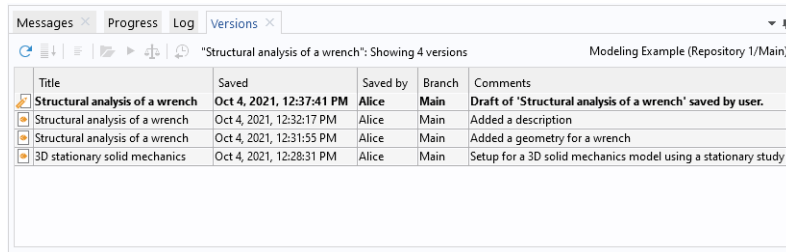
You are going to continue the modeling of the wrench using a draft. Make sure that you have opened the latest (top) version in the **Versions** window.

Add a generic steel material for the wrench and save your work as a new draft.





- 1 Right-click **Component 1 > Materials** () and select **Add Material from Library** ().
- 2 In the **Add Material** window, click to expand the **Built-In** tree node. Scroll down to find **Structural steel**, right-click, and select **Add to Component 1**.
- 3 Close the **Add Material** window.
- 4 From the **File** menu, select **Save Draft** (). You can also press the keyboard shortcut Ctrl+S.

You have created a first version of a draft of the model. You can see this *draft version* as a new row in the **Versions** window on top of the three versions of the original model. The draft version uses a separate pen icon () to distinguish it from the *regular versions* (). Note that the regular versions belong to the original model, not the

draft itself — they are included in the table to make it easier for you to track where the draft originated from.



The screenshot shows the 'Versions' window in COMSOL Desktop. The window title is 'Messages Progress Log Versions' and the subtitle is 'Structural analysis of a wrench': Showing 4 versions. The window content is a table with the following data:



Title	Saved	Saved by	Branch	Comments
 Structural analysis of a wrench	Oct 4, 2021, 12:37:41 PM	Alice	Main	Draft of 'Structural analysis of a wrench' saved by user.
 Structural analysis of a wrench	Oct 4, 2021, 12:32:17 PM	Alice	Main	Added a description
 Structural analysis of a wrench	Oct 4, 2021, 12:31:55 PM	Alice	Main	Added a geometry for a wrench
 3D stationary solid mechanics	Oct 4, 2021, 12:28:31 PM	Alice	Main	Setup for a 3D solid mechanics model using a stationary study




Saving Drafts of Models


SAVING ADDITIONAL DRAFT VERSIONS

Specify the load applied to the wrench and save your draft changes.

- 1 Select **Parameters 1** () in the **Model Builder** window.
- 2 In the **Settings** window's **Parameters** table, enter these settings:
 - In the **Name** column, enter F.
 - In the **Expression** column, enter 150[N].
 - In the **Description** column, enter Applied Force.
- 3 From the **File** menu, select **Save Draft** ()

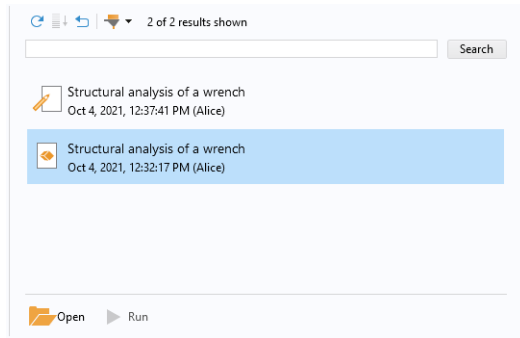
Selecting **Save Draft** a second time creates a second version of your draft — the **Versions** window now shows two draft versions and three regular versions. As for the regular versions of the original model, you can inspect an older draft version by selecting the row in the table and clicking **Open** ()

A draft is a model in its own right in the database — existing side by side with the original model. You can switch back and forth between them in the COMSOL Desktop simply by opening one or the other. To demonstrate this:

- 1 From the **File** menu, select **Open From** ()

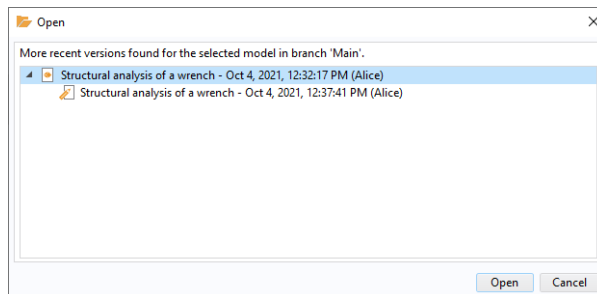
- 2 In the **Open** window, choose your database, **Modeling Example**, in the list of options.

The **Open** window shows the latest version of the draft (🔍) and the latest version of the original model (📁) in the search result.



- 3 Select the version of the original model (📁) and click the **Open** button (📁).

Model Manager detects that there is an ongoing draft of the original model with a draft version newer than the latest version of the model. A dialog box is shown in which you can choose to open that draft version instead.



- 4 Select the top node in the tree in the dialog box and click **Open** to open the original model. The latest version of the model is opened in the COMSOL Desktop — neither the **Structural steel** node nor the **Parameters** setting is present in the model tree, as expected.



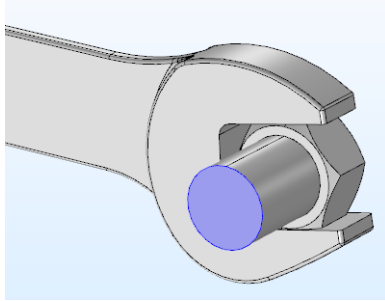
You could at this point continue working with the original model, thereby implicitly discarding your draft work. The draft itself can be manually deleted from the database at some later time.

Choosing instead to continue with your draft, open its latest version again:

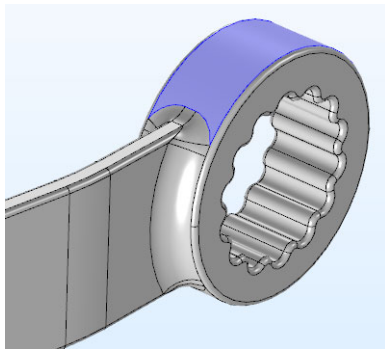
- 1 From the **File** menu, select **Open From** (📁).
- 2 In the **Open** window, choose your database in the list of options.
- 3 Select the draft version (📄) and click **Open** (📄).



Finish the component setup by defining boundary conditions and mesh settings:

- 1 Right-click **Solid Mechanics (solid)** (🔧) and select **Fixed Constraints** (🔒).
- 2 In the **Graphics** window, rotate the geometry and select the front surface of the partially modeled bolt. The **Boundary** number in the **Selection** list is **35**.

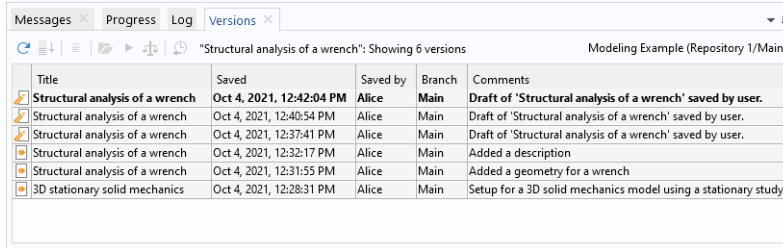








- 3 Right-click **Solid Mechanics (solid)** (🔧) once more and select **Boundary Load** (📄).
- 4 Select the top socket face (boundary 111) in the **Graphics** window.



- 5 In the **Settings** window for **Boundary Load**, under **Force**, select **Total force** as the **Load type** and enter -F in the text field for the **z** component.
- 6 Select **Mesh I** (). In the **Settings** window for **Mesh**, under **Physics-Controlled Mesh**, select **Finer** from the **Element size** list.
- 7 Click the **Build All** () button in the **Settings** window.


With the basic setup finished, select **Save Draft** () in the **File** menu to save a third draft version.



Title	Saved	Saved by	Branch	Comments
 Structural analysis of a wrench	Oct 4, 2021, 12:42:04 PM	Alice	Main	Draft of 'Structural analysis of a wrench' saved by user.
 Structural analysis of a wrench	Oct 4, 2021, 12:40:54 PM	Alice	Main	Draft of 'Structural analysis of a wrench' saved by user.
 Structural analysis of a wrench	Oct 4, 2021, 12:37:41 PM	Alice	Main	Draft of 'Structural analysis of a wrench' saved by user.
 Structural analysis of a wrench	Oct 4, 2021, 12:32:17 PM	Alice	Main	Added a description
 Structural analysis of a wrench	Oct 4, 2021, 12:31:55 PM	Alice	Main	Added a geometry for a wrench
 3D stationary solid mechanics	Oct 4, 2021, 12:28:31 PM	Alice	Main	Setup for a 3D solid mechanics model using a stationary study

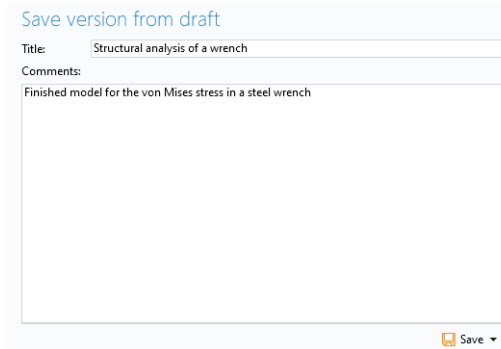
FINISHING YOUR DRAFT

With the component setup finished, it is time to save your draft work back to the original model:

- 1 From the **File** menu, select **Save as Version** ().

The **Save** window opens with your database preselected in the list. The header reads **Save version from draft** as a new version of the original model will be saved from the draft.

- 2 Write **Finished model for the von Mises stress in a steel wrench** in the **Comments** field.




Save version from draft

Title:

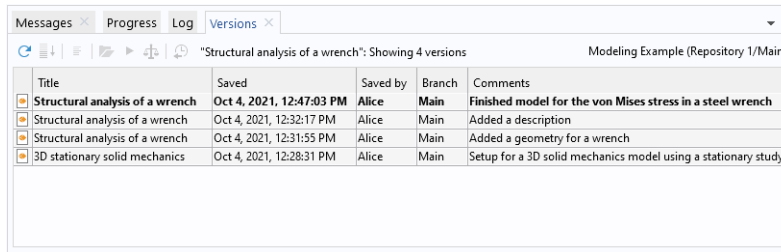
Comments:

Finished model for the von Mises stress in a steel wrench

 Save ▾

- 3 Click the **Save** button ().

Open the **Versions** window to see that all draft versions are now gone and replaced by your new, fourth, version of the original model.



Title	Saved	Saved by	Branch	Comments
Structural analysis of a wrench	Oct 4, 2021, 12:47:03 PM	Alice	Main	Finished model for the von Mises stress in a steel wrench
Structural analysis of a wrench	Oct 4, 2021, 12:32:17 PM	Alice	Main	Added a description
Structural analysis of a wrench	Oct 4, 2021, 12:31:55 PM	Alice	Main	Added a geometry for a wrench
3D stationary solid mechanics	Oct 4, 2021, 12:28:31 PM	Alice	Main	Setup for a 3D solid mechanics model using a stationary study



The draft is automatically deleted when you save it back to the original model. This deletion is not permanent though — see [Deleting Items](#) and references therein to learn how you may recover your draft versions.

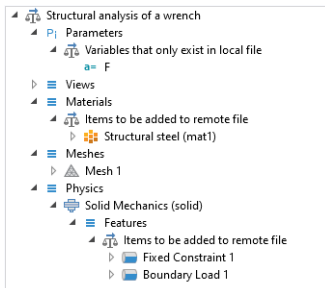
Comparing Versions

You can see all changes made to the model when you saved it from your draft.

- 1 Right-click the second row from the top in the **Versions** window and select **Compare** (🔍).

The **Comparison Result** window is opened with a comparison between the current model in the COMSOL Desktop and the selected version.

- 2 In the **Comparison Result** window, click the **Expand All** button (📄) in the toolbar.
- 3 The expanded tree shows, for example, the force parameter, the steel material, the mesh settings, and the two boundary conditions, added from your draft.

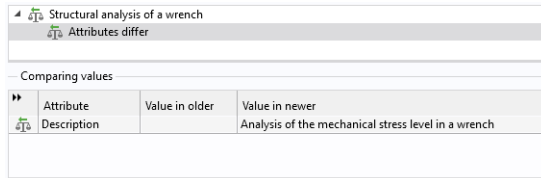


You can also compare two versions with each other:

- 1 Select the second and third rows in the **Versions** window, right-click either one, and select **Compare** (.

The **Comparison Result** window is updated with a comparison between these two versions.

- 2 Expand the tree and select the **Attributes differ** child node. In the **Comparing values** table you will find the description you added in the third model version.





Attribute	Value in older	Value in newer
Description		Analysis of the mechanical stress level in a wrench

- 3 Close the **Comparison Result** window.




Comparing Models Saved in Databases

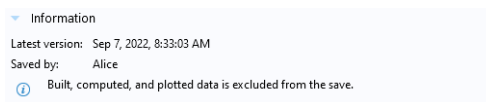
Excluding Built, Computed, and Plotted Data


Right-click **Study I** () and select **Compute** () to solve the model. When the computation finishes after a few seconds, the von Mises stress is displayed in a default Volume plot in the **Graphics** window.




Storing simulation data generated by a model can require a large amount of disk space usage. For such data that is reproducible — such as *built, computed, and plotted data* — it may be undesirable, or even impossible due to sheer size, to save it in the database.

- 1 Select the root node in the **Model Builder** window.
- 2 In the **Settings** window, under **Built, computed, and plotted data** in the **Save** section, select **Exclude** in the **In database** list. You can leave the **On file** list as is.
- 3 From the **File** menu, select **Save as Version** (.

The **Save** window opens for your database with the message **Built, computed, and plotted data is excluded from the save** shown in the **Information** section.



- 4 Write **Saved** without generated simulation data in the **Comments** field.
- 5 Click **Save** ().


The model remains in its solved state in the COMSOL Desktop after the save. Go to the **File** menu, select **Revert to Saved** (), and click **Yes** in the dialog box that appears. The latest saved version is opened. You can reproduce the, by now, lost solution by right-clicking **Study 1** () and selecting **Compute** ().




