



DxDataBook™ User's Guide

Software Version EE 7.9

**© 1999-2010 Mentor Graphics Corporation
All rights reserved.**

This document contains information that is proprietary to Mentor Graphics Corporation. The original recipient of this document may duplicate this document in whole or in part for internal business purposes only, provided that this entire notice appears in all copies. In duplicating any part of this document, the recipient agrees to make every reasonable effort to prevent the unauthorized use and distribution of the proprietary information.

This document is for information and instruction purposes. Mentor Graphics reserves the right to make changes in specifications and other information contained in this publication without prior notice, and the reader should, in all cases, consult Mentor Graphics to determine whether any changes have been made.

The terms and conditions governing the sale and licensing of Mentor Graphics products are set forth in written agreements between Mentor Graphics and its customers. No representation or other affirmation of fact contained in this publication shall be deemed to be a warranty or give rise to any liability of Mentor Graphics whatsoever.

MENTOR GRAPHICS MAKES NO WARRANTY OF ANY KIND WITH REGARD TO THIS MATERIAL INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

MENTOR GRAPHICS SHALL NOT BE LIABLE FOR ANY INCIDENTAL, INDIRECT, SPECIAL, OR CONSEQUENTIAL DAMAGES WHATSOEVER (INCLUDING BUT NOT LIMITED TO LOST PROFITS) ARISING OUT OF OR RELATED TO THIS PUBLICATION OR THE INFORMATION CONTAINED IN IT, EVEN IF MENTOR GRAPHICS CORPORATION HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

RESTRICTED RIGHTS LEGEND 03/97

U.S. Government Restricted Rights. The SOFTWARE and documentation have been developed entirely at private expense and are commercial computer software provided with restricted rights. Use, duplication or disclosure by the U.S. Government or a U.S. Government subcontractor is subject to the restrictions set forth in the license agreement provided with the software pursuant to DFARS 227.7202-3(a) or as set forth in subparagraph (c)(1) and (2) of the Commercial Computer Software - Restricted Rights clause at FAR 52.227-19, as applicable.

Contractor/manufacturer is:

Mentor Graphics Corporation

8005 S.W. Boeckman Road, Wilsonville, Oregon 97070-7777.

Telephone: 503.685.7000

Toll-Free Telephone: 800.592.2210

Website: www.mentor.com

SupportNet: supportnet.mentor.com/

Send Feedback on Documentation: supportnet.mentor.com/doc_feedback_form

TRADEMARKS: The trademarks, logos and service marks ("Marks") used herein are the property of Mentor Graphics Corporation or other third parties. No one is permitted to use these Marks without the prior written consent of Mentor Graphics or the respective third-party owner. The use herein of a third-party Mark is not an attempt to indicate Mentor Graphics as a source of a product, but is intended to indicate a product from, or associated with, a particular third party. A current list of Mentor Graphics' trademarks may be viewed at: www.mentor.com/trademarks.

Table of Contents

Chapter 1

Introduction to DxDataBook	9
DxDataBook Features	10
DxDataBook Requirements	10
Libraries	11
Data Sources	12
Configuration File	13

Chapter 2

Planning Your Library Strategy	15
DxDataBook Information Gathering Process	15
DxDataBook Component Data Structure	16
Shared Symbol Component Data	17
Property-Based Component Data	19
DxDataBook Library Structure	19
Single and Multiple Databases	19
Symbol Library Categories	20
Property Tracking	21
DxDataBook Database Tables	26
Data Source Creation	26
Table Column Constructs	27
Number and String Data	28

Chapter 3

Choosing a Data Source	29
DxDataBook: Supported Databases and Operating Systems	29
Oracle on UNIX/Linux	30
MySQL on UNIX/Linux	31
DxDataBook Data Sources	33
Client/Server Database System	34
DMS Connector Database	36
DxDataBook Web Server	46
Microsoft Access Database	52
Microsoft Excel Spreadsheets	54

Chapter 4

Defining Data Sources, Libraries, and Symbols in the Configuration File	57
Creating a DxDataBook Configuration File	58
Arranging Library Tables	59
Specifying Data Sources	63
Selecting Library Tables	64
Modifying Property Types	66

Modifying Property Names.	68
Specifying a Symbol for Library Components.	69
Editing a DxDataBook Configuration File	71
Editing Library Information	72
Specifying Symbols to Exclude	88
Setting Configuration File Preferences.	90
Connecting to a DxDataBook Configuration File	98
Filtering Library Data in the DxDataBook Configuration File	99
Modifying a DxDataBook Configuration File with the DBCtool Utility	101
 Chapter 5	
Placing and Verifying Schematic Components with DxDataBook.	103
Invoking DxDataBook.	103
Editing DxDataBook Properties	105
Working with Database Libraries in DxDataBook	109
Opening a Database Library	109
Searching for Components in a Database Library	110
Modifying Schematic Components with DxDataBook	117
Adding a Schematic Component	118
Annotating a Schematic Component	121
Setting Schematic Component Property Visibility.	123
Verifying Schematic Components with DxDataBook.	124
Verifying Components on a Single Schematic Sheet	125
Verifying a Design with Multiple Schematic Sheets	127
Viewing Component Data Sheets and Web Pages in DxDataBook	129
Viewing a Data Sheet	129
Viewing a Web Page.	130
 Chapter 6	
Scripting with DxDataBook.	131
DxDataBook Objects	131
Attributes Object.	132
Component Object	132
Properties Object.	133
DxDataBook Scripting Events.	140
Application_AddComponent Event	141
Application_AfterAddComponent Event	142
Application_AfterAnnotateComponent Event	143
Application_AnnotateComponent Event	144
Application_LoadComponent Event	145
Application_SelectComponent Event.	146
Application_ViewDocument Event	146
Loading a DxDataBook Script.	147
 Appendix A	
Frequently Asked Questions	149
DxDataBook Basics.	149
Setting Up and Configuring DxDataBook.	151

Table of Contents

Using DxDataBook	155
Using DxDataBook with a DxDesigner Schematic	159
Using DxDataBook on the Web	162
DxDataBook: System Administrator Tasks	164
Troubleshooting DxDataBook	165
 Appendix B	
Configuration File Format	169
 End-User License Agreement	

List of Figures

Figure 1-1. DxDataBook Design Environment	11
Figure 2-1. File-Based (Shared) xx04 Symbol	18
Figure 2-2. Inverter Symbol with Property Information	24
Figure 3-1. Client/Server Database System	35
Figure 3-2. DMS Connector, DxDataBook, DataFusion, and DMS Configuration	37
Figure 3-3. DMS Connector Installation	39
Figure 3-4. DataFusion Connection Parameters	41
Figure 3-5. DataFusion Production Library Restrictions	43
Figure 3-6. DxDataBook Web Server	47
Figure 3-7. DxDataBook to Microsoft Access Configuration	52
Figure 3-8. Adding a Microsoft Access Driver (Windows XP Example)	53
Figure 4-1. Library Wizard — Library Table Arrangement	60
Figure 4-2. Vertically Linked Tables	61
Figure 4-3. Horizontally Linked Tables	62
Figure 4-4. Library Wizard — Data Source Names	63
Figure 4-5. Library Wizard — Library Tables	65
Figure 4-6. Library Wizard — Property Types	66
Figure 4-7. Library Wizard — Property Names	68
Figure 4-8. Library Wizard — Library Components Symbol	69
Figure 4-9. Configure Dialog Box (Libraries Tab)	72
Figure 4-10. Configure Dialog Box — Adding a Library	73
Figure 4-11. Add Library Dialog Box — Advanced Options	75
Figure 4-12. Configure Dialog Box — Adding a Table	78
Figure 4-13. Reordering Libraries Dialog Box	86
Figure 4-14. Configure Dialog Box (Symbols Tab)	89
Figure 4-15. Configure Dialog Box (Preferences Tab)	91
Figure 5-1. DxDataBook Window (CL View — Default)	104
Figure 5-2. DxDataBook Properties Dialog Box	106
Figure 5-3. Multiple Search Conditions	112
Figure 5-4. Query Builder Dialog Box	113
Figure 5-5. DxDataBook Search Window	114
Figure 5-6. Adding a Generic Component	119
Figure 5-7. New Live Verification Window	125
Figure A-1. Internet/Intranet Integration Flow	163

List of Tables

Table 2-1. Resistor Library Properties Table	25
Table 3-1. Supported Database Formats and Operating Systems	29
Table 4-1. DxDataBook Changes/Configuration File Edits	71
Table 4-2. Field Type Icons	83
Table 4-3. Configuration > Preferences (Advanced)	92
Table 4-4. Configuration > Preferences (Annotate to Selected Component)	93
Table 4-5. Configuration > Preferences (Component Annotation)	93
Table 4-6. Configuration > Preferences (Instantiation/Annotation)	95
Table 4-7. Configuration > Preferences (Component Loading)	96
Table 4-8. Configuration > Preferences (Component Tracking)	98
Table 4-9. DxDataBook Library Filters	99
Table 5-1. DxDataBook Properties Dialog Box Contents	106
Table 5-2. Operators and Data Types	111
Table 5-3. Numeric Comparison Operators	115
Table 6-1. Attributes Object Properties	132
Table 6-2. Component Object Properties	132
Table 6-3. Properties Object Property	133
Table 6-4. Properties Object Methods	134
Table A-1. Supported DxDataBook Configuration Settings	152
Table B-1. XML File Structure	170
Table B-2. Element/Subelement Attributes	173

Chapter 1

Introduction to DxDataBook

DxDataBook™ is a database browser application that uses the Open Database Connectivity (ODBC) and Internet/Intranet standards to provide access to component information stored in an ODBC-compliant database.

DxDataBook design tasks include:

- Searching for a component in a component database based on library properties, and receive a list of components meeting the criteria you specify.
- Selecting an entire schematic, a portion of a schematic, or a component, load it into DxDataBook, and query the database.
- Adding properties to an existing component.
- Selecting an entire schematic, a portion of a schematic, or a component and verify that the parts exist in the corporate database.

The advantages of DxDataBook include reducing the time needed to search for schematic parts, making component datasheets and supporting information readily available from the component list, and ensuring that schematic part data is correct.

DxDataBook appears as a DxDataBook window. For information about invoking DxDataBook, opening and searching the database library, and adding and verifying schematic parts, see Chapter 5, “[Placing and Verifying Schematic Components with DxDataBook.](#)”

DxDataBook can also be invoked from the Dashboard and appears as a separate application. The Dashboard version differs in its Graphical User Interface (GUI), and unlike the integrated version, translates PAR or VDB files into the current non-proprietary format and displays SQL queries from the database.

Notes

- Mentor Graphics no longer supports the Dashboard version of DxDataBook, and recommends that you use the DxDesigner® version.
- A dxdblocal version of DxDataBook is available which works only with Microsoft Access and Excel data sources, and is only available in the PADS DS and ES suites.

Related Topics

- [DxDataBook Features](#)

- [DxDataBook Requirements](#)

DxDataBook Features

DxDataBook key features include:

- Grouping component information residing in a corporate database into component libraries to eliminate unnecessary library redundancy.
- Controlling access into one or more corporate-approved databases containing component information that can then be shared by multiple designers. Component information stored in a database becomes a centralized resource and, if desired, can be separately managed by another group within the company dedicated to that task.
- Providing the ability to quickly search for parts in component libraries based on property values, receive a list of components that meet the specified criteria, and preview the symbol choices for those components. After locating a desired component, you can drag and drop a symbol for the component into an open schematic and automatically back-annotate the property values on the generic symbol with property information from the database.
- Providing reference information, such as data sheets or supplier profiles.
- Populating a schematic with generic components and matching the components to a list of unique corporate-approved parts. Using this approach minimizes problems with part availability and manufacturing later in the production cycle.
- Selecting a component in DxDesigner, loading the search criteria for the component into DxDataBook, and querying the database for matching component objects. You can then add property values to, or update property values on, an existing component already on a DxDesigner schematic with new information from the database.
- Reducing errors by manual editing of a schematic through verification. Select an entire schematic, a portion of a schematic, or a component in DxDesigner and verify that the components exist in the corporate database and the part number, value, footprint, and other components are all consistent.
- Detecting properties missing from components in a schematic.

Related Topics

- [Introduction to DxDataBook](#)
 - [DxDataBook Requirements](#)

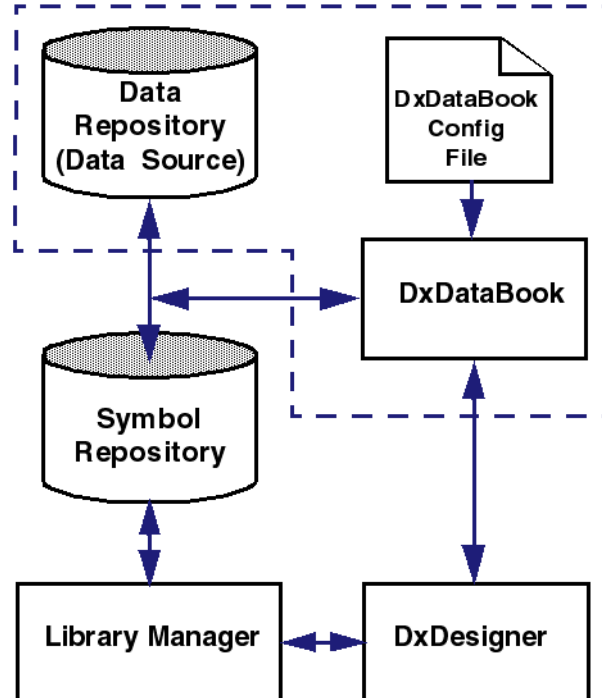
DxDataBook Requirements

DxDataBook requires the following:

- DxDesigner schematic capture application software.
- DxDataBook database browser software.
- ODBC-compliant relational database populated with corporate parts data and property information (see [Data Sources](#)).
- Configuration file linking DxDataBook to the relational database (see [Configuration File](#)).
- Library of graphic symbols to which the data from the relational database can be associated when placed on the schematic.

[Figure 1-1](#) shows the DxDataBook design environment and how it integrates with other applications. The region defined by dotted lines shows the scope of the topics contained in this manual.

Figure 1-1. DxDataBook Design Environment



Related Topics

- [Introduction to DxDataBook](#)
 - [DxDataBook Features](#)

Libraries

To use DxDataBook, you need to perform the following library-related tasks:

- Plan your library strategy.
- Build your symbol library.
- Construct a database containing component information.

Note

These tasks are usually performed by your administrator or librarian (see Chapter 2, “[Planning Your Library Strategy](#)”).

Related Topics

- [DxDataBook Requirements](#)
 - [Data Sources](#)
 - [Configuration File](#)

Data Sources

The ODBC-compliant data source that contains the DxDataBook component data can be located anywhere on your company’s enterprise network (for example, on the local system, a shared drive, or a server), or it can be located on the Internet. The DxDataBook client-server model allows multiple designers to use DxDataBook to connect to the same data source, usually through a mapped drive or other access method. Allowing many users to share the same component information ensures consistency throughout the entire design environment.

Supported ODBC-compliant data sources include (see Chapter 3, “[Choosing a Data Source](#)”):

- [Client/Server Database System](#)
- [DMS Connector Database](#)
- [DxDataBook Web Server](#)
- [Microsoft Access Database](#)
- [Microsoft Excel Spreadsheets](#)
- [MySQL on UNIX/Linux](#)
- [Oracle on UNIX/Linux](#)

The DxDataBook client-server model uses open standards to provide cross-platform functionality, allowing designers using either Windows or UNIX to use the same data source.

Note

Data sources that require a user-configured ODBC driver are not supported for UNIX.

Related Topics

- [DxDataBook Requirements](#)
 - [Libraries](#)
 - [Configuration File](#)

Configuration File

To use DxDataBook, you must have a valid a DxDataBook configuration file. The DxDataBook configuration file:

- Defines the DxDataBook library structure.
- Maps property names to field (column) names contained in the database tables, and defines how to view and process those properties in the DxDataBook user interface.
- Contains the necessary information required to connect to a particular data source.

Mentor Graphics software recognizes a file as a DxDataBook configuration file if the filename has a *.dbc* extension.

To create and then manage the contents of the DxDataBook configuration file, you use configuration management tools and wizards that are built into DxDataBook and accessed through the **Configure** menu (see Chapter 4, “[Editing a DxDataBook Configuration File](#)”). For larger groups or an enterprise, a designated database librarian may be responsible for making these changes.

The DxDataBook configuration file is only edited when adding or updating a component library (for example, adding or removing a column from a database table).

Related Topics

- [DxDataBook Requirements](#)
 - [Libraries](#)
 - [Data Sources](#)

Chapter 2

Planning Your Library Strategy

This topic describes how to plan library management, build your symbol library, and construct a database containing component information. These topics are of interest to an administrator or librarian who must plan a library strategy, create or modify a symbol library, and create a database and populate it with component information.

The following steps are required before you can use DxDataBook with DxDesigner:

1. Plan your library strategy.
2. Connect to the component data source (see Chapter 3, “[Choosing a Data Source](#)”).
3. Create a DxDataBook configuration file (see Chapter 4, “[Defining Data Sources, Libraries, and Symbols in the Configuration File](#)”).

Related Topics

- [DxDataBook Information Gathering Process](#)
- [DxDataBook Component Data Structure](#)
- [DxDataBook Library Structure](#)
- [DxDataBook Database Tables](#)

DxDataBook Information Gathering Process

You first need to gather information to help you create symbols and component data, and then organize the information in a way that is useful to designers and satisfies requirements for tools in the design process.

To create a library, it is important to have an understanding of your design environment and processes, as well as the component information that is required by these processes. The more designers share and rely on this information, the more time should be spent planning the library structure and data content that you want to manage through DxDataBook. Poorly constructed or validated component data that has proliferated throughout multiple designs can be problematic and costly to fix later in the design cycle.

When gathering information, it is important to keep good notes that can be used later to create your library structure, property list, and component database.

Here are some important questions to consider for the information gathering process:

- How many designers use DxDataBook to access component and library data?
- Which applications does the library data need to support? Is there manufacturing-specific information required by downstream applications? Is there non-EDA data (such as cost, supplier name, lead time, and approval check) that must be annotated to a design as part of the component data?
- Are designers permitted to add parts to the library, or is there a formal request process and published corporate library standards?
- What is the process to qualify a part for use in a design, purchasing the part, and placing a software representation of the part into the official library of released parts (assuming there is one)? Are there layers of approval before a part can be used in a design?
- Are designers allowed to use unapproved parts in a design (for example, parts downloaded from an online source or user-defined parts based on independent research) and then later replace them with an approved version? Are unapproved and approved parts cached in separate locations on the file system?
- Is there one global list of approved parts within the company or several lists of approved parts on a project-by-project basis?
- Is there a defined company part numbering system for components and does that system yield a unique identifier for each component? Who is responsible for assigning a company part number to a component?
- Are there other systems in the company that might require access to (or are a source of) component data? For example, is there a system to manage corporate Manufacturing Resource Planning (MRP) and component data which needs to be compatible with component information contained in the library database and displayed in DxDataBook?

Related Topics

- [DxDataBook Library Structure](#)
- [DxDataBook Database Tables](#)
- [DxDataBook Component Data Structure](#)

DxDataBook Component Data Structure

To implement a library strategy and construct a database, you need to understand the data required for a “complete” component. The design process that you want the libraries to satisfy, and the applications the libraries need to support, dictate these requirements. Different component categories, such as TTL and discrete parts, may have different requirements and therefore need properties with values that satisfy those requirements.

For example, if your design process consists of design entry using DxDesigner, simulation using an HDL simulator, and layout using Expedition, then the component data model includes

DxDesigner symbols, VHDL models, and Allegro shapes as component representations. All of the necessary parametric data (such as thermal, electrical, and cost component information) or properties must exist to satisfy any simulation parameters that must pass to the simulation and all of the properties required to forward annotate. A component is not complete unless it has all the necessary data and properties to support these applications and processes.

Note

The properties required by specific applications is beyond the scope of this manual. For specific tool requirements, consult the manuals for those applications.

If your goal is to also create a Bill of Materials (BOM) that itemizes all the components on your schematic by corporate part number, then the company part number data for each component is required. Within the database connected to DxDataBook, each component must have a unique identifier (this identifier is usually a corporate part number).

You can often use the unique identifier to index part information in MRP and component databases. Corporate part numbers are generally unique and can be overloaded or aliased to other properties, such as Device, minimizing the amount of parametric data required.

Note

The Expedition and Netlist flows use different properties.

You can also accommodate variations in the design process by including the component representations and parametric information required by the additional steps.

The following methods can be used to organize component data:

- [Property-Based Component Data](#)
- [Shared Symbol Component Data](#)

Related Topics

- [DxDataBook Information Gathering Process](#)

Shared Symbol Component Data

For large libraries, use the shared symbol component data method for storing property data (for example, symbols, models, and footprints) is to separate some of the component-specific property data from the symbol and store it in a different location, such as a database. This allows different components to share a common symbol, with information from the database being annotated to the symbol when the component is selected and placed on a schematic.

[Figure 2-1](#) shows a common inverter symbol that is shared between multiple 74xx04 components.

Figure 2-1. File-Based (Shared) xx04 Symbol



Using a 10,000 resistor library as an example, you could create the library using only one DxDesigner resistor symbol that is shared between all components. The symbol library has only a single resistor symbol, while a separate database holds all of the properties that make each resistor unique, and they can be applied when the component is placed on a schematic.

Storing data using a library of shared symbols and a separate database for component-specific properties has additional benefits beyond reducing the time to develop and maintain symbols:

- Enhanced component searching and instantiation capabilities.

Using DxDataBook to separate and view component data, you can use property or design criteria stored in the property database to narrow the search for a specific part, and then instantiate the selected part into DxDesigner.

For example, you can use a 110 ohm, 0.25 watt resistor to locate other resistors that meet your pre-specified conditions from the 10,000 resistors available in the library. You can then scroll through the list of resistors to pick the one to instantiate, or specify additional search conditions to further narrow the list of acceptable resistors. You can also instantiate a generic resistor and then apply component-specific properties to make it unique.

- Ability to leverage existing component data.

At your company, MRP and component databases might already contain much of the information needed for the design process, indexed by corporate part numbers. You can access this data directly with DxDataBook to provide up-to-date component information to designers. MRP databases typically contain part approval status and other data, such as cost, that you can make available to designers to help them make better component selections.

Related Topics

- [DxDataBook Component Data Structure](#)
 - [Property-Based Component Data](#)

Property-Based Component Data

The most common method used to store component-specific property data (such as Cost or PKG_TYPE) is adding the data to the DxDesigner symbol as property values, where each specific component has its own dedicated symbol. This method is acceptable, yet inefficient because it is difficult to create and maintain the stored data.

For example, if a library contains 10,000 resistors, and each resistor has the same graphical symbol, only the resistor value (or some other property) changes from one symbol to the next. A property-based configuration would be inefficient and require 10,000 unique symbols, one for each resistor. A high level of effort is then required to develop the library, add a new property to each library component, and assign meaningful names to each symbol.



Tip: Mentor recommends that you use the shared symbol component data method to store property data (see [Shared Symbol Component Data](#)).

Related Topics

- [DxDataBook Component Data Structure](#)

DxDataBook Library Structure

Once you have gathered information about your design environment, you need to determine the following:

- The number of databases that need to be created and connected to DxDataBook (see [Single and Multiple Databases](#)).
- A library structure within each database (see [Symbol Library Categories](#)).
- The type of property information to track and store (see [Property Tracking](#)).

Your library structure determines how tables are created in the database and populated with component information, and how libraries are defined in the DxDataBook configuration file.

Related Topics

- [DxDataBook Information Gathering Process](#)
- [DxDataBook Component Data Structure](#)
- [DxDataBook Database Tables](#)

Single and Multiple Databases

DxDatabook allows you to share information stored in a common database between many designers and present the same information in different ways.

The recommended method for information sharing is maintaining multiple databases (minimum of two databases): One database for part information “in development” and the other database for “released” part information. The database with parts in development has the same library and table structure as the database with released parts. The key difference between the databases is that the in development parts database remains in a restricted area that cannot be accessed by designers because the part data might be incomplete and not yet fully tested. As parts are developed, tested, and approved, they are then moved to the released parts database, which has general access.

You can share information between separate databases, although more administrative resources are required. Maintaining a similar structure between databases (that is, a similar set of tables within the database with common columns) allows data from one database to be exported and imported into another. In more complex data models, certain properties between databases can be linked and it is possible to create views where tables in one database have read-only access to information contained in another database. If multiple databases are exact copies of one another, it is possible to create and maintain a single master DxDataBook configuration file that can be connected to each copy.

Note

Sharing content between multiple databases is beyond the scope of this manual. Consult your database documentation for more information.

A single database can also be used to serve the entire designer community, yet this may not be the most practical method. For example, if each engineering project manages its own list of qualified and approved parts, or requires widely-varying lists of properties because of applications or processes specific to that project, then each engineering project may require its own dedicated database.

Related Topics

- [DxDataBook Library Structure](#)
 - [Symbol Library Categories](#)
 - [Property Tracking](#)

Symbol Library Categories

In DxDataBook, component library categories should be fairly broad because components can be further subdivided using properties located inside the library. For example, you have a library called “Resistors.” To further subdivide resistor components within the library, and to make the library searchable, you create a “Type” property with values of “Fixed,” “Network,” and “Variable.” This scheme allows designers to search the “Resistors” library in DxDataBook for a specific type of resistor.

Although DxDataBook does not restrict the categories or names you can choose for your libraries, and does not require library names to match the names of corresponding tables within the database, the following guidelines will help you improve database performance and reduce maintenance:

- Use a single table or a group of smaller, related tables for each library (library structure dependent).
- Choose table names that have intuitive relationships to library names.

For example, if you plan to have libraries named “Resistors,” “Inductors,” and “Capacitors,” you should also have tables with the same names in your database, rather than use a single table named “Passives” that corresponds to multiple libraries.

- Join tables within the database that are associated with a single library.

For example, you can join the “Fixed,” “Network,” and “Variable” tables that are associated with the “Resistors” library.

The following are examples of DxDataBook library categories:

- | | | | |
|-------------------------|--------------|------------------|------------------|
| • Analog IC | • Fuses | • Mechanical | • Power Supplies |
| • Capacitors | • Glue Logic | • Memory | • Resistors |
| • Connectors | • Hybrids | • Microprocessor | • Switches |
| • Digital (other types) | • Inductors | • Optical | • Transistors |
| • Diode | • Lamps | • PLD | |

If you are performing a hierarchical design, you might also have library categories for modules that exist above the component level, such as design and clock modules.

Related Topics

- [DxDataBook Library Structure](#)
 - [Single and Multiple Databases](#)
 - [Property Tracking](#)

Property Tracking

Once you determine the data that needs to be managed and placed on the schematic design, you can decide on the property information to track and the location for this information (that is, on the symbol or in the database).

Note



See [DxDesigner Properties Glossary](#) for more information about Expedition properties.

Related Topics

- [DxDataBook Library Structure](#)
 - [Single and Multiple Databases](#)
 - [Symbol Library Categories](#)
 - [Property Categories](#)

Property Categories

Property information can be categorized as follows:

- Properties with a value fixed on the symbol.
For example, a CLASS property on the symbol with a value fixed to CMOS. Fixed properties that reside on the symbol do not need to appear in the database.
- Properties with a value that comes from the database and has a placeholder value on the symbol.
For example, a PKG_TYPE property on the symbol with a question mark (?) or quotation marks (“ ”) placeholder value.
 - Properties with a value that comes from the database should have an associated column (field) in a database table. The column name in the database does not need to be the same as the property name on the symbol, since the mapping between the column and property names occurs in the DxDataBook configuration file.
For example, a PKG_TYPE property on a symbol that corresponds to a “Package Type” column in the database table.
- Properties that do not appear on the symbol or have placeholder values, but do have a value that you want annotated to the symbol when you place it on the schematic. Properties that are annotated to the symbol without a corresponding placeholder value are usually placed at some predetermined origination point (by default, at the lower left corner of the symbol with coordinates of 0,0).
- Properties that appear as columns in the database, but are not annotated to the symbol or otherwise transferred to a design.
For example, a database with component information can have several columns noting who approved the component at various steps in the information development process. It is not necessary to annotate this information to the schematic.

- Columns that appear in the database, but are not annotated to the schematic can be useful as search criteria, even though the information is not transferred to a design.

For example, a “Description” column containing certain keywords that a designer can use in a search to return a list of matching components.



Tip: Creating a column in the database to hold reference information (such as a link to a datasheet or the URL of a supplier profile) that can be searched in DxDataBook can be helpful when performing parts research.

In [Figure 2-2](#), an example of different inverter symbols with property information is shown:

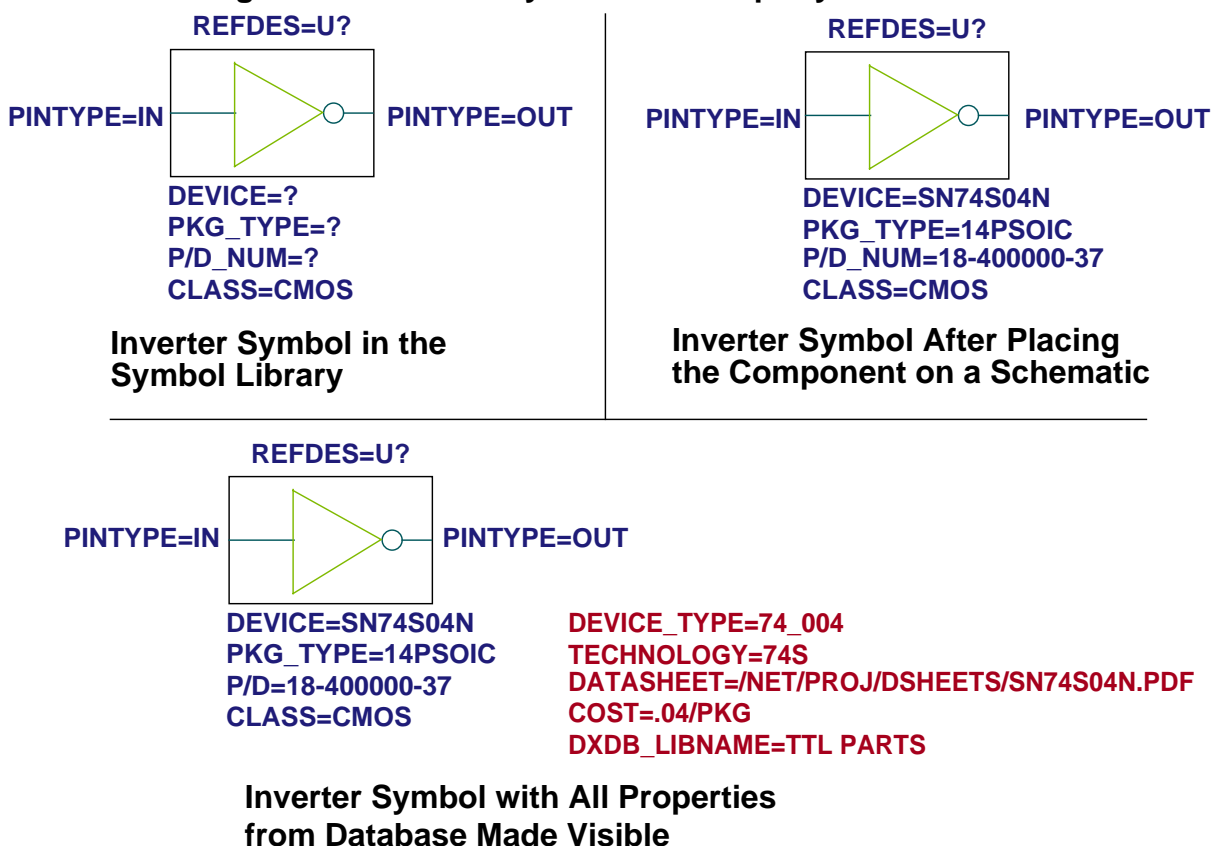
- The inverter component (placed symbol) with properties marked “visible” showing.
- The inverter symbol in DxDesigner after placing an inverter component on the schematic (with only visible properties showing).
- The inverter component showing annotated information from the database (properties that are normally invisible).

Note



There are several properties — shown in red text — with values that come from the component database, but do not have placeholder values on the symbol and generally have their visibility set to “invisible” (see [Setting Schematic Component Property Visibility](#)).

Figure 2-2. Inverter Symbol with Property Information



The overlapping placement of properties is not important because the properties (shown in red text) are usually invisible and do not appear when viewing a schematic, but the properties are still reported in the DxDesigner Property Editor window when selecting a component in the design.

Related Topics

- [Property Tracking](#)
- [Property and Database Information](#)

Property and Database Information

Before constructing or modifying symbols and creating a database, it is sometimes useful to summarize the properties and database information to track in a table (see [Table 2-1](#)).

The resistor library properties table contains columns with information such as approvals, the supplier name, and the Databook name. This information is never annotated to the schematic when placing the component, but is visible in the DxDataBook window and can be used as search criteria.

In some cases, the property name on a symbol or the property name that appears in DxDataBook can be different from the column name in the database table. Visibility of the property value is usually controlled by the placeholder property on the symbol (see [Setting Schematic Component Property Visibility](#)).

Table 2-1. Resistor Library Properties Table

Property Name	Column Name in Database	On Symbol?	Fixed or Placeholder Value on Symbol	Place on Schematic?	Example Value
	Req Approval	No	N/A	No	<initials>
	Test Approval	No	N/A	No	<initials>
	Final Approval	No	N/A	No	<initials>
	Databook	No	N/A	No	Sinclair Passive, 1996, Vol I
	Preferred Supplier	No	N/A	No	Yes
	Symbol	No	N/A	No	discretes:res
	Supplier Name	No	N/A	No	Ed's house of parts
Description	Description	No	N/A	Yes, invisible	3.9 Ohm, 5% Tol, 15v
DEVICE	Device	Yes	Placeholder	Yes, visible	R
DXDB_LIBNAME ₁	N/A	No	N/A	Yes, invisible	Resistors
PKG_TYPE	Footprint	Yes	Placeholder	Yes, visible	R1206
P/D_NUM	Part Number (primary key)	Yes	Placeholder	Yes, visible	13-2766-03
Prefix	N/A (symbol only)	Yes	Fixed	Yes, invisible	R
REFDES	N/A (symbol only)	Yes	Fixed	Yes, visible	R?
Tolerance	Tol	No	N/A	Yes, invisible	5
Value	Resistance	Yes	Placeholder	Yes, visible	3.9
VOLTAGE	Voltage	No	N/A	Yes, invisible	15

1. DXDB_LIBNAME is a system-supplied property.

Related Topics

[Property Tracking](#)

[Property Categories](#)

DxDataBook Database Tables

This topic describes methods you can use to:

- Create data sources for DxDataBook database tables in your library structure (see [Data Source Creation](#)).
- Construct database table columns (see [Table Column Constructs](#)).
- Add number and string data to your database tables (see [Number and String Data](#)).

Related Topics

- [DxDataBook Information Gathering Process](#)
- [DxDataBook Component Data Structure](#)
- [DxDataBook Library Structure](#)

Data Source Creation

When creating a data source to use with DxDataBook, you can use one of the following methods:

- Create a data source outline (that is, a data source that has structure, but no data), connect DxDataBook to the data source, build a DxDataBook configuration file, and populate the database with component information.
- Create a data source outline that is partially populated with data, connect DxDataBook to the data source, build a DxDataBook configuration file, and fully populate the database with information later.

For example, a database might be partially populated with values as the result of a symbol extraction, with additional columns such as “Part Number” and “Description” added, but do not contain values.

- Construct and populate the data source, connect the data source to DxDataBook, and build the DxDataBook configuration file only after all of the data is present. Modifications to the database tables are only made as new parts are needed.

Related Topics

- [DxDataBook Database Tables](#)
 - [Table Column Constructs](#)
 - [Number and String Data](#)

Table Column Constructs

The table columns and names can vary, but each table with component information used by DxDataBook should at a minimum contain:

- A column that serves as a unique identifier, or primary key, for rows in the table. For component information, this unique identifier is usually the component part number.
- A column containing the name of the DxDesigner symbol that resides in the symbol library and is associated with a part. When a component (for example, a resistor or capacitor) has multiple variations of a base symbol, as denoted by a numeric extension after the period (.), the “Symbol” column value should only contain the base symbol name and not the extension.

