

IBM Visual Warehouse for Windows NT

IBM

# Managing DataGuide

Version 5 Release 2



IBM Visual Warehouse for Windows NT

IBM

# Managing DataGuide

Version 5 Release 2

**Note**

Before using this information and the product it supports, be sure to read the general information under "Notices" on page 163.

**Fifth Edition (November 1998)**

This edition replaces and makes obsolete the previous edition, SC26-3362-03. The technical changes for this edition are indicated by a vertical bar to the left of a change.

This edition applies to Version 5 Release 2 of Visual Warehouse (5639-VW5), and to any subsequent releases until otherwise indicated in new editions or technical newsletters. Make sure you are using the correct edition for the level of the product.

Order publications through your IBM representative or the IBM branch office serving your locality.

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

**Copyright International Business Machines Corporation 1994, 1998. All rights reserved.**

Note to U.S. Government Users — Documentation related to restricted rights — Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract with IBM Corp.

---

## Contents

<b>Introduction</b> . . . . .	vi
Who should use this book . . . . .	vi
How to use this book . . . . .	vi
This book describes administrator tasks . . . . .	vi
You can perform DataGuide tasks with the user interface or tag language . . . . .	vii
How to send your comments . . . . .	viii
<b>Chapter 1. Setting up the DataGuide information catalog</b> . . . . .	1
Authorizing users to DataGuide . . . . .	2
Creating the information catalog . . . . .	3
Registering a server node and information catalog . . . . .	13
Opening a DataGuide for Windows information catalog . . . . .	16
Ensuring that users can start programs from DataGuide . . . . .	16
Authorizing DataGuide users to manage objects . . . . .	18
Setting and changing status values for comments . . . . .	19
<b>Chapter 2. Organizing your information resources</b> . . . . .	21
Understanding DataGuide information categories and object types . . . . .	21
Creating your own object types . . . . .	23
Updating an object type . . . . .	32
Deleting an object type . . . . .	36
<b>Chapter 3. Populating the catalog with information</b> . . . . .	37
Creating an object . . . . .	37
Copying an object . . . . .	39
Updating an object . . . . .	40
Deleting an object . . . . .	42
<b>Chapter 4. Making the information catalog convenient for users</b> . . . . .	44
Grouping objects by subject . . . . .	44
Creating a linked relationship between objects . . . . .	47
Associating contact names with objects . . . . .	49
Associating comments and objects . . . . .	51
Associating a program with object types . . . . .	55
Creating a dictionary facility . . . . .	62
Creating a support facility . . . . .	63
<b>Chapter 5. Expanding and automating your information catalog</b> . . . . .	64
Extracting descriptive data from other sources . . . . .	64
Logging deletions from your information catalog . . . . .	70
Importing and exporting tag language files . . . . .	70
<b>Chapter 6. Exchanging metadata with other products</b> . . . . .	77
Exchanging and synchronizing metadata with Visual Warehouse and DB2 OLAP Server . . . . .	77

Exchanging MDIS-conforming metadata with other products . . . . .	83
<b>Chapter 7. Maintaining DataGuide</b> . . . . .	91
Preventing problems . . . . .	91
Solving problems . . . . .	93
<b>Appendix A. DataGuide extract programs</b> . . . . .	98
Extract programs supplied with DataGuide . . . . .	98
Planning to run extract programs . . . . .	99
<b>Appendix B. Predefined DataGuide object types</b> . . . . .	100
Accessing DataGuide sample data . . . . .	100
Predefined object type models . . . . .	102
Predefined object type descriptions . . . . .	106
<b>Appendix C. Tag language</b> . . . . .	115
Rules for writing tag language files . . . . .	115
How DataGuide reads tag language files . . . . .	116
Valid data types for DataGuide descriptive data . . . . .	117
How to read the tag language syntax diagrams . . . . .	118
ACTION.OBJINST . . . . .	118
ACTION.OBJTYPE . . . . .	123
ACTION.RELATION . . . . .	127
COMMENT . . . . .	128
COMMIT . . . . .	129
DISKCNTL . . . . .	130
INSTANCE . . . . .	131
NL . . . . .	136
OBJECT . . . . .	136
PROPERTY . . . . .	143
RELTYPE . . . . .	146
TAB . . . . .	147
<b>Appendix D. What a tag language file should look like</b> . . . . .	149
Start your tag language file with DISKCNTL . . . . .	149
Define your additions, changes, and deletions . . . . .	149
Committing changes to the database . . . . .	151
Putting comments in the tag language file . . . . .	151
<b>Appendix E. Performing DataGuide functions from the command line</b> . . . . .	152
Invoking DataGuide from the command line . . . . .	152
Creating an information catalog from the command line . . . . .	154
<b>Glossary</b> . . . . .	158
<b>Notices</b> . . . . .	163
Trademarks . . . . .	163

**Bibliography** . . . . . 165  
**Index** . . . . . 166

---

## Introduction

This book is designed to help administrators adapt and tailor DataGuide, the Visual Warehouse information catalog, to meet their organization's needs.

---

### Who should use this book

This book assumes that you're familiar with the tasks that users perform with DataGuide, such as searching and browsing for information. These tasks are described in *Using DataGuide* which is located in the **DataGuide** folder.

---

### How to use this book

DataGuide administrators need to ensure that the descriptive data users need is available, that it's easy to find and use, that it's as current as it needs to be, and that it's protected from unauthorized access.

### This book describes administrator tasks

The tasks of an administrator are in these categories:

#### **Setting up the DataGuide information catalog**

You authorize users, create the information catalog, set up some sample information for your users, and give them access to the software and information resources they need. Description of these tasks begins on page 1.

#### **Organizing your information resources**

You determine what kinds of information resources your organization wants to describe in your information catalog. You create object types that describe the characteristics of different kinds of information, and update and delete these object types as needed. Description of these tasks begins on page 21.

#### **Populating the catalog with information**

You create objects of various types and place them in your information catalog. To do this, you translate information into terms with which users are familiar. Description of these tasks begins on page 37.

#### **Making the information catalog convenient for users**

You group objects together to make them easier to browse, add contact names so that users can find someone to ask about the information, and set up programs so that users can start them and retrieve the information they want. You can maintain a *support facility* to inform users about changes in the information catalog, and a *dictionary facility* as a central location for quick reference to terminology used in the information catalog. Description of these tasks begins on page 44.

#### **Expanding and automating your information catalog**

You use the DataGuide tag language to make it easier to work with large amounts of descriptive data at once. To do this, you *import* and *export* tag language files. You might extract descriptive data from your organization's existing database



catalogs, modeling tools, and user files. Application programmers can write their own customized extract program. You combine information catalogs to keep descriptive data current and appropriately synchronized with its sources.

You can keep a log of objects, object types, or relationships that are deleted from your information catalog. You can transfer the log to a tag language file and use it to duplicate the deletions in other information catalogs, for example, "shadow" information catalogs in a distributed environment. Description of these tasks begins on page 64.

### **Maintaining DataGuide**

You might also perform some routine database administration tasks, such as backing up the DataGuide information catalog, although these tasks are not part of the actual management of DataGuide. You prevent or solve some of the problems that your users could have with DataGuide. Description of these tasks begins on page 91.

## **You can perform DataGuide tasks with the user interface or tag language**

DataGuide provides a user interface for performing DataGuide tasks. DataGuide also provides a tag language, which you can use to perform many of the same tasks. The tag language is more difficult to use in that it requires you to learn syntax rules and use them to code a tag language file, but it is especially powerful for performing tasks in bulk.

Throughout this book, DataGuide tasks are described first as you would perform them using the user interface. When there is a tag language equivalent for performing the DataGuide task, it is presented following the user interface description, under a heading "<task> using the DataGuide tag language."

### **The user interface**

To use the user interface, start from the administrator's DataGuide Catalog window, shown here.



## The tag language

The easiest way to use the tag language is to cut and paste the tag language templates that are provided online directly into your tag language file. To use an online tag language template:

- 1 Press F1 from any product window (after you've opened an information catalog). A help window opens.
- 2 Click on the **Contents** push button at the bottom of the help window.
- 3 Double-click on the **Tag language template you can copy and modify** link that follows the task you want to perform.  
The tag language panel opens for that task.
- 4 Select **Services > Copy** from the menu bar to copy the entire panel to the clipboard.
- 5 Paste the template from the clipboard into the desired tag file.
- 6 Edit the variables in the template. Short descriptions for these variables are provided online. For more lengthy descriptions, see the section on the task you want to perform within this book or Appendix C, "Tag language" on page 115 for the complete tag language reference.

---

## How to send your comments

Your feedback helps IBM to provide quality information. If you have any comments about this book or other Visual Warehouse publications, visit the following Web site:

<http://www.software.ibm.com/data/vw/>

The Web site has a feedback page where you can enter and submit your comments.

## Chapter 1. Setting up the DataGuide information catalog

**If you plan to use DataGuide in a LAN environment:** After installing DataGuide (see *Planning and Installing Visual Warehouse and DataGuide*), you perform seven main tasks:

- 1 Authorize yourself and your users to use your DataGuide information catalog.
- 2 When you install Visual Warehouse, a default information catalog is created for you on DB2 UDB for Windows NT. If you do not use the default information catalog, you can create and identify the DataGuide 5.2 information catalog depending on your configuration, as described in Table 1.

Table 1. Creating and identifying information catalogs by configuration

DataGuide for Windows Administrator on DB2 UDB for Windows NT server	DataGuide for Windows Administrator on client <sup>1</sup>
1 Create a DataGuide 5.2 information catalog on the server.	1 Register the server node and remote information catalog.
2 Register the server node and remote information catalog on DataGuide for Windows User client workstations.	2 Create the DataGuide 5.2 information catalog (by either defining a new information catalog or migrating an existing information catalog).
	3 Register the server node and remote information catalog on DataGuide for Windows User client workstations.

**Note:**

1. With DB2 for AIX, DB2 for MVS, DB2 for OS/390, DB2 for OS/2, DB2 for OS/400, DB2 PE, DB2 UDB EEE, or DB2 UDB for Windows NT remote server

- 3 Open the DataGuide information catalog as an administrator.
- 4 Ensure that users can launch applications to access the information that is described in the information catalog.
- 5 Authorize specific users to perform object management tasks.
- 6 Set the status choices available to users when they create comments in your information catalog.
- 7 If you haven't done so already, create a sample information catalog to help your users learn to use DataGuide. (This task is described as part of the installation process in *Planning and Installing Visual Warehouse and DataGuide*, and is also reproduced in "Accessing DataGuide sample data" on page 100 for your convenience.)

**If you plan to use DataGuide as a stand-alone product:** After installing DataGuide (see *Planning and Installing Visual Warehouse and DataGuide*):

- 1 Create the database that you plan to use for your information catalog.

## Authorizing users to DataGuide

- 2 Create the DataGuide 5.2 information catalog (this information catalog will be unable to communicate with information catalogs on other workstations).
- 3 Complete steps 3, 6, and 7 from the previous task list.

---

## Authorizing users to DataGuide

You must authorize your users to use both DataGuide for Windows and DataGuide for the Web.

### Authorizing users to DataGuide for Windows

Start by deciding who will act as the primary and backup administrators of your information catalog. Depending on your hardware and software configuration and where you want to place your DataGuide information catalog, you might need to get authorization to add user IDs and passwords from other data processing administrators.

Table 2 shows the level of authority required to create user IDs and passwords for each DataGuide configuration.

---

*Table 2. Database storage locations for DataGuide information catalogs and the authority required to administer them*

Database location for DataGuide information catalog	Authority required
DB2 for OS/2 server, file server	LAN domain administrator
DB2 for OS/2 server, non-file server	System administrator for the server
DATABASE 2 for MVS (DB2 for MVS)	RACF administrator
DATABASE 2 for AIX (DB2 for AIX), DATABASE 2 Parallel Edition for AIX (DB2 PE) <sup>1</sup> , or DB2 Universal Database Enterprise Extended Edition (DB2 UDB EEE) <sup>1</sup>	AIX administrator
DATABASE 2 for OS/400 (DB2 for OS/400)	OS/400 security officer
DATABASE 2 for Windows NT	Administrator for the server

**Notes:**

1. If you plan to store your DataGuide information catalog on DB2 PE or DB2 UDB EEE (DB2 Universal Database Enterprise-Extended Edition—this DB2 UDB product is the follow-on product to DB2 PE), follow the instructions for DB2 for AIX. Defining an information catalog and all metadata management on DB2 PE or DB2 UDB EEE is identical to doing so for DB2 for AIX.

Unless a specific DB2 product is named, throughout this book the generic term “DB2” is used to denote the DB2 database that stores your DataGuide information catalog on your platform of choice.

---

**Note for DataGuide for the Web users:** See *Planning and Installing Visual Warehouse and DataGuide* for information on authorizing users to DataGuide for the Web.

## Creating the information catalog

---

### Creating the information catalog

You can create a DataGuide information catalog on one of these database management systems:

#### **DB2 for OS/2**

You need to install the administration tools for DB2 for OS/2.

#### **DB2 for MVS**

You must have the DB2 Connect product installed on your workstations to use DB2 for MVS.

#### **DB2 for OS/390**

DB2 for OS/390 is the follow-on product to DB2 for MVS. You must have the DB2 Connect product installed on your workstations to use DB2 for OS/390.

#### **DB2 for OS/400**

You must have the DB2 Connect product installed on your workstations to use DB2 for OS/400.

#### **DB2 for AIX**

You must have TCP/IP installed on your workstations to use DB2 for AIX.

#### **DB2 PE or DB2 UDB EEE**

If you plan to store your DataGuide information catalog on DB2 PE, or the follow-on release to DB2 PE, DB2 Universal Database Enterprise Extended Edition, follow the instructions for DB2 for AIX. Defining an information catalog and all metadata management on DB2 PE or DB2 UDB EEE is very similar to doing so for DB2 for AIX. Exceptions are noted in the steps on “Defining an information catalog on DB2 for AIX or DB2 PE or DB2 UDB EEE” on page 8.

#### **DB2 UDB for Windows NT**

You must have TCP/IP or NetBIOS installed on your workstations to use DB2 UDB for Windows NT.

#### **DB2 for Windows 95**

You must have DB2 for Windows 95 Version 5.0 single-user installed on your workstation. DB2 for Windows 95 information catalogs are stand-alone on your workstation and cannot be accessed from other workstations.

The database management system you choose depends on how large you want your DataGuide information catalog to be.

You can define your information catalog using the DataGuide user interface or from an MS-DOS command prompt. To define your information catalog using the DataGuide user interface, use this chapter. To define your information catalog from an MS-DOS command prompt, see “Creating an information catalog from the command line” on page 154.

### Defining an information catalog on DB2 for OS/2

Table 3 on page 4 describes the tasks you need to complete before defining a DataGuide information catalog on DB2 for OS/2.

## Creating the information catalog

*Table 3. Preparing to define an information catalog on DB2 for OS/2*

Task	Who	
	Remote database administrator	You
Create the database in which the information catalog will be stored.	X	
Ensure that you have SYSADM authority to define the new information catalog.	X	

### **To define the information catalog:**

- 1** Click on the **Start** push button.
- 2** Select **Programs**.
- 3** Select **Visual Warehouse**.
- 4** Select **DataGuide**.
- 5** Double-click or select the **Create Information Catalog** icon.  
The Create Information Catalog window opens.
- 6** From the **Select type of information catalog** list, select **DB2 for OS/2**.
- 7** Click on the **OK** push button.  
The Define Catalog on DB2 for OS/2 window opens.
- 8** In the **Information catalog name** field, type the alias name for the remote database that is cataloged on your local workstation.
- 9** Select a character from the **Not-applicable symbol** list:
  - a** Click on the down arrow to display a list of valid symbols.
  - b** Select the symbol you want to use from the list.
- 10** In the **Primary administrator's user ID** field, type the user ID of the person who will be the primary administrator of DataGuide. This user ID must have SYSADM authority.
- 11** In the **Backup administrator's user ID** field, type the user ID of the person who will back up the primary administrator.
- 12** Select the **Import common object types** check box to have DataGuide initialize your information catalog with object types you can use to exchange metadata with other conforming products.
- 13** Click on the **Define** push button.  
The Connect to Information Catalog window opens.
- 14** In the **User ID** field, type the LAN user ID (specified with UPM on the remote workstation).

## Creating the information catalog

**15** In the **Password** field, type the password for the user ID you entered in the **User ID** field.

**16** Click on the **Connect** push button.

Your new catalog is defined, and two DataGuide program icons are created in your **DataGuide** folder. One icon represents administrator functions, which you can use only if you have DataGuide Administrator installed on your workstation. The other icon represents user functions.

If you receive an error message stating that DataGuide could not import the common object types, follow the steps for initializing your information catalog with the predefined object types (see page 101) in order to add the object types to your new information catalog.

### Defining an information catalog on DB2 for OS/390

Table 4 describes the tasks you or your remote database administrator need to complete before defining a DataGuide information catalog on DB2 for OS/390.

Task	Who	
	Remote database administrator	You
Create the database in which the information catalog will be stored.  DataGuide provides a sample JCL file that the database administrator can change to create the database, storage groups, and table spaces. The sample file is called DGCRTDB.JCL and resides in the \VWLIB\BIN directory in the path where DataGuide is installed.	X	
Ensure that you have SYSADM authority to define the new information catalog.	X	
Ask your database administrator for the following names:  Database name Storage group name for tables Storage group name for indexes	X	X

#### To define the information catalog:

- 1** Click on the **Start** push button.
- 2** Select **Programs**.
- 3** Select **Visual Warehouse**.
- 4** Select **DataGuide**.
- 5** Double-click or select the **Create Information Catalog** icon.

The Create Information Catalog window opens.

## Creating the information catalog

- 6** From the **Select type of information catalog** list, select **DB2 for MVS**.
- 7** Click on the **OK** push button.

The Define Catalog on DB2 for MVS window opens.
- 8** In the **Information catalog name** field, type the alias name for the remote database that is cataloged on your local workstation.
- 9** In the **DB2 database name** entry field, type the name of the DB2 database.
- 10** In the **Name of storage group for tables** entry field, type the name of the storage group that you will use for tables.
- 11** In the **Name of storage group for indexes** entry field, type the name of the storage group that you will use for indexes.
- 12** Select a character from the **Not-applicable symbol** list:
  - a** Click on the down arrow to display a list of valid symbols.
  - b** Select the symbol you want to use.
- 13** In the **Primary administrator's user ID** field, type the user ID of the person who will be the primary administrator of DataGuide. This user ID must have SYSADM authority on DB2 for MVS.
- 14** In the **Backup administrator's user ID** field, type the user ID of the person who will back up the primary administrator.
- 15** If you want to save the property values of each object in uppercase, select the **Save object values in upper case** check box. If you select this box, the values are stored in the MVS database in uppercase, but you can enter the values in lowercase when you search for them using DataGuide.
- 16** Select the **Import common object types** check box to have DataGuide initialize your information catalog with object types you can use to exchange metadata with other conforming products.
- 17** Click on the **Define** push button.

The Connect to Information Catalog window opens.
- 18** In the **User ID** field, type your RACF user ID.
- 19** In the **Password** field, type the password for the user ID that you entered in the **User ID** field.
- 20** Click on the **Connect** push button.

Your new catalog is defined, and two DataGuide program icons are created in your **DataGuide** folder. One icon represents administrator functions, which you can use only if you have DataGuide Administrator installed on your workstation. The other icon represents user functions.

If you receive an error message stating that DataGuide could not import the common object types, follow the steps for initializing your information catalog with



## Creating the information catalog

the predefined object types (see page 101) in order to add the object types to your new information catalog.

### Defining an information catalog on DB2 for OS/400

Table 5 describes the tasks you or your remote database administrator need to complete before defining a DataGuide information catalog on DB2 for OS/400.

Task	Who	
	Remote database administrator	You
Create the database in which the information catalog will be stored.	X	
Ensure that you have ALLOBJ authority to define the new information catalog.	X	

#### To define the information catalog:

- 1 Click on the **Start** push button.
- 2 Select **Programs**.
- 3 Select **Visual Warehouse**.
- 4 Select **DataGuide**.
- 5 Double-click or select the **Create Information Catalog** icon.  
The Create Information Catalog window opens.
- 6 From the **Select type of information catalog** list, select **DB2 for OS/400**.
- 7 Click on the **OK** push button.  
The Define Catalog on DB2 for OS/400 window opens.
- 8 In the **Information catalog name** field, type the alias name of the remote database that is cataloged on your local workstation.
- 9 Select a character from the **Not-applicable symbol** list:
  - a Click on the down arrow to display a list of valid symbols.
  - b Select the symbol you want to use.
- 10 In the **Primary administrator's user ID** field, type the user ID of the person who will be the primary administrator of DataGuide.  
The user ID must have ALLOBJ authority on DB2 for OS/400.
- 11 In the **Backup administrator's user ID** field, type the user ID of the person who will back up the primary administrator.

## Creating the information catalog

- 12** Select the **Import common object types** check box to have DataGuide initialize your information catalog with object types you can use to exchange metadata with other conforming products.
- 13** Click on the **Define** push button.  
The Connect to Information Catalog window opens.
- 14** In the **User ID** field, type your OS/400 user ID.
- 15** In the **Password** field, type the password for the user ID you entered in the **User ID** field.
- 16** Click on the **Connect** push button.

Your new catalog is defined, and two DataGuide program icons are created in your **DataGuide** folder. One icon represents administrator functions, which you can use only if you have DataGuide Administrator installed on your workstation. The other icon represents user functions.

If you receive an error message stating that DataGuide could not import the common object types, follow the steps for initializing your information catalog with the predefined object types (see page 101) in order to add the object types to your new information catalog.

## Defining an information catalog on DB2 for AIX or DB2 PE or DB2 UDB EEE

Table 6 on page 9 describes the tasks you or your remote database administrator need to complete before defining a DataGuide information catalog on DB2 for AIX or DB2 PE. These steps also apply to the DB2 Universal Database Enterprise Extended Edition (DB2 UDB EEE), which is the subsequent release to DB2 PE.

## Creating the information catalog

Table 6. Preparing to define an information catalog on DB2 for AIX or DB2 PE or DB2 UDB EEE		
Task	Who	
	Remote database administrator	You
Create or identify the database in which the information catalog will be stored. If the DB2 for AIX database is created on the remote host, the database administrator might want to specify an authentication level for database security reasons.	X	
If you are defining a DB2 PE information catalog, enter the following SQL command from a DB2 Command Line Processor: <code>CREATE NODEGROUP FLG32K ON NODE number</code> where <code>number</code> is the identifying number of the node.	X	
Ensure that you have SYSADM authority to define the new information catalog.	X	
Ask your database administrator for the name of the database.	X	X
If you are defining a DB2 UDB EEE information catalog, enter the following SQL commands from a DB2 Command Line Processor: <code>CREATE NODEGROUP FLG32K ON NODE number</code> <code>CREATE REGULAR TABLESPACE FLG32K IN NODEGROUP FLG32K</code> <code>MANAGED BY SYSTEM USING ('FLG32K')</code> where <code>number</code> is the identifying number of the node.	X	

### To define the information catalog:

- 1 Click on the **Start** push button.
- 2 Select **Programs**.
- 3 Select **Visual Warehouse**.
- 4 Select **DataGuide**.
- 5 Double-click or select the **Create Information Catalog** icon.  
The Create Information Catalog window opens.
- 6 From the **Select type of information catalog** list, select **DB2 or DB2 PE for AIX**.
- 7 Click on the **OK** push button.  
The Define Catalog on DB2 or DB2 PE for AIX window opens.
- 8 In the **Information catalog name** field, type the alias name of the remote database that is cataloged on your local workstation.
- 9 Select a character from the **Not-applicable symbol** list:
  - a Click on the down arrow to display a list of valid symbols.

## Creating the information catalog

- b** Select the symbol you want to use.
- 10** In the **Primary administrator's user ID** field, type the user ID of the person who will be the primary administrator of DataGuide.  
This user ID must have SYSADM authority.
- 11** In the **Backup administrator's user ID** field, type the user ID of the person who will back up the primary administrator.
- 12** Select the **Import common object types** check box to have DataGuide initialize your information catalog with object types you can use to exchange metadata with other conforming products.
- 13** Click on the **Define** push button.  
The Connect to Information Catalog window opens.
- 14** In the **User ID** field, type your AIX user ID.
- 15** In the **Password** field, type the password for the user ID you entered in the **User ID** field.  
Passwords for DB2 for AIX, DB2 PE, and DB2 UDB EEE databases are case-sensitive; you must type them exactly as specified.
- 16** Click on the **Connect** push button.  
Your new catalog is defined, and two DataGuide program icons are created in your **DataGuide** folder. One icon represents administrator functions, which you can use only if you have DataGuide Administrator installed on your workstation. The other icon represents user functions.  
  
If you receive an error message stating that DataGuide could not import the common object types, follow the steps for initializing your information catalog with the predefined object types (see page 101) in order to add the object types to your new information catalog.

## Defining an information catalog on DB2 UDB for Windows NT or Windows 95

Table 7 on page 11 describes the tasks you or your remote database administrator need to complete before defining a DataGuide information catalog on DB2 UDB for Windows NT or DB2 for Windows 95. DB2 for Windows 95 information catalogs are stand-alone on your workstation and cannot be accessed from other workstations.

## Creating the information catalog

<i>Table 7. Preparing to define an information catalog on DB2 UDB for Windows NT or Windows 95</i>		
Task	Who	
	Remote database administrator	You
Create or identify the database in which the information catalog will be stored. If the DB2 UDB for Windows NT database is created on the remote host, the database administrator might want to specify an authentication level for database security reasons.	X	
Ensure that you have administrator authority to define the new information catalog.	X	
Ask your database administrator for the name of the database.	X	X

### **To define the information catalog:**

- 1** Click on the **Start** push button.
- 2** Select **Programs**.
- 3** Select **Visual Warehouse**.
- 4** Select **DataGuide**.
- 5** Double-click or select the **Create Information Catalog** icon.  
The Create Information Catalog window opens.
- 6** From the **Select type of information catalog** list, select **DB2 UDB for Windows NT or 95**.
- 7** Click on the **OK** push button.  
The Define Catalog on DB2 UDB for Windows NT or 95 window opens.
- 8** In the **Information catalog name** field, type the name you want to assign to the local information catalog or the alias name of the remote database that is cataloged on your local workstation.
- 9** Select a character from the **Not-applicable symbol** list:
  - a** Click on the down arrow to display a list of valid symbols.
  - b** Select the symbol you want to use.
- 10** In the **Primary administrator's user ID** field, type the user ID of the person who will be the primary administrator of DataGuide.  
This user ID must have SYSADM authority.
- 11** In the **Backup administrator's user ID** field, type the user ID of the person who will back up the primary administrator.

## Creating the information catalog

- 12** Select the **Import common object types** check box to have DataGuide initialize your information catalog with object types you can use to exchange metadata with other conforming products.
- 13** Click on the **Define** push button.  
The Connect to Information Catalog window opens.
- 14** In the **User ID** field, type the user ID required by the database storing your information catalog:
  - DB2 UDB for Windows NT (local)**  
Windows NT user ID
  - DB2 UDB for Windows NT (remote)**  
LAN user ID, specified with User Manager on the remote workstation
  - DB2 for Windows 95**  
Windows 95 user ID
- 15** In the **Password** field, type the password for the user ID you entered in the **User ID** field.  
  
Passwords for DB2 UDB for Windows NT and DB2 for Windows 95 databases are case-sensitive; you must type them exactly as specified.
- 16** Click on the **Connect** push button.  
  
Your new catalog is defined, and two DataGuide program icons are created in your **DataGuide** folder. One icon represents administrator functions, which you can use only if you have DataGuide Administrator installed on your workstation. The other icon represents user functions.  
  
If you receive an error message stating that DataGuide could not import the common object types, follow the steps for initializing your information catalog with the predefined object types (see page 101) in order to add the object types to your new information catalog.

## Migrating an information catalog to DataGuide 3.1

You can migrate a DataGuide 1.1 for Windows information catalog to DataGuide 3.1 for Windows using the following steps.

### *To migrate an information catalog:*

- 1** Export all object types, objects, and relationships from your existing DataGuide information catalog (see “Exporting metadata” on page 72 for more information).
- 2** Create a new DataGuide Version 3.1 information catalog. Select the **Import common object types** check box on the Define catalog window.  
  
If you receive an error message stating that DataGuide could not import the common object types, follow the steps for initializing your information catalog with the predefined object types (see page 101) in order to add the object types to your new information catalog.

## Registering a server node and information catalog

- 3 Edit the tag language file containing the metadata from your DataGuide Version 1.1 information catalog to make the object type property names conform to the Version 3.1 object type specifications. You must update the object type property names in the object type definitions, object instance definitions, and within the UUI reference section of relationships (see Appendix C, “Tag language” on page 115 for reference information about the DataGuide tag language). For a complete mapping of the Version 1.1 object type definitions to the Version 3.1 object type definitions, see the file README.MIG in the VWSWIN\DGWIN\TYPES directory on the drive where you installed DataGuide.
- 4 Import your edited tag language file into the Version 3.1 information catalog (see “Importing tag language files” on page 71).

### Migrating a DataGuide 3.1 information catalog to DataGuide 5.2

To migrate a DataGuide 3.1 information catalog to a DataGuide 5.2 information catalog, run the following command from the DB2 Command Line Processor:

```
DG_UPGRD.BAT catalog_name userid password
```

where:

*catalog\_name* is the name of the information catalog you are upgrading

*userid* is the user ID of the DataGuide administrator or the database administrator

*password* is the password for the user ID

---

## Registering a server node and information catalog

An information catalog can either be local (stored on your workstation) or remote. If an information catalog is remote, you must register it and the server where it resides. Table 8 on page 14 describes the tasks you or your remote database administrator need to complete before registering a server node and information catalog.

## Registering a server node and information catalog

*Table 8. Preparing to register a server node and information catalog*

Task	Who	
	Remote database administrator	You
<p>Edit the appropriate protocol command file for your DataGuide users' environment. There are four protocol command files available:</p> <p style="padding-left: 40px;">           DGNTBIOS uses NetBIOS protocol            DGTCP/IP uses TCP/IP protocol            DGCPIC uses CPIC protocol            DGIPXSPX uses IPX/SPX protocol         </p> <p>The command files in this list are provided as examples, but you are not limited to those. If you change the command files or create new ones, they must have a file extension of BAT and reside in the \\VWSLIB\PROTOCOL directory.</p> <p>You need to modify each file to add the node name where the remote information catalog resides.</p>		X
Ask your database administrator for the node name of the remote server.	X	X
Ask your database administrator for the database name under which your information catalog is cataloged on the remote server.	X	X

### **To register a server node:**

- 1** Click on the **Start** push button.
- 2** Select **Programs**.
- 3** Select **Visual Warehouse**.
- 4** Select **DataGuide**.
- 5** Double-click or select the **Register Server Node and Information Catalog** icon.  
The Register Server Node and Information Catalog window opens.
- 6** Select the **Register new server node** radio button, and click on the **OK** push button.  
The Register New Server Node window opens.
- 7** Select a command file that is appropriate for your environment. The **Contents of command file** field displays the protocol and database commands that will establish the connection between DataGuide and the remote database where your information catalog is stored.
- 8** Click on the **Register** push button.
- 9** The Register Server Node and Information Catalog window remains open. You can register other server nodes or information catalogs, or click on the **Cancel** push button to close the window.



## Registering a server node and information catalog

### *To register a remote information catalog:*

- 1 Click on the **Start** push button.
- 2 Select **Programs**.
- 3 Select **Visual Warehouse**.
- 4 Select **DataGuide**.
- 5 Double-click or select the **Register Server Node and Information Catalog** icon.  
The Register Server Node and Information Catalog window opens.
- 6 Select the **Register new information catalog** button, and click on the **OK** push button.  
The Register New Information Catalog window opens.
- 7 In the **Information catalog name** field, type the name the remote catalog will have on your local workstation.
- 8 In the **Server information catalog name** field, type the name the information catalog has on the remote server.
- 9 From the **Server node ID** list, select the server node or DDCS or DB2 Connect gateway workstation where the remote information catalog is located.
- 10 Click on the **Register** push button.  
The Connect to Information Catalog window opens.
- 11 In the **User ID** field, type the user ID required by the database storing your information catalog:
  - DB2 for OS/2 (remote)**  
LAN user ID, specified with UPM on the remote workstation
  - DB2 for MVS or DB2 for OS/390**  
RACF user ID
  - DB2 for OS/400**  
OS/400 user ID
  - DB2 for AIX**  
AIX user ID
  - DB2 PE or DB2 UDB EEE**  
AIX user ID
  - DB2 UDB for Windows NT (remote)**  
LAN user ID, specified with User Manager on the remote workstation
- 12 In the **Password** field, type the password for the user ID you entered in the **User ID** field.  
  
Passwords for DB2 for AIX, DB2 PE, DB2 UDB EEE, and DB2 UDB for Windows NT databases are case-sensitive; you must type them exactly as specified.
- 13 Click on the **Connect** push button.

## Ensuring users can launch programs from DataGuide

Your information catalog is ready to use with DataGuide 5.2 Two DataGuide program icons are created in your **DataGuide** folder. One icon represents administrator functions, which you can use only if you have DataGuide Administrator installed on your workstation. The other icon represents user functions.

- 14 The Register Server Node and Information Catalog window remains open. You can register other server nodes or information catalogs, or click on the **Cancel** push button to close the window.

---

## Opening a DataGuide for Windows information catalog

You start working with DataGuide by opening an information catalog.