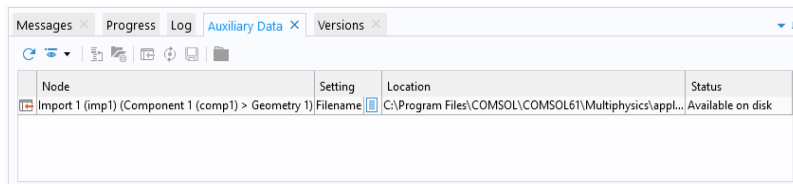
Built, Computed, and Plotted Data

Importing Auxiliary Data to the Database


You may have noticed that, while the model is version controlled in the database, the same is not true for the *CAD input file*. You can import the file to the database as follows:

- 1 From the **Windows** menu () in the **Layout** section on the **Home** toolbar, select **Auxiliary Data** to open the **Auxiliary Data** window.

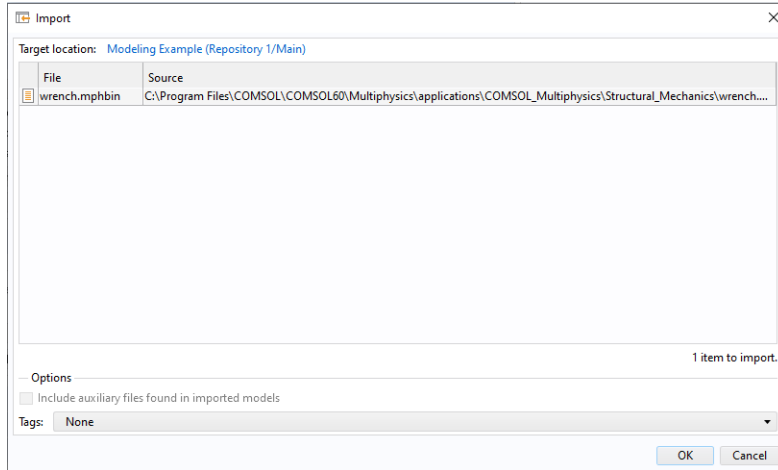
The **Auxiliary Data** window shows a table with input and output data referenced by nodes in the model tree. In this case, a single row for the CAD file used by the **Import** () node is shown.




Node	Setting	Location	Status
Import 1 (imp1) (Component 1 (comp1) > Geometry 1)	Filename	C:\Program Files\COMSOL\COMSOL61\Multiphysics\appl...	Available on disk

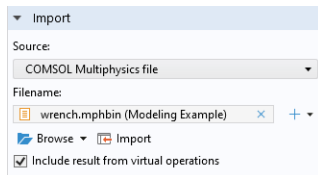
- 2 Select the table row, right-click, and select **Import to Database** ().

3 The **Import** dialog box shows the file `wrench.mphbin` in a table.




4 Click **OK** to import the file into the database.

Select the **Import** () node in the **Model Builder** window. The **Filename** field in the **Import** section in the **Settings** window now shows a reference to the file imported into the database.




Finish by saving the model to the database:

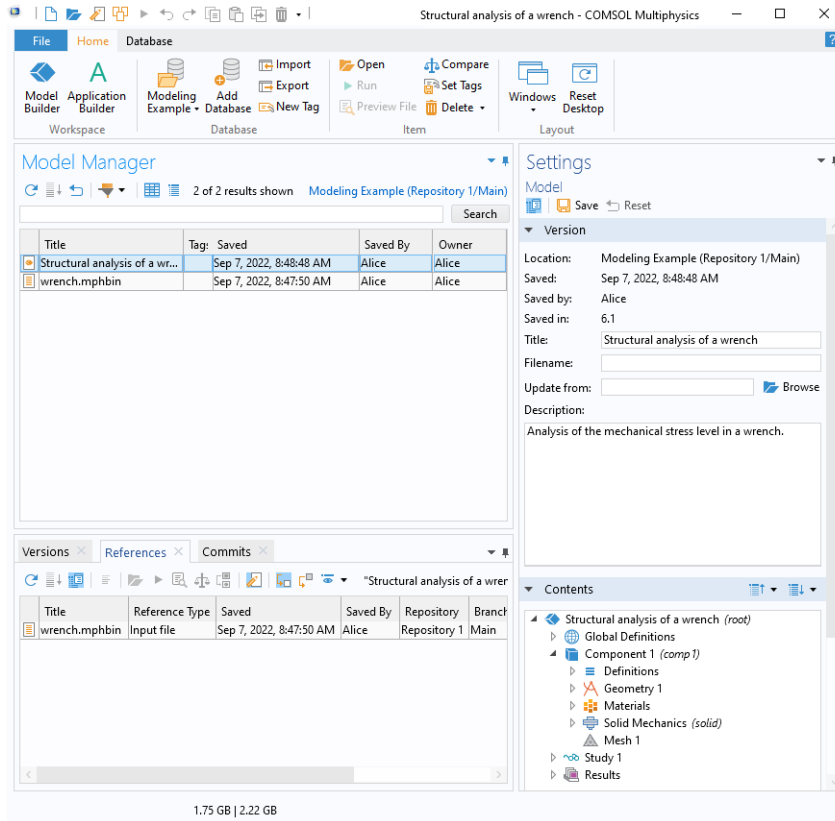
- 1 From the **File** menu, select **Save as Version** ()
- 2 Write `Referenced CAD file from database` in the **Comments** field. Click **Save** ()



The Auxiliary Data Window for Database Input and Output





The Model Manager Workspace

Click the **Model Manager** () button in the **Workspace** section of the **Home** toolbar in Model Builder to open [The Model Manager Workspace](#) — a workspace in the COMSOL Desktop dedicated to database-specific tasks. The COMSOL Desktop switches to display the toolbar for the Model Manager, as well as opens windows belonging to the workspace. You will find the latest versions of your model and CAD file in [The Model Manager Window](#).



To learn more on what you can do in the workspace, you can, for example:

- Select the model and expand the model tree in the [The Contents Section of The Settings Window](#). You will find that you can browse the content of a model without opening it.

- Search for your model by applying various [Item and Content Filters](#), for example, a [Physics](#) filter on a Solid Mechanics interface () or a [Parameter](#) filter for the applied force of 150[N].
- Right-click the model and select **References** () to see the database relationship between the model and the CAD file in [The References Window](#).
- Right-click the CAD file and select **Versions** () to see all versions of the file — currently only one — in [The Versions Window](#).
- Right-click the model and select **Commits** () to open [The Commits Window](#). Double-click the third table row from the top. You will see details on the commit in which a new version of the original model was saved from your draft, and the draft itself was deleted.


This concludes this introductory tutorial. You are encouraged to further explore the Model Manager workspace by working through the remaining example tutorials in this chapter.

Example: Browsing, Organizing, and Searching Models and Data Files

In this tutorial section, you will learn how you can browse and search for models and data files stored in a database using the Model Manager search functionality. You will also learn how you can assign tags to your models and data files to achieve better organization in the database, thereby making it easier to retrieve these items in the future. The tutorial uses a demo database containing models and data files imported from the COMSOL Application Libraries. You can download and open this local database from within the COMSOL Desktop environment.

- [Downloading the Demo Database for Model Manager](#)
- [Searching and Browsing the Demo Database](#)
- [Using a Tag Tree for Organization and Retrieval in a Database](#)
- [Creating and Assigning Tags](#)
- [Searching on Model Contents](#)
- [Using the Model Manager Search Syntax](#)

Downloading the Demo Database for Model Manager

To add the demo database for Model Manager to the COMSOL Desktop, select **Download Demo Database for Model Manager** () in the **File>Help** menu. The database is downloaded as a compressed archive from the COMSOL web site and unpacked to the default **Directory for local databases** as set in the **Preferences** dialog box — see also [New Local Database](#). Once finished, [The Model Manager Workspace](#) is automatically opened with the demo database preselected in the **Database** section on the **Home** toolbar.



You can also manually download the demo database directly from the COMSOL web site — see the instructions on <https://www.comsol.com/model/demo-database-for-model-manager-104691>. This is useful if you, for example, want to keep a pristine backup of the database for future reference once you have finished the tutorials in this chapter.

Searching and Browsing the Demo Database


The demo database contains a few thousand items that have been imported to the database from files found in the COMSOL Application Libraries. These are files you can use, for example, to learn how to build models in the COMSOL Desktop modeling environment or use as templates for your own simulation models and applications. This tutorial shows how to use these models and data files as an example of a larger database in which the Model Manager tools for browsing, searching, and organizing items become useful.



The Application Libraries Window in the *COMSOL Multiphysics* Reference Manual


The **Model Manager** window lists these models and data files in a table. If you scroll down the table, you will notice that the table only contains the first 100 items in the database sorted alphabetically on their titles. This cutoff is an optimization to not cause unnecessary data traffic between the COMSOL Desktop and the database — something that may be especially important when connected to a Model Manager server over the network.

Title	Tags	Saved	Saved By	Owner
1D Isothermal Lithium-Air Battery	Batteries, Lithium-Ion	Sep 15, 2022, 8:50:14 AM	COMSOL	COMSOL
1D Isothermal Lithium-Ion Battery	Batteries, Lithium-Ion	Sep 15, 2022, 8:50:15 AM	COMSOL	COMSOL
1D Isothermal Nickel-Cadmium Battery	Batteries, General	Sep 15, 2022, 8:49:20 AM	COMSOL	COMSOL
1D Isothermal Nickel-Metal Hydride Battery	Batteries, General	Sep 15, 2022, 8:49:22 AM	COMSOL	COMSOL
1D Isothermal Zinc-Silver Oxide Battery	Batteries, General	Sep 15, 2022, 8:49:32 AM	COMSOL	COMSOL
1D Lithium-Ion Battery Drive-Cycle Monit...	Batteries, Lithium-Ion	Sep 15, 2022, 8:50:18 AM	COMSOL	COMSOL
1D Lithium-Ion Battery for Thermal Models	Thermal Management	Sep 15, 2022, 8:51:31 AM	COMSOL	COMSOL
1D Lithium-Ion Battery Model Charge Co...	Tutorials	Sep 15, 2022, 9:26:50 AM	COMSOL	COMSOL
1D Lithium-Ion Battery Model for the Cap...	Batteries, Lithium-Ion	Sep 15, 2022, 8:50:02 AM	COMSOL	COMSOL
1D Plane Slider Bearing	Thin-Film Flow	Sep 15, 2022, 8:55:30 AM	COMSOL	COMSOL
1D Step Bearing	Thin-Film Flow	Sep 15, 2022, 8:55:32 AM	COMSOL	COMSOL
2D Lithium-Ion Battery	Batteries, Lithium-Ion	Sep 15, 2022, 8:50:35 AM	COMSOL	COMSOL
2D Non-Newtonian Slot-Die Coating	Tutorials	Sep 15, 2022, 9:41:26 AM	COMSOL	COMSOL
3D Analysis of a Bipolar Transistor	Transistors	Sep 15, 2022, 9:52:45 AM	COMSOL	COMSOL
3D Density-Gradient Simulation of a Nano...	Transistors	Sep 15, 2022, 9:53:12 AM	COMSOL	COMSOL
3D ICP Reactor, Argon Chemistry	Inductively Coupled Plasmas	Sep 15, 2022, 9:40:32 AM	COMSOL	COMSOL
3D Supersonic Flow in a Channel with a B...	High Mach Number Flow	Sep 15, 2022, 8:52:50 AM	COMSOL	COMSOL
7030CuNi_pol.csv	General Corrosion	Sep 15, 2022, 8:31:15 AM	COMSOL	COMSOL
9010CuNi_pol.csv	General Corrosion	Sep 15, 2022, 8:31:15 AM	COMSOL	COMSOL
A 3D Model of a Diesel Particulate Filter	Reactors with Porous Catalysts	Sep 15, 2022, 9:01:43 AM	COMSOL	COMSOL
absorptive_muffler_geom_sequence_para...	Automotive	Sep 15, 2022, 8:28:28 AM	COMSOL	COMSOL
absorptive_muffler_parameters.txt	Automotive	Sep 15, 2022, 8:28:28 AM	COMSOL	COMSOL
Absorptive Muffler	Automotive	Sep 15, 2022, 8:43:56 AM	COMSOL	COMSOL
Absorptive muffler geom sequence	Automotive	Sep 15, 2022, 8:43:58 AM	COMSOL	COMSOL
ac_corrosion_parameters.txt	General Corrosion	Sep 15, 2022, 8:31:16 AM	COMSOL	COMSOL
ac_corrosion_variables.txt	General Corrosion	Sep 15, 2022, 8:31:16 AM	COMSOL	COMSOL

Click the **Show More** button () in the toolbar to append the next 100 items to the bottom of the table. You can repeat this as many times as you like until all items found in the database have been appended to the table.

You use the **Model Manager** window to find models and data files in the database by writing search terms in the text field above the table.

1 Write **busbar** and click **Search**. You can also press Enter.


The table is updated to only show the items whose title, description, assigned tags, or filename fields match the word busbar. In this case, the number of search results is less than 100 so the **Show More** button () is disabled.


2 Write **bus** and click **Search**.


The search terms you write in the text field will only match complete words by default. In this case, you get zero results as there are no items with the word bus in the searched fields. You can append a *wildcard* asterisk character to match words that start with a search term. Write for example **bus*** to get back the same search result as before.

3 Write **electrical heating** and click **Search**.

When you write multiple search terms separated by spaces, the search will match on all terms using Boolean AND-logic. In this case, you find all items for which both the word electrical and the word heating is found in the searched item fields.

Select the **Thermal Analysis of a Bipolar Transistor** model () in the table. The **Settings** window updates to show various metadata fields for this model. You will find that the **Description** field for the model contains the two search words electrical and heating, although not next to each other.


You can require that all search terms must match as a sentence by enclosing the search terms with quotations. Write **"electrical heating"** in the text field and click **Search**. The **Thermal Analysis of a Bipolar Transistor** model () is now no longer present in the search result.

Click the **Reset** button () to clear the text field and restore the search to match all models and data files in the database.

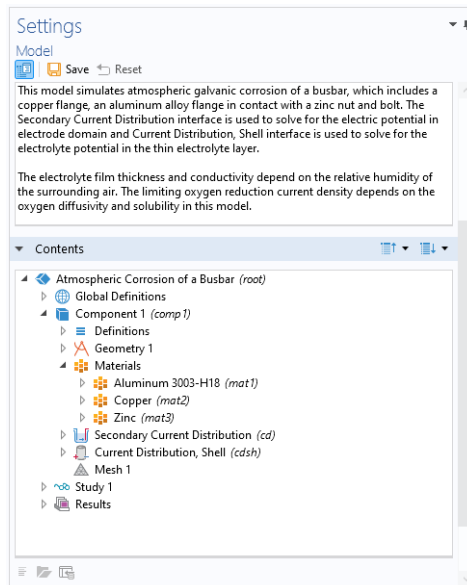




The Model Manager Window


The default behavior in the **Model Manager** window is to search the latest versions of items. This is also the version shown in the **Settings** window when you select a model or data file in the search result table. From the **Settings** window, you can, for example, see the point in time when the version was saved in the **Saved** field, the name of the user that saved the version in the **Saved by** field, and the title of the version in the **Title** field.