For example, a capacitor uses a *cap.1* symbol to specify horizontal rotation and *cap.2* to specify vertical rotation. In this example, *cap* is the base symbol name and it should appear as “cap” in the “Symbol” column.

If there is more than one base symbol name associated with a particular component, specify the base symbol names in a comma separated list.

For example:

```
dl, dled, dled_bright
```



Tip: You can also use symbols associated with parts from the part data set in the Central Library (avoids the need to populate parametric database tables with a “Symbol” column for associated part numbers). To access the symbol data, right-click in the DxDataBook window and select **Configure > Edit Configuration > Libraries > Use symbol data from Central Library**.

Notes

- You can eliminate the “Symbol” column when all components in a library use the same symbol and instead globally associate one particular symbol with a library through the DxDataBook configuration file.
- Do not use wildcards when specifying symbol names in the database table because it can lead to unexpected matches as the symbol search engine explores all the graphic libraries specified for a project.

Related Topics

- [DxDataBook Database Tables](#)
 - [Data Source Creation](#)
 - [Number and String Data](#)

Number and String Data

Columns that contain number data (such as, resistance or capacitance values) should only use numeric instead of string format to specify the data type. For example, a capacitance value of 10e-12 for a 10 pF capacitor. This format allows the designer to use DxDataBook to create search conditions that perform greater than (>), less than (<), and range searches on numeric values and also order the search results in ascending or descending order (see Chapter 5, “[Creating Search Queries](#)”).

For numeric values, you can control how magnifiers and units are displayed in DxDataBook through the configuration file (see Chapter 4, “[Editing Library Properties](#)”).

For example:

- The value of a capacitor can exist in the database table as 10e-12, but have a pico magnifier and unit applied such that the capacitance value appears in a DxDataBook search window as 10 pF.
- A resistor with a value of 1500000 is located in the database, but is displayed as 1.5 MEG in DxDataBook.

Related Topics

- [DxDataBook Database Tables](#)
 - [Data Source Creation](#)
 - [Table Column Constructs](#)

Chapter 3

Choosing a Data Source

This topic describes DxDataBook Open Database Connectivity (ODBC) compliant data sources and how to make these data sources compatible with your system. With this information, you can plan a library strategy, create or modify a symbol library, create a database and populate it with component information, and configure DxDataBook to communicate with that database.

Note



The DxDataBook Configuration Wizard can be used to create one or more DxDesigner database configurations, which specify how to connect to the data source and organize tabular data in the database into libraries for display in DxDataBook (see Chapter 4, “[Defining Data Sources, Libraries, and Symbols in the Configuration File](#)”).

Related Topics

- [DxDataBook: Supported Databases and Operating Systems](#)
- [DxDataBook Data Sources](#)

DxDataBook: Supported Databases and Operating Systems

[Table 3-1](#) lists all of the database formats and operating systems supported by DxDataBook.

Table 3-1. Supported Database Formats and Operating Systems

Database Formats	Operating Systems			
	Windows XP Windows 2000	Solaris v8.x Solaris v9.x Solaris v10.x	HP-UX v11.0 HP-UX v11.i	Linux v3 Update 4
CSV	Read-only	Read-only; also requires DxWebPack	Read-only; also requires DxWebPack	Read-only; also requires DxWebPack
Microsoft Access	Full support	Read-only; also requires DxWebPack	Read-only; also requires DxWebPack	Read-only; also requires DxWebPack

Table 3-1. Supported Database Formats and Operating Systems (cont.)

Database Formats	Operating Systems			
	Read-only	Read-only; also requires DxWebPack	Read-only; also requires DxWebPack	Read-only; also requires DxWebPack
Microsoft Excel	Read-only	Read-only; also requires DxWebPack	Read-only; also requires DxWebPack	Read-only; also requires DxWebPack
Microsoft SQL Server (v6.5, v7.0, v2000)	Full support	Read-only; also requires DxWebPack	Read-only; also requires DxWebPack	Read-only; also requires DxWebPack
MySQL (v4.x)	Full support	Read-only; also requires DxWebPack	Read-only; also requires DxWebPack	Read-only; also requires DxWebPack
Oracle (v9.x)	Full support	Full support	Full support	Full support
Text	Read-only	Read-only; also requires DxWebPack	Read-only; also requires DxWebPack	Read-only; also requires DxWebPack
Worldwide Web Server	Read-only	Read-only	Read-only	Read-only

Related Topics

- [DxDataBook Data Sources](#)
 - [Oracle on UNIX/Linux](#)
 - [MySQL on UNIX/Linux](#)

Oracle on UNIX/Linux

To use DxDataBook with an Oracle database on a UNIX or Linux operating system, observe the following requirements and guidelines:

- Make sure that you have defined the ORACLE_HOME environment variable as the Oracle installation location.
- If ORACLE_HOME is not defined, DxDataBook checks the following environment variables for the ORA_NLS directory, which contains files required to run Oracle with DxDataBook:
 - **TNS_ADMIN** set to \$SDD_HOME/common/<platform>/ORA_NLS
 - **ORA_NLS33** set to \$SDD_HOME/common/<platform>/ORA_NLS
 - **ORA_TZFILE** set to \$SDD_HOME/common/<platform>/ORA_NLS/timezone.dat

where <platform> represents the name of the platform-specific directory in your installation location.

- Make sure that you have set read and write access to all files in the ORA_NLS directory:
 - ***.nlb** — Files with messages.
 - **Tnsnames.ora** — File contains configuration for connections (TNS_ADMIN environment variable should point to file location).
 - **Timezone.dat** — ORA_TZFILE environment variable points to file location.

Related Topics

- [DxDataBook: Supported Databases and Operating Systems](#)
 - [MySQL on UNIX/Linux](#)

MySQL on UNIX/Linux

This topic describes how to connect DxDataBook to a MySQL database on a UNIX or Linux operating system.

Procedure

1. Download MySQL from the MySQL software website.
2. Download Qt from the Nokia Qt website.
3. Download and install the dedicated ODBC driver from the MySQL Developer Zone website.
4. Configure and compile MySQL sources using the following parameters:

```
./configure --without-server --without-extra-tools --without-bench  
--with-extra-charsets=complex --enable-thread-safe-client --enable-  
local-infile --with-innodb --disable-shared --without-debug --  
without-docs --without-man  
  
make
```

Note



The Linux GCC 3.2 or Solaris Sun Studio 8 compiler (or newer) is required.

5. Edit the following files:

<path_to_Qt_sources>/src/sql/qsqldatabase.cpp

and

<path_to_Qt_sources>/plugins/src/sqldrivers/mysql/mysql.pro

- a. Edit the *qsqldatabase.cpp* file, line 682:

```
d->driver->setLastError( QSqlError( "Driver not loaded","Driver not loaded" ) );
```

Changes to

```
d->driver->setLastError( QSqlError( "Driver not loaded", type ) );
```

- b. Edit the *mysql.pro* file, lines 11-17:

```
unix {  
    OBJECTS_DIR = .obj  
  
    !contains( LIBS, .*mysql.* ) {  
  
        LIBS *= -lmysqlclient  
  
    }  
}
```

Changes to

```
unix {  
    OBJECTS_DIR = .obj  
  
    !contains( LIBS, .*mysql.* ) {  
  
        LIBS *=  
<path_to_mysql_sources>/libmysql/.libs/libmysqlclient.a  
    }  
    INCLUDEPATH += <path_to_mysql_sources>/include  
}
```

6. Configure Qt by copying the code into a text editor and running the following script:

```
#!/bin/sh  
  
ARCH=`uname`  
SRCROOT=<path_to_Qt_sources>  
OBJROOT=${SRCROOT}/out/${ARCH}  
CFG_FLAGS="-disable-opengl -no-pch -no-tablet -no-nas-sound \  
-no-sm -no-xinerama -no-xrender -no-xft -no-xkb -no-xcursor  
-no-xrandr -stl \ -enable-xml -enable-sql"  
  
TYPEOBJ="-shared -plugin-sql-mysql"  
case ${ARCH} in  
    HP-UX)  
        PLATFORM=hpx-acc  
        MYSQL_INCLUDE=<path_to_MySQL_sources>/include  
        MYSQL_LIBS=<path_to_MySQL_sources>/libmysql/.libs  
        ;;  
    Linux)  
        PLATFORM=linux-g++  
        MYSQL_INCLUDE=<path_to_MySQL_sources>/include  
        MYSQL_LIBS=<path_to_MySQL_sources>/libmysql/.libs  
        ;;  
    *)  
        #solaris  
        PLATFORM=solaris-cc  
        MYSQL_INCLUDE=<path_to_MySQL_sources>/include  
        MYSQL_LIBS=<path_to_MySQL_sources>/libmysql/.libs  
    esac
```



```
;;
esac
```

7. Change paths where indicated and execute the script in the directory to which it was saved.

8. Set the libraries paths in the Qt source:

```
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:<path_to_Qt_sources>/lib
```

9. Configure and compile the Qt source:

```
./configure -platform ${PLATFORM} -fast -release ${TYPEOBJ} -prefix  
${OBJROOT} -bindir ${OBJROOT}/bin -libdir ${OBJROOT}/lib  
${CFG_FLAGS}
```

```
make
```

10. Copy the MySQL Qt drivers from

```
<path_to_Qt_sources>/plugins/sqldrivers/libqsqlmysql.so
```

to

```
$SDD_HOME/common/${SDD_PLATFORM}/lib/.sqldrivers/
```

Results

DxDataBook is connected to a MySQL database on a UNIX or Linux operating system.

Related Topics

- [DxDataBook: Supported Databases and Operating Systems](#)
 - [Oracle on UNIX/Linux](#)

DxDataBook Data Sources

DxDataBook is compatible with a variety of ODBC-compliant data sources, such as the DMS Connector, DxDataBook web server, and Microsoft Access databases. To configure DxDataBook so that it can communicate with these data sources:

- The database containing tabular component data must be located on your local system or in a network location accessible by all users. For example, if you are using a Microsoft Access database as a data source, the *.mdb* file must reside in a location that all users can access from their local system.
- An ODBC driver for the specified data source type must be located on your computer.

UNIX



Data sources that require a user-configured ODBC driver are not supported for UNIX.

Once DxDataBook is configured, you need to decide which ODBC-compliant data sources serve as the repository for your component data, and then setup the database files and an ODBC driver for each of the data sources.

DxDataBook supports the following data sources:

- [Client/Server Database System](#)
- [DMS Connector Database](#)
- [DxDataBook Web Server](#)
- [Microsoft Access Database](#)
- [Microsoft Excel Spreadsheets](#)

After you have created the database file and setup the ODBC driver for your data sources, use the DxDataBook Configuration Wizard to create a database configuration (*.dbc*) file that communicates with the data sources (see [Creating a DxDataBook Configuration File](#)).

Note



DxDataBook can also access Central Library (*.lmc*) files as data sources.

Related Topics

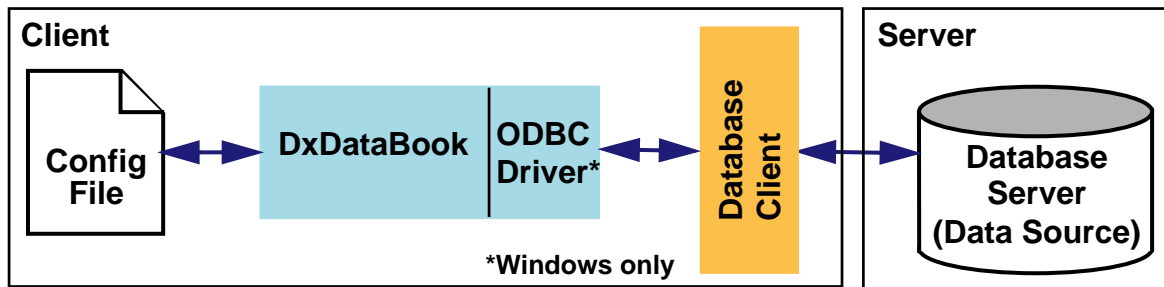
- [DxDataBook: Supported Databases and Operating Systems](#)

Client/Server Database System

For the client/server database system, you must setup ODBC drivers on each client computer that connects to the database system (for example, SQL Server, Oracle, Sybase, or Informix). The database network layer handles communication between the client and server.

[Figure 3-1](#) shows how DxDataBook is configured to connect to a client/server database.

Figure 3-1. Client/Server Database System



Note



DxDataBook does not support ODBC drivers for UNIX.

DxDataBook can be integrated with any available 32-bit ODBC driver and DxDataBook comes with the following drivers installed:

- Microsoft Access
- Microsoft Excel
- SQL Server
- Text driver

Note



DxDataBook does not support 16-bit ODBC drivers (upgrade to a 32-bit ODBC driver before using the driver with DxDataBook).

You can also use the following third-party ODBC drivers:

- Fox dBase
- Paradox
- Oracle



Tip: Contact your vendor for information about a specific ODBC driver and its installation.

Related Topics

- [DMS Connector Database](#)
- [DxDataBook Web Server](#)
- [Microsoft Access Database](#)

- [Microsoft Excel Spreadsheets](#)
 - [Configuring an Oracle Client-Server Database](#)

Configuring an Oracle Client-Server Database

The following topic shows you how to configure DxDataBook with an Oracle client/server database.

Procedure

1. Set the ORACLE_HOME environment variable to point to your Oracle installation directory.
2. Set the SHLIB_PATH environment variable to include \${ORACLE_HOME}/lib.
3. If the database you have chosen requires a username and password to access the data, set the DX_CONNECTSTRING environment variable.

This environment variable allows DxDataBook to automatically log into the database and contains the username and password syntax, or it can contain only placeholders (for security reasons).

4. Set the DX_CONNECTSTRING environment variable as follows:

```
setenv DX_CONNECTSTRING " ;DSN=%dsn;UID=%uid;PWD=%pwd; "
```

You can also substitute your username for *username*:

```
setenv DX_CONNECTSTRING " ;DSN=%dsn;UID=username;PWD=%pwd; "
```

The login routine sets the value automatically.

Results

The ORACLE_HOME, SHLIB_PATH, and DX_CONNECTSTRING environment variables have been set and DxDataBook is now configured to run with the Oracle client/server database.

Related Topics

- [DxDataBook Data Sources](#)
 - [Client/Server Database System](#)

DMS Connector Database

DMS™ Connector is a web application supported on all DxDesigner platforms that provides access to Data Management System (DMS) component information for DxDataBook. The DMS Connector application is a set of JavaServer Pages (JVP) that installs on a web server (for

example, Tomcat web server) and requires a Servlet container. You can use any JSP container to support DMS Connector.

Note



You can use DMS Connector to search the entire DMS database, yet the scope of DMS Connector in relation to DxDataBook is limited to component searches.

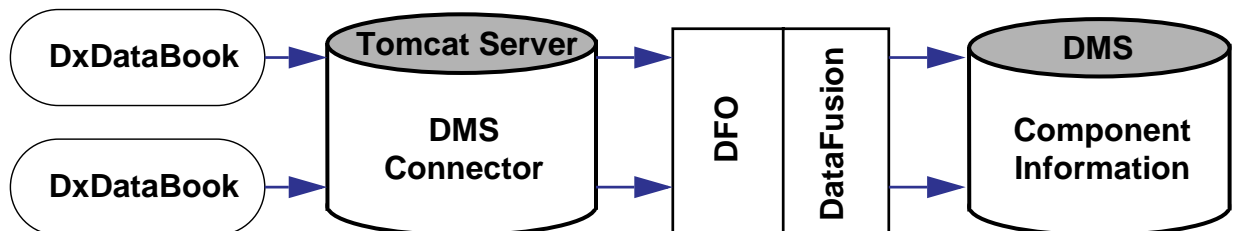
DxDataBook uses DMS Connector to interact with a DMS DataFusion™ server, which connects to a DMS Database. The DMS Connector option allows you to configure a DMS database as a DxDataBook data source.

The DMS Connector architecture consists of the following components:

- DxDataBook add-in to DxDesigner
- DMS Connector
- DataFusion Objects (DFO)
- DataFusion Server
- DMS Component Database

Figure 3-2 shows how these components interact with each other to provide a seamless component search and verification capability. DMS Connector acts as a bridge between DxDataBook and DataFusion (DMS).

Figure 3-2. DMS Connector, DxDataBook, DataFusion, and DMS Configuration



Related Topics

- [Client/Server Database System](#)
- [DxDataBook Web Server](#)
- [Microsoft Access Database](#)
- [Microsoft Excel Spreadsheets](#)

DMS Connector Requirements

In addition to installing DMS Connector, you also need to install the following components:

- DxDesigner
- DxDataBook
- DMS with DataFusion
- Java Runtime Environment (JRE 1.4.2 or newer) which can be downloaded from

<http://java.sun.com>

- Tomcat Application Server (v5.0 or newer) which can be downloaded from

<http://tomcat.apache.org/>

- Oracle with the DMS database

DMS Connector is packaged as a web archive (*.war*) file and is included in the DMS installation.

Note

Microsoft Internet Information Server (IIS) is not a JSP container and cannot be used as a server for DMS Connector.

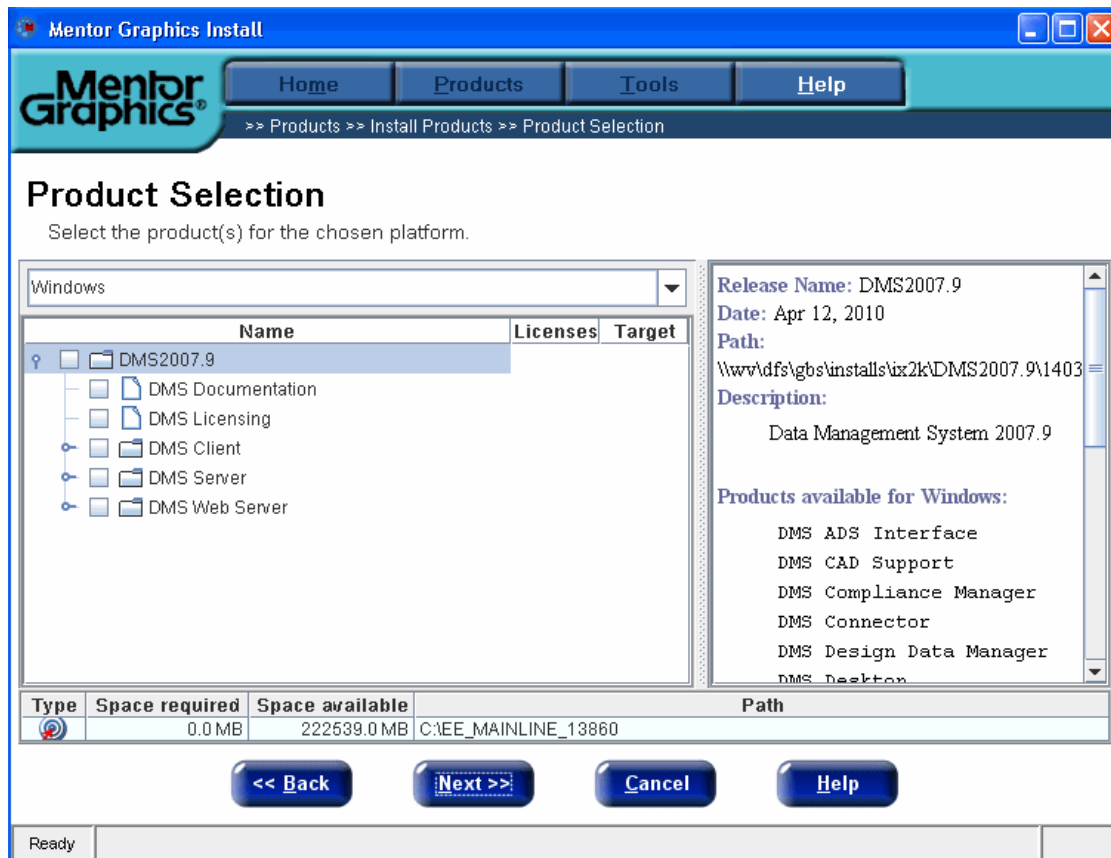
Related Topics

- [DMS Connector Database](#)
 - [DMS Connector Installation](#)
 - [Setting Up and Configuring the DMS Connector](#)
 - [Starting the DMS Connector Server](#)
 - [Running DMS Connector with DxDataBook](#)

DMS Connector Installation

[Figure 3-3](#) shows the Mentor Graphics Install window which is displayed during DMS installation and contains the DMS Connector option.

Figure 3-3. DMS Connector Installation



- Select the DMS Connector option and click Next.

A web archive DMSConnector (.war) file is copied into the \$DBEDIR/java directory of the DMS client.

After you have finished installing DMS, you need to setup and configure DMS Connector (see [Setting Up and Configuring the DMS Connector](#)).

Related Topics

- [DMS Connector Database](#)
 - [DMS Connector Requirements](#)
 - [Starting the DMS Connector Server](#)
 - [Running DMS Connector with DxDataBook](#)

Setting Up and Configuring the DMS Connector

The following topic describes how to setup and configure the DMS Connector.

Prerequisites

- Install DMS Connector supporting software (see [DMS Connector Requirements](#))
- Install DMS Connector (see [DMS Connector Installation](#))

Procedure

1. Copy the *DMSCConnector.war* file to your server location.

If you are using the Tomcat server:

- a. After installation, use the DMS Configurator post-installation program (program is displayed automatically once the installation process is completed).
- b. Select the Install DMS2DxDB Connector to Browser option.

The DMS Connector software is installed and the *DMSCConnector.war* file is copied from the \$DBEDIR/java directory to your <tomcat_home>/webapps directory.

If you are using a different web server:

- c. Close the DMS Configurator and copy your *DMSCConnector.war* file from the \$DBEDIR/java directory to a location in your server directory where the server can find the file.

Note



You may need to consult your server documentation to determine where to copy the file.

2. Deploy DMS Connector to the Tomcat server.
 - a. Installing DMS Connector provides a web archive (.war) file called *DMSCConnector.war*, which you need to place in the webapps directory of your Tomcat installation located at <tomcat_home>/webapps.

For example:

```
C:\Apache_Tomcat\jakarta-tomcat-4.1.12-LE-jdk14\webapps
```

Once Tomcat is running, the .war file is extracted and placed into a directory named *DMSCConnector*.

3. Start the Tomcat server.

Once the .war file is copied and the Tomcat server is started, DMS Connector can be configured.

4. Specify DataFusion connection parameters to DMS Connector.

Note



Once you have entered the information needed to connect to a DMS database, the **Connection**, **Characteristics**, **DxDataBook**, and **Select Production Library** tabs appear on the updated DMS Databook Connector page.

- a. On the DMS Databook Connector page, click Connection to display the DMS Connector parameters (see [Figure 3-4](#)).

Figure 3-4. DataFusion Connection Parameters

The screenshot shows the 'DMS Databook Connector' interface with the 'Connection' tab selected. The interface includes the Mentor Graphics logo and a status bar indicating 'Production Library: no production library selected'. Below the tabs, there are links to 'Go to Taxonomy' and 'Go to DxDataBook Configurations Page'. The main form contains the following fields:

- IOR File**: A text input field with a placeholder example '(e.g. file:/c:/dms/bin/DataFusion.ior)'.
- Username**: A text input field.
- Password**: A text input field.
- Database**: A text input field with a placeholder example '(e.g. dmsDB/dmsDB@orc)'.
- Global query limit**: A text input field with the value '2000'.
- Login method**: A dropdown menu with 'Local' selected.
- DxDB Encoding**: A dropdown menu with 'Disabled (default)' selected.
- Save**: A button at the bottom of the form.

- b. Enter values for the following form fields:
 - **IORFile** — URL for the *DataFusion.ior* file which can be a local file system or an online location.

For example:
`http://www.compservice.com/df/DataFusion.ior`
 - **Username** — Name for DMS Connection users. All DxDataBook users must use this name to log into DMS.
 - **Password** — DMS password all users need to access DMS Connection.
 - **Database** — DMS database name of the form:
`<oracle_user_name>/<oracle_password>@<connection_string>`
 - **DxDB Encoding** — DxDataBook-compliant character encoding so information is correctly transferred from the server to DxDataBook.

Note



The DxDB Encoding option should only be used when the library or configuration names do not use standard printable ASCII characters.

- c. Click **Save**.

The information is saved in a property file at the following server location:

```
<tomcat_home>/webapps/DMSConnector/DMSConnector.properties
```

Sample output file:

```
#DMS Connector
#Thu Mar 10 05:53:27 EDT
Username=jdesigner
Password=noodle66
IORFile=file\:/C\:/Central/Projects/Mentor/dms/dms2005.vcd/2005/dms
/bin/df.ior
Database=s12005/s12005@orcl
```

5. Specify DataFusion component characteristics for the DMS database.
 - a. On the DMS Databook Connector page, select Characteristics to view the list of component characteristics for the DMS database.

If your librarian has already configured the DataBook tab sheet for a Toolbox object, you do not need to edit the component characteristics.
 - b. Select the appropriate check boxes to make the designated component characteristics visible in DxDataBook.
 - c. Click Save Configuration to export the configuration file (*.dbc* extension).
6. Specify DataFusion to configure DxDataBook to view datasheets within the DMS database.
 - On the DMS Databook Connector page, select DxDataBook to learn how to load a helper script so that you can use DxDataBook to pull datasheets residing in the DMS database that are associated with components.
7. Specify DataFusion production library restrictions.

A production library is a collection of components and is created as an object in the DMS database. The production library settings determine which components can be used for instantiation into a design.

For example, an administrator can choose which production library is appropriate for an engineering group working on a specific design (DxDataBook components are automatically filtered by the production library setting).

- a. On the DMS Databook Connector page, click Select Production Library to configure production library restrictions (see [Figure 3-5](#)).

Figure 3-5. DataFusion Production Library Restrictions

Please select¹ preferred production library to restrict executed queries.

Disable restriction ▼ Restrict

Selected

Disable restriction

Other available

Project_Vidar

Compliant_RoHS

Compliant_ITAR

Project_DDR

Compliant_WEEE

Special

Disable restriction

Production library for user: admin or no restriction has been specified by configuration.

varies please check DMS user configuration

- b. Select one of the following options from the menu:
 - **Selected** — Shows production library currently being used for filtering.
 - **Other available** — Shows list of approved production libraries in DMS that can be used as a filter.
 - **Special** — Displayed only when restrictions are not defined in DMS for users configured for DMS Connector (for example, restrictions such as a “Default Production Library” defined or a list of allowed production libraries).
 - **Default** — Displayed only if the Default Production Library characteristic has a value (not shown in [Figure 3-5](#)).

The Disable restriction option (default setting) ignores all production libraries and the DxDataBook query searches the entire DMS database.

- c. Click Restrict.

Example

1. Configure the Tomcat server to connect to port 8880.
2. Once Tomcat is started, use the following URL format to connect to the Tomcat webpage:

`http://localhost:8880`

3. To access and test the DMS Connector go to

`http://localhost:8880/DMSConnector`

and enter the following information:

- *Datafusion.ior* file.
- User name.
- Password.

- Case-sensitive connection (such as dms2005/dms2005@ORCL or DMS_ALIAS).

Note

Use the same entry for the connection shown in the *df_launcher.cfg* file for DataFusion.

These settings are saved to the following file:

C:\apache-tomcat-6.0.16\webapps\DMSCConnector\WEB-INF\DMSCConnector.properties

Here is a sample *DMSCConnector.properties* file:

```
#DMS Connector
#Mon Dec 07 10:29:43 PST 2009
Username=admin
IORFile=file:///C:/MentorGraphics/2007DMS/SDD_HOME/dms/bin/DataFusion.i
or
LoginMethod=local
Database=dms07/dms07@tuck10
Password=admin
CharsetEnabled=no
Charset=ISO-8859-1
CharacterEscaping=automatic
MaxComponentCount=2000
```

If all settings are correct and connection was successful, run DataFusion to connect to the *DataFusion.i* file. This launches a web browser that displays a DMS Connector Component Taxonomy list.

Results

DMS Connector is setup and configured, and the server is now accessible.

Related Topics

- [DMS Connector Database](#)
 - [Starting the DMS Connector Server](#)
 - [Running DMS Connector with DxDataBook](#)

Starting the DMS Connector Server

This topic shows you how to start the DMS Connector server.

Prerequisites

- Setup and configure the DMS Connector (see [Setting Up and Configuring the DMS Connector](#))

Procedure

1. In Windows, select **Start > Mentor Graphics SDD > DxUtilities > DMS Connector** to start DMS Connector.
2. Check to make sure the Tomcat server is working correctly by launching a web browser (for example, Internet Explorer) and entering the URL address that you used to connect to the Tomcat webpage.

For example:

`http://localhost:8880/DMSConnector`

If the server is working, the web browser displays a DMSConnector Component Taxonomy list.

3. Run DxDataBook and connect to the new Tomcat server.
4. Copy the URL from Step 2 and use it when connecting to DxDataBook.

Results

The DMS Connector server is started and can now be used with DxDataBook.

Related Topics

- [DMS Connector Database](#)
 - [DMS Connector Requirements](#)
 - [DMS Connector Installation](#)
 - [Running DMS Connector with DxDataBook](#)


Running DMS Connector with DxDataBook

This topic describes how to run the DMS Connector server with DxDataBook.

Prerequisites

- Start the DMS Connector server (see [Starting the DMS Connector Server](#))

Procedure

1. Launch DxDesigner.
2. In the DxDesigner window, click the DxDataBook button .
3. In the DxDataBook window, edit your configuration:
 - a. Right-click and select **Configure > New**.

- b. In the New Configuration dialog box, select the “I know the location of a DxDataBook Base Configuration File” or “I would like to use the DMS to set up DxDataBook” option.
- c. Enter the URL used to connect to the Tomcat web page.

For example:

```
http://localhost:8880/DMSConnector
```

- d. Click Connect.

If the URL you entered is valid, any DMS configurations at that location populate the DMS configurations table.

- e. Select a configuration from the list.
- f. Click **OK**.

Results

DxDataBook is connected to the DMS Connector server and you can now access the DxDataBook libraries.

Related Topics

- [DMS Connector Database](#)
 - [DMS Connector Requirements](#)
 - [DMS Connector Installation](#)
 - [Setting Up and Configuring the DMS Connector](#)

DxDataBook Web Server

The DxDataBook web server, which is part of the Design Exchange Web Pack, is a Microsoft Internet Information Server (IIS) extension.

Note



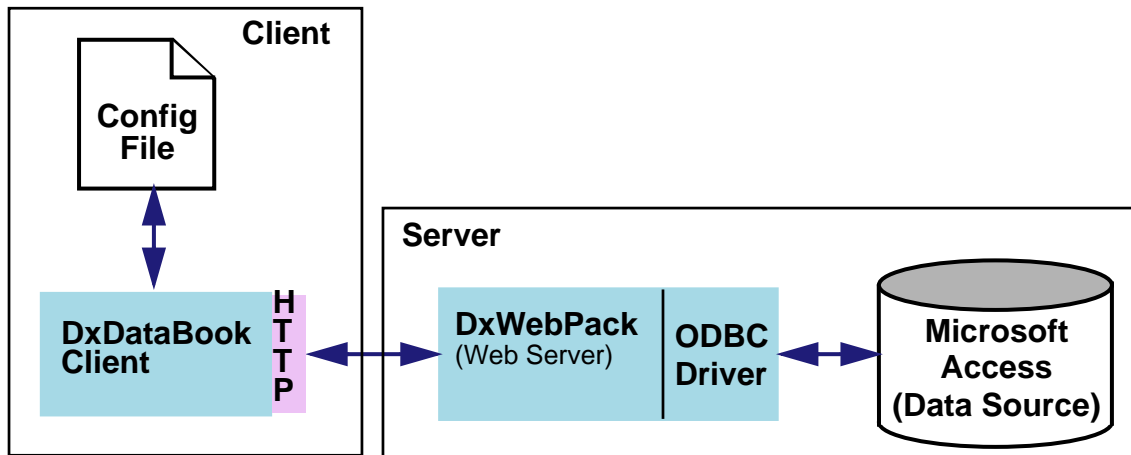
The DxDataBook web server only operates with the Windows version of DxDataBook.

The DxDataBook web server allows you to move the ODBC setup and configuration from your local machine to an existing web server system. The web server handles the ODBC connections to the database, so that individual clients do not need to have ODBC drivers installed before using DxDataBook. You only need to specify a URL in DxDataBook to start searching the web server for components. Your environment can be a mixture of direct ODBC and DxDataBook

web server access, allowing some data to be pulled directly using ODBC and other data to be pulled from the web server.

Figure 3-6 shows DxDataBook configured to connect to a Microsoft Access database using a web server.

Figure 3-6. DxDataBook Web Server



Note



DxDataBook web server also has a view-only HTML interface for access to component data through a web browser (symbols can only be previewed using this interface if they exist on the local system where the data is being viewed).

Once the DxDataBook web server is configured (see [Web Server Requirements](#)), and a DxDataBook configuration file is available as described in Chapter 4, “[Defining Data Sources, Libraries, and Symbols in the Configuration File](#),” type the following URL into your browser to access the data:

`http://computername.yourcompany.com/dx/dxdb.html`

The URL applies to the default installation of the DxDataBook web server and may have been modified for your configuration.

Related Topics

- [Client/Server Database System](#)
- [DMS Connector Database](#)
- [Microsoft Access Database](#)
- [Microsoft Excel Spreadsheets](#)

Web Server Requirements

This topic lists requirements for the web server interface on client and server systems.

Client System

The client system requires DxDesigner 2004 SPac1 or newer software and uses one of the following operating systems:

- Windows 2000 with Service Pack 4 (or newer)
- Windows XP with Service Pack 1a (or newer)
- Solaris 8 or 9
- HP-UX 11.00 or 11.i
- Linux EWS 3.0

Server System

The server system requires the following software:

- Windows 2000 with Service Pack 4 or newer (server or workstation), or Windows XP Service Pack 1a (or newer).
- Internet Information Server (IIS) 4.0 or 5.0.

The server must also have a valid DxDataBook server license.

Related Topics

- [DxDataBook Web Server](#)
 - [Installing Web Server Software](#)
 - [Setting the ODBC-to-Web Server Connection](#)
 - [Defining Data Sources with DxDBConfig](#)

Installing Web Server Software

You first need to install the Microsoft Internet Information Server (IIS) web server and Design Exchange Web Pack software before configuring the DxDataBook web server.

Note



See Windows help for the IIS installation procedure (Windows installation CD-ROM is required).

Prerequisites

- Verify that your system meets the DxDataBook web server requirements (see [Web Server Requirements](#))

Procedure

1. Insert the ISD CD into your disk drive and perform the procedures described in the “Managing Mentor Graphics PCB Software” document to launch the Mentor Graphics installation program.
2. In the Product Selection screen, navigate to the DxUtilities folder and select DxWebPack.
3. Follow the instructions and install the DxWebPack software.

Results

The DxDataBook web server software is installed and now you can setup the ODBC-to-DxDataBook web server connection.

Related Topics

- [DxDataBook Web Server](#)
 - [Setting the ODBC-to-Web Server Connection](#)
 - [Defining Data Sources with DxDBConfig](#)

Setting the ODBC-to-Web Server Connection

To use the DXDataBook web server to access the databases, you need to setup the ODBC-to-web server connection. In this configuration, the web server handles ODBC connections.

Note



Individual client servers do not need ODBC drivers installed before using DxDataBook.

Prerequisites

- Install the DxDataBook web server software (see [Installing Web Server Software](#))

Procedure

1. In Windows, select **Start > Control Panel** and double-click Administrative Tools (see [Figure 3-8](#)).
2. In the Administrative Tools window, double-click Data Sources (ODBC).

3. In the ODBC Data Source Administrator dialog box, make sure the **User DSN** tab is selected.

Check to make sure your Data Source Name (DSN) is not already used for an existing User DSN.



Tip: If your User DSN is already used, either remove the User DSN or change the name of the DSN being added.

4. Click **System DSN** (tab) > **Add**.
5. In the Create New Data Source dialog box, select the type of database connection you want to create.
6. Click Finish.
7. Add your DSN and database information.
8. Click **OK**.
9. Run the Configure Databook Server utility that is included with the DxWebPack software.

This utility adds the system DSN through an alias by which the server is identified by clients.

Results

The web server is now included as an ODBC data source and added to the local configuration, making it available to use as a data source for DxDataBook.