- 1 Click on the **Start** push button.
- 2 Select **Programs**.
- 3 Select **Visual Warehouse**.
- 4 Select **DataGuide**.
- 5 Double-click on the icon representing the information catalog you want to open.  
The Open DataGuide Catalog window opens.
- 6 In the **User ID** field, type the user ID required for the operating system on which your information catalog resides.
- 7 In the **Password** field, type the password for the user ID you entered in the **User ID** field.  
  
Passwords for DB2 for AIX, DB2 PE, DB2 UDB EEE, DB2 UDB for Windows NT, and DB2 for Windows 95 databases are case sensitive; you must type them exactly as specified.
- 8 Click on the **Open** push button. The DataGuide Catalog window opens.

You can also open your information catalog from an MS-DOS prompt; see "Invoking DataGuide from the command line" on page 152 for more information.

---

## Ensuring that users can start programs from DataGuide

You set up the objects in your information catalog so that your users can run application programs to work with the actual information that those DataGuide objects describe. Users can run the application programs they are familiar with, including the programs that were originally used to create the information.

Ensure that the following requirements are met:

- Your users need the appropriate application software installed on their workstations or LAN.

## Ensuring users can launch programs from DataGuide

- A program that can be started from the command line with the command `start program_name` and without a path can be launched by DataGuide users, regardless of where the program is installed.

Many programs write their path to the program registry when they are installed. The path is then retrieved by the `start` command. If a program does not write its path to the program registry, it might be necessary for you to add to the Path environment variable on your users' workstations the directory path where the program is installed.

- Your users need the necessary authorization to the databases or file systems where the information they need is stored.
- The Programs objects in the information catalog must include the correct invocation syntax for the operating systems on which your users will run the programs.

### Additional requirements for DataGuide for the Web users

There are specific considerations to be aware of when you set up the Web environment so that DataGuide for the Web users can start programs.

Ensure that the following requirements are met:

The data that users want to use with the application program must be accessible to the Web server. For example, the DataGuide sample data file must be located in a directory on the Web server.

The program that users want to start must be installed on the Web client.

For example, if users are accessing a Lotus 1-2-3 file, then Lotus 1-2-3 must be installed on the Web client.

If the application program is a Java applet, the application does not need to be installed; it can be accessed directly from the Web browser.

The DataGuide for the Web client should also have any necessary browser plug-ins. The DataGuide for the Web server must be able to locate any associated files used by the plug-in. For example, if the user wants to view an Adobe Acrobat file, then the user should have the browser plug-in for the Acrobat Reader installed on the DataGuide for the Web client. The DataGuide for the Web server must be able to locate the file that the user wants to view to download it to the client.

The required MIME types must be identified in the Web server configuration file for the application program that users will start. An `AddType` directive with the file extension of the program users want to start must be included in the configuration file. For example, if users want to start a Lotus 1-2-3 spreadsheet with a file type of WK4, then the `AddType` directive for Lotus Domino Go Webserver would be defined as:

```
AddType .WK4 application/x-lotus123 binary
```

If users are using a Web server other than Lotus Domino Go Webserver, the MIME types are defined differently. See your Web server documentation for more information.

## Authorizing DataGuide users to manage objects

For some versions of Netscape Navigator, helpers are set up to recognize file types and start the corresponding application program. Microsoft Internet Explorer does not use helpers. Instead, Internet Explorer uses the file type and program associations used by Windows Explorer; no set-up is required for Internet Explorer to recognize a file type.

The **URL to access data** property must be defined for the DataGuide object from which users want to start the program. The value for the property is a link to directly launch the program.

### ***To start a program from a DataGuide for the Web object:***

1. In the list pane, click on the object from which you want to start the program.

The object description page opens in the description pane.

2. Find the **URL to access data** property.
3. Click the property value.

The Web browser is launched using the Web address specified by the property value.

---

## Authorizing DataGuide users to manage objects

You can give specific DataGuide users in your organization authority to manage and update objects in an information catalog. Authorized DataGuide users can do tasks such as creating, updating and exporting objects.

### ***To authorize DataGuide users:***

- 1 From the menu bar of the DataGuide Catalog window, select **Catalog > Manage DataGuide users**.

The Manage DataGuide Users window opens.

- 2 In the **New user ID** field, type the user ID of the DataGuide user you want to authorize to manage objects.
- 3 Click on the **Add** push button. The new user ID is added to the **Users** list.
- 4 Click on the **OK** push button to complete the authorization and commit the changes to the information catalog.

### ***To remove a DataGuide user ID from the list of authorized user IDs:***

- 1 Select the user ID from the **Users** list.
- 2 Click on the **Remove** push button.
- 3 Click on the **OK** push button to complete the authorization and commit the changes to the information catalog.

***Setting and changing DataGuide administrators:*** You can use the Manage DataGuide Users window to change the user IDs of your DataGuide administrators.

## Setting and changing status values for comments

You can enter the user IDs of the primary and backup administrators in the **Primary user ID** and **Backup user ID** fields respectively.

Before changing the primary administrator, ensure that the new primary administrator has SYSADM authority if your information catalog is stored in a DB2 UDB for Windows NT, DB2 for AIX, DB2 PE, DB2 UDB EEE, DB2 for MVS, DB2 for OS/390, or DB2 for OS/2 database. If your information catalog is stored in a DB2 for OS/400 database, make sure the new primary administrator has ALLOBJ authority.

If you enter an incorrect user ID for both primary and backup administrators and cannot access your information catalog, you can use the DataGuide ALTERKA command to set the correct user ID. To change an administrator for an information catalog, enter the following command at an MS-DOS command prompt:

```
X:\VWLIB\BIN\ALTERKA
```

where *x* is the drive where DataGuide is installed.

When prompted, enter (separated by blanks) the information catalog name, user ID, and password:

```
DGV5SAMP longods secret
```

When prompted again, enter (separated by blanks) the administrator user ID you want to alter, the action (add, delete, or update) on the user ID, and the user ID type (primary or back-up):

```
valdezma D B
```

The example deletes the back-up administrator user ID, *valdezma*.

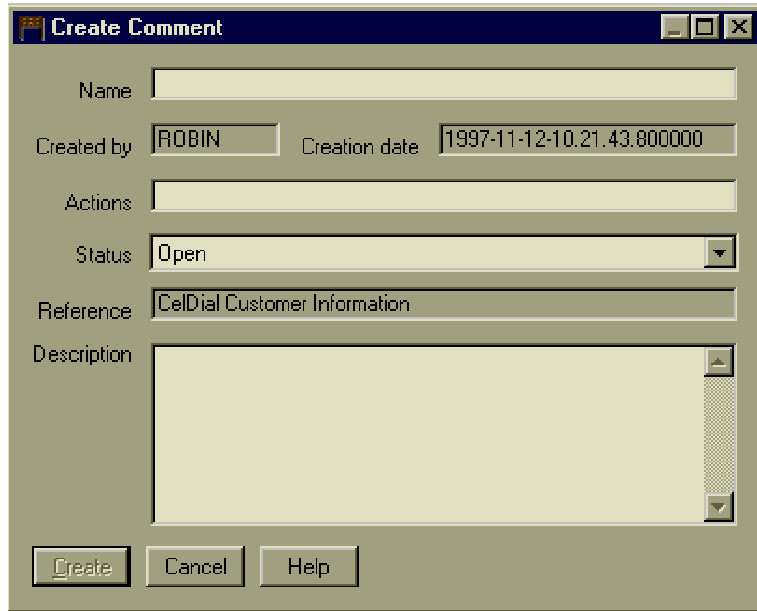
---

## Setting and changing status values for comments

You can set the list of available status choices for users to assign to comments in your information catalog. For example, status choices might be Open, Pending, Action required, and Closed.

The status choices you identify appear in the **Status** list in the Create Comment (as shown in Figure 1 on page 20), Copy Comment, and Update Comment windows.

## Setting and changing status values for comments



The screenshot shows a 'Create Comment' dialog box with the following fields and values:

- Name: [Empty]
- Created by: ROBIN
- Creation date: 1997-11-12-10.21.43.800000
- Actions: [Empty]
- Status: Open
- Reference: CelDial Customer Information
- Description: [Empty text area]

Buttons at the bottom: Create, Cancel, Help.

Figure 1. Create Comment window showing status choices

### To set or change the comment status choices:

- 1 From the menu bar of the DataGuide Catalog window, select **Catalog > Update comment status list** from the menu bar.  
The Update Comment Status List window opens.
- 2 Set the list of up to ten status choices by typing one status in each **Comment status field choice** field. The order of the list you enter is the order of the **Status** list in the various comment windows.
- 3 Click on the **Update** push button when you finish setting status values.  
To close the window without updating the status list, click on the **Cancel** push button.

## DataGuide information categories and object types

---

### Chapter 2. Organizing your information resources

By now, you have set up DataGuide by creating an information catalog and ensuring you and your users can open it. Next, you must complete some necessary preparatory tasks so that you can populate your information catalog with descriptive data about your organization's information resources.

Start by organizing the information you want to include. For example, you can plan to include descriptive data about your organization's personnel records, financial spreadsheets, building plans, and digital images from advertising campaigns. Each of these items is a different type of information resource.

When you've categorized the types of information you want to include in your information catalog, you identify the types of information in your information catalog.

---

### Understanding DataGuide information categories and object types

To organize your information resources in the information catalog, you create object types. An *object type* is a classification for objects that is used to reflect a type of business information, such as a table, report, or image. For example, you might create an object type called Image (Figure 2), which describes a set of objects that are digital bitmap images. For each object type, you define a set of *properties*, which describe the characteristics of the object type. For an object type called Image, you might define properties such as Resolution, Size, and Color.

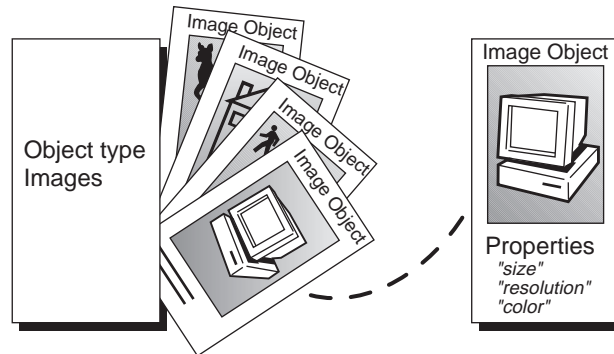


Figure 2. An information resource's characteristics become properties of a DataGuide object type

Every object type must belong to a DataGuide *category*. An object type's category affects how DataGuide handles it. Except for the Program and Attachment categories, you can create object types in any of the following DataGuide categories:

<b>Category</b>	<b>Definition</b>
<b>Grouping</b>	Object types that can contain other object types.

## DataGuide information categories and object types

<b>Elemental</b>	Non-Grouping object types that are the building blocks for other DataGuide object types.
<b>Contact</b>	Object types that identify a reference for more information about an object. More information might include the person who created the information that the object represents, or the department responsible for maintaining the information.
<b>Program</b>	A Programs object type that identifies and describes applications capable of processing the actual information represented by DataGuide objects types. The only object type belonging to the Program category is the Programs object type, which is defined when you create an information catalog.
<b>Dictionary</b>	Object types that define terminology that is specific to your business.
<b>Support</b>	Object types that provide additional information about your information catalog or enterprise.
<b>Attachment</b>	A Comments object type that identifies additional information attached to another DataGuide object. The only object type belonging to the Attachment category is the Comments object type, which is defined when you create an information catalog.

Table 9 summarizes the relationships among DataGuide's object type categories.

Category	Can contain/ contained by	Links with	Contacts associated	Comments attached	Programs launch from
Grouping	Contains other Grouping or Elemental objects	Other Grouping or Elemental objects	Yes	Yes	Yes
Elemental	Contained by any Grouping object	Other Grouping or Elemental objects	Yes	Yes	Yes
Contact	None	None	No	Yes	Yes
Program	None	None	No	Yes	No
Dictionary	None	None	No	Yes	Yes
Support	None	None	No	Yes	Yes
Attachment	None	None	No	No	Yes

You can establish object types for your information catalog in any of three ways:

- Use the object types that come with DataGuide in the sample information catalog (see Appendix B, "Predefined DataGuide object types" on page 100 for information about creating the sample information catalog and a description of the object types it includes).



## Creating your own object types

- Modify the object types that come with DataGuide to fit your organization's needs (see "Updating an object type" on page 32 for information about modifying an object type).
- Create your own object types.

---

### Creating your own object types

When you create your own object types, start by creating a prototype for each object type you need. Then create one or two sample objects (see Chapter 3, "Populating the catalog with information" on page 37 for information about objects). Check how the objects appear in a Description View, especially the order in which the properties are listed. Try entering different values for each property to be sure you have the right data types and sizes. You might want to consult with your database administrator and some of your users to ensure that the properties you specify meet your work group's needs.

If you aren't satisfied with your prototype, you can easily delete it and your sample objects and start over. After you create an object type, the only way to change or delete its properties is to delete the object type *and all objects of that type* and create a new object type.

Also consider how many object types you will need. DataGuide limits the number of object types you can create in an information catalog to 999 999, and the number of objects you can create for each type to 99 999 999. This limit includes all the object types you have ever created, even the ones you have deleted.

You can create an object type using the DataGuide windows or tag language.

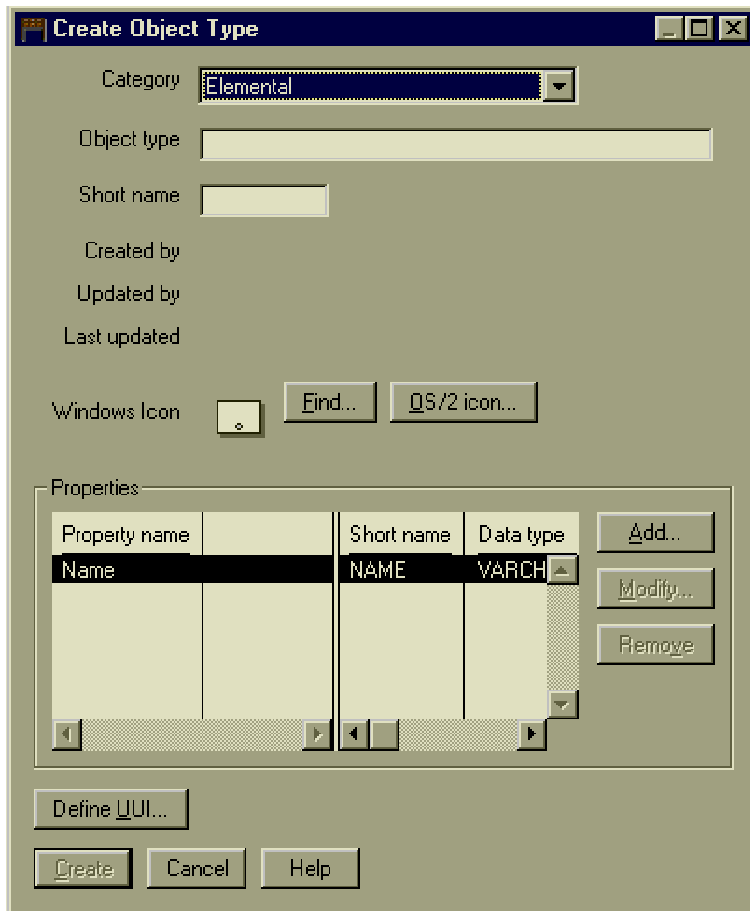
### Creating an object type using the DataGuide windows

Start from the DataGuide Catalog window.

- 1 Click mouse button 2 on the **Object types** icon.
- 2 Select the **Open as > Icon list** menu choice from the pop-up menu.
- 3 Click mouse button 2 on the **New Object Type** icon.
- 4 Select the **Open** menu choice from the pop-up menu.

The Create Object Type window opens.

## Creating your own object types



- 5 Click on the down arrow to display a list of DataGuide categories and select one.
- 6 Type a unique external name for the new object type in the **Object type name** field.  
The rules for object type names are:
  - 80 character maximum.
  - It must not contain null characters.
  - It must not be all blank characters.
- 7 Type a unique short name for the new object type in the **Short name** field.  
The rules for short names are:
  - 8 character (SBCS) maximum.
  - First character must be uppercase or lowercase English alphabetic, @, #, or \$.

## Creating your own object types

- Subsequent characters must be uppercase or lowercase English alphanumeric, @, #, \$, or \_.
- 8** Optional: Identify the object type's icons. The default Windows icon for the selected category is displayed in the **Windows icon** field. To identify a different icon to represent the object type:
- Click on the **Find** push button to locate a different Windows icon and display it in the window.
  - Click on the **OS/2 icon** push button to identify an OS/2 icon you want to use to represent the selected object type.
- 9** Define all the properties for the object type (see “Defining the object type's properties” for detailed information).
- Click on:**    **To:**
- Add**            Define additional properties
- Modify**        Change a property before you create the object type
- Remove**        Remove a property before you create the object type
- Select the property, then select **Remove**.
- 10** Click on the **Define UUI** push button to choose up to five properties that constitute the universal unique identifier (UUI) for the object type (see “Defining a universal unique identifier for the object type” on page 27 for detailed information).
- 11** Click on the **Create** push button to save your changes in the database.
- Your changes display in the Object Types window, but not in other windows until you close and re-open them.
- To close the window without creating an object type, click on the **Cancel** push button.

### Defining the object type's properties

Each object type can have up to 255 properties. The order in which you define the properties is the order in which the user will see them. You cannot change or rearrange the properties after you create the object type.

DataGuide defines five properties that are common to all DataGuide object types. These five properties are summarized in Table 10.

Table 10 (Page 1 of 2). DataGuide object type common properties

External name of property	Property short name	Definition
Object type identifier	OBJTYPID1	DataGuide generates this value, which uniquely identifies the object type of an object within the scope of the local DataGuide information catalog.

## Creating your own object types

Table 10 (Page 2 of 2). DataGuide object type common properties

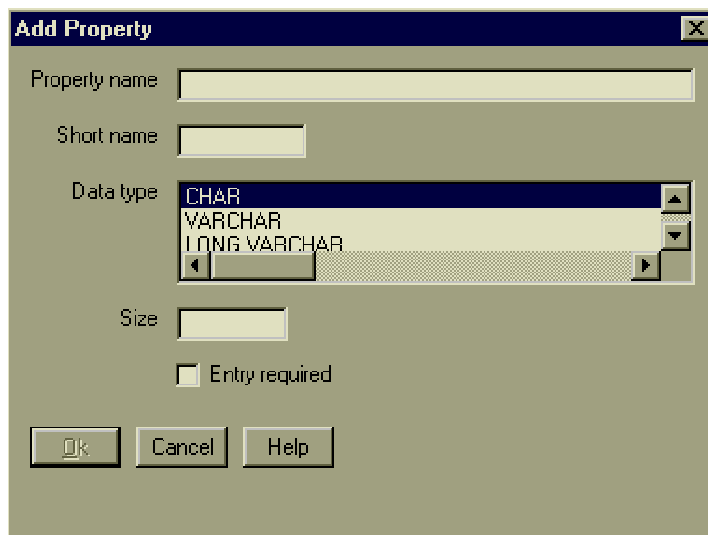
External name of property	Property short name	Definition
Instance identifier	INSTIDNT <sup>1</sup>	DataGuide generates this value, which uniquely identifies an object within the scope of the local DataGuide information catalog.
Name	NAME	You provide the name of an object. Choose names that users readily recognize and understand.
Last Changed Date and Time	UPDATIME <sup>1</sup>	DataGuide generates this value, indicating the date and time the object was last changed.
Last Changed By	UPDATEBY <sup>1</sup>	DataGuide generates this value, showing the user ID of the DataGuide session that last updated the Last Changed Date and Time property.

**Note:**

1. If you selected the **Hide system generated properties** check box in the DataGuide Settings notebook, you won't see this property in the object's description.

**To define additional properties:**

- 1 Select the **Add** button next to the **Properties** list. The Add Property window opens.



Use this window to define properties.

- 2 In the **Property name** field, type a name for the property.

## Creating your own object types

The rules for property names are:

- 80 character maximum.
- It must not contain null characters.
- It must not be all blank characters.

**3** In the **Short name** field, type a unique property short name.

The rules for short names are:

- 8 character (SBCS) maximum.
- First character must be uppercase or lowercase English alphabetic, @, #, or \$.
- Subsequent characters must be uppercase or lowercase English alphanumeric, @, #, \$, or \_.
- It must not be an SQL reserved word.
- It must be unique; if you type a name that already exists in this object type, DataGuide asks you for another name.

**4** Select a data type for the property from the **Data type** list:

<b>CHAR</b>	Up to 254 characters
<b>VARCHAR</b>	Up to 4000 characters
<b>LONG VARCHAR</b>	Up to 32700 characters
<b>TIMESTAMP</b>	Exactly 26 characters, in this format: yyyy-mm-dd-hh.mm.ss.nnnnnn:

You can have up to 14 LONG VARCHAR properties for an object type.

- 5** Type a size for the property in the **Size** field. The size must be within the range for the data type you selected.
- 6** If you want to require entry of this property whenever you create an object of this type, check **Entry required**.
- 7** Click on the **OK** push button to return to the Object Types window.

### Defining a universal unique identifier for the object type

All object types must have at least one property that is part of the *universal unique identifier*, or UUI. The UUI is a string of characters that enables DataGuide to tell one object from another. This requirement enables you to import the contents of one information catalog into another.

For example, in an information catalog for your manufacturing division, you can have an object named Product List that shows all products manufactured by the division. The sales division's information catalog might also have an object named Product List that shows all products sold by the sales division.

Without a way to uniquely identify these objects, you risk overwriting the descriptive data when you combine information catalogs.

## Creating your own object types

DataGuide prevents overwriting by having you define the UUI. You do not have to come up with unique names on your own or know what every object in another DataGuide information catalog is called.

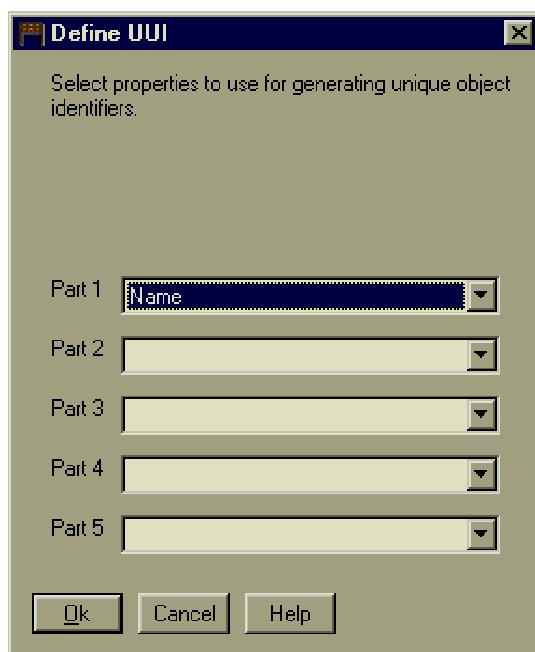
You choose up to five properties of an object type and designate them in whatever order you want. The values for each of these properties, in the order you give them, become the UUI for any object of that type.

When you import an object into your information catalog, DataGuide compares the values of the UUI properties to see if they match those of an existing object. If all the UUI properties have the same value in both objects, DataGuide treats the two as the same object. It updates the values in the existing object's non-UUI properties. If the UUI properties have different values, DataGuide adds the incoming object to the information catalog.

If you want to designate a property that you are sure is unique by itself, such as a purchase requisition number or international standard book number (ISBN), you do not need to designate all five properties. You can fill in the values for the UUI properties that you do not need with the not-applicable symbol or when you create an object type, you can give it fewer UUI properties. (The not-applicable symbol is a hyphen unless you identified a different symbol when you created the information catalog.)

For performance reasons, be careful to select UUI properties so that the total number of characters in their combined values is fewer than 254.

**To define the UUI, start from the Define UUI window:**



## Creating your own object types

- 1 Select up to five properties as parts of the UUI. The part number determines the position that the property has in the UUI sequence.

DataGuide limits your choice of properties:

- You can choose only required properties.
- You can't choose properties whose data type is LONG VARCHAR.
- You can't choose properties whose data type is VARCHAR and whose length exceeds the 254 byte maximum.
- You can't use the same property for more than one part.
- You can't skip parts (for example, you can't choose only parts 1, 3, and 5).

For each part, choose a property:

- a Select the down arrow next to the **Part** field to see a list of available properties.
  - b Select a property.
- 2 When you finish filling in the parts, click **OK** to accept the UUI definition and return to the Object Types window.

## Creating an object type using DataGuide tag language

- 1 Enter the following lines in your tag language file:

```
:ACTION.OBJTYPE(ADD)
:OBJECT.TYPE(short_name_of_object_type)
    PHYNAME (name_of_table)
    CATEGORY(category_of_object_type)
    EXTNAME(external_name_of_object_type)
    ICOFILE(name_of_OS/2_icon_file)
    ICWFILE(name_of_Windows_icon_file)
```

After each keyword, type an appropriate value within the parentheses:

Keyword	Value
TYPE	The short name of the object type. The rules for short names are: <ul style="list-style-type: none"><li>- 8 character (SBCS) maximum.</li><li>- First character must be uppercase or lowercase English alphabetic, @, #, or \$.</li><li>- Subsequent characters must be uppercase or lowercase English alphanumeric, @, #, \$, or _.</li><li>- It must be unique to the information catalog.</li></ul>
PHYNAME	The name of a DB2 table where DataGuide stores objects of this type.  If your DB2 tables follow naming conventions, you can use PHYNAME to give the underlying tables in your information catalog a different name from the object type name.

## Creating your own object types

If you don't specify this property, DataGuide uses the short name you gave as TYPE.

You can add PHYNAME only if you use a tag language file to create the object type. You can't add it through the EUI.

CATEGORY	The category: GROUPING, ELEMENTAL, CONTACT, DICTIONARY, or SUPPORT.
EXTNAME	The external name of the object type. The rules for external names are: <ul style="list-style-type: none"> <li>- 80 character maximum.</li> <li>- It must not contain null characters.</li> <li>- It must not be all blank characters.</li> </ul>
ICOFIELD	The name of the OS/2 icon file, including its extension. You give the drive and path information where the icon file exists as part of the IMPORT command when you import your tag language file.
ICWFIELD	The name of the Windows icon file, including its extension. You give the drive and path information where the icon file exists as part of the IMPORT command when you import your tag language file.

### 2 Type lines for each property you want to give your object type:

```
:PROPERTY.SHRTNAME(short_name) DT(data_type) DL(size)
    UUISEQ(position_in_UUI) NULLS(y_or_n) EXTNAME(property_name)
```

Keyword	Value								
SHRTNAME	The property short name. The rules for property short names are: <ul style="list-style-type: none"> <li>- 8 character (SBCS) maximum.</li> <li>- First character must be uppercase or lowercase English alphabetic, @, #, or \$.</li> <li>- Subsequent characters must be uppercase or lowercase English alphanumeric, @, #, \$, or _.</li> <li>- It must not be an SQL reserved word.</li> <li>- It must be unique; if you type a name that already exists in this object type, DataGuide asks you for another name.</li> </ul>								
DT	The data type: <b>C</b> , <b>V</b> , <b>L</b> , or <b>T</b> . <table> <tr> <td><b>C (CHAR)</b></td> <td>Up to 254 characters</td> </tr> <tr> <td><b>V (VARCHAR)</b></td> <td>Up to 4000 characters</td> </tr> <tr> <td><b>L (LONG VARCHAR)</b></td> <td>Up to 32700 characters</td> </tr> <tr> <td><b>T (TIMESTAMP)</b></td> <td>26 characters, in this format: yyyy-mm-dd-hh.mm.ss.nnnnnn</td> </tr> </table>	<b>C (CHAR)</b>	Up to 254 characters	<b>V (VARCHAR)</b>	Up to 4000 characters	<b>L (LONG VARCHAR)</b>	Up to 32700 characters	<b>T (TIMESTAMP)</b>	26 characters, in this format: yyyy-mm-dd-hh.mm.ss.nnnnnn
<b>C (CHAR)</b>	Up to 254 characters								
<b>V (VARCHAR)</b>	Up to 4000 characters								
<b>L (LONG VARCHAR)</b>	Up to 32700 characters								
<b>T (TIMESTAMP)</b>	26 characters, in this format: yyyy-mm-dd-hh.mm.ss.nnnnnn								
DL	The size for the property.								



## Creating your own object types

UUISEQ	The position this property has in the UUI: <b>1, 2, 3, 4,</b> or <b>5</b> . Include this keyword only if you want the property to be part of the UUI.
NULLS	Whether entry is required: <b>N</b> Entry is required <b>Y</b> Entry is <i>not</i> required
EXTNAME	The property name. The rules for property names are: <ul style="list-style-type: none"><li>- 80 character maximum.</li><li>- It must not contain null characters.</li><li>- It must not be all blank characters.</li></ul>

If you want to make the NAME property part of the UUI for this object type, the only keywords you can have for the property are SHRTNAME and UUISEQ. DataGuide defines values for other keywords, so you don't specify them or their values here.

After adding all properties for your object type, your tag language file looks something like Figure 3 on page 32. Figure 3 on page 32 shows an abbreviated version of the "Relational tables and views" object type, which is one of the pre-defined object types provided with DataGuide. The complete object type definition is available in the X:\VWSWIN\DGWIN\TYPES directory.

## Updating an object type

---

```
:COMMENT.-----
:COMMENT.Generating the object definitions for "Relational tables and views"
:COMMENT.-----
:ACTION.OBJTYPE (MERGE)
:OBJECT.TYPE (TABLES) CATEGORY(GROUPING) PHYNAME(TABLES)
      EXTNAME(Relational tables and views) ICWFILE(flgnytatab.ico)
:PROPERTY. SHRTNAME(NAME) UUISEQ( )
:PROPERTY. SHRTNAME(SHRTDESC)  DT(V)  DL(25 )  UUISEQ( )  NULLS(Y)
      EXTNAME(Short description)
:PROPERTY. SHRTNAME(LONGDESC)  DT(L)  DL(327 )  UUISEQ( )  NULLS(Y)
      EXTNAME(Long description)
:PROPERTY. SHRTNAME(ACTIONS)   DT(V)  DL(254)  UUISEQ( )  NULLS(Y)
      EXTNAME(Actions)
:PROPERTY. SHRTNAME(REMARKS)   DT(V)  DL(254)  UUISEQ( )  NULLS(Y)
      EXTNAME(Catalog remarks)
:PROPERTY. SHRTNAME(RESPNBLS)  DT(V)  DL(8 )  UUISEQ( )  NULLS(Y)
      EXTNAME(For further information...)
:PROPERTY. SHRTNAME(SERVER)    DT(V)  DL(8 )  UUISEQ( )  NULLS(Y)
      EXTNAME(Database host server name)
:PROPERTY. SHRTNAME(DBALIAS)   DT(C)  DL(8)    UUISEQ( )  NULLS(Y)
      EXTNAME(Local database alias)
:PROPERTY. SHRTNAME(DBNAME)    DT(V)  DL(8 )  UUISEQ(1)  NULLS(N)
      EXTNAME(Database or subsystem name)
:PROPERTY. SHRTNAME(OWNER)     DT(V)  DL(8 )  UUISEQ(2)  NULLS(N)
      EXTNAME(Table owner)
:PROPERTY. SHRTNAME(TABLE)     DT(V)  DL(8 )  UUISEQ(3)  NULLS(N)
      EXTNAME(Table name)
:PROPERTY. SHRTNAME(SRCOWNER)  DT(C)  DL(3 )  UUISEQ( )  NULLS(Y)
      EXTNAME(Base table owner name)
:PROPERTY. SHRTNAME(SRCTBNAM)  DT(C)  DL(128)  UUISEQ( )  NULLS(Y)
      EXTNAME(Base table name)
:PROPERTY. SHRTNAME(FRESHDAT)  DT(C)  DL(26)  UUISEQ( )  NULLS(Y)
      EXTNAME(Table data last refreshed)
:PROPERTY. SHRTNAME(RUNMODE)   DT(C)  DL(3 )  UUISEQ( )  NULLS(Y)
      EXTNAME(Transformation program run mode)
:PROPERTY. SHRTNAME(LASTRUN)   DT(C)  DL(26)  UUISEQ( )  NULLS(Y)
      EXTNAME(Transformation program last run)
:PROPERTY. SHRTNAME(RUNFREQ)   DT(V)  DL(8 )  UUISEQ( )  NULLS(Y)
      EXTNAME(Transformation program run frequency)
:PROPERTY. SHRTNAME(SOURCE)    DT(V)  DL(32)  UUISEQ( )  NULLS(Y)
      EXTNAME(Transformation program type)
:PROPERTY. SHRTNAME(CRTTIME)   DT(C)  DL(26)  UUISEQ( )  NULLS(Y)
      EXTNAME(Timestamp source definition created)
:PROPERTY. SHRTNAME(SRCDATCF)  DT(C)  DL(26)  UUISEQ( )  NULLS(Y)
      EXTNAME(Timestamp source definition last changed)
```

---

Figure 3. Sample tag language file for an object type

---

## Updating an object type

You can make *only* the following changes to an existing object type:

- Change the external name
- Change the representative icon
- Add properties

You can update an object type using the DataGuide windows or tag language.

## Updating an object type

### Updating an object type using the DataGuide windows

Start from the DataGuide Catalog window.

- 1 Click mouse button 2 on the **Object types** icon.
- 2 Select the **Open as > Icon list** menu choice from the pop-up menu.
- 3 Click mouse button 2 on the icon of the object type you want to change.
- 4 Select the **Open** menu choice from the pop-up menu.

The Update Object Type window opens.

- 5 To change the external name, type a new name in the **Object type name** field. The rules for object type names are:

- 80 character maximum.
- It must not contain null characters.
- It must not be all blank characters.