- 1 Write busbar in the text field again and click **Search**.
- 2 Select the **Atmospheric Corrosion of a Busbar** model () in the search result table.

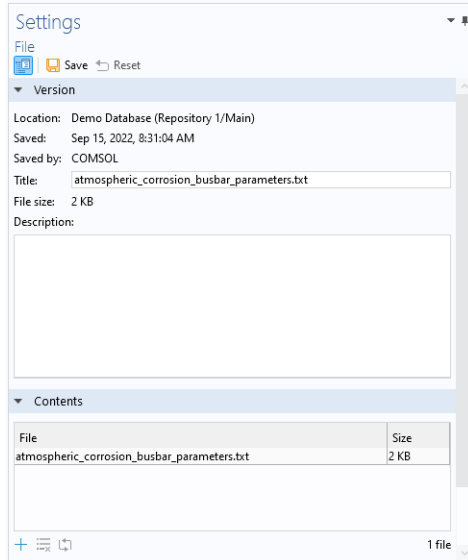
In the **Contents** section in the **Settings** window, you can explore the model tree of the model as it looked when the version was saved to the database. Expand some of the nodes in the tree to discover that this is a model with a **3D** component node containing three material nodes and two physics interface nodes.




From the **Contents** tree, you can insert components, geometry sequences, materials, and physics into the model currently opened in the COMSOL Desktop using the **Insert into Opened Model** button (). You can also select a node and click **Open Node** () to open the model with the node selected in the **Model Builder** window.

- 3 Select the `atmospheric_corrosion_busbar_parameters.txt` file () in the search result table.

This is a file containing parameters that can be used in a **Parameters** node in the **Model Builder** window. You can double-click the table row in the **Model Manager** window to preview the data file with the default application used for a text file on your computer. The **Settings** window shows, for example, the file size when the file is stored on a file system.

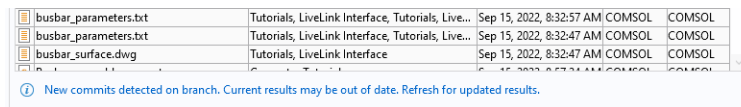


- 4 In the **Description** field in the **Settings** window, write Parameters for modeling atmospheric corrosion of a busbar. Click the **Save** button ().

The **Save File** dialog box is opened with a suggested save comment.

- 5 Change the suggested save comment to Updated the file description.
- 6 Click **OK** to save the changed description in a new version of the data file.


The **Saved** and **Saved by** field are updated in the **Settings** window to reflect the new latest version. The **Model Manager** window is, however, not automatically updated. Click the link text informing you that the current search result is out of date at the bottom of the window to refresh the search result.



7 Select **busbar_assembly.iam** () in the search result table.

This is a *fileset*, an item containing multiple data files that are version controlled as a collective whole. In this case, the data is a CAD assembly containing a main assembly file and several component files. You can see the file contents of the fileset in the **Contents** section in the **Settings** window.

8 Click the **Reset** button () to reset the search result.



The **Settings** window is where you primarily view and update items in a Model Manager database. When you click the **Save** button () in the top toolbar, a new version is saved using the current values found in the various fields in the window. You can, for example, change the title or description of an item, update a model from an MPH-file on your computer, or update the contents of a file or fileset using data files on your computer. The **Settings** window is also where you update other types of objects found in a database.



The Settings Window

APPLYING BASIC FILTERS

Writing search terms in the text field in the **Model Manager** window is often the quickest way of finding a particular model or data file given that you know, for example, its title. As a complement to this, you can apply separate search filters.



1 Click the **Add Filter** button () in the toolbar of the **Model Manager** window and select **Filename** () .

The **Filter** dialog box is opened enabling you to apply a filter on the filename used by a model or data file when exported to the file system.

2 In the **Keyword** field, write `busbar.mph`. Click **OK**.

The search result contains a single model. Select the model and verify that the **Filename** field in the **Settings** window indeed has the expected value.


A filter on a filename must include the file extension to return any matches in the search result. Alternatively, you can use a wildcard in the filter:



1 Click on the **busbar.mph** filter pill () and select **Edit** () .



2 In the **Keyword** field, change to `busbar`.

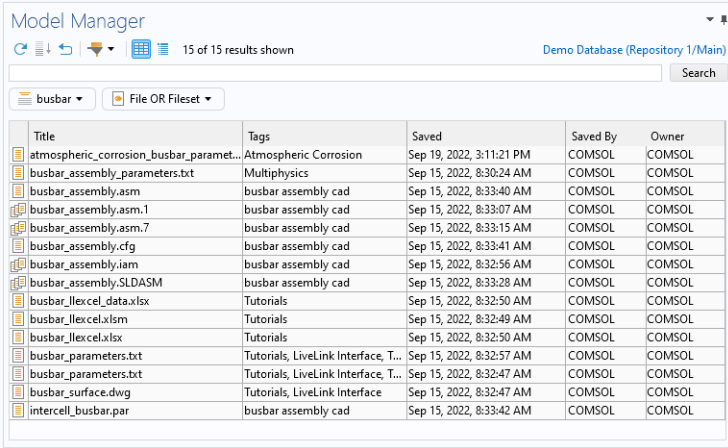
- 3 Select **Prefix Match** under **Options** to automatically append a wildcard asterisk character to the searched value. Click **OK**.

The search result contains multiple models and data files, all with a filename starting with busbar.

- 4 Click on the **busbar*** filter pill () and select **Remove** () to remove the filter.

As you may have previously noticed, writing **busbar** in the text field and clicking **Search** led to a search result containing items that did not have that particular word in their titles. Instead the word busbar was found in the description or assigned tags of the items. You can address this by applying an explicit filter on the title field. Click the **Add Filter** button () and select **Item Title** (). In the **Filter** dialog box, write **busbar** in the **Text** field. Click **OK**. The search result now only contains models and data files with the word busbar in their titles.

You can add as many filters as you like. Click the **Add Filter** button () in the toolbar again and select **Item Type** (). Select the **File** and **Fileset** check boxes. Click **OK**. The search result is further reduced to only contain data files.



The screenshot shows the Model Manager window with a search filter applied. The search bar contains 'busbar' and the 'File OR Fileset' option is selected. The results table is as follows:

Title	Tags	Saved	Saved By	Owner
atmospheric_corrosion_busbar_paramet...	Atmospheric Corrosion	Sep 19, 2022, 3:11:21 PM	COMSOL	COMSOL
busbar_assembly_parameters.txt	Multiphysics	Sep 15, 2022, 8:30:24 AM	COMSOL	COMSOL
busbar_assembly.asm	busbar assembly cad	Sep 15, 2022, 8:33:40 AM	COMSOL	COMSOL
busbar_assembly.asm.1	busbar assembly cad	Sep 15, 2022, 8:33:07 AM	COMSOL	COMSOL
busbar_assembly.asm.7	busbar assembly cad	Sep 15, 2022, 8:33:15 AM	COMSOL	COMSOL
busbar_assembly.cfg	busbar assembly cad	Sep 15, 2022, 8:33:41 AM	COMSOL	COMSOL
busbar_assembly.iam	busbar assembly cad	Sep 15, 2022, 8:32:56 AM	COMSOL	COMSOL
busbar_assembly.SLDASM	busbar assembly cad	Sep 15, 2022, 8:33:28 AM	COMSOL	COMSOL
busbar_llexcel_data.xlsx	Tutorials	Sep 15, 2022, 8:32:50 AM	COMSOL	COMSOL
busbar_llexcel.xlsxsm	Tutorials	Sep 15, 2022, 8:32:49 AM	COMSOL	COMSOL
busbar_llexcel.xlsx	Tutorials	Sep 15, 2022, 8:32:50 AM	COMSOL	COMSOL
busbar_parameters.txt	Tutorials, LiveLink Interface, T...	Sep 15, 2022, 8:32:57 AM	COMSOL	COMSOL
busbar_parameters.txt	Tutorials, LiveLink Interface, T...	Sep 15, 2022, 8:32:47 AM	COMSOL	COMSOL
busbar_surface.dwg	Tutorials, LiveLink Interface	Sep 15, 2022, 8:32:47 AM	COMSOL	COMSOL
intercell_busbar.par	busbar assembly cad	Sep 15, 2022, 8:33:42 AM	COMSOL	COMSOL

Click the **Reset** button () to reset the search result.

Using a Tag Tree for Organization and Retrieval in a Database

There are various ways a user of the COMSOL Multiphysics software may organize their simulation work when relying solely on the file system for storage. Perhaps they have set up a folder structure containing their MPH-files and associated data files. The files are maybe placed in various subfolders corresponding to different projects (in a

generic sense). Perhaps there is a shared folder with data files used as input in several models and in different projects. There could also be a template folder with a few models containing some common setups and to be used as starting points for new projects. Version control of files could be achieved via a strict naming convention: say `model_v1.mph`, `model_v2.mph`, `model_v3.mph`, etc. and `data_file_v1.txt`, `data_file_v2.txt`, `data_file_v3.txt`, etc. To keep track of the ongoing work, each series of version-controlled MPH-files could perhaps have an associated work log file that describes what was changed between each version. Each data file might have an associated *used-by* text file, containing a list of MPH-files known to use this particular data file as input (or there could be a single spreadsheet document containing a table with all such relations).


The Model Manager tools in the COMSOL Desktop modeling environment helps you with these organizational and version-control aspects of your simulation work by, for example, keeping a version history of items, automatically keeping track of references between items, and enabling you to save comments, descriptions, and other metadata with your items. One such organizational metadata is the concept of assigning *tags* to items, which is the Model Manager analogue to placing such items in folders on your computer. Much like folders on the file system, tags can be assigned to other tags, resulting in a *tag tree* structure of tags, models, and data files.


The demo database for Model Manager already contains such a tag tree. These tags were automatically created from the corresponding folders that contained the models and data files in the COMSOL Application Libraries when these files were imported into the database.

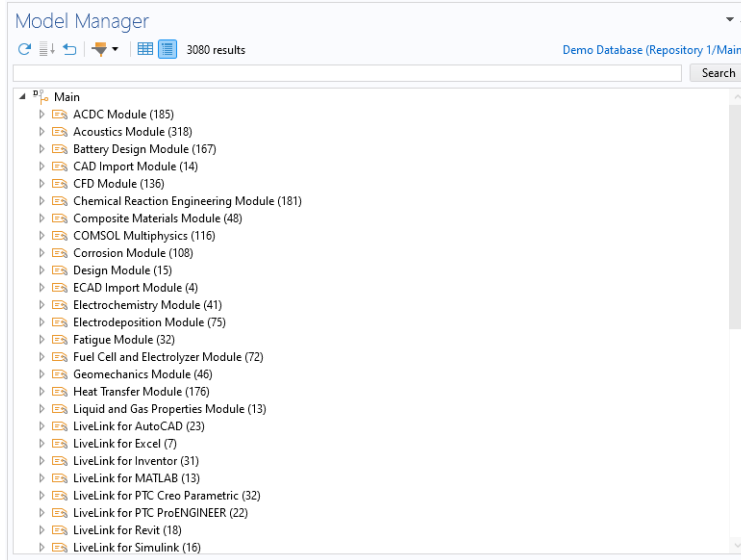


The folder structure in the COMSOL Application Libraries, and thus the resulting tag tree in the demo database for Model Manager, is heavily influenced by the COMSOL product suite itself — with emphasis on the available add-ons and interfacing products, and their associated model tutorials. The tag structure you create in your own Model Manager databases are likely to show little resemblance to this tree.

You can view the tag tree from the **Model Manager** window by switching from the [Table View](#) to the [Tree View](#). While the former view is useful when you quickly want to find models and data files given a specific set of search and filter criteria, the latter is more suited to browsing the items in the database without having a particular model or data file in mind.

1 In the **Model Manager** window, click the **Tree** button () in the toolbar.

The **Model Manager** window switches to show a tree of tags (). There are about 50 tags at the top level, with several hundred more tags found under these top tags.




2 Expand the tags **ACDC Module>Tutorials, Cables**. You will find 38 items under the last tag tree node. In the Model Manager database, these items are all assigned a tag with title **Tutorials, Cables**. The **Tutorials, Cables** tag is itself assigned the **ACDC Module** tag. The number of items found under each tag tree node in the search result is written within parentheses next to the tag's title.

3 Write `tutorials, cables` in the text field and click **Search**. The tree only shows the 38 items, the **Tutorials, Cables** tag, and the **ACDC Module** tag.



Punctuation characters, such as commas, are ignored when searching in a Model Manager database. Write `tutorials cables` to get the same search result.

4 Write `acdc module` and click **Search**. The search result contains all 185 items found under the **ACDC Module** tag. You may be surprised, however, to discover a few other tags on the top level in the tag tree.

5 Expand **ACDC Module>Electromagnetics and Optimization**. One of the *tagged* items is the **Topology Optimization of a Magnetic Circuit** model (). Also expand **Acoustics**


Module>Optimization and notice that the same model is in fact also found under these latter tags.

You can assign multiple tags to the same item in a Model Manager database. In this case, Model Manager discovered during the import that the model **Topology Optimization of a Magnetic Circuit** was stored as identical MPH-files in multiple folders in the COMSOL Application Libraries. Rather than importing duplicate models, a single model was imported and assigned multiple tags.

- 6 Write `topology optimization magnetic circuit` and click **Search**. You will discover that the model, together with four other items, was assigned three different tags during the import.
- 7 Click the **Reset** button () to clear the current search terms and restore the tag tree. Click the **Table** button () to switch back to the [Table View](#).