Related Topics

- [DxDataBook Web Server](#)
 - [Web Server Requirements](#)
 - [Defining Data Sources with DxDBConfig](#)

Defining Data Sources with DxDBConfig

DxDBConfig allows you to perform the following tasks:

- Set up the DxDataBook web server.
- Define data sources for DxDataBook clients.
- Specify the database configuration file for your web browser when running the web extension to DxDataBook.

Prerequisites

- Install the DxDataBook web server (see [Installing Web Server Software](#))

Procedure

1. Run the DxDBConfig program and add database aliases.
2. Restart the IIS server.

Note



See the DxDBConfig README file for more information.

Results

You have defined the DxDataBook data sources using DxDBConfig which allows you to specify client access and browser settings.

Related Topics

- [DxDataBook Web Server](#)
 - [Web Server Requirements](#)
 - [Setting the ODBC-to-Web Server Connection](#)
 - [Specifying the Client Access Settings](#)
 - [Specifying the Browser Settings](#)

Specifying the Client Access Settings

The **DxDBConfig** tab allows you to add or remove database aliases. Database aliases are used to define the set of data source names visible to the DxDataBook client.

- To add or remove an alias, fill-in the necessary information and make the appropriate selections for the following options:
 - **Database Alias** — Type an alias for the data source listed in ODBC Data Source Name into the Database Alias text field.
 - **ODBC Data Source Name** — Select the ODBC data source to add an alias for a data source.
 - **ODBC Connect Name** — Select a data source in ODBC Data Source Name to add the ODBC connect name.

Related Topics

- [Defining Data Sources with DxDBConfig](#)

- [Specifying the Browser Settings](#)

Specifying the Browser Settings

The **DxDBConfig** tab allows you to specify the database configuration file that you want to use in the DxDataBook web extension.

- **Configuration File (*.dbc)** — Enter the name of the DxDataBook database configuration (.dbc) file that you want to use when viewing data in a web browser with the DxDataBook web extension.

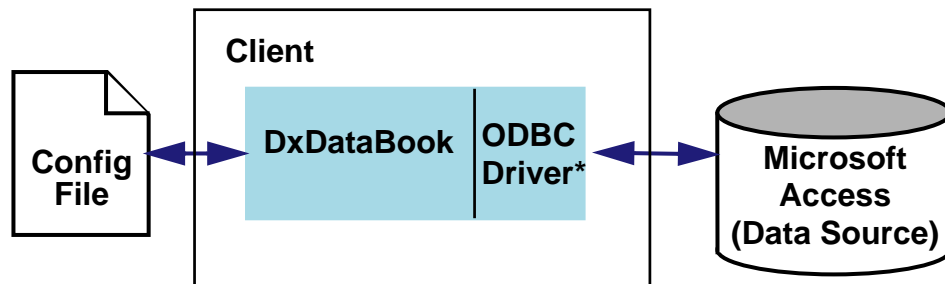
Related Topics

- [Defining Data Sources with DxDBConfig](#)
 - [Specifying the Client Access Settings](#)

Microsoft Access Database

This topic describes how to create a Microsoft Access data source on a local system (see [Figure 3-7](#)). The data source must be installed and accessible before you can create a DxDataBook configuration file that uses the Microsoft Access database as a source for library data.

Figure 3-7. DxDataBook to Microsoft Access Configuration



Prerequisites

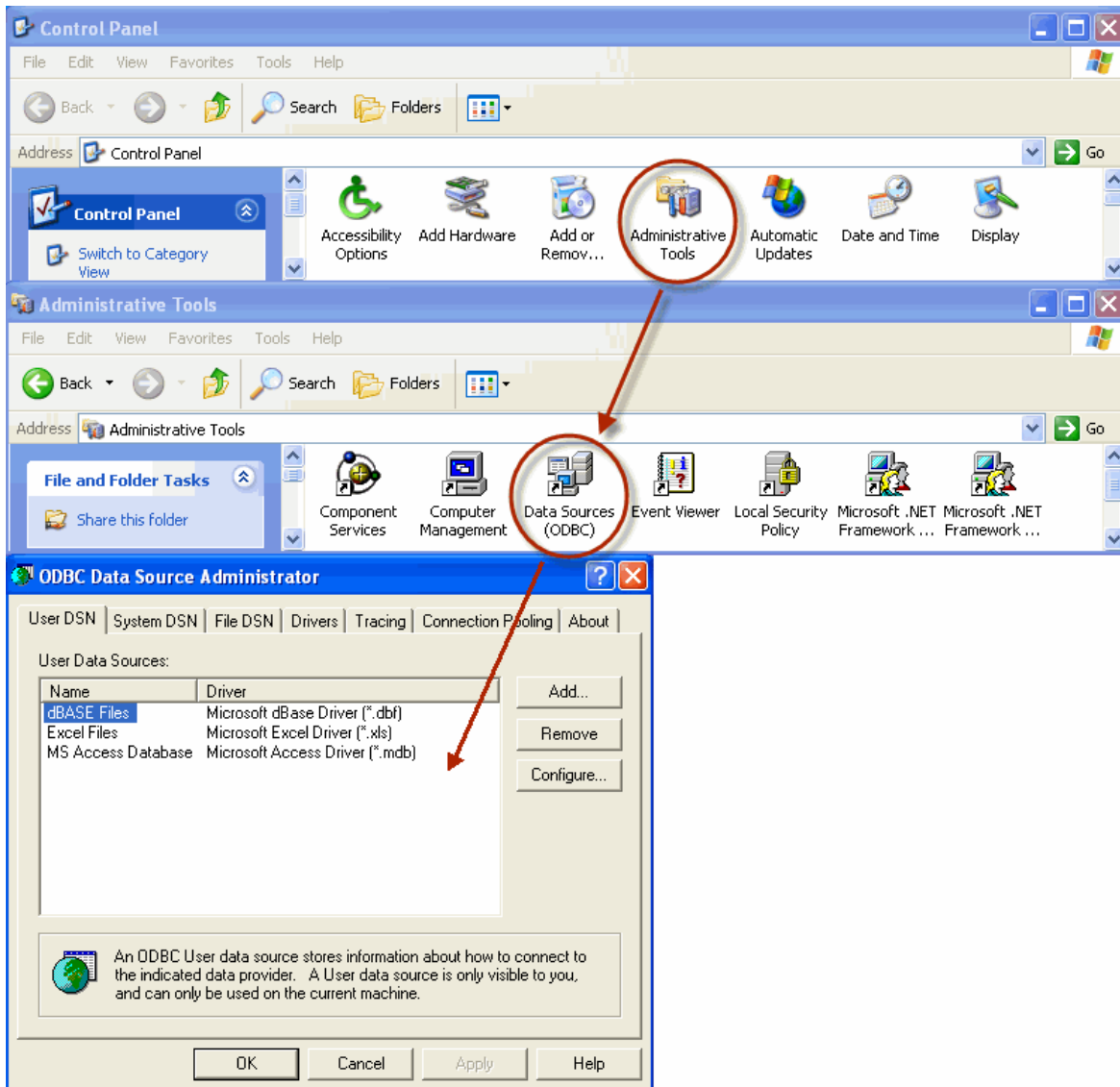
- DxDesigner and DxDataBook software is installed on your computer
- Microsoft Access is installed or available in a network location

Procedure

1. If a Microsoft Access database (.mdb file) does not already exist, use Microsoft Access to create one and populate it with component data (see Chapter 2, “[Planning Your Library Strategy](#)”).
2. Copy the Microsoft Access database (.mdb file) to a network location that is accessible by all users.


3. Open the Windows Control Panel and double-click Administrative Tools (see [Figure 3-8](#)).

Figure 3-8. Adding a Microsoft Access Driver (Windows XP Example)



4. In the Administrative Tools window, click Data Sources (ODBC).
5. In the ODBC Data Source Administrator dialog box, click Add.
6. In the Create New Data Source dialog box, select Microsoft Access Driver (*.mdb) and click Finish.
7. In Data Source Name, type the name for the database (for example, "Parts," "Components," or "DxDB Parts").
8. Click Select.

9. If the database is accessed over a network, type the Universal Naming Convention (UNC) path to the *.mdb* file in Database Name.

 **Note** Do not use the browse option when entering the UNC path to the *.mdb* file because it yields a path containing a network drive letter, which could be different for each user. If the computer name in the UNC path is changed, you will need to rebuild the DxDataBook configuration file (see Chapter 4, “[Creating a DxDataBook Configuration File](#)”).

The following is a sample UNC path:

```
\\server1\D\data\dxdb\dxdb_parts.mdb
```

10. Click **OK** in the dialog boxes.

Results

The Microsoft Access database is now included as an ODBC data source and added to the local configuration, which makes it available to use as a data source for DxDataBook.


Related Topics

- [Client/Server Database System](#)
- [DMS Connector Database](#)
- [DxDataBook Web Server](#)
- [Microsoft Excel Spreadsheets](#)

Microsoft Excel Spreadsheets

DxDataBook can be configured to use data stored in Microsoft Excel 5.0, 95, 97, and 2000 spreadsheets. DxDataBook accesses Microsoft Excel spreadsheets using the Microsoft Excel ODBC driver (see [Adding an ODBC Data Source](#)).

A named range, which corresponds to a specific DxDataBook table, is added to the Excel spreadsheet which maps the ODBC driver to the different spreadsheet components (see [Adding Named Ranges](#)).

 **Tip:** DxDataBook performance is not optimal when using Excel spreadsheets and text files as data sources (Mentor recommends using relational databases as data sources).

Related Topics

- [Client/Server Database System](#)

- [DMS Connector Database](#)
- [DxDataBook Web Server](#)
- [Microsoft Access Database](#)

Adding Named Ranges

The following topic describes how to add named ranges to an Excel spreadsheet.

Procedure

1. Open a Microsoft Excel spreadsheet.
2. Select the range of cells that make up a table (usually the entire spreadsheet) and include the header row in the selection.
3. Select **Insert > Name > Define**.
4. In the Define Name dialog box, enter the name for the table and click Add.
5. Click **OK**.
6. Save the spreadsheet and exit Excel.

Note



If DxDataBook does not recognize the tables, try creating a blank database in Microsoft Access. If this does not work, there may be a problem with the named range or ODBC driver setup.

Results

You have added named ranges to your Excel spreadsheet.

Related Topics

- [Microsoft Excel Spreadsheets](#)
 - [Adding an ODBC Data Source](#)

Adding an ODBC Data Source

The following topic describes how to add an ODBC data source.

Procedure

1. In Windows, select **Start > Control Panel** and double-click Administrative Tools (see [Figure 3-8](#)).
2. In the Administrative Tools window, double-click Data Sources (ODBC).

3. In the ODBC Data Source Administrator dialog box, click Add.
4. In the Create New Data Source dialog box, select the Microsoft Excel Driver [*.xls] option and click Finish.
5. In the ODBC Microsoft Excel Setup dialog box, type a name that you want to use to identify the ODBC connection into Data Source Name.
6. In the **Version** menu, select Excel 5.0/7.0 or Excel 97-2000.
7. Click Select Workbook and select the *.xls* file that contains the data.
8. Click **OK**.

Results

The Excel spreadsheet is now included as an ODBC data source and added to the local configuration, which makes it available to use as a DxDataBook data source.

Related Topics

- [Microsoft Excel Spreadsheets](#)
 - [Adding Named Ranges](#)

Chapter 4

Defining Data Sources, Libraries, and Symbols in the Configuration File

This topic describes how to define data sources, libraries and symbols in the DxDataBook configuration file (<filename>.dbc).

Note



Your system administrator or librarian is generally responsible for creating and editing the DxDataBook configuration file.

The configuration file:

- Contains information needed to connect DxDataBook to a database.
- Specifies how information stored in the database is displayed in the form of DxDataBook libraries and component properties.
- Determines symbol location when placing a component on a schematic.

If your library management methodology requires the ability to connect to different data sources for different purposes (for example, a database with development parts and a database with production parts, or several databases with project-specific parts), multiple <filename>.dbc files may exist and can be managed on an individual or group basis.



Tip: To keep the configuration file safe and avoid corruption, set the .dbc file access permissions to read-only for DxDataBook users, and only allow an administrator or librarian edit rights.

See “[Using DxDataBook Data Editor in Library Manager](#)” in the *Library Manager Process Guide* for more information about the DxDataBook configuration file.

Related Topics

- [Creating a DxDataBook Configuration File](#)
- [Editing a DxDataBook Configuration File](#)
- [Connecting to a DxDataBook Configuration File](#)
- [Filtering Library Data in the DxDataBook Configuration File](#)
- [Modifying a DxDataBook Configuration File with the DBCtool Utility](#)

Creating a DxDataBook Configuration File

To create a new configuration file for DxDataBook, you must use the New Configuration dialog box to set up a new *.dbc* file and the Library Wizard to add one or more libraries to the configuration:

- [Arranging Library Tables](#)
- [Specifying Data Sources](#)
- [Selecting Library Tables](#)
- [Modifying Property Types](#)
- [Modifying Property Names](#)
- [Specifying a Symbol for Library Components](#)

UNIX



Data sources that require a user-configured ODBC driver are not supported for UNIX.

Prerequisites

- Verify that a database containing component information tables already exists before creating a new configuration file (see Chapter 2, “[Planning Your Library Strategy](#)”)
- When creating a DxDataBook configuration file, you must type the name of the driver in the DSN text field when specifying how DxDataBook connects to the database (see Chapter 3, “[Setting the ODBC-to-Web Server Connection](#)”)

Note



If you do not know the location of the appropriate ODBC driver for the database, consult your system administrator or librarian.

Procedure

1. In the DxDataBook window, right-click to display the DxDataBook menu.
2. Select **Configure > New** to display the New Configuration dialog box.
3. In the New Configuration dialog box, select the I would like to use the Library Wizard to set up DxDataBook option and click **OK** to start the Library Wizard.

Note



You can also create or edit a configuration file using an XML editor.

Results

The Library Wizard is invoked and it can be used to create a DxDataBook configuration file.

Related Topics

- [Editing a DxDataBook Configuration File](#)
- [Connecting to a DxDataBook Configuration File](#)
- [Filtering Library Data in the DxDataBook Configuration File](#)
- [Modifying a DxDataBook Configuration File with the DBCtool Utility](#)

Arranging Library Tables

A library defines the mapping between DxDataBook and database tables containing component information, allowing DxDataBook to access information stored in the component database. A library is also a collection of components with similar properties that you define according to your information needs.

For example, you can organize all the capacitor information from the database into a single library, and then reference four standard capacitor symbols to allow for horizontal, vertical, polarized, and non-polarized placement.

A DxDataBook library does not require the tables in a database to follow any special database schema or columns in the database tables to have specific names.

Note



The type of data (string or numeric) used in the database table columns affects how search operators are managed.

Every DxDataBook configuration file must have at least one defined library. When creating a new DxDataBook configuration file, you must define a library before DxDataBook can use the configuration file. If the DxDataBook configuration file already exists, you can add additional libraries to that configuration file using the Library Wizard.

Prerequisites

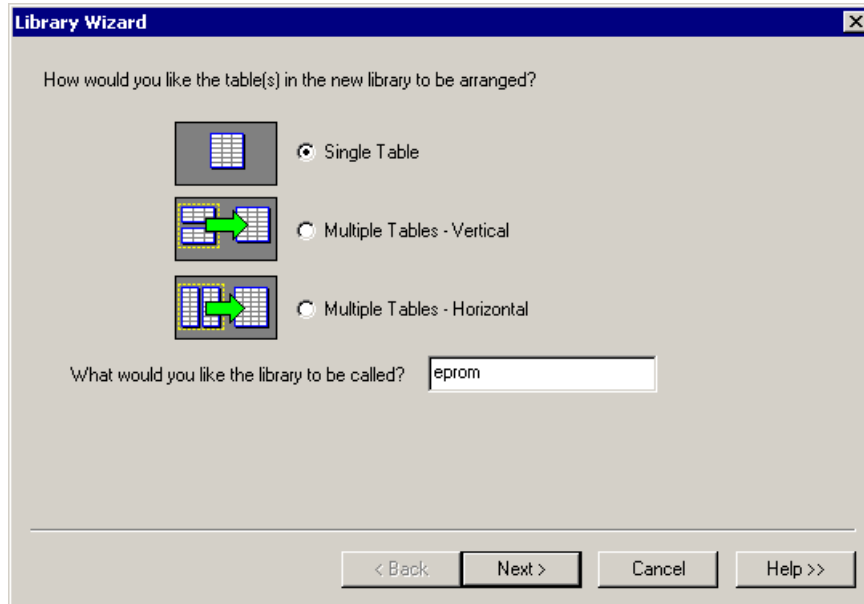
- Invoke the DxDataBook Library Wizard (see [Creating a DxDataBook Configuration File](#))

Procedure

1. Start DxDataBook and make sure that it is connected to a configuration file (see [Connecting to a DxDataBook Configuration File](#)).
2. In the DxDataBook window, right-click to display the DxDataBook menu.

3. Select **Configure > New** to display the New Configuration dialog box.
4. In the New Configuration dialog box, select the I would like to use the Library Wizard to set up DxDataBook option and click **OK** to open the Library Wizard (see [Figure 4-1](#)).

Figure 4-1. Library Wizard — Library Table Arrangement



5. Select one of the following table arrangement options:

Note



DxDataBook provides the ability to link multiple tables vertically or horizontally. This capability exists mainly for using text and Excel databases as a data source. For other data sources, use the built-in table linking mechanisms in the database and connect DxDataBook to the main or virtual table by selecting the Single Table option. Linking tables in the data model, instead of through the Library Wizard, simplifies and improves efficiency of DxDataBook management and can also speed up component searches.

- **Single Table** — All library data is contained within a single table.
- **Multiple Tables - Vertical** — All library data is contained within multiple, vertically linked tables. Each table must have the same structure and field data types, contain the same number of columns, and have the same column names. Each table might have a different number of rows.

Vertically linked tables can provide a single search interface for similar components stored in multiple tables. For example, you could search on capacitors, but the tables in the data source are divided into polarized and non-polarized capacitors (see [Figure 4-2](#)). Vertical linking is similar to concatenating or appending table data.

Figure 4-2. Vertically Linked Tables

Primary Key	A	B	C	D	E	F
P-AAAA						
P-AAAB						
P-AAAC						
P-AAAD						
P-AAAE						

Primary Key	A	B	C	D	E	F
NP-BBBA						
NP-BBBB						
NP-BBBC						
NP-BBBD						

Polarized Capacitor Table

Non-Polarized Capacitor Table

Column Names are the Same and Have the Same Type of Data

- **Multiple Tables - Horizontal** — All library data is contained within multiple, horizontally linked tables. Horizontally linked tables join disparate information for the same set of components.

Horizontal linking is usually performed when different organizations within the same company want to control and share information. Horizontal linking allows each department full read access to other databases or tables within the database. For example, if manufacturing information (Cost, Lead-Time) is in one database table, and engineering (Value, Tolerance) is in another database or database table, you can horizontally link the tables.

Horizontal linking also reduces the duplication of data within a table. For example, component information can be located in one table and manufacturer information (name, address, and web page) can be placed in a separate manufacturer table, with the tables linked through the manufacturer name (see [Figure 4-3](#)).

Figure 4-3. Horizontally Linked Tables

Component Information Table:

Primary Key	Manuf. Name	A	B	C	D
P-AAAA	TI				
P-AAAB	MOT				
P-AAAC	TI				
P-AAAD	MOT				
P-AAAE	SHLZ				

Manufacturers Table:

Manuf. Name	Name	Address	Web Page
TI	Texas Instr	123 Cherry Ln	ti.com
MOT	Motorola	870 Lawn View	mot.com
SHLZ	Shultz	3912 Disk Drive	shultz.com

Resulting Virtual Table:

Primary Key	Manuf. Name	Name	Address	Web Page	A	B	C	D
P-AAAA	TI	Texas Instr	123 Cherry Ln	ti.com				
P-AAAB	MOT	Motorola	870 Lawn View	mot.com				
P-AAAC	TI	Texas Instr	123 Cherry Ln	ti.com				
P-AAAD	MOT	Motorola	870 Lawn View	mot.com				
P-AAAE	SHLZ	Shultz	3912 Disk Drive	shultz.co				

6. In the What would you like the library to be called? text field, type a name for the library.
7. Click Next to display the next page of the Library Wizard.

Results

The library table arrangement (single, multiple tables - vertical, or multiple tables - horizontal) has been specified.

Related Topics

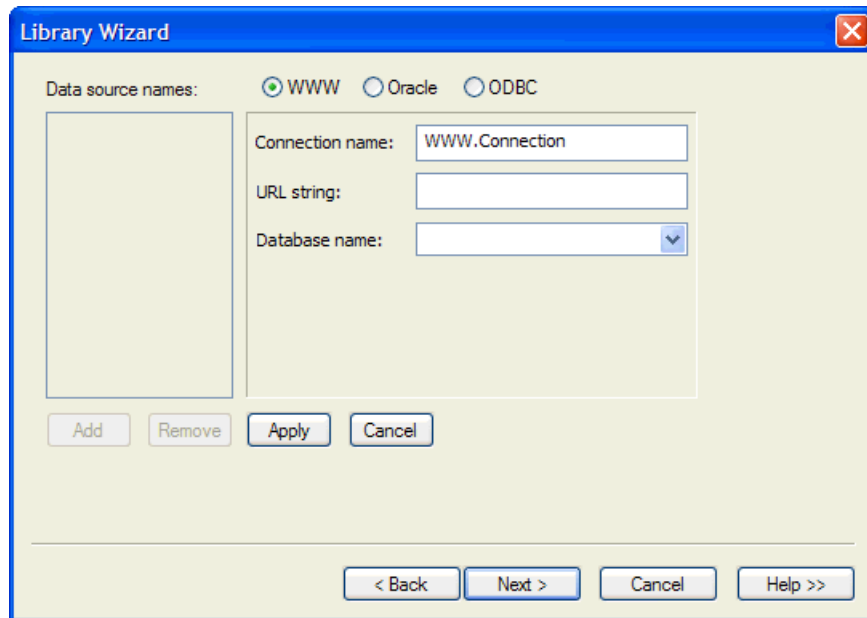
- [Specifying Data Sources](#)
- [Selecting Library Tables](#)

- [Modifying Property Types](#)
- [Modifying Property Names](#)
- [Specifying a Symbol for Library Components](#)

Specifying Data Sources

To create a new DxDataBook configuration file, the library data source names need to be specified (see [Figure 4-4](#)).

Figure 4-4. Library Wizard — Data Source Names



UNIX



Data sources that require a user-configured ODBC driver are not supported for UNIX.

Prerequisites

- [Arranging Library Tables](#)

Procedure

1. In the Library Wizard, click Add.

To remove a DxDataBook server, select a server name from the Data source names column and click Remove.

2. Select a database source at the top of the page (WWW is shown in [Figure 4-4](#)).

3. In the Connection Name text field, type the name you want to give to this data source.
4. Fill in the remaining fields to access the location of the data source you are adding.
5. Click **Apply** to display the connection name you specified in the Data source names column.
6. [Optional] To add more than one DxDataBook server location, repeat the preceding steps.
7. Click Next to display the next page of the Library Wizard.

Results

The DxDataBook library data source names are specified.

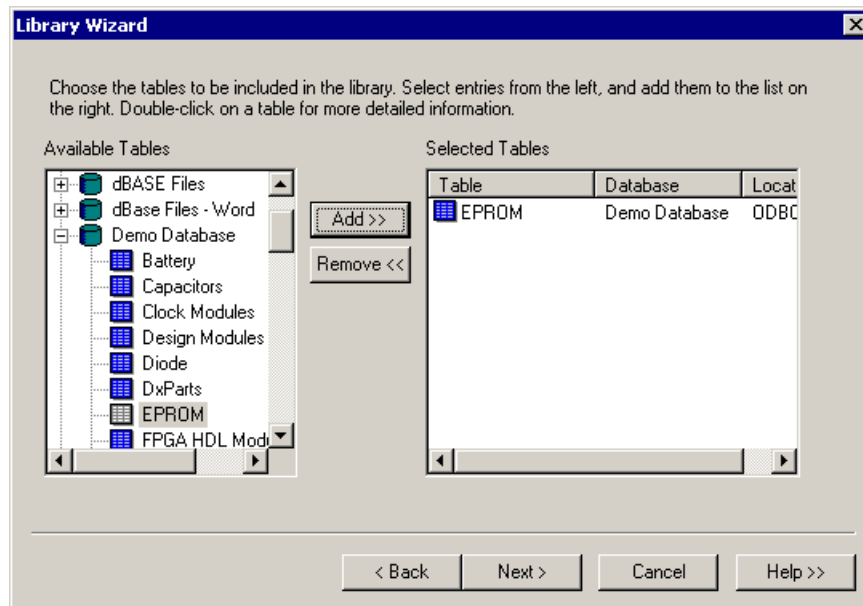
Related Topics

- [Creating a DxDataBook Configuration File](#)
 - [Selecting Library Tables](#)
 - [Modifying Property Types](#)
 - [Modifying Property Names](#)
 - [Specifying a Symbol for Library Components](#)

Selecting Library Tables

To create a new DxDataBook configuration file, select which database tables to use in the library (see [Figure 4-5](#)).

Figure 4-5. Library Wizard — Library Tables



Prerequisites

- [Specifying Data Sources](#)

Procedure

1. In the Library Wizard, select the table to add to the library from Available Tables.
2. Click Add to add the table to Selected Tables.

To remove a table from the Selected Tables, so it is not included in a library being created, select the table and click Remove.

3. Click Next to display the next page of the Library Wizard.

Note



Once a table has been added to the library, the only way to prevent the configuration file from using the table is to edit the file (see [Editing Library Information](#)).

Results

The DxDataBook library database tables are selected.

Related Topics

- [Creating a DxDataBook Configuration File](#)
 - [Arranging Library Tables](#)

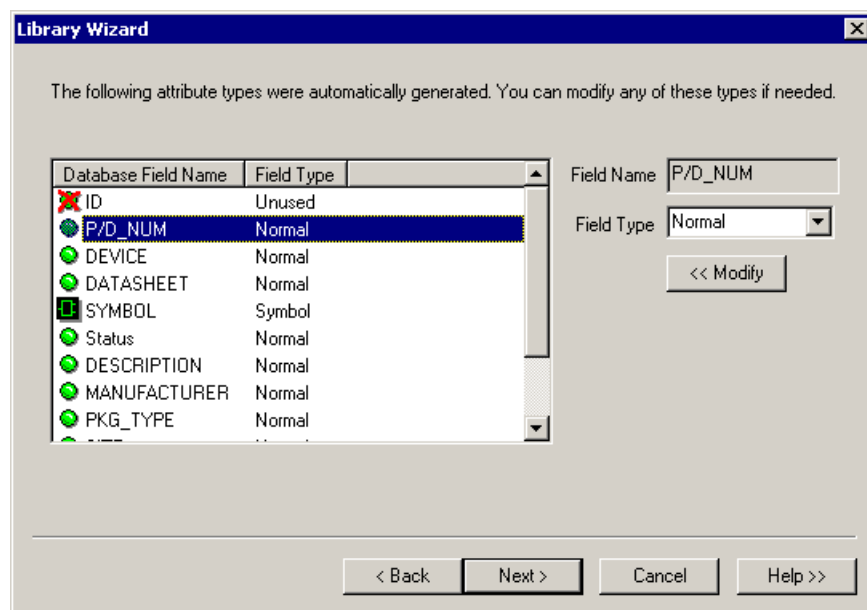
- [Modifying Property Types](#)
- [Modifying Property Names](#)
- [Specifying a Symbol for Library Components](#)

Modifying Property Types

The Library Wizard property types page lists the field (column) names located in the database tables you specified, along with the type associated with each field (see [Figure 4-6](#)).

DxDataBook automatically generates the Database Field Name and Field Type mapping.

Figure 4-6. Library Wizard — Property Types



Prerequisites

- [Selecting Library Tables](#)

Procedure

1. In Library Wizard Database Field Name column, select the database field name or type you want to change. The selected field name and type appear in the right-hand side text fields.
2. In the Field Type column, select a field type:
 - **Normal** — The field (column) in the database table contains normal property values that DxDataBook back annotates to the schematic when placing or updating a component.

- **Symbol** — The field (column) in the database table contains the name of the symbol that DxDataBook uses for a particular component. Only one database field name should be designated with the Symbol field type.



Tip: The "Use symbol data from Central Library" option can also be used to create a Symbol field in the database (see "Tip" in Chapter 2, "[Table Column Constructs](#)").

You can also designate a single symbol that applies to the entire library using the Library Wizard (see [Specifying a Symbol for Library Components](#)).

- **Document** — The field (column) in the database table provides a path to a document or the URL of a web page. Setting the field type to Document allows users to single-click a property value in a DxDataBook query results window, and automatically display the datasheet in a viewer or web page in a browser. DxDataBook does not back-annotate a property with field type set to Document to the schematic. If back-annotation of a path or URL to the schematic as a property is desired, set the field type to Normal.
- **Unused** — DxDataBook ignores the field (column). This choice is useful for database ID fields, approvals, or any other data whose purpose is informational and should not be back annotated to a schematic.
- **Unique ID** — Identifies the field (column) as the unique identifier, or primary key, for records in the library. Only one field in the library can be designated with the Unique ID field type.



Tip: You can also double-click a field name or type to change the field type. Each time you double-click, the field type changes to the next type in the list. Double-click until the desired type appears.

3. Click Modify to change the selected field type to the one you want to specify.
4. Click Next to display the next page of the Library Wizard.

Results

The database field names and types for the DxDataBook library tables are specified.

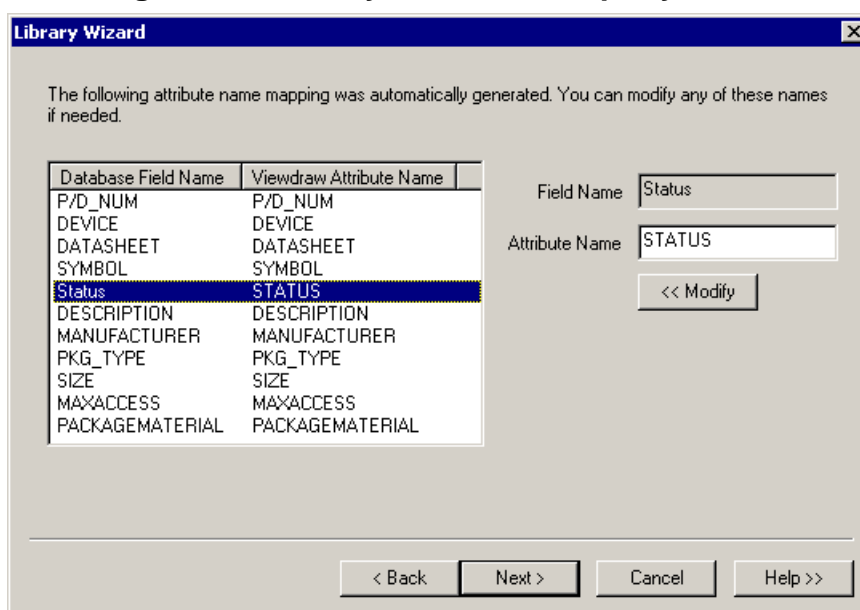
Related Topics

- [Creating a DxDataBook Configuration File](#)
 - [Arranging Library Tables](#)
 - [Specifying Data Sources](#)
 - [Modifying Property Names](#)

Modifying Property Names

The Library Wizard property names page is used to modify the DxDataBook library database table property names (see [Figure 4-7](#)). The Library Wizard automatically generates the mapping between the field name in the database table and the property name that appears in DxDataBook and DxDesigner.

Figure 4-7. Library Wizard — Property Names



Prerequisites

- [Modifying Property Types](#)

Procedure

1. In Library Wizard Database Field Name column, select a database field name or the DxDesigner property name. The database field and DxDesigner property names appear in the right-hand text fields.
2. In the Attribute Name text field, change the name to how it should appear in DxDesigner and then click Modify.
3. Repeat the procedure for all of the property names that you want to change.
4. Click Next to display the next page of the Library Wizard.

Results

The property names for the DxDataBook library database tables are specified.

Related Topics

- [Creating a DxDataBook Configuration File](#)
 - [Arranging Library Tables](#)
 - [Specifying Data Sources](#)
 - [Selecting Library Tables](#)
 - [Specifying a Symbol for Library Components](#)

Specifying a Symbol for Library Components

The Library Wizard library components symbol page is used to specify a particular symbol to associate with all of the components in the library (see [Figure 4-8](#)).

Note



This Library Wizard page is only displayed if you did not identify a database field to contain the names of symbols.

Figure 4-8. Library Wizard — Library Components Symbol

The screenshot shows a Windows-style dialog box titled "Library Wizard". The main text area contains the following text: "Since no database fields were identified as a 'Symbol' type, the symbol assignment will be done at the library level." Below this is a prompt: "Enter a symbol name to be used for all components in this library (ex: CAP.1 or CAP*):". Under the prompt is a single-line text input field. At the bottom of the dialog, there are four buttons: "< Back", "Next >", "Cancel", and "Help >>".

In DxDataBook, the recommended connectivity between library and symbols is to always have a Symbol column in the database table that contains the name of the symbol to use for a particular component.



Tip: The "Use symbol data from Central Library" option can help avoid adding duplicate data by using symbol data directly from the Central Library (see "Tip" in Chapter 2, "Table Column Constructs").

For parts that have the same symbol filename, but different extensions (such as .1 or .2 for rotation of discrettes, or ANSI versus IEEE standards), do not enter the extension. When searching the symbols files specified by the library search order in the *viewdraw.ini* file, DxDataBook automatically applies a wild card to the extensions. This allows a users to select the symbol from the drop down in the symbol previewer.

Prerequisites

- [Modifying Property Names](#)

Procedure

1. In the Library Wizard Enter a symbol name to be used for all components in this library text field, type a symbol name.

For example, if your library contains capacitors, you can use the following symbol names:

- **CAP.1** — Numeric symbol names.
 - **CAP.*** — Symbol names with wildcards.
 - **CAPH.&, CAPV.*** — List of values with wildcards.
 - **MYPARTS;CAP.*** — Aliases.
2. Click Next to display a summary of the DxDataBook library settings.

To make changes to the library settings in the Library Wizard pages, use the Back button. If the library settings are correct, click Finish to create the library.

Results

A specific symbol is specified and associated with all of the components in the library. Also, the library settings are reviewed and the library is created.

Related Topics

- [Creating a DxDataBook Configuration File](#)
 - [Arranging Library Tables](#)
 - [Specifying Data Sources](#)
 - [Selecting Library Tables](#)

- [Modifying Property Types](#)

Editing a DxDataBook Configuration File

DxDataBook configuration files are set up initially by a system administrator or librarian as part of an enterprise or corporate library strategy (see Chapter 2, “[Planning Your Library Strategy](#)”). A configuration file may need to be edited if there is a change to the data model in the database or information in the DxDataBook graphical user interface.

Note



The *.dbc* file uses XML (ASCII) format and can be edited with a text or XML editor (see Appendix B, [Configuration File Format](#)).

[Table 4-1](#) shows DxDataBook changes that may occur and whether or not these changes require edits to the configuration file.