- 6 The default Windows icon for the selected category is displayed in the **Windows icon** field. To identify a specific icon to represent the object type:

- Click on the **Find** push button to locate a different Windows icon and display it in the window.
- Click on the **OS/2 icon** push button to identify an OS/2 icon you want to use to represent the selected object type.

- 7 To add a property to the object type (unless the type is Comments, which cannot be extended), click on the **Add** push button.

The Add Property window opens.

- a Type a name for the property in the **Property name** field. The rules for property names are:

- 80 character maximum.
- It must not contain null characters.
- It must not be all blank characters.

- b Type a unique property short name in the **Short name** field. The rules for short names are:

- 8 character (SBCS) maximum.
- First character must be uppercase or lowercase English alphabetic, @ (at sign), # (number sign), or \$ (dollar sign).
- Subsequent characters must be uppercase or lowercase English alphanumeric, @ (at sign), # (number sign), \$ (dollar sign), or \_ (underscore).
- It must not be an SQL reserved word.
- It must be unique; if you type a name that already exists in this object type, DataGuide asks you for another name.

- c Select a data type from the list.

- d Type a size in the **Size** field.

## Updating an object type

- e Click on the **OK** push button to add the property.

To close the window without adding a property, click on the **Cancel** push button.

This step is not available for OS/400 information catalogs.

- 8 To change a property you added during the current update action, select it in the **Properties** box and then click on the **Modify** or **Remove** push button as appropriate.

This step is not available for OS/400 information catalogs.

- 9 Click on the **Update** push button to save your changes in the database.

Your changes display in the Object Types window, but not in other windows until you close and reopen them.

To close the window without updating the object type, click on the **Cancel** push button.

## Updating an object type using DataGuide tag language

- 1 Enter the following lines in your tag language file:

```
:ACTION.OBJTYPE(UPDATE)
:OBJECT.TYPE(short_name_of_object_type)
```

- 2 To change the external name, add the following line:

```
EXTNAME(new_external_name_of_object_type)
```

- 3 To change the object type's icon, add the following line:

```
ICOFIELD(new_OS/2_icon_filename)
ICWFIELD(new_Windows_icon_filename)
```

After each keyword, type an appropriate value within the parentheses:

<b>Keyword</b>	<b>Value</b>
TYPE	The short name of the object type you are updating.
EXTNAME	The new external name of the object type. The rules for external names are: <ul style="list-style-type: none"><li>- 80 character maximum.</li><li>- It must not contain null characters.</li><li>- It must not be all blank characters.</li></ul>
ICOFIELD	The name of the new OS/2 icon file, including its extension. You give the drive and path information where the icon file exists as part of the IMPORT command when you import your tag language file.

## Updating an object type

**ICWFILE** The name of the new Windows icon file, including its extension. You give the drive and path information where the icon file exists as part of the **IMPORT** command when you import your tag language file.

- 4 To add an optional property, enter the following lines in your tag language file:

```
:ACTION.OBJTYPE (APPEND)
:OBJECT.TYPE (short_name_of_object_type)
:PROPERTY.SHRTNAME (short_name_of_new_property) DT (data_type) DL (size)
      UISEQ ( ) NULLS (y) EXTNAME (external_name_of_new_property)
```

After each keyword, type an appropriate value within the parentheses.

Any property you add to an object type after you create it must be an optional property, so the value for **UISEQ** must be 0 and **NULLS** must be Y.

<b>Keyword</b>	<b>Value</b>
<b>TYPE</b>	The short name of the object type you are updating.
<b>SHRTNAME</b>	The property short name. The rules for property short names are: <ul style="list-style-type: none"><li>- 8 character (SBCS) maximum.</li><li>- First character must be uppercase or lowercase English alphabetic, @ (at sign), # (number sign), or \$ (dollar sign).</li><li>- Subsequent characters must be uppercase or lowercase English alphanumeric, @ (at sign), # (number sign), \$ (dollar sign), or _ (underscore).</li><li>- It must not be an SQL reserved word.</li><li>- It must be unique; if you type a name that already exists in this object type, DataGuide asks you for another name.</li></ul>
<b>DT</b>	The data type: <b>C</b> , <b>V</b> , <b>L</b> , or <b>T</b> . <b>C (CHAR)</b> Up to 254 characters <b>V (VARCHAR)</b> Up to 4000 characters <b>L (LONG VARCHAR)</b> Up to 32700 characters <b>T (TIMESTAMP)</b> Exactly 26 characters, in this format: yyyy-mm-dd-hh.mm.ss.nnnnnn
<b>DL</b>	The size for the property.
<b>EXTNAME</b>	The external name of the property. The rules for property names are: <ul style="list-style-type: none"><li>- 80 character maximum.</li><li>- It must not contain null characters.</li><li>- It must not be all blank characters.</li></ul>

## Deleting an object type

---

### Deleting an object type

When you delete an object type, all objects of that type are also deleted (unless any objects are Grouping objects that contain objects of a different object type). You can delete an object type from your information catalog using the DataGuide windows or tag language.

### Deleting an object type using the DataGuide windows

- 1 (Optional) Search for objects of the object type you want to delete to ensure that you do not want to retain any of them.
- 2 Click mouse button 2 on the **Object types** icon in the DataGuide Catalog window.
- 3 Select the **Open as > Icon list** menu choice from the pop-up menu.
- 4 Click mouse button 2 on the icon of the object type you want to delete.
- 5 Select the **Delete** menu choice from the pop-up menu.  
The Delete window opens.
- 6 Click on the **Delete** push button to delete the object type.
- 7 Click on the **Yes** push button to confirm deletion.

When you delete an object type, DataGuide closes all windows that are directly related to that object type.

### Deleting an object type using DataGuide tag language

Enter the following lines in your tag language file:

```
:ACTION.OBJTYPE(DELETE_EXT)  
:OBJECT.TYPE(short_name_of_object_type)
```

After each keyword, type an appropriate value within the parentheses:

<b>Keyword</b>	<b>Value</b>
TYPE	The short name of the object type you are deleting.

---

### Chapter 3. Populating the catalog with information

After you define the object types you need, you populate, or fill, the DataGuide information catalog with objects. An *object* is an item that represents a unit or distinct grouping of information. Every object is associated with an object type. For example, if you have an object type called Image, you might have an object of that type called **My\_DBA**, which describes a bitmap photograph of the database administrator that you work with.

This chapter describes how to:

- Create objects
- Copy existing objects
- Update existing objects
- Delete objects

---

#### Creating an object

You create objects of various types to represent the actual information available in your organization. You can create objects using the DataGuide windows or tag language.

#### Creating an object using the DataGuide windows

Start from the DataGuide Catalog window.

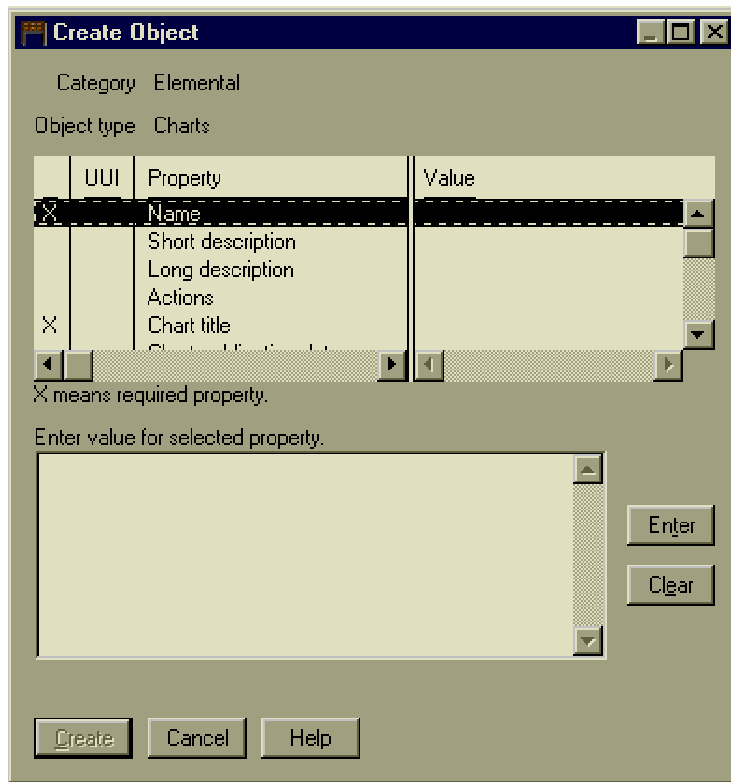
- 1** Click mouse button 2 on the **Object types** icon in the DataGuide Catalog window.
- 2** Select the **Open as > Icon list** menu choice from the pop-up menu.
- 3** Click mouse button 2 on the icon of the object type for which you want to create an object.

You *cannot* create an object using either the **Programs** or **Comments** icons. You create a Programs object when you associate a program with an existing object type (see “Associating a program with object types” on page 55). You create a Comments object from an existing object of another object type (see “Associating comments and objects” on page 51).

- 4** Select the **Create object** menu choice from the pop-up menu.

The Create Object window opens.

## Creating an object



- 5 Select a property from the **Properties/Values** list box.
- 6 Type a value for the property in the **Enter value for selected property** field.
- 7 Click on the **Enter** push button to move the value to the **Value** column in the **Properties/Values** list box.  
If you want to erase what you entered in the **Enter value for selected property** field, click on the **Clear** push button.
- 8 Click on the **Create** push button when you finish entering values.  
To close the window without creating an object, click on the **Cancel** push button.

## Creating an object using the DataGuide tag language

You can create many objects at the same time using the DataGuide tag language. You can include the tag language for creating an object in the same tag language file in which you defined the object type, after the object type definition. The properties can go in any order, and you can omit properties for which you don't have a value.

Enter the following lines in your tag language file, using as many `short_name(value_for_property)` lines as necessary to identify all the object type properties.



## Copying an object

```
:ACTION.OBJINST(ADD)
:OBJECT.TYPE(short_name_of_object_type)
:INSTANCE.short_name(value_for_property)
        short_name(value_for_property)
        short_name(value_for_property)
```

After each keyword, type an appropriate value within the parentheses:

Keyword	Value
TYPE	The short name of the object type for which you are creating an object.
short_name	The short name of the object type property.

For each object, type the short name of each of the object type properties, followed by a value for the property in parentheses. Figure 4 shows an example of tag language to create an object. The example is based on the object type created with the definition in Figure 3 on page 32.

---

```
:COMMENT.-----
:COMMENT. Creating objects of object type
:COMMENT. "Relational tables and views"
:COMMENT.-----
:ACTION.OBJINST(ADD)
:OBJECT.TYPE(TABLES)
:INSTANCE.NAME(Customer)
        SHRTDESC(Customer information table)
        LONGDESC(Customer number, name, CelDial rep, customer contact information.)
        ACTIONS(Click on 'Start Program...' to invoke Visualizer TableViewer.)
        REMARKS(DB2 table) DBNAME(DGWDATA) OWNER(USERID) TABLE(CUSTOMER)
        URL(http://$$$@@@/dataguide/db2www/dg_tableviewer.mac/Table_Login?DATABASE=
DGWDATA&TABLE=CUSTOMER&OWNER=USERID)
        SOURCE(DB2 SYSTEM CATALOGS)
```

---

Figure 4. Creating an object with tag language

---

## Copying an object

You can create a new object that has the values of an existing object. (For information about copying a comment, see “Copying a comment” on page 52.) Start from one of the following windows:

- Search Results
- Collection
- Found In
- Contacts
- Subjects
- Tree View

## Updating an object

Linked With

- 1 Click mouse button 2 on the object you want to copy.
- 2 Select the **Copy** menu choice from the pop-up menu.  
The Copy Object window opens.
- 3 Select a property from the **Properties/Values** list box.
- 4 Edit the value for the property in the **Enter value for selected property** field.  
*You must change at least one UUI value for your new object to be unique.*  
If you want to erase the existing value in the **Enter value for selected property** field, click on the **Clear** push button.
- 5 Click on the **Enter** push button to move the changed value to the **Value** column in the **Properties/Values** list box.
- 6 Click on the **Copy** push button when you finish changing values.  
To close the window without copying an object, click on the **Cancel** push button.

---

## Updating an object

You can change values for an existing object using the DataGuide windows or tag language. (For information about updating a comment, see "Updating a comment" on page 52.)

## Updating an object using the DataGuide windows

Start from one of the following windows:

Search Results  
Collection  
Found In  
Contacts  
Subjects  
Tree View  
Linked With

- 1 Click mouse button 2 on the object you want to update.
- 2 Select the **Update** menu choice from the pop-up menu.  
The Update Object window opens.
- 3 Select a property from the **Properties/Values** list box.
- 4 Edit the value for the property in the **Enter value for selected property** field.  
If you want to erase the existing value in the **Enter value for selected property** field, click on the **Clear** push button.
- 5 Click on the **Enter** push button to move the changed value to the **Value** column in the **Properties/Values** list box.

## Updating an object

- 6 Click on the **Update** push button when you finish changing values.  
To close the window without updating an object, click on the **Cancel** push button.

### Updating an object using the DataGuide tag language

- 1 Enter the following lines in your tag language file:  

```
:ACTION.OBJINST(UPDATE)
:OBJECT.TYPE(short_name_of_object_type)
```
- 2 Enter the following lines, filling in the UUI properties and property values of the object you want to change:

```
:INSTANCE.SOURCEKEY(UUI_short_name(value_for_property)
    UUI_short_name(value_for_property)
    UUI_short_name(value_for_property))
```

After each keyword, type an appropriate value within the parentheses:

Keyword	Value
---------	-------

UUI_short_name	The short name of a UUI property of the object type.
----------------	--

The properties and values specified after the SOURCEKEY keyword are the UUI. When you created the object type, you defined up to five properties in a certain order to make up the UUI. When you type in those properties and values, DataGuide checks the values in the order defined in the object type to locate a particular object.

Completely enclose in parentheses all the properties and values after the SOURCEKEY keyword.

- 3 Type the short name of each of the object's properties that you want to update, followed by the new value in parentheses for the property.

```
short_name(new_value_for_property)
```

You do not have to include all the object's properties. Any properties you leave out won't be updated.

Figure 5 shows an example of tag language to update an object. The example is based on the object with the same UUI properties and values created in Figure 4 on page 39.

---

```
:ACTION.OBJINST(UPDATE)
:OBJECT.TYPE(TABLES)
:INSTANCE.SOURCEKEY(DBNAME(DGWDATA) OWNER(USERID)
    TABLE(CUSTOMER))
    SHRTDESC(Mobile phone customer information table)
```

---

Figure 5. Updating an object with tag language

In this example, the value in SHRTDESC is updated.

## Deleting an object

---

### Deleting an object

You can delete an object from your information catalog using the DataGuide windows or tag language. (For information about deleting a comment, see “Deleting a comment” on page 53.)

### Deleting an object using the DataGuide windows

Start from one of the following windows:

- Search Results
- Collection
- Found In
- Contacts
- Subjects
- Tree View
- Linked With

- 1 Click mouse button 2 on the object you want to delete.
- 2 Select the **Delete** menu choice from the pop-up menu.
  - If the object you are deleting is a Grouping object, the Delete Tree window opens.
  - If the object you are deleting is not a Grouping object, the Delete window opens.
- 3 (Optional) Deselect any objects in the **Object** list box that you do not want to delete.
- 4 If you are deleting a Grouping object, you must decide what you want DataGuide to do with the objects contained in the Grouping object:
  - Delete all the contained objects by selecting the **Delete all underlying objects** check box.
  - Keep all the contained objects, but delete their relationships to the Grouping object by deselect the **Delete all underlying objects** check box.
- 5 Click on the **Delete** push button to delete the object.

The object is deleted from the information catalog.

### Deleting an object using the DataGuide tag language

- 1 To delete a Grouping object and all objects it contains, enter the following line in your tag language file:  

```
:ACTION.OBJINST(DELETE_TREE_ALL)
```

To delete a Grouping object and all relationships in which it participates, including the underlying tree structure, enter the following line in your tag language file:

```
:ACTION.OBJINST(DELETE_TREE_REL)
```

To delete a non-Grouping object, enter the following line in your tag language file:

## Deleting an object

```
:ACTION.OBJINST(DELETE)
```

- 2 Enter the following line, filling in the object type of the object you want to delete:

```
:OBJECT.TYPE(short_name_of_object_type)
```

- 3 Enter the following lines, filling in the UUI properties and property values of the object you want to delete:

```
:INSTANCE.SOURCEKEY(UUI_short_name(value_for_property)  
    UUI_short_name(value_for_property)  
    UUI_short_name(value_for_property))
```

After each keyword, type an appropriate value within the parentheses:

Keyword	Value
TYPE	The short name of the object type for which you are deleting an object.
UUI_short_name	The short name of a UUI property of the object type for which you are deleting an object.

The properties and values specified after the SOURCEKEY keyword are the UUI. When you created the object type, you defined certain properties in a certain order to make up the UUI. When you type in those properties and values, DataGuide checks the values in the order defined in the object type to locate a particular object.

Completely enclose in parentheses all the properties and values after the SOURCEKEY keyword.

Figure 6 shows an example of tag language to delete a Grouping object and all objects it contains. The example is based on the object created in Figure 4 on page 39.

---

```
:ACTION.OBJINST(DELETE_TREE_ALL)  
:OBJECT.TYPE(TABLES)  
:INSTANCE.SOURCEKEY(DBNAME(DGWDATA) OWNER(USERID)  
    TABLE(CUSTOMER))
```

---

Figure 6. Deleting an object with tag language

In this example, the table object identified by DBNAME(DGWDATA) OWNER(USERID) TABLE(CUSTOMER) is deleted.

## Grouping objects by subject

---

### Chapter 4. Making the information catalog convenient for users

You can make your DataGuide information catalog more convenient for your users. With DataGuide you can:

- Group objects by subject area for easy browsing
- Link related objects together
- Add contact names to objects
- Associate comments and objects
- Set up object types to start programs
- Set up glossaries of standard terminology for users
- Provide support and helpful information for users

Grouping objects, linking them, adding contact names to them, and associating comments with them are all ways of establishing relationships between objects. When you begin creating relationships, try to use a top-down approach, both to enhance performance and avoid errors. For example, if you know that the object CelDial Marketing Information contains the object Advertising Information, which in turn contains Advertisements on the WWW, then place Advertising Information into CelDial Marketing Information before placing Advertisements on the WWW into Advertising Information.

Especially try to avoid relating an object with one that already has relationships several levels deep.

---

## Grouping objects by subject

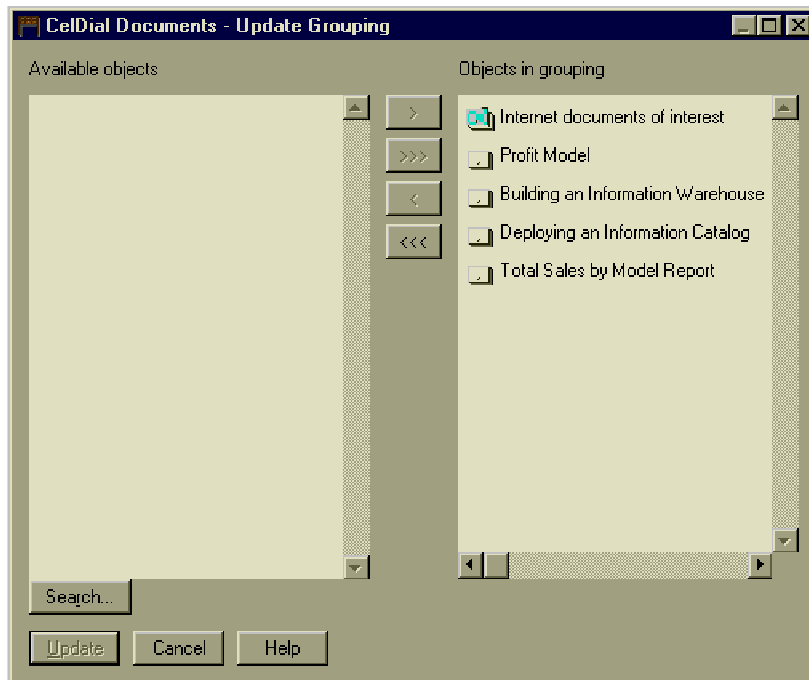
You can organize DataGuide objects by grouping them together. By nesting the groupings, you can organize your information catalog hierarchically. DataGuide shows the highest-level groupings in the Subjects window, represented by the **Subjects** icon in the DataGuide Catalog window.

You can group objects using the DataGuide windows or tag language.

### Grouping objects by subject using the DataGuide windows

- 1 Either create a Grouping category object (see “Creating an object” on page 37 for details) or locate an existing Grouping object in one of the following windows:
  - Search Results
  - Collection
  - Found In
  - Subjects
  - Linked With
  - Tree View
- 2 Click mouse button 2 on the object.
- 3 Select **Update grouping** from the pop-up menu.  
The Update Grouping window opens.

## Grouping objects by subject



#### 4 To add objects to the grouping:

- a Click on the **Search** push button.

The Define Search - Grouping window opens. Use this window to search for objects you want to include. Objects that fit your search criteria are returned to the Update Grouping window in the **Available objects** list.

- b Select one or more objects in the **Available objects** list.
- c Click on the **>** push button to move selected objects to the **Objects in grouping** list.

#### 5 To remove objects from the grouping:

- a Select one or more objects in the **Objects in grouping** list.
- b Click on the **<** push button to move selected objects out of the **Objects in grouping** list.

#### 6 Click on the **Update** push button.

Changes appear as soon as you make them from a **Tree View**, but you must close and reopen any other window for your updates to appear.

To close the window without changing the grouping, click on the **Cancel** push button.

## Grouping objects by subject

### Grouping objects by subject using DataGuide tag language

To create groupings of DataGuide objects with tag language, specify a Contains relationship between an object categorized as Grouping and an object categorized as Grouping or Elemental.

- 1 To add an object to a grouping, enter the following line in your tag language file:  
`:ACTION.RELATION(ADD)`
- 2 To delete an object from a grouping, enter the following line in your tag language file:  
`:ACTION.RELATION(DELETE)`
- 3 Specify the Contains relationship by typing the following lines, filling in the type of the Grouping object for SOURCETYPE and the object type you want to include in the group for TARGETTYPE:  
`:RELTYPE.TYPE(CONTAIN) SOURCETYPE(short_name_of_object_type)  
TARGETTYPE(short_name_of_object_type)`
- 4 Type the following lines, filling in the UUI properties and property values of the Grouping object:  
`:INSTANCE.SOURCEKEY(UUI_short_name(value_for_property)  
UUI_short_name(value_for_property)  
UUI_short_name(value_for_property))`
- 5 Type the following lines, filling in the UUI properties and property values of the object you want to include in the grouping:  
`TARGETKEY(UUI_short_name(value_for_property)  
UUI_short_name(value_for_property)  
UUI_short_name(value_for_property))`

After each keyword, type an appropriate value within the parentheses:

<b>Keyword</b>	<b>Value</b>
SOURCETYPE	The short name of the source object type.
TARGETTYPE	The short name of the target object type.
UUI_short_name	The short name of a UUI property of the object type.

Completely enclose in parentheses all the properties and values after the SOURCEKEY and TARGETKEY keywords.

Figure 7 on page 47 shows an example of tag language to add an object to a Grouping object. The example assumes that you already created the source and target objects.



## Creating a linked relationship between objects

---

```
:ACTION.RELATION(ADD)
:RELTYPE.TYPE(CONTAIN) SOURCETYPE(TABLES) TARGETTYPE(COLUMN)
:INSTANCE.SOURCEKEY(DBNAME(DGWDATA) OWNER(USERID)
    TABLE(AR_HISTORY))
    TARGETKEY(DBNAME(DGWDATA) OWNER(USERID) TABLE(AR_HISTORY)
    COLUMN(BAL3 ))
```

---

Figure 7. Adding an object to a grouping with tag language

In this example, the object identified by `DBNAME(DGWDATA) OWNER(USERID) TABLE(AR_HISTORY) COLUMN(BAL3 )` is placed in the Grouping object identified by `DBNAME(DGWDATA) OWNER(USERID) TABLE(AR_HISTORY)`.

---

## Creating a linked relationship between objects

To show users that data represented by one object is related to data represented by another object, you create a linked relationship by associating objects. You can link objects using the DataGuide windows or tag language.

### Linking objects using the DataGuide windows

To create a linked relationship, start from one of the following windows:

- Search Results
- Collection
- Found In
- Subjects
- Tree View
- Linked With

- 1 Click mouse button 2 on the object you want to associate with other objects in a linked relationship. The object's pop-up menu opens.
- 2 Select **Update links**.  
The Update Links window opens.
- 3 Link other objects to the selected object:
  - a Click on the **Search** push button.  
The Define Search - Links window opens. Use this window to search for objects you want to include in the linked relationship with the selected object. Objects that fit your search criteria are returned to the Update Links window in the **Available objects** list.
  - b Select one or more objects in the **Available objects** list.
  - c Click on the **>** push button to add selected objects to the **Linked objects** list.
- 4 To remove objects from the linked relationship:
  - a Select one or more objects in the **Linked objects** list.

## Creating a linked relationship between objects

- b** Click on the < push button to remove selected objects from the **Linked objects** list.
- 5** When you finish adding and removing objects, click on the **Update** push button. All linked relationships are updated.  
  
To close the window without changing any objects, click on the **Cancel** push button.

## Linking objects using DataGuide tag language

To link associated DataGuide objects with tag language, specify a Link relationship between two objects categorized as Grouping or Elemental.

- 1** To create a link between two objects, enter the following line in your tag language file:  

```
:ACTION.RELATION(ADD)
```
- 2** To remove a link between two objects, enter the following line in your tag language file:  

```
:ACTION.RELATION(DELETE)
```
- 3** Specify the Link relationship by typing the following lines, filling in the type of the two objects you are linking for SOURCETYPE and TARGETTYPE:  

```
:RELTYPE.TYPE(LINK) SOURCETYPE(short_name_of_object_type)
    TARGETTYPE(short_name_of_object_type)
```
- 4** Type the following lines, filling in the UUI properties and property values of one of the objects you are linking:  

```
:INSTANCE.SOURCEKEY(UUI_short_name(value_for_property)
    UUI_short_name(value_for_property)
    UUI_short_name(value_for_property))
```
- 5** Type the following lines, filling in the UUI properties and property values of the other object you are linking:  

```
TARGETKEY(UUI_short_name(value_for_property)
    UUI_short_name(value_for_property)
    UUI_short_name(value_for_property))
```

After each keyword, type an appropriate value within the parentheses:

<b>Keyword</b>	<b>Value</b>
SOURCETYPE	The short name of the source object type.
TARGETTYPE	The short name of the target object type.
UUI_short_name	The short name of a UUI property of the object type.

Completely enclose in parentheses all the properties and values after the SOURCEKEY and TARGETKEY keywords.

## Associating contact names with objects

Figure 8 on page 49 shows an example of tag language to create a linked relationship between two Grouping objects. The example assumes that you already created the source and target objects.

---

```
:ACTION.RELATION(ADD)
:RELTYPE.TYPE(LINK) SOURCETYPE(TABLES) TARGETTYPE(TABLES)
:INSTANCE.SOURCEKEY(DBNAME(DGWDATA) OWNER(USERID)
    TABLE(COMPONENTS))
    TARGETKEY(DBNAME(DGWDATA) OWNER(USERID) TABLE(CUSTSHIP))
```

---

Figure 8. Linking two objects with tag language

In this example, two relational tables are linked.

---

## Associating contact names with objects

You can help information catalog users in your organization find a person responsible for the actual data described by objects. You can identify contacts for objects using the DataGuide windows or tag language.

### Associating contact names with objects using DataGuide windows

- 1 Create a Contact category object.
- 2 Either create a Grouping or Elemental category object (see “Creating an object” on page 37 for details) or locate an existing Grouping or Elemental object in one of the following windows:
  - Search Results
  - Collection
  - Found In
  - Subjects
  - Tree View
  - Linked With
- 3 Click mouse button 2 on the Grouping or Elemental object.
- 4 Select the **Contacts > Associate** menu choice from the pop-up menu.

The Associate Contacts window opens.
- 5 To add a contact to the object:
  - a Click on the **Search** push button.

The Define Search - Contacts window opens. Use this window to search for contact objects you want to include. Objects that fit your search criteria are returned to the Associate Contacts window in the **Available contacts** list box.
  - b Select one or more objects in the **Available contacts** list.
  - c Click on the **>** push button to move selected objects to the **Contacts** list.

## Associating contact names with objects

- 6 To delete a contact from an object:
  - a Select one or more objects in the **Contacts** list.
  - b Click on the < push button to move selected objects out of the **Contacts** list.
- 7 Click on the **Associate** push button. You must close and reopen the window for your updates to appear.

To close the window without adding or deleting contacts, click on the **Cancel** push button.

## Associating contact names with objects using DataGuide tag language

To add a Contact object to another DataGuide object with tag language, specify a Contact relationship between them.

- 1 To add a contact, enter the following line in your tag language file:

```
:ACTION.RELATION(ADD)
```
- 2 To delete a contact, enter the following line in your tag language file:

```
:ACTION.RELATION(DELETE)
```
- 3 Enter the following lines, filling in the type of the object that is associated with the contact for SOURCETYPE:

```
:RELTYPE.TYPE(CONTACT) SOURCETYPE(short_name_of_object_type)
TARGETTYPE(short_name_of_contact_object)
```
- 4 Enter the following lines, filling in the UUI properties and property values for the object that is associated with the contact:

```
:INSTANCE.SOURCEKEY(UUI_short_name(value_for_property)
UUI_short_name(value_for_property)
UUI_short_name(value_for_property))
```
- 5 Enter the following lines, filling in the UUI properties and property values of the Contact object:

```
TARGETKEY(UUI_short_name(value_for_property)
UUI_short_name(value_for_property)
UUI_short_name(value_for_property))
```

After each keyword, type an appropriate value within the parentheses:

Keyword	Value
SOURCETYPE	The short name of the source object type.
UUI_short_name	The short name of a UUI property of the object type.

Completely enclose in parentheses all the properties and values after the SOURCEKEY and TARGETKEY keywords.

## Associating comments and objects

Figure 9 on page 51 shows an example of tag language to add a contact to a database object. The example assumes that you already created the source and target objects.

---

```
:ACTION.RELATION(ADD)
:RELTYPE.TYPE(CONTACT) SOURCETYPE(DATABASE) TARGETTYPE(CONTACT)
:INSTANCE.SOURCEKEY(SERVER(STL11W71) DBNAME(DGWDATA)
    DBTYPE(RELATIONAL))
    TARGETKEY(NAME(Robin Noble-Thomas) RESPONSE(EUI team lead))
```

---

Figure 9. Adding a contact to a table object with tag language

In this example, the Contact object identified by NAME(Robin Noble-Thomas) RESPONSE(EUI team lead) is added to the object identified by SERVER(STL11W71) DBNAME(DGWDATA) DBTYPE(RELATIONAL).

---

## Associating comments and objects

With DataGuide you can attach comments to objects like you attach a “sticky” note to a page in a book. The note might contain additional information for yourself or other users of the book, and you can later remove the note and discard it.

In DataGuide, a *comment* is an object that annotates another object. For example, you might attach a comment to a chart object that contains notes about the data in the chart.

### Creating a comment

You can create comments about specific DataGuide objects. You can use comments as a reminder to yourself or as a way to communicate with other DataGuide users of your information catalog in your organization.

Start from one of the following windows:

- Search Results
- Collection
- Found In
- Contacts
- Subjects
- Tree View
- Linked With
- Attachments

- 1 Click mouse button 2 on the object to which you want to attach a comment.  
This step is not necessary if you started from the Attachments window.
- 2 Select the **Attachments > Create comment** menu choice from the pop-up menu.  
If you are starting from the Attachments window, select **Attachments > Create comment** from the menu bar.

## Associating comments and objects

The Create Comment window opens.

- 3 Type a name for the comment in the **Name** field.
- 4 Assign a status to the comment.
- 5 (Optional) In the **Actions** field, type specific actions you want someone to take based on the text of the comment.
- 6 Type the complete text of the comment in the **Description** field.
- 7 Click on the **Create** push button to create the comment and attach it to the specified object.

To close the window without creating a comment, click on the **Cancel** push button.

## Copying a comment

You can copy existing comments to create unattached comments, which you can later attach to an object.

Start from one of the following windows:

Search Results  
Collection  
Attachments

- 1 Click mouse button 2 on the comment that you want to copy.
- 2 Select the **Copy** menu choice from the pop-up menu.

The Copy Comment window opens.

- 3 (Optional) Change any of the following values:

Name  
Actions  
Status  
Description

- 4 Click on the **Copy** push button to create the copied comment.

To close the window without copying a comment, click on the **Cancel** push button.

## Updating a comment

You can change various values, including the text, of existing comments.

Start from one of the following windows:

Search Results  
Collection  
Attachments

- 1 Click mouse button 2 on the comment you want to update.

## Associating comments and objects

- 2 Select the **Update** menu choice from the pop-up menu.

The Update Comment window opens.

**Update Comment**

Name: DG User's Guide related comment

Created by: USERID      Creation date: 1996-10-23-17.11.270000

Actions:

Status: Open

Reference: CelDial sales for 1997

Description: Hello, it's Michelle (aka USERID). Shouldn't we have a related contacts for the CelDial sales spreadsheets to match the UG scenario? They can be the names you've created for the sample data base, but right now they don't seem to exist.

Update    Cancel    Help

- 3 Change at least one of the following values:

- Name
- Actions
- Status
- Description

- 4 Click on the **Update** push button to change the comment.

To close the window without updating the comment, click on the **Cancel** push button.

## Deleting a comment

You can delete an existing comment, whether or not it is attached to an object.