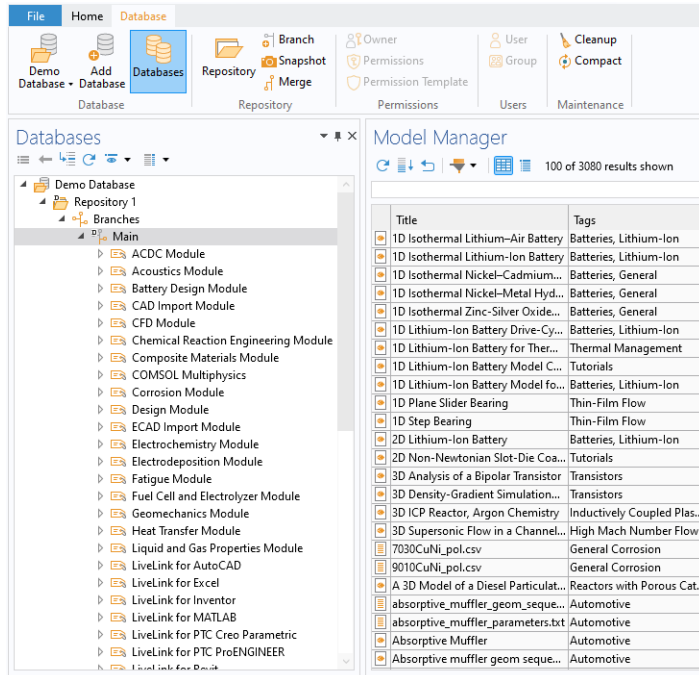
BROWSING THE TAG TREES OF MULTIPLE DATABASES

The **Model Manager** window enables you to search and browse a specific branch in a specific repository in a Model Manager database. If you would prefer to browse multiple branches in different repositories, and perhaps in different databases, at once, you can use the **Databases** window.

- 1 On the **Database** toolbar, in the **Database** section, click the **Databases** button (). The **Databases** window opens next to the **Model Manager** window. The window shows all databases that you have added to the COMSOL Desktop as a tree.

- Expand the **Main** (📁) branch node for the demo database.

The same tag tree available in the **Model Manager** window is shown as a subtree to the branch node. Unlike the **Model Manager** window, however, there is no search functionality available in the window.



- If you have added any other databases than the demo database, you are encouraged to explore their database subtrees as well. This includes, for example, the database created in the tutorial [Example: Modeling Using Version Control](#).
- Click the **Databases** button (📁) again to close the **Databases** window.


The **Databases** window is also where you perform more advanced database operations including, for example, creating and merging branches, and managing users and groups in a Model Manager server database.






The Databases Window

Creating and Assigning Tags



The tags in the demo database were all created from folders on the file system. You can also create new tags directly in the database and assign them to your models and data files.

- 1 Write **tutorials** in the text field in the **Model Manager** window and click **Search**.
The search result contains about 500 items, all somehow related to various modeling tutorials for the COMSOL Desktop modeling environment
- 2 Click the **Tree** button () in the toolbar to show the tag tree.
- 3 Browse the tree by expanding some of the tag tree nodes. You will quickly discover that the various tutorial models are spread out over a multitude of tags with various tutorial-related titles.

You are going to create a new tag in the database that you will assign to all models used in tutorials. Having such a tag will make it easier to quickly find all tutorials in the future.

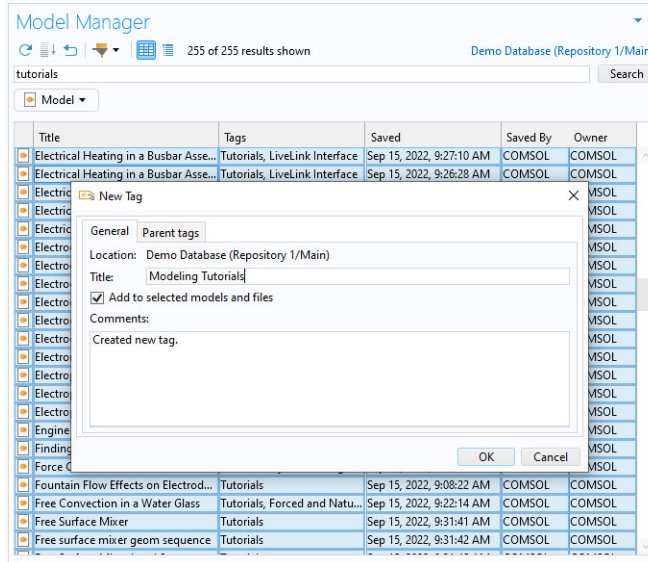
- 1 Click the **Table** button () to switch back to the **Table View**.
- 2 Click the **Add Filter** button () in the toolbar and select **Item Type** ().
The **Filter** dialog box is opened enabling you to apply a separate filter on the item types to include in the search result.
- 3 In the **Filter** dialog box, select the **Model** check box. Click **OK**.

The search result is reduced to about 200 models.

- 1 Repeatedly click the **Show More** button () in the toolbar until all models have been appended to the table.
- 2 Select one of the table rows and press Ctrl+A to subsequently select all table rows.
- 3 On the **Home** toolbar, in the **Database** section, click the **New Tag** button ().
The **New Tag** dialog box is opened with the check box **Add to selected models and files** already selected. The dialog box enables you to create a new tag and simultaneously assign the tag to the current selection of models and data files in the **Model Manager** window.

- Write **Modeling Tutorials** in the **Title** field. Click **OK**.

The save takes a few seconds as Model Manager makes the new tag assignment available for searching and filtering.



- At the bottom of the **Model Manager** window, click the link text informing you that the current search result is out of date.

The current search is reloaded. Notice that the **Tags** column contains your new tag assigned to all items in the search result.

- Click the **Reset** button (↶) to clear the current search term and filter. Click the **Tree** button (☰) in the toolbar to switch to the **Tree View**. Expand the **Modeling Tutorials** tag tree node to find all tutorial models.

The **Model Manager** window shows the first 100 items under a tag tree node. Expand the **Show More** tree node (☰↓) to reveal the next 100 items.

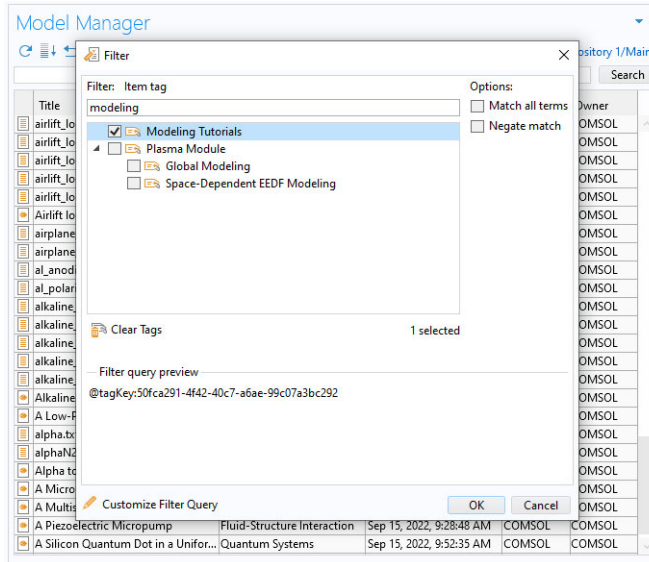
- Click the **Table** button (☰) to return to the **Table View**.



Tagging Models and Files

With the new tag for modeling tutorials in the database, it is a simple matter of quickly finding such models by applying a tag filter.

- 1 Click the **Add Filter** button (🗑️) in the toolbar and select **Item Tag** (🏷️).
The **Filter** dialog box is opened enabling you to apply a separate filter on the assigned tags of items to include in the search result.
- 2 Write `modeling` in the text field above the tag tree to filter the available tags on their titles.




- 3 Select the **Modeling Tutorials** check box and click **OK**.
The search result only includes the models that are assigned the **Modeling Tutorial** tag (🏷️). From here one can continue to drill down in the search result by writing search terms or applying additional filters.
- 4 Click on the **Modeling Tutorials** filter pill (🏷️) and select **Remove** (🗑️) to remove the filter.


In this case, you could have also found all tutorial models by simply writing `modeling tutorials` in the text field and clicking **Search**. But for tags whose titles are likely to also appear in the titles or descriptions of model and data files, using an explicit **Item Tag** filter (🏷️) is a better strategy.

ASSIGNING EXISTING TAGS TO ITEMS


The tags in the demo database are all of a purely descriptive nature related to what the models and data files contain. You can also use tags for simple project and workflow management.

- 1 Click the **New Tag** button () in the **Database** section on the **Home** toolbar.
- 2 In the **Title** field, write **Projects**.
- 3 Make sure that the check box **Add to selected models and files** is cleared. Click **OK**.


A new tag that can be used to group models and data files involved in various simulation projects is created in the database.

- 1 Click the **New Tag** button () again.
- 2 In the **Title** field, write **In Progress**.
- 3 Click the **Parent tags** tab.

A tag tree is shown enabling you place the new tag under an existing tag.


- 4 Find and select the **Projects** tag () — for example by typing **projects** in the text field above the tree.
- 5 Click **OK** to create the tag.

For the purpose of this part of the tutorial, assign the **In Progress** tag to a few select models:

- 1 Write **electrical heating** in the **Model Manager** window and click **Search**.
- 2 Select one of the table rows and press **Ctrl+A** to subsequently select all table rows.
- 3 On the **Home** toolbar, in the **Database** section, click the **Set Tags** button ()

The **Set Tags** dialog box is opened enabling you to set the assigned tags of one or more items. You will notice that some of the check boxes in the tag tree have an *indeterminate* selection. These are tags that are assigned to some, but not all, of the items. Leaving these indeterminate check boxes as-is means that these assignments will not be modified when you click **OK**.

- 4 Find and select the **In Progress** tag — for example by typing **progress** in the text field above the tree.
- 5 Click **OK** to assign the **In Progress** tag to the items.

Much like the modeling tutorials, you can now find these models and data files by searching on the tag title or by applying the tag as a filter. Perhaps you use the **In Progress** tag () , for example, as a way of signaling to your coworkers that you are



currently working on updates to these models, which could be a more lightweight alternative to setting restrictive permissions for the models.




Adding and Removing Tag Assignments

ASSIGNING TAGS WHEN SAVING MODELS


So far you have worked exclusively with tags from the Model Manager workspace. You can also assign tags directly when saving a model version from the **Save** window. To this end, create a new blank model in the COMSOL Desktop to act as a placeholder for a model tutorial.

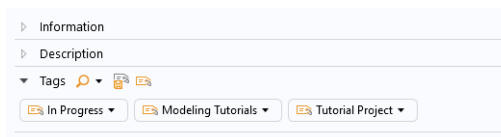
- 1 From the **File** menu, select **New**. Click **Blank Model** ()
- 2 From the **File** menu, select **Save To** ()
- 3 In the **Save** window, choose the demo database in the list of options.
- 4 In the **Title** field, write My Tutorial.
- 5 In the **Comments** field, write A new model tutorial.


The **Tags** section in the **Save** window shows the tags assigned to the model, which in the present case are none.

- 1 Click the **Add Tag** button ()
- 2 Write **Modeling Tutorials** and press Enter.
The **Modeling Tutorials** tag is added as a *tag pill* in the **Tags** section.
- 3 Repeat this also for the **In Progress** tag.

You can also create a new tag in the database from the **Save** window.


- 1 Click the **New Tag** button () to open the **New Tag** dialog box.
- 2 Write **Tutorial Project** in the **Title** field.
- 3 On the **Parent tags** tab, select the **Projects** tag in the tag tree.
- 4 Click **OK**.







With the three tags added to the **Tags** section, you are ready to save the new model in the database. Click **Save** ()

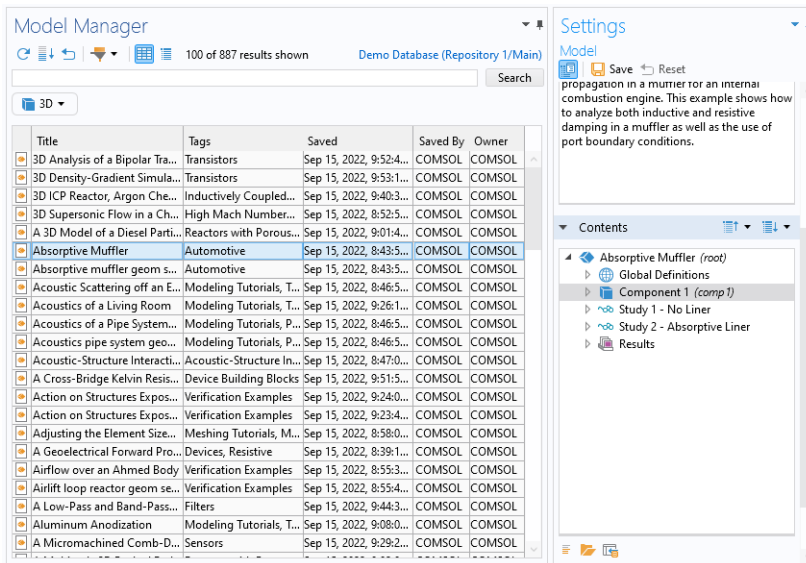
Return to the Model Manager workspace. Find and select the new model in the **Model Manager** window — for example by first searching on `my tutorial`. The assigned tags are all shown as tag pills in the **Tags** section in the **Settings** window. You can also verify that you can find the model under these three tags in the **Tree View** of the **Model Manager** window.

Searching on Model Contents

Applying an **Item Tag** filter () in the **Model Manager** window is a quick and easy way of finding models and data files in the database. This requires, however, that you have already assigned tags to your items based on some preplanned organizational structure. As you have already seen, you can also apply other types of search filters — including, for example, when an item version was last saved or the user that saved that version. You can even search on the node properties, parameters, settings, and other contents found “deep” inside models.

- 1 Click the **Reset** button () in the **Model Manager** window to reset the search result. Click the **Table** button () to see the **Table View** if not already shown.
- 2 Click the **Add Filter** button () in the toolbar and select **Space Dimension** ()
The **Filter** dialog box is opened enabling you to apply a filter on the space dimension of components found inside models.
- 3 Select the **3D** check box in the list of space dimensions.
- 4 Click **OK** to apply the filter.

The search result includes with about 900 models. Select any one of them and verify that the model indeed has a 3D component in the **Contents** section in the **Settings** window.



You can extend our filter to also match models with 2D components.

- 1 Click the **3D** filter pill in the Model Manager window and select **Edit** (🔧).
- 2 In the **Filter** dialog box, select the **2D** check box.
- 3 Click **OK** to apply the updated filter.

The search result count increases with a few hundred models having a 2D component.