Table 4-1. DxDataBook Changes/Configuration File Edits

DxDataBook Change	Configuration File Edits Required?
Adding rows with new part information to an existing database table.	No
Creating a new database table with new part information.	Yes
Adding a new column to an existing database table.	Yes
Removing a column from a database table.	Yes
Renaming a column in a database table.	Yes
Changing a value or values in an existing column in a database table.	No
Reordering the rows in a database table.	No

Related Topics

- [Creating a DxDataBook Configuration File](#)
- [Connecting to a DxDataBook Configuration File](#)
- [Filtering Library Data in the DxDataBook Configuration File](#)
- [Modifying a DxDataBook Configuration File with the DBCtool Utility](#)
 - [Editing Library Information](#)
 - [Specifying Symbols to Exclude](#)

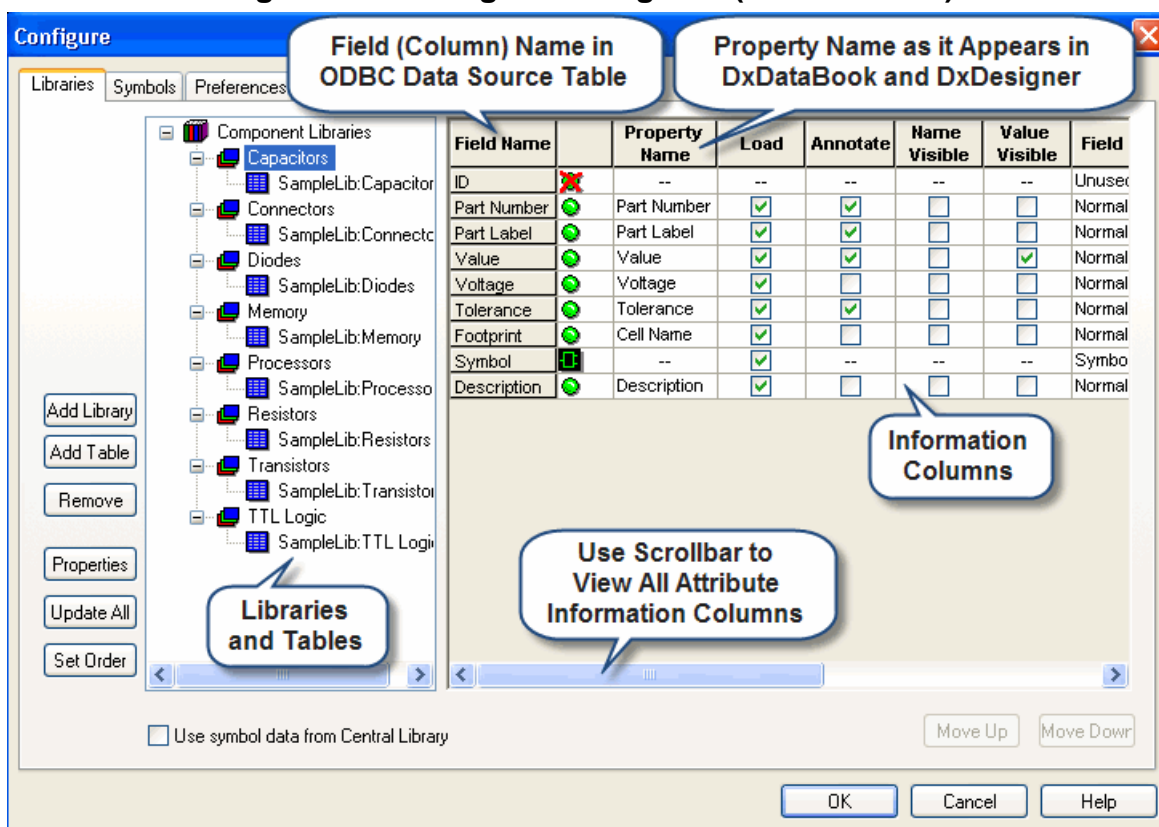
- [Setting Configuration File Preferences](#)

Editing Library Information

The following topics describe how to edit DxDataBook library information (see [Figure 4-9](#)):

- [Adding, Editing or Deleting a Library](#)
- [Adding a Table to a Library](#)
- [Associating Symbols with Library Components](#)
- [Editing Library Properties](#)
- [Reordering Libraries and Fields](#)
- [Setting Multiple Library Properties to the Same Type](#)

Figure 4-9. Configure Dialog Box (Libraries Tab)



Adding, Editing or Deleting a Library

You can add a new component library for the active configuration file, and specify the table or tables that contain data for selected library (see [Adding a Table to a Library](#)). This topic also describes how to edit or delete an existing component library.

Note



Component Libraries group together one or more tables containing information for the same types of parts (for example, all capacitors can share a component library). The tables within a component library can be from the same or different relational databases.

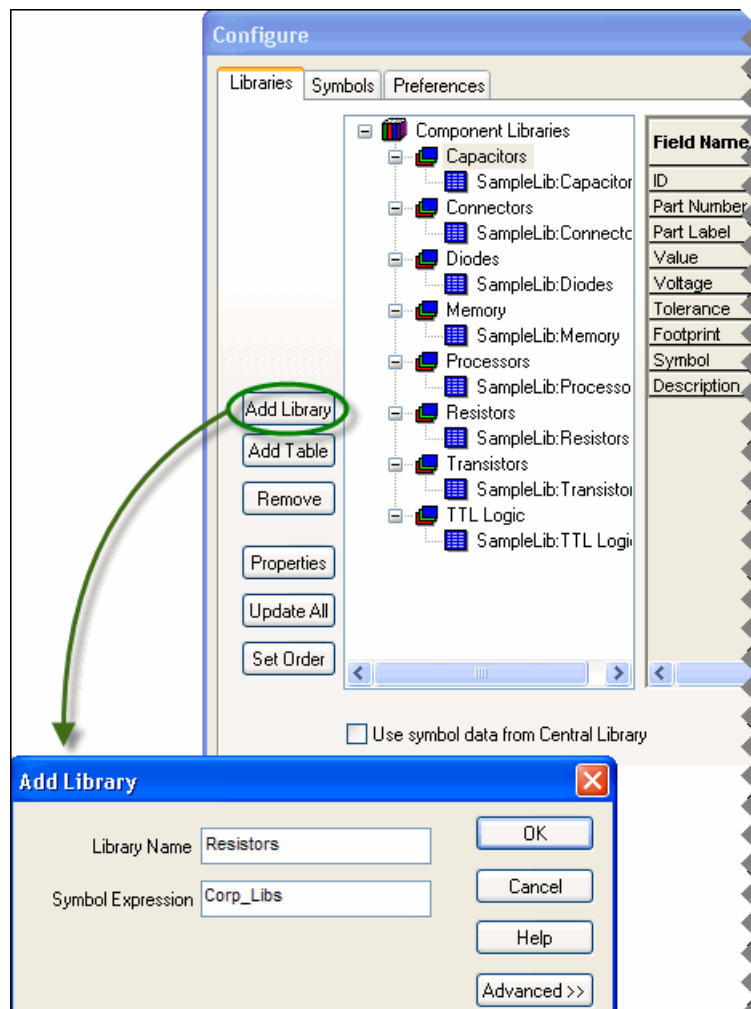
Procedure

1. In the DxDataBook window, select **Configure > Edit Configuration**.

The Configure dialog box is displayed with the **Libraries** tab selected. The left pane displays the component libraries and tables from the data source contained in each library in tree-view format. The right pane shows detailed information for each library.

2. Click Add Library (see [Figure 4-10](#)).

Figure 4-10. Configure Dialog Box — Adding a Library



3. In the Add Library dialog box, type the name of the library to add in the Library Name text field.

You can also edit existing DxDataBook library information (such as the library name and symbol expression) using the Add Library dialog box.

4. If the database table or tables you plan to associate with the library does not have a field that contains the symbol name, use the Symbol Expression option to specify a symbol expression with which to find a valid symbol.

Note



Mentor Graphics recommends including a Symbol column in the database table with each line containing a symbol file name for a particular component as a value (see Chapter 2, “[Data Source Creation](#)”). To enter a symbol name into the database, type the base symbol file name (for example, type *res* if the a resistor has *res.1* and *res.2* symbols).

For example, if you have a capacitor library, and there is no symbol field in the database that shares the same generic symbol, you might set this field to any of the following values: *cap.1*, *cap**, *caph**, or *capv**.

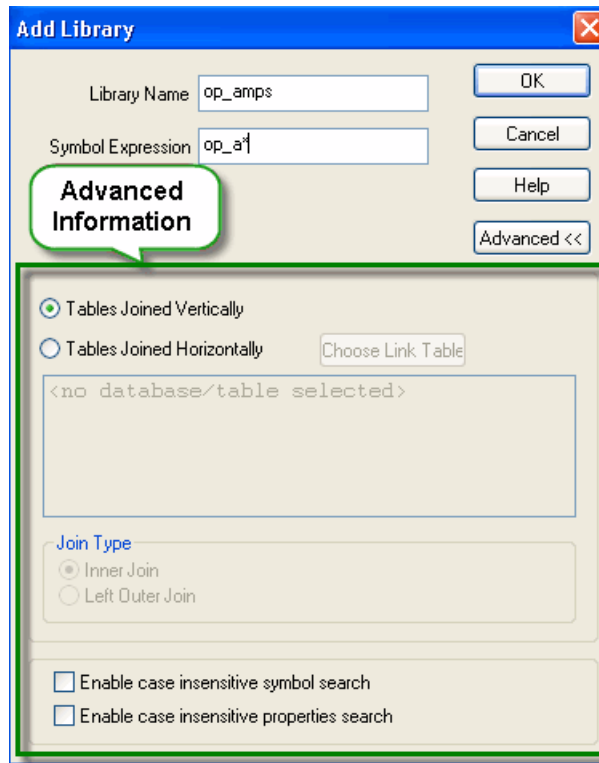
If a Symbol field in your database is specified as a Symbol field type in the configuration file, leave the Symbol Expression option empty.

5. [Optional] If your library contains data from multiple tables, click Advanced to specify how you want to link tables in the library (see [Figure 4-11](#)). If your library only contains data from a single table, or if the tables are already linked in the database, this step is not required.



Tip: The ability to link tables vertically and horizontally is primarily used for databases that do not have this capability, such as Microsoft Excel or text files. For other data sources, use the built-in table linking mechanisms in the database.

Figure 4-11. Add Library Dialog Box — Advanced Options



- a. Select one of the following options:
 - **Tables Joined Vertically** — Links tables vertically to provide a single search interface for similar components stored in multiple tables. Tables to link must have the same structure and field data types, and the tables must contain the same number of columns with the same column names.
 - **Tables Joined Horizontally** — Links tables horizontally and provides access to component properties that reside in multiple tables. Horizontally linked tables join disparate information about the same set of components to provide access to all the information in a single virtual table. Selecting Tables Joined Horizontally enables the Choose Link Table and Join Type options.
- See [Arranging Library Tables](#) for more information about vertically and horizontally linked tables.
- b. Click Choose Link Table to select the table to which to link (see [Adding a Table to a Library](#)). DxDataBook displays the database and table names to which you have linked.
 - c. Under Join Type, select one of the following options:
 - **Inner Join** — Combines records from two tables to a query's results only if the values of the joined fields meet certain specified criteria. The values of the joined fields must be equal.

- **Left Outer Join** — Includes records even if there are no related records in the joined tables.
- d. Select one or both of the following options to enable “case insensitive” symbol and property searches when running a database query:
 - **Enable case insensitive symbol search** — Case insensitive searches are performed on symbols in the database tables (default - option not selected).
 - **Enable case insensitive properties search** — Case insensitive searches are performed on properties in the database tables (default - option not selected).

Note



If symbols or properties on the schematic contain mixed case letters (for example, Cap.1 and cap.1 symbols), and the corresponding option is not selected, a query may produce verification errors.

If these option is not selected and properties on the schematic use mixed case letters, this can lead to verification errors.

1. Click **OK**.

To delete a component library:

Caution



If a component library that is connected to a DXDB_LIBNAME property is deleted from the configuration file, an error message is generated (a DXDB_LIBNAME property is automatically created by DxDataBook when placing or annotating a DxDesigner component to indicate the library where the component information originated). To resolve this problem, add the library using the same name and table to the configuration file. If the library name changes, you must update the DXDB_LIBNAME value throughout the entire design.

1. In the DxDataBook window, select **Configure > Edit Configuration**.
2. In the left pane under Component Libraries, select the library and click Remove.
3. Click **Yes** in the dialog box.

Results

A new DxDataBook component library is added to the active configuration file and tables that contain data for selected library are specified.

Related Topics

- [Editing Library Information](#)

- [Associating Symbols with Library Components](#)
- [Editing Library Properties](#)
- [Reordering Libraries and Fields](#)
- [Setting Multiple Library Properties to the Same Type](#)

Adding a Table to a Library

You can use DxDataBook to add a table to a library for the active configuration file (table resides within a database that contains component information). A DxDataBook library might display data contained in one or several database tables. You can use database joins to link together tables of disparate information. You can also use a DxDataBook or Access query to link these tables together.

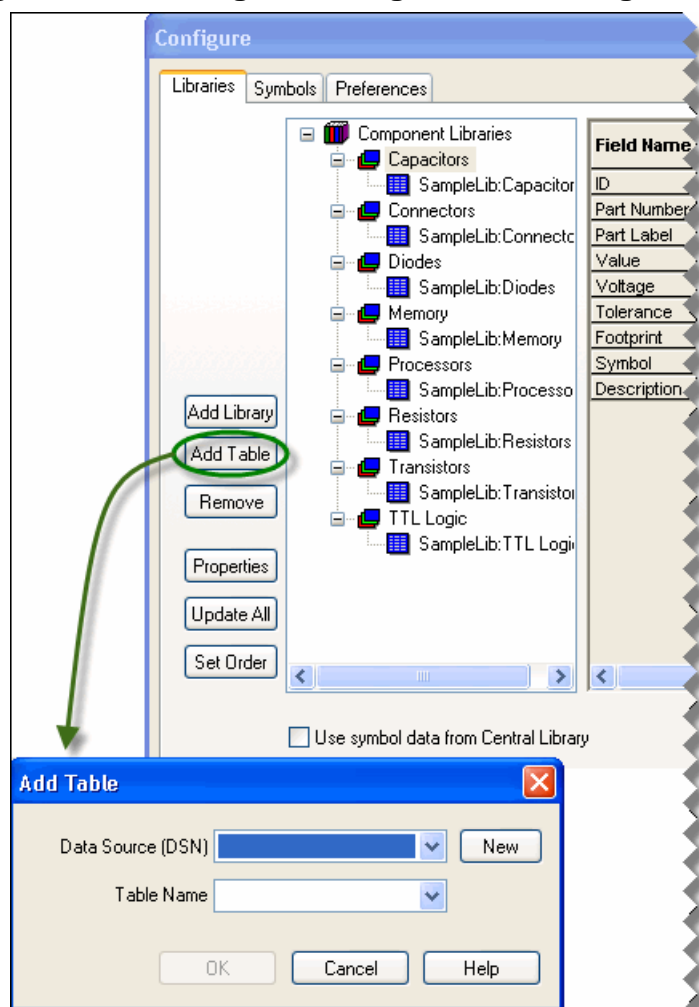
Procedure

1. In the DxDataBook window, select **Configure > Edit Configuration**.

The Configure dialog box is displayed with the **Libraries** tab selected. The left pane displays the component libraries and tables from the data source contained in each library in tree-view format. The right pane shows detailed information for each library.

2. In the left pane under Component Libraries, select a library.
3. Click Add Table (see [Figure 4-12](#)).

Figure 4-12. Configure Dialog Box — Adding a Table



4. In the Add Table dialog box, select the configuration file from the **Data Source Name (DSN)** menu.

You can also click New to connect to a new SQL Data Source.

5. In the **Table Name** menu, select a table and click **OK**.

Results

A database table is added to a DxDataBook library for the active configuration file.

Related Topics

- [Editing Library Information](#)
 - [Adding, Editing or Deleting a Library](#)
 - [Associating Symbols with Library Components](#)

- [Editing Library Properties](#)
- [Reordering Libraries and Fields](#)
- [Setting Multiple Library Properties to the Same Type](#)

Associating Symbols with Library Components

You can associate symbols with components using one of the following methods:

- **Global** — Associates a symbol at the library level and is only appropriate when all the components in a library share the same symbol.
- **Individual** — Uses a column in a database table to specify a different symbol for each component in the library so the association is made individually for each component.

Note



The DxDataBook Library Wizard also provides global and individual association options (see [Specifying a Symbol for Library Components](#)). You use either the Library Wizard or edit an existing DxDataBook configuration file to specify which field (column) in a database table should have the Symbol field type value. If you do not identify a field as the source of the symbol association, specify a library-level symbol name.

Mentor Graphics recommends creating a data model where all tables have a Symbol column that individually associates a symbol to a specific component. Using a similar structure to identify a component symbol simplifies understanding and maintaining database tables.



Tip: The "Use symbol data from Central Library" option is an alternative to creating a Symbol field in the database (see "Tip" in Chapter 2, [Table Column Constructs](#)).

Related Topics

- [Editing Library Information](#)
 - [Adding, Editing or Deleting a Library](#)
 - [Adding a Table to a Library](#)
 - [Editing Library Properties](#)
 - [Reordering Libraries and Fields](#)
 - [Setting Multiple Library Properties to the Same Type](#)
 - [Globally Associating Symbols](#)
 - [Individually Associating Symbols](#)

Globally Associating Symbols

The global method is used to associate a symbol at the library level and is only appropriate when all the components in a library share the same symbol.

Procedure

1. In the DxDataBook window, select **Configure > Edit Configuration**.

The Configure dialog box is displayed with the **Libraries** tab selected (see [Figure 4-9](#)). The left pane displays the component libraries and tables from the data source contained in each library in tree-view format. The right pane shows detailed information for each library.

2. In the left pane under Component Libraries, select a library.
3. In the Add Library dialog box, type the name of a symbol or the appropriate wildcard expression in the Symbol Expression text field and click **OK**.

DxDataBook applies the symbol expression against all the symbols in your search path (for example, cap*, cap.1, res*, or res.1).

4. Repeat the procedure for other libraries as needed.

Results

The selected symbol is associated at the library level using the global association method.

Related Topics

- [Associating Symbols with Library Components](#)
 - [Individually Associating Symbols](#)

Individually Associating Symbols

The individual method uses a column in a database table to specify a different symbol for each component in the library so the association is made individually for each component.

Note



Only use individual method to associate symbols when the table in the data source does not already contain a column named “Symbol.”

Procedure

1. In the DxDataBook window, select **Configure > Edit Configuration**.

The Configure dialog box is displayed with the **Libraries** tab selected (see [Figure 4-9](#)). The left pane displays the component libraries and tables from the data source contained

in each library in tree-view format. The right pane shows detailed information for each library.

2. In the left pane under Component Libraries, select a library.
3. In the right pane under the Field Types column, click in the cell that corresponds to the appropriate field name in your database and select Symbol from the dropdown list.
4. Click **OK**.
5. Repeat the procedure for other libraries as needed.

If a component is selected in the Search Results window, DxDataBook selects a symbol based on the contents of the field for which you have set the symbol type.

Results

A column is used in a database table to specify a different symbol for each component in the library so the association is made individually for each component.

Related Topics

- [Associating Symbols with Library Components](#)
 - [Globally Associating Symbols](#)

Editing Library Properties

To edit DxDataBook configuration file library properties, use the **Configure > Libraries** page (see [Figure 4-9](#)).

Note



If you are using the Expedition flow, a symbol entry is no longer needed in the *.dbc* file because the Part Number property is automatically linked to the symbol used in the parts database (PDB) definition.

Procedure

1. In the DxDataBook window, select **Configure > Edit Configuration**.

The Configure dialog box is displayed with the **Libraries** tab selected. The left pane displays the component libraries and tables from the data source contained in each library in tree-view format. The right pane shows detailed information for each library.
2. In the left pane under Component Libraries, select a library or table.
3. In the right pane, select the property to edit (use the scroll bar to view all property columns).
 - a. In the Property Name column, select a name.

The Property Name is the name of the property as it appears in DxDataBook and DxDesigner, and it does not have to be the same as the name in the Field Name column, which is the name of the column in the ODBC data source table.

- b. In the Load column, use the check boxes to specify which properties to load into DxDataBook when using Load Component to load a component from DxDesigner.
- c. In the Annotate column, use the check boxes to specify which properties to back-annotate to DxDesigner when using Annotate Unique or Annotate Generic.
- d. In the Name Visible column, use the check boxes to specify the properties whose names should be visible in a DxDesigner schematic after you back-annotate the property from DxDataBook to DxDesigner.

The Name Visible column is primarily used to set visibility for properties that do not have a placeholder property name already on the symbol. For properties that have a placeholder on the symbol, visibility can be set through the symbol property instead of the configuration file.

If a placeholder property does not already exist on the symbol to define the property location, the default is to add the back-annotated properties from the database to the symbol starting at the lower left corner (0,0) and proceeding downward.

- e. In the Value Visible column, use the check boxes to specify the properties whose values should be visible in a DxDesigner schematic after you back-annotate the property from DxDataBook to DxDesigner.
- f. In the Instance Value column, use the check boxes to specify the properties that should possess the Instance Value value when instantiating components into a DxDesigner schematic.

To see the Instance Value column, instance values must have been enabled by selecting this option in the DxDesigner Project Settings dialog box (in DxDesigner, select **Project > Settings > Project (tab) > Instance Values**).

- g. In the Field Type column, select one of the following:
 - **Normal** — A field (column) in the database whose values represent “normal” DxDesigner properties and have no special meaning to DxDataBook. DxDataBook passes the value of a field whose field type is Normal to DxDesigner as a property value.
 - **Symbol** — A special field (column) in the database whose value identifies the symbol that DxDataBook should display in the DxDataBook symbol preview window and that DxDesigner should use when placing the component on a schematic.

Only one field name can be designated with the Symbol field type. If there is field name whose field type is already defined as Symbol, DxDataBook does not allow specifying the field type of another field name as Symbol.






- **Document** — A field (column) in the database whose values refer to an external document, such as a path to a datasheet or web page. Setting the field type to Document allows a user to single click the property value in DxDataBook and automatically display the datasheet in a viewer or a web page in a browser.

DxDataBook does not back-annotate a property whose field type is Document to the schematic. If back annotation of a path or URL to the schematic as a property value is desired, set the field type to Normal instead.

- **Unused** — A field (column) in the database whose values should not be displayed in DxDataBook or passed to DxDesigner.
- **Unique ID** — A field (column) in the database that contains the unique identifier, or primary key, for records in the library. Only one field in the library can be designated with the Unique ID field type.

The Field Type column you select places an icon corresponding to that field type in the Icon column (see [Table 4-2](#)).

Table 4-2. Field Type Icons

Icon	Field Type
	Normal
	Symbol
	Document
	Unused
	Unique ID

- In the Magnitude column, select a magnitude to show numeric property values in DxDataBook for the specified magnitude.

For example, selecting “Kilo” in the Magnitude column removes other valid magnitude choices in the adjacent Valid Magnitude column. The “Percent” magnitude can be used for fields such as Tolerance to append the percent symbol to the numeric value in DxDataBook. When a field contains numeric values such as size for which a magnitude is never applied, set the Magnitude column to “None.”

Some properties such as resistance, capacitance, and inductance have numeric values, but require that the data be presented using more than one magnitude (for example, your library specification states that resistance can be expressed in milliohms, ohms, Kilo-ohms, and Megaohms). When more than one magnitude applies, set the Magnitude column to “Automatic,” and then use the Valid Magnitude column to specify which magnitudes are valid.

For DxDataBook to use the Magnitude column to convert field values, the data must exist in the database table in a pure numeric form (for example, 10e-12 to represent the value of a 10 pF capacitor) and the column in the data source must be specified as numeric. The Magnitude column setting does not apply to field values that reside in a database table as string data.

- i. In the Valid Magnitude column, click and select the valid magnitude for the selected property. The Valid Magnitude column only contains choices if the Magnitude column is set to “Automatic.”

For example, capacitor manufacturers do not use milli or nano magnifiers when specifying capacitance. Therefore, using the Valid Magnitude column to exclude milli and nano magnifiers prevents display of numeric data with these magnifiers in DxDataBook.

- j. In the Units column, specify an appropriate unit for the selected property to append to the magnitude. For example, you might want the Value property for a capacitor to have always have an “F,” or the Value property for an inductor to always have an “H.”
- k. In the VM (Variant Manager) Transient column, select or clear the setting as desired. It is not necessary to check entries in the VM (Variant Manager) Transient column if your design flow does not use the DxVariant Manager application.

Note

The VM (Variant Manager) Transient column setting is only useful for component data used with ePD3.1 or earlier versions of the DxVariant Manager application.

- In the VM (Variant Manager) Invariant column, select or clear the setting as desired for the property. When selected, the VM (Variant Manager) Invariant column indicates that only parts with the property (for example, package type) are allowed as part substitutes in DxVariant Manager. It is not necessary to check entries in the VM (Variant Manager) Invariant column if your design flow does not use the DxVariant Manager application.

Results

The DxDataBook library properties are edited for the active configuration file.

Related Topics

- [Editing Library Information](#)
 - [Adding, Editing or Deleting a Library](#)
 - [Adding a Table to a Library](#)
 - [Associating Symbols with Library Components](#)

- [Reordering Libraries and Fields](#)
- [Setting Multiple Library Properties to the Same Type](#)

Reordering Libraries and Fields

The following topics describe how to reorder DxDataBook configuration file libraries and fields:

- [Reordering Libraries](#)
- [Reordering Fields in a Library](#)

Reordering Libraries

Note

The library order affects the appearance of libraries in DxDataBook and how parts are mapped to libraries when loaded from DxDesigner (as long as the part does not have a DXDB_LIBNAME property).

Procedure

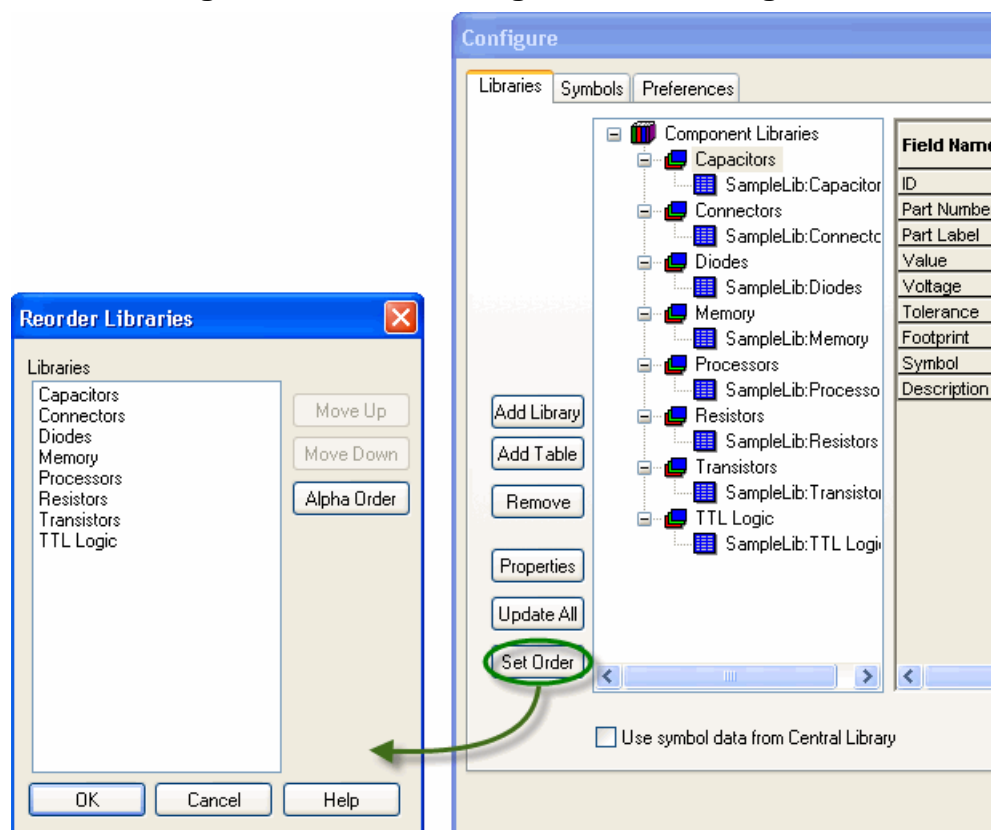
1. In the DxDataBook window, select **Configure > Edit Configuration**.

The Configure dialog box is displayed with the **Libraries** tab selected. The left pane displays the component libraries and tables from the data source contained in each library in tree-view format. The right pane shows detailed information for each library.

2. In the left pane, click Set Order to display the Reorder Libraries dialog box (see [Figure 4-13](#)).

The libraries are listed in the order they appear in the tree-view.

Figure 4-13. Reordering Libraries Dialog Box



3. Select a library and use Move Up, Move Down, and Alpha Order options to change the library order.
4. Click **OK**.

Results

The DxDataBook libraries are rearranged in the tree-view in the order that you specified.

Reordering Fields in a Library

Procedure

1. In the DxDataBook window, select **Configure > Edit Configuration**.
2. In the left pane under Component Libraries, select a library.
3. In the right pane, click the desired row in the Field Name column.

Note



DxDataBook enables the Move Up and Move Down buttons only if you select a library (not a table) in the right pane.

4. Click Move Up or Move Down to reorder the library fields.
5. Click **OK**.

Results

The DxDataBook library fields are reordered for the active configuration file.

Related Topics

- [Editing Library Information](#)
 - [Adding, Editing or Deleting a Library](#)
 - [Adding a Table to a Library](#)
 - [Associating Symbols with Library Components](#)
 - [Editing Library Properties](#)
 - [Setting Multiple Library Properties to the Same Type](#)

Setting Multiple Library Properties to the Same Type

You can select multiple library field names and set their properties to the same type simultaneously if the selected properties are set to the same Field Type. If you select field names with a different Field Type (Normal and Document), you must change each Field Type individually.

Procedure

1. In the DxDataBook window, select **Configure > Edit Configuration**.

The Configure dialog box is displayed with the **Libraries** tab selected. The left pane displays the component libraries and tables from the data source contained in each library in tree-view format. The right pane shows detailed information for each library.
2. In the left pane under Component Libraries, select a library.
3. In the right pane, select two or more field names from the Field Name column:
 - For multiple “adjacent” field names, click the first field name, press and hold *Shift*, and then click the last field name in the range.
 - For multiple “non-adjacent” field names, click the first field name, press and hold *Ctrl*, and then click the other field names.
4. Double-click the icon in the column until you get the desired icon type.

DxDataBook simultaneously changes the icon types for all selected field names.

Note



If there is a field type that is already identified as “Symbol,” DxDataBook does not let you specify another field type as “Symbol” because it uses the Symbol field to collectively identify components in a library.

Results

The DxDataBook library properties are set to the same time simultaneously for the active configuration file.

Related Topics

- [Editing Library Information](#)
 - [Adding, Editing or Deleting a Library](#)
 - [Adding a Table to a Library](#)
 - [Associating Symbols with Library Components](#)
 - [Editing Library Properties](#)
 - [Reordering Libraries and Fields](#)

Specifying Symbols to Exclude

You can specify which symbols to exclude when loading components into DxDataBook from DxDesigner for verification.

Procedure

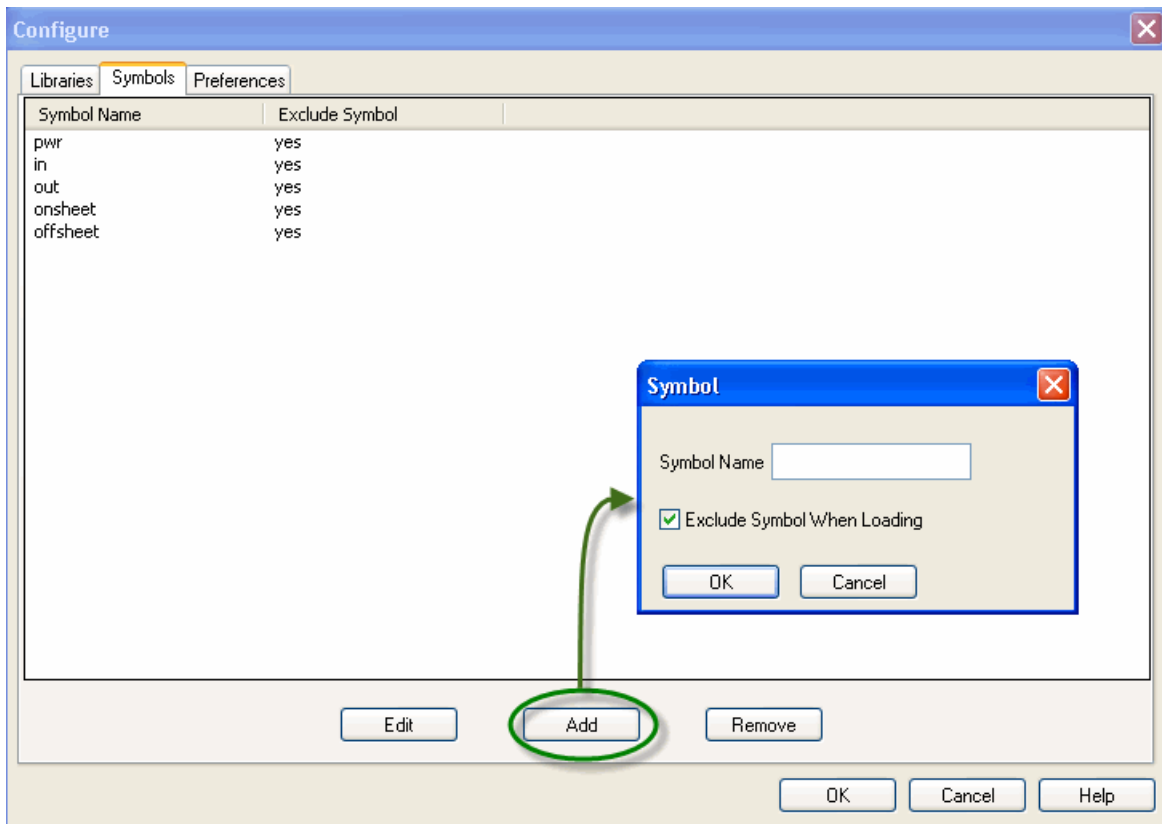
1. In the DxDataBook window, select **Configure > Edit Configuration**.

The Configure dialog box is displayed. The left pane displays the component libraries and tables from the data source contained in each library in tree-view format. The right pane shows detailed information for each library.

2. In the Configure dialog box, click the **Symbols** tab.

The Symbols page lists the symbols (if any) you previously specified to include or exclude when you load a component from DxDesigner. For example, [Figure 4-14](#) shows the Symbols sheet configured to exclude power, ground, in, out, onsheet, and offsheet symbols.

Figure 4-14. Configure Dialog Box (Symbols Tab)



3. Click Add.
4. In the Symbol dialog box:
 - a. Type the name of the symbol in the Symbol Name text field.
 - b. Select the Exclude Symbol When Loading option to avoid loading the symbol when DxDataBook loads a component from DxDesigner.
 - c. Click **OK**.

DxDataBook adds the symbol to the Symbol Name column and shows the symbol status in the Exclude Symbol column (symbol excluded if *Yes* displayed).

To edit a symbol:

1. In the Symbol Name column, select the symbol to edit.
2. Click Edit.
3. In the Symbol dialog box, edit the symbol information and click **OK**.

Results

Symbols have been specified for exclusion when loading components into DxDataBook from DxDesigner for verification.