Start from one of the following windows:

- Search Results
- Collection
- Attachments

- 1 Click mouse button 2 on the comment you want to delete.
- 2 Select the **Delete** menu choice from the pop-up menu.

The Delete window opens.

## Associating comments and objects

- 3 (Optional) Deselect any comments in the **Object** list box that you do not want to delete.
- 4 Click on the **Delete** push button to delete the comment.

The comment is deleted from the information catalog.

## Attaching and detaching comments and objects

You can attach existing unattached comments to specified objects. You can also detach comments from specified objects. To attach and detach existing comments, start from one of the following windows:

- Search Results
- Collection
- Found In
- Contacts
- Subjects
- Tree View
- Linked With

- 1 Click mouse button 2 on the object for which you want to change the Attachment relationships.
- 2 Select the **Attachments > Associate comments** menu choice from the pop-up menu.  
The Associate Comments window opens.
- 3 To attach additional comments to the object:
  - a In the **Available comments** list, select one or more comments that you want to attach to the selected object.
  - b Click on the **>** push button to move the selected comments to the **Current comments** list.
- 4 To detach comments from the object:
  - a In the **Current comments** list, select one or more comments that you want to detach from the selected object.
  - b Click on the **<** push button to move the selected comments to the **Available comments** list.
- 5 Click on the **Associate** push button to save the specified Attachment relationships.

To close the window without changing the Attachment relationships, click on the **Cancel** push button.



## Associating a program with object types

---

### Associating a program with object types

DataGuide makes it easy to start a program that can retrieve the actual data that an object describes. For example, if you have objects describing graphic charts, you can set up a graphic program, such as CorelDRAW!, so that you can retrieve the actual charts for editing, copying, or printing.

DataGuide for Windows can start any program that runs on the Windows platform you are using, or that can be started from an MS-DOS command prompt.

A single object type can start more than one program (for example, the object type Spreadsheet can have both Lotus 1-2-3 and Microsoft Excel associated with it).

To enable an object to start a program, you create an association between a Programs object and any object type not categorized as Program.

### Creating a Programs object

You can create a Programs object using the DataGuide windows or tag language.

#### Creating a Programs object using DataGuide windows

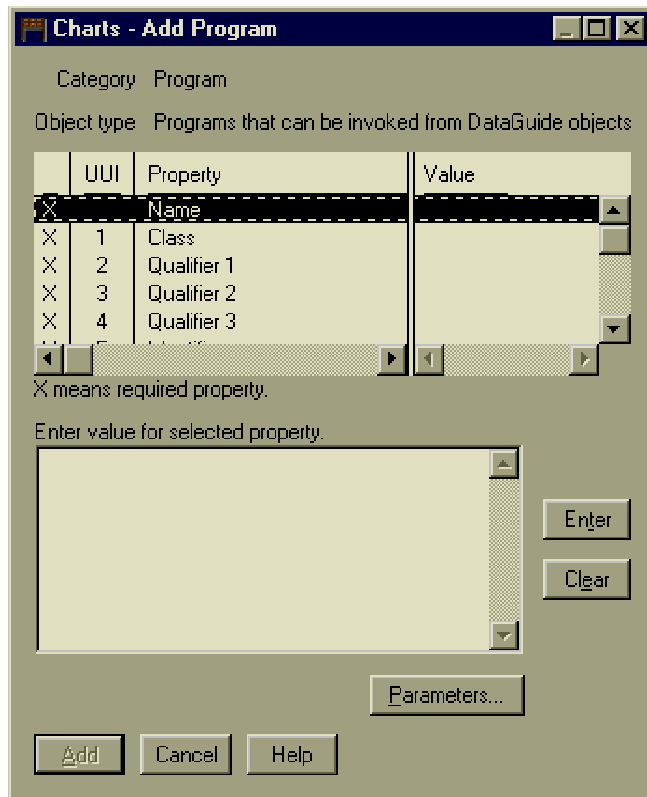
- 1 Click mouse button 2 on the **Object types** icon in the DataGuide Catalog window.
- 2 Select the **Open as > Icon list** menu choice from the pop-up menu.
- 3 Click mouse button 2 on the icon of the object type for which you want the program to start.
- 4 Select the **Associate programs** menu choice from the pop-up menu.

The Programs window displays a list of programs currently associated with the selected object type.

- 5 Click on the **Add** push button.

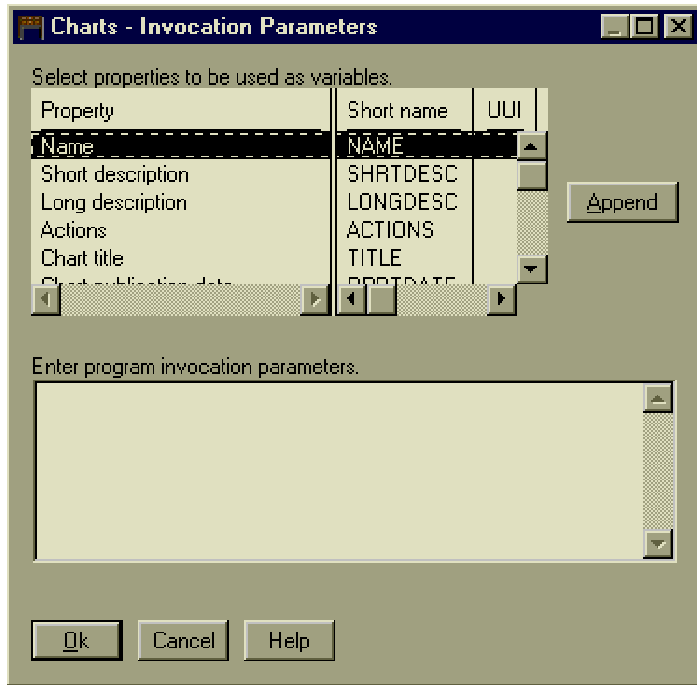
The Add Program window opens.

## Associating a program with object types



- 6 Select a property from the **Properties/Values** list box.
- 7 In the **Enter value for selected property** field, type a value for the property.  
For more specific information about entering these values, see "Supplying values for required Programs object properties" on page 57.
- 8 Click on the **Enter** push button to move the value to the **Value** column in the **Properties/Values** list box.  
If you want to erase what you entered in the **Enter value for selected property** field, click on the **Clear** push button.
- 9 Click on the **Parameters** push button to specify the properties whose values you want to be used as program parameters. The Invocation Parameters window opens.

## Associating a program with object types



10 Click on the **Add** push button.

To close the window without adding a program, click on the **Cancel** push button.

**Supplying values for required Programs object properties:** Programs objects have several required properties that allow you to differentiate among Programs objects when you want the same program to handle more than one object type. These properties are described in Table 11 on page 58.

## Associating a program with object types

Property	Example	Description
Name	View home page using Netscape Navigator	This value is displayed in the <b>Select one or more programs to start</b> list box when a user chooses to start a program from an object. When you are associating one program with several object types, you can enter the same value for the Name property on each.
Class	Browser	You can enter any values for these properties that help you classify and identify the Programs object. You can enter the not-applicable symbol if you don't have a value for a property. (The not-applicable symbol is a hyphen unless you identified a different symbol when you created the information catalog.)
Qualifier 1	Navigator	
Qualifier 2	Windows 95	
Qualifier 3	3.	
Identifier	start netscape.exe	
<b>Note:</b> 1. The combination of the Class, Qualifiers 1, 2, and 3, and the Identifier properties must be unique across for all program objects in the information catalog. Each instance of an object type must be different.		

For the property called Start by invoking, enter the file name of the program and the recommended starting parameters. For Windows NT, Windows 95 and Windows 98, the recommended starting parameter is `START filename.exe`. The PATH statement must contain the directory where the program is located.

If the file name of the program is in high performance file system (HPFS) format and contains blanks, surround the path and file name of the program with two sets of quotation marks, as in this example:

```
" "D:\PROGPATH\My Program.EXE" "
```

If the program name contains blanks, you can't specify any other start options for the Start by invoking property. Instead, enter the options in the Parameters property. Note that you should not change the value of the HANDLES property.

### Creating a Programs object using DataGuide tag language

Enter the following lines in your tag language file:

## Associating a program with object types

```
:ACTION.OBJINST(ADD)
:OBJECT.TYPE(PROGRAMS)
:INSTANCE.NAME(name_of_program)
    UUICLASS(class_of_program)
    UUIQUAL1(identifier)
    UUIQUAL2(identifier)
    UUIQUAL3(identifier)
    UUIIDENT(identifier)
    HANDLES(short_name_of_object_type)
    STARTCMD(command_to_start_program)
    PARMLIST(list_of_program_parameters)
    SHRTDESC(description_of_program)
```

After each keyword, type an appropriate value within the parentheses:

Keyword	Value
NAME	The external name (up to 80 characters) of the program.
UUICLASS	A classifying property, such as Spreadsheet.
UUIQUAL1, 2, 3	Additional identifying properties.
UUIIDENT	Additional identifying property.
HANDLES	The short name of the object type that this Programs object handles. This property is required.
STARTCMD	The command required to start the program. This property is required. You do not need to start the command processor (command.exe) as part of the value of the STARTCMD keyword.
PARMLIST	The parameters you want to start the program with.
SHRTDESC	A short description of the program.

Figure 10 shows an example of tag language that sets up a program to handle spreadsheet objects. The example assumes that you have the object type SPRDSHET in your information catalog.

---

```
:ACTION.OBJINST(ADD)
:OBJECT.TYPE(PROGRAMS)
:INSTANCE.NAME(Lotus 1-2-3 for Windows)
    UUICLASS(SPRDSHET)
    UUIQUAL1(Lotus 1-2-3)
    UUIQUAL2(Windows)
    UUIIDENT(123w.exe)
    HANDLES(SPRDSHET)
    STARTCMD(start /f /win 123w.exe)
    PARMLIST(%LISTSRCE%)
    SHRTDESC(Lotus 1-2-3 for Windows)
```

---

Figure 10. Setting up a program to handle spreadsheet objects

## Associating a program with object types

### Copying a program associated with an object type

You can create a program association based on values of an existing association:

- 1 Click mouse button 2 on the **Object types** icon in the DataGuide Catalog window.
- 2 Select the **Open as > Icon list** menu choice from the pop-up menu.
- 3 Click mouse button 2 on the icon of the object type from which you want to copy the program association.
- 4 Select the **Associate programs** menu choice from the pop-up menu.  
The Programs window displays a list of programs currently associated with the selected object type.
- 5 Select the program you want to copy.
- 6 Click on the **Copy** push button.  
The Copy Program window opens.
- 7 Select a property from the **Properties/Values** list box.
- 8 In the **Enter value for selected property** field, edit the value for the property.  
If you want to erase the existing value in the **Enter value for selected property** field, click on the **Clear** push button.
- 9 Click on the **Enter** push button to move the changed value to the **Value** column in the **Properties/Values** list box.
- 10 Click on the **Parameters** push button to update the list of properties whose values you want to be used as program parameters.
- 11 Click on the **Copy** push button.  
To close the window without copying the Programs object, click on the **Cancel** push button.

### Updating a program association for an object type

You can change values for an existing association between a program and objects of a specified object type using the DataGuide windows or tag language.

#### Updating a program association for an object type using DataGuide windows

- 1 Click mouse button 2 on the **Object types** icon in the DataGuide Catalog window.
- 2 Select the **Open as > Icon list** menu choice from the pop-up menu.
- 3 Click mouse button 2 on the icon of the object type for which you want to update the program association.
- 4 Select the **Associate programs** menu choice from the pop-up menu.  
The Programs window displays a list of programs currently associated with the selected object type.

## Associating a program with object types

- 5** Select from the list the program you want to update.
- 6** Click on the **Update** push button.  
The Update Program window opens.
- 7** Select a property from the **Properties/Values** list box.
- 8** In the **Enter value for selected property** field, edit the value for the property.  
If you want to erase the existing value in the **Enter value for selected property** field, click on the **Clear** push button.
- 9** Click on the **Enter** push button to move the changed value to the **Value** column in the **Properties/Values** list box.
- 10** Click on the **Parameters** push button to update the list of properties whose values you want to be used as program parameters.
- 11** Click on the **Update** push button when you finish changing values.  
To close the window without updating the Programs object, click on the **Cancel** push button.

### Updating a program association using DataGuide tag language

You can update programs that handle objects using DataGuide's tag language. You do this the same way you update other objects with tag language. See "Updating an object" on page 40 for information.

## Creating a dictionary facility

### Disassociating a program from an object type

You can delete the association between a program and objects of a specified object type using the DataGuide windows or tag language.

#### Disassociating a program from an object type using DataGuide windows

- 1 Click mouse button 2 on the **Object types** icon in the DataGuide Catalog window.
- 2 Select the **Open as > Icon list** menu choice from the pop-up menu.
- 3 Click mouse button 2 on the icon of the object type for which you want to delete the program association.
- 4 Select the **Associate programs** menu choice from the pop-up menu.  
The Programs window displays a list of programs currently associated with the selected object type.
- 5 Select the program you want to delete.
- 6 Click on the **Delete** push button.

#### Disassociating a program from an object type using DataGuide tag language

To delete the association between a Programs object and an object type using tag language, delete the Programs object that handles the particular object type:

- 1 Enter the following lines in your tag language file:  

```
:ACTION.OBJINST(DELETE)
:OBJECT.TYPE(PROGRAMS)
```
- 2 Enter the following lines, filling in the UUI properties and property values of the object you want to delete:  

```
:INSTANCE.SOURCEKEY(UUICLASS(class_of_program)
    UUIQUAL1(identifier)
    UUIQUAL2(identifier)
    UUIQUAL3(identifier))
```

Enter only UUI properties for which you have existing values in the information catalog. Completely enclose in parentheses all the properties and values after the SOURCEKEY keyword.

---

## Creating a dictionary facility

You can set up an icon where users can quickly find definitions or synonyms of the business terms you use in your information catalog, such as a dictionary, glossary, thesaurus, or synonym finder.

When you set up this icon for the first time, it appears in every user's DataGuide Catalog window as a saved search. Whenever your users want to know what term is



## Creating a support facility

used for an item in the information catalog, they can double-click on the saved search icon to view a list of entries in the glossary.

DataGuide comes with a Glossary object type.

### ***To create a dictionary facility:***

- 1 Create a Dictionary category object type. See “Creating your own object types” on page 23 for information on creating object types.
- 2 Create new objects of this object type.
- 3 Close the DataGuide information catalog.

When you log on to DataGuide again, the new dictionary facility appears in your DataGuide Catalog window as a saved search. The new dictionary facility also appears in every DataGuide user's Catalog window.

---

## Creating a support facility

You can set up an icon to provide users with any kind of support or helpful information about your information catalog.

You might use this icon to inform your users of new items in the information catalog or announce to them when you plan to make updates to the information catalog.

When you set up this icon for the first time, it appears in every user's DataGuide Catalog window as a saved search. Whenever your users want to view current entries in the support facility, they can double-click on the saved search icon to view a list of items.

DataGuide comes with a News object type.

### ***To create a support facility:***

- 1 Create a Support category object type. See “Creating your own object types” on page 23 for information on creating object types.
- 2 Create new objects of this object type.
- 3 Close the DataGuide information catalog.

When you log on to DataGuide again, the new support facility appears in your DataGuide Catalog window as a saved search. The new support facility also appears in every DataGuide user's Catalog window.

## Extracting descriptive data from other sources

---

### Chapter 5. Expanding and automating your information catalog

As you build your DataGuide information catalog, you will probably find that you need ways to expand it and automate your processes for maintaining it. You might want to exchange objects with other information catalogs or combine the contents of one information catalog with another. For example, you might decide to centralize some of your metadata, or make the contents of your information catalog available to another organization. You might want to exchange DataGuide metadata with metadata from other products.

DataGuide provides tag language for working with large amounts of descriptive data at one time and for exchanging objects among information catalogs to coordinate multiple information catalogs.

- A tag language file can contain descriptive data that you:
  - Export from another DataGuide information catalog (see “Exporting metadata” on page 72)
  - Extract from another source (see “Extracting descriptive data from other sources”)
  - Write using any word processing program (see “You can perform DataGuide tasks with the user interface or tag language” on page vii)
  - Exchange with products that generate DataGuide tag language (see “Exchanging and synchronizing metadata with Visual Warehouse and DB2 OLAP Server” on page 77)
  - Exchange with another product that produces metadata that conforms to the Metadata Interchange Specification (MDIS) (see “Exchanging MDIS-conforming metadata with other products” on page 83)
- A tag language file can comprise deletions you logged in one information catalog that you want to include in other information catalogs (see “Logging deletions from your information catalog” on page 70).

You can import a tag language file that contains descriptive data that you want to include in an information catalog. See “Importing tag language files” on page 71 for more information.

---

### Extracting descriptive data from other sources

The easiest way to fill your information catalog with descriptive data is to use existing descriptions of your organization's information resources. Many databases and desktop applications already contain valuable descriptive data that you can transfer to your information catalog.

Using the DataGuide tag language, you can:

- Extract the descriptive data
- Change the descriptive data

## Extracting descriptive data from other sources

- Add to the descriptive data (if necessary) to meet your work group's needs
- Import descriptive data into your information catalog

Extracting descriptive data also makes updating or refreshing your information catalog easier. You can edit the tag language files using any word processing program that can import and export ASCII text files.

### Extracting descriptive data with the extract programs provided by DataGuide

DataGuide comes with a set of programs that can extract descriptive data from various sources. You can install these extract programs when you first install DataGuide or at any time. DataGuide for Windows places them in \VWLIB\SAMPLES\DGEXTxxx subdirectories.

To use the extract programs, you must be familiar with the specific operating environments in which they run. See Appendix A, "DataGuide extract programs" on page 98 for information on running the specific extract programs you need.

### Writing customized descriptive data extract programs

You might need to write an extract program if you want to copy data from existing data catalogs other than the ones for which DataGuide provides extract programs. Your extract program must translate this descriptive data into the DataGuide tag language. You can then import your descriptive data directly into DataGuide as object types and objects.

#### Planning your extract program input and output

The format of your existing data is the default input format. Your extract program must produce a tag language file as output. However, the properties you decide to include about your information source determine the tag language output your extract program needs to produce.

This output tag language file can contain some or all of the following tags:

<b>:DISKNTL.</b>	Identifies the sequential number of the current storage diskette, and specifies whether this file is continued on more diskettes
<b>:ACTION.</b>	Specifies that an action (add, update, delete, append, or merge) occurs involving an object type, object, or relationship
<b>:OBJECT.</b>	Identifies an object type and its properties
<b>:PROPERTY.</b>	Identifies a property for an object type being defined
<b>:INSTANCE.</b>	Identifies an object or relationship
<b>:RELTYPE.</b>	Identifies the type of relationship being added or deleted
<b>:COMMIT.</b>	Identifies a DataGuide database commit point
<b>:COMMENT.</b>	Allows you to add comments to the tag language file
<b>:NL.</b>	Allows you to include multi-line property values (for non-UUI properties)

## Extracting descriptive data from other sources

**:TAB.** Allows you to insert tabs in non-UUI property values

For detailed information about the DataGuide tag language, see Appendix C, “Tag language” on page 115 and Appendix D, “What a tag language file should look like” on page 149.

### Formatting your output tag language file

If you store your tag language file on one or more diskettes, your extract program must place a `:DISKCNTL.` tag at the beginning of the tag language file so that DataGuide knows how many diskettes contain your file.

For example, on the first diskette, specify:

```
:DISKCNTL.SEQUENCE(1, +)
```

The `:DISKCNTL.` tag must be the first tag in the file on each diskette.

### Creating object types and objects with an extract program

In DataGuide, descriptive data is stored as properties of an object. An object type describes a set of properties that each object has. If you extract descriptive data that includes a set of properties that does not currently exist in an object type in the information catalog, your extract program must produce tag language that creates an object type.

For example, the database catalog of your database might describe several tables stored in the database. This catalog contains the following properties that you want to store in your DataGuide information catalog:

- 8-character identifier of the data source
- 10-character name of the table
- 80-character variable-length description of the table
- 8-character owner of table

To produce an object type containing these properties, your extract program must produce a tag language file containing the tags shown in Figure 11.

---

```
:ACTION.OBJTYPE(ADD)
:OBJECT.TYPE(MYTABLE)
    CATEGORY(GROUPING)
    EXTNAME(The tables on my data source)
:PROPERTY.EXTNAME(Data source name)
    DT(C) DL(8) SHRTNAME(DSNAME) UISEQ(2) NULLS(N)
:PROPERTY.EXTNAME(Name of the table)
    DT(C) DL(10) SHRTNAME(TABNAME) UISEQ(1) NULLS(N)
:PROPERTY.EXTNAME(Description of table)
    DT(V) DL(80) SHRTNAME(TABDESC) NULLS(Y)
:PROPERTY.EXTNAME(Owner of table)
    DT(V) DL(8) SHRTNAME(TABOWNER) NULLS(Y)
```

---

Figure 11. Tag language output from an extract program to create an object type

## Extracting descriptive data from other sources

When you generate a new object type, you must specify at least one property that has the UUISEQ option with a value of 1. You can specify up to four additional properties that have the UUISEQ option with a value of 2, 3, 4, or 5. UUISEQ specifies the position of the property in the UUI that uniquely identifies an object to a DataGuide information catalog.

The database catalog might have descriptive data for three tables that you want to store in your information catalog. You can write your extract program to read the descriptive data for these three tables from your database catalog, and write the tag language file that DataGuide can import to generate three objects of the MYTABLE object type.

Suppose your tables have the following properties:

---

Data source name	Table name	Table description	Owner
MYDATA	EMPLOYEE	Personnel information about company employees	LONGO
MYDATA	SALES	Data about 1997 sales-to-date	VALDEZ
MYDATA	CUSTOMER	Shipping information about customers	MARSH

---

Your extract program must produce the tags shown in Figure 12, which you must insert in the tag language file after the tags that define the object type.

---

```
:ACTION.OBJINST(ADD)
:OBJECT.TYPE(MYTABLE)
:INSTANCE.NAME(Personnel information)
    DSNAME(MYDATA)
    TABNAME(EMPLOYEE)
    TABDESC(Personnel information about company employees)
    TABOWNER(LONGO)
:INSTANCE.NAME(Annual sales information)
    DSNAME(MYDATA)
    TABNAME(SALES)
    TABDESC(Data about 1997 sales-to-date)
    TABOWNER(VALDEZ)
:INSTANCE.NAME(Customer shipping information)
    DSNAME(MYDATA)
    TABNAME(CUSTOMER)
    TABDESC(Shipping information about customers)
    TABOWNER(MARSH)
```

---

Figure 12. Tag language output for the object type MYTABLE

**Reusing existing object type definitions:** The tag language used to produce the object type definitions shipped with DataGuide is available for you to copy or include in

## Extracting descriptive data from other sources

your own extract programs. The tag language is in the VWSWINDGWINTYPES directory on the drive where you installed DataGuide.

### Merging object types and objects

You will probably use your extract program many times to extract descriptive data for refreshing your information catalog. In this case, it's important to *merge* your object types and objects so that you don't add new objects every time you import the tag language file.

*If two DataGuide information catalogs contain the same object types, you must merge the object types before you can merge any objects. You should also merge object types if you aren't sure whether the information catalogs contain the same object types. For example, you want to import the contents of an information catalog (the source) that contains a Table object type into another information catalog (the target) that also contains a Table object type. You aren't sure if the object types have the same properties. To merge objects, DataGuide requires that their object type definitions be alike. If the object types have the same properties, you can merge them without problems. If they don't, then the UII properties must be identical, and all the properties of the object type in the source information catalog must match properties of the object type in the target information catalog.*

If the object type in the source information catalog has more properties than the object type in the target information catalog, you can update the object type in the target information catalog before you merge the two.

Your extract program must create tags to merge object types and objects, as shown in Figure 13. DataGuide doesn't update an object type's external name or its icon in the process of merging.

---

```
:ACTION.OBJTYPE(MERGE)
:OBJECT.TYPE(short_name_of_object_type)
    CATEGORY(category_of_object_type)
    EXTNAME(extended_name_of_object_type)
    ICOFILE(OS/2_icon_file_name)
    ICWFILE(Windows_icon_file_name)
:PROPERTY.SHRTNAME(short_name) DT(data_type) DL(size)
    UISEQ(position_in_UII) NULLS(y_or_n) EXTNAME(extended_name)
```

---

Figure 13. Tag language output for merging object types

#### Restriction

You cannot merge the Programs or the Comments object types.

To merge objects, your extract program must create the tags shown in Figure 14 on page 69.

## Extracting descriptive data from other sources

---

```
:ACTION.OBJINST(MERGE)
:OBJECT.TYPE(short_name_of_object_type)
:INSTANCE.NAME(extended_name_of_object)
    UUI_short_name(value_for_property)
    .
    .
    short_name(value_for_property)
    .
    .
```

---

Figure 14. Tag language output for merging objects

### Committing changes to the DataGuide database

When DataGuide imports a tag language file, it stores the descriptive data defined in the tag language file in the DataGuide database. However, DataGuide does not automatically commit these additions to its database until it reaches the end of a tag language file. DataGuide looks for a checkpoint tag (:COMMIT.), which tells it to issue a commit call to the database. If the tag language file is long, your extract program should place :COMMIT. tags at regular intervals in the tag language file. Each checkpoint tag should have an identifier that is unique in your tag language file.

For example, you might want your extract program to include a :COMMIT. tag after the complete specification of an :ACTION. tag (after all the information that needs to be specified for an action). Your program can place a :COMMIT. tag in the tag language file after it generates all the tags to create an object type or an object.

If DataGuide fails to import part of your tag language file, you need to import the rest of your file starting at a successful checkpoint. DataGuide issues a rollback to the database to the point of the last checkpoint. When you try to import this file again, DataGuide starts from the checkpoint matching the ID of the last committed checkpoint. For example, you might want your program to put the following checkpoint into the tag language file after the tags that create the object type:

```
:COMMIT.CHKPID(objtype1)
```

### Restrictions on extract programs

When your extract program generates values for the properties, DataGuide does not remove leading blanks. For example, if your program generates TABNAME( EMPLOYEE) instead of TABNAME(EMPLOYEE) for a property defined to contain only 8 bytes, DataGuide returns an error message because the value is 10 bytes long instead of the 8 bytes defined for the TABNAME property in the object type definition.

If your extract program must generate tag language for values containing parentheses, it must surround these parentheses with single quotation marks so that DataGuide does not assume that the parentheses are delimiters. For example, if the value is a phone number (800) 555-1234, then your extract program needs to represent this value as PHONENUM(' ( '8 ' ) ' 555-1234).

## Importing and exporting tag language files

When DataGuide imports a tag language file, it ignores any characters with a hexadecimal value less than X'20'. Your descriptive data can contain only alphanumeric characters or timestamps.

---

### Logging deletions from your information catalog

You can keep a log of objects, object types, or relationships that are deleted from your information catalog. You can transfer the log to a tag language file and use it to duplicate the deletions in other information catalogs, for example, “shadow” information catalogs in a distributed environment.

To protect against erroneous deletions in other information catalogs, you should examine the contents of a delete history tag language file before importing it to any other information catalog, especially if you delete Grouping objects.

To work with the delete history, start from your DataGuide Catalog window.

- 1** Select **Catalog > Manage delete history** from the menu bar.  
The Manage Delete History window opens.
- 2** Select the **Record delete history** check box to begin or continue logging deletions.  
To stop logging deletions, deselect the **Record delete history** check box.
- 3** (Optional) To copy the existing log of deletions to a tag language file:
  - a** Select the **Transfer to a tag language file** check box.
  - b** Specify the directory path and name of a new or existing file to which you want to copy the log. Any information in an existing file is overwritten.
- 4** (Optional) To erase the current log of deletions, select the **Reset the delete history** check box.
- 5** Click on the **OK** push button to save your selections.  
To close the window without changing delete history selections, click on the **Cancel** push button.

---

### Importing and exporting tag language files

You can import DataGuide tag language files into your information catalog.

You can also export DataGuide objects and object types in the form of tag language files from your information catalog.

For information about importing and exporting tag language files that conform to the Metadata Interchange Specification (MDIS), see “Exchanging MDIS-conforming metadata with other products” on page 83.



## Importing and exporting tag language files

### Importing tag language files

#### Important information about importing

- You can import actions, such as deletions from an information catalog in the form of a delete history tag language file. To protect against erroneous deletions in other information catalogs, examine the contents of a delete history tag language file before importing it to any other information catalog, especially if you delete Grouping objects or object types. If you plan to import a delete history tag language file, ensure that you are not currently logging deletions; the deletions will be recorded during import and affect import performance. For information about delete history, see “Logging deletions from your information catalog” on page 70.
- Importing a tag file that was created using an OS/2 editor and that contains accented characters causes unreadable data in the DataGuide information catalog.

To import a tag language file to your DataGuide information catalog, start from your DataGuide Catalog window.

- 1 Select **Catalog > Import** from the menu bar.

The Import window opens.

- 2 In the **Import path and filename** field, type the directory path and file name of the tag language file you want to import.

If you type only the file name, DataGuide assumes the tag language file is on the drive and path pointed to by the DGWPATH environment variable.

- 3 In the **Icon path** field, type the directory path from which you want to import icon files.

- 4 In the **Log path and filename** field, type a new directory path and file name to change the file destination for messages generated during import. If you type only the file name, DataGuide places the log file on the drive and path pointed to by the DGWPATH environment variable.

- 5 Indicate where you want to begin importing the tag language file from.

- Select the **Start at beginning** radio button to start at the beginning.
- Select the **Start at checkpoint** radio button to start at the last point at which DataGuide successfully committed changes to the information catalog.

- 6 Click on the **Import** push button to begin importing the specified tag language file. The Import window remains open with a progress indicator. A message indicates when import is complete.

To close the window without importing a tag language file, click on the **Cancel** push button.

## Importing and exporting tag language files

### Exporting metadata

To export DataGuide objects from your DataGuide information catalog to a tag language file, start from one of the following windows:

- Search Results
- Collection
- Found In
- Contacts
- Subjects
- Tree View
- Attachments
- Linked With

- 1 Click mouse button 1 on the objects you want to export. On Windows platforms, hold down the Shift key to select multiple objects.
- 2 Click mouse button 2 on any one of the selected objects.
- 3 Select the **Export** menu choice from the pop-up menu.

The Export window opens.

- 4 In the **Export path and filename** field, type the directory path and file name of the tag language file to which you want to export the selected objects.

If you type only the file name, DataGuide assumes that the tag language file is on the drive and path pointed to by the DGWPATH environment variable.

Give a new name to the export tag language file each time you export DataGuide objects. DataGuide does not append to or write over export tag language files.

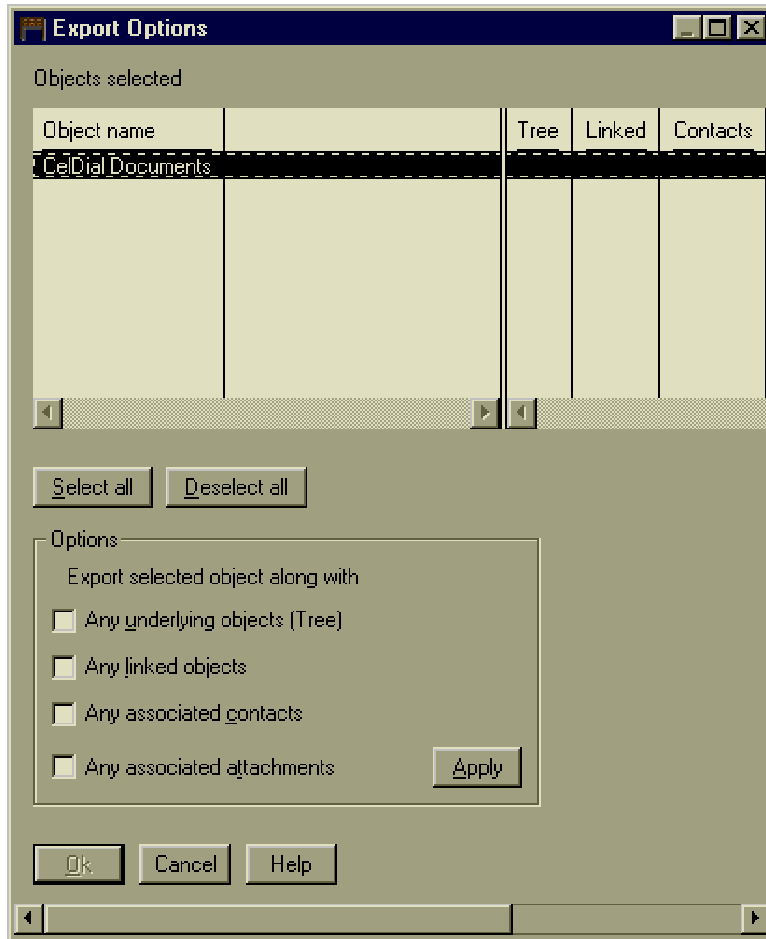
- 5 In the **Icon path** field, type the directory path to which you want to export icon files.

If icon files already exist in the path, DataGuide replaces them with the new icons.

- 6 To change the file destination for messages generated during export, type a new directory path and file name in the **Log path and filename** field. If you type only the file name, DataGuide places the log file on the drive and path pointed to by the DGWPATH environment variable.

- 7 (Optional) You specified the default export options for all objects on the **Export** page of the DataGuide Settings notebook. To specify individual export options for selected objects, click on the **Options** push button to open the Export Options window.

## Importing and exporting tag language files



Use this window to identify additional related objects to export:

- a Select one or more objects in the **Objects selected** box.
- b In the **Options** box, select one or more options for the set of selected objects:
  - All underlying objects (Tree)** Exports a Grouping object and all the objects it contains.
  - Any linked objects** Exports an object and all objects linked to it.
  - Any associated contacts** Exports an object and all Contact objects associated with it.
  - Any associated attachments** Exports an object and all comments attached to it.