The default behavior in the **Filter** dialog box is to combine filter options with Boolean **OR**-logic. You can update our filter to use Boolean **AND**-logic instead.

- 1 Click the space dimension filter and select **Edit** (🔧) again.
- 2 In the **Options** list in the **Filter** dialog box, select the **Match all terms** check box.
- 3 Click **OK**.

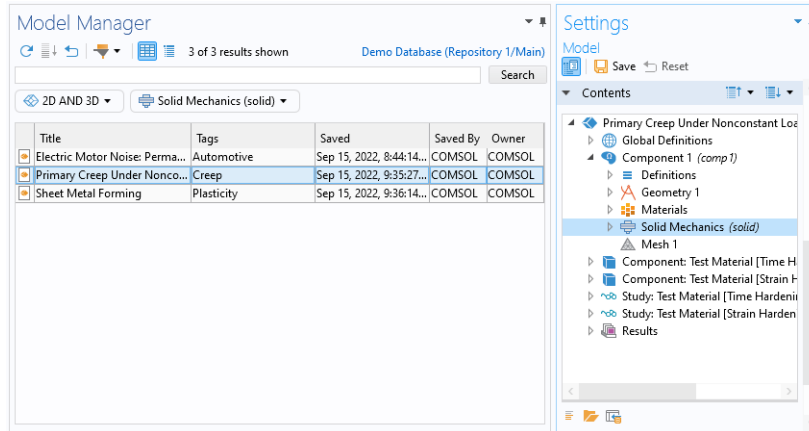
Select any model in the search result and verify that the model indeed has both a 2D component and a 3D component in the **Contents** section in the **Settings** window. You can combine the space dimension filter with a separate filter on a physics interface.

- 1 Click the **Add Filter** button (➕) and select **Physics** (⚙️).
- 2 In the text field above the tree, write `solid mechanics`. Click **Search**.

3 Select the **Structural Mechanics>Solid Mechanics (solid)** check box.

4 Click **OK**.






The search result only contains three models. All of them use the Solid Mechanics interface in at least one component.




Item and Content Filters




SEARCHING NODE PROPERTIES

You can find models based on the properties of specific nodes in the model tree.


- 1 Click the **Reset** button () to remove all filters.
- 2 Write **wrench** and click **Search**.
- 3 Double-click the **Stresses and Strains in a Wrench** model () in the search result.
The model version is loaded from the database and opened in the COMSOL Desktop. This is the same model you built in the tutorial [Example: Modeling Using Version Control](#).
- 4 Right-click **Component 1** () and select **Properties** () to open the **Properties** window for the node.
- 5 In the **Comments** field in the **Properties** window, write **A 3D component** for the structural integrity of a wrench.
- 6 From the **File** menu, select **Save To** ().

- 7 In the **Comments** field in the **Save** window, write Added a node comment for the component for easier future retrieval.
- 8 Click **Save** () to save a new version of the model.

Return to the Model Manager workspace. You can now find this model by searching on the node comment you wrote in the **Properties** window.


- 1 Click the **Reset** button () to remove the search term.
- 2 Click the **Add Filter** button () and select **Node Properties>Comment** ()
- 3 In the **Filter** dialog box, write component structural integrity wrench in the **Text** field.
- 4 Select the **Match all terms** check box under **Options**. This step is done to require that all four words appear in the matched node comment.
- 5 Click **OK**.

The wrench model is indeed the sole search result.

Double-click the component node in the **Contents** section of the **Settings** window for the **Stresses and Strains in a Wrench** model () . A **Details** dialog box is shown containing node properties and settings for the component. You can find your comment in the **Comment** row of the **Node** table. Click **OK** to close the dialog box.






All node and setting values shown in the **Details** dialog box for a node in the **Contents** section can be applied as a filter in the Model Manager search functionality. See [Content Filters](#) for further details.

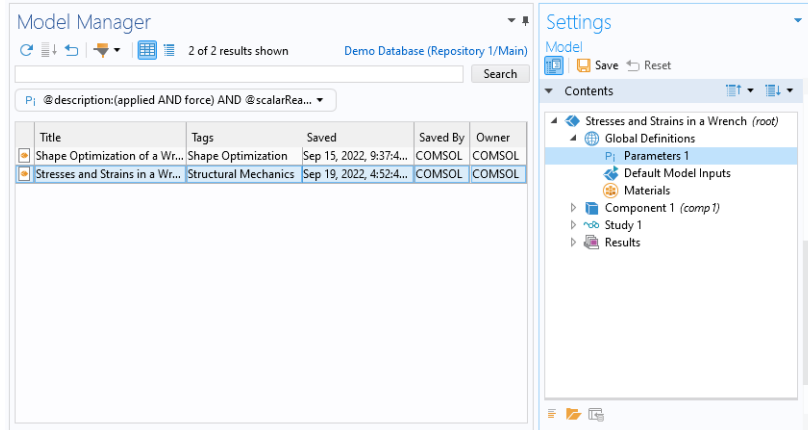
Click the **Reset** button () to remove the comment filter.

SEARCHING PARAMETER SETTINGS

The model for the structural integrity of a wrench uses an **Applied force** parameter for the total load on the wrench, in this case set to 150N. You can search for other models using the same parameter and force range.




- 1 Click the **Add Filter** button () and select **Parameter** (P_i).
- 2 In the **Description** field, write applied force.
- 3 Select the **Match all terms** check box under **Options**.
- 4 Write 100 in the **From** field and 200 in the **To** field.
- 5 Click **OK**.

There are two models in the search result. Select the **Shape Optimization of a Wrench** model () and double-click a parameters node under **Global Definitions** () in the **Contents** section. You will find that there is indeed a row for the **Applied force** with a force value within the searched range in the **Settings** table.



Using the Model Manager Search Syntax

All filters that you can apply via the **Filter** dialog box have an equivalent search syntax that you can write directly in the text field in the **Model Manager** window. This syntax is displayed under **Filter query preview** in the **Filter** dialog box.

- 1 Click the **Reset** button () to reset the search result.
- 2 Click the **Add Filter** button () and select **Item Type** ().
- 3 In the **Filter** dialog box, select the **Model** check box.

The expression under **Filter query preview** reads `@itemType:model`. The part before the colon identifies the item field to filter on; the part after the colon is the value to filter on.

- 4 Select the **Application** check box.
The expression now reads `@itemType:(application OR model)`. The expression will match any item that is either a model *or* an application.
- 5 Select the **Negate match** check box under **Options**.
The expression reads `NOT @itemType:(application OR model)`. The expression will match any item that is *neither* a model nor an application.
- 6 Click **Cancel** to close the dialog box without applying the filter.

You can try out the Model Manager search syntax in the text field.

1 Write `@itemType:fileset` and click **Search**.

The search result only contains filesets. The part before the colon is case-insensitive — writing `@itemtype:fileset` yields the same result.

2 Write `@itemType:fileset AND @filetype:asm` and click **Search**.


The search result contains all filesets containing a data file with a CAD assembly file extension. Spaces are automatically interpreted as Boolean AND so you can leave out the AND symbol in this case.

3 Write `@itemtype:fileset @filetype:asm @title:busbar` and click **Search**.

The search result only contains CAD assemblies with busbar in their title.

The search syntax can also be used for searching model contents. You can, for example, find the wrench model with the node comment. Write

```
@node{@comment:(component structural integrity wrench)}
```

and click **Search**. The **Stresses and Strains in a Wrench** model () is again the sole match in the search result.

You may have noticed that these steps did not specify that the comment should appear in a 3D component node. Write

```
@component{@spacedimension:3 @comment:(component structural integrity wrench)}
```

and click **Search**. The result is the same. Change the space dimension value to 2 and verify that you no longer get any matches.

A full treatise of the capabilities of the Model Manager search syntax is well beyond the scope of this tutorial. You can, for example, write nested search expressions that specify that only models with a specific component space dimension, such that the component itself contains a particular material and physics interface, should match.



Example: Using Advanced Version Control Tools in the Model Manager


The Model Manager introduces many concepts and terms related to version control that are new to the COMSOL Desktop modeling environment. This includes, for example, versions, drafts, commits, branches, and snapshots. In this section, you will become more acquainted with these concepts and terms by working through an example tutorial. You are strongly recommended to first go through the tutorial [Example: Modeling Using Version Control](#) if you have not already done so.

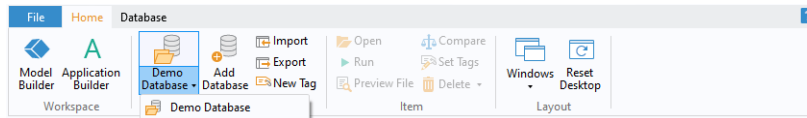
The tutorial makes use of the demo database for Model Manager provided via the COMSOL web site. If you have not already added this database to the COMSOL Desktop, follow the instructions in [Downloading the Demo Database for Model Manager](#).

- [Using a Draft to Update a Single Model](#)
- [Working with Commits](#)
- [Using a Branch to Update Many Models](#)

Using a Draft to Update a Single Model


In the tutorial [Example: Modeling Using Version Control](#), you built a model for the structural integrity of a wrench. That model can also be found in the demo database for Model Manager.

- 1 Click the **Model Manager** () button in the **Workspace** section of the **Home** toolbar in Model Builder to open the Model Manager workspace.
- 2 In the **Database** section of the **Home** toolbar, select the demo database for Model Manager via the database selector expand button.




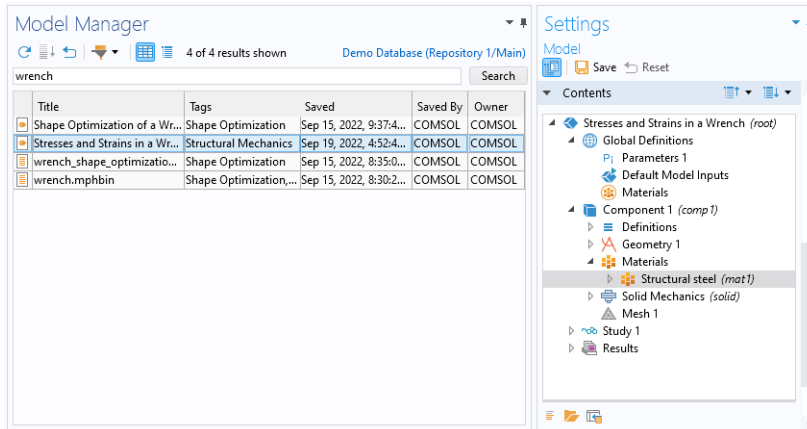
The **Model Manager** window is updated with a search result for the demo database.

- 3 Write **wrench** in the text field and click **Search**.

- 4 Select the **Stresses and Strains in a Wrench** model () in the table.


The **Settings** window is updated to show the latest version of the model. If you have already worked through the tutorial [Example: Browsing, Organizing, and Searching Models and Data Files](#), this is the version in which you added a node comment to the component node. Otherwise, it is the first version that was imported from the COMSOL Application Libraries.



- 5 Expand **Component 1 > Materials** () in the **Contents** section of the **Settings** window to find that the latest version of the model uses a Structural steel material.





You will make a small update to this model by changing to a low alloy steel material. In the process, you will encounter some new version control tools that go beyond what you learned in the tutorial [Example: Browsing, Organizing, and Searching Models and Data Files](#).

- 1 Select **Structural steel** () in the **Contents** section and click the **Open Node** button () in the toolbar below the tree.

The model version is loaded from the database and opened in the Model Builder workspace with the **Component 1 > Materials > Structural steel** node () selected.

- 2 Right-click **Component 1 > Materials > Structural steel** () and select **Delete** (). Click **Yes** in the **Confirm Delete** dialog box.

The material node is removed from the model tree.

- 3 Right-click **Component 1 > Materials** () and select **Add Material from Library** ().
- 4 In the **Add Material** window, click to expand the **Built-In** tree node. Scroll down to find **Steel AISI 4340**, right-click, and select **Add to Component 1**.

- 5 From the **File** menu, select **Save To** (📁).
- 6 In the **Comments** field, write Changed the material to a low alloy steel.
- 7 Click **Save** (💾).

A new version of the model has been saved to the database. In the tutorial [Example: Modeling Using Version Control](#), you saw that you could view the current version history of the model opened in the COMSOL Desktop from the **Versions** window in the Model Builder workspace. You can also view this version history from the Model Manager workspace.

- 1 Open the Model Manager workspace.
- 2 Right-click **Stresses and Strains in a Wrench** in the **Model Manager** window and select **Versions** (📅).

The **Versions** window is opened with the first table row highlighted in bold indicating that the version is currently opened in the COMSOL Desktop. There are three versions in total (or two if you have not gone through the tutorial [Example: Browsing, Organizing, and Searching Models and Data Files](#)).

The screenshot shows two windows from the COMSOL Desktop interface. The top window is the **Model Manager**, displaying a table of models. The second row, **Stresses and Strains in a Wrench**, is selected and highlighted in blue. Below it is a **Versions** window for the selected model, showing a table of three versions. The first row in the Versions table is bolded, indicating it is the current version.

Title	Tags	Saved	Saved By	Owner
Shape Optimization of a Wrench	Shape Optimization	Sep 15, 2022, 9:37:44 AM	COMSOL	COMSOL
Stresses and Strains in a Wrench	Structural Mechanics	Sep 19, 2022, 4:52:49 PM	COMSOL	COMSOL
wrench_shape_optimization_parameters.txt	Shape Optimization	Sep 15, 2022, 8:35:03 AM	COMSOL	COMSOL
wrench.mphbin	Shape Optimization, Structural M...	Sep 15, 2022, 8:30:27 AM	COMSOL	COMSOL


Title	Saved	Saved By	Branch	Comments
Stresses and Strains in a Wrench	Sep 20, 2022, 8:40:57 AM	COMSOL	Main	Changed the material to a low alloy steel.
Stresses and Strains in a Wrench	Sep 19, 2022, 4:52:49 PM	COMSOL	Main	Added a node comment for the component for easier future retrieval.
Stresses and Strains in a Wrench	Sep 15, 2022, 8:59:19 AM	COMSOL	Main	Imported from 'wrench.mph'.




The Versions Window

Imagine now that you regret saving a new version of the model with the changed material — you would rather have kept the Structural steel material. One solution is that you open the previous version in the COMSOL Desktop and immediately save

that as a new latest version. Yet a simpler solution is to *restore* the previous version directly in the database.