Related Topics

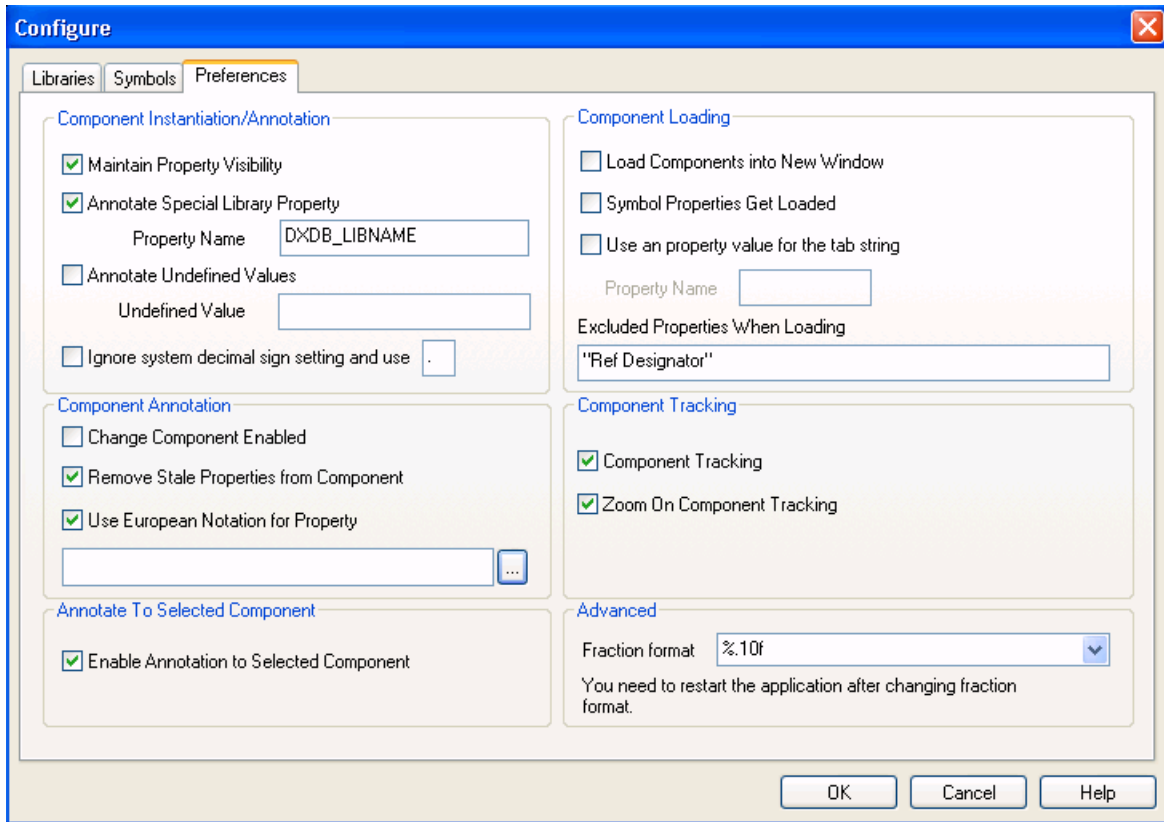
- [Editing a DxDataBook Configuration File](#)
 - [Editing Library Information](#)
 - [Setting Configuration File Preferences](#)

Setting Configuration File Preferences

You can specify the following DxDataBook configuration file preferences (see [Figure 4-15](#)):

- [Advanced: Fraction Format](#)
- [Annotate to Selected Component](#)
- [Component Annotation](#)
- [Component Instantiation/Annotation](#)
- [Component Loading](#)
- [Component Tracking](#)

Figure 4-15. Configure Dialog Box (Preferences Tab)



Advanced: Fraction Format

To access: In the DxDataBook menu, select **Configure > Edit Configuration > Preferences** (**Advanced** section).

Use this dialog box to set fraction format preferences for the DxDataBook configuration file (see [Figure 4-15](#)).

Table 4-3. Configuration > Preferences (Advanced)

Field	Description	Values
Fraction format	<p>Determines how fractional numbers appear in DxDataBook (format follows Visual C++ programming language specifications).</p> <p>For example, choosing %.10f displays all fractional values with up to 10 digits to the right of the decimal point (additional digits are rounded).</p> <p>Tip: If you are unable to find a property value in DxDataBook using search or verify, type %g into the Fraction Format menu text field (ensures that numerical data in the relational database is set to the appropriate fractional format and numerical values are not rounded).</p>	%.10f, %.16f, or %f

Related Topics

- [Setting Configuration File Preferences](#)
 - [Annotate to Selected Component](#)
 - [Component Annotation](#)
 - [Component Instantiation/Annotation](#)
 - [Component Loading](#)
 - [Component Tracking](#)

Annotate to Selected Component

To access: In the DxDataBook menu, select **Configure > Edit Configuration > Preferences** (**Annotate to Selected Component** section).

Use this dialog box to annotate properties to selected components in DxDesigner for the DxDataBook configuration file (see [Figure 4-15](#)).

Table 4-4. Configuration > Preferences (Annotate to Selected Component)

Field	Description	Values
Enable Annotation to Selected Component	Annotates properties to designated components in DxDesigner.	None

Related Topics

- [Setting Configuration File Preferences](#)
 - [Advanced: Fraction Format](#)
 - [Component Annotation](#)
 - [Component Instantiation/Annotation](#)
 - [Component Loading](#)
 - [Component Tracking](#)

Component Annotation

To access: In the DxDataBook menu, select **Configure > Edit Configuration > Preferences (Component Annotation section)**.

Use this dialog box to set component annotation preferences for the DxDataBook configuration file (see [Figure 4-15](#)).

Table 4-5. Configuration > Preferences (Component Annotation)

Field	Description	Values
Change Component Enabled	<p>Changes the DxDesigner component when a new symbol is specified.</p> <p>Note: When this option is selected, the graphic displayed in the symbol previewer replaces the graphic on the schematic.</p> <p>For example, if an analog graphic library and a resistor graphic library use the <i>res</i> symbol name, the graphic for both libraries is replaced on the schematic (may not be the intended result).</p>	None

Table 4-5. Configuration > Preferences (Component Annotation) (cont.)

Field	Description	Values
Remove Stale Properties from Component	<p>Removes properties from the component when annotating properties from DxDataBook into an existing component in DxDesigner if the following condition(s) are met:</p> <ul style="list-style-type: none"> • A property value is empty. • A property value matches the Undefined Value option value, and the Annotate Undefined Values option is not selected. • The Annotate option (Libraries tab) is not selected for a property. • A more generic version of the component that does not contain the entire set of defined properties is being annotated. <p>For example, when a DxDesigner component is loaded into DxDataBook and properties are removed from the query, resulting in an increased number of possible candidates.</p>	None
Use European Notation for Property	<p>Displays all or selected properties in configuration using European notation (based on International Electrotechnical Commission 62 Standards). Notation is applied to DxDataBook search window values and schematic symbol labels.</p>	None

Related Topics

- [Setting Configuration File Preferences](#)
 - [Advanced: Fraction Format](#)

- [Annotate to Selected Component](#)
- [Component Instantiation/Annotation](#)
- [Component Loading](#)
- [Component Tracking](#)

Component Instantiation/Annotation

To access: In the DxDataBook menu, select **Configure > Edit Configuration > Preferences** (**Component Instantiation/Annotation** section).

Use this dialog box to set component instantiation and annotation preferences for the DxDataBook configuration file (see [Figure 4-15](#)).

Table 4-6. Configuration > Preferences (Instantiation/Annotation)

Field	Description	Values
Maintain Property Visibility	Maintains the visibility of existing properties, based on the current component if annotating, or symbol if adding a new part.	None
Annotate Special Library Property	Annotates components with a property whose value defines the database library name used when the part was added to the schematic or updated (default property name is DXDB_LIBNAME).	None
Annotate Undefined Values	<p>Annotates undefined values, such as Blank, Empty, or Null, to a component property in DxDesigner when the property has no value specified in the database.</p> <p>This information is used primarily when migrating existing data from ViewDatabook II.</p> <p>Note: Relational database structures do not require every field to have a value (in ViewDatabook II, <i>.par</i> files required a value in every column).</p>	

Table 4-6. Configuration > Preferences (Instantiation/Annotation) (cont.)

Field	Description	Values
Ignore system decimal sign setting and use	<p>Ignores regional settings when placing components into DxDesigner.</p> <p>If this option is selected, the decimal point for the component value is shown with a comma (,) in DxDataBook, but placed with a period in DxDesigner.</p> <p>For example, if the component value equals 2,2K (European notation) in DxDataBook, then its value is 2.2K when placed in DxDesigner.</p>	Decimal point in the form of a period (.) or comma (,)

Related Topics

- [Setting Configuration File Preferences](#)
 - [Advanced: Fraction Format](#)
 - [Annotate to Selected Component](#)
 - [Component Annotation](#)
 - [Component Loading](#)
 - [Component Tracking](#)

Component Loading

To access: In the DxDataBook menu, select **Configure > Edit Configuration > Preferences** (**Component Loading** section).

Use this dialog box to set component loading preferences for the DxDataBook configuration file (see [Figure 4-15](#)).

Table 4-7. Configuration > Preferences (Component Loading)

Field	Description	Values
Load Components into New Window	Loads components into a new window instead of the open Search window.	None

Table 4-7. Configuration > Preferences (Component Loading) (cont.)

Field	Description	Values
Symbol Properties Get Loaded	Loads symbol properties when loading a component from DxDesigner. DxDataBook scans the component and symbol properties when loading a component from a DxDesigner schematic, and only adds annotated properties to the component.	None
Use a Property Value for the Tab String	Selects a property type specified when loading a component from DxDesigner. The value DxDesigner uses when instantiating the component with any valid user-defined property is replaced, such as REFDES.	None
Excluded Properties when Loading	Excludes the properties specified when loading a component from DxDesigner.	

Related Topics

- [Setting Configuration File Preferences](#)
 - [Advanced: Fraction Format](#)
 - [Annotate to Selected Component](#)
 - [Component Annotation](#)
 - [Component Instantiation/Annotation](#)
 - [Component Tracking](#)

Component Tracking

To access: In the DxDataBook menu, select **Configure > Edit Configuration > Preferences** (**Component Tracking** section).

Use this dialog box to set component tracking preferences for the DxDataBook configuration file (see [Figure 4-15](#)).

Table 4-8. Configuration > Preferences (Component Tracking)

Field	Description	Values
Check Component Tracking	Highlights the relevant components on the schematic as DxDataBook processes each group.	None
Zoom On Component Tracking	Zooms in on the DxDesigner component when selected in DxDataBook (zoom action dependent on the component tracking preference selected). Note: This functionality is only available after components have been loaded from a DxDesigner schematic into DxDataBook using a load component or verify operation.	None

Related Topics

- [Setting Configuration File Preferences](#)
 - [Advanced: Fraction Format](#)
 - [Annotate to Selected Component](#)
 - [Component Annotation](#)
 - [Component Instantiation/Annotation](#)
 - [Component Loading](#)

Connecting to a DxDataBook Configuration File

The base configuration file (*.dbb* extension) is used to create a centrally defined DxDataBook configuration file, allowing users to add their own libraries. Each user has a personal configuration file (*.dbc* extension) which is linked to the *.dbb* file.



Tip: The *.dbb* file can be stored on a file server, a web server, or an FTP site.

A configuration scheme can be implemented using a *.dbb* file independently of the DxWebPack (see Chapter 3, “[Installing Web Server Software](#)” for DxWebPack information).

Note



In EE2007.5 and newer releases, the *dxdb.ini* file is no longer used to control which *.dbc* file is loaded into DxDataBook (use the *.dbb* file instead — see [Modifying a DxDataBook Configuration File with the DBCtool Utility](#)).

Related Topics

- [Creating a DxDataBook Configuration File](#)
- [Editing a DxDataBook Configuration File](#)
- [Filtering Library Data in the DxDataBook Configuration File](#)

Filtering Library Data in the DxDataBook Configuration File

The *dxdbfilt.ini* file contains filters which are applied to library data described in the configuration file when the libraries are viewed in DxDataBook (once the configuration file is loaded, DxDataBook checks the primary project directory and WDIR path for the *dxdbfilt.ini* file).

These filters can be used to set pre-defined search criteria. For example, a filter can be configured to restrict resistor searches to resistors from a specific supplier or to only show approved components.

The *dxdbfilt.ini* file contains one or more sections that define a particular filter to apply to one or more libraries. The filter section uses the following format (see [Table 4-9](#)):

```
[<FilterName>]  
AppliesTo = <LibraryList>  
Fields = <FieldList>  
Condition = <SQL_expression>
```

Table 4-9. DxDataBook Library Filters

Filter	Description
[<FilterName>]	A user-defined name that describes the filter functionality (name is displayed in the DxDataBook status bar when the filter is active).
AppliesTo = <LibraryList>	A comma-separated and case-sensitive wildcard list of DxDataBook libraries to which the filter should be applied. To apply the filter to all libraries, use the asterisk (*) wildcard symbol.

Table 4-9. DxDataBook Library Filters (cont.)

Filter	Description
Fields = <FieldList>	<p>A list of the database fields (that is, columns in the database tables) used by the Condition value. The list requirements include:</p> <ul style="list-style-type: none">• Arguments must be separated by commas and format is case-sensitive.• Full field names with no wildcards.• Field names must match names used in DxDataBook Library Configuration dialog box (see Editing a DxDataBook Configuration File).
Condition = <SQL_expression>	<p>An SQL-style expression that contains the filter to apply. For example, STATUS='approved'</p>

Note



The filter is only applied to a DxDataBook library if the AppliesTo text field contains the library name and the library contains all of the fields listed in the fields list.



Tip: You can define more than one filter and apply multiple filters to a library.

The following is a *dxdbfilt.ini* file example:

```
[Value Filter]
AppliesTo = capacitors,resistors
Fields = value
Condition = value>2.2E-06

[Approved Filter]
AppliesTo = *
Fields = STATUS
Condition = STATUS='approved'

[Only DIP]
AppliesTo = TTL
Fields = PKG_BIND,APPLICATION
Condition = PKG_BIND='DIP' AND APPLICATION='oem'
```

Related Topics

- [Creating a DxDataBook Configuration File](#)
- [Connecting to a DxDataBook Configuration File](#)
- [Modifying a DxDataBook Configuration File with the DBCtool Utility](#)

Modifying a DxDataBook Configuration File with the DBCtool Utility

The Design Exchange DBCtool utility is used to:

- Create a web server connection from a *.dbc* file.

For example:

```
dbctool /changeserver /oldserver=testserver /newserver=prodserver  
/in=test.dbc /out=prod.dbc
```

- Convert a current *.dbc* file to work with earlier versions of DxDataBook.

For example:

```
dbctool /version=1.2 /in=test.dbc /out=test12.dbc
```

Converts a *.dbc* file from version 1.3 to a version 1.2 file.

- Convert a *.dbc* file into a *.dbb* base configuration file.

For example:

```
dbctool /in=test.dbc /out=test.dbb
```

- Convert custom or user-defined attributes into common (standard) properties.

Options:

- **/convertattrs** — Converts attributes into properties.
- **/mapfile=<mapfile>** — Points to the */convertattrs* *map.cfg* file (if not set, default *map.cfg* file located in installation folder is used).

Note



User-defined property mappings are supported.

For example:

```
dbctool /in=attrbased.dbc /out=propbased.dbc /convertattrs
```

Converts custom attributes specified in the *map.cfg* file into common properties (attributes not listed in the *map.cfg* file are unchanged and appear as specified in the *attrbased.dbc* file).

Procedure: Starting the DBCtool Utility

1. In Windows, select **Start > Run**.
2. In the Run dialog box, type *cmd* and click **OK**.

3. In the MS-DOS command window, type dbctool.

The command usage information is displayed along with examples.

Results

The DBCtool is running in a MS-DOS command window.

Related Topics

- [Creating a DxDataBook Configuration File](#)
- [Editing a DxDataBook Configuration File](#)
- [Connecting to a DxDataBook Configuration File](#)
- [Filtering Library Data in the DxDataBook Configuration File](#)

Chapter 5

Placing and Verifying Schematic Components with DxDataBook

This topic describes how to perform the following DxDataBook tasks:

- [Invoking DxDataBook](#)
- [Editing DxDataBook Properties](#)
- [Working with Database Libraries in DxDataBook](#)
- [Modifying Schematic Components with DxDataBook](#)
- [Verifying Schematic Components with DxDataBook](#)
- [Viewing Component Data Sheets and Web Pages in DxDataBook](#)

To perform these tasks, you must connect DxDataBook to a valid configuration file containing library information (see [Defining Data Sources, Libraries, and Symbols in the Configuration File](#)).

Invoking DxDataBook

You need to invoke DxDataBook in order to access your database library and place components on the DxDesigner schematic.

Prerequisites

- DxDataBook software has been installed and configured, and a data source and configuration file exists on your local system or in a network location.

Note



If the DxDataBook window appears with inactive fields, it is not connected to a data source using a valid configuration file (see Chapters [3](#) and [4](#) for more information).

Procedure

1. In Windows, select **Start > All Programs > Mentor Graphics SDD > Design Entry > DxDesigner**.

Note



You first need to open an existing DxDesigner project or create a new project before starting DxDataBook (see the [DxDesigner User's Guide](#) for project information).


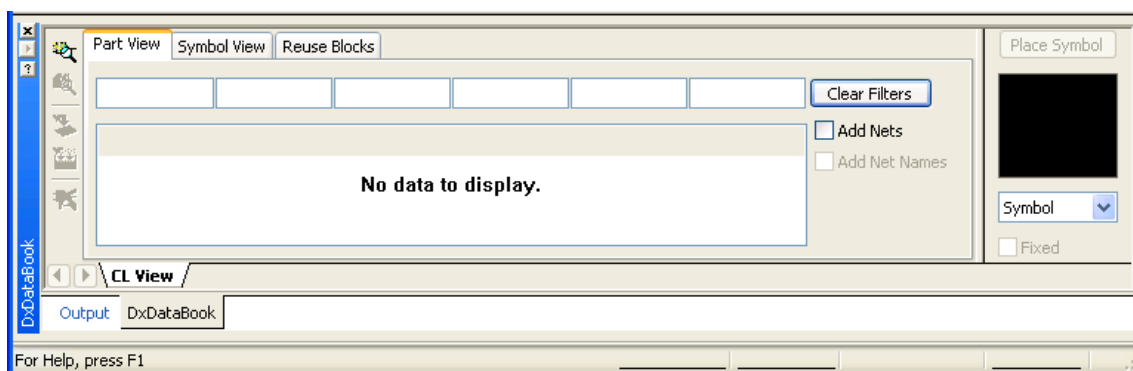
2. In the DxDesigner window, click the DxDataBook button  or select **View > DxDataBook** to open a DxDataBook window in DxDesigner (see [Figure 5-1](#)).

Figure 5-1. DxDataBook Window (CL View — Default)



By default, the DxDataBook window is located at the bottom of the DxDesigner window and the CL View tab is displayed. The CL View (Central Library View) contains the following tabs:

- **Part View** — Items in the **Part View** are parts that contain symbol, cell, and padstack (including pads and hole) data for a package. Parts are normally used to create or edit designs with both logical (electrical) and physical (cell) characteristics.

The **Part View** lists symbols by part number (Expedition workflow only).



Tip: You can use wildcards (*) in the Part View window to filter parts. For example, typing PN12*89* into the Part filter produces a match for the PN1234890ABC and PN12ABC89DEF parts (recommend using a complete search string that ends with a wildcard).

- **Symbol View** — Items in the **Symbol View** are symbols only. There is no cell or padstack data associated with them. Symbols are normally used to create or edit designs with logical (electrical) characteristics only. In addition, you can use symbols as a basis to build and create new parts.



Tip: To automatically update symbol partitions on local symbols, in the Symbol column right-click on the local symbols, select Substitute symbol(s), and click Yes in the message dialog box. The selected local symbols are replaced with library symbols of the same name.

- **Reuse Blocks** — Items in **Reuse Blocks** are entire designs that can be treated as a single component when placed in your project (see “[Placing a Logical-Only Reusable Block in a Host Design](#)” in the *Reusable Blocks Process Guide* for more information).

Reuse Blocks lists reusable blocks in the library (Expedition workflow only).

The right-side of the DxDataBook window displays selected symbols in a viewer. You can place symbols on the schematic by clicking Place Symbol and right-clicking on the schematic. For more information about adding symbols to your design, see “[Adding Symbols from the Central Library \(Expedition Workflow\)](#)” in the *DxDesigner User’s Guide*.

3. To exit DxDataBook, select **File > Exit**.

Results

DxDataBook has been invoked and is available to access your database library and place components on the DxDesigner schematic.

Related Topics

- [Working with Database Libraries in DxDataBook](#)
- [Modifying Schematic Components with DxDataBook](#)
- [Verifying Schematic Components with DxDataBook](#)
- [Viewing Component Data Sheets and Web Pages in DxDataBook](#)

Editing DxDataBook Properties

To access: In the DxDataBook window, **Right-click > Properties** (see [Figure 5-2](#)).

Use this dialog box to edit DxDataBook startup, search, appearance, and color properties (see [Table 5-1](#)).

Note



Changes made through the DxDataBook Properties dialog box are stored in the local registry and only affect the DxDataBook preferences on that system (invoking DxDataBook on a different system might result in different preference settings).121

Figure 5-2. DxDataBook Properties Dialog Box

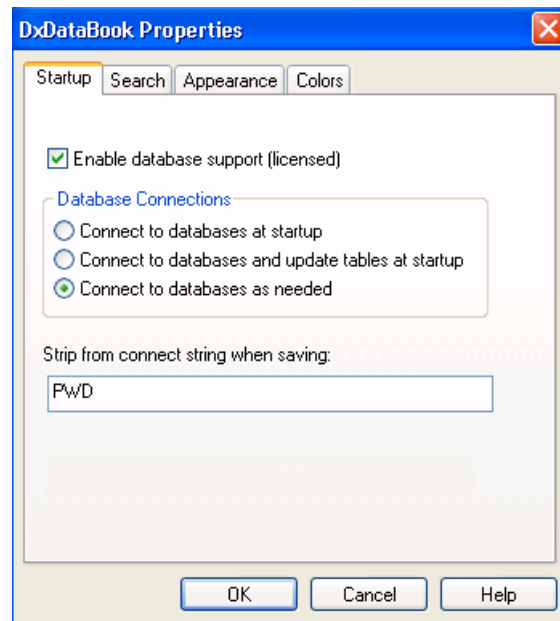


Table 5-1. DxDataBook Properties Dialog Box Contents

Field (Tab)	Description
Startup	<p>If selected (default), the Enable database support (licensed) option allows you to select one of the following Database Connections settings:</p> <ul style="list-style-type: none"> • Connect to databases at startup (default) — DxDataBook establishes a connection to all databases referenced in the configuration file at startup. • Connect to databases and update tables at startup — DxDataBook connects to all databases in the configuration file at startup and checks the databases for new or removed table fields (configuration file is modified to reflect any changes). • Connect to databases as needed — No connections to databases at startup, reducing DxDataBook startup time. The first time you execute a query against a library, DxDataBook connects to the database referenced by the library. <p>Note: To store the database password in the <i>.dbc</i> file so it does not have to be entered upon startup, clear the Strip from Connect String option (information in this box is used for security to strip passwords when creating the configuration file.)</p>

Table 5-1. DxDataBook Properties Dialog Box Contents (cont.)

Field (Tab)	Description
Search	<p>The Display section contains an option to specify whether DxDataBook should display records with identical fields, but different symbols, in the same row or in different rows.</p> <p>In the Component Search window, the Symbol field shows the name of each symbol as a comma separated list. With this box checked, multiple symbol choices appear in the Symbol Preview menu when adding components to a DxDesigner schematic.</p> <p>The Performance section contains settings that limit the size of query results (default is 400) and specifies the amount of time to wait before timing out (default is 60 seconds). When specifying a time out, always enter the value in seconds.</p> <p>Performance section options:</p> <ul style="list-style-type: none"> • Automatically detect bad search criteria — Bad search criteria is detected and additional searches are automatically run when the initial search does not produce any results. • Automatically query libraries when selected — DxDataBook automatically performs a query on the selected library. When cleared, you must first select the library and then click search (!) icon in the search window to get search results. <p>Verification specifies whether or not to ignore composites (that is, eliminates composite symbols from consideration during live or hierarchical verification or verifies all components, including composites).</p>

Table 5-1. DxDataBook Properties Dialog Box Contents (cont.)

Field (Tab)	Description
Appearance	<p>Display numeric values with the magnitudes configured in the DxDataBook configuration file (see Chapter 4, “Editing Library Information”) — Select this option to have DxDataBook convert numeric values and display them using a magnitude suffix (for example, to display a capacitance value of 10 nF). Clear this option to have DxDataBook display the same value, but with the actual numeric value from the database (for example, 10e-9).</p> <p>Auto-size query results columns — Select this option to have DxDataBook automatically adjust the width of each column in the Results pane and show complete cell data. Clear this option to have DxDataBook optimize the size of the data cells, which might require some manual resizing to see all content.</p> <p>Show <All> — First item in the library combo box (default). Clear this option to have <All> as the last item in the library combo box.</p> <p>Display options:</p> <ul style="list-style-type: none"> • Database Field Names — DxDataBook displays the field names from the database table column headings for each library listed in the Search window. • Viewdraw Property Names — DxDataBook displays the DxDesigner property names for each library listed in the Search window. <p>Note: A designer using DxDataBook to capture a schematic generally finds the property names setting more useful than the database field names setting.</p> <p>Tip: Select DxDesigner Property Names when linking multiple tables and for viewing the property names instead of table names.</p>
Colors	<ul style="list-style-type: none"> • Search — Changes search text, highlighted text, link text, and background colors. • Verify — Changes verify text and background. • Restore Defaults — Restores the colors to their original settings. <p>Each color property menu contains the following option:</p> <ul style="list-style-type: none"> • Other — Selects a color different from those available in the menu color palettes

Working with Database Libraries in DxDataBook

The following DxDataBook database library topics are described:

- [Opening a Database Library](#)
- [Searching for Components in a Database Library](#)

Related Topics

- [Invoking DxDataBook](#)
- [Modifying Schematic Components with DxDataBook](#)
- [Verifying Schematic Components with DxDataBook](#)
- [Viewing Component Data Sheets and Web Pages in DxDataBook](#)

Opening a Database Library


To place database library components on your DxDesigner schematic, you must first open a database library.

Note



Use the DxDataBook sample database provided for this topic.

Procedure

1. In Windows, select **Start > All Programs > Mentor Graphics SDD > Design Entry > DxDesigner**.
2. In the DxDesigner window, select **File > Open > Project**.
3. In the Open dialog box, navigate to `<install_dir>\wv\tutor\Designer\Designer.dproj` and click Open.
4. In the DxDesigner window, click the DxDataBook button  or select **View > DxDataBook**.



Tip: You can also use Ctrl-Alt-Delete (shortcut key) to open the DxDataBook window.

5. In the DxDataBook window, right-click and select **Configure > Open**.
6. In the Open dialog box, navigate to `<install_dir>\dx\tutor\dxdb\samples\config` and select the *sample.dbcs* file.

7. Click Open.
8. In the Project Navigator, expand the Design Roots node and click Sheet 1 to open the sample schematic.

Results

The database library is open and you can search for schematic components.

Related Topics

- [Working with Database Libraries in DxDataBook](#)
 - [Searching for Components in a Database Library](#)

Searching for Components in a Database Library

DxDataBook allows you to search for and select components from an ODBC-compliant database library located on your company network (for example, on the local system, a shared drive, or a server), company intranet, or the Internet.

You can also add conditions to the search query to narrow the search to components that meet specific requirements. The search results display a list of potential component candidates and DxDataBook is used to place a component and transfer the property information to the DxDesigner schematic, as described in [Modifying Schematic Components with DxDataBook](#).

The following database library search topics are described:

- [Adding Search Conditions](#)
- [Creating Search Queries](#)
- [Performing a Search](#)

Related Topics

- [Working with Database Libraries in DxDataBook](#)
 - [Opening a Database Library](#)

Adding Search Conditions

When searching for components in a database library, you can add conditions that must be met by components displayed in the results list to help filter your search.

Procedure

1. In the DxDataBook window, click the operator located under the appropriate column name (for example, Tolerance) and select an operator from the dropdown list (see [Table 5-2](#)).

Table 5-2. Operators and Data Types

Operator	Applicable Data Type
=	Numerical or string (default operator)
>	Numerical
>=	Numerical
<	Numerical
<=	Numerical
!=	Numerical or string
like	String

2. In the field to the right of the operator, click and use one of the following methods to specify a value (see [Figure 5-3](#)):
 - Click the arrow to select an existing value from the dropdown list.
 For example, if you select the Tolerance column in a Resistors library, the dropdown list shows all Tolerance values that exist in the library.
 - Type a value into the text field.
 For example, typing *R12** into the PKG_TYPE column text field generates a new results list with components containing both “R1206” and “R1210” as PKG_TYPE values.
 You can also add search conditions by selecting a component in the search results list, right-click on a property to be used as a condition, and selecting the Add Condition option.

Note



You can also use a wildcard when typing a value (see [Wildcard Searches](#)).

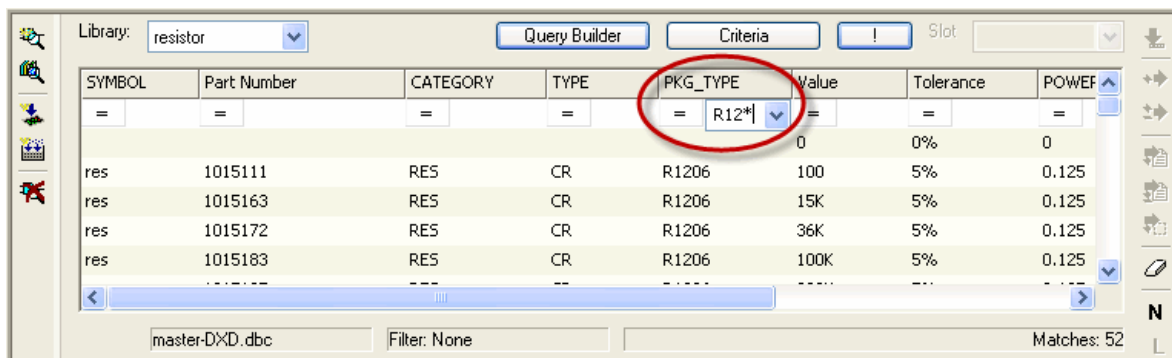
3. Set additional search conditions as needed to further filter search resultsR12.

For example:

- Resistors Part Label value matches “R-100%” AND
- Resistors Rating value is “0.125” AND
- Resistors Tolerance value matches”0.1”

Only one resistor (R10-0805) in the library matches all three search conditions.

Figure 5-3. Multiple Search Conditions



4. To view or remove the DxDataBook search conditions:
 - Click Criteria to view the search conditions in a dialog box.
 - In the Field list, select the property to remove and click the hyphen (-) button.

You can also right-click the column containing the search condition and select the Remove Condition option (or clear all search conditions by selecting Remove All Conditions).

Results

Search conditions have been set to filter search results for components in a database library.

Related Topics

- [Searching for Components in a Database Library](#)
 - [Creating Search Queries](#)
 - [Performing a Search](#)

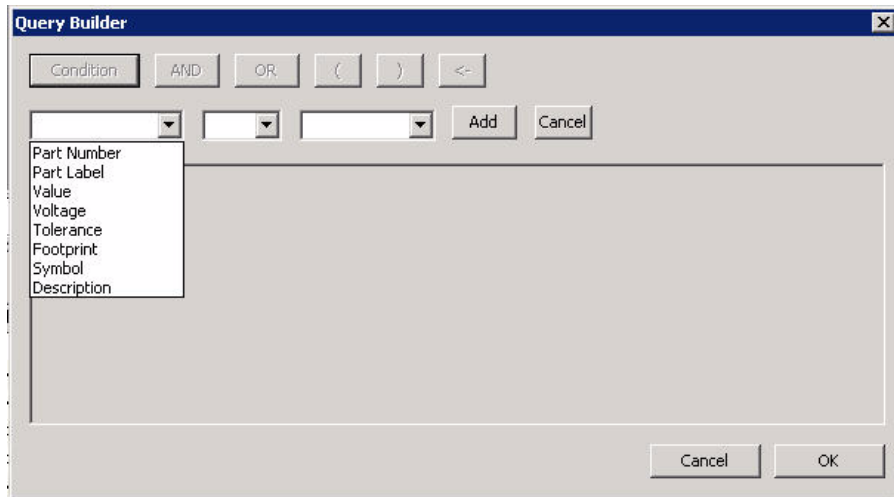
Creating Search Queries

DxDataBook allows you to create database library search queries.

Procedure

1. In the DxDataBook window, click Query Builder to display the Query Builder dialog box (see Figure 5-4).

Figure 5-4. Query Builder Dialog Box



2. In the Query Builder dialog box, click Condition to construct a search condition for the query.

3. In the first text field, click the arrow to select a property from the dropdown list.

The library selected in the DxDataBook window **Library** menu determines which properties are displayed in the list.

4. In the second text field, click the arrow to select an operator for the search condition (see [Table 5-3](#)).

5. In the third text field, use one of the following methods to specify a value:

- Click the arrow to select an existing value from the dropdown list.
- Type a value into the text field.

You can also use a wildcard when typing a value (see [Wildcard Searches](#)).

6. Click Add to save the query for the search condition.
7. Continue to add search condition queries by choosing the AND or OR boolean operators.

Use the parenthesis () buttons to establish the execution order for the conditions and the arrow (<-) button to remove the last input entered into the query.

The AND operator indicates that the query only finds components that meet all specified conditions whereas the OR operator-based query finds components that meet any of the conditions.

8. Click **OK**.

The search results appear in the DxDataBook Search Results section.

Results

You have created DxDataBook database library search queries.

Related Topics

- [Searching for Components in a Database Library](#)
 - [Adding Search Conditions](#)
 - [Performing a Search](#)

Performing a Search

DxDataBook allows you to search for components in the database libraries, and supports using numeric comparison operators (see [Numeric Comparison Character Searches](#)) and string wildcards (see [Wildcard Searches](#)) in your searches.

Procedure

1. In the DxDataBook window (see [Figure 5-5](#)), connect to the DxDataBook configuration file that references the data source you are searching (see Chapter 4, “[Connecting to a DxDataBook Configuration File](#)”).


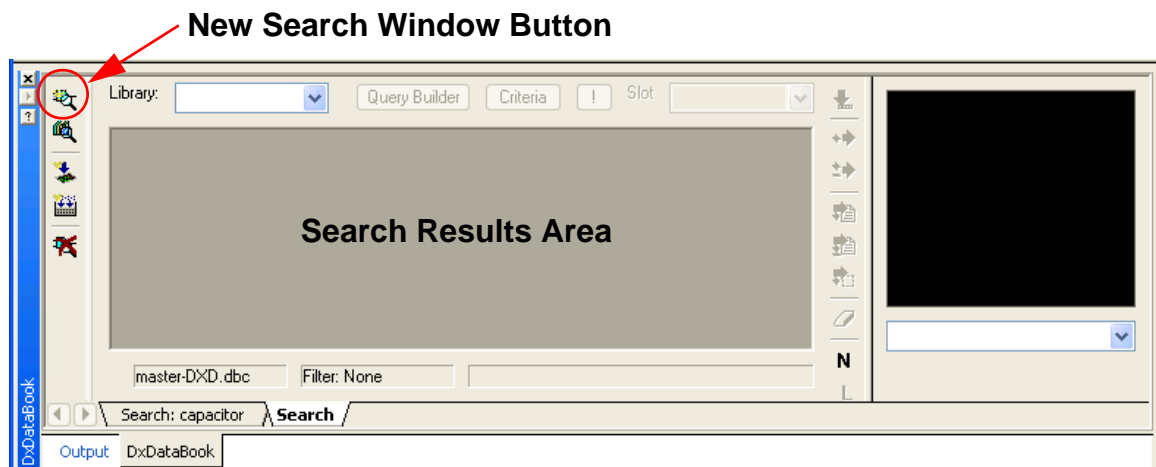
By default, DxDataBook opens a Search window when you initially connect to a configuration file (click the New Search Window button  to open a new Search window). To open a different configuration file or set up a new DxDataBook configuration file, see Chapter 4, “[Defining Data Sources, Libraries, and Symbols in the Configuration File](#).”

Figure 5-5. DxDataBook Search Window



2. From the **Library** menu, select a library from the list of available libraries.

Selecting <All> searches all libraries in the DxDataBook configuration file in succession (after choosing <All>, you must click the exclamation point (!) button to perform each successive search).

By default, when you select a library, DxDataBook returns a list of all components in that library to the search results area. However, you can change this default behavior by clearing the Automatically query libraries when selected option in the **Search** tab of the DxDataBook properties dialog box (see [Editing DxDataBook Properties](#)).

3. To clear the search result list, click the right-hand eraser button in the DxDataBook window.

You can also perform a “reverse search” by selecting a component on the DxDesigner schematic sheet and **Right-click > Load into DxDataBook** (component record is loaded into the DxDataBook window).

Notes:

- If the automatic querying is turned off, click the (!) button to see search results.
- If the search results exceed the display limit (search results shown in red text), select **Properties > Search > Query Results Limit** to change the number of results to display (default setting is display 400 search results).

Results

The database library has been searched for components and the results are displayed in the DxDataBook Search window.

Related Topics

- [Searching for Components in a Database Library](#)
 - [Adding Search Conditions](#)
 - [Creating Search Queries](#)

Numeric Comparison Character Searches

DxDataBook supports using numeric comparison characters (see [Table 5-3](#)) in your database library search conditions.

Table 5-3. Numeric Comparison Operators

Operator	Function	Example
=	Represents values equal to the specified numeric value.	The = 20 operator returns a list of components for which the numeric value of the specified property equals 20.

Table 5-3. Numeric Comparison Operators (cont.)