## Importing and exporting tag language files

- c** To set the export options for the selected objects, click on the **Apply** push button.
  - d** (Optional) Repeat steps 7a on page 73 through 7c for different objects.
  - e** Click on the **OK** push button.
- 8** Click on the **Export** push button to begin exporting the specified objects.
- The Export window remains open with a progress indicator. A message indicates when the export process is complete.
- To close the window without exporting objects, click on the **Cancel** push button.

## Solving import and export problems

Sometimes errors in your tag language files can stop the import process. When this happens, you can look in two files, the echo and log files, that DataGuide creates during the process, to see what caused the problem.

The *echo* file records the tag language lines DataGuide has processed. The *log* file records what happens during the import or export processes.

### Reading the echo file

DataGuide gives the echo file the name of the tag language file, plus an extension of ECH. For example, if you are importing TABLEOBJ.TAG (a tag language file), TABLEOBJ.ECH is the name of the echo file. DataGuide automatically places the echo file on the drive and path pointed to by the DGWPATH environment variable.

The echo file contains uncommitted changes to the information catalog, so the few tags of an echo file tell you which line in your tag language file caused the import process to stop. Figure 15 on page 75 shows an example of an echo file.

## Importing and exporting tag language files

---

```
:COMMENT. -----
:COMMENT. Create Lotus Notes Object
:COMMENT. -----
:ACTION.OBJTYPE(MERGE)
:OBJECT.TYPE(LOTNOTE)
      CATEGORY(GROUPING)
      EXTNAME(Lotus Notes Database)
      PHYNAM(LOTUSDB)
      ICOFILE(lotusico)
:PROPERTY.SHRTNAME(LNSERVER)
      DT(C) DL(3 ) UUISEQ(1) NULLS(N) EXTNAME(Lotus Notes Server Name)
:PROPERTY.SHRTNAME(DBNAME)
      DT(C) DL(15) UUISEQ(2) NULLS(N) EXTNAME(Database filename)
:PROPERTY.SHRTNAME(MANAGERS)
      DT(T) DL(5 ) UUISEQ( ) NULLS(Y) EXTNAME(Managers)
:PROPERTY.SHRTNAME(POLICY)
      DT(V) DL(41 ) UUISEQ( ) NULLS(Y) EXTNAME(Policy Information)
:PROPERTY.SHRTNAME(NAME)
      UUISEQ(3)
:COMMIT.CKPID(Add Instances)
```

---

Figure 15. Echo file showing tag language lines that DataGuide processed

In Figure 15, the object type LOTNOTE was created with five properties. The import process ended after the checkpoint. From here, you can look in the log file to find error messages about the cause.

### Reading the log file

The log file includes the times and dates when the process started and stopped. It also includes any error messages for problems that occurred during the process.

You can specify a name for the log file or allow DataGuide to give it the name of your tag language file plus an extension of LOG. For example, if you are importing TABLEOBJ.TAG (a tag language file), TABLEOBJ.LOG is the name of the log file.

You can specify a drive and path where DataGuide places the log file. If you type only the file name, DataGuide places the log file on the drive and path pointed to by the DGWPATH environment variable.

Figure 16 on page 76 shows an example of a log file.

## Importing and exporting tag language files

---

```
Import started: Tag language file -- h:\rxlnotes\dg2lot.tag
1996/9/28 16:1 :3

FLG 5 5: Unable to create object type LOTNOTE. Reason code is 345 8.
Extended code is 8.
Import terminated with error(s). The database has been rolled back to
either the last COMMIT tag executed or the beginning of the tag language file

Import ended: Tag language file -- h:\rxlnotes\dg2lot.tag
1996/9/28 16:1 :39
```

---

*Figure 16. Log file showing reason and extended codes for problems DataGuide encountered while importing*

In this example, DataGuide could not create the object type LOTNOTE, as indicated by reason code 34508. If you look up the reason code in the online book *Visual Warehouse and DataGuide Messages & Reason Codes*, the explanation states:

The length value is invalid for the indicated property in the definition area because of the defined data type.

The book also states that the extended code indicates the sequence number of the property that caused the error.

The property specified by the extended code (property 8) has a length that is not valid for the data type. Because DataGuide generates the first five properties of all object types (OBJTYPID, INSTIDNT, NAME, UPDATIME, and UPDATEBY), the eighth property is the MANAGERS property.

In the echo file, you can see that the length for this property is defined as 50, but the type is defined as T (for TIMESTAMP). The property called Managers cannot have a data type of TIMESTAMP, so the correct data type is C (for CHAR). Edit your tag language file to correct the data type and restart the import from the last checkpoint.

Place frequent commit checkpoints in the file so that DataGuide only rolls back to the last checkpoint. See “Committing changes to the DataGuide database” on page 69 for more information about commit checkpoints.

## Exchanging and synchronizing metadata

---

### Chapter 6. Exchanging metadata with other products

You can exchange metadata with other IBM and non-IBM products. This chapter describes how to exchange synchronized metadata with the Visual Warehouse server and DB2 OLAP Server or Hyperion Essbase. It also describes how to exchange metadata with any product that conforms to the Metadata Interchange Specification (MDIS).

---

#### Exchanging and synchronizing metadata with Visual Warehouse and DB2 OLAP Server

To exchange synchronized metadata among DataGuide, Visual Warehouse, and DB2 OLAP Server/Essbase, you must complete steps from within DataGuide and Visual Warehouse.

The following list provides an overview of the tasks involved in exchanging synchronized metadata. You must complete the following tasks from within Visual Warehouse:

1. Identify Visual Warehouse metadata to exchange and export to DataGuide.
2. Create a business view that runs the synchronization program when the business view is processed.
3. Schedule the synchronized metadata exchange by scheduling the business view to run.
4. Promote that business view to test and then promote it to production.

The Visual Warehouse online help describes how to use the Visual Warehouse user interface to perform the previous tasks. The following sections describe how to prepare for metadata synchronization

#### Before you begin: establishing the environment for exchange

Before you begin the steps for exchanging synchronized metadata, you need to ensure you have established the correct environment.

- 1 Ensure that you have installed and configured the necessary Visual Warehouse components on the correct workstations to enable metadata exchange.

***For exchanging Visual Warehouse metadata with DataGuide:***

- You must have an agent installed that runs on Windows NT.
- The DataGuide Administrator must be installed locally on both the Visual Warehouse administrative client and Visual Warehouse Windows NT agent site if they are different workstations.
- Both the administrative client and Windows NT agent site must have DB2 connectivity to the Visual Warehouse control database and the DataGuide catalog APIs.

## Exchanging and synchronizing metadata

### ***For exchanging DB2 OLAP Server/Essbase metadata with DataGuide:***

- Both DataGuide Administrator and DB2 OLAP/Essbase client must be installed locally on the Visual Warehouse Windows NT agent site. The correct DB2 OLAP/Essbase client local drive is specified on the ARBORPATH environment variable.
  - The Visual Warehouse Windows NT agent site must have access to the DB2 OLAP Server Essbase APIs and the DataGuide catalog APIs.
  - The DB2 OLAP Server environment variable entries must be specified as System variables, not as User variables.
- 2 Ensure that both the DataGuide and Visual Warehouse administrator user IDs have Windows NT administrator privileges.

### **Preparing to exchange metadata with DB2 OLAP Server**

To exchange metadata with DB2 OLAP Server, you must first identify the metadata that you want to exchange and then set up the synchronized exchange. Use the following steps to create metadata objects in DataGuide and register them for synchronization.

### **Identifying DB2 OLAP Server/Essbase objects to exchange**

Complete the following steps from a workstation where DataGuide Administrator is installed:

- 1 Edit the control file `X:\VWSLIB\EXCHANGE\DG2OLAP.CTL` to identify the Essbase objects about which you want to exchange metadata with DataGuide (X is the drive where DataGuide is installed). Identify each object separately as follows:

```
servername.applicationname.databasesname.outlinename
```

where:

*servername*

Is the name of the OLAP Server (whether it is the DB2 OLAP Server or an Essbase server) on which Essbase is running.

*applicationname*

Is the name of the Essbase application that contains the database identified by *databasesname*.

*databasesname*

Is the name of the Essbase database that contains the outline identified by *outlinename*.

*outlinename*

Is the name of the Essbase outlines about which you want to exchange metadata.

For example:

```
st111w71.sample.basic.basic  
st111w71.sample.interntl.interntl  
st111w71.demo.basic.basic
```



## Exchanging and synchronizing metadata

- 2 From an MS-DOS command prompt, run the following program. Enter the required parameters on the same line, separated by blanks:

```
flgnxoln ouuserid opassword filename duserid dpassword dgname autogenerate delete
```

where:

*ouuserid*

DB2 OLAP Server/Essbase supervisor user ID

*opassword*

Password for the DB2 OLAP Server/Essbase supervisor user ID

*filename*

Full path and file name of the control file where, in step 1, you identified the DB2 OLAP Server/Essbase metadata for exchange with DataGuide

*duserid*

DataGuide administrator user ID for the information catalog that will be used for exchanging metadata

*dpassword*

Password for the DataGuide administrator user ID

*dgname*

Name of the DataGuide information catalog that will be used for exchanging metadata.

*autogenerate*

Enter:

**Y** Indicates that you want to generate the object names and descriptions in the information catalog.

**N** Indicates that you want to preserve the object names and descriptions, if they exist, in the information catalog.

*delete*

Enter:

**Y** Indicates that if objects already exist in the information catalog, they should be deleted and then added.

**N** Indicates that if objects already exist in the information catalog, they should be updated.

For example (the line break in this example is not significant):

```
flgnxoln olapadm olappass x:\vwslib\exchange\dg2olap.ct1  
dgadmin dgpass dgv5samp Y Y
```

To verify that the flgnxoln program ran successfully, view the file X:\VWSLIB\EXCHANGE\DGV3OLAP.OUT (X is the drive where you installed DataGuide).

## Exchanging and synchronizing metadata

### Setting up and scheduling synchronized metadata exchange with DB2 OLAP Server

To set up synchronized metadata exchange, you must first create and schedule a business view that runs the Synchronize DB2 OLAP to DataGuide program. When you create the business view, verify that the **Default VW AgentSite** is specified on the **Information** page of the Business View notebook.

To begin the synchronization process, you must schedule processing of the business view. Specify the schedule for the business view on the Schedule page of the Business View notebook. After scheduling the business view, you must promote it to test status, and then to production status.

After the DB2 OLAP Server business view runs, the DB2 OLAP objects that you identified in “Identifying DB2 OLAP Server/Essbase objects to exchange” on page 78 are checked for updates since the last time metadata was exchanged with the DataGuide information catalog. If there have been updates, the updated metadata is propagated to the DataGuide information catalog.

For detailed steps on creating and scheduling a business view that runs the synchronization program, see the *Synchronize metadata between DB2 OLAP Server or Hyperion Essbase and DataGuide* task in the Visual Warehouse online help.

You can check the processing status for the DB2 OLAP server synchronization business view. The processing status file is stored on `X:\VWSWIN\EXCHANGE\DG3OLAP.OUT`, where `X` is the drive where you installed DataGuide. When there is new processing status, it is appended to the existing OUT file.

### Preparing to exchange metadata with Visual Warehouse

To exchange metadata with Visual Warehouse, that you must first identify the metadata you want to exchange and then set up the synchronized exchange.

#### Identifying Visual Warehouse metadata to exchange

Table 12 provides the mapping between Visual Warehouse and DataGuide object types. Visual Warehouse uses this mapping when you export metadata to DataGuide.

Table 12. Mapping between Visual Warehouse and DataGuide object types

Visual Warehouse object type	DataGuide object type
Business view	Transformation (at the table or column level)
Column	Columns or fields
Information resource (data source or warehouse)	Databases, Files, IMS database definitions
Subject	Business subject areas
Table	IMS segments, Relational tables and views

For more information on exporting metadata to DataGuide, see the Visual Warehouse online help.

## Exchanging and synchronizing metadata

To identify Visual Warehouse metadata for exchange, complete these steps:

- 1 From a Visual Warehouse desktop, select **File > Export to DataGuide**.

The Export Warehouse window opens.

- 2 Select the information resources that contain the business views from which you want to export metadata.

- 3 Click **Select**.

The Select Business Views window opens.

- 4 Select the business views about which you want to exchange metadata.

- 5 Click **OK** to return to the Export Warehouse window.

- 6 Click **Settings**.

The Export to DataGuide: Settings window opens.

- 7 Update the fields in the Export to DataGuide: Settings window. You can use this window to specify how data is mapped when it is exported. For example, you can choose to map source tables to target tables or source columns to target columns.

- 8 Click **OK** to return to the Export Warehouse window.

- 9 Specify a file name where Visual Warehouse can write processing messages in the **File name** field.

You can view this file after the export process completes to determine whether the metadata exchange completed successfully, and if the exchange was not successful, where to find more details.

- 10 Click **Export**.

The DataGuide Settings window opens. Specify the information required to connect to your information catalog.

- 11 Click **OK** to export the business views that you selected.

### Understanding how metadata is synchronized with Visual Warehouse

After Visual Warehouse registers the metadata for an imported object in the DataGuide information catalog, you can synchronize the metadata between Visual Warehouse and DataGuide. This process is called *metadata synchronization* and occurs when you process a business view that starts a synchronization program.

When you use metadata synchronization, the metadata for an object that is registered in the DataGuide catalog is automatically or periodically updated based on the schedule of the business view associated with the synchronization program. The metadata for business views is updated when you promote the business view to production status. Metadata is not updated in DataGuide in the following situations:

When a new object is created in Visual Warehouse

## Exchanging and synchronizing metadata

When the name of an object in Visual Warehouse that you previously exported to DataGuide is changed

When an object is deleted in Visual Warehouse, information about the deleted object is stored in the Visual Warehouse control database. During metadata synchronization, Visual Warehouse propagates these deletions to the DataGuide information catalog before importing other changes into the information catalog. When metadata synchronization has completed successfully, Visual Warehouse removes the entries in the control database. Because Visual Warehouse removes the entries, Visual Warehouse can propagate deletions to only one information catalog. If you need to make the deletions to a second information catalog, you must delete those items manually.

If you change the name of an object in Visual Warehouse that you previously exported to DataGuide, you must export the object again to update the information in DataGuide. If the object you previously exported is a business view, you can export the business view with the same warehouse or subject that you exported previously. The business views that exist in the previous version of the warehouse or subject in the DataGuide catalog are not overwritten (unless you choose to export new business views that have the same names). Both the business view with the changed name and the previous version of this business view now exist in the DataGuide catalog.

When you synchronize the exchange of metadata, you can choose a static exchange, a dynamic exchange, or both a static and dynamic exchange. If you choose static synchronization, object type properties that do not change frequently (such as column names, data types and descriptions of tables) are updated in DataGuide when they change in Visual Warehouse. If you choose dynamic synchronization, object type properties that change frequently (such as the date and time when the processing of a business view updates a table) are updated in DataGuide.

### Setting up and scheduling synchronized metadata exchange

To set up synchronized metadata exchange, you must create a subject business view that runs one of the following programs:

- Synchronize dynamic VW metadata to DataGuide

- Synchronize static VW metadata to DataGuide

When you create the business view, verify that the **Default VW AgentSite** is specified on the **Information** page of the Business View notebook.

To begin the synchronization process, you must schedule processing of the business view. Specify the schedule for the business view on the Schedule page of the Business View notebook. After scheduling the business view, you must promote it to test status and then to production status.

For detailed steps on creating and scheduling a business view that runs the dynamic synchronization program, see the *Update the DataGuide information catalog with business view processing information* task in the Visual Warehouse online help. For detailed steps on creating and scheduling a business view that runs the static

## Exchanging MDIS-conforming metadata

synchronization program, see the *Update the DataGuide information catalog with definitional information* task in the Visual Warehouse online help.

After the synchronization program runs, the registered metadata is checked for updates since the last time metadata was exchanged with the DataGuide information catalog.

Table 13 shows where you can check the processing status for the synchronization business views. X is the drive where you installed DataGuide.

Business view	Processing status file	When there is new processing status:
Synchronize static VW metadata to DataGuide	X:\VWSWIN\LOGGING\DGV3XCHG.OUT	The OUT file is replaced
Synchronize dynamic VW metadata to DataGuide	X:\VWSWIN\LOGGING\DGV3VWSD.OUT	It is appended to the existing OUT file.

## Exchanging MDIS-conforming metadata with other products

DataGuide includes predefined object types that can be exchanged with metadata from other Visual Warehouse components and other MDIS-conforming products from IBM and other companies. This section describes how to use the MDIS conversion utility to convert MDIS-conforming metadata from another product into a DataGuide tag language file or to convert DataGuide metadata into an MDIS-conforming tag language file.

This section also describes how to import MDIS metadata directly into, and export MDIS metadata directly from, your DataGuide information catalog.

For information about the Metadata Interchange Specification, including complete MDIS object type definitions, visit the Meta Data Coalition's Web site at <http://www.MDCinfo.com>.

### Note to those currently using MDIS with other products and Visual Warehouse

**3.1:** If you already had MDIS configuration and profile files, the Visual Warehouse installation program did not overwrite them. However, before you use the MDIS function of DataGuide for the first time, you must merge the information in the DataGuide MDIS profile and configuration files with your existing files. Complete the following steps:

- 1 Check the MDIS environment variable setting to locate your existing MDIS profile file (MDISTOOL.PRO) and configuration file (MDISTOOL.CFG).
- 2 Using a text editor, append the contents of X:\VWSLIB\METADATA\PROFILES\MDISTOOL.PRO to your existing profile file. (X is the drive where you installed DataGuide.)

## Exchanging MDIS-conforming metadata

- 3 Using a text editor, append the contents of X:\VWSLIB\METADATA\PROFILES\MDISTOOL.CFG to your existing configuration file. (X is the drive where you installed DataGuide.)

## Exchanging metadata using the MDIS conversion utilities

You can use the MDISDGC command to convert metadata that conforms to the Metadata Interchange Specification (MDIS) into a DataGuide tag language file. You can then import the DataGuide tag language file into DataGuide using the methods described under “Importing tag language files” on page 71.

You can export metadata from DataGuide as described under “Exporting metadata” on page 72. You can convert the exported tag language file into an MDIS-conforming tag language file using the DGMDISC command.

### Converting MDIS-conforming metadata into a DataGuide tag language file

Use the MDISDGC command from a DB2 command line to convert MDIS-conforming metadata into a DataGuide tag language file, which you can then import into your DataGuide information catalog. All MDISDGC command variables are required.

```
MDISDGC userid password input_tagfile output_tagfile logfile
```

#### *userid*

Your Windows NT user ID. User IDs for Windows NT are case-sensitive; you must enter them exactly as specified.

#### *password*

The password for *userid*. Passwords for Windows NT are case-sensitive; you must enter them exactly as specified.

#### *input\_tagfile*

Full path and file name of the MDIS-conforming tag language file that you want to convert into a DataGuide tag language file. If you type only the file name, DataGuide assumes the *input\_tagfile* is on the drive and path pointed to by the DGWPATH environment variable.

#### *output\_tagfile*

Full path and file name for the DataGuide tag language file created by MDISDGC. If you type only the file name, DataGuide places the *output\_tagfile* on the drive and path pointed to by the DGWPATH environment variable.

#### *logfile*

Full path and file name destination for messages generated during conversion by DataGuide. If you type only the file name, DataGuide places the *logfile* on the drive and path pointed to by the DGWPATH environment variable.

## Exchanging MDIS-conforming metadata

### Converting a DataGuide tag language file into MDIS-conforming metadata

Use the DGMDISC command from a DB2 command line to convert a DataGuide tag language file into MDIS-conforming metadata, which you can then use with other MDIS-compliant products. All DGMDISC command variables are required.

```
DGMDISC userid password input_tagfile output_tagfile logfile
```

#### *userid*

Your Windows NT user ID. User IDs for Windows NT are case-sensitive; you must enter them exactly as specified.

#### *password*

The password for *userid*. Passwords for Windows NT are case-sensitive; you must enter them exactly as specified.

#### *input\_tagfile*

Full path and file name of the DataGuide tag language file that you want to convert into an MDIS-conforming tag language file. If you type only the file name, DataGuide assumes the *input\_tagfile* is on the drive and path pointed to by the DGWPATH environment variable.

#### *output\_tagfile*

Full path and file name for the MDIS-conforming tag language file created by DGMDISC. If you type only the file name, DataGuide places the *output\_tagfile* on the drive and path pointed to by the DGWPATH environment variable.

#### *logfile*

Full path and file name destination for messages generated during conversion by DataGuide. If you type only the file name, DataGuide places the *logfile* on the drive and path pointed to by the DGWPATH environment variable.

### Importing MDIS-conforming tag language files

To import an MDIS tag language file directly into your information catalog, enter the DGUIDE command from a DOS command prompt. Adhere to the following rules for the command syntax:

- All the parts, except where specified, are case insensitive.
- Each keyword must be preceded by either a / or - character.
- All keywords that follow the DGUIDE command are required. All keywords that follow the /MDIS\_IMPORT keyword are required.
- Underlined choices are defaults.

## Exchanging MDIS-conforming metadata

```
DGUIDE /USERID userid /PASSWORD password /DGNAME dname/MDIS_IMPORT filename  
/LOGFILE filename name/ADMIN
```

Optional keywords:

```
/TRACE_ | 1 | 2 | 3 | 4
```

For example, to import MDIS metadata into your information catalog, type the following command (the line break in this example is not significant, continue typing until the text wraps to the next line):

```
DGUIDE /USERID longods /PASSWORD secret /DGNAME DGV5SAMP /ADMIN  
/MDIS_IMPORT c:\mdis.tag /LOGFILE c:\mdis.log
```

### **/ADMIN**

Specifies that you are logging on as an administrator. You must be logged on as an administrator to import metadata.

### **/DGNAME**

Your DataGuide information catalog's name.

If the information catalog is local, specify the database name. If the information catalog is remote, specify the alias under which it was cataloged.

Example:

```
/DGNAME DGV5SAMP
```

### **/LOGFILE**

This parameter is required.

Specifies the file destination for messages that DataGuide generates during MDIS import or MDIS export. Unless you specify a full drive, path, and file name, DataGuide places the file in the path specified on the DGWPATH environment variable. You must specify a fixed drive.

Example:

```
/LOGFILE d:\tagfile.log
```

### **/MDIS\_IMPORT**

Imports the MDIS-conforming tag language file that you specify. Unless you specify the full drive, path, and file name, DataGuide assumes that the file is in the path specified on the DGWPATH environment variable.

Example:

```
/MDIS_IMPORT d:\tagfile.tag
```

The information catalog into which you import MDIS metadata must include, but is not limited to, valid MDIS object type definitions.



## Exchanging MDIS-conforming metadata

### **/PASSWORD**

Your password for this user ID.

Example:

```
/PASSWORD secret
```

Passwords for DB2 for AIX, DB2 PE, DB2 UDB for Windows NT, and DB2 for Windows 95 databases are case sensitive; you must type them exactly as specified.

### **/TRACE**

The level of trace information to send to the DataGuide trace file. Each higher level includes the functions of the levels below it (for example 3 includes the functions of levels 0, 1, 2, and 3). You might need to specify a higher level if you call IBM Software Support to diagnose DataGuide problems.

- 0** The default. Includes all messages and warning, error, and severe error conditions.
- 1** Includes entry and exit records of the highest level DataGuide functions.
- 2** Includes extremely granular entry and exit records of the DataGuide functions.
- 3** Includes input and output parameters (excluding input or output structures).
- 4** Includes all input or output structures that are passed to and used by DataGuide.

### **/USERID**

Your DataGuide information catalog user ID. Type the user ID required by the database where the information catalog resides. For example, the user ID might be your local, LAN, OS/400, AIX, or MVS TSO user ID.

Example:

```
/USERID longods
```

## Exporting tag language files that conform to MDIS

To export an MDIS tag language file directly from your information catalog, enter the DGUIDE command from an MS-DOS command prompt. Adhere to the following rules for the command syntax:

- All the parts, except where specified, are case insensitive.
- Each keyword must be preceded by either a / or - character.
- All keywords that follow the DGUIDE command are required. All keywords that follow the /MDIS\_EXPORT keyword are required.

## Exchanging MDIS-conforming metadata

```
DGUIDE /USERID userid /PASSWORD password /DGNAME dname/MDIS_EXPORT filename  
/LOGFILE filename /OBJTYPE object_type /OBJECTS name
```

Optional keywords:

```
/ADMIN  
/TRACE_ |1|2|3|4
```

For example, to export MDIS metadata from your information catalog to a file, type the following command (the line break in this example is not significant, you can continue typing until the text wraps to the next line):

```
DGUIDE /USERID longods /PASSWORD secret /DGNAME DGV5SAMP /ADMIN  
/MDIS_EXPORT c:\mdis.tag /LOGFILE c:\mdis.log  
/OBJTYPE database /OBJECTS server 1.payroll.valdezma
```

### **/ADMIN**

Specifies that you are logging on as an administrator. If you don't specify this optional keyword for the DGUIDE command, you are logged on as a user. You can export metadata as a user, however, you cannot perform all administrator tasks.

### **/DGNAME**

Your DataGuide information catalog's name.

If the information catalog is local, specify the database name. If the information catalog is remote, specify the alias under which it was cataloged.

Example:

```
/DGNAME DGV5SAMP
```

### **/LOGFILE**

Specifies the file destination for messages that DataGuide generates during MDIS import or MDIS export.

Unless you specify a full drive, path, and file name, DataGuide places the file in the path specified on the DGWPATH environment variable. You must specify a fixed drive.

Example:

```
/LOGFILE d:\tagfile.log
```

### **/MDIS\_EXPORT**

Exports MDIS-conforming metadata into an MDIS-conforming tag language file with the name that you specify. Unless you specify the full drive, path, and file name, DataGuide places the file in the path specified on the DGWPATH environment variable.

Example:

```
/MDIS_EXPORT d:\tagfile.tag
```

## Exchanging MDIS-conforming metadata

The information catalog from which you export MDIS metadata is not limited to containing MDIS metadata, but /MDIS\_EXPORT only exports metadata that conforms to MDIS.

### /OBJECTS

This parameter is required.

Specifies the objects you want to export. Depending on the object type that you specified on the /OBJTYPE keyword, the *name* value is from three to five property values, separated by periods.

<b>/OBJTYPE</b>	<b>/OBJECTS</b>
<b>Database</b>	<i>ServerName.DatabaseName.OwnerName</i>
<b>Dimension</b>	<i>ServerName.DatabaseName.OwnerName.DimensionName</i>
<b>Subschema</b>	<i>ServerName.DatabaseName.OwnerName.SubschemaName</i>
<b>Record</b>	<i>ServerName.DatabaseName.OwnerName.RecordName</i>
<b>Element</b>	<i>ServerName.DatabaseName.OwnerName.RecordName.ElementName</i>

In the previous list, the parts of the name are represented with their MDIS name. To find the equivalent DataGuide names, see *Integrating Applications with the Visual Warehouse Solution*, available from the Visual Warehouse Web site at <http://www.software.ibm.com/data/vw/>.

### /OBJTYPE

This is a required parameter.

Specifies one of the following MDIS object types that you want to export:

- Database
- Dimension
- Subschema
- Record
- Element

The object type name is not case sensitive.

Example:

```
/MDIS_EXPORT d:\tagfile.tag /OBJTYPE record
```

### /PASSWORD

Your password for this user ID.

Example:

```
/PASSWORD secret
```

Passwords for DB2 for AIX, DB2 PE, DB2 UDB for Windows NT, and DB2 for Windows 95 databases are case sensitive; you must type them exactly as specified.

## Exchanging MDIS-conforming metadata

### **/TRACE**

The level of trace information to send to the DataGuide trace file. Each higher level includes the functions of the levels below it (for example 3 includes the functions of levels 0, 1, 2, and 3). You might need to specify a higher level if you call IBM Software Support to diagnose DataGuide problems.

- 0** The default. Level includes all messages and warning, error, and severe error conditions.
- 1** Includes entry and exit records of the highest level DataGuide functions.
- 2** Includes extremely granular entry and exit records of the DataGuide functions.
- 3** Includes input and output parameters (excluding input or output structures).
- 4** Includes all input or output structures that are passed to and used by DataGuide.

### **/USERID**

Your information catalog user ID. Type the user ID required by the database where the information catalog resides. For example, the user ID might be your local, LAN, OS/400, AIX, or MVS TSO user ID.

Example:

```
/USERID longods
```

---

### Chapter 7. Maintaining DataGuide

DataGuide administration tasks fall into two categories: *preventing* problems and *solving* problems.

---

#### Preventing problems

You can keep DataGuide running smoothly by:

- Monitoring available disk space
- Ensuring your LAN configuration provides enough resources for DataGuide
- Ensuring users can access the information catalog concurrently
- Backing up your DataGuide databases and supporting software regularly
- Reminding your users to back up their personal files regularly

Your LAN or database administrator can help with most of these tasks, or see your database documentation for more information.

#### Monitoring available disk space

Regularly monitor the space available on the server drive containing the DataGuide database, so that your organization doesn't run out of space for your information catalog as the catalog grows. If this happens, DataGuide can fail and users won't be able to access the information catalog.

Also monitor the drive on the user's workstation that contains the SWAPPER.DAT (OS/2) or paging (Windows) file. On Windows NT, you can view or modify this file by:

1. Opening the Control Panel.
2. Double-clicking **System** to open the System Properties notebook.
3. From the Performance notebook page, clicking the **Virtual Memory** push button.
4. Editing the **Total paging file size** field.
5. Clicking **OK** to close the System Properties notebook.

***If your information catalog is stored in a DB2 for OS/2 database:*** Monitor the space available on the drive where DB2 for OS/2 writes the log files it creates as it runs. If the drive doesn't have enough space for the log file, DB2 for OS/2 can fail, closing DataGuide. You might need to increase the number or size of these log files. To find out about or change the DB2 for OS/2 log files, use the DB2 for OS/2 tools.

***If your Information catalog is stored in a DB2 Universal Database:*** See the online help or the online help for the DB2 Universal Database Control Center for information on modifying the size of the log files.

## Preventing problems

### Ensuring users can access the information catalog concurrently

When you use DataGuide with a DB2 information catalog, each user or administrator who accesses your information catalog is considered to be an agent using DB2. For DataGuide information catalogs stored in DB2, the maximum number idle agents must be higher than the default, which is 3. You should also increase the maximum concurrent agents.

To determine the current setting for idle and concurrent agents, enter the following command from the DB2 Command Line Processor:

```
get database manager configuration
```

To change the number of idle agents, enter the following command:

```
update database manager configuration use max_idleagents num
```

where *num* is the new number of idle agents.

To change the number of concurrent agents, enter the following command:

```
update database manager configuration use maxagents num
```

where *num* is the new number of concurrent agents.

### Backing up DataGuide databases and configuration information

To avoid losing your DataGuide data in case of a hardware or software failure, establish a routine for backing up your DataGuide databases and configuration information, and the software that supports them.

How frequently you back up these components depends on how frequently you make changes to your information catalog and on your organization's policies for backups.

Your routine should include the following tasks:

- Backing up your LAN server system.
- Backing up each DataGuide database.  
If your DataGuide database is on DB2 for MVS, DataGuide uses two table spaces to store the data. You must back up both table spaces at the same time, because the data stored in these table spaces is interrelated.
- Backing up DataGuide configuration information.
- Backing up your data to tape, to a separate physical or LAN drive, or to diskettes.
- Backing up your data before making major changes to it.
- Backing up your data after importing tag language files that contain major changes to the DataGuide information catalog.
- Backing up your data on a weekly basis if you make frequent changes to it.

Work with your LAN or database administrator to carry out your backup routine.

## Solving problems

### DataGuide databases

Backing up DataGuide databases is crucial to ensure that you can recover your descriptive data if your databases become inconsistent or corrupted.

You can use either of two methods to back up DataGuide databases:

- Use the database management system's backup function. See the documentation for your database management system for more information.
- Export all of the data into tag language files. See "Exporting metadata" on page 72 for information.

### DataGuide configuration information

DataGuide creates some configuration information on administrators' and users' workstations. The stored information consists of user-specific data, such as collections and saved searches for a particular information catalog. DataGuide for Windows stores this information in the system registry.

Enter `REGEDIT` at an MS-DOS prompt to view the Windows system registry in the Windows Registry Editor.

To locate the working directory for DataGuide for Windows, check the `DGWPATH` environment variable. (Under Windows NT, you can find the environment variable on the Control Panel under Environment Variables. Under Windows 95, you can find the environment variable in your `AUTOEXEC.BAT` file.) Then, search for the working directory name in the Registry Editor.

There is one branch, or folder, in the registry with the suffix `INI` for each DataGuide information catalog you access. There is also one folder containing wildcard settings, the last user ID logged on with, and the last information catalog used. Your administrator folder also includes default export settings.

The online help for the Registry Editor explains how to import and export selected branches or how to restore the entire registry.

Whenever there are any major changes or additions to your information catalog, you should back up your, and all your users', DataGuide configuration information.

---

## Solving problems

DataGuide gives you some resources to help you solve problems. These resources are:

- Online information and messages
- A utility for resetting a logged-on administrator user ID
- A procedure for restoring your information catalog using backed up data
- DataGuide trace file

You can also contact IBM Software Support.

## Solving problems

### Using online information and messages

DataGuide provides extensive online information and messages to help you solve problems. When you or your users receive a message, use the online help first to resolve the problem.

You can find help for DataGuide messages and explanations, and extended codes for DataGuide reason codes in the HTML online book *Visual Warehouse and DataGuide Messages & Reason Codes*, which is located in the **Visual Warehouse** folder.