- 1 Right-click the second table row from the top in the **Versions** window and select **Restore Version** ().
- 2 In the **Restore Version** dialog box, click **OK**.
The selected version is automatically restored as a new latest version in the database.
- 3 Click **Yes** when asked if you also want to open the restored version.
The selected version is opened in the Model Builder workspace.




An added benefit to using the **Restore Version** () functionality as opposed to manually opening and saving the model in the COMSOL Desktop is that you avoid any model migrations in case the model was originally saved in an older version of the COMSOL Multiphysics software.







Restore Version

You can verify that the opened model indeed uses the old Structural steel material by expanding **Component 1 > Materials** ()

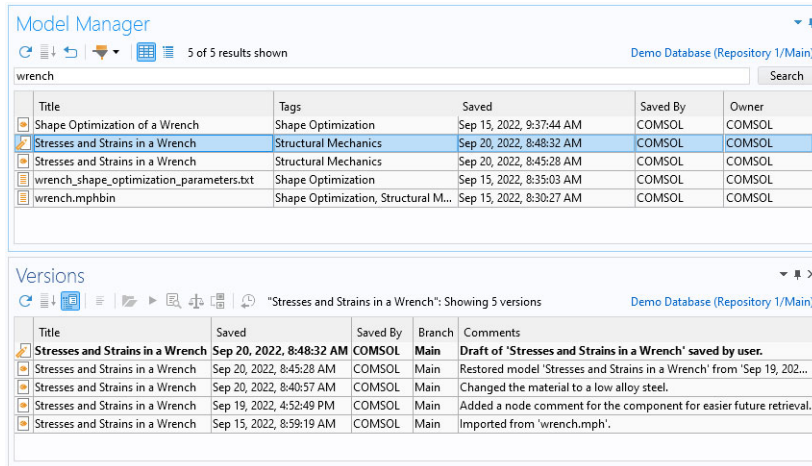
You already learned in the tutorial [Example: Modeling Using Version Control](#) that a better strategy to making changes to a model is by first creating a draft of the model. A draft is tailor-made for the use case of performing updates to a model without deciding upfront whether the changes are worth keeping.

- 1 Select **Component 1 > Materials > Structural steel** () and press Del. Click **Yes** to delete the Structural steel material.
- 2 In the **Add Material** window, right-click **Steel AISI 4340** and select **Add to Component 1** to add the low alloy steel material once more.
- 3 Press Ctrl+S to save a draft of the model.

Open the Model Manager workspace. Click the **Refresh** button () in the **Model Manager** window to refresh the table. You will find both the **Stresses and Strains in a Wrench** regular model () and a new **Stresses and Strains in a Wrench** draft model () in the search result.

- 1 Select the **Stresses and Strains in a Wrench** draft model () in the **Model Manager** window.

The **Versions** window shows the single version of the draft model in the top table row. For convenience, the table also contains the four versions of the regular model that the draft originated from.



Model Manager

5 of 5 results shown Demo Database (Repository 1/Main)

wrench Search

Title	Tags	Saved	Saved By	Owner
Shape Optimization of a Wrench	Shape Optimization	Sep 15, 2022, 9:37:44 AM	COMSOL	COMSOL
Stresses and Strains in a Wrench	Structural Mechanics	Sep 20, 2022, 8:48:32 AM	COMSOL	COMSOL
Stresses and Strains in a Wrench	Structural Mechanics	Sep 20, 2022, 8:45:28 AM	COMSOL	COMSOL
wrench_shape_optimization_parameters.txt	Shape Optimization	Sep 15, 2022, 8:35:03 AM	COMSOL	COMSOL
wrench.mphbin	Shape Optimization, Structural M...	Sep 15, 2022, 8:30:27 AM	COMSOL	COMSOL

Versions



"Stresses and Strains in a Wrench": Showing 5 versions Demo Database (Repository 1/Main)



Title	Saved	Saved By	Branch	Comments
Stresses and Strains in a Wrench	Sep 20, 2022, 8:48:32 AM	COMSOL	Main	Draft of 'Stresses and Strains in a Wrench' saved by user.
Stresses and Strains in a Wrench	Sep 20, 2022, 8:45:28 AM	COMSOL	Main	Restored model 'Stresses and Strains in a Wrench' from 'Sep 19, 2022...
Stresses and Strains in a Wrench	Sep 20, 2022, 8:40:57 AM	COMSOL	Main	Changed the material to a low alloy steel.
Stresses and Strains in a Wrench	Sep 19, 2022, 4:52:49 PM	COMSOL	Main	Added a node comment for the component for easier future retrieval.
Stresses and Strains in a Wrench	Sep 15, 2022, 8:59:19 AM	COMSOL	Main	Imported from 'wrench.mph'.

- 2 Select the **Stresses and Strains in a Wrench** regular model () in the **Model Manager** window.

The **Versions** window shows the four versions of the regular model.

You can finish the draft work by saving the draft back as a new version of the regular model.

- 1 From the **File** menu, select **Save To** ().
- 2 In the **Comments** field, write Changed the material to a low alloy steel via a draft.
- 3 Click **Save** ().

The **Stresses and Strains in a Wrench** draft model () is automatically deleted and no longer visible in the **Model Manager** window. Select the **Stresses and Strains in a Wrench** regular model (). The **Versions** window shows five versions for the model.

Working with Commits

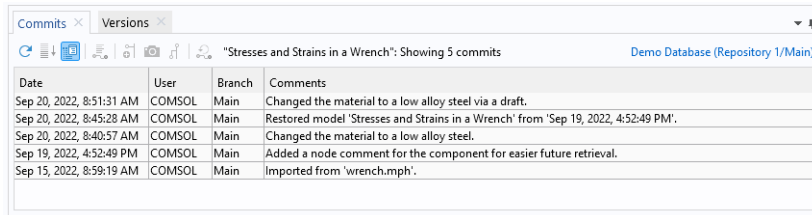
At this point, you may be satisfied with the new low alloy steel material for the wrench. But what if you realize that you wanted to perform further changes to your draft before those changes were saved back to the regular model? You could create a new draft and continue your work, but in the meantime the regular model would contain changes that you are perhaps not yet prepared to share with your coworkers. You could restore the previous version of the regular model using **Restore Version** (🕒) from the **Versions** window, but then you would have to redo the work you did in the first draft.

Model Manager solves this issue via the concept of *commits*. Every time you perform changes that involve items in a Model Manager database, all those changes are saved as a *collective whole* in a *commit*. Each commit identifies what was changed, when those changes were performed, and the user that performed the changes. Saving a new version of an item is a special case of such a change. Other examples include deleting items and assigning tags to items.

A useful property of commits is that they can be *reverted*. Reverting a commit means to save a new commit in which the opposite changes to those in the reverted commit are performed.

- 1 Right-click the **Stresses and Strains in a Wrench** regular model (📦) in the **Model Manager** window and select **Commits** (🕒).

The **Commits** window is opened in the Model Manager workspace. The window shows all commits in which the selected model was changed. The commits are sorted in chronological order with the latest commit at the top.




Date	User	Branch	Comments
Sep 20, 2022, 8:51:31 AM	COMSOL	Main	Changed the material to a low alloy steel via a draft.
Sep 20, 2022, 8:45:28 AM	COMSOL	Main	Restored model 'Stresses and Strains in a Wrench' from 'Sep 19, 2022, 4:52:49 PM'.
Sep 20, 2022, 8:40:57 AM	COMSOL	Main	Changed the material to a low alloy steel.
Sep 19, 2022, 4:52:49 PM	COMSOL	Main	Added a node comment for the component for easier future retrieval.
Sep 15, 2022, 8:59:19 AM	COMSOL	Main	Imported from 'wrench.mph'.

- 2 Select the top table row and click the **Commit Details** button (📄) in the toolbar. You can also double-click the table row.

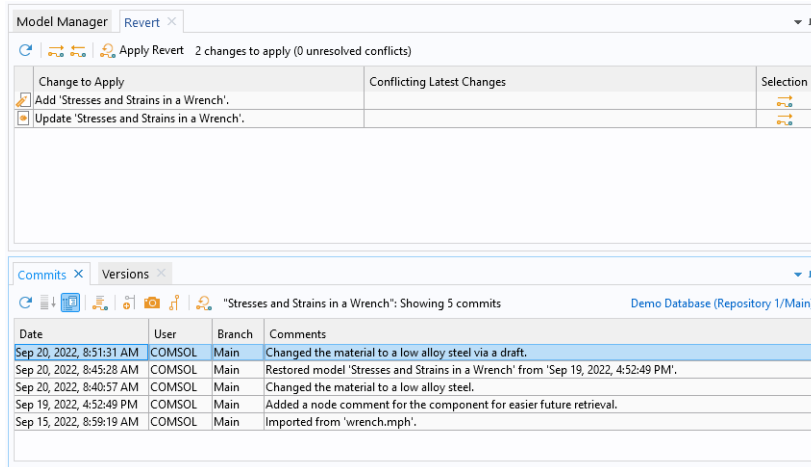
The **Commit Details** dialog box is opened showing details for the commit in which the draft model was saved back to the regular model. As seen in the **Changes** table, two item changes were performed in this commit: a new version of the regular model (📦) was saved and the draft model (✏️) was deleted.


3 Click **OK** to close the **Commit Details** dialog box.

You will revert the changes made in the latest commit. This will restore the regular model to its previous version *and* recover the deleted draft.

1 Select the top table row and click the **Revert** button () in the toolbar.

The **Revert** window is opened in the Model Manager workspace. The table contains the changes that reverts the selected commit.







2 Click the **Apply Revert** button () in the toolbar.

The **Apply Revert** dialog box is opened with a suggested commit comment.

3 Change to Reverted updates to model made from draft in the **Comments** field. Click **OK**.

The reverting commit is saved to the database and the **Revert** window is automatically closed.


4 Click the **Refresh** button () in the **Model Manager** window. Both the **Stresses and Strains in a Wrench** regular model () and the **Stresses and Strains in a Wrench** draft model () are shown in the search result.

5 Right-click the regular model and select **Versions** (). You can see that the **Versions** window now contains a later version of the regular model than the bold-highlighted

one opened in the COMSOL Desktop. Select the draft model and verify that there are two versions of the draft — these two versions are identical in content.



Reverting






At this point, you could open the **Stresses and Strains in a Wrench** draft model () and continue your draft work. For the conclusion of this part of the tutorial, instead delete the draft again.

- 1 Right-click the **Stresses and Strains in a Wrench** draft model () in the **Model Manager** window and select **Delete** ().

The **Delete Draft** dialog box is opened. The **Item** table contains the draft model, the sole item that will be deleted.

- 2 Click **OK** to delete the draft.

You should view the list of commits in the **Commits** window as a history of changes made in the database. To see the complete list of all changes going back to when the database was created:




- 1 Click the **Tree** button () in the **Model Manager** window to switch to the **Tree View**.
- 2 Right-click the **Main** root node () in the tree and select **Commits** () .
The **Commits** window shows the 100 most recent commits saved in the demo database. You can double-click some of the table rows to see which items were involved in each commit from the **Commit Details** dialog box.
- 3 Click the **Show More** button () a few times to append older commits to the bottom of the table — there are about 4 000 commits in total.
- 4 Click the **Table** button () in the **Model Manager** window to switch back to the **Table View**.




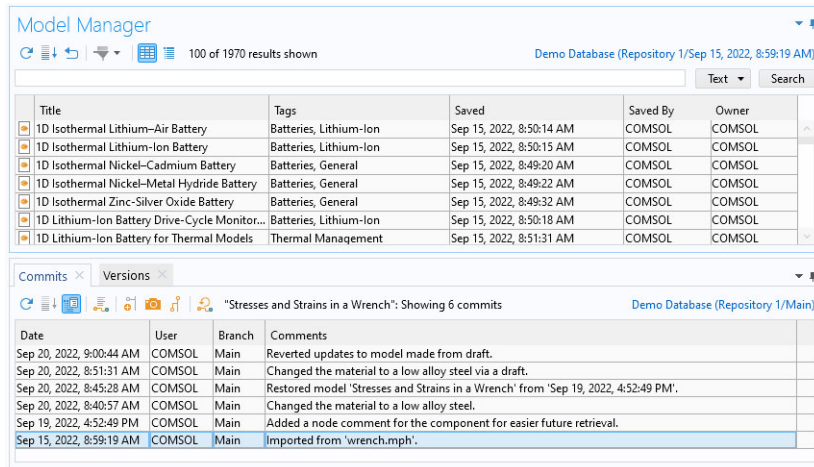
The Commits Window

BROWSING AND SEARCHING WITH RESPECT TO COMMITS

Commits serve a secondary purpose in a Model Manager database beyond grouping a collection of changes: a commit can be used to browse and search the particular state of your database at the time when the commit was saved.

- 1 Right-click the **Stresses and Strains in a Wrench** regular model () and select **Commits** ().
- 2 In the **Commits** window, right-click the last table row at the bottom and select **Search in Commit** ().


The **Model Manager** window is updated to show the latest item versions *at the time* the wrench model was first imported in the database. Click the **Reset** button (). You will notice that the database had about one thousand less item at that time than when the original import completed.



The screenshot shows the Model Manager interface. The top window displays a table of items with columns: Title, Tags, Saved, Saved By, and Owner. Below it, the Commits window is open, showing a table of commits with columns: Date, User, Branch, and Comments.

Title	Tags	Saved	Saved By	Owner
1D Isothermal Lithium-Air Battery	Batteries, Lithium-Ion	Sep 15, 2022, 8:50:14 AM	COMSOL	COMSOL
1D Isothermal Lithium-Ion Battery	Batteries, Lithium-Ion	Sep 15, 2022, 8:50:15 AM	COMSOL	COMSOL
1D Isothermal Nickel-Cadmium Battery	Batteries, General	Sep 15, 2022, 8:49:20 AM	COMSOL	COMSOL
1D Isothermal Nickel-Metal Hydride Battery	Batteries, General	Sep 15, 2022, 8:49:22 AM	COMSOL	COMSOL
1D Isothermal Zinc-Silver Oxide Battery	Batteries, General	Sep 15, 2022, 8:49:32 AM	COMSOL	COMSOL
1D Lithium-Ion Battery Drive-Cycle Monitor...	Batteries, Lithium-Ion	Sep 15, 2022, 8:50:18 AM	COMSOL	COMSOL
1D Lithium-Ion Battery for Thermal Models	Thermal Management	Sep 15, 2022, 8:51:31 AM	COMSOL	COMSOL

Date	User	Branch	Comments
Sep 20, 2022, 9:00:44 AM	COMSOL	Main	Reverted updates to model made from draft.
Sep 20, 2022, 8:51:31 AM	COMSOL	Main	Changed the material to a low alloy steel via a draft.
Sep 20, 2022, 8:45:28 AM	COMSOL	Main	Restored model 'Stresses and Strains in a Wrench' from 'Sep 19, 2022, 4:52:49 PM'.
Sep 20, 2022, 8:40:57 AM	COMSOL	Main	Changed the material to a low alloy steel.
Sep 19, 2022, 4:52:49 PM	COMSOL	Main	Added a node comment for the component for easier future retrieval.
Sep 15, 2022, 8:59:19 AM	COMSOL	Main	Imported from 'wrench.mph'.