Operator	Function	Example
>	Represents values greater than the specified numeric value.	The >20 operator matches all numeric values of the specified property greater than 20.
<	Represents values less than the specified numeric value.	The < 1.25 operator matches all numeric values of the specified property less than 1.25.
>=	Represents values greater than or equal to the specified numeric value.	The >= 32 operator matches all numeric values of the specified property greater than or equal to 32.
<=	Represents values less than or equal to the specified numeric value.	The <= 0.05 operator matches all numeric values of the specified property less than or equal to 0.05.
!=	Represents values not equal to the specified value.	The != 34 operator matches all numeric values of the specified property except those that equal 34.

To use the numeric comparison characters in a search condition, select a Field value from the dropdown list and then select an operator.

Related Topics

- [Performing a Search](#)
 - [Wildcard Searches](#)

Wildcard Searches

DxDataBook supports using wildcards in your database library search conditions. By default, DxDataBook uses the standard SQL percent (%) wildcard to match multiple string characters and underscore (_) wildcard to match an individual character.

Note



To search for components using wildcards, the database is required to contain string data and the “like” operator must be used.

The following wildcard characters are supported:

- Percent sign (%) which represents any of zero or more (multiple) characters.

For example:

- **string%** — Matches any value beginning with *string*.
- **%string** — Matches any value ending with *string*.
- Underscore (**_**) which represents any individual character.

For example:

- **string_** — Matches *string* followed by one character.
- **_string** — Matches any two characters followed by *string*.

The question mark (?) and asterisk (*) wildcard symbols can also be used in a database library search if you specified wildcards should be allowed when first connecting to the database (see Chapter 3, “[Setting Up and Configuring the DMS Connector](#)”). The question mark (?) wildcard is used to match a single character and the asterisk (*) wildcard to match multiple characters.

For example, using RESISTO? as a search string only yields RESISTOR whereas RESISTO* yields a list of all parts in the database that include the RESISTO string (such as RESISTOR and RESISTOR123).

Related Topics

- [Performing a Search](#)
 - [Numeric Comparison Character Searches](#)

Modifying Schematic Components with DxDataBook

DxDataBook allows you to add and annotate schematic components using the following commands:

- **Add** — Adds a generic or unique version of the component to the schematic (see [Adding a Schematic Component](#)).
- **Annotate** — Annotates a component already on a schematic with property information from a library database (see [Annotating a Schematic Component](#)).

A system administrator often adds a placeholder property to specify its location, visibility, and other characteristics. The placeholder property is also given a default value that can be replaced by a value from the database library (use Add or Annotate). Any new properties do not have a placeholder location or symbol value and DxDataBook adds the information using the configuration file. If undefined, the default location for a new property is at coordinates 0,0 and extends below the point of origin, and the properties are not visible (see [Setting Schematic Component Property Visibility](#)).

Related Topics

- [Invoking DxDataBook](#)
- [Working with Database Libraries in DxDataBook](#)
- [Verifying Schematic Components with DxDataBook](#)
- [Viewing Component Data Sheets and Web Pages in DxDataBook](#)

Adding a Schematic Component

After completing a component search in your database library, you can add generic (see [Adding a Generic Component to a Schematic](#)) or unique (see [Adding a Unique Component to a Schematic](#)) versions of the DxDataBook components to the DxDesigner schematic.

Related Topics

- [Modifying Schematic Components with DxDataBook](#)
 - [Annotating a Schematic Component](#)
 - [Setting Schematic Component Property Visibility](#)

Adding a Generic Component to a Schematic

The Add Generic command adds a generic DxDataBook component to a DxDesigner schematic, and DxDataBook only transfers the property names and values that are common to all components displayed in the search results.

A generic component generally does not point to a specific manufacturer, use a unique technology or package, or have all the parametric information needed to associate the component to a unique part. In addition, a generic component on a DxDesigner schematic can be replaced by corresponding unique components.



For example, searching a resistor library returns a list of resistors where the pwrRating and Value properties are the only ones with same values across the search results (that is, the search results in a value of *100* and a power rating of *0.25* for all entries). Adding a generic resistor to the DxDesigner schematic results in a 100 ohm resistor (Value=100) with a power rating of 0.25 (pwrRating=0.25), and all placeholder properties retain default values on the resistor symbol.

Prerequisites

- [Searching for Components in a Database Library](#)

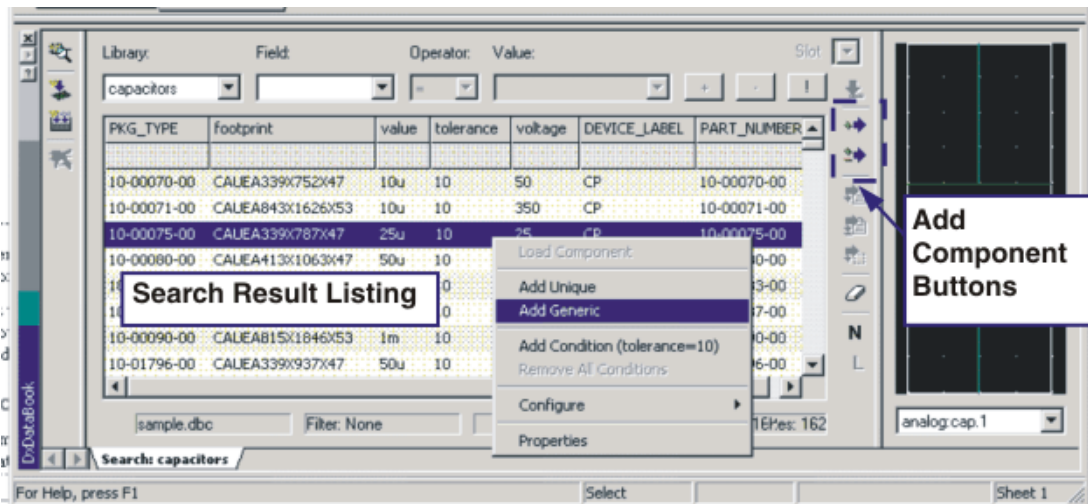
Procedure

1. In the DxDesigner window, open the schematic sheet.

2. In the DxDataBook window, verify that you are connected to the DxDataBook configuration file that references the data source to search (see Chapter 4, “[Connecting to a DxDataBook Configuration File](#)”).
3. [Optional] Click the New Search Window button  to open a new Search window.
4. In the DxDataBook window, perform a search to generate a list of qualified components (see [Searching for Components in a Database Library](#)).
5. Select a row containing a component and click the Add New Component With Common Properties button  or **Right-click > Add Generic** to add the generic component to the schematic (see [Figure 5-6](#)).

You can also add net stubs and labels when instantiating the symbol on the schematic by clicking the “N” or “L” buttons located on the right side of the DxDataBook search list. Net labeling is only enabled if the Add Net Stub function is active.

Figure 5-6. Adding a Generic Component



6. In the DxDesigner schematic, place your cursor on the desired location and left-click to place the component.

To display the generic properties of the component:

- Single-click the component on the schematic to display the properties in the DxDesigner Properties window.
- Double-click the component on the schematic to display the properties in the Component Properties dialog box (**Properties** tab).

Results

A generic version of a DxDataBook component is added to the DxDesigner schematic.

Related Topics

- [Adding a Schematic Component](#)
 - [Adding a Unique Component to a Schematic](#)


Adding a Unique Component to a Schematic

The Add Unique command adds a unique DxDataBook component to a DxDesigner schematic. A unique component represents an actual physical device that has a company and manufacturer part number, and is supplied by a specific supplier, implemented by a particular technology, and uses a specific package. The unique component contains all of the property names and values from a selected row in the DxDataBook Search window and the properties are transferred when the component is placed on the schematic. A unique match is then created between the component on the schematic and a specific component in a DxDesigner library.

Prerequisites

- [Searching for Components in a Database Library](#)

Procedure


1. In the DxDesigner window, open the schematic sheet.
2. In the DxDataBook window, verify that you are connected to the DxDataBook configuration file that references the data source to search (see Chapter 4, “[Connecting to a DxDataBook Configuration File](#)”).
3. [Optional] Click the New Search Window button  to open a new Search window.
4. In the DxDataBook window, perform a search to generate a list of qualified components (see [Searching for Components in a Database Library](#)).
5. In the search list, select a row containing a unique component.
6. [Optional] If the component has more than one symbol, use the **Symbol Preview** menu to select the desired component symbol.
7. If the component has more than one cell, you can select a different cell to associate with the component. When you select a cell for the component, the Cell Name property associated with that component assumes the value of the alternate cell name.

Note



Not all components are associated with alternate cells (see “[Specifying Alternate Cells](#)” in the *DxDesigner User’s Guide*).

8. Add a unique component to the DxDesigner schematic using one of the following methods:

- Drag and drop the symbol from the Symbol Preview window into the schematic.
- Click the Add New Component With Common Properties button  or **Right-click > Add Unique**.

You can also add net stubs and labels when instantiating the symbol on the schematic by clicking the “N” or “L” buttons located on the right side of the DxDataBook search list. Net labeling is only enabled if the Add Net Stub function is active.

9. In the DxDesigner schematic, place your cursor on the desired location and left-click to place the component.

To display the unique properties of the component:

- Click the component on the schematic to display the properties in the DxDesigner Properties window.
- Double-click the component on the schematic to display the properties in the Component Properties dialog box (**Properties** tab).

Results

A unique version of a DxDataBook component is added to the DxDesigner schematic.

Related Topics

- [Adding a Schematic Component](#)
 - [Adding a Generic Component to a Schematic](#)

Annotating a Schematic Component

Loading a component from DxDesigner into DxDataBook uses the existing property values on the schematic to create a search filter. This search filter can be used to find replacement components from a DxDataBook library.

For example, if a generic version of 10 volt, 1 kilohm resistor is placed on a schematic, and then selected and loaded into DxDataBook, the search results list contains matching 10 volt, 1 kilohm resistors from the DxDataBook resistor library.

Once a component is loaded into DxDataBook, the Annotate command is used to annotate the component located in DxDesigner with generic (common) or unique (all) property information from your DxDataBook library.

The following commands are used to annotate generic and unique components:

- **Annotate Generic** — Only transfers property names and values that are common to all candidate components displayed in the search results to the component or components selected in DxDesigner.
- **Annotate Unique** — Transfers all property names and values associated with a specific component selected from the search result list.

Prerequisites

- [Searching for Components in a Database Library](#)


Procedure

1. In the DxDesigner window, open the schematic sheet.

Note



You can select multiple components on different schematic sheets and hierarchy levels, even if the components are not loaded into DxDataBook, and also simultaneously change properties for multiple objects.

2. In the DxDataBook window, verify that you are connected to the DxDataBook configuration file that references the data source to search (see Chapter 4, “[Connecting to a DxDataBook Configuration File](#)”).
3. Click the Add New Component With Common Properties button  or **Right-click > Load Component**.

Note



The **Library** menu is disabled when Load Component is selected to prevent accidental selection of an inappropriate library for the selected component (DxDataBook uses the DXDB_LIBNAME property to determine the library of origin for the component or components selected in DxDesigner).

Based on the settings selected, the list of matching components are either loaded into the existing DxDataBook Search window or in a new Search window. As component information is loaded, DxDataBook displays the search conditions used.

In the Expedition design flow, you have the option to retrieve pin numbers from the parts database (PDB) for the component (Get Pin Numbers from PDB option).

4. [Optional] You can apply additional search conditions to further refine the list of matching components that appear in DxDataBook.
5. Annotate the component or components selected in DxDesigner with generic or unique property information:

- Click the Annotate Component with Common Properties button or right click > **Annotate Generic** to apply the properties common to the components displayed in the search results list to the components selected in DxDesigner.
- Select a component in the search results list and click Annotate Component with All Properties or right click > **Annotate Unique** to apply all of the property names and values to the components selected in DxDesigner.

Results

The components located in DxDesigner are annotated with generic or unique property information from your DxDataBook database library.

Related Topics

- [Modifying Schematic Components with DxDataBook](#)
 - [Adding a Schematic Component](#)
 - [Setting Schematic Component Property Visibility](#)

Setting Schematic Component Property Visibility

New properties are defined as properties backannotated to a component on the schematic from DxDataBook, and these properties do not already exist on a symbol or component.

Note



Mentor Graphics does not recommend that designers edit the DxDataBook configuration file to control new property visibility (contact your system administrator and request that a placeholder property value be added to a symbol).

Procedure

To set visibility for new properties added from DxDataBook to the schematic:

1. In the DxDesigner window, select **Project > Settings** to display the Project Settings dialog box.
2. Select **Properties (tab) > Display Properties**.

Note



New property visibility is defined when the DxDataBook configuration file is created, in accordance with corporate specifications or library strategy (see Chapter 4, “[Defining Data Sources, Libraries, and Symbols in the Configuration File](#)”).

To set visibility for individual properties that are backannotated to the schematic from DxDataBook:

1. In the DxDataBook window, right-click and select **Configure > Edit Configuration > Libraries** (tab).
2. Select the desired properties to make visible on the schematic when adding or annotating a component (only the property names or values are visible).

For more information about editing configuration file library information, see Chapter 4, “[Editing Library Information](#).”

Results

Visibility for new properties added from DxDataBook to the DxDesigner schematic is set.

Related Topics

- [Modifying Schematic Components with DxDataBook](#)
 - [Adding a Schematic Component](#)
 - [Annotating a Schematic Component](#)

Verifying Schematic Components with DxDataBook

The DxDataBook verification functionality ensures that there is a one-to-one correspondence between the component property information in the DxDataBook database and the current schematic design (that is, the correct database part data is also present in your design).

There are several DxDataBook verification methods:

- **Component (Live) Verification** — Verify individual, multiple, or all components for the currently active schematic sheet (see [Verifying Components on a Single Schematic Sheet](#)).
- **Hierarchical Verification** — Verify an entire design that has more than one schematic sheet (see [Verifying a Design with Multiple Schematic Sheets](#)).

Related Topics

- [Invoking DxDataBook](#)
- [Working with Database Libraries in DxDataBook](#)
- [Modifying Schematic Components with DxDataBook](#)
- [Viewing Component Data Sheets and Web Pages in DxDataBook](#)

Verifying Components on a Single Schematic Sheet

The Component (Live) Verification command allows you to verify selected DxDesigner components on a single schematic sheet against a component database.

Procedure

1. In the DxDesigner window, open a schematic sheet.
2. In the DxDataBook window, verify that you are connected to the DxDataBook configuration file that references the data source to search (see Chapter 4, “[Connecting to a DxDataBook Configuration File](#)”).
3. In the schematic sheet, select one or more components for verification.

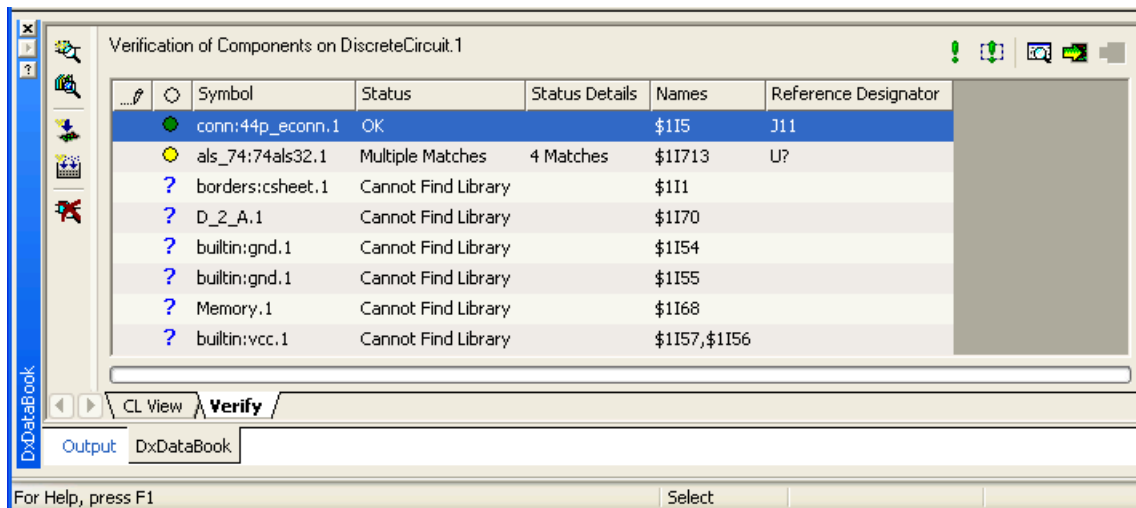
Note



If none of the schematic components are selected, DxDataBook automatically verifies all components on the sheet.

4. In the DxDataBook window, click the New Live Verification Window button .

Figure 5-7. New Live Verification Window



5. In the Live Verification window (see [Figure 5-7](#)), do one of the following:
 - To verify selected components on a schematic sheet, click the Verify All Selected Components button or **Right-click > Verify Selected Components**.
 - To verify all components on a schematic sheet, click Verify All Components or **Right-click > Verify All Components on Sheet**.

The schematic components are displayed in the Verify pane of the DxDataBook Verification window. For each component, the symbol name, status, status details, and

handle denoting the schematic location in which a component appears is displayed. Verification status for each component is indicated by the following icons:

- **Green icon** — Schematic component is correctly matched in the database (*OK* status).
- **Green icon encircled by a black arrow** — Schematic component failed verification, but was still evaluated (annotation of one or more properties to the component is pending).
- **Yellow icon** — Component has multiple matches in the database or component has missing properties (“Multiple Matches” or “Missing Properties” status).
- **Red icon** — Component does not appear in the database (“Zero Matches” status).
- **Question mark** — Component does not have a matching library given the data on the component and the current DxDataBook configuration file.

A question mark may be displayed for any of the following reasons:

- The component was not added to the design using DxDataBook.
- The library in the current DxDataBook configuration file was removed or renamed.
- The component was placed on the design while connected to a different DxDataBook configuration file.

Note



Hierarchical components are generally assigned a question mark when verifying a single schematic sheet.



Tip: Click a row in the Verify pane to highlight the component location in the currently active schematic.

6. In the DxDataBook Verify pane, do one of the following to verify matches in the database:
 - Double-click a component row.
 - Select a component and click Search for Component.
 - **Right-click > Search.**

DxDataBook displays database matches for the component in the Verification window Search pane, and graphical representations in the Symbol Preview pane. The component information displayed includes the configuration file name, and the number of filters (if any) and database matches. The pencil icon shown in the Verify pane indicates that the selected component is displayed in the Search pane.

7. For matching components, click Annotate Generic, Annotate Unique, or Update to write changes for a component back to the schematic (see [Annotating a Schematic Component](#)).

Results

Selected DxDesigner components have been verified on a single schematic sheet against a component database, and are now available for backannotation to the schematic.

Related Topics

- [Verifying Schematic Components with DxDataBook](#)
 - [Verifying a Design with Multiple Schematic Sheets](#)

Verifying a Design with Multiple Schematic Sheets

A flat design contains multiple schematic sheets that exist on the same hierarchical level, and it does not have composite symbols with underlying schematics on these sheets. A hierarchical design contains composite symbols and underlying schematics exist below each composite symbol.

Note



Hierarchical designs may have properties (such as values for passive components) that are only applied when a schematic is opened by traversing through the hierarchy starting at the top level. This allows a group of circuitry to be instantiated as a block multiple times while still having unique reference designators.

The New Hierarchical Verification Window command allows you to verify any multi-sheet design, whether it is flat or hierarchical.

Procedure

1. In the DxDesigner window, open a design.
2. In the DxDataBook window, verify that you are connected to the DxDataBook configuration file that references the data source to search (see Chapter 4, “[Connecting to a DxDataBook Configuration File](#)”).
3. In the DxDataBook window, click the New Hierarchical Verification Window button  .
4. In the Hierarchical Verification window, click the Verify All Selected Components in Current Design button or **Right-click > Verify All Components in Design**.

The schematic components are displayed in the Verify pane of the DxDataBook Hierarchical Verification window. For each component, the symbol name, status, status

details, and handle denoting the schematic location in which a component appears is displayed.



Tip: To see the location of the symbol on a schematic, highlight the component row in the Verify pane and right-click to select the Visit option (action highlights component in schematic).

5. In the DxDataBook Verify pane, do one of the following to verify matches in the database:

- Double-click a component row.
- Select a component and click Search for Component.
- **Right-click > Search.**

DxDataBook displays database matches for the component in the Verification window Search pane, and graphical representations in the Symbol Preview pane. The component information displayed includes the configuration file name, and the number of filters (if any) and database matches. The pencil icon shown in the Verify pane indicates that the selected component is displayed in the Search pane.

6. [Optional] If the component has more than one symbol, use the **Symbol Preview** menu to select the desired component symbol.
7. In the DxDataBook Search pane, select a component and click the Add Part to Pending List option.


DxDataBook updates a buffer with information from the database about the component and also updates the symbol status icon with a circle to indicate the component contains pending changes.

8. For hierarchical designs, click Update Design after updating all of the components to apply the changes.
9. In the Apply Changes to Design dialog box, click Apply Changes.

DxDataBook applies the Back-Annotation File (.baf extension) to the selected schematic and automatically reruns the verification, which should result in green status icons for all components in the Verify pane (see [Verifying Components on a Single Schematic Sheet](#) for status icon information).



Tip: To view .baf file details in Microsoft Notepad before applying changes, click **View BAF File** in the Apply Changes to Design dialog box.

 **Note** In a database that contains numeric extensions for the different graphical representations (for example, 74act00.1), the status icons never change to a green color because verification includes the data in the Symbol column which always results in a “Multiple Matches” status.

Results


DxDesigner components have been verified in a flat or hierarchical multi-sheet design.

Related Topics

- [Verifying Schematic Components with DxDataBook](#)
 - [Verifying Components on a Single Schematic Sheet](#)

Viewing Component Data Sheets and Web Pages in DxDataBook

You can use DxDataBook to view component-related information in a web page (see [Viewing a Web Page](#)) or data sheet (see [Viewing a Data Sheet](#)).

 **Note** To view web pages and data sheets, your system administrator first needs to add WebPage and Datasheet to the component database library.

Related Topics

- [Invoking DxDataBook](#)
- [Working with Database Libraries in DxDataBook](#)
- [Modifying Schematic Components with DxDataBook](#)
- [Verifying Schematic Components with DxDataBook](#)

Viewing a Data Sheet

DxDataBook allows you to view component-related information in a data sheet.

Procedure

1. In the DxDataBook window, perform a search to create a list of components (see [Searching for Components in a Database Library](#)).

2. In the DxDataBook window search results list, select a row containing the component to open in a data sheet.
3. In the DataSheet column, click the link for the selected component.

DxDataBook downloads the selected component file and opens it with the appropriate program.

Results

You have opened component-related information in a data sheet using DxDataBook.

Related Topics

- [Viewing Component Data Sheets and Web Pages in DxDataBook](#)
 - [Viewing a Web Page](#)

Viewing a Web Page

DxDataBook allows you to view component-related information in a web page.

Procedure

1. In the DxDataBook window, perform a search to create a list of components (see [Searching for Components in a Database Library](#)).
2. In the DxDataBook window search results list, select a row containing the component to open in a web page.
3. In the WebPage column, click the web site address (URL) link for the selected component.

The component web page opens in your web browser.

Results

You have opened component-related information in a web page using DxDataBook.

Related Topics

- [Viewing Component Data Sheets and Web Pages in DxDataBook](#)
 - [Viewing a Data Sheet](#)

Chapter 6

Scripting with DxDataBook

Scripts can be created to automate DxDataBook tasks, such as manipulating design objects, menus, and system responses. When scripts are used, DxDataBook components become objects that you control programmatically (that is, scripts are event handlers and executing a certain event causes DxDataBook to call the script).

Note



DxDataBook scripting does not support C++ or dual interfaces.

Related Topics

- [DxDataBook Objects](#)
- [DxDataBook Scripting Events](#)
- [Loading a DxDataBook Script](#)

DxDataBook Objects

To create a script to manipulate DxDataBook objects, you need to know which objects to use and the relationship between these objects. The DxDataBook object model consists of objects and their associated properties, methods, and events.

DxDataBook objects are organized into the following types:

- [Properties Object](#)
- [Component Object](#)
- [Properties Object](#)

Related Topics

- [DxDataBook Scripting Events](#)
- [Loading a DxDataBook Script](#)

Attributes Object

The attributes object is a collection of the properties added to the component from DxDataBook (see [Table 6-1](#)).

Table 6-1. Attributes Object Properties

Property	Description
Name	Applies to the attributes object and is a String type (Name property sets/returns the attribute). Syntax: <i>object.Name</i> Example: For each attr in obj.Attributes attrNames= attrNames & attr.Name
Value	Applies to the attributes object and is a String type (Value property sets/returns the attribute). Syntax: <i>object.Value</i> Example: MsgBox Attributes ("Name") [.Value]
NameVisible	Applies to the attributes object and is a Boolean type (sets/returns Visibility setting). Syntax: <i>object.NameVisible</i>
ValueVisible	Applies to the attributes object and is a Boolean type (sets/returns Visibility setting). Syntax: <i>object.ValueVisible</i>

Related Topics

- [DxDataBook Objects](#)
 - [Component Object](#)
 - [Properties Object](#)

Component Object

The component object represents a collection of components loaded into or from DxDesigner, and consists of a single “Property” (see [Count Property](#)) and “Method” (see [Item Method](#)). The component object has a Symbol, Name, Properties, and Library property (see [Table 6-2](#)).

Table 6-2. Component Object Properties

Property	Description
Symbol	Applies to the component object and is a String type. Syntax: <i>object.Symbol</i> Example: MsgBox obj.Symbol

Table 6-2. Component Object Properties (cont.)

Property	Description
Name	Applies to the property object and is a String type. Syntax: <i>object.Name</i> Example: For each attr in obj.Properties attrNames=attrNames & attr.Name & " "
Properties	Applies to the component object and is a Properties type. Syntax: <i>object.Properties</i> Example: attribs=obj.Properties name=obj.Properties("Name")
Library	Applies to the component object and is a String type. Syntax: <i>object.Library</i>

Related Topics

- [DxDataBook Objects](#)
 - [Properties Object](#)
 - [Properties Object](#)

Properties Object

The properties object is a collection of properties, and it consists of a Count property (see [Table 6-3](#)) and Item, Add, and Remove methods (see [Table 6-4](#)). The properties object also includes:

- [Properties Object Types](#)
- [Properties Object Methods](#)

Note



Property names are case-sensitive.

Table 6-3. Properties Object Property

Property	Description
Count	Applies to the component and attributes object, and is a Long type. Syntax: <i>object.Count</i> Parameter: <i>object</i> (object expression that evaluates component or attributes objects)

Table 6-4. Properties Object Methods

Method	Description
Item	Applies to a components or attributes object (default method). Syntax: <i>object.Item arg</i> or <i>object arg</i> Parameters: <i>arg</i> (either a zero-based index of a property or the Name of a property)
Add	Applies to an attributes object where <i>name</i> is the attribute name, <i>value</i> is the attribute value, and <i>nameVisible</i> , <i>valueVisible</i> and <i>isOat</i> are Boolean types. Syntax: <i>object.Add name, value, nameVisible, valueVisible, and isOat</i>
Remove	Applies to an attributes object. Syntax: <i>object.Remove arg</i> Parameters: <i>arg</i> (either an index or Name of a property) Example: <code>Objs.Remove (3)</code>

Related Topics

- [DxDataBook Objects](#)
 - [Properties Object](#)
 - [Component Object](#)

Properties Object Types

Object properties are characteristics of DxDataBook objects and consist of the following types:

- [Attributes Property](#)
- [Count Property](#)
- [Library Property](#)
- [Name Property](#)
- [NameVisible Property](#)
- [Symbol Property](#)
- [Value Property](#)
- [ValueVisible Property](#)

Related Topics

- [Properties Object](#)
 - [Properties Object Methods](#)

Attributes Property

The Attributes property has the following characteristics:

- Applies to the Component object and is a Collection type.
- Syntax: *object.Attributes*
- Example:

```
attribs=obj.Attributes name=obj.Attributes("Name")
```

Related Topics

- [Properties Object Types](#)
 - [Count Property](#)
 - [Library Property](#)
 - [Name Property](#)
 - [NameVisible Property](#)
 - [Symbol Property](#)
 - [Value Property](#)
 - [ValueVisible Property](#)

Count Property

The Count property has the following characteristics:

- Applies to the Component and Attributes objects, and is a Long type.
- Syntax: *object.Count*
- Parameters: *object* (expression which evaluates Component and Attributes objects).

Related Topics

- [Properties Object Types](#)
 - [Attributes Property](#)
 - [Library Property](#)

- [Name Property](#)
- [NameVisible Property](#)
- [Symbol Property](#)
- [Value Property](#)
- [ValueVisible Property](#)

Library Property

The Library property has the following characteristics:

- Applies to the Component object and is a String type.
- Syntax: *object.Library*

Related Topics

- [Properties Object Types](#)
 - [Attributes Property](#)
 - [Count Property](#)
 - [Name Property](#)
 - [NameVisible Property](#)
 - [Symbol Property](#)
 - [Value Property](#)
 - [ValueVisible Property](#)

Name Property

The Name property has the following characteristics:

- Applies to the Attributes object and is a String type.
- Syntax: *object.Name*
- Example:

```
For each attr in Compobj.Attributes  
  attrNames = "attrNames" & attr.Name,, "Name".
```

Related Topics

- [Properties Object Types](#)

- [Attributes Property](#)
- [Count Property](#)
- [Library Property](#)
- [NameVisible Property](#)
- [Symbol Property](#)
- [Value Property](#)
- [ValueVisible Property](#)

NameVisible Property

The NameVisible property has the following characteristics:

- Applies to the Property object and is a Boolean type.
- Syntax: *object.NameVisible = Boolean*

Related Topics

- [Properties Object Types](#)
 - [Attributes Property](#)
 - [Count Property](#)
 - [Library Property](#)
 - [Name Property](#)
 - [Symbol Property](#)
 - [Value Property](#)
 - [ValueVisible Property](#)

Symbol Property

The Symbol property has the following characteristics:

- Applies to the Component object and is a String type (sets/returns user-defined names).
- Syntax: *object.Symbol*

Related Topics

- [Properties Object Types](#)
 - [Attributes Property](#)

- [Count Property](#)
- [Library Property](#)
- [Name Property](#)
- [NameVisible Property](#)
- [Value Property](#)
- [ValueVisible Property](#)

Value Property

The Value property has the following characteristics:

- Applies to the Attributes object.
- Syntax: *object.Value*
- Example:

```
MsgBox Attributes("Name") .Value
```

Related Topics

- [Properties Object Types](#)
 - [Attributes Property](#)
 - [Count Property](#)
 - [Library Property](#)
 - [Name Property](#)
 - [NameVisible Property](#)
 - [Symbol Property](#)
 - [ValueVisible Property](#)

ValueVisible Property

The ValueVisible property has the following characteristics:

- Applies to the Property object and is a Boolean type (sets and returns the Visibility setting).
- Syntax: *object.ValueVisible = Boolean*

Related Topics

- [Properties Object Types](#)

- [Attributes Property](#)
- [Count Property](#)
- [Library Property](#)
- [Name Property](#)
- [NameVisible Property](#)
- [Symbol Property](#)
- [Value Property](#)

Properties Object Methods

You can use the following scripting methods to manipulate objects:

- [Add Method](#)
- [Item Method](#)
- [Remove Method](#)

Related Topics

- [Properties Object](#)
 - [Properties Object Types](#)

Add Method

To access: In the DxDataBook window, **Right-click > Configure > Scripting**.

The *Add* method has the following characteristics:

- Applies to the Attributes object where *name* is the attribute name, *value* the attribute value, and *nameVisible*, *valueVisible* and *isOat* are Boolean types.
- Syntax: *object.Add name, value, nameVisible, valueVisible, isOat*

Related Topics

- [Properties Object Methods](#)
 - [Item Method](#)
 - [Remove Method](#)

Item Method

To access: In the DxDataBook window, **Right-click > Configure > Scripting**.

The *Item* method has the following characteristics:

- Applies to the Components and Properties objects (default method).
- Syntax: *object.Item arg* or *object arg*
- Parameters: *arg*
A zero-based index of a property or the Name of a property.

Related Topics

- [Properties Object Methods](#)
 - [Add Method](#)
 - [Remove Method](#)

Remove Method

To access: In the DxDataBook window, **Right-click > Configure > Scripting**.

The *Remove* method has the following characteristics:

- Applies to the Properties object.
- Syntax: *object.Remove arg*
- Parameters: *arg*
An index or Name of an attribute.
- Example:

```
Objs.Remove (3)
```

Related Topics

- [Properties Object Methods](#)
 - [Add Method](#)
 - [Item Method](#)

DxDataBook Scripting Events

An event is a user-defined action or occurrence to which a DxDataBook script can respond. The following DxDataBook scripting events can be used in a script:

- [Application_AddComponent Event](#)
- [Application_AfterAddComponent Event](#)

- [Application_AfterAnnotateComponent Event](#)
- [Application_AnnotateComponent Event](#)
- [Application_LoadComponent Event](#)
- [Application_SelectComponent Event](#)
- [Application_ViewDocument Event](#)

Note

All DxDataBook events are enabled by default.

Related Topics

- [DxDataBook Objects](#)
- [Loading a DxDataBook Script](#)

Application_AddComponent Event

The Application_AddComponent event is used in a DxDataBook script for any controls that need to be implemented before the component is added to the DxDesigner schematic (for example, to prevent unapproved parts from being added to a schematic by making sure the approved parts properties exist before instantiating a component).

The Application_AddComponent event has the following characteristics:

- Applies to the Component object and is called by the script before a unique or a generic component is added to the schematic.
- Syntax: *Application_AddComponent (object)*

Note

Component.Continue = FALSE setting cancels the add component function.

- Example:

```
Function Application_AddComponent(component)
'''
    ''' This event handler performs a few operations as an example of what
    ''' can be done
    '''
    ''' Add a property
    component.Attributes.Add "TEST", "SCRIPTING", TRUE, TRUE, TRUE
    ' set the visibility of a property
    set attr = component.Attributes.Item("Value")
    attr.NameVisible = FALSE
    ' Remove a property
    component.Attributes.Remove("Tolerance")
```

```
' If the Continue property is TRUE, then the component will be
' added to the schematic
' If the Continue property is FALSE, then the component is not added
  Component.Continue = TRUE
End Function
```

Related Topics

- [DxDataBook Scripting Events](#)
 - [Application_AfterAddComponent Event](#)
 - [Application_AfterAnnotateComponent Event](#)
 - [Application_AnnotateComponent Event](#)
 - [Application_LoadComponent Event](#)
 - [Application_SelectComponent Event](#)
 - [Application_ViewDocument Event](#)

Application_AfterAddComponent Event

The `Application_AfterAddComponent` event is used in a DxDataBook script to make direct changes to the component on a DxDesigner schematic.

The `Application_AfterAddComponent` event has the following characteristics:

- Applies to the Component object and is called by the script after a generic or unique component is added to the schematic.
- Syntax: *Application_AfterAddComponent object*
- Example:

```
Function Application_AfterAddComponent(component)
'''
''' This event handler lists the instance name of the component after
''' it is added
'''
''' Hook up to running DxDesigner
Set view = Viewdraw.ActiveView
''' When adding, this list always has one element - but we can still
''' use For Each to get to it
  For Each name In component.Instances
    ' Select the component
    ' Not required, but is usefull to know how for other operations
    view.SelectByName name
    MsgBox "InstanceName = " & name,, "UID"
  Next
End Function
```

Related Topics

- [DxDataBook Scripting Events](#)
 - [Application_AddComponent Event](#)
 - [Application_AfterAnnotateComponent Event](#)
 - [Application_AnnotateComponent Event](#)
 - [Application_LoadComponent Event](#)
 - [Application_SelectComponent Event](#)
 - [Application_ViewDocument Event](#)

Application_AfterAnnotateComponent Event

The `Application_AfterAnnotateComponent` event is used in a DxDataBook script to make direct changes to the component on a DxDesigner schematic

The `Application_AfterAnnotateComponent` event has the following characteristics:

- Applies to the Component object and is called by the script after annotating a component on the schematic.
- Syntax: *Application_AfterAnnotateComponent*
- Example:

```
Function Application_AfterAnnotateComponent(component)
'''
''' This event handler resets the slot of a component after it is
''' annotated and reports the UID and count.
'''
''' Hook up to running DxDesigner
Set view = Viewdraw.ActiveView
''' A loop is needed since more than once component might be annotated
Set CompInst = component.Instances
' Count example
CompInstCount = CompInst.count
MsgBox CompInstCount,, "Instance Count"
  For Each name In CompInst
    ' Select the component
    view.SelectByName name
    ' Reset slot and REFDES
    Viewdraw.ExecuteCommand "pdbslot 3 No"
    MsgBox "Reset slot for " & name,, "Change Slot Message"
  Next
End Function
```

Related Topics

- [DxDataBook Scripting Events](#)

- [Application_AddComponent Event](#)
- [Application_AfterAddComponent Event](#)
- [Application_AnnotateComponent Event](#)
- [Application_LoadComponent Event](#)
- [Application_SelectComponent Event](#)
- [Application_ViewDocument Event](#)

Application_AnnotateComponent Event

The `Application_AnnotateComponent` event is used in a DxDataBook script to prevent unapproved parts from being added to the DxDesigner schematic by making sure the approved part property exists before instantiating a component.