### Resolving administrator log-on problems

If DataGuide closes unexpectedly while you are working on your information catalog, you must reset any administrator user ID that was logged on when the problem occurred.

From an MS-DOS command prompt, enter the command:

```
X:\VWSLIB\BIN\CLEARKA
```

where *x* is the drive where DataGuide for Windows is installed.

When prompted, enter the name, user ID, and password for the information catalog separated by blanks. For example:

```
DGV5SAMP longods secret
```

Then open your DataGuide information catalog again.

If your information catalog is stored in a DB2 for MVS Version 4.1 database and DataGuide is forced to close with errors indicating that database communication is lost, your DB2 for MVS system administrator might need to increase the value of system parameter IDTHTOIN.

### Recovering DataGuide components and data

If you experience a hardware or software failure, you can lose your DataGuide database, your descriptive data, and parts of the DataGuide product. If you backed up the necessary components and data, you can restore your system, DataGuide program, and data.

If you experience a system failure, perform the following steps after the database server's hard disk is restored and before your users try to access the information catalog:

- 1 Recover your database management system and reinstall DataGuide, as necessary.
- 2 Restore the DataGuide databases using your backup files.
  - If your backup file is a database image copy:
    - a See the documentation for your database management system for information on restoring your DataGuide database.



## Solving problems

- b** Bind the restored databases to DataGuide by issuing the SQLBIND command from a command prompt. Table 14 on page 95 indicates the bind files you should use for the DataGuide product you are restoring.

Table 14. DataGuide Version 5.2 bind files

If you are using:	Use these bind files:
DataGuide for Windows Administrator	FLGNJKA1.BND FLGNJKW1.BND FLGNJKX1.BND
DataGuide for Windows User	FLGNJKW1.BND FLGNJKX1.BND

Specify the following parameter values for the bind files:

<b>Datetime format</b>	ISO
<b>Grant execute privilege</b>	PUBLIC
<b>Row blocking</b>	ALL (does not apply to FLGNJKA0.BND or FLGNJKA1.BND)

- If your backup file is a DataGuide tag language file produced by the DataGuide export process:
  - a** Recreate your information catalog using the **Create Information Catalog** icon in the **DataGuide** folder.
  - b** Import the tag language file to restore your data.

### Using trace files for problem diagnosis

DataGuide automatically creates a file, called a *trace file*, that can help you diagnose problems.

The trace file is created each time you log on to DataGuide. It includes the time and date you started DataGuide, product version, service level, and information about what is happening in DataGuide while it's running.

DataGuide gives the trace file the name of the information catalog you are using plus an extension of TRC. DataGuide for Windows places the trace file on the drive and path specified by the DGWPATH environment variable. (On Windows NT, you can find this variable on the Control Panel under Environment Variables. On Windows 95, you can find this variable in your AUTOEXEC.BAT file.) For example, if you are using the DGIVP information catalog, look for a trace file named DGIVP.TRC on the drive and path specified by the DGWPATH environment variable.

DataGuide overwrites the trace file each time you log on.

## Solving problems

### Important!

If DataGuide unexpectedly closes, *immediately* go to a command prompt and rename the trace file before you call IBM Software Support. If you don't do this, when you restart DataGuide you will overwrite the trace file and clear the records that you need to explain your problem.

You can trace DataGuide activities at five levels. The default level is 0. At this level, DataGuide records only error messages that you see on the screen. If DataGuide encounters a severe error, it automatically increases the trace level to record the error number and any associated extended codes. You can look up the error numbers and some extended codes in the online book *Visual Warehouse and DataGuide Messages & Reason Codes*, which is located in the **Visual Warehouse** folder. An *extended code* is an additional explanatory code that provides more information about an error in a specific situation. However, the trace file can contain extended codes for DataGuide or Structured Query Language (SQL). You might have to look in the associated message reference for additional information.

You might need higher trace levels to help IBM Software Support diagnose a problem. To do so, you must start DataGuide from a command prompt and use the /TRACE option. See “Invoking DataGuide from the command line” on page 152 for more information.

Figure 17 shows an example of a trace file.

---

```
DataGuide Trace File

Date and Time   : Fri Nov 21 15: 9:34 1998
Product Version : V5R2M
Service Level   :

Starting Trace... >> PID/TID = 83/2
Starting Trace Level:
IltyListObjTypes >> PID/TID = 83/2
New Reason Code: 31
New Extended Code: -3 8
Old Reason Code:
Old Extended Code:
-----
```

---

Figure 17. Trace file showing DataGuide reason and extended codes

### What to do with trace files:

- 1 Look up error messages in the online book *Visual Warehouse and DataGuide Messages & Reason Codes*, which is located in the **Visual Warehouse** folder.

## Solving problems

- 2 Follow the actions recommended in the book.
- 3 If you see an extended code, look it up in the following documentation:
  - For SQL codes: DB2 messages book (see “Bibliography” on page 165)
  - For DataGuide extended codes: *Visual Warehouse and DataGuide Messages & Reason Codes*.

In Figure 17 on page 96, the New Reason Code: 31000 reason code specifies an unexpected database error. The New Extended Code: -30080 is an SQL code for a communications error with the database.

## Extract programs supplied with DataGuide

---

### Appendix A. DataGuide extract programs

DataGuide gives you a set of extract programs that can extract descriptive data from several database management systems and desktop applications. This appendix briefly describes how to begin using these extractors.

---

#### Extract programs supplied with DataGuide

You can extract descriptive data from the following sources.

- BACHMAN Database Administrator
- Quattro Pro
- Business Objects Version
- CoreIDRAW!
- Harvard Graphics
- IBM DB2 for AIX
- IBM DB2 for MVS
- IBM DB2 for OS/2
- IBM DB2 for OS/390
- IBM DB2 for OS/400
- IBM DB2 Parallel Edition
- IBM DB2 UDB
- IBM DB2 UDB EEE
- IBM DB2 OLAP Server and Essbase
- IBM DataAtlas (IMS and relational definitions)
- IBM DataJoiner
- IBM DataPropagator Relational (DPROP/R)
- Lotus 1–2–3
- Lotus Approach
- Lotus Freelance Graphics
- Microsoft Excel
- Microsoft Word
- ODBC extractor
- Oracle
- Sybase
- Texas Instruments Information Engineering Facility (TI/IEF)
- WordPerfect

To use these extract programs, you must be familiar with the specific operating environments in which they run.

## Planning to run extract programs

---

### Planning to run extract programs

To effectively use the extract programs to populate a DataGuide information catalog, complete the following preparatory steps:

**1** Identify descriptive data for extraction.

The DataGuide extract programs for relational databases extract descriptive data for Table and Column types of objects.

See Appendix B, "Predefined DataGuide object types" on page 100 for a description of these object types. Because the relational catalogs of the databases don't contain all the properties that these objects require, the extract programs produce only the descriptive data for the properties that the catalogs contain. You can fill in the missing information by editing the tag language file. For complete information on writing tag language files, see Appendix C, "Tag language" on page 115.

The extract programs for desktop applications create their own object types specifically for these applications.

**Important!**

This step is especially important if you plan to extract data for storage in a DB2 for OS/400 information catalog. After you create object types in a DB2 for OS/400 information catalog, you cannot update them to add properties; you must create exactly the object types you want to fit the data you plan to extract. You might want to create separate object types that are specifically tailored to the extracted data; for example, you might already have an object type called Columns and you can create a new object type Columns2 for extracted data.

**2** Select the appropriate extract program.

When you know what descriptive data you want to extract, choose the appropriate extract program and decide if you need to modify it.

The DGWEXT program for IBM's DB2 family of relational databases needs no modification. IBM supports and services the DGWEXT program. The other extract programs that come with DataGuide are only sample programs. You might need to modify them to work with your own environment and databases.

**3** Read the documentation about the extract program you want to use.

Each extractor exists in its own subdirectory. Information about, and steps for using, each of the extractors is supplied in a README file in that extractor's subdirectory.

## Creating a sample information catalog

---

### Appendix B. Predefined DataGuide object types

DataGuide includes predefined object types that can be exchanged with metadata from other Visual Warehouse components and other MDIS-conforming products from IBM and other companies. This appendix describes all of the predefined DataGuide object types, including how the object type properties map to MDIS object types. For information about the Metadata Interchange Specification, including complete MDIS object type definitions, visit the Meta Data Coalition's Web site at <http://www.MDCinfo.com>.

DataGuide provides both the predefined object types and sample objects of each type within the sample information catalog. The sample information catalog includes at least one object type for each of the seven DataGuide categories. This appendix describes how to create the sample information catalog. For details of DataGuide object type capabilities, see "Creating your own object types" on page 23.

---

### Accessing DataGuide sample data

This section describes how you can create a sample information catalog that contains both the predefined object types and objects of those types, or how you can import only the predefined object types.

### Creating a sample information catalog

You can help your users learn to use DataGuide by setting up a sample DB2 UDB for Windows NT information catalog, with which they can experiment. The sample information catalog includes all of the predefined DataGuide object types, and objects of those types, which describe information typically required in a business setting, in this case, the fictional CelDial business environment. *Using DataGuide*, provided online in the **DataGuide** folder, uses the CelDial business scenario to introduce DataGuide using examples from this sample information catalog.

You create the DB2 database on the platform you want and then use the Create Information Catalog utility to create the sample information catalog. The sample information catalog utility populates your newly created information catalog with the sample data.

To create the sample information catalog:

- 1 To define a new information catalog, follow the steps for your database platform in Chapter 1, "Setting up the DataGuide information catalog" on page 1 or in the online help. When you create it, name your sample information catalog DGV5SAMP. If you already have an information catalog named DGV5SAMP you can name the new information catalog something else, but be sure to notify your users of the new name.
- 2 At an MS-DOS command prompt, enter:

```
X:\VWLIB\SAMPLES\SAMPDATA\DGWDEMO
```

where *x* is the drive where you installed DataGuide.

To get information about additional parameters that you can specify with this command, type `DGWDEMO ?` at the MS-DOS command prompt.

**3** Press Enter. You are prompted to change any, or all, of the following four default options.

**1** The fully qualified path for the Web server on which you installed DataGuide for the Web (for example, `dgntserv2.stl.ibm.com`).

**2** The administrator user ID for the information catalog.

**3** The password for the user ID entered in the previous step. Passwords for DB2 for AIX, DB2 PE, DB2 UDB for Windows NT, and DB2 for Windows 95 databases are case-sensitive; you must type them exactly as specified.

**4** The name of the information catalog you created for the sample data. This is DGV5SAMP, unless you named it differently because you have an existing catalog named DGV5SAMP.

After you press Enter the last time, the utility proceeds to load the sample data. The utility notifies you when it finishes loading the sample data. The sample information catalog is now ready to use.

**Note for DataGuide for the Web users:** The sample data in the `\VWSLIB\SAMPLES\SAMPDATA` directory must be copied to the DataGuide for the Web server. You must copy it to a directory named DGV5SAMP under the root directory. For example:

For Lotus Domino Go Webserver, create the directory `WEBSERVE\HTML\DGV5SAMPLE`.

For Microsoft Internet Information Server (IIS), create the directory `INETPUB\WWWROOT\DGV5SAMPLE`.

For Apache Web Server on AIX, create the directory `/usr/local/apache/htdocs/dgv5sample`.

## Initializing your information catalog with the predefined object types

If you were unable to import the common object types when you created your information catalog, and you do not want to import the entire sample tag language file, you can import the object types using the following command at an MS-DOS command prompt:

```
X:\VWSLIB\SAMPLES\SAMPDATA\DGWDEMO /T userid password dname
```

where:

*X* Is the drive where you installed DataGuide.

*userid*

Is the administrator user ID for the information catalog.

## Predefined object type models

### *password*

Is the password for *userid*. Passwords for DB2 for AIX, DB2 PE, DB2 UDB for Windows NT, and DB2 for Windows 95 databases are case sensitive; you must type them exactly as specified.

### *dgname*

Is the name of the information catalog that you want to initialize with the predefined object types.

---

## Predefined object type models

The DataGuide predefined object types follow the six data models shown in Figures 18 through 23.

Figure 18 shows the object types participating in the relational model.

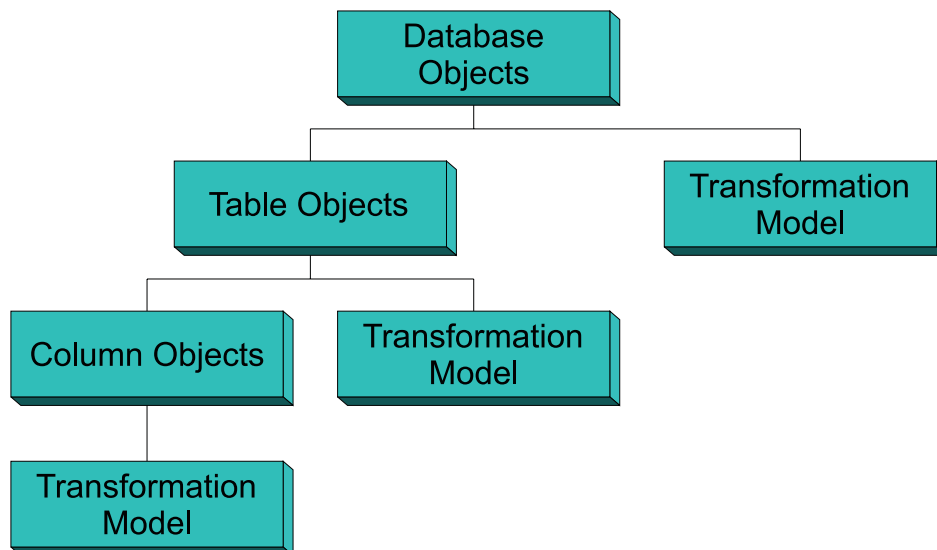


Figure 18. Relational model and the predefined object types



## Predefined object type models

Figure 19 shows the object types participating in the hierarchical models.

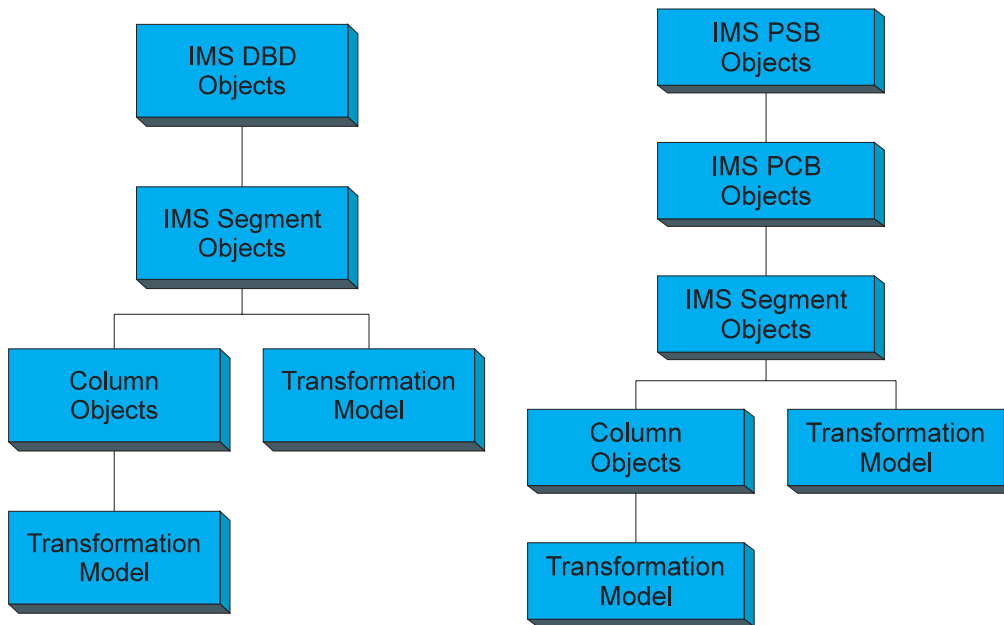


Figure 19. Hierarchical models and the predefined object types

## Predefined object type models

Figure 20 shows the object types participating in the file models.

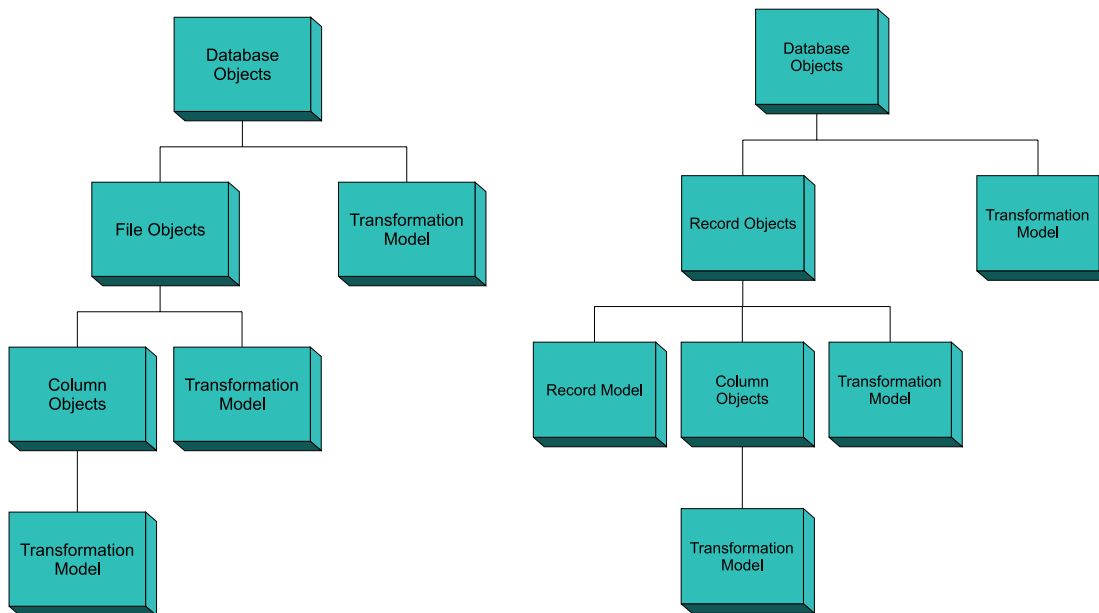


Figure 20. File models and the predefined object types

Figure 21 shows the object types participating in the multi-dimensional (OLAP) model.

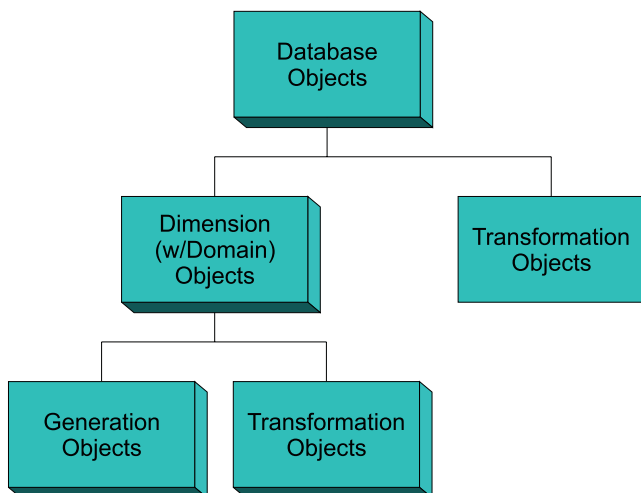


Figure 21. Multi-dimensional model and the predefined object types

## Predefined object type models

Figure 22 shows the object types participating in the transformation models.

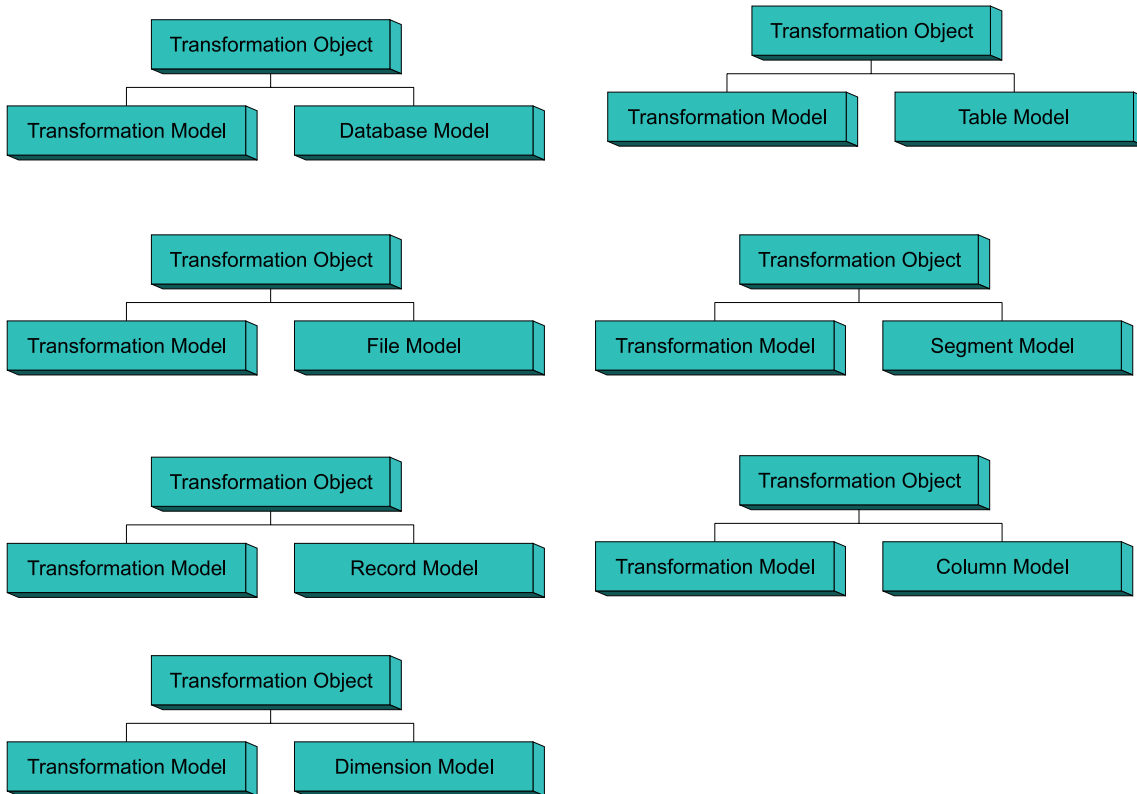


Figure 22. Transformation model and the predefined object types

## Descriptions of predefined DataGuide object types

Figure 23 shows the object types participating in the subject area model.

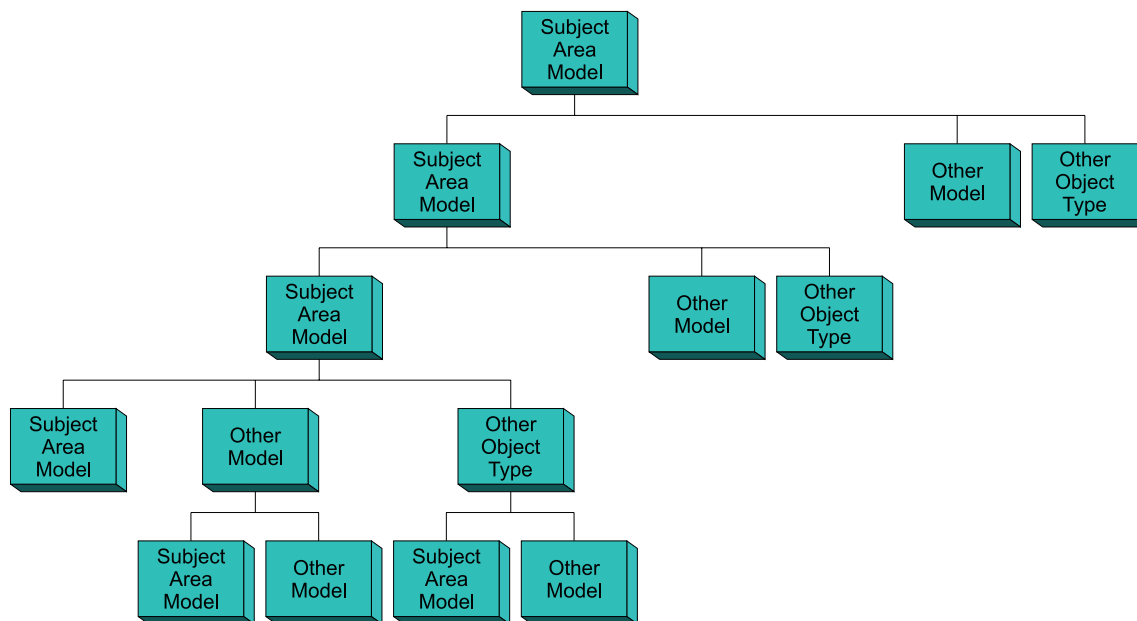


Figure 23. Subject area model and the predefined object types

---

### Predefined object type descriptions

Sample DataGuide object types are organized by category and described beginning on page 107.

For a comprehensive tables that describe each object type and its property specifications see *Integrating Applications with the Visual Warehouse Solution*, available from the Visual Warehouse Web site at <http://www.software.ibm.com/data/vw/>.

### Grouping category

The Grouping category contains the following object types:

- Application data
- Business subject areas
- Columns or fields
- Databases
- Dimensions within a multi-dimensional database
- Elements
- Files

## Descriptions of predefined DataGuide object types

- IMS database definitions (DBD)
- IMS program control blocks (PCB)
- IMS program specification blocks (PSB)
- IMS segments
- Members within a multi-dimensional database
- Multi-dimensional databases
- Records
- Relational tables and views
- Subschemas
- Transformations

### **“Application data” object type**

DataGuide uses the “Application data” object type internally for some MDIS metadata exchanges. Objects of this object type might appear in your information catalog, but you won't use this object type to create objects.

The tag language for defining the “Application data” object type is in the file FLGNYAPL.TYP in the \VSWIN\DGWIN\TYPES directory.

### **“Business subject areas” object type**

The “Business subject areas” object type represents logical groupings of objects.

The tag language for defining the “Business subject areas” object type is in the file FLGNYINF.TYP in the \VSWIN\DGWIN\TYPES directory.

### **“Columns or fields” object type**

The “Columns or fields” object type represents columns within a relational table, fields within a file, or fields within an IMS segment.

The tag language for defining the “Columns or fields” object type is in the file FLGNYCOL.TYP in the \VSWIN\DGWIN\TYPES directory.

### **“Databases” object type**

The “Databases” object type represents relational databases.

The tag language for defining the “Databases” object type is in the file FLGNYDAT.TYP in the \VSWIN\DGWIN\TYPES directory.

### **“Dimensions within a multi-dimensional database” object type**

The “Dimensions within a multi-dimensional database” object type represents dimensions within a multi-dimensional database. A dimension is comprised of members.

The tag language for defining the “Dimensions within a multi-dimensional database” object type is in the file FLGNYDIM.TYP in the \VSWIN\DGWIN\TYPES directory.

## Descriptions of predefined DataGuide object types

### **“Elements” object type**

The “Elements” object type represents MDIS Element objects that do not map directly to the “Columns or fields” object type.

The tag language for defining the “Elements” object type is in the file FLGNYELE.TYP in the \VSWIN\DGWIN\TYPES directory.

### **“Files” object type**

The “Files” object type represents a file within a file system.

The tag language for defining the “Files” object type is in the file FLGNYFIL.TYP in the \VSWIN\DGWIN\TYPES directory.

### **“IMS database definitions (DBD)” object type**

The “IMS database definitions (DBD)” object type represents IMS database definitions.

The tag language for defining the “IMS database definitions (DBD)” object type is in the file FLGNYDBD.TYP in the \VSWIN\DGWIN\TYPES directory.

### **“IMS program control blocks (PCB)” object type**

The “IMS program control blocks (PCB)” object type represents IMS program control blocks.

The tag language for defining the “IMS program control blocks (PCB)” object type is in the file FLGNYPCB.TYP in the \VSWIN\DGWIN\TYPES directory.

### **“IMS program specification blocks (PSB)” object type**

The “IMS program specification blocks (PSB)” object type represents IMS program specification blocks.

The tag language for defining the “IMS program specification blocks (PSB)” object type is in the file FLGNYPSB.TYP in the \VSWIN\DGWIN\TYPES directory.

### **“IMS segments” object type**

The “IMS segments” object type represents IMS segments.

The tag language for defining the “IMS segments” object type is in the file FLGNYSEG.TYP in the \VSWIN\DGWIN\TYPES directory.

### **“Members within a multi-dimensional database” object type**

The “Members within a multi-dimensional database” object type represents a member within a multi-dimensional database. A member is part of a dimension and a dimension is part of a multi-dimensional database.

The tag language for defining the “Members within a multi-dimensional database” object type is in the file FLGNYMEM.TYP in the \VSWIN\DGWIN\TYPES directory.

## Descriptions of predefined DataGuide object types

### **“Multi-dimensional databases” object type**

The “Multi-dimensional databases” object type represents multi-dimensional databases.

The tag language for defining the “Multi-dimensional databases” object type is in the file FLGNYOLA.TYP in the \VSWIN\DGWIN\TYPES directory.

### **“Records” object type**

The “Records” object type represents MDIS Record objects that do not map directly to the “Files” or “Relational tables or views” object types. Records are comprised of Elements.

The tag language for defining the “Records” object type is in the file FLGNYREC.TYP in the \VSWIN\DGWIN\TYPES directory.

### **“Relational tables and views” object type**

The “Relational tables and views” object type represents tables or views of relational databases.

The tag language for defining the “Relational tables and views” object type is in the file FLGNYTAB.TYP in the \VSWIN\DGWIN\TYPES directory.

### **“Subschemas” object type**

The “Subschemas” object type represents logical groupings of records within a database.

The tag language for defining the “Subschemas” object type is in the file FLGNYSUB.TYP in the \VSWIN\DGWIN\TYPES directory.

### **“Transformations” object type**

The “Transformations” object type represents expressions or logic used to populate columns of data within the target relational database. Transformations objects indicate either the expression used to convert source operational data to target columns, or the one-to-one mapping of source fields to target columns.

The tag language for defining the “Transformations” object type is in the file FLGNYFLT.TYP in the \VSWIN\DGWIN\TYPES directory.

## **Elemental category**

The Elemental category contains the following object types:

- Audio clips
- Charts
- Documents
- Images or graphics
- Internet documents
- Lotus Approach queries

## Descriptions of predefined DataGuide object types

- Presentations
- Spreadsheets
- Text-based reports
- Video clips

### **“Audio clips” object type**

The “Audio clips” object type represents files containing audio information. These objects might represent electronic (AUD files) or hardcopy (for example, CDs, tapes) audio information.

The tag language for defining the “Audio clips” object type is in the file FLGNYAUD.TYP in the \VSWIN\DGWIN\TYPES directory.

### **“Charts” object type**

The “Charts” object type represents either hardcopy or electronic charts.

The tag language for defining the “Charts” object type is in the file FLGNYCHA.TYP in the \VSWIN\DGWIN\TYPES directory.

### **“Documents” object type**

The “Documents” object type represents books, manuals, and technical papers. These publications might be printed or electronic, found locally or within a library.

The tag language for defining the “Documents” object type is in the file FLGNYDOC.TYP in the \VSWIN\DGWIN\TYPES directory.

### **“Images or graphics” object type**

The “Images or graphics” object type represents graphic images, such as bitmaps.

The tag language for defining the “Images or graphics” object type is in the file FLGNYIMA.TYP in the \VSWIN\DGWIN\TYPES directory.

### **“Internet documents” object type**

The “Internet documents” object type represents Web sites and other documents on the Internet that might be of interest.

The tag language for defining the “Internet documents” object type is in the file FLGNYINT.TYP in the \VSWIN\DGWIN\TYPES directory.

### **“Lotus Approach queries” object type**

The “Lotus Approach queries” object type represents available Lotus Approach queries for use with your organization's data.

The tag language for defining the “Lotus Approach queries” object type is in the file FLGNYAPR.TYP in the \VSWIN\DGWIN\TYPES directory.



## Descriptions of predefined DataGuide object types

### **“Presentations” object type**

The “Presentations” object type represents various hardcopy or electronic presentations. These presentations might include product, customer, quality, and status presentations.

The tag language for defining the “Presentations” object type is in the file FLGNYPRE.TYP in the \VSWIN\DGWIN\TYPES directory.

### **“Spreadsheets” object type**

The “Spreadsheets” object type represents desktop spreadsheets (for example, Lotus 1-2-3 or Microsoft Excel spreadsheets).

The tag language for defining the “Spreadsheets” object type is in the file FLGNYSSH.TYP in the \VSWIN\DGWIN\TYPES directory.

### **“Text-based reports” object type**

The “Text-based reports” object type represents either hardcopy or electronic reports.

The tag language for defining the “Text-based reports” object type is in the file FLGNYREP.TYP in the \VSWIN\DGWIN\TYPES directory.

### **“Video clips” object type**

The “Video clips” object type represents files containing video information. These objects might represent electronic (AVI files) or hardcopy (for example, video tapes or laser disks) video information.

The tag language for defining the “Video clips” object type is in the file FLGNYVID.TYP in the \VSWIN\DGWIN\TYPES directory.

## **Contact category**

The Contact category contains the “People to contact” object type.

### **“People to contact” object type**

The “People to contact” object type identifies a person or group that is responsible for single or multiple objects within the information catalog.

The tag language for defining the “People to contact” object type is in the file FLGNYCON.TYP in the \VSWIN\DGWIN\TYPES directory.

## **Dictionary category**

The Dictionary category contains the “Glossary entries” object type. The “Glossary entries” object type represents definitions for terms used in the information catalog.

The tag language for defining the “Glossary entries” object type is in the file FLGNYGLO.TYP in the \VSWIN\DGWIN\TYPES directory.