- 3 Right-click the table row with the comment **Changed the material to a low alloy steel** in the **Commits** window and select **Search in Commit** ().

The **Model Manager** window is updated to show the latest item versions at the time when you replaced the material in the wrench model for the first time in this tutorial.

The most useful commit to search in the **Model Manager** window is the latest one, which is also the default behavior. To switch back to searching with respect to this commit:

- 1 Click the link button in the upper-right corner in the **Model Manager** window — you will find it above the **Search** button.

The **Select Location** dialog box is opened enabling you to select the *commit location* to search with respect to.

- 2 Select the **Main** branch node () in the tree. The branch serves as an implicit stand-in for the latest commit saved in the database.



You will return to the concept of branches later in this tutorial.

- 3 Click **OK**.



Locations


You can record a *snapshot* of a particular commit in case the state of the database holds a special meaning at the point in time when the commit was saved. One such commit for the demo database is the last one in which a file was imported from the COMSOL Application Libraries.

- 1 Click the link button in the upper-right corner in the Model Manager window.
- 2 Under **Snapshots** () , select the single leaf snapshot node () named after the version number of the COMSOL Multiphysics software used to import the COMSOL Application Libraries.
- 3 Click **OK**.

The Model Manager window shows the model and data file versions that were the latest versions in the database when the demo database was initially prepared.



Recording Snapshots

Restore the search in the **Model Manager** window by clicking the link button in the upper-right corner in the **Model Manager** window and selecting the **Main** branch node () in the **Select Location** dialog box. You can also restore the search by selecting the demo database via the database selector expand button in the **Database** section of the **Home** toolbar.

Using a Branch to Update Many Models

In the previous example, you used a draft to replace a material in a model. This enabled you to do the change in isolation and at your own pace — all while leaving the main model untouched in the interim. This strategy is the recommended approach when working on changes to a *single* model, but how should you proceed if there is more than one model you need to update? You could create drafts of all the models, do your updates to each draft, and then save each draft back to its original model. This strategy works in principle but is rather tedious and risks leaving the models in a half-finished state while your work is progressing. If you later want to undo your changes, there would potentially be a lot of cleaning up to do in the form of reverting commits.




As a concrete example, say you have a few models that all share the same CAD geometry. Imagine that your CAD engineer colleague has sent you a new geometry that you would like to update your models with. If the models are stored on a file system shared with other simulation engineers, you perhaps begin by copying all the MPH-files to a new folder on the file system. You replace the geometry in the models using the new CAD data file, test out your changes, and then overwrite the original MPH-files when you are satisfied with the update.

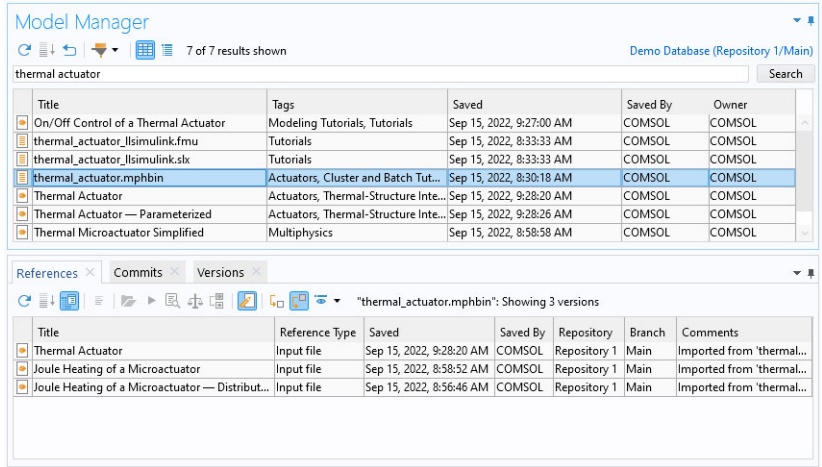
Model Manager enables you to solve such a multi-update problem for models and data files using the concept of *branches*. A branch is another name for the sequence, or *history*, of commits shown in the **Commits** window. You can create a new branch from the initial **Main** branch, thereby creating an *alternative history* of commits. You can think of the commits on the **Main** branch as the trunk of a growing tree, with the oldest commit found at the base of the tree. The alternative commit sequences are tree branches growing out from this tree trunk. When you save commits on such a new branch, any changes made to models and data files are invisible to users working on the **Main** branch — thereby enabling you to do your changes in isolation until you are ready to share your work by *merging* the changes to the **Main** branch.

CREATING THE NEW BRANCH




As an example of working simultaneously with many model changes, you will replace a CAD geometry used as an input file by a few models in the demo database for Model Manager.

- 1 Write `thermal actuator` in the text field in the **Model Manager** window and click **Search**.





- Right-click the **thermal_actuator.mphbin** data file () and select **References** (). The **References** window is opened in the Model Manager workspace showing three models using the selected data file as an input file. The selected file is a CAD geometry stored in the COMSOL native CAD format **mphbin**, which the models all reference via a geometry **Import** node ().





You will assign a tag to the three models and the data file to collect them together in the database.

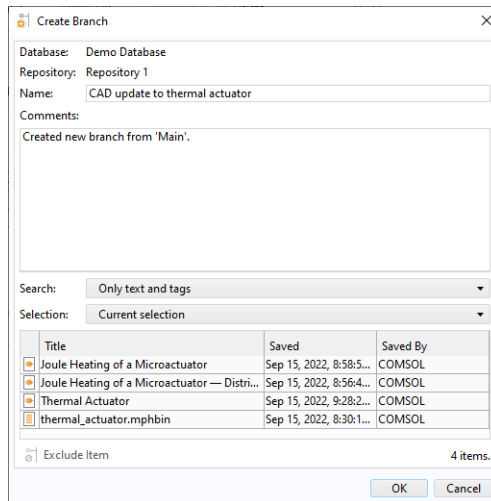
- Press **Ctrl** and select the **thermal_actuator.mphbin** data file () and the **Thermal actuator** model () in the **Model Manager** window.
- On the **Home** toolbar, in the **Database** section, click the **New Tag** button ().
- In the **Title** field, write **CAD Update Project**.
- Click **OK**.



A new tag is created in the database. The tag is also assigned to the CAD data file and one of the models referencing the file.

- Write **joule heating** and click **Search**.
- Press **Ctrl** and select the **Joule Heating of a Microactuator** model () and the **Joule Heating of a Microactuator — Distributed Parameter Version** model ().
- On the **Home** toolbar, in the **Database** section, click the **Set Tags** button ().
- In the **Set Tags** dialog box, select the **CAD Update Project** tag tree node ().
- Click **OK**.

You are now ready to create a new branch in your database that contains the three models and the data file.


- 1 Write `cad update project` and click **Search**.
- 2 Press **Ctrl** and select the three models and the `thermal_actuator.mphbin` data file () in the **Model Manager** window.
- 3 Right-click any one of the selected items and select **Branch** ()
The **Create Branch** dialog box is opened with the three models and the single data file listed in a table at the bottom of the dialog box.
- 4 In the **Name** field, write `CAD update to thermal actuator`.
- 5 In the **Search** list, select `Only text and tags`. This reduces the disk space used by indexed search data at the expense of less search and filter capabilities on the new branch. This is a reasonable trade-off as the new branch will only contain four, easy-to-find, items.
- 6 Click **OK**.



You could have also right-clicked the **CAD Update Project** tag tree node () in the **Tree View** of the **Model Manager** window and selected **Branch** ()




Branching


A branch containing the three models and the CAD geometry data file has been created in the database. Click the **Reset** button () in the Model Manager window — you will indeed only see four items in the search result.



You may find the empty search result obtained for `cad update project` surprising when the corresponding search on the **Main** branch yielded four items. A branch whose **Search** field has been set to **Only text and tags** does not support searching on text fields such as the title or description *and* the titles of assigned tags at the same time. Write `cad update project` and select **Tags** from the list next to the **Search** button in the **Model Manager** window to match all four items again. See [Matching Only Text and Tags](#) for more details.

An initial commit was saved to the database when the **CAD update to thermal actuator** branch was created. You can see this commit as the top table row in the **Commits** window. The table also includes the commits on the **Main** branch up to the point where the new branch was created.

- 1 Click the link button in the upper-right corner in the **Model Manager** window — you will find it above the **Search** button.
- 2 In the **Select Location** dialog box, select the **Main** branch node () in the tree.
- 3 Click **OK**.

The **Model Manager** window is updated to show the latest versions of items on the **Main** branch. Also, the **Commits** window only shows the commits on the **Main** branch, not on the **CAD update to thermal actuator** branch. To return to your new branch, click the link button in the **Model Manager** window, select the **CAD update to thermal actuator** branch node () in the tree, and click **OK**.

Double-click the top table row in the **Commits** window. The **Commit Details** dialog box is opened for the initial commit on the **CAD update to thermal actuator** branch. You will find that the **Changes** table is empty as no items were changed when the new branch was created — the latest item versions and the tag assignments on the new branch are,




at least initially, *identical* to those on the **Main** branch. You can compare this with the file system analogy of copying MPH-files to a new location on the file system.





While the **CAD update to thermal actuator** branch contains copies of the four versions selected in the **Main** branch, the new branch uses little additional disk space for these copies thanks to data deduplication in Model Manager.

UPDATING THE CAD INPUT FILE

You will update the CAD data file in the database with a file found in the COMSOL installation folder.

- 1 Select the **thermal_actuator.mphbin** data file () in the **Model Manager** window.
- 2 In the **Description** field in the **Settings** window, write CAD geometry for a thermal actuator.
- 3 In the **Contents** section in the **Settings** window, select **thermal_actuator.mphbin**.
- 4 Click the Replace button () and locate the file `thermal_actuator.mphbin` in the application library folder of the COMSOL installation folder. Its default location in Windows[®] is
`C:\Program Files\COMSOL\COMSOL61\Multiphysics\applications\
COMSOL_Multiphysics\Multiphysics\thermal_actuator.mphbin`
Double-click to replace, or click **Open**.
- 5 Click **Save** () in the **Settings** window.
- 6 Click **OK** in the **Save File** dialog.

The CAD data file is now updated with a new version on the **CAD update to thermal actuator** branch. Right-click the **thermal_actuator.mphbin** data file () and select **Versions** () to open the **Versions** window for the file. The window shows all versions of the file *with respect to the CAD update to thermal actuator* branch. The top table row is the latest version on the branch, the middle row is an identical copy of the version found on the **Main** branch. For convenience, the original version on the **Main** branch is

also appended at the bottom of the table. This last version is also the only version that users currently browsing the **Main** branch sees.

Title	Saved	Saved By	Branch	Comments
thermal_actuator.mphbin	Sep 20, 2022, 9:36:01 AM	COMSOL	CAD update to th...	Updated 'thermal_actuator.mphbin' from 'thermal_actuator.mphbin'.
thermal_actuator.mphbin	Sep 20, 2022, 9:30:46 AM	COMSOL	CAD update to th...	Created new branch from 'Main'.
thermal_actuator.mphbin	Sep 15, 2022, 8:30:18 AM	COMSOL	Main	Imported from 'thermal_actuator.mphbin'.

UPDATING THE MODELS

You will continue your work on the branch by using the new CAD geometry in your models.

- 1 Double-click the **Thermal actuator** model () in the **Model Manager** window.
The model is opened in the Model Builder workspace.
- 2 In the **Model Builder** window, select **Geometry 1 > Import** ().
- 3 In the **Settings** window, click the expand button next to **Browse** () and select **Browse From** ().
- 4 In the **Select File** window, select the demo database in the list of options.
- 5 Click the **Main** link button — you will find it above the **Search** button.
- 6 In the **Select Location** dialog box, select the **CAD update to thermal actuator** branch node (). Click **OK**.
- 7 Double-click the **thermal_actuator.mphbin** data file () to select it as a new input file.

The model now references the updated CAD geometry in the new branch. Click the **Import** button () to import the geometry into the model. Finish by saving a new version of the model.

- 1 From the **File** menu, select **Save To** ().
The **Save** window is opened for the demo database with the new branch automatically selected in the **Location** field.
- 2 In the **Comments** field, write Updated geometry using new CAD data.
- 3 Click **Save** ().

A new version of the model is saved to the **CAD update to thermal actuator** branch. Much like the updated data file, this model version is not visible on the **Main** branch.

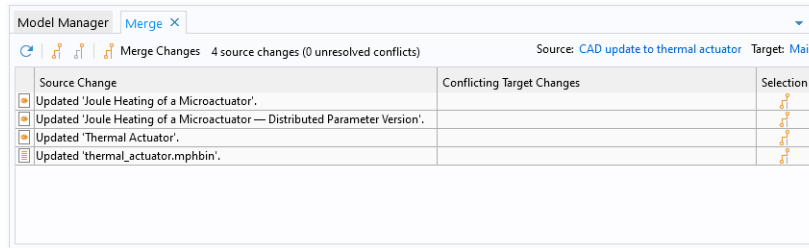
Open the Model Manager workspace and repeat the update for the remaining two models, **Joule Heating of a Microactuator** and **Joule Heating of a Microactuator — Distributed Parameter Version**. In the **Select File** window, you may use the **Recent** (🔍) list option to quickly select the **thermal_actuator.mphbin** data file (📄).

MERGING THE CHANGES

With the model updates completed, the final step is to *merge* the changes from the **CAD update to thermal actuator** branch to the **Main** branch. This will save the latest versions on the *source* branch as new latest versions on the *target* branch.

- 1 Open the Model Manager workspace.
- 2 On the **Database** toolbar, in the **Database** section, click the **Databases** button (📄) to open the **Databases** window.
- 3 Right-click the **CAD update to thermal actuator** branch node (🔗) and select **Merge** (🔗).