The `Application_AnnotateComponent` event has the following characteristics:

- Applies to the Component object and is called by the script before annotating a component on a schematic.
- Syntax: *Application_AnnotateComponent*
- Example:

```
Function Application_AnnotateComponent(component)
'''
''' Display a message box listing which properties will be annotated
'''
dim string
  For Each attr In component.Attributes
    sAttr = sAttr & CHR(10) & attr.Name & "=" & attr.Value
  next
  MsgBox "Properties being annotated are: " & CHR(10) & sAttr,, "Info"
  ''' If the Continue property is TRUE, the attributes will be
  ''' annotated to the component
  ''' If the Continue property is FALSE, the attributes will not be
  ''' annotated
  Component.Continue = TRUE
End Function
```

Related Topics

- [DxDataBook Scripting Events](#)
 - [Application_AddComponent Event](#)
 - [Application_AfterAddComponent Event](#)
 - [Application_AfterAnnotateComponent Event](#)
 - [Application_LoadComponent Event](#)

- [Application_SelectComponent Event](#)
- [Application_ViewDocument Event](#)

Application_LoadComponent Event

The Application_LoadComponent event is used in a DxDataBook script to override the mapping from the loaded component to the library used in the DxDataBook search.

The Application_LoadComponent event has the following characteristics:

- Applies to the Component object and is called by the script when loading components from the schematic and before searching the DxDataBook database for the components.
- Syntax: *Application_LoadComponent*
- Example:

```
Function Application_LoadComponent(components)
'
' Display a dialog box showing the loaded components
'
dim sCmpData
For Each component In components
    sCmpData= sCmpData & component.Name & CHR(10)
    sCmpData= sCmpData & " Symbol = " & component.Symbol & CHR(10)
    sCmpData= sCmpData & " Library = " & component.Library & CHR(10)
    Set attrColl = component.Attributes
    For Each attr In attrColl
        sCmpData= sCmpData & " " & attr.Name & "=" & attr.Value & CHR(10)
    Next
    sCmpData= sCmpData & CHR(10)
next
MsgBox "The loaded components are:" & CHR(10) & CHR(10) & sCmpData
End Function
```

Related Topics

- [DxDataBook Scripting Events](#)
 - [Application_AddComponent Event](#)
 - [Application_AfterAddComponent Event](#)
 - [Application_AfterAnnotateComponent Event](#)
 - [Application_AnnotateComponent Event](#)
 - [Application_SelectComponent Event](#)
 - [Application_ViewDocument Event](#)

Application_SelectComponent Event

The Application_SelectComponent event is used in a DxDataBook script to update a third-party viewer when selecting different components.

The Application_SelectComponent event has the following characteristics:

- Applies to the Properties object and is called by the script when a row is selected in the DxDataBook Component Search results window.
- Syntax: *Application_SelectComponent object*
- Example:

```
Function Application_SelectComponent(Attributes)
    '
    ' Display a message box listing the properties of the selected row
    '
    dim sAttData
    For Each attr In Attributes
        attrName = attr.Name
        attrValue = attr.Value
        sAttData= sAttData & CHR(10) & attrName & "=" & attrValue
    Next
    MsgBox "The selected properties are:" & CHR(10) & sAttData,, "Summary"
End Function
```

Related Topics

- [DxDataBook Scripting Events](#)
 - [Application_AddComponent Event](#)
 - [Application_AfterAddComponent Event](#)
 - [Application_AfterAnnotateComponent Event](#)
 - [Application_AnnotateComponent Event](#)
 - [Application_LoadComponent Event](#)
 - [Application_ViewDocument Event](#)

Application_ViewDocument Event

The Application_ViewDocument event is used in a DxDataBook script to dynamically build URLs to Web pages.

The Application_ViewDocument event has the following characteristics:

- Applies to the Properties and Property objects, and is called by the script when you click a Web page link and before the document viewer launches, so that it can build a URL.

- Syntax: *Application_ViewDocument objects*
- Arguments: *arg1* (row of properties selected in the Component Search results window) and *arg2* (property containing the document you want to view).
- Example:

```
Function Application_ViewDocument(properties, document)
    ' Launch notepad to view the document specified in the DxDataBook
    ' Document field
    set wsh = CreateObject("WScript.Shell")
    ' Searches path for notepad.exe
    wsh.Run("notepad.exe " & "C:\DataSheets\" & document.Value)
    ' set the document value to empty (") to prevent DxDataBook
    ' from launching the viewer
End Function
```

Note

Directory pointers may also be used to point to a central data sheet repository, as shown in the example above.

Related Topics

- [DxDataBook Scripting Events](#)
 - [Application_AddComponent Event](#)
 - [Application_AfterAddComponent Event](#)
 - [Application_AfterAnnotateComponent Event](#)
 - [Application_AnnotateComponent Event](#)
 - [Application_LoadComponent Event](#)
 - [Application_SelectComponent Event](#)

Loading a DxDataBook Script

This topic tells you how to load a script into DxDataBook.

Prerequisites

- Create a DxDataBook script

Procedure

1. In DxDataBook window, select **Configure > Scripting > Settings** to open the Scripting dialog box.
2. In the Filename text field, type the script file name or click the browse button to select a script file.

3. [Optional] To edit the script file, click Edit.
4. In the Language section, select the programming language used to write your script.
If you select the User-Defined option, type the programming language name into the ProgID text field.
5. Click **OK**.



Tip: You can make changes to the *.vbs* file using a text editor and then reload the script to implement the changes.

To reload a script, select **Configure > Scripting > Reload Script**.

Results

The script is loaded into DxDataBook and is available for use.

Related Topics

- [DxDataBook Objects](#)
- [DxDataBook Scripting Events](#)

Appendix A

Frequently Asked Questions

This topic contains DxDataBook frequently asked questions (FAQs) which provide you with important how-to and troubleshooting information.

The following DxDataBook FAQs are discussed:

- [DxDataBook Basics](#)
- [Setting Up and Configuring DxDataBook](#)
- [Using DxDataBook](#)
- [Using DxDataBook with a DxDesigner Schematic](#)
- [Using DxDataBook on the Web](#)
- [DxDataBook: System Administrator Tasks](#)
- [Troubleshooting DxDataBook](#)

DxDataBook Basics

The following DxDataBook FAQs are included in this topic:

- [What is DxDataBook?](#)
- [What is a DxDataBook library?](#)
- [What is a library property?](#)
- [Can I use DxDataBook to add components to a corporate database?](#)

Related Topics

- [Setting Up and Configuring DxDataBook](#)
- [Using DxDataBook](#)
- [Using DxDataBook with a DxDesigner Schematic](#)
- [Using DxDataBook on the Web](#)
- [DxDataBook: System Administrator Tasks](#)
- [Troubleshooting DxDataBook](#)

What is DxDataBook?

DxDataBook is the Design Exchange tool that provides component search and selection capabilities from databases located anywhere on the Internet or your company's intranet. It accelerates the design entry stage of PCB designs by adding components or component data to a DxDesigner schematic using an interactive library management system, which works with a corporate database.

Related Topics

- Chapter 1, [“Introduction to DxDataBook”](#)

What is a DxDataBook library?

A library is a collection of components with similar properties that you define according to your information needs. A DxDataBook Library is an entity that exists independently of any database and does not require any special database schema. For instance, you would likely put all capacitors into a single library and save libraries in a DxDataBook configuration file.

A table refers to an actual database table. For example, in Access, data is separated into tables. An Access query yields a table, as far as DxDataBook is concerned.

A DxDataBook library can also contain tables. You can use database joins to link together tables of disparate information. A DxDataBook or an Access query can be used to link these tables together.

Related Topics

- Chapter 2, [“Planning Your Library Strategy”](#)

What is a library property?

Library properties contain information on what symbol was used to create the part. You can specify the name of a property in the Property Name text field. If this property is present on a component, it represents the name of a library where DxDataBook should search for the part information.

This is a DxDataBook optimization to make loading faster. If this property is not found on the component, next it looks for the Library symbol expression to find a match, and then it runs queries to find the library.

Can I use DxDataBook to add components to a corporate database?

Because users of DxDataBook have read-only access to a corporate database, this is not part of the intended functionality. DxDataBook lets users create their own database tables and store them in a DxDataBook library, effectively appending the data to the corporate database.

Related Topics

- Chapter 3, [“Choosing a Data Source”](#)

Setting Up and Configuring DxDataBook

The following DxDataBook FAQs are included in this topic:

- [After setup is there anything I have to configure?](#)
- [Why do I get errors with my evaluation copy of DxDataBook?](#)
- [When importing from ViewDataBookII, which configuration settings does DxDataBook use?](#)
- [How do I setup DxDataBook to work with Microsoft Excel spreadsheets?](#)
- [How do I setup DxDataBook to work with Internet proxy servers?](#)

Related Topics

- [DxDataBook Basics](#)
- [Using DxDataBook](#)
- [Using DxDataBook with a DxDesigner Schematic](#)
- [Using DxDataBook on the Web](#)
- [DxDataBook: System Administrator Tasks](#)
- [Troubleshooting DxDataBook](#)

After setup is there anything I have to configure?

You need to configure the license and installation environment variables, and setup the server for DxDataBook client and browser access.

Prerequisites

- Install the DxDesigner schematic capture application and DxDataBook database browser software (see Chapter 1, “[DxDataBook Requirements](#)”)

Procedure

1. Add the following environment variables:
 - **LM_LICENSE_FILE** — Points to a license file or license server.
 - **WDIR** — Points to the *\standard* directory of an eProduct Designer installation
2. Restart the machine to make the system environment variables take effect.

To make the server ready for DxDataBook client access:

1. Run the ODBC Administrator, and add System DSNs to identify the databases you would like to connect to.
2. Start DxDBCConfig.

3. In DxDBConfig, select the DxDataBook **Client Access** tab and add Database Aliases to the list. Database Aliases allow you to define the set of DSNs that are visible to DxDataBook clients.
4. Restart the IIS server to apply the changes.
5. In Windows, select **Start > All Programs > Microsoft Peer Web Services > Internet Service Manager**.
6. In the Service Manager dialog box, select WWW, and click the Stop and Start buttons.

To prepare the server for browser access:

1. Run DxDataBook (the client) and set up a DxDataBook User Configuration (.dbc) file to define the library structure to be visible from web browsers.
2. In the DxDBConfig tool, select the **Browser Access** tab and set the filename of your .dbc file.
3. Restart the IIS server to pick up the changes.

Results

The license and installation environment variables are added, and the server for DxDataBook client and browser access is setup.

Related Topics

- Chapter 3, “[DxDataBook Data Sources](#)”

Why do I get errors with my evaluation copy of DxDataBook?

Check with your system administrator to see if there is a firewall. The DxDataBook evaluation license uses an IP address located at Mentor Graphics to issue a access to your DxDataBook application. If your company firewall prevents reading-in access, reading the license produces error messages. Consult your system administrator for more information on firewall restrictions.

When importing from ViewDataBookII, which configuration settings does DxDataBook use?

DxDataBook supports most of the ViewDataBookII configuration options. [Table A-1](#) shows which configuration settings are supported in DxDataBook.

Table A-1. Supported DxDataBook Configuration Settings

Configuration File/Feature	In DxDataBook Configuration?	Imported during Translation?
vllm.scm		
OrganizeByAttr	No	No

Table A-1. Supported DxDataBook Configuration Settings (cont.)

Configuration File/Feature	In DxDataBook Configuration?	Imported during Translation?
MapAttrs	Yes	Yes
ExcludeSymsWhenLoading	Yes	Yes
ExcludeAttrsWhenLoading	Yes	Yes
ExcludeAttrsWhenAnnotating	Yes	Yes
AttrVisVisible	Yes	Yes
AttrVisInvisible	Yes	Yes
AttrVisValue	Yes	Yes
AttrVisName	Yes	Yes
MaintainAttrVis	Yes	Yes
SymAttrsGetLoaded	Yes	Yes
ExcludeAttrsFromDisplay	No ¹	No
CategoryDisplayAttrsDefault	No	No
CategoryDisplayAttrs	No	No
RequiredAttrs	No	No
TextColorsEnabled	Not Applicable (N/A)	N/A
ComponentTracking	Yes	No
UndefinedValue	Yes	Yes
AnnotateUndefined	Yes	Yes
TestValues	No ²	No
LibNameAttr	Yes	Yes
UseLibraryAlias	No	No
DisplayComponentNames	No	No
MemberAttr	No	No
vdbsetup.scm		
CommonValueColor	No	No
LoadedValueColor	N/A	N/A
AddWithLibName	No	No
VtypeAddTrailingWildcard	No	No
VtypeAddLeadingWildcard	No	No

Table A-1. Supported DxDataBook Configuration Settings (cont.)

Configuration File/Feature	In DxDataBook Configuration?	Imported during Translation?
vllm.scm		
DefineAttr	No	No
AddLibrary	Yes ³	Yes
LibExcludeComp	No	No
DbExcludeAttr	Yes	No
DbSetDefaultMagnitude	Yes	No
vllm.ini		
vllm.sort_attr_names	No	No
vllm.search_for_symbols	No	No
vllm.[database]_search_for_symbols	No	No
vllm.hunt_wdir	N/A	N/A
vllm.hunt_entire_wdir	N/A	N/A
vllm.vdraw_ini_locations	N/A	N/A
vllm.ignore_param_attrs	No	No
vllm.ParKeyField	N/A ⁴	Yes
vllm.ParDelimiter	N/A ⁵	Yes
vllm.ParUndefinedValue	No	No
vllm.change_comp_enable	Yes	No

1. You can exclude them totally, so that they won't show up in display or be annotated, but there is no way to just make a property invisible in the search window.

2. DxDataBook does test fields to see if they are empty. If so then it won't annotate that particular property. Also, DxDataBook does test against the UndefinedValue string and acts according to the AnnotateUndefined flag.

3. DxDataBook supports the concept of a library. When doing a PAR/VDB translation, the libraries are automatically created.

4. The translator uses the ParKeyField to create link tables. Beyond that, there is no user configuration inside DxDataBook.

5. The translator uses the ParDelimiter to translate the PAR files. There is no need for this to be in the DxDataBook user configuration.

How do I setup DxDataBook to work with Microsoft Excel spreadsheets?

DxDataBook can access Microsoft Excel spreadsheets using the Microsoft Excel ODBC driver. You can access data contained in Excel 5.0., Excel 95 and 97 (previous versions of Excel can also be used, although the setup procedure is different).

The first step is to add “named ranges” to the spreadsheet. This corresponds with a table in DxDataBook. The second step is to configure the ODBC data source driver.

Related Topics

- Chapter 3, “[Microsoft Excel Spreadsheets](#)”

How do I setup DxDataBook to work with Internet proxy servers?

If the remote system is outside a corporate firewall, and the company is using a proxy server for web access, you may need to configure your system to allow DxDataBook to access the remote system.

Procedure

1. In the Windows **Control Panel**, double-click the **Internet Options** button.

Note



If the **Control Panel** does not contain an Internet Options button, install Microsoft Plus! or Internet Explorer.

2. Select **Connections > LAN settings**.
3. In the Proxy server section, select the Use a proxy server for your LAN option.
4. Type the address and port number of your proxy server in the Address and Port text fields.

Results

Your system is configured to allow DxDataBook access the remote Internet proxy system.

Related Topics

- Chapter 3, “[DxDataBook Web Server](#)”

Using DxDataBook

The following DxDataBook FAQs are included in this topic:

- [Do I have to redo all of the visibility controls every time I re-create or reconfigure a library?](#)
- [How do I run queries against horizontally joined tables?](#)
- [How do I use PKT files for package-specific properties?](#)
- [How do I load an existing .dbc file and write it back out as a .dbb file?](#)

- [If I move a library, do I need to re-create a .dbc or .dcc file from scratch?](#)
- [In the Results window, some of my critical fields need widely varying widths—do I have to resize columns every time I run a query?](#)
- [How do I convert whole numbers to integers and vice versa?](#)
- [Are there any reserved keywords for documentation-type properties?](#)
- [When attempting to perform a search, why do I get an ODBC error message?](#)

Related Topics

- [DxDataBook Basics](#)
- [Setting Up and Configuring DxDataBook](#)
- [Using DxDataBook with a DxDesigner Schematic](#)
- [Using DxDataBook on the Web](#)
- [DxDataBook: System Administrator Tasks](#)
- [Troubleshooting DxDataBook](#)

Do I have to redo all of the visibility controls every time I re-create or reconfigure a library?

No, the defaults for Name Visible and Value Visible are off (not selected). For library-specific exceptions – such as Value for discretes – you need to set them if you recreate or reconfigure a library.

Related Topics

- Chapter 2, [“Planning Your Library Strategy”](#)

How do I run queries against horizontally joined tables?

When you run a query against horizontally joined tables, the query result contains field names from multiple tables. It is possible that two tables may have identical field names (such as a part number field), so you need to give the table field names unique names.

This is handled in SQL by using a combination of the table name and field name separated by a period.

For example:

```
tablename.fieldname
```

Because the above `tablename.fieldname` syntax is generated by DxDataBook only when using horizontally joined tables, this is the only time that this can cause a problem.

If you observe the following guidelines when naming tables and fields, most of these types of problems can be averted.

- Keep name to a maximum of 64 characters.

- Use any combination of letters, numbers, spaces, and special characters, except a period (.), an exclamation point (!), an accent grave (`), and brackets ([]).
- Do not begin name with leading spaces.
- Do not use control characters (ASCII values 0 through 31).

Related Topics

- Chapter 5, “[Creating Search Queries](#)”

How do I use PKT files for package-specific properties?

Physical information about symbols should not be in the database. You should not store symbol graphic with physical information. You can store a pure graphical representation of a symbol. Physical information for multiple packet files can be linked together through a database property.

Related Topics

- Chapter 4, “[Defining Data Sources, Libraries, and Symbols in the Configuration File](#)”

How do I load an existing .dbc file and write it back out as a .dbb file?

Use the DBCtool.exe utility.

Related Topics

- Chapter 4, “[Modifying a DxDataBook Configuration File with the DBCtool Utility](#)”

If I move a library, do I need to re-create a .dbc or .dcc file from scratch?

No. If you can keep the ODBC data source the same, the configuration file does not need to be updated. All the path information is preserved by the ODBC data source.

In the Results window, some of my critical fields need widely varying widths—do I have to resize columns every time I run a query?

With WVO 7.5 or greater, you can set a preference for auto-sizing columns that contain critical fields (such as Value and Tolerance for discretes, Description for hardware). Select **Edit > User Preferences** to open the User Preferences dialog box. In the Component Search section, select the Auto-size query results column option.

How do I convert whole numbers to integers and vice versa?

For example, the .vdb file contains a Value column with values in the form 1K, 1MEG, etc. This column is translated into INTEGER type, and the values are not recognized inside DxDataBook.

Check the table definition to see what Magnitude you are using. To check this, open the Configure Libraries dialog box and scroll across until you see the Magnitude field. Ensure this field is Auto, not null.

You can also use the following workarounds:

- Modify the *.vdb* file(s) to be full-numerical.
- Update the configuration for the ODBC setup so that type of Value is CHAR.

Are there any reserved keywords for documentation-type properties?

DOCUMENT is a recognized keyword for DxDataBook to automatically generate a Document type field.

When attempting to perform a search, why do I get an ODBC error message?

In SQL, a period (.) is a special character. When using the ODBC text driver, the filename actually becomes the table name. If the filename has an extension, the text driver keeps this as a part of the table name.

The solution is to modify the table filenames so they do not contain an extension. You can do this by either modifying the existing ODBC data source or adding a new one.

Procedure

1. Rename all table files, removing the extension (for example, change *capacitors.txt* to *capacitors*).

Note



Do not rename the *schema.ini* file.

2. Edit the *schema.ini* file, removing the extensions from all files referenced in this file.
3. Edit the link table file, removing the extensions from all files referenced in this file.
4. Start the ODBC Data Source Administrator and add *. to the Extensions List.
5. Modify the DxDataBook configuration file to point to the new tables.

To add a new ODBC data source:

1. Copy the directory containing the database table files to a new directory.
2. Rename all table files, removing the extension (for example, change the *capacitors.txt* file to *capacitors*).

Note



Do not rename the *schema.ini* file

3. Edit the link table file, removing the extensions from all files referenced in this file.
4. Start the ODBC Data Source Administrator and add a new ODBC Text Driver Data Source to point to the new directory.
5. In the ODBC Text Setup dialog box, add *. to the Extensions List.
6. Select the *. extension and click Define Format.
7. For each table, select the Column Name Header option, set the delimiter type, click Guess, and set the column types.
8. Close the ODBC Administrator.

Modify the DxDataBook configuration file to point to the new ODBC Data Sources and tables.

Results

Removed the table filename extension in the ODBC data source to resolve the error condition.

Related Topics

- Chapter 3, “[DxDataBook Data Sources](#)”

Using DxDataBook with a DxDesigner Schematic

The following DxDataBook FAQs are included in this topic:

- [What are the basics of using DxDataBook with schematics in DxDesigner?](#)
- [Why would I add or annotate with generic components?](#)
- [When I try to add a part to my schematic, I get a parameter not optional message. What does that mean?](#)
- [How would I prepare a bill of materials?](#)

Related Topics

- [DxDataBook Basics](#)
- [Setting Up and Configuring DxDataBook](#)
- [Using DxDataBook](#)
- [Using DxDataBook on the Web](#)
- [DxDataBook: System Administrator Tasks](#)
- [Troubleshooting DxDataBook](#)

What are the basics of using DxDataBook with schematics in DxDesigner?

You can think of DxDataBook as a connectivity tool for linking DxDesigner with corporate parts databases. Once you have set up your libraries in DxDataBook, you are ready to apply this information to any new or existing DxDesigner schematic.

Existing schematics

For an existing schematic, open it and select the components you want to match with your corporate parts database. Then select **Components > Load from DxDesigner** to display all the components in your library that match the properties set for this component in the schematic.

At this stage, you can annotate the generic symbol information in the schematic—property information that is common to all the approved parts in your database. Select **Components > Annotate Generic**. If you are ready to annotate with a unique part, you can select from the results just the component you want narrow down the list to just the part you want to use by performing searches based on particular property values. When you have chosen the part, select it and **Components > Annotate Unique**.

New schematics

For a new schematic, you can add unique or generic components from your DxDataBook libraries as you draw the schematic. Use the DxDataBook search feature to narrow down the list to just the parts you want. Then select **Component > Add Generic** or select one and select **Components > Add Unique**.

You can also use DxDataBook to verify the integrity of the entire design, ensuring the information on the schematic is consistent with the corporate database.

You can also use DxDataBook to advise which parts of the design have not been uniquely identified and automatically annotate any missing property information.

Related Topics

- Chapter 5, [“Adding a Unique Component to a Schematic”](#)
- Chapter 5, [“Verifying Components on a Single Schematic Sheet”](#)
- Chapter 5, [“Verifying a Design with Multiple Schematic Sheets”](#)

Why would I add or annotate with generic components?

A designer may want to postpone selecting a part from a particular component manufacturer until later in the design process (for example, after functional simulation). When it is time to select the manufacturer, the designer can select the component and DxDataBook automatically displays the available manufacturers and their respective status from production control. The decision can be based on such criteria as lead time, price, or inventory level. Once the selection is made, you can annotate the component’s unique properties to the generic component in the schematic.

Related Topics

- Chapter 5, “[Adding a Generic Component to a Schematic](#)”

When I try to add a part to my schematic, I get a *parameter not optional* message. What does that mean?

You have probably installed the ViewAnalog Web update, and the DxDesigner executable is slightly incompatible. You can fix this problem by installing WVO 7.5. If you are still using WVO 7.4, download the patch from the DxDesigner web site.

How would I prepare a bill of materials?

To generate a bill of materials (BOM), you would use DxDataBook, DxDesigner, and a number of other utilities. You could create a batch file to automate the whole process once you have it setup the way you want.

Procedure

1. Annotate the DxDesigner schematic with component cost data when you add or annotate components.

Note



When you load components from DxDesigner, you need to ignore the cost property.

2. Set up ViewDRC to generate a *.baf* file with the up-to-date cost data by adding the LibAdvisDxDB and LibAdvisAnnotate parameters.
3. Run the BAF2VL utility to back-annotate the schematic with the updated cost data. To start BAF2VL, [click here](#).
4. Set up the Partslist tool to generate a data file containing component cost data.

Note



Partslist utilizes an initialization file (*design.ini* by default) to determine what properties to process and how to process them. It then writes the desired data to a file that you specify (*design.lst* by default).



Tip: When the CLASS=LAB property is placed on a symbol or component, it keeps the component from getting added to the layout netlist file and partlist. To add the component to the partlist, but not the netlist, alias the CLASS=LAB to something else. You can do this with the DEVICE property (such as MECHANICAL), which keeps the component out of the netlist file and places it in the partlist.

Results

You created a batch file to automate the BOM generation process.

Using DxDataBook on the Web

The following DxDataBook FAQs are included in this topic:

- [How does DxDataBook work on the Web?](#)
- [What is the licensing requirement for the DxDataBook web server?](#)
- [What are the system requirements for the Design Exchange Web Pack?](#)
- [How do I install the DxDataBook Web Server?](#)
- [What does the Design Exchange Web Pack install?](#)
- [How do I access my DxDataBook server?](#)

Related Topics

- [DxDataBook Basics](#)
- [Using DxDataBook with a DxDesigner Schematic](#)
- [Setting Up and Configuring DxDataBook](#)
- [DxDataBook: System Administrator Tasks](#)
- [Using DxDataBook](#)
- [Troubleshooting DxDataBook](#)

How does DxDataBook work on the Web?

DxDataBook can access databases located on remote systems using HTTP (see [Figure A-1](#)). A DxDataBook web server is installed on the remote system, and the DxDataBook client uses a URL to connect to the remote system.

The DxDataBook web server, which is part of the Design Exchange Web Pack, is an extension to the Microsoft Internet Information Server (IIS). You can install the Web Pack with the PC version of DxDataBook.

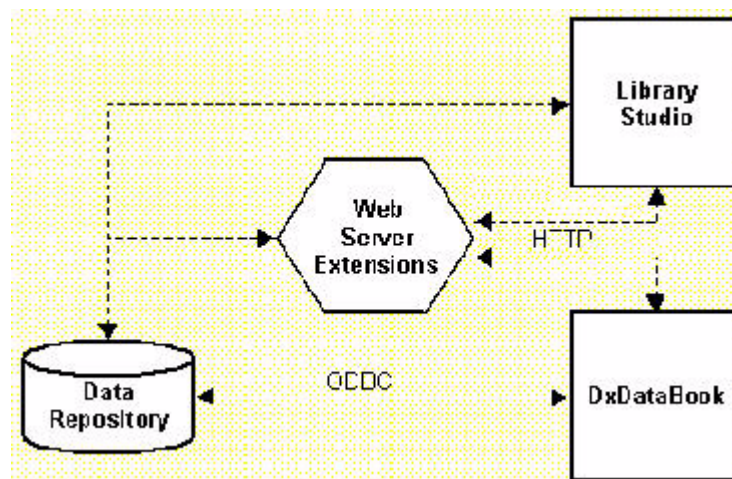
It provides a way to move ODBC setup and configuration from your desktop to a server machine. You only need to specify a URL in DxDataBook to start performing component searches. Your environment can be mixed between direct ODBC access and access through a DxDataBook web server, with some data pulled from the server and some directly using ODBC.

DxDataBook Web also has an HTML interface for lightweight access to the data from web browsers.

Point your browser to the following location to access data:

`http://<machine.yourcompany.com>/dxdb/default.html`

Figure A-1. Internet/Intranet Integration Flow



Related Topics

- Chapter 3, “[DxDataBook Web Server](#)”

What is the licensing requirement for the DxDataBook web server?

The web server uses a single DxDataBook license. Each DxDataBook client also uses a license.

What are the system requirements for the Design Exchange Web Pack?

Your system must meet the following requirements in order to use Design Exchange Web Pack:

- The OS of your Server or Workstation must be Windows 2000 or XP.
- You must be logged in with administrator privileges.
- You must have already installed eProduct Designer.
- You must have installed Internet Information Server [US] 2.0 or greater - 3.0 or 4.0 is recommended.

About IIS

If your browser can connect to <http://localhost>, you probably have IIS.

For IIS 3.0, **Start > Programs > Microsoft Peer Web Services**.

For IIS 4.0, **Start > Programs > Windows Option Pack - Microsoft Personal Web Server**.

Note



Do not confuse **Personal Web Server** with the Win95 version by the same name – this is not supported.

How do I install the DxDataBook Web Server?

The DxDataBook Web Server has a separate install. Run the eProduct Designer cdinstal.exe again and click the Design Exchange Servers button. On the Design Exchange Information & Installation window, click the Design Exchange Web Pack button.

Check the requirements information and installation process description. Click the Install button to start the installation. Upon completion, read the additional configuration information in your Internet browser.

What does the Design Exchange Web Pack install?

This installation creates a Web Servers group under the DxDesigner Servers group in your Windows **Start** menu and includes the following:

- **DxDBWeb Configuration** — Runs the administrator program, which allows you to set up and administer the DxDataBook web server extension.
- **DxDesigner View Only Configuration** — Runs the HTML-based administrator program, which allows you to set up and administer the DxDesigner View Only web server extension.
- **Design Exchange on the Web** — Launches your web browser and connects to the web server, allowing you to view DxDataBook data and DxDesigner schematics with your browser.
- **Design Exchange Web Pack installation Notes** — Launches your web browser and displays pages which contain additional installation and setup instructions.

How do I access my DxDataBook server?

Use a web browser and type the following URL:

`http://<machine.domain.com>/dx/dxdb.html`

When configuring new libraries using DxDataBook, use this URL:

`http://<machine.domain.com>/dx/dxdbweb.html`

The DxDataBook Library Wizard prompts you to enter DxDataBook Server URLs. Remote data sources appear in the list of available database tables. Their names are determined according to the database aliases you (or others) setup when configuring the server.

DxDataBook: System Administrator Tasks

The following DxDataBook FAQs are included in this topic:

- [What do I need to try the DxDesigner Web Viewer?](#)
- [What is the licensing requirement for the DxDataBook web server?](#)

Related Topics

- [DxDataBook Basics](#)
- [Using DxDataBook with a DxDesigner Schematic](#)
- [Setting Up and Configuring DxDataBook](#)
- [Using DxDataBook on the Web](#)
- [Using DxDataBook](#)
- [Troubleshooting DxDataBook](#)

What do I need to try the DxDesigner Web Viewer?

DxDesigner VO (View Only) comes with the DxDataBook Web Pack. The prerequisites are the same as for the rest of the install. You must have the IIS (Internet Information Server) — which is also referred to as Peer Web Services — on your system.

If your browser can connect to `http://localhost`, you probably have IIS.

For IIS 3.0, **Start > Programs > Microsoft Peer Web Services**.

For IIS 4.0, **Start > Programs > Windows Option Pack - Microsoft Personal Web Server**.

Note



Do not confuse **Personal Web Server** with the Win95 version by the same name – this is not supported.

Related Topics

- Chapter 3, “[DxDataBook Web Server](#)”

What is the licensing requirement for the DxDataBook web server?

The web server uses a single DxDataBook license (each DxDataBook client also uses a license).

Troubleshooting DxDataBook

The following DxDataBook FAQs are included in this topic:

- [I get an Unable to create IDispatch message.](#)
- [My Symbol Previewer does not work when I select View > Symbol Preview.](#)
- [When I annotate components in DxDesigner, why is the annotation not always correct?](#)
- [When I launch DxDataBook I get an incompatible ODBC drivers or updated ODBC drivers needed message.](#)
- [When loading components from DxDesigner, the Results window does not show matching parts in the DxDataBook library. Is there something wrong with the symbol search path?](#)

- When using DxDataBook to connect to a Microsoft Access database on the network, I get an error message.

Related Topics

- [DxDataBook Basics](#)
- [Using DxDataBook with a DxDesigner Schematic](#)
- [Setting Up and Configuring DxDataBook](#)
- [Using DxDataBook on the Web](#)
- [Using DxDataBook](#)
- [DxDataBook: System Administrator Tasks](#)

I get an *Unable to create IDispatch* message.

In order for DxDataBook to be able to launch DxDesigner, DxDesigner must be registered on your system.

Procedure

1. In Windows, select **Start > Run**.
2. In the Run dialog box, type *dcomcnfg*.
3. In the Distributed COM Configuration Properties dialog box, select the **Default Security** tab.
4. In the Default Launch Permissions section, click the Edit Default button to open the Registry Value Permissions dialog box.

Make sure the registry value is set to Allow Launch for the current user.

5. Close the dialog box, and launch DxDataBook and DxDesigner again.

If DxDesigner still fails to launch, check the DCOM Launch Permissions to see if they are preventing DxDataBook from launching DxDesigner.

Results

DxDesigner is registered on your system and can now be launched from DxDataBook.

My Symbol Previewer does not work when I select View > Symbol Preview.

The previewer requires an OCX control and there may be a problem with registering it.

Procedure

1. In Windows, select **Start > Run**.
2. Type the following into the Run dialog box:

```
regsver32.exe <dxdb_install_dir>\presym.ocx
```

(where *<dxdb_install_dir>* is the full path to the DxDataBook installation directory)

Errors are likely to be caused by either of the following:

- A required DLL is missing — An error message is usually generated if this is the case. If the message supplies the name of a missing DLL, attempt to locate the file, copy it into the DxDataBook install directory, and re-register the control
- An old version of OLEAUT32.DLL is being used by the system. If an error message only states that the control failed to register, then the problem may be caused by an old version of OLEAUT32.DLL. The version must be 2.20.4037 or greater. This is likely only on Windows 95 systems that do not have Internet Explorer 3.01 or newer installed.

Results

The OCX control is registered and you now use the Symbol Previewer.

When I annotate components in DxDesigner, why is the annotation not always correct?

By default, DxDataBook uses the database field name, so that is where the footprint=0805 came from. In the **Libraries** tab of the Configure dialog box, you need to set up the mapping between database field names and DxDesigner property names.

Related Topics

- Chapter 4, “[Editing Library Information](#)”

When I launch DxDataBook I get an *incompatible ODBC drivers or updated ODBC drivers needed* message.

You need to install ODBC 3.0. eProduct Designer installs ODBC 3.0 automatically. If you are using WVO 7.4, download the ODBC 3.0 update from the Design Exchange web site.

Note



You need to delete (or, as a precaution, move into a temporary directory) the *odbc*.dll* files in the /winnt/system32 directory and then perform the installation. The installation does not delete ODBC files that already exist.

Related Topics

- Chapter 3, “[DxDataBook Data Sources](#)”

When loading components from DxDesigner, the Results window does not show matching parts in the DxDataBook library. Is there something wrong with the symbol search path?

The symbol search path is determined by the primary project and libraries that you set up in Dashboard. It is likely that there are discrepancies in how property values are expressed in DxDesigner and in DxDataBook. For example, a capacitor value may be expressed differently – that is, 0.01 instead of 10n. To resolve this problem, you can set up DxDataBook to ignore certain properties when loading.

Procedure

1. Select **Configure > Libraries** to open the Configure Libraries dialog box and select the appropriate library from the tree.
2. In the table of properties, clear the Load option for the property that is causing the problem.

Results

DxDataBook is now setup to ignore the properties you specified, eliminating the property value discrepancy between DxDesigner and DxDataBook.