## Descriptions of predefined DataGuide object types

### Support category

The Support category contains the following object types:

- DataGuide news
- Online news services
- Online publications

#### **“DataGuide news” object type**

The “DataGuide news” object type conveys to end users information about changes to the information catalog.

The tag language for defining the “DataGuide news” object type is in the file FLGNYDGN.TYP in the \VWSWIN\DGWIN\TYPES directory.

#### **“Online news services” object type**

The “Online news services” object type represents news and information services that can be accessed online.

The tag language for defining the “Online news services” object type is in the file FLGNYOLN.TYP in the \VWSWIN\DGWIN\TYPES directory.

#### **“Online publications” object type**

The “Online publications” object type represents publications and other documents that can be accessed through online services.

The tag language for defining the “Online publications” object type is in the file FLGNYOLP.TYP in the \VWSWIN\DGWIN\TYPES directory.

### Program category

The Program category can contain only the Programs object type. The Programs object type is created when an information catalog is created and is used to define an application capable of processing a particular object type.

In the sample information catalog, DGV5SAMP , the Programs object type is named “Programs that can be invoked from DataGuide objects.”

### Attachment category

The Attachment category can contain only the “Comments” object type. The “Comments” object type is created when an information catalog is created.

The Comments object type is used to comment on other objects in the information catalog.

## Descriptions of predefined DataGuide object types

### Predefined program objects

Program object types shown in Table 15 are provided in the DataGuide sample information catalog. The table also shows the property name you use to associate with the DataGuide program object when launching a program.

Table 15. Generic predefined program objects in the sample information catalog

Type of information	Program name	Object type	Property name
Multi-media files	Microsoft Media Player	Audio clips	Audio clip filename
	Microsoft Media Player	Business subject areas	Filename
	Microsoft Media Player	Presentations	Presentation filename
	Microsoft Media Player	Video clips	Video clip filename
Bitmap files	Microsoft Paint	Images or graphics	Graphic filename
	Microsoft Paint	People to contact	Contact's picture filename
Spreadsheet files	Microsoft Excel	Spreadsheets	Spreadsheet filename
	Microsoft Paint	Spreadsheets	Spreadsheet filename
	Lotus 1-2-3	Spreadsheets	Spreadsheet filename
Web pages	Netscape Navigator	Online news	URL to access data
	Netscape Navigator	Online publications	URL to access data
	Microsoft Internet Explorer	Internet documents	URL to access data
	Microsoft Internet Explorer	Online news	URL to access data
	Microsoft Internet Explorer	Online publications	URL to access data

## Descriptions of predefined DataGuide object types

Table 16 lists specific business partners who have applications that are integrated with DataGuide. The information in this table similar to that in Table 15 on page 113.

Table 16. Predefined program objects in sample information catalog — business partners

Type of information	Program name	Object type	Property name
Lotus	Approach	Lotus Approach	Approach object filename
	Freelance Graphics	Presentations	Presentation object filename
Hyperion	Lotus 1-2-3 with Essbase Spreadsheet add-in	Spreadsheet	Spreadsheet filename
Brio	Brio Query	Text-based report	Report filename
	Netscape Navigator (use with Brio.Insights plug-in)	Text-based report	URL to access data
	Microsoft Internet Explorer (use with Brio.Insights plug-in)	Text-based report	URL to access data
BusinessObjects	BusinessObjects	Databases	None
	BusinessObjects	Report filename	Report filename
	Microsoft Excel (used with BusinessQuery add-in)	Spreadsheets	Spreadsheet filename
	Microsoft Internet Explorer (used to access WebIntelligence Java applet)	Spreadsheets	Spreadsheet filename
	Netscape Navigator (used to access WebIntelligence Java applet)	Internet documents	URL to access data
Cognos	PowerPlay	Text-based reports	Report filename
	Impromptu	Text-based reports	Report filename
	Microsoft Internet Explorer (used with Impromptu Web Query)	Internet documents	URL to access data
	Netscape Navigator (used with Impromptu Web Query)	Internet documents	URL to access data
	Netscape Navigator (used to access PowerPlay Web edition HTML pages)	Internet documents	URL to access data

---

## Appendix C. Tag language

The DataGuide tag language allows you to format your descriptive data so that you can import it into DataGuide. The tag language tells DataGuide what to do with the descriptive data that it imports. DataGuide also exports descriptive data into tag language files so that you can back up your DataGuide information catalog or transfer data from one information catalog to another.

By formatting descriptive data with the tag language, you can move descriptive data from one information catalog to another and define DataGuide object types and objects. You can also write and use extract programs to extract descriptive data from other sources, such as a relational database management system catalog, that can be imported into DataGuide. Table 17 shows the tags in the DataGuide tag language and the actions that these tags perform.

---

*Table 17. DataGuide tags*

Task	Tag names	For details
Record diskette sequence	DISKCNTL	see page 130
Identify action to be taken on input data	ACTION.OBJINST ACTION.OBJTYPE ACTION.RELATION	see page 118 see page 123 see page 127
Describe data to the DataGuide information catalog	OBJECT PROPERTY INSTANCE RELTYP	see page 136 see page 143 see page 131 see page 146
Identify when changes are committed and where check point occurs	COMMIT	see page 129
Identify user comments	COMMENT	see page 128
Format data	NL TAB	see page 136 see page 147

---

### Rules for writing tag language files

The rules explained in this section apply to all tag language files.

- Each tag name must start with a colon and end with a period. Do not put spaces between the colon and the tag name, or between the tag name and the period. For example:

```
:ACTION.OBJINST.
```

The tag name must be one of the tag names listed in Table 17.

- Include at least one keyword with all tags except COMMENT, NL, or TAB.
- Write the keyword and its value like this:

```
keyword(value)
```

- Specify keywords in any order. The only exception is that the SOURCEKEY keyword of the INSTANCE tag must be the first keyword.
- Use a blank to separate keywords.
- Enclose in parentheses the value of a keyword. If the value contains a parenthesis, enclose the parenthesis in a pair of apostrophes, for example:  
keyword(value('1'))
- Do not use OBJTYPID, INSTIDNT, UPDATIME, or UPDATEBY as property short names (*short\_names*) with the PROPERTY or INSTANCE tags.
- These property names are reserved by DataGuide:

OBJTYPID  
INSTIDNT  
NAME  
UPDATIME  
UPDATEBY

You can specify NAME as the *short\_name* on the PROPERTY tag if you are identifying NAME as a UII property for an object type when using ACTION.OBJTYPE(ADD) or ACTION.OBJTYPE(MERGE), as shown:

```
:PROPERTY.SHRTNAME(NAME) UIISEQ(1)
```

---

## How DataGuide reads tag language files

When you code a tag language file, consider how DataGuide reads tag language files:

- DataGuide reads the entire tag language file as a continuous data stream.
- DataGuide treats any character with a hexadecimal value under X'20' (except for tab and new line character tags specified in property values) as a control character and ignores that character.
- DataGuide considers a tag complete when it encounters the next tag in the tag language file.
- Tags and keywords are not translated into national languages.
- Only the values for the keywords in Table 18 are enabled for double-byte character set (DBCS) support.

---

Table 18 (Page 1 of 2). Keyword values enabled for DBCS

Tag name	Keywords	Variable value
OBJECT	EXTNAME	<i>ext_name</i>
	ICOFIELD	<i>OS/2_ICON_file_name</i>
	ICWFIELD	<i>Windows_ICON_file_name</i>
PROPERTY	EXTNAME	<i>ext_name</i>
COMMIT	CHKPID	<i>checkpt_id</i>

Table 18 (Page 2 of 2). Keyword values enabled for DBCS

Tag name	Keywords	Variable value
INSTANCE	<i>UUI_short_name</i> or <i>short_name</i>	<i>UUI_property_value</i> or <i>property_value</i>

All user-defined property values can use DBCS characters.

- DataGuide accepts DBCS blanks only in the keyword values shown in Table 19. If DBCS blanks are found anywhere else in the tag language file, errors can occur.

Table 19. Keyword values enabled for DBCS blank characters

Tag name	Keywords
ACTION	OBJTYPE OBJINST RELATION
OBJECT	All keywords
PROPERTY	All keywords
RELTYPE	All keywords
COMMIT	CHKPID
INSTANCE	<i>UUI_short_name</i> or <i>short_name</i>

## Valid data types for DataGuide descriptive data

Table 20 shows the valid data types for DataGuide descriptive data.

Table 20. Valid data types for DataGuide descriptive data

Data type	Description
CHAR	Fixed-length character string between 1 and 254 bytes long.  Pad the value on the right with trailing blanks if the value is shorter than the defined data length for the property.
TIMESTAMP	26-character timestamp in the following format: <i>yyyy-mm-dd-hh.mm.ss.nnnnnn</i>
LONG VARCHAR	Long varying-length character string between 1 and 32 700 bytes long.  You cannot specify a property with a data type of LONG VARCHAR as a UUI property.
VARCHAR	Varying-length character string between 1 and 4 000 bytes long.

DataGuide automatically removes trailing blanks from variable values and adjusts their length accordingly before validating and accepting the request.

## ACTION.OBJINST

If a required value is not specified or contains all blanks, DataGuide inserts the values shown in Table 21 on page 118.

Table 21. DataGuide-supplied values

Data type	Supplied value
CHAR	A not-applicable symbol as the first character and padded with trailing blanks to fill the defined length.
TIMESTAMP	9999-12-31-24.00.00.000000
LONG VARCHAR	A not-applicable symbol
VARCHAR	A not-applicable symbol

### How to read the tag language syntax diagrams

Code the tags and keywords exactly as they appear in the text. The tags and keywords are represented like this:

```
:tagname.keyword( ) keyword( )
```

Valid values that can be substituted for variables are described in the keyword list. The values are represented like this: *variable*

In tag descriptions, when each pair of keywords or values in a series is separated by a vertical bar, you must include one of the terms with the tag. For example, the syntax for the PROPERTY tag includes the NULLS keyword values NULLS(Y|N). You must code either NULLS(Y) or NULLS(N).

## ACTION.OBJINST

Identifies the action to be performed on the object that is described with the tags following the ACTION tag.

### Context

ACTION.OBJINST is used to create, delete, or maintain DataGuide objects.

ACTION.OBJINST is followed by one or more OBJECT and INSTANCE tags, which define the object being acted upon.

### Syntax

```
:ACTION.OBJINST(option)
```



## ACTION.OBJINST

### Options

The following options are valid for ACTION.OBJINST:

```
ADD
DELETE
DELETE_TREE_ALL
DELETE_TREE_REL
MERGE
UPDATE
```

### ACTION.OBJINST(ADD)

Adds an object.

#### Context

---

```
:ACTION.OBJINST(ADD)
:OBJECT.TYPE( )
:INSTANCE.short_name( )
:INSTANCE.short_name( )

:OBJECT.TYPE( )
:INSTANCE.short_name( )
:INSTANCE.short_name( )
```

---

Figure 24. Using the ACTION.OBJINST tag when adding objects

#### Rules

- The object must not already exist.
- Both the OBJECT tag and the INSTANCE tag must follow the ACTION.OBJINST(ADD) tag.
  - The OBJECT tag identifies the object type for the new object.
  - The INSTANCE tag specifies the property values for the new object.
- One or more INSTANCE tags can follow a single OBJECT tag, if the objects are for the same object type.
- One or more sets of an OBJECT tag with INSTANCE tags can follow an ACTION.OBJINST(ADD) tag to describe objects of different object types to be added.

### ACTION.OBJINST(DELETE)

Deletes an object.

## ACTION.OBJINST

### Context

---

```
:ACTION.OBJINST(DELETE)
:OBJECT.TYPE()
:INSTANCE.SOURCEKEY(UUI_short_name()...)

:OBJECT.TYPE()
:INSTANCE.SOURCEKEY(UUI_short_name()...)
```

---

Figure 25. Using the ACTION.OBJINST tag when deleting objects

### Rules

- The specified object must already exist.
- Both the OBJECT tag and the INSTANCE tag must follow the ACTION.OBJINST(DELETE) tag.
  - The OBJECT tag identifies the object type for the object being deleted.
  - The INSTANCE tag specifies the UUI property values for the object being deleted.
- One or more INSTANCE tags can follow a single OBJECT tag, if the objects are for the same object type.
- One or more sets of an OBJECT tag with INSTANCE tags can follow an ACTION.OBJINST(DELETE) tag to describe objects of different object types to be deleted.
- If the object being deleted is a Grouping object, it cannot contain another object. If it does, the delete fails. Use ACTION.OBJINST(DELETE\_TREE\_ALL) instead.

### ACTION.OBJINST(DELETE\_TREE\_ALL)

Deletes a Grouping category object, all Comments objects attached to it, and all ATTACHMENT, CONTACT, and LINK relationships in which it participates. Deletes all objects contained in the Grouping category object, all Comments objects attached to them, and all ATTACHMENT, CONTACT, and LINK relationships in which they participate.

### Context

---

```
:ACTION.OBJINST(DELETE_TREE_ALL)
:OBJECT.TYPE()
:INSTANCE.SOURCEKEY(UUI_short_name()...)

:OBJECT.TYPE()
:INSTANCE.SOURCEKEY(UUI_short_name()...)
```

---

Figure 26. Using the ACTION.OBJINST tag when deleting Grouping category objects and contained objects

## ACTION.OBJINST

### Rules

- The specified object must already exist and be a Grouping category object.
- Both the OBJECT tag and the INSTANCE tag must follow the ACTION.OBJINST(DELETE\_TREE\_ALL) tag.
  - The OBJECT tag identifies the object type for the object being deleted.
  - The INSTANCE tag specifies the UUI property values for the object being deleted.
- One or more INSTANCE tags can follow a single OBJECT tag, if the objects are for the same object type.
- One or more sets of an OBJECT tag with INSTANCE tags can follow an ACTION.OBJINST(DELETE\_TREE\_ALL) tag to describe objects of different object types to be deleted.

### ACTION.OBJINST(DELETE\_TREE\_REL)

Deletes a Grouping category object, all Comments objects attached to it, and all ATTACHMENT, CONTACT, CONTAIN, and LINK relationships in which it participates.

### Context

---

```
:ACTION.OBJINST(DELETE_TREE_REL)
:OBJECT.TYPE( )
:INSTANCE.SOURCEKEY(UUI_short_name( ) . . . )

:OBJECT.TYPE( )
:INSTANCE.SOURCEKEY(UUI_short_name( ) . . . )
```

---

*Figure 27. Using the ACTION.OBJINST tag when deleting Grouping category objects and relationships*

### Rules

- The specified object must already exist and be a Grouping category object.
- Both the OBJECT tag and the INSTANCE tag must follow the ACTION.OBJINST(DELETE\_TREE\_REL) tag.
  - The OBJECT tag identifies the object type for the object being deleted.
  - The INSTANCE tag specifies the UUI property values for the object being deleted.
- One or more INSTANCE tags can follow a single OBJECT tag, if the objects are for the same object type.
- One or more sets of an OBJECT tag with INSTANCE tags can follow an ACTION.OBJINST(DELETE\_TREE\_REL) tag to describe objects of different object types to be deleted.

## ACTION.OBJINST

### ACTION.OBJINST(MERGE)

Searches for the input object's UUI in the DataGuide information catalog to see whether the input object exists.

If the object exists, DataGuide updates the property values of the object in the DataGuide information catalog. If the object does not exist, DataGuide creates a new object.

#### Context

---

```
:ACTION.OBJTYPE(MERGE)
:OBJECT.TYPE() CATEGORY() EXTNAME() PHYNAME() ICOFILE() ICWFILE()
:PROPERTY.EXTNAME() DT() DL() SHRTNAME() NULLS() UUISEQ()

:ACTION.OBJINST(MERGE)
:OBJECT.TYPE()
:INSTANCE.short_name()
```

---

Figure 28. Using the ACTION.OBJINST tag when merging objects

#### Rules

- If the object exists, DataGuide updates the property values of the object in the DataGuide information catalog. If the object does not exist, DataGuide creates a new object.
- Both the OBJECT tag and the INSTANCE tag must follow the ACTION.OBJINST(MERGE) tag.
  - The OBJECT tag identifies the object type for the object being merged.
  - The INSTANCE tag specifies the property values for the object being merged.
- You must have an ACTION.OBJTYPE(MERGE) tag for a given object type earlier in the tag language file than the ACTION.OBJINST(MERGE) tag for the same object type. This is because DataGuide must ensure that the object type exists in the Information catalog it is importing into before it can try to add or update (merge) objects.

You cannot use ACTION.OBJTYPE(MERGE) for an object type belonging to the Program or Attachment categories, because you cannot create new Program or Attachment object types. However, you can use ACTION.OBJINST(MERGE) with Program objects, without specifying the ACTION.OBJTYPE(MERGE) first.

### ACTION.OBJINST(UPDATE)

Updates the value of an object.

#### Context

## ACTION.OBJTYPE

---

```
:ACTION.OBJINST(UPDATE)
:OBJECT.TYPE( )
:INSTANCE.SOURCEKEY(UII_short_name(...)) short_name()
```

---

*Figure 29. Using the ACTION.OBJINST tag when updating objects*

### Rules

- The specified object must already exist.
  - Both the OBJECT tag and the INSTANCE tag must follow the ACTION.OBJINST(UPDATE) tag.
    - The OBJECT tag identifies the object type for the object being updated.
    - The INSTANCE tag specifies the UII property values, which identify the object being updated, and the property values that are being updated.
- Only the property values specified on the INSTANCE tag are updated.

---

## ACTION.OBJTYPE

Identifies the action to be performed on the object type described with the tags following ACTION.OBJTYPE.

### Context

ACTION.OBJTYPE is used to create, delete, or maintain DataGuide object types.

ACTION.OBJTYPE is followed by one or more OBJECT and PROPERTY tags, which define the object type being acted upon.

### Syntax

```
:ACTION.OBJTYPE(option)
```

### Options

The following options are valid with ACTION.OBJTYPE:

```
ADD
APPEND
DELETE
DELETE_EXT
MERGE
UPDATE
```

## ACTION.OBJTYPE

### ACTION.OBJTYPE(ADD)

Creates the object type.

#### Context

---

```
:ACTION.OBJTYPE(ADD)
:OBJECT.TYPE() CATEGORY() EXTNAME() PHYNAME() ICOFILE() ICWFILE()
:PROPERTY.EXTNAME() DT() DL() SHRTNAME() NULLS() UISEQ()
:PROPERTY.EXTNAME() DT() DL() SHRTNAME() NULLS() UISEQ()
```

---

Figure 30. Using the ACTION.OBJTYPE tag when adding object types

#### Rules

- The object type must not exist.
- An OBJECT tag and its associated PROPERTY tags must immediately follow the ACTION.OBJTYPE(ADD) tag.
  - The OBJECT tag defines the attributes of the new object type.
  - The PROPERTY tags define the properties belonging to the new object type. DataGuide automatically defines the following required properties for every object type:
    - OBJTYPID
    - INSTIDNT
    - NAME
    - UPDATIME
    - UPDATEBY
- You cannot add object types belonging to the Program or Attachment categories.

### ACTION.OBJTYPE(APPEND)

Appends a property to an existing object type.

#### Context

---

```
:ACTION.OBJTYPE(APPEND)
:OBJECT.TYPE()
:PROPERTY.EXTNAME() DT() DL() SHRTNAME() NULLS() UISEQ()
```

---

Figure 31. Using the ACTION.OBJTYPE tag when adding properties to object types

#### Rules

- The object type must already exist.
- The property being appended must not exist.
- Do not assign the property a UISEQ value other than 0 (the default). Appended properties must be optional with NULLS(Y) and cannot be part of the UII.

## ACTION.OBJTYPE

- An OBJECT tag and one or more PROPERTY tags must immediately follow the ACTION.OBJTYPE(APPEND) tag.
  - The OBJECT tag identifies the object type being appended.
  - Each PROPERTY tag defines a property being appended.
- You cannot append to object types belonging to the Attachment category.

### ACTION.OBJTYPE(DELETE)

Deletes the object type.

#### Context

---

```
:ACTION.OBJTYPE(DELETE)  
:OBJECT.TYPE( )
```

---

*Figure 32. Using the ACTION.OBJTYPE tag when deleting object types*

#### Rules

- The object type must already exist. No objects of the object type can exist.
- One or more OBJECT tags must follow ACTION.OBJTYPE(DELETE). Each OBJECT tag identifies the object type being deleted.
- You cannot delete object types belonging to the Program or Attachment categories.

### ACTION.OBJTYPE(DELETE\_EXT)

Deletes the object type and objects of that object type.

#### Context

---

```
:ACTION.OBJTYPE(DELETE_EXT)  
:OBJECT.TYPE( )
```

---

*Figure 33. Using the ACTION.OBJTYPE tag when deleting object types and all objects of that type*

#### Rules

- The object type must exist.
- The object cannot contain objects of a different object type.
- One or more OBJECT tags must follow the ACTION.OBJTYPE(DELETE) tag. Each OBJECT tag identifies the object type being deleted.
- You cannot delete object types belonging to the Program or Attachment categories.

### ACTION.OBJTYPE(MERGE)

Checks the DataGuide information catalog for the input object type name to see if the object type already exists.

## ACTION.OBJTYPE

If the object type exists, DataGuide compares properties of the input object type to the properties of the stored object type. If the properties match, then the object types are treated as identical; if not, the input object type is invalid.

If the object type does not exist, DataGuide creates a new object type.

### Context

---

```
:ACTION.OBJTYPE(MERGE)
:OBJECT.TYPE() CATEGORY() EXTNAME() PHYNAME() ICOFILE() ICWFILE()
:PROPERTY.EXTNAME() DT() DL() SHRTNAME() NULLS() UISEQ()

:ACTION.OBJINST(MERGE)
:OBJECT.TYPE()
:INSTANCE.short_name()
```

---

Figure 34. Using the ACTION.OBJTYPE tag when merging object types

### Rules

- An OBJECT tag and its associated PROPERTY tags must immediately follow the ACTION.OBJTYPE(MERGE) tag.
  - The OBJECT tag defines the object type being merged.
  - Each PROPERTY tag defines a property belonging to the object type.
- Before you can merge objects, you must merge object types to ensure that a valid object type exists in the target information catalog. Therefore, an ACTION.OBJTYPE(MERGE) tag is required earlier in the tag language file than an ACTION.OBJINST(MERGE) tag.
- You cannot merge object types belonging to the Program or Attachment categories.

## ACTION.OBJTYPE(UPDATE)

Changes an object type external name and ICON file information.

### Context

---

```
:ACTION.OBJTYPE(UPDATE)
:OBJECT.TYPE() EXTNAME() ICOFILE() ICWFILE()
```

---

Figure 35. Using the ACTION.OBJTYPE tag when updating object types

### Rules

- The object type must already exist.
- One or more OBJECT tags must follow the ACTION tag.



## ACTION.RELATION

---

### ACTION.RELATION

Identifies the action to be performed on the relationship described with the tags following ACTION.RELATION.

#### Context

ACTION.RELATION is used to create or delete DataGuide relationships.

ACTION.RELATION is followed by one or more RELTYPE and INSTANCE tags, which define the relationships being acted upon.

#### Syntax

```
:ACTION.RELATION(option)
```

#### Options

The following options are valid with ACTION.RELATION:

ADD  
DELETE

#### ACTION.RELATION(ADD)

Defines an ATTACHMENT, CONTACT, CONTAIN, or LINK relationship.

##### Context

---

```
:ACTION.RELATION(ADD)  
:RELTYPE.TYPE() SOURCETYPE() TARGETTYPE()  
:INSTANCE.SOURCEKEY(UUI_short_name()...) TARGETKEY(UUI_short_name()...)
```

---

Figure 36. Using the ACTION.RELATION tag when adding relationships

##### Rules

- If the specified relationship does not exist, the relationship is added. If the specified relationship exists, DataGuide writes an informational message and continues processing.
- A RELTYPE tag and one or more INSTANCE tags must immediately follow the ACTION.RELATION(ADD) tag.
  - The RELTYPE tag defines the type of relationship being added and specifies the object types of the objects being associated.
  - Each INSTANCE tag specifies the UUI property values that identify the two objects being associated.

## COMMENT

### **ACTION.RELATION(DELETE)**

Deletes a relationship.

#### **Context**

---

```
:ACTION.RELATION(DELETE)
:RELTYPE.TYPE() SOURCETYPE() TARGETTYPE()
:INSTANCE.SOURCEKEY(UUI_short_name()...) TARGETKEY(UUI_short_name()...)
```

---

*Figure 37. Using the ACTION.RELATION tag when deleting relationships*

#### **Rules**

- The relationship is deleted if it exists; otherwise, DataGuide writes an informational message and continues processing.
- A RELTYPE tag and one or more INSTANCE tags must immediately follow the ACTION.RELATION(DELETE) tag.
  - The RELTYPE tag defines the type of relationship being deleted and specifies the object types of the associated objects.
  - Each INSTANCE tag specifies the UUI property values that identify the two associated objects.

---

## COMMENT

Identifies comments in the tag language file. Place this tag between any complete tag specifications in your file.

DataGuide ignores comments when importing a tag language file.

### **Syntax**

```
:COMMENT.your comments
```

---

```
:COMMENT.This is the text of a comment.
```

---

*Figure 38. Example of a COMMENT tag*

### **Rules**

- You cannot place a COMMENT tag between another tag and its keywords or between keywords.

## COMMIT

- The comment text must not contain any DataGuide tags (for example `:ACTION.`) because each tag is ended by either the end of the file or by the beginning of the next valid DataGuide tag.

---

### COMMIT

Identifies a commit point. Requests that DataGuide commit the current changes to the database.

If DataGuide encounters an error while importing a tag language file, it rolls back all changes made to the DataGuide information catalog since the last time changes were committed.

Include COMMIT checkpoints at regular intervals so that you import DataGuide tag language files more efficiently.

Including COMMIT checkpoints before and after defining or deleting object types, sets of objects, and sets of relationships can help maintain the integrity of your descriptive data.

Regular COMMIT checkpoints limit the number of changes DataGuide cancels when it rolls back the DataGuide information catalog.

Frequent COMMIT checkpoints make the echo file easier to read if there are errors in the tag language file. When the COMMIT tag is processed successfully, DataGuide clears the echo file of the tags that were processed before the COMMIT tag, so that the echo file only contains tags describing uncommitted changes.

### Context

Place this tag after one or more complete action specifications (a set of ACTION, OBJECT, RELTYPE, and INSTANCE tags).

### Syntax

```
:COMMIT.CHKPID(checkpt_id)
```

---

```
:COMMIT.CHKPID(Added_relationships)
```

---

*Figure 39. Example of a COMMIT tag*

### Keywords

#### CHKPID

Required keyword.

## DISKCNTL

### *checkpt\_id*

An identifier that DataGuide saves when it processes a COMMIT tag.

If the import of a tag language file fails after a COMMIT tag is processed successfully, you need to import the rest of the tag language file starting at the last checkpoint (an option that is available with the Import function). DataGuide uses the stored *checkpt\_id* to locate the proper COMMIT tag.

The value of *checkpt\_id* must be unique within each tag language file. Otherwise, the results of restart processing are unpredictable.

The maximum length of *checkpt\_id* is 26 characters.

*checkpt\_id* is not case-sensitive.

## Rules

Specify a COMMIT tag when the data is consistent.

To prevent the target information catalog transaction log from filling up, specify COMMIT tags at regular intervals in the tag language file.

An ACTION tag must follow the COMMIT tag, if additional data in the same tag language file needs to be processed.

---

## DISKCNTL

Identifies the diskette sequence number when the tag language file is stored on one or more diskettes.

## Context

When one tag language file is stored on or more diskettes, DISKCNTL is the first tag on each diskette.

## Syntax

```
:DISKCNTL.SEQUENCE(nn, + | -)
```

---

```
:DISKCNTL.SEQUENCE( 1, +)
```

---

*Figure 40. Example of a DISKCNTL tag for the first of a sequence of diskettes*

## Keywords

### **SEQUENCE**

Required keyword

## INSTANCE

*nn*

A one-digit or two-digit number indicating the number of the diskette in sequence.

The first number for any sequence of disks must be 1 or 01. This value increases by 1 for subsequent diskettes, so that the numbers for a set of three diskettes is 1, 2, and 3, or 01, 02, and 03.

+

Additional diskettes containing the tag language file follow this one.

-

The last or only diskette containing the tag language file.

### Rules

If this tag is specified, it must be the first tag in each tag language file. If the tag is missing and the tag language file is on diskette, the import program assumes that the tag language file is contained on one diskette.

If a tag language file is stored on the fixed disk, this tag is not applicable. If the tag is present, it is ignored.

---

## INSTANCE

Defines or identifies objects or relationships to be acted upon.

### Context

This tag is required following:

**:ACTION.OBJINST** The INSTANCE tag follows an OBJECT tag.

**:ACTION.RELATION** The INSTANCE tag follows a RELTYPE tag.

### Syntax

There are four formats for the INSTANCE tag, depending on the format of the ACTION tag:

#### **ACTION.OBJINST(ADD) or ACTION.OBJINST(MERGE)**

Adding or merging objects

```
:INSTANCE.short_name (property_value) . . .
```

#### **Context**

---

```
:ACTION.OBJINST(ADD)
```

```
:OBJECT.TYPE()
```

```
:INSTANCE.short_name()
```

---

Figure 41. Using the INSTANCE tag when adding objects

## INSTANCE

---

```
:ACTION.OBJINST(MERGE)
:OBJECT.TYPE()
:INSTANCE.short_name()
:short_name()
:short_name()
```

---

Figure 42. Using the *INSTANCE* tag when merging objects

### Keywords

#### *short\_name*

Identifies each property by its 8-character short name. This value is not case sensitive; you can specify this value using uppercase or lowercase characters. If an *INSTANCE* tag has multiple short names associated with it, use only one *INSTANCE* tag followed the short names as shown in Figure 42.

#### *property\_value*

Specifies the value of the property for the given object. This value is case sensitive.

### Rules

- When adding an object:
  - You must specify all UUI values, a value for the *NAME* property, and values for any other properties defined as required.
  - You can omit a property that does not have a value to be added from the *INSTANCE* tag. However, if an omitted property is a required property with a *CHAR*, *VARCHAR*, or *LONG VARCHAR* data type, a not-applicable symbol is generated and stored in the DataGuide information catalog. If an omitted required property has a *TIMESTAMP* data type, then DataGuide generates and stores the value 9999-12-31-24.00.00.000000.
- When merging an object:
  - You must specify all UUI values, to ensure that matching objects can be identified.
  - You can omit a property that does not have a value to be added or updated. However, if the defined object does not exist, and the omitted property is required, then a not-applicable symbol is generated and stored in the DataGuide information catalog.

### **ACTION.OBJINST(DELETE) or ACTION.OBJINST(DELETE\_TREE\_ALL) or ACTION.OBJINST(DELETE\_TREE\_REL)**

Deleting an object

```
:INSTANCE.SOURCEKEY(UUI_short_name (UUI_property_value) . . . )
```

## INSTANCE

### Context

---

```
:ACTION.OBJINST(DELETE)
:OBJECT.TYPE( )
:INSTANCE.SOURCEKEY(UUI_short_name( ) . . .)
```

---

Figure 43. Using the *INSTANCE* tag when deleting objects

---

```
:ACTION.OBJINST(DELETE_TREE_ALL)
:OBJECT.TYPE( )
:INSTANCE.SOURCEKEY(UUI_short_name( ) . . .)
```

---

Figure 44. Using the *INSTANCE* tag when deleting Grouping category objects and contained objects

---

```
:ACTION.OBJINST(DELETE_TREE_REL)
:OBJECT.TYPE( )
:INSTANCE.SOURCEKEY(UUI_short_name( ) . . .)
```

---

Figure 45. Using the *INSTANCE* tag when deleting Grouping category objects and relationships

### Keywords

#### SOURCEKEY

Specifies the UUI property values that identify a particular object.

*SOURCEKEY* must be the first keyword of the *INSTANCE* tag.

#### *UUI\_short\_name*

Identifies a UUI property name by its 8-character short name. Specify all of the *UUI\_short\_name(UUI\_property\_value)* combinations. The *UUI\_short\_name* is not case sensitive; you can specify this value using uppercase or lowercase characters.

#### *UUI\_property\_value*

Specifies the value of a UUI property for a particular object. This value is case sensitive.

**Rules:** You must specify one *UUI\_short\_name(value)* combination for each property defined as a UUI property for the object type. Each object type has one or more properties defined as UUI properties. These properties are used by DataGuide to uniquely identify an object in the information catalog.

## INSTANCE

### **ACTION.OBJINST(UPDATE)**

Updating property values for an object

```
:INSTANCE.SOURCEKEY(UII_short_name (UII_property_value) . . . )  
    short_name (property_value) . . .
```

#### **Context**

---

```
:ACTION.OBJINST(UPDATE)  
:OBJECT.TYPE()  
:INSTANCE.SOURCEKEY(UII_short_name(. . .) short_name())
```

---

Figure 46. Using the *INSTANCE* tag when updating objects

#### **Keywords**

##### **SOURCEKEY**

Specifies the UII property values that identify a particular object.

*SOURCEKEY* must be the first keyword of the *INSTANCE* tag.

##### *UII\_short\_name*

Identifies a UII property by its 8-character short name. The *UII\_short\_name* is not case sensitive; you can specify this value using uppercase or lowercase characters.

##### *UII\_property\_value*

This value is case sensitive. With *UII\_short\_name*, specifies the value of a UII property for a particular object.

##### *short\_name*

Identifies the property to be updated by its 8-character short name. The *short\_name* is not case sensitive; you can specify this value using uppercase or lowercase characters.

You cannot specify the following property short names because you cannot update these properties: *OBJTYPID*, *INSTIDNT*, *UPDATIME*, *UPDATEBY*.