The **Merge** window is opened in the Model Manager workspace. The window lists all changes that will be merged from the source branch to the target branch — in this case there are four changes corresponding to the updated items.



- 4 Click the **Merge Changes** button (🔗) in the toolbar.
- 5 In the **Merge** dialog box, click **OK**.

The **Merge** window is closed and the four changes are saved as a single commit to the **Main** branch.

- 6 Select the **Main** branch node (🔗) in the **Databases** window.



The **Model Manager** window is automatically updated to show the search result for **Main** branch node.



Merging

The three models and the data file are now updated on the **Main** branch. At this point, you could either continue your work on the **CAD update to thermal actuator** branch, save versions of your models, and merge any new changes to the **Main** branch.

Opting instead to delete the branch.

- 1 Right-click the **CAD update to thermal actuator** branch node () and select **Delete** ()
- 2 Click **Yes** in the **Delete Branch** dialog box.

This concludes this tutorial on the version control tools available in the Model Manager. While creating branches is useful when you want to make sweeping changes involving many models and data files, it comes at the cost of added complexity. You should always consider if a draft can solve your particular version control problem before you create a new branch.

Glossary

This [Glossary of Terms](#) contains terms related to databases and version control as they relate to the Model Manager tools in the COMSOL Multiphysics® software and documentation. For references to further information about a term, see the [index](#).

Glossary of Terms

access-control list A list of granted permissions and their grantees — that is, users and groups — assigned to a database object.

auxiliary data Data that is referenced as input or output by a model in the COMSOL Desktop environment. The data may, for example, be a version-controlled data file in a Model Manager database or a file on the file system.

boolean field The type for a field whose value is either `true` or `false`.

branch A sequence of commits in a database, ordered chronologically by their commit date to form a *commit history*. A special commit is the most recent one on the branch, and the branch itself often acts as a representative of this commit. The most recent commit, and thus the branch itself, also identifies the latest versions of items.

New branches can be created by *branching off* from any commit on a *parent branch*, thereby starting an alternative commit history. Changes made on a new branch can be *merged* back to its parent branch.

built, computed, and plotted data Generated simulation data that can be recreated from a model as needed. Includes built geometries and meshes, computed solutions, and plotted results.

commit A set of related item changes that have been saved to a database. The changes may involve adding and updating items, assigning tags to items, and deleting items. The changes are saved to the database as a “unit” and, as such, can also be *reverted* as a unit.

Each commit is associated with a date and time when the changes were saved, that is the *commit date*, the user that saved the changes, and an optional save comment. Each commit also identifies the set of item versions that were the *latest versions* at the time of the commit, as well as all tag assignments present at that time. Given a particular commit, such item versions and tag assignments can be browsed and searched in the database.

content filter A search filter on model content — that is, node properties, parameters, features, and other settings in the model tree of a model.

data file A version-controlled file stored in a database that is neither a model or application file (mph) nor a physics file (mphphb). Typically abbreviated as just *file* in Model Manager. Examples include CAD data, interpolation functions, plots, and reports.

database configuration The configuration values used to connect to a local database or a server database accessed via a Model Manager server.

date field The type for a field whose value is a date and time.

draft model A transient model meant for intermediate modeling work, and whose versions are saved as a separate version history *split off* from that of its *origin* model. Once a draft has been completed, it can be saved back as a new version of the model it originated from.

everyone A special group of users that all users automatically are members of. Can be used when assigning permissions to database objects.

file A shorthand for a data file.

fileset An item consisting of multiple data files that are version controlled as a collective whole in a database. This includes, for example, a CAD assembly with multiple external component files.

item A model, data file, or tag stored in a database.

item filter A search filter on general item settings and metadata. This includes, for example, the time when the item was last modified, the user that last modified the item, and the item's assigned tags.

item type Type distinguishing a model, application, or physics in a database.

group A collection of users and other groups.

keyword field The type for a field whose value is a string. Typically found for short, name-like search data such as filenames or node names.

local database A database stored on the same computer as the COMSOL Multiphysics process runs on. Meant for single-user use.

(commit) location A branch, commit, or snapshot in the database. Each location is either a commit in its own right or acts as a natural representation of a commit — the most recent commit for a branch and the referenced commit for a snapshot.

Browsing or searching items in a database is always done with respect to a fixed location. The encountered item versions are the ones that were the latest at the time of the commit. For a branch, which is the default location, these item versions are the latest at the present time.

(item version) location An identifier that uniquely locates a model version or data file version stored in a Model Manager database.

merge The operation of applying item changes made in one branch to another branch.

model A COMSOL Multiphysics simulation model, application, or physics stored in a database.

negated match Reverse the matching criterion of a filter.

numeric field The type for a field whose value is a real or complex scalar.

origin model A model that another model “originates” from — a concept arising, for example, when creating a new model from an existing model or when saving a new draft.

owner The user that owns a database object. Such a user can set permission requirements for accessing the object.

permission template A reusable template of permissions granted to users and groups.

phrase match Require that search words match a sequence of words in a text.

regular model The standard type for models saved to a database.

repository A container for a collection of items and their versions in the database.

revert The operation of undoing a set of changes made in a commit.

root administrator A special account with a Model Manager server that always passes permission requirement checks.

save type Type distinguishing a regular model from a draft model.

selection field The type for a field whose value belongs to a predetermined set of values.

server database A database accessed via a Model Manager server. Meant for multiple-user use.

snapshot A reference to a commit in a database. The item versions and tag assignments that were the latest at the time of the commit are *recorded* in the snapshot.

tag Version-controlled metadata that can be assigned to models, data files, and even other tags. Useful for finding and organizing items in a database.

text field The type for a field whose value is a text.

user A user that has connected to a Model Manager database.

version The result when creating, updating, or restoring an item in a database.

wildcard match Use a placeholder in a search word that matches zero or more arbitrary characters.

I n d e x

- A**
 - access-control lists 113
 - add database (window) 22
 - adding
 - branches 161
 - databases 21
 - groups 109
 - permission requirements 112
 - permission templates 119
 - repositories 97
 - snapshots 103
 - tag assignments 96
 - tags 95
 - users 108
 - assigning tags 96
 - auxiliary data (window) 46
 - auxiliary data files 59
 - deleting 101
 - exporting 106
 - importing 104
 - permanently deleting 122
 - previewing 91
 - renaming 99
 - restoring 100
 - settings 60
- B**
 - backup, manual 130
 - backward compatibility 27
 - boolean fields 139
 - branches 64
 - creating 161
 - default 65
 - deleting 77
 - full text search in 135
 - merging 165
 - partial 163
 - restoring 77
 - searching in 132
 - settings 65
 - built, computed, and plotted data 121
 - deleting 125
- C**
 - cloud drives 22
 - cluster 90
 - commits 63
 - comments 81
 - details 80
 - full text search in 136
 - history 79
 - locations 67
 - merging from 165
 - reverting 169
 - saving 98
 - searching in 133
 - sequence 160
 - commits (window) 79
 - toolbar 80
 - comparing
 - different models 41
 - versions 40
 - with latest version 41
 - with opened model 41
 - comparison result (window) 40
 - content filters 143
 - author (node) 146
 - comment (node) 146
 - created (node) 145
 - label (node) 145
 - last modified (node) 146
 - last modified by (node) 146
 - name (node) 145
 - parameter 143
 - physics 144
 - setting 144
 - space dimension 144

- study step 145
 - tag (node) 145
 - type (node) 145
 - version (node) 146
- copy locations 91
- creating
 - branches 161
 - groups 109
 - local databases 22
 - permission templates 119
 - repositories 97
 - snapshots 103
 - tags 95
 - users 108
- D** database configurations 127
 - deleting 127
 - settings 127
- database permissions 114
- database toolbar 50
- databases
 - adding 21
 - auxiliary data 46
 - backward compatibility 27
 - cleanup 129
 - comparing models from 40
 - configuring 127
 - local 21
 - opening models from 29
 - saving models to 31
 - server 21
- databases (window) 75
 - toolbar 76
 - tree 76
- date fields 138
- date shorthands 138
- default
 - branch 65
 - repository 68
- deleting
 - branches 77
 - database configurations 127
 - groups 77
 - items 101
 - permission templates 77
 - repositories 77
 - snapshots 77
 - users 77
- draft models 55
 - saving 36
- E** editing
 - auxiliary data files 60
 - branches 65
 - comments 81
 - database configurations 127
 - groups 69
 - models 57
 - permission templates 71
 - repositories 68
 - snapshots 66
 - tags 62
 - users 69
- emailing COMSOL 12
- escaping reserved characters 151
- export (window) 45
- exporting
 - items 106
 - to new local database 125
- F** field expressions 147
- field types 137
 - boolean 139
 - date 138
 - keyword 137
 - numeric 138
 - selection 139
 - text 137
- files 59

- copy location 92
 - filesets 59
 - filtering 137
 - content 143
 - items 140
 - full text search 134
- G** granting permissions 112
- groups 69
 - adding 109
 - deleting 77
 - group members 110
 - restoring 77
 - settings 69
- H** home toolbar 48
- I** importing items 104
- index maintenance 128
- indexes directory 23
- internet resources 12
- item filters 140
 - description 141
 - file type 142
 - filename 142
 - item type 141
 - last modified 141
 - last modified by 142
 - owner 142
 - save type 141
 - tag 141
 - title 141
- item types 54, 59
- items 62
 - deleting 101
 - renaming 99
 - saving versions 98
 - version locations 91
- K** keyboard shortcuts
 - deleting 101
 - opening workspaces 48
 - renaming 99
- keyword fields 137
- knowledge base, COMSOL 13
- L** local databases 21, 129
 - compacting 129
 - creating 22
 - multiple COMSOL Multiphysics processes 24
 - opening 23
 - permanently deleting 130
- locations
 - commits 67
 - item versions 91
 - searching in 132
 - selecting 67
- M** maintenance (window) 123
 - toolbar 124
- merge (window) 166
 - toolbar 166
- merge conflicts 167
 - manually resolving 167
- merging 165
- model content 57
 - searching 143
- Model Manager
 - database 53
 - database toolbar 50
 - home toolbar 48
 - opening workspace 48
 - workspace windows 51
- model manager (window) 72
 - table view 73
 - toolbar 73
 - tree view 74
- Model Manager server 21
- models 53
 - built, computed, and plotted data 121

- comparing 40
 - copy location 92
 - deleting 101
 - drafts 55
 - exporting 106
 - importing 104
 - locking 89
 - origin 39
 - permanently deleting 122
 - renaming 99
 - restoring 100
 - saving 31
 - saving drafts 36
 - settings 57
- moving 129
- N**
 - named nodes 153
 - negated matching 150
 - network drives 22
 - node field expressions 152
 - numeric fields 138
- O**
 - open (window) 29
 - toolbar 30
 - opening
 - local databases 23
 - models 29
 - on cluster 90
 - origin models 39
 - owners 111
 - transferring ownership 111
- P**
 - partial branches 163
 - permanently deleting
 - items 122
 - local databases 130
 - permission templates 77
 - version 122
 - permission templates 70
 - adding 119
 - deleting 77
 - permanently deleting 77
 - restoring 77
 - settings 71
 - permissions
 - catalog 114
 - custom 113
 - everyone 114
 - granting 112
 - levels 117
 - owner 114
 - templates 70
 - phrase matching 150
 - preferences
 - cluster 90
 - databases directory 23
 - enable Model Manager 20
 - unsaved settings 78
- R**
 - range matching 151
 - recording snapshots 103
 - references (window) 85
 - toolbar 87
 - regular models 54
 - removing
 - permission requirements 112
 - tag assignments 96
 - renaming items 99
 - repositories 67
 - adding 97
 - default 68
 - deleting 77
 - restoring 77
 - settings 68
 - resources directory 23
 - restoring
 - branches 77
 - groups 77
 - permission templates 77

- repositories 77
- snapshots 77
- users 77
- versions 100
- revert (window) 169
 - toolbar 170
- revert conflicts 170
 - manually resolving 171
- reverting 169
- root tag 96
- S** save (window) 31
- save types 54
- saving
 - drafts 36
 - models 31
- search syntax 147
- searching
 - escaping reserved characters 151
 - filters 137
 - full text search 134
 - index maintenance 128
 - negation 150
 - operator precedence 149
 - phrases 150
 - ranges 151
 - syntax 147
 - syntax catalog 155
 - wildcards 149
- select file (window)
 - input 42
 - output 43
 - toolbar 43
- select model (window) 37
- selection fields 139
- server databases 21
 - connecting to 25
- setting field expressions 154
- settings (window) 77
 - auxiliary data files 60
 - branches 65
 - database configurations 127
 - groups 69
 - models 57
 - permission templates 71
 - repositories 68
 - snapshots 66
 - tags 62
 - toolbar 79
 - users 69
- snapshots 66
 - deleting 77
 - full text search in 136
 - merging from 165
 - recording 103
 - restoring 77
 - searching in 133
 - settings 66
- SQLite 23
- T** tagging 96
- tags 61
 - assigning 96
 - creating 95
 - deleting 102
 - exporting 107
 - importing 105
 - pins 34
 - renaming 99
 - restoring 100
 - root 96
 - settings 62
- technical support, COMSOL 12
- text fields 137
- toolbar
 - commits (window) 80
 - database 50
 - databases (window) 76

- home 48
 - maintenance (window) 124
 - merge (window) 166
 - model manager (window) 73
 - open (window) 30
 - references (window) 87
 - revert (window) 170
 - select file (window) 43
 - settings (window) 79
 - versions (window) 83
 - versions (window) (Model Builder) 39
- U** users 68
- adding 108
 - deleting 77
 - group memberships 109
 - restoring 77
 - settings 69
- V** versions
- comments 85
 - details 84
 - history 82
 - locations 91
 - maintenance 123
 - permanently deleting 122
 - references 85
 - restoring 100
 - saving 98
 - searching 132
 - versions (window) 82
 - Model Builder 38
 - toolbar 83
- W** websites, COMSOL 13
- wildcard matching 149
 - windows
 - add database 22
 - auxiliary data 46
 - commits 79
 - comparison result 40
 - databases 75
 - export 45
 - maintenance 123
 - merge 166
 - model manager 72
 - open 29
 - references 85
 - revert 169
 - save 31
 - select file 42–43
 - select model 37
 - settings 77
 - versions 82
 - versions (Model Builder) 38
 - working copies 44