Related Topics

- Chapter 6, “[Properties Object Types](#)”

When using DxDataBook to connect to a Microsoft Access database on the network, I get an error message.

The Microsoft Access ODBC driver creates a temporary lock (*.ldb*) file in the directory in which the database (*.mdb*) file is located. In order for you to access the database, you must create or update the lock file. To do this, you need Write permission in the directory, although the *.mdb* file itself only needs to be read-only access.

DxDataBook opens databases in a read-only mode, which allows for simultaneous access.

Procedure

1. If the location of the *.mdb* file is not known, launch the ODBC Administrator from the Windows Control Panel, select the ODBC Data Source, and click the Configure button.
2. Set NTFS permissions on the directory in which the database is located to Full Access.
3. Set the *.mdb* file to Read-Only. This gives users access to update the lock (*.ldb*) file, but they cannot modify the database by using Microsoft Access.

Results

The temporary lock (*.ldb*) file has been updated and the Microsoft Access database is now available to all users.

Appendix B

Configuration File Format

The DxDataBook configuration (*.dbc*) file uses standard XML (ASCII) format, and contains elements and subelements with attributes (see [Table B-2](#)). The *.dbc* file contents are edited with a text or XML editor.

Most of the *.dbc* file settings can also be edited in the DxDataBook window: Right-click and select **Configure > Edit Configuration** (see Chapter 4, “[Editing a DxDataBook Configuration File](#)”).

Notes

- Your system administrator or librarian is generally responsible for creating and editing the DxDataBook configuration file.
- Manually editing the *.dbc* file may require deleting component libraries associated with the Data Source Name (DSN) and then adding different libraries back to the file.

Format

A *.dbc* file conforms to the following restrictions:

- The XML file contains sections which must appear in the following order (see [Table B-1](#)):
 1. [Declaration Statement](#)
 2. [dbc Element](#)
 3. [CObject](#)
 4. [CConfigSym](#)
 5. [CConfigLib](#)
 6. [CConfigLibEntry](#)
 7. [CConfigAtt](#)
 8. [CConfigTable](#)
 9. [CConfigPref](#)
 10. [CConfigScripting](#)
 11. [End Statement](#)
- The XML file content is case sensitive.

Parameters

Standard XML file syntax:

```
<element attribute1="value1" ... > elements' contents </element>
```

The XML file elements can be nested and are called subelements.

For example,

```
<element>
<subelement1> ... </subelement1>
<subelement2> ... </subelement2>
</element>
```

If an element statement does not contain information, it is displayed in shorthand notation:

```
<element attribute1="value1" ... />
```

All XML file elements need to be numbered in order of appearance in the file.

For example,

```
<element1><element2><element3><element3><element4><element4><element1>
<element5><element5><element1>
```

See [Example](#) for sample XML file containing nested and numbered elements.

Table B-1. XML File Structure

File Section	Example Value	Description
Declaration Statement	<pre><?xml version="1.0" encoding="UTF-8"?></pre>	The file begins with a standard XML declaration which includes a UTF-8 encoding statement.
dbc Element	<pre><dbc version="1.0"></pre>	The dbc element is the top-level or main element of the XML file with version number as its attribute (this element contains all of the other file elements/subelements).
CObject	<pre><CObject0 overallSchema="21" flag="DX Databook Overlay Configuration File" BaseConfigurationURL=""> <CConfig1> ... </CConfig1> <CObList128 CPersistList="0"/> </CObject0></pre>	<p>CObject contains the DxDataBook main configuration (CConfig element) and internal window data (CObList element).</p> <p>Note: Do not modify the CObList element or its attributes.</p>

Table B-1. XML File Structure (cont.)

File Section	Example Value	Description
CConfigSym	<pre><CConfigSym2> <CObArray3 CXMLTypedPtrArraySize="0" / > </CConfigSym2></pre>	<p>CConfigSym contains the list of symbols and it is a subelement of CConfig.</p> <p>Note: CConfigSym cannot be used to store symbols.</p>
CConfigLib	<pre><CConfigLib4> <CObArray5 CXMLTypedPtrArraySize="8"> <CConfigLibEntry6 > ... </CConfigLibEntry6> ... </CObArray5> </CConfigLib4></pre>	<p>CConfigLib contains the list of libraries, and it is a subelement of CConfig with the following subelements:</p> <ul style="list-style-type: none"> • CObArray which has the following subelements: • CConfigLibEntry — See CConfigLibEntry section “Description” for more information. • CConfigTableEntry — Contains subelement CStringList which lists all table fields. • CConfigAtt — See CConfigAtt section “Description” for more information. • CConfigTable — Contains the list of tables (see CConfigTable section “Description” for more information).
CConfigLibEntry	<pre><CConfigLibEntry6> <CConfigAtt7 ... > ... </CConfigAtt7> <CConfigTable32> ... </CConfigTable32> </CConfigLibEntry6></pre>	<p>CConfigLibEntry is a subelement of CObArray and it contains the DxDataBook library configuration.</p>

Table B-1. XML File Structure (cont.)

File Section	Example Value	Description
CConfigAtt	<pre> <CConfigAtt7 UseCentralLibrarySymbols="0"> <CObArray8 CXMLTypedPtrArraySize="2"> <CConfigAttEntry9> ... </CConfigAttEntry9> <CConfigAttEntry9> ... </CConfigAttEntry9> </CObArray8> </CConfigAtt7> </pre>	<p>CConfigAtt contains the list of properties and it is a subelement of CConfigLib (CConfigAtt has subelement CConfigAttEntry).</p> <p>Note: CConfigAttEntry is also a subelement of CObArray.</p>
CConfigTable	<pre> <CConfigTable32> <CObArray33 CXMLTypedPtrArraySize="1"> <CConfigTableEntry34> ... </CConfigTableEntry34> </CObArray33> </CConfigTable32> </pre>	<p>CConfigTable contains the list of tables and it is a subelement of CConfigLib (CConfigTable has subelement CObArray — See CConfigLib section “Description” for more information about CObArray).</p>
CConfigPref	<pre> <CConfigPref125 LoadIntoNewWindow="0 " SymbolAttributesGet- Loaded="0 " ComponentTracking="1 " AnnotateUndefinedAttributes=" 0 " Chan-geCompEnabled="0 " ComponentTrackingZoom="0 " MaintainAttVisibility="0 " LibraryAttEnabled="1 " LibraryAttName="DXDB_LIBNAME " AnnotateUndefinedAttributesValue=" " UseLoadTabAttribute="0 " LoadTabAttributeName=" " An- notateSelected="0 " AnnotateSelectedPrefix="ALT_* " AnnotateRemoveAttri- butes="1 " ExcludedAttList="REFDES"/> </pre>	<p>CConfigPref contains the configuration file preferences and it is a subelement of CConfig.</p>

Table B-1. XML File Structure (cont.)

File Section	Example Value	Description
CConfigScripting	<pre> <CConfigScripting126 OnAddComponentEnabled="1 " OnAnnoComponentEnabled="1 " OnLoadComponentEnabled="1 " OnSelectComponentEnabled="1 " OnViewDocumentEnabled="1 " OnAfterAddComponentEnabled="1 " OnAfterAnnoCom- ponentEnabled="1"> <CConfigScriptLanguage127 Filename="" type="0" pro- gID="VBScript"/> </CConfigScripting126> </pre>	CConfigScripting is a subelement of CConfig and it contains DxDataBook scripting (automation control) information. CConfigScripting has subelement CConfigScriptLanguage.
End Statement	</dbc>	The file ends with the XML end statement.

Table B-2. Element/Subelement Attributes

Element/Subelement	Attribute	Description
CConfigAtt	UseCentralLibrarySymbols="0"	Symbol data from Central Library is used. Values are 0 (disabled) and 1 (enabled).
CConfigAttEntry	Fieldname="ID"	Name of column in ODBC data source table.
	AttName="ID"	Name of property as it appears in DxDataBook and DxDesigner.
	DefaultValue=""	Value used if property not present (value dependent on property).
	ExcludeWhenAnnotating="0"	Properties to exclude when back-annotating to DxDesigner. Values are 0 (disabled) and 1 (enabled).
	ExcludeWhenLoading="0"	Properties to exclude when loading into DxDataBook. Values are 0 (disabled) and 1 (enabled).
	m_bNameVisible="0"	Property names are visible in DxDesigner schematic after back-annotating. Values are 0 (disabled) and 1 (enabled).
	m_bValueVisible="0"	Property values are visible in DxDesigner schematic after back-annotating. Values are 0 (disabled) and 1 (enabled).

Table B-2. Element/Subelement Attributes (cont.)

Element/Subelement	Attribute	Description
	AttType="0"	Field type. Values are 0 (Normal), 1 (Symbol), 2 (Document), 3 (Unused), 4 (Unique ID).
	ValueType="0"	Type of value. Values are 0 (Undefined), 1 (String), 2 (Numeric), 3 (Numeric_Int), 4 (Numeric_Float), 5 (Numeric_Double).
	MagType="0"	Numeric property values displayed in DxDataBook. Values are 0 (Unknown), 1 (Atto), 2 (Femto), 3 (Pico), 4 (Nano), 5 (Micro), 6 (Milli), 7 (Unity), 8 (Kilo), 9 (Mega), 10 (Giga), 11 (Tera), 12 (Auto), 13 (Percent).
	UnitsString=""	Unit suffix annotated to symbols. Values are expressed as a string (for example, "mm" or "in").
	AddAsOat="0"	For internal use only.
	Required="0"	Properties that are required. Values are 0 and 1 (property is required and included).
	ValidMag="2047"	Valid magnitude for selected property. Value is the number, which is interpreted as a binary mask that enables values described in MagType (each MagType value is interpreted as one bit).
	ValidMagDir="63"	For internal use only.
	FutureUseBool1="0", FutureUseBool2="0", FutureUseStr1="", FutureUseStr2=""	For internal use only.
	MatchCondition="-"	SQL database language relation used for matching. Values are all supported SQL values (for example, ==). Note: Only applicable if DxDataBook is used by EVM.

Table B-2. Element/Subelement Attributes (cont.)

Element/Subelement	Attribute	Description
	AttrUnplace="0"	Property placement. Values are 0 (Name), 1 (Value), 2 (NameAndValue), 3 (Hide), 4 (Delete), 5 (None).
	SerialKey="0"	For internal use only.
CConfigLibEntry	LibraryName="name"	Library name.
	JoinTable="false"	Tables linked to provide access to component properties that reside in multiple tables. Values are "true" or "false."
	Locked="0"	For internal use only.
	SymbolExpression=""	Symbol expression used to find a valid symbol.
	JoinType="0"	Tables linked vertically or horizontally. Values are 0 (vertical join) or 1 (horizontal join).
	HorizJoinType="0"	Records from tables combined if values of joined fields meet specified criteria (inner join) or records included even if related records are not in joined tables (left outer). Values are 0 (inner) or 1 (left outer).
CConfigPref	LoadIntoNewWindow="0"	New window is created when components loaded. Values are 0 (window not created) or 1.
	SymbolAttributesGetLoaded="0"	Symbol properties loaded along with component properties. Values are 0 (properties not loaded) or 1.
	ComponentTracking="0"	Automatic selection of component in DxDesigner when selected in Search/Verify windows. Values are 0 (component not selected) or 1.
	AnnotateUndefinedAttributes="0"	Values matching undefined value are annotated. Values are 0 (values not annotated) or 1.
	ChangeCompEnabled="0"	New symbol can be overwritten when annotating to a component. Values are 0 (symbol not overwritten) or 1.

Table B-2. Element/Subelement Attributes (cont.)

Element/Subelement	Attribute	Description
	ComponentTrackingZoom="0"	Zoom To Fit is performed on selection. Values are 0 (Zoom To Fit not performed) or 1.
	MaintainAttVisibility="0"	Properties change their visibility when annotated. Values are 0 (visibility not changed) or 1.
	LibraryAttEnabled="0"	Special library property is annotated. Values are 0 (property not annotated) or 1.
	LibraryAttName="DXDB_LIBNAME"	Name of the property which holds the library name (for example, DXDB_LIBNAME).
	AnnotateUndefinedAttributesValue=""	String to match values (if a value matches, it is not annotated).
	UseLoadTabAttribute="0"	Load tabs in DxDataBook search window use "LoadTabAttributeName." Values are 0 or 1 (load tabs use "LoadTabAttributeName").
	AnnotateSelected="0"	Annotate properties to selected components. Values are 0 (not enabled) or 1.
	AnnotateSelectedPrefix="ALT_*	"AnnotateSelectedPrefix" used in conjunction with "AnnotateSelected" attribute.
	AnnotateRemoveAttributes="0"	Properties without values are removed during annotation. Values are 0 (properties not removed) or 1.
	ExcludedAttList="REFDES"	Comma-separated list of properties to be excluded when loading (for example, "Ref Designator").
CConfigScripting	OnAddComponentEnabled="1"	"OnAddComponentEnabled" automation callback is enabled. Values are 0 or 1 (enabled).
	OnAnnoComponentEnabled="1"	"OnAnnoComponentEnabled" automation callback is enabled. Values are 0 or 1 (enabled).
	OnLoadComponentEnabled="1"	"OnLoadComponentEnabled" automation callback is enabled. Values are 0 or 1 (enabled).

Table B-2. Element/Subelement Attributes (cont.)

Element/Subelement	Attribute	Description
	OnSelectComponentEnabled="1"	"OnSelectComponentEnabled" automation callback is enabled. Values are 0 or 1 (enabled).
	OnViewDocumentEnabled="1"	"OnViewDocumentEnabled" automation callback is enabled. Values are 0 or 1 (enabled).
	OnAfterAddComponentEnabled="1"	"OnAfterAddComponentEnabled" automation callback is enabled. Values are 0 or 1 (enabled).
	OnAfterAnnoComponentEnabled="1"	"OnAfterAnnoComponentEnabled" automation callback is enabled. Values are 0 or 1 (enabled).
CConfigScriptLanguage	Filename=" "	Script filename.
	type="0"	Script programming language. Values are 0 (VBScript), 1 (JScript), 2 (User-Defined).
	progID="VBScript"	User-defined programming language name.
CConfigTableEntry	Assembly="ODBC; ALIAS=SampleLib; CNS={ODBC; DSN=SampleLib}; DDS={Microsoft Access Driver (*.mdb)}; "	Data source configuration file.
	TableName="Capacitors"	Database table name (for example, Capacitors).
CObArray	CXMLTypedPtrArraySize	Value is number of libraries listed in CObArrayElement.
CStringList	ListSize="1"	Number of fields stored in properties. Value is number of fields (decimal format).
	ListItem0="ID"	Lists of different types in .dbc file format. Values are dependent on list type used.
dbc	overallSchema="21"	Internal version number.

Table B-2. Element/Subelement Attributes (cont.)

Element/Subelement	Attribute	Description
	flag="Dx Databook Base Configuration File"	<p>File type with</p> <p>"Dx Databook Overlay Configuration File"</p> <p>or</p> <p>"Dx Databook Overlay Configuration File" BaseConfigurationURL=""></p> <p>values.</p> <p>Note: For flag="Dx Databook Base Configuration File", the value is the <i>.dbb</i> file (see Ch. 4, "Connecting to a DxDataBook Configuration File" for more information).</p>

Example

```
<?xml version="1.0" encoding="UTF-8"?>
<dbc version="1.0">
  <CObject0>
    <CConfig1>
      <CConfigSym2>
        <CObArray3/>
      </CConfigSym2>
    </CConfig1>
    <CObList4/>
  </CObject0>
</dbc>
```

Related Topics

- Chapter 4, “[Defining Data Sources, Libraries, and Symbols in the Configuration File.](#)”

End-User License Agreement

The latest version of the End-User License Agreement is available on-line at:
www.mentor.com/eula

IMPORTANT INFORMATION

USE OF THIS SOFTWARE IS SUBJECT TO LICENSE RESTRICTIONS. CAREFULLY READ THIS LICENSE AGREEMENT BEFORE USING THE SOFTWARE. USE OF SOFTWARE INDICATES YOUR COMPLETE AND UNCONDITIONAL ACCEPTANCE OF THE TERMS AND CONDITIONS SET FORTH IN THIS AGREEMENT. ANY ADDITIONAL OR DIFFERENT PURCHASE ORDER TERMS AND CONDITIONS SHALL NOT APPLY.

END-USER LICENSE AGREEMENT ("Agreement")

This is a legal agreement concerning the use of Software (as defined in Section 2) between the company acquiring the license ("Customer"), and the Mentor Graphics entity that issued the corresponding quotation or, if no quotation was issued, the applicable local Mentor Graphics entity ("Mentor Graphics"). Except for license agreements related to the subject matter of this license agreement which are physically signed by Customer and an authorized representative of Mentor Graphics, this Agreement and the applicable quotation contain the parties' entire understanding relating to the subject matter and supersede all prior or contemporaneous agreements. If Customer does not agree to these terms and conditions, promptly return or, if received electronically, certify destruction of Software and all accompanying items within five days after receipt of Software and receive a full refund of any license fee paid.

1. ORDERS, FEES AND PAYMENT.

- 1.1. To the extent Customer (or if and as agreed by Mentor Graphics, Customer's appointed third party buying agent) places and Mentor Graphics accepts purchase orders pursuant to this Agreement ("Order(s)"), each Order will constitute a contract between Customer and Mentor Graphics, which shall be governed solely and exclusively by the terms and conditions of this Agreement, any applicable addenda and the applicable quotation, whether or not these documents are referenced on the Order. Any additional or conflicting terms and conditions appearing on an Order will not be effective unless agreed in writing by an authorized representative of Customer and Mentor Graphics.
- 1.2. Amounts invoiced will be paid, in the currency specified on the applicable invoice, within 30 days from the date of such invoice. Any past due invoices will be subject to the imposition of interest charges in the amount of one and one-half percent per month or the applicable legal rate currently in effect, whichever is lower. Prices do not include freight, insurance, customs duties, taxes or other similar charges, which Mentor Graphics will invoice separately. Unless provided with a certificate of exemption, Mentor Graphics will invoice Customer for all applicable taxes. Customer will make all payments free and clear of, and without reduction for, any withholding or other taxes; any such taxes imposed on payments by Customer hereunder will be Customer's sole responsibility. Notwithstanding anything to the contrary, if Customer appoints a third party to place purchase orders and/or make payments on Customer's behalf, Customer shall be liable for payment under such orders in the event of default by the third party.
- 1.3. All products are delivered FCA factory (Incoterms 2000) except Software delivered electronically, which shall be deemed delivered when made available to Customer for download. Mentor Graphics retains a security interest in all products delivered under this Agreement, to secure payment of the purchase price of such products, and Customer agrees to sign any documents that Mentor Graphics determines to be necessary or convenient for use in filing or perfecting such security interest. Mentor Graphics' delivery of Software by electronic means is subject to Customer's provision of both a primary and an alternate e-mail address.

2. **GRANT OF LICENSE.** The software installed, downloaded, or otherwise acquired by Customer under this Agreement, including any updates, modifications, revisions, copies, documentation and design data ("Software") are copyrighted, trade secret and confidential information of Mentor Graphics or its licensors, who maintain exclusive title to all Software and retain all rights not expressly granted by this Agreement. Mentor Graphics grants to Customer, subject to payment of applicable license fees, a nontransferable, nonexclusive license to use Software solely: (a) in machine-readable, object-code form; (b) for Customer's internal business purposes; (c) for the term; and (d) on the computer hardware and at the site authorized by Mentor Graphics. A site is restricted to a one-half mile (800 meter) radius. Customer may have Software temporarily used by an employee for telecommuting purposes from locations other than a Customer office, such as the employee's residence, an airport or hotel, provided that such employee's primary place of employment is the site where the Software is authorized for use. Mentor Graphics' standard policies and programs, which vary depending on Software, license fees paid or services purchased, apply to the following: (a) relocation of Software; (b) use of Software, which may be limited, for example, to execution of a single session by a single user on the authorized hardware or for a restricted period of time (such limitations may be technically implemented through the use of authorization codes or similar devices); and (c) support services provided, including eligibility to receive telephone support, updates, modifications, and revisions. For the avoidance of doubt, if Customer requests any change or enhancement to Software, whether in the course of receiving support or consulting services, evaluating Software or

otherwise, any inventions, product improvements, modifications or developments made by Mentor Graphics (at Mentor Graphics' sole discretion) will be the exclusive property of Mentor Graphics.

3. **ESC SOFTWARE.** If Customer purchases a license to use development or prototyping tools of Mentor Graphics' Embedded Software Channel ("ESC"), Mentor Graphics grants to Customer a nontransferable, nonexclusive license to reproduce and distribute executable files created using ESC compilers, including the ESC run-time libraries distributed with ESC C and C++ compiler Software that are linked into a composite program as an integral part of Customer's compiled computer program, provided that Customer distributes these files only in conjunction with Customer's compiled computer program. Mentor Graphics does NOT grant Customer any right to duplicate, incorporate or embed copies of Mentor Graphics' real-time operating systems or other embedded software products into Customer's products or applications without first signing or otherwise agreeing to a separate agreement with Mentor Graphics for such purpose.
4. **BETA CODE.**
 - 4.1. Portions or all of certain Software may contain code for experimental testing and evaluation ("Beta Code"), which may not be used without Mentor Graphics' explicit authorization. Upon Mentor Graphics' authorization, Mentor Graphics grants to Customer a temporary, nontransferable, nonexclusive license for experimental use to test and evaluate the Beta Code without charge for a limited period of time specified by Mentor Graphics. This grant and Customer's use of the Beta Code shall not be construed as marketing or offering to sell a license to the Beta Code, which Mentor Graphics may choose not to release commercially in any form.
 - 4.2. If Mentor Graphics authorizes Customer to use the Beta Code, Customer agrees to evaluate and test the Beta Code under normal conditions as directed by Mentor Graphics. Customer will contact Mentor Graphics periodically during Customer's use of the Beta Code to discuss any malfunctions or suggested improvements. Upon completion of Customer's evaluation and testing, Customer will send to Mentor Graphics a written evaluation of the Beta Code, including its strengths, weaknesses and recommended improvements.
 - 4.3. Customer agrees that any written evaluations and all inventions, product improvements, modifications or developments that Mentor Graphics conceived or made during or subsequent to this Agreement, including those based partly or wholly on Customer's feedback, will be the exclusive property of Mentor Graphics. Mentor Graphics will have exclusive rights, title and interest in all such property. The provisions of this Subsection 4.3 shall survive termination of this Agreement.
5. **RESTRICTIONS ON USE.**
 - 5.1. Customer may copy Software only as reasonably necessary to support the authorized use. Each copy must include all notices and legends embedded in Software and affixed to its medium and container as received from Mentor Graphics. All copies shall remain the property of Mentor Graphics or its licensors. Customer shall maintain a record of the number and primary location of all copies of Software, including copies merged with other software, and shall make those records available to Mentor Graphics upon request. Customer shall not make Software available in any form to any person other than Customer's employees and on-site contractors, excluding Mentor Graphics competitors, whose job performance requires access and who are under obligations of confidentiality. Customer shall take appropriate action to protect the confidentiality of Software and ensure that any person permitted access does not disclose or use it except as permitted by this Agreement. Log files, data files, rule files and script files generated by or for the Software (collectively "Files") constitute and/or include confidential information of Mentor Graphics. Customer may share Files with third parties excluding Mentor Graphics competitors provided that the confidentiality of such Files is protected by written agreement at least as well as Customer protects other information of a similar nature or importance, but in any case with at least reasonable care. Standard Verification Rule Format ("SVRF") and Tcl Verification Format ("TVF") mean Mentor Graphics' proprietary syntaxes for expressing process rules. Customer may use Files containing SVRF or TVF only with Mentor Graphics products. Under no circumstances shall Customer use Software or allow its use for the purpose of developing, enhancing or marketing any product that is in any way competitive with Software, or disclose to any third party the results of, or information pertaining to, any benchmark. Except as otherwise permitted for purposes of interoperability as specified by applicable and mandatory local law, Customer shall not reverse-assemble, reverse-compile, reverse-engineer or in any way derive from Software any source code.
 - 5.2. Customer may not sublicense, assign or otherwise transfer Software, this Agreement or the rights under it, whether by operation of law or otherwise ("attempted transfer"), without Mentor Graphics' prior written consent and payment of Mentor Graphics' then-current applicable transfer charges. Any attempted transfer without Mentor Graphics' prior written consent shall be a material breach of this Agreement and may, at Mentor Graphics' option, result in the immediate termination of the Agreement and licenses granted under this Agreement. The terms of this Agreement, including without limitation the licensing and assignment provisions, shall be binding upon Customer's permitted successors in interest and assigns.
 - 5.3. The provisions of this Section 5 shall survive the termination of this Agreement.
6. **SUPPORT SERVICES.** To the extent Customer purchases support services for Software, Mentor Graphics will provide Customer with available updates and technical support for the Software which are made generally available by Mentor Graphics as part of such services in accordance with Mentor Graphics' then current End-User Software Support Terms located at <http://supportnet.mentor.com/about/legal/>.

7. LIMITED WARRANTY.

7.1. Mentor Graphics warrants that during the warranty period its standard, generally supported Software, when properly installed, will substantially conform to the functional specifications set forth in the applicable user manual. Mentor Graphics does not warrant that Software will meet Customer's requirements or that operation of Software will be uninterrupted or error free. The warranty period is 90 days starting on the 15th day after delivery or upon installation, whichever first occurs. Customer must notify Mentor Graphics in writing of any nonconformity within the warranty period. For the avoidance of doubt, this warranty applies only to the initial shipment of Software under the applicable Order and does not renew or reset, by way of example, with the delivery of (a) Software updates or (b) authorization codes or alternate Software under a transaction involving Software re-mix. This warranty shall not be valid if Software has been subject to misuse, unauthorized modification or improper installation. MENTOR GRAPHICS' ENTIRE LIABILITY AND CUSTOMER'S EXCLUSIVE REMEDY SHALL BE, AT MENTOR GRAPHICS' OPTION, EITHER (A) REFUND OF THE PRICE PAID UPON RETURN OF SOFTWARE TO MENTOR GRAPHICS OR (B) MODIFICATION OR REPLACEMENT OF SOFTWARE THAT DOES NOT MEET THIS LIMITED WARRANTY, PROVIDED CUSTOMER HAS OTHERWISE COMPLIED WITH THIS AGREEMENT. MENTOR GRAPHICS MAKES NO WARRANTIES WITH RESPECT TO: (A) SERVICES; (B) SOFTWARE WHICH IS LICENSED AT NO COST; OR (C) BETA CODE; ALL OF WHICH ARE PROVIDED "AS IS."

7.2. THE WARRANTIES SET FORTH IN THIS SECTION 7 ARE EXCLUSIVE. NEITHER MENTOR GRAPHICS NOR ITS LICENSORS MAKE ANY OTHER WARRANTIES EXPRESS, IMPLIED OR STATUTORY, WITH RESPECT TO SOFTWARE OR OTHER MATERIAL PROVIDED UNDER THIS AGREEMENT. MENTOR GRAPHICS AND ITS LICENSORS SPECIFICALLY DISCLAIM ALL IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT OF INTELLECTUAL PROPERTY.

8. **LIMITATION OF LIABILITY.** EXCEPT WHERE THIS EXCLUSION OR RESTRICTION OF LIABILITY WOULD BE VOID OR INEFFECTIVE UNDER APPLICABLE LAW, IN NO EVENT SHALL MENTOR GRAPHICS OR ITS LICENSORS BE LIABLE FOR INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES (INCLUDING LOST PROFITS OR SAVINGS) WHETHER BASED ON CONTRACT, TORT OR ANY OTHER LEGAL THEORY, EVEN IF MENTOR GRAPHICS OR ITS LICENSORS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. IN NO EVENT SHALL MENTOR GRAPHICS' OR ITS LICENSORS' LIABILITY UNDER THIS AGREEMENT EXCEED THE AMOUNT PAID BY CUSTOMER FOR THE SOFTWARE OR SERVICE GIVING RISE TO THE CLAIM. IN THE CASE WHERE NO AMOUNT WAS PAID, MENTOR GRAPHICS AND ITS LICENSORS SHALL HAVE NO LIABILITY FOR ANY DAMAGES WHATSOEVER. THE PROVISIONS OF THIS SECTION 8 SHALL SURVIVE THE TERMINATION OF THIS AGREEMENT.

9. **LIFE ENDANGERING APPLICATIONS.** NEITHER MENTOR GRAPHICS NOR ITS LICENSORS SHALL BE LIABLE FOR ANY DAMAGES RESULTING FROM OR IN CONNECTION WITH THE USE OF SOFTWARE IN ANY APPLICATION WHERE THE FAILURE OR INACCURACY OF THE SOFTWARE MIGHT RESULT IN DEATH OR PERSONAL INJURY. THE PROVISIONS OF THIS SECTION 9 SHALL SURVIVE THE TERMINATION OF THIS AGREEMENT.

10. **INDEMNIFICATION.** CUSTOMER AGREES TO INDEMNIFY AND HOLD HARMLESS MENTOR GRAPHICS AND ITS LICENSORS FROM ANY CLAIMS, LOSS, COST, DAMAGE, EXPENSE OR LIABILITY, INCLUDING ATTORNEYS' FEES, ARISING OUT OF OR IN CONNECTION WITH CUSTOMER'S USE OF SOFTWARE AS DESCRIBED IN SECTION 9. THE PROVISIONS OF THIS SECTION 10 SHALL SURVIVE THE TERMINATION OF THIS AGREEMENT.

11. INFRINGEMENT.

11.1. Mentor Graphics will defend or settle, at its option and expense, any action brought against Customer in the United States, Canada, Japan, or member state of the European Union which alleges that any standard, generally supported Software product infringes a patent or copyright or misappropriates a trade secret in such jurisdiction. Mentor Graphics will pay any costs and damages finally awarded against Customer that are attributable to the action. Customer understands and agrees that as conditions to Mentor Graphics' obligations under this section Customer must: (a) notify Mentor Graphics promptly in writing of the action; (b) provide Mentor Graphics all reasonable information and assistance to settle or defend the action; and (c) grant Mentor Graphics sole authority and control of the defense or settlement of the action.

11.2. If a claim is made under Subsection 11.1 Mentor Graphics may, at its option and expense, (a) replace or modify Software so that it becomes noninfringing, or (b) procure for Customer the right to continue using Software, or (c) require the return of Software and refund to Customer any license fee paid, less a reasonable allowance for use.

11.3. Mentor Graphics has no liability to Customer if the claim is based upon: (a) the combination of Software with any product not furnished by Mentor Graphics; (b) the modification of Software other than by Mentor Graphics; (c) the use of other than a current unaltered release of Software; (d) the use of Software as part of an infringing process; (e) a product that Customer makes, uses, or sells; (f) any Beta Code; (g) any Software provided by Mentor Graphics' licensors who do not provide such indemnification to Mentor Graphics' customers; or (h) infringement by Customer that is deemed willful. In the case of (h), Customer shall reimburse Mentor Graphics for its reasonable attorney fees and other costs related to the action.

11.4. THIS SECTION IS SUBJECT TO SECTION 8 ABOVE AND STATES THE ENTIRE LIABILITY OF MENTOR GRAPHICS AND ITS LICENSORS AND CUSTOMER'S SOLE AND EXCLUSIVE REMEDY WITH RESPECT TO ANY ALLEGED PATENT OR COPYRIGHT INFRINGEMENT OR TRADE SECRET MISAPPROPRIATION BY ANY SOFTWARE LICENSED UNDER THIS AGREEMENT.

12. TERM.

- 12.1. This Agreement remains effective until expiration or termination. This Agreement will immediately terminate upon notice if you exceed the scope of license granted or otherwise fail to comply with the provisions of Sections 2, 3, or 5. For any other material breach under this Agreement, Mentor Graphics may terminate this Agreement upon 30 days written notice if you are in material breach and fail to cure such breach within the 30 day notice period. If a Software license was provided for limited term use, such license will automatically terminate at the end of the authorized term.
- 12.2. Mentor Graphics may terminate this Agreement immediately upon notice in the event Customer is insolvent or subject to a petition for (a) the appointment of an administrator, receiver or similar appointee; or (b) winding up, dissolution or bankruptcy.
- 12.3. Upon termination of this Agreement or any Software license under this Agreement, Customer shall ensure that all use of the affected Software ceases, and shall return it to Mentor Graphics or certify its deletion and destruction, including all copies, to Mentor Graphics' reasonable satisfaction.
- 12.4. Termination of this Agreement or any Software license granted hereunder will not affect Customer's obligation to pay for products shipped or licenses granted prior to the termination, which amounts shall immediately be payable at the date of termination.
13. **EXPORT.** Software is subject to regulation by local laws and United States government agencies, which prohibit export or diversion of certain products, information about the products, and direct products of the products to certain countries and certain persons. Customer agrees that it will not export Software or a direct product of Software in any manner without first obtaining all necessary approval from appropriate local and United States government agencies.
14. **U.S. GOVERNMENT LICENSE RIGHTS.** Software was developed entirely at private expense. All Software is commercial computer software within the meaning of the applicable acquisition regulations. Accordingly, pursuant to US FAR 48 CFR 12.212 and DFAR 48 CFR 227.7202, use, duplication and disclosure of the Software by or for the U.S. Government or a U.S. Government subcontractor is subject solely to the terms and conditions set forth in this Agreement, except for provisions which are contrary to applicable mandatory federal laws.
15. **THIRD PARTY BENEFICIARY.** Mentor Graphics Corporation, Mentor Graphics (Ireland) Limited, Microsoft Corporation and other licensors may be third party beneficiaries of this Agreement with the right to enforce the obligations set forth herein.
16. **REVIEW OF LICENSE USAGE.** Customer will monitor the access to and use of Software. With prior written notice and during Customer's normal business hours, Mentor Graphics may engage an internationally recognized accounting firm to review Customer's software monitoring system and records deemed relevant by the internationally recognized accounting firm to confirm Customer's compliance with the terms of this Agreement or U.S. or other local export laws. Such review may include FLEXIm or FLEXnet (or successor product) report log files that Customer shall capture and provide at Mentor Graphics' request. Customer shall make records available in electronic format and shall fully cooperate with data gathering to support the license review. Mentor Graphics shall bear the expense of any such review unless a material non-compliance is revealed. Mentor Graphics shall treat as confidential information all information gained as a result of any request or review and shall only use or disclose such information as required by law or to enforce its rights under this Agreement. The provisions of this section shall survive the termination of this Agreement.
17. **CONTROLLING LAW, JURISDICTION AND DISPUTE RESOLUTION.** The owners of the Mentor Graphics intellectual property rights licensed under this Agreement are located in Ireland and the United States. To promote consistency around the world, disputes shall be resolved as follows: This Agreement shall be governed by and construed under the laws of the State of Oregon, USA, if Customer is located in North or South America, and the laws of Ireland if Customer is located outside of North or South America. All disputes arising out of or in relation to this Agreement shall be submitted to the exclusive jurisdiction of Portland, Oregon when the laws of Oregon apply, or Dublin, Ireland when the laws of Ireland apply. Notwithstanding the foregoing, all disputes in Asia (except for Japan) arising out of or in relation to this Agreement shall be resolved by arbitration in Singapore before a single arbitrator to be appointed by the Chairman of the Singapore International Arbitration Centre ("SIAC") to be conducted in the English language, in accordance with the Arbitration Rules of the SIAC in effect at the time of the dispute, which rules are deemed to be incorporated by reference in this section. This section shall not restrict Mentor Graphics' right to bring an action against Customer in the jurisdiction where Customer's place of business is located. The United Nations Convention on Contracts for the International Sale of Goods does not apply to this Agreement.
18. **SEVERABILITY.** If any provision of this Agreement is held by a court of competent jurisdiction to be void, invalid, unenforceable or illegal, such provision shall be severed from this Agreement and the remaining provisions will remain in full force and effect.
19. **MISCELLANEOUS.** This Agreement contains the parties' entire understanding relating to its subject matter and supersedes all prior or contemporaneous agreements, including but not limited to any purchase order terms and conditions. Some Software may contain code distributed under a third party license agreement that may provide additional rights to Customer. Please see the applicable Software documentation for details. This Agreement may only be modified in writing by authorized representatives of the parties. All notices required or authorized under this Agreement must be in writing and shall be sent to the person who signs this Agreement, at the address specified below. Waiver of terms or excuse of breach must be in writing and shall not constitute subsequent consent, waiver or excuse.