##### *property\_value*

With *short\_name*, specifies the new value of the property for the given object. This value is case sensitive.

**Rules:** You must specify one *UII\_short\_name(value)* combination for each property defined as a UII property for the object type. Each object type has one or more properties defined as UII properties. These properties are used by DataGuide to uniquely identify an object in the information catalog.

If you specify a property value, that value is updated in the DataGuide information catalog. If you do not specify a property value, the value is not updated.



## INSTANCE

### **ACTION.RELATION(ADD) or ACTION.RELATION(DELETE)**

Adding or deleting relationships

```
:INSTANCE.SOURCEKEY(UII_short_name (UII_property_value)...)
      TARGETKEY(UII_short_name (UII_property_value)...)

```

#### **Context**

---

```
:ACTION.RELATION(ADD)
:RELTYPE.TYPE() SOURCEKEY() TARGETKEY()
:INSTANCE.SOURCEKEY(UII_short_name()...) TARGETKEY(UII_short_name()...)

```

---

Figure 47. Using the INSTANCE tag when adding relationships

---

```
:ACTION.RELATION(DELETE)
:RELTYPE.TYPE() SOURCEKEY() TARGETKEY()
:INSTANCE.SOURCEKEY(UII_short_name()...) TARGETKEY(UII_short_name()...)

```

---

Figure 48. Using the INSTANCE tag when deleting relationships

#### **Keywords**

##### **SOURCEKEY**

Specifies the UII property values that identify the first object in a relationship.

<b>When the relationship is:</b>	<b>The SOURCEKEY identifies:</b>
Contains	The Grouping category object
Contact	The object the contact is for
Attachment	The object the comment is for
Link	Either object to be linked

SOURCEKEY must be the first keyword of the INSTANCE tag.

##### **TARGETKEY**

Specifies the UII property values that identify the second object in a relationship.

<b>When the relationship is:</b>	<b>The TARGETKEY identifies:</b>
Contains	The Elemental category object
Contact	The Contact category object
Attachment	The Attachment category object
Link	Either object to be linked

TARGETKEY must be the second keyword of the INSTANCE tag.

##### *UII\_short\_name*

Identifies a UII property name by its 8-character short name. This value is not case sensitive; you can specify this value using uppercase or lowercase characters.

## OBJECT

### *UUI\_property\_value*

Specifies the value of a UUI property for a particular object. This value is case sensitive.

**Rules:** For each object, you must specify one *UUI\_short\_name(value)* combination for each property defined as a UUI property for the object type. Each object type has one or more properties defined as UUI properties. These properties are used by DataGuide to uniquely identify an object in the information catalog.

You must separate each *UUI\_short\_name(value)* and *short\_name(value)* pair with a blank, as shown in Figure 49.

---

```
:INSTANCE.SOURCEKEY(UUIname1(value1) UUIname2(value2)) sname3(value3) sname4(value4)
```

---

Figure 49. Example of an *INSTANCE* tag with several short names

Leading blanks included between the parentheses for a value become part of the value; trailing blanks are removed. DataGuide counts these blanks as part of the data length when determining whether the length of the value is valid. Including extra leading or trailing blanks on a value that make the entire value longer than the maximum length for a value causes an error.

---

## NL

Specifies a new line within a property value.

DataGuide only reads NL tags specified within non-UUI property values and ignores all others.

## Syntax

```
:NL.
```

## Rules

Use NL tags only within the specification of *property\_values* in *INSTANCE* tags.

---

## OBJECT

Defines the attributes for an object type or identifies an object type.

## Context

This tag is required immediately following:

```
ACTION.OBJTYPE  
ACTION.OBJINST
```

## Syntax

```
:OBJECT.TYPE(type) CATEGORY(category) EXTNAME(ext_name)
      PHYNAME(table_name) ICOFILE(OS/2_ICON_file_name)
      ICWFILE(Windows_ICON_file_name)
```

Different OBJECT tag keywords are required or valid depending on the type of ACTION tag the OBJECT tag follows.

**ACTION.OBJTYPE(ADD) or ACTION.OBJTYPE(MERGE)**

Adding or merging object types

**Context**


---

```
:ACTION.OBJTYPE(ADD)
:OBJECT.TYPE() CATEGORY() EXTNAME() PHYNAME() ICOFILE() ICWFILE()
:PROPERTY.EXTNAME() DT() DL() SHRTNAME() NULLS() UUISEQ()
```

---

Figure 50. Using the OBJECT tag when adding object types

---

```
:ACTION.OBJTYPE(MERGE)
:OBJECT.TYPE() CATEGORY() EXTNAME() PHYNAME() ICOFILE() ICWFILE()
:PROPERTY.EXTNAME() DT() DL() SHRTNAME() NULLS() UUISEQ()
```

---

Figure 51. Using the OBJECT tag when merging object types

**Keywords****TYPE**

Specifies the name of an object type.

Required keyword.

**type**

Defines and identifies the short name for a specific object type.

The value of *type* must be unique to an object type across all related DataGuide information catalogs containing the same object type, so that objects of this object type can be shared among these information catalogs. If the value of *type* already exists, it is used as a search argument.

The maximum length for the value is 8 characters. The value is stored in uppercase characters. This value can start with the characters A - Z, @, #, or \$, and can contain any of these characters plus 0 - 9 and \_. No leading or embedded blanks are allowed.

After you create the object type, you cannot change the value of *type*.

## OBJECT

### CATEGORY

Specifies which DataGuide category this object type belongs to.

Required keyword.

#### *category*

Specifies a DataGuide object category. This value can be one of the following:

GROUPING  
ELEMENTAL  
SUPPORT  
CONTACT  
DICTIONARY

You cannot specify PROGRAM or ATTACHMENT as the category for a new object type.

You cannot modify the information on this keyword after the object type is defined.

### EXTNAME

Specifies a longer, descriptive name for the object type. Required keyword.

#### *ext\_name*

Specifies an extended, descriptive name for the object type. The maximum length for *ext\_name* is 80 characters.

This name must be unique within related DataGuide information catalogs.

The value of *ext\_name* is stored in mixed case.

You can modify the information on this keyword after the object type is defined.

### PHYNAME

Specifies the name used when creating the database table containing information about this object type.

Optional keyword.

#### *table\_name*

Specifies the name used when creating the database table containing object type information.

The maximum length of the name is defined when DataGuide is installed.

*table\_name* must be unique within the DataGuide information catalog and cannot contain any SQL reserved words.

By default, *table\_name* is the *type* specified for the **TYPE** keyword. This value is not case sensitive; you can specify this value using uppercase or lowercase characters.

This value can start with the characters A - Z, @, #, or \$, and can contain any of these characters, plus 0 - 9 and \_. No leading or embedded blanks are allowed. This value cannot be any of the SQL reserved words for the database used for the DataGuide information catalog.

After the table is created, you cannot change its name.

## OBJECT

### ICOFIELD

Specifies the file containing the OS/2 icon associated with the object type.

Optional keyword.

#### *OS/2\_ICON\_file\_name*

Specifies the name of the OS/2 ICON file to be associated with the object type. The maximum length of *OS/2\_ICON\_file\_name* is 254 characters. However, this name, combined with the icon path (ICOPATH), can have a maximum length of 259, so the true allowable maximum length depends on the length of the icon path. This file can have any extension. This value is not case sensitive; you can specify this value using uppercase or lowercase characters.

You cannot specify the drive and path information that identifies where the ICON file resides using this keyword; you must specify this information as an input parameter for the FLGImport API call (see *Programming Guide and Reference*), the import function on the user interface (see "Importing tag language files" on page 71), or the IMPORT option of the DGUIDE command (see "Invoking DataGuide from the command line" on page 152).

You can modify this value after the object type is created using ACTION.OBJTYPE(UPDATE). However, once you specify an icon file to be associated with an object type, you can change the associated icon, but you cannot redefine the object type to have no associated icon at all.

### ICWFIELD

Specifies the file containing the Windows icon associated with the object type.

Optional keyword.

#### *Windows\_ICON\_file\_name*

Specifies the name of the Windows ICON file to be associated with the object type. The maximum length of *Windows\_ICON\_file\_name* is 254 characters. However, this name, combined with the icon path (ICOPATH), can have a maximum length of 259, so the true allowable maximum length depends on the length of the icon path. This file can have any extension. This value is not case sensitive; you can specify this value using uppercase or lowercase characters.

You cannot specify the drive and path information that identifies where the ICON file resides using this keyword; you must specify this information as an input parameter for the FLGImport API call (see *Programming Guide and Reference*), the import function on the user interface (see "Importing tag language files" on page 71), or the IMPORT option of the DGUIDE command (see "Invoking DataGuide from the command line" on page 152).

You can modify this value after the object type is created using ACTION.OBJTYPE(UPDATE). However, once you specify an icon file to be associated with an object type, you can change the associated icon, but you cannot redefine the object type to have no associated icon at all.

## ACTION.OBJTYPE(APPEND)

### *Context*

## OBJECT

---

```
:ACTION.OBJTYPE ( APPEND )  
:OBJECT .TYPE ( )  
:PROPERTY .EXTNAME ( ) DT ( ) DL ( ) SHRTNAME ( ) NULLS ( ) UISEQ ( )
```

---

Figure 52. Using the *OBJECT* tag when adding properties to object types

### Keywords

#### TYPE

Specifies the name (*type*) of an object type.

Required keyword.

#### *type*

Identifies a specific object type by its 8-character short name.

### **ACTION.OBJTYPE(DELETE) or ACTION.OBJTYPE(DELETE\_EXT)**

Deleting an existing object type.

### Context

---

```
:ACTION.OBJTYPE ( DELETE )  
:OBJECT .TYPE ( )
```

---

Figure 53. Using the *OBJECT* tag when deleting object types

---

```
:ACTION.OBJTYPE ( DELETE_EXT )  
:OBJECT .TYPE ( )
```

---

Figure 54. Using the *OBJECT* tag when deleting object types and all objects of that type

### Keywords

#### TYPE

Specifies the name (*type*) of an object type.

Required keyword.

#### *type*

Identifies a specific object type by its 8-character short name.

### **ACTION.OBJTYPE(UPDATE)**

Updating object type information.

### Context

## OBJECT

---

```
:ACTION.OBJTYPE(UPDATE)
:OBJECT.TYPE() EXTNAME() ICOFILE() ICWFILE()
```

---

Figure 55. Using the OBJECT tag when updating object types

### Keywords

#### TYPE

Specifies the name (*type*) of an object type.

Required keyword.

#### *type*

Identifies a specific object type by its 8-character short name. You cannot update this value.

#### EXTNAME

Specifies a descriptive name for the object type. Optional keyword.

#### *ext\_name*

Specifies an extended, descriptive name for the object type. The maximum length for *ext\_name* is 80 characters.

You can update this value.

This name must be unique within related DataGuide information catalogs.

The value of *ext\_name* is stored in mixed case.

#### ICOFILE

Specifies the file containing the OS/2 icon associated with the object type.

Optional keyword.

#### *OS/2\_ICON\_file\_name*

Specifies the name of the OS/2 ICON file to be associated with the object type.

You can update this value.

The maximum length of *OS/2\_ICON\_file\_name* is 254 characters. You cannot specify the drive and path information that identifies where the ICON file resides using this keyword; you must specify this information as an input parameter for the FLGImport API call, the import function on the user interface, or the IMPORT option of the DGUIDE command.

#### ICWFILE

Specifies the file containing the Windows icon associated with the object type.

Optional keyword.

#### *Windows\_ICON\_file\_name*

Specifies the name of the Windows ICON file to be associated with the object type.

You can update this value.

## OBJECT

The maximum length of *Windows\_ICON\_file\_name* is 254 characters. You cannot specify the drive and path information that identifies where the ICON file resides using this keyword; you must specify this information as an input parameter for the FLGImport API call, the import function on the user interface, or the IMPORT option of the DGUIDE command.

### **ACTION.OBJINST**

Adding, updating, deleting, or merging objects

#### **Context**

---

```
:ACTION.OBJINST(ADD)
:OBJECT.TYPE()
:INSTANCE.short_name()
```

---

*Figure 56. Using the OBJECT tag when adding objects*

---

```
:ACTION.OBJINST(MERGE)
:OBJECT.TYPE()
:INSTANCE.short_name()
```

---

*Figure 57. Using the OBJECT tag when merging objects*

---

```
:ACTION.OBJINST(UPDATE)
:OBJECT.TYPE()
:INSTANCE.SOURCEKEY(UUI_short_name()...) short_name()
```

---

*Figure 58. Using the OBJECT tag when updating objects*

---

```
:ACTION.OBJINST(DELETE)
:OBJECT.TYPE()
:INSTANCE.SOURCEKEY(UUI_short_name()...)
```

---

*Figure 59. Using the OBJECT tag when deleting objects*

#### **Keywords**

##### **TYPE**

Specifies the name (*type*) of an object type.

Required keyword.

##### *type*

Identifies a specific object type by its 8-character short name.



## PROPERTY

---

### PROPERTY

Defines a property that belongs to an object type.

This tag is required following these ACTION tags:

```
:ACTION.OBJTYPE(ADD)
:ACTION.OBJTYPE(MERGE)
:ACTION.OBJTYPE(APPEND)
```

### Syntax

```
:PROPERTY.EXTNAME(ext_name) DT(data_type) DL(data_length)
    SHRTNAME(short_name) NULLS(Y | N) UUISEQ(UUI_number)
```

### Context

---

```
:ACTION.OBJTYPE(ADD)
:OBJECT.TYPE() CATEGORY() EXTNAME() PHYNAME() ICOFILE() ICWFILE()
:PROPERTY.EXTNAME() DT() DL() SHRTNAME() NULLS() UUISEQ()
```

---

*Figure 60. Using the PROPERTY tag when adding object types*

---

```
:ACTION.OBJTYPE(MERGE)
:OBJECT.TYPE() CATEGORY() EXTNAME() PHYNAME() ICOFILE() ICWFILE()
:PROPERTY.EXTNAME() DT() DL() SHRTNAME() NULLS() UUISEQ()
```

---

*Figure 61. Using the PROPERTY tag when merging object types*

---

```
:ACTION.OBJTYPE(APPEND)
:OBJECT.TYPE()
:PROPERTY.EXTNAME() DT() DL() SHRTNAME() NULLS() UUISEQ()
```

---

*Figure 62. Using the PROPERTY tag when adding properties to object types*

### Keywords

#### EXTNAME

Specifies a descriptive name for the property.

Required keyword.

## PROPERTY

### *ext\_name*

Specifies an extended descriptive name.

The maximum length of *ext\_name* is 80 characters. The *ext\_name* must be unique within the object type. *ext\_name* is stored in mixed case.

### **DT**

Specifies the data type for the property.

Required keyword.

### *data\_type*

The data type for the property. You can specify this value in either uppercase or lowercase. Valid values are:

- C** Character
- V** Variable character
- L** Long variable character
- T** Timestamp

### **DL**

Specifies the data length or maximum data length for the property.

Required property.

### *data\_length*

The data length or maximum data length for the property. Valid values for *data\_length* depend on the *data\_type* defined for this property:

<b>data_type</b>	<b>Maximum value for data_length</b>
C (character)	Maximum length is 254
V (variable character)	Maximum length is 4000
L (long variable character)	Maximum length is 32700
T (timestamp)	Always 26 characters

### **SHRTNAME**

Specifies the property short name.

Required keyword.

### *short\_name*

The short name for the property. *short\_name* can be up to 8 characters long. This value can contain only SBCS characters.

This value is stored as uppercase characters; any lowercase characters are translated into uppercase.

This value can start with the characters A - Z, @, #, or \$, and can contain any of these characters, plus 0 - 9 and \_. No leading or embedded blanks are allowed.

This value cannot be any of the SQL reserved words for the database used for the DataGuide information catalog. Do not specify the property short names of any of the following required properties for every DataGuide object type: OBJTYPID, INSTIDNT, UPDATIME, or UPDATEBY.

## PROPERTY

### NULLS

Specifies whether a value for the property is required for every object. This value can be specified in uppercase or lowercase.

Required keyword.

**Y** indicates that this value can be null. When appending a new property using the ACTION.OBJTYPE(APPEND) tag, you must specify NULLS(Y), because appended properties must be optional.

**N** indicates that a value for this property is required. If no data is provided for a required property when an object is added to the DataGuide information catalog, DataGuide enters a not-applicable symbol for the required value if it has a data type of CHAR, VARCHAR, or LONG VARCHAR. For a required value with a data type of TIMESTAMP, DataGuide enters the following value:  
9999-12-31-24.00.00.000000

### UUISEQ

Identifies the properties used in the UUI.

Optional keyword; the default value is 0. The UUISEQ keyword is optional for properties that are not part of the UUI. The UUI is a set of properties defined by the administrator as the key that uniquely identifies each object.

### *UUI\_number*

Specifies the position of the property in the UUI sequence. Valid values are 0, 1, 2, 3, 4, and 5. The value 0 means the property is not part of the UUI. A nonzero value for *UUI\_number* indicates that the property is part of the UUI.

All object types defined in the tag language file must have at least one property that is part of the UUI. The UUI can consist of up to 5 properties.

At least one property must be defined as part of the UUI.

When assigning *UUI\_number* values to more than one property, the numbers of the UUI properties must range from 1 to the number of properties in the UUI. For example, if three properties are defined as part of the UUI, the *UUI\_number* values must be 1, 2, and 3. You cannot skip numbers in the sequence. The *UUI\_number* values do not need to be in the same order that the properties are specified.

## Rules

- You can define the DataGuide reserved property NAME as part of the UUI when you add a new object type or merge object types. Figure 63 shows the general syntax for identifying NAME as a UUI property.

---

```
:ACTION.OBJTYPE(ADD)
:OBJECT.TYPE() CATEGORY() EXTNAME() PHYNAME() ICOFILE() ICWFILE()
:PROPERTY.SHRTNAME(NAME) UUISEQ()
```

---

Figure 63. Example of specifying the NAME property as part of the UUI

## RELTYPE

Empty parentheses in this figure denote values that must be provided when used in a tag language file.

- The maximum length of the UUI fields is 254 bytes.

---

## RELTYPE

Identifies the type of relationship that is being added or deleted and the object types of the objects involved in the relationship.

This tag is required immediately following these tags:

```
:ACTION.RELATION(ADD)
:ACTION.RELATION(DELETE)
```

## Syntax

```
:RELTYPE.TYPE(Contain | CONTACT | ATTACHMENT | LINK)
           SOURCETYPE(source_type) TARGETTYPE(target_type)
```

## Context

---

```
:ACTION.RELATION(ADD)
:RELTYPE.TYPE() SOURCETYPE() TARGETTYPE()
:INSTANCE.SOURCEKEY(UUI_short_name()...) TARGETKEY(UUI_short_name()...)
```

*Figure 64. Using the RELTYPE tag when adding relationships*

---

```
:ACTION.RELATION(DELETE)
:RELTYPE.TYPE() SOURCETYPE() TARGETTYPE()
:INSTANCE.SOURCEKEY(UUI_short_name()...) TARGETKEY(UUI_short_name()...)
```

*Figure 65. Using the RELTYPE tag when deleting relationships*

## Keywords

### TYPE

Specifies the type of relationship.

Required keyword.

Valid values are:

### ATTACHMENT

Attachment relationship: target object is attached to the source object.

### CONTACT

Contact relationship: source object is associated with the target Contact object.

## TAB

### CONTAIN

Contains relationship: source object contains the target object.

### LINK

Link relationship: source object is linked with the target object.

### SOURCETYPE

Identifies the source object type.

Required keyword.

#### *source\_type*

The source object type name *source\_type* corresponds to the *type* value for the TYPE keyword of the OBJECT tag. The maximum length for *source\_type* is 8 characters. This value is not case sensitive; you can specify this value using uppercase or lowercase characters.

For an Attachment relationship, *source\_type* is a non-Attachment object type name.

For a Contains relationship, *source\_type* is the container object type name.

For a Contact or link relationship *source\_type* is the Grouping or Elemental object type name.

### TARGETTYPE

Identifies the target object type.

Required keyword.

#### *target\_type*

The target object type name. *target\_type* corresponds to the *type* value for the TYPE keyword on the OBJECT tag. The maximum length for *target\_type* is 8 characters. This value is not case sensitive; you can specify this value using uppercase or lowercase characters.

For an Attachment relationship, *target\_type* is the Attachment object type name.

For a Contains relationship, *target\_type* is the containee's object type name.

For a Contact relationship, *target\_type* is the Contact object type name.

For a link relationship, *target\_type* is a Grouping or Elemental object type name.

---

## TAB

Specifies a tab within a property value.

DataGuide only reads TAB tags specified within non-UII property values and ignores all others.

## Syntax

:TAB.

## TAB

### Rules

Use TAB tags only within the specification of *property\_values* in INSTANCE tags.

---

## Appendix D. What a tag language file should look like

You can use the tags to tell DataGuide to add, delete, and update object types and objects. DataGuide tags are contextual; you specify tags in different combinations depending on what you want to do.

---

### Start your tag language file with DISKCNTL

Start the tag language file with a DISKCNTL tag if the file is on a removable disk, such as a diskette. For example:

```
:DISKCNTL.SEQUENCE( 1, +)
```

If the tag language file is on more than one diskette, then DISKCNTL must be the first tag in each section of the tag language file on each diskette. If the tag language file is on a fixed disk, then DISKCNTL is ignored.

---

### Define your additions, changes, and deletions

You use the tag language to define both actions and what you are acting on.

#### Defining what you want to do

The ACTION tag tells DataGuide what you want to do. The keyword tells DataGuide what kind of information you want to maintain. The option tells DataGuide what task you want to perform.

**:ACTION.OBJINST**(*option*)

Maintaining objects.

**:ACTION.OBJTYPE**(*option*)

Maintaining object types.

**:ACTION.RELTYPE**(*option*)

Maintaining object relationships.

#### Defining the information

After you have told DataGuide what you want to do, you need to define precisely what information you are adding, changing, or deleting.

To define:	Use these tags:
Existing object type	OBJECT
Object type to be merged	OBJECT and PROPERTY
New object type	OBJECT and PROPERTY
New properties for an object type	OBJECT and PROPERTY
New or existing object	OBJECT and INSTANCE
New or existing object relationship	RELTYPE and INSTANCE

## Putting it all together

The keywords and values required for OBJECT, INSTANCE, and PROPERTY tags are different depending on what they are identifying to add, change, or delete. The sequence of tags within each ACTION tag is:

### **:ACTION.OBJINST**(*option*)

```
:ACTION.OBJINST(ADD)
:OBJECT.TYPE()
:INSTANCE.short_name() ...

:ACTION.OBJINST(DELETE)
:OBJECT.TYPE()
:INSTANCE.SOURCEKEY(UII_short_name()...)

:ACTION.OBJINST(DELETE_TREE_ALL)
:OBJECT.TYPE()
:INSTANCE.SOURCEKEY(UII_short_name()...)

:ACTION.OBJINST(DELETE_TREE_REL)
:OBJECT.TYPE()
:INSTANCE.SOURCEKEY(UII_short_name()...)

:ACTION.OBJINST(MERGE)
:OBJECT.TYPE()
:INSTANCE.short_name() ...

:ACTION.OBJINST(UPDATE)
:OBJECT.TYPE()
:INSTANCE.SOURCEKEY(UII_short_name()...) short_name()
```

### **:ACTION.OBJTYPE**(*option*)

```
:ACTION.OBJTYPE(ADD)
:OBJECT.TYPE() CATEGORY() EXTNAME() PHYNAME() ICOFILE()
:PROPERTY.EXTNAME() DT() DL() SHRTNAME() NULLS() UIISEQ()

:ACTION.OBJTYPE(APPEND)
:OBJECT.TYPE()
:PROPERTY.EXTNAME() DT() DL() SHRTNAME() NULLS() UIISEQ()

:ACTION.OBJTYPE(DELETE)
:OBJECT.TYPE()

:ACTION.OBJTYPE(DELETE_EXT)
:OBJECT.TYPE()

:ACTION.OBJTYPE(MERGE)
:OBJECT.TYPE() CATEGORY() EXTNAME() PHYNAME() ICOFILE() ICWFILE()
:PROPERTY.EXTNAME() DT() DL() SHRTNAME() NULLS() UIISEQ()

:ACTION.OBJTYPE(UPDATE)
:OBJECT.TYPE() EXTNAME() ICOFILE() ICWFILE()
```

### **:ACTION.RELATION**(*option*)

```
:ACTION.RELATION(ADD)
:RELTYPETYPE(CONTAIN | CONTACT | ATTACHMENT | LINK) SOURCETYPE(type) TARGETTYPE(type)
:INSTANCE.SOURCEKEY(UII_short_name()...) TARGETKEY(UII_short_name()...)
```



```
:ACTION.RELATION(DELETE)
:RELTYPE.TYPE(CONTAIN | CONTACT | ATTACHMENT | LINK) SOURCETYPE(type) TARGETTYPE(type)
:INSTANCE.SOURCEKEY(UUI_short_name()...) TARGETKEY(UUI_short_name()...)
```

For specific information about the format of the INSTANCE, OBJECT, and PROPERTY tags, see “INSTANCE” on page 131, “OBJECT” on page 136, or “PROPERTY” on page 143.

---

## Committing changes to the database

The COMMIT tag tells DataGuide to commit changes to the DataGuide database. When DataGuide processes a COMMIT tag, it empties the echo file before it starts processing the next set of tags, so that the echo file only contains tags describing uncommitted changes.

If DataGuide encounters an error, it rolls back the database to the last committed checkpoint. Insert COMMIT tags in your file to keep your data consistent, and to limit the number of changes that are canceled when the database is rolled back.

You can insert a COMMIT tag after any complete set of tags that define an action. Do not insert a COMMIT tag between the ACTION tag and the last tag defining the data associated with the ACTION tag.

```
:COMMIT.CHKPT(2 )
```

---

## Putting comments in the tag language file

You can use the COMMENT tag to put information in the tag language file, such as notes and labels, that you do not want to import into your DataGuide information catalog.

```
:COMMENT.Updating the LASTDATE property
```

---

## Appendix E. Performing DataGuide functions from the command line

You can perform some DataGuide functions from an MS-DOS command prompt.

To:	See:
Open a DataGuide information catalog	"Invoking DataGuide from the command line"
Import a tag language file into your DataGuide information catalog	"Invoking DataGuide from the command line"
Import MDIS metadata into your DataGuide information catalog	"Importing MDIS-conforming tag language files" on page 85
Export MDIS metadata from your DataGuide information catalog	"Importing MDIS-conforming tag language files" on page 85
Create an information catalog	"Creating an information catalog from the command line" on page 154

---

### Invoking DataGuide from the command line

To open a DataGuide information catalog from an MS-DOS command prompt, enter the DGUIDE command. Keep in mind the following rules for the command syntax:

- None of the parts, except where specified, are case sensitive.
- Each keyword must be preceded by either a slash (/) or hyphen (-) character.
- All keywords that follow DGUIDE on the same line are required. All keywords that follow /IMPORT on the same line are required if you choose to use /IMPORT.
- Underlined choices are defaults.

```
DGUIDE /USERID userid /PASSWORD password /DGNAME dname
```

Optional keywords:

```
/ADMIN  
/TRACE_1|2|3|4  
/IMPORT filename /LOGFILE filename /RESTART B|C
```

Optional import keyword:

```
/ICOPATH iconpath
```

For example, to open the sample DataGuide information catalog as an administrator, type:

```
DGUIDE /USERID longods /PASSWORD secret /DGNAME DGV5SAMP /ADMIN
```

**/ADMIN**

Specifies that you are logging on as an administrator. If you don't specify this optional keyword for the DGUIDE command, you are logged on as a user and you cannot perform administrator tasks.

**/DGNAME**

Your information catalog name.

If the information catalog is local, give the database name. If the information catalog is remote, give the alias under which it was cataloged.

Example:

```
/DGNAME DGV5SAMP
```

**/ICOPATH**

Valid only with /IMPORT; optional.

Indicates that you are importing icons and specifies the icon path that the import function will use. DataGuide assumes that the path is the same as the one where you installed DataGuide unless you specify a full drive and path. You must specify a fixed drive.

Example:

```
/ICOPATH d:\icons\
```

**/IMPORT**

Imports the tag language file you specify. Unless you specify the full drive, path, and file name, DataGuide assumes that the file is in the path specified on the DGWPATH environment variable.

Example:

```
/IMPORT d:\tagfile.tag
```

This keyword bypasses the DataGuide user interface and performs the import function as a batch command.

**/LOGFILE**

Valid only with /IMPORT; required with /IMPORT.

Specifies the file destination for messages DataGuide generates during import. Unless you specify a full drive, path, and file name, DataGuide places the file in the path specified on the DGWPATH environment variable. You must specify a fixed drive.

Example:

```
/LOGFILE d:\tagfile.log
```

**/PASSWORD**

Your password for this user ID.

Example:

```
/PASSWORD secret
```

Passwords for DB2 for AIX, DB2 PE, DB2 UDB EEE, DB2 UDB for Windows NT, and DB2 for Windows 95 databases are case-sensitive; you must type them exactly as specified.

#### **/RESTART**

Valid only with /IMPORT; required with /IMPORT.

Indicates which option the import function uses. The valid options are:

- B** Imports the tag language file from the beginning.
- C** The default. Imports the tag language file from the last point at which DataGuide successfully committed changes to the information catalog.

#### **/TRACE**

The level of trace information to be sent to the DataGuide trace file. Each higher level includes the functions of the levels below it (3 includes the functions of levels 0, 1, 2, and 3). You might have to specify a higher level if you call IBM Software Support to diagnose DataGuide problems.

- 0** The default. Includes all messages and warning, error, and severe error conditions.
- 1** Includes entry and exit records of the highest level DataGuide functions.
- 2** Includes extremely granular entry and exit records of the DataGuide functions.
- 3** Includes input and output parameters (excluding input or output structure).
- 4** Includes all input or output structures that are passed to and used by DataGuide.

#### **/USERID**

Your DataGuide information catalog user ID. Depending on the database location of the information catalog you are opening, type the user ID required by the database. For example, the user ID might be your local, LAN, OS/400, AIX, or MVS TSO user ID.

Example:

```
/USERID longods
```

---

## **Creating an information catalog from the command line**

To create a DataGuide information catalog from an MS-DOS command prompt, enter the CREATEIC command. Keep in mind the following rules for the command syntax:

- None of the parts, except where specified, are case sensitive.
- Each keyword must be preceded by either a slash (/) or hyphen (-) character.
- All keywords that follow CREATEIC on the same line are required.
- If your information catalog will be stored in a DB2 for MVS database, you must include the required MVS keywords.
- Underlined choices are defaults.

```
CREATEIC /DBTYPE dbtype /DGNAME dname /USERID userid /PASSWORD password /KAl primary_admin
```

**Optional keywords:**

```
/NAS -|symbol  
/KA2 backup_admin  
/MVSDB dbname /TSTORGP table_stor /XSTORGP index_stor
```

**Optional MVS keyword:**

```
/MVSUPPER |M
```

For example, to create an information catalog on a remote DB2 PE database, type:

```
CREATEIC /DBTYPE DB26 PE /DGNAME DGV5SAMP /USERID longods /PASSWORD secret /KAl longods
```

**/DBTYPE**

Specifies the type of DB2 database in which you want to store your information catalog. Valid choices are:

- DB22** Specifies a DB2 for OS/2 database.
- DB2** Specifies a DB2 for MVS or DB2 for OS/390 database. You must have either the Distributed Database Connection Services (DDCS) or DB2 Connect product installed on your workstations to use DB2 for MVS or DB2 for OS/390.
- DB2400** Specifies a DB2 for OS/400 database. You must have either the DDCS or DB2 Connect product installed on your workstations to use DB2 for OS/400.
- DB26000** Specifies a DB2 for AIX database. You must have TCP/IP installed on your workstations to use DB2 for AIX.
- DB26000PE** Specifies a DB2 PE or DB2 UDB EEE database.
- DB2NT** Specifies a DB2 UDB for Windows NT database. You must have TCP/IP or NetBIOS installed on your workstations to use a remote DB2 UDB for Windows NT database.
- DB295** Specifies a DB2 for Windows 95 database. You must have DB2 for Windows 95 Version 2.1.2 single-user installed on your workstation. DB2 for Windows 95 information catalogs are standalone on your workstation and cannot be accessed from other workstations.

**/DGNAME**

Specifies the database name. If the database is local, specify the database name. If the database is remote, specify the alias name for the remote database that is cataloged on your local workstation.

**/USERID**

Specifies the user ID required by the database that stores your information catalog:

**DB2 for OS/2 (local)**

Local user ID, specified with UPM on your workstation

**DB2 for OS/2 (remote)**

LAN user ID, specified with UPM on the remote workstation

**DB2 for MVS**

RACF user ID

**DB2 for OS/400**

OS/400 user ID

**DB2 for AIX**

AIX user ID

**DB2 PE or DB2 UDB EEE**

AIX user ID

**DB2 UDB for Windows NT (local)**

Windows NT user ID

**DB2 UDB for Windows NT (remote)**

LAN user ID, specified with User Manager on the remote workstation

**DB2 for Windows 95**

Windows 95 user ID

**/PASSWORD**

Specifies the password for the user ID that you entered on the /USERID keyword.

Passwords for DB2 for AIX, DB2 PE, DB2 UDB EEE, DB2 UDB for Windows NT, and DB2 for Windows 95 databases are case-sensitive; you must type them exactly as specified.

**/NAS**

Specifies the character you want to use to indicate property values that are not applicable. You can choose from the following special characters:

!	;	#	\$	%	*	(
)	+	,	-	.	/	:
{	}	=	?	@	[	]
-						

The default is a hyphen (-).

**/KA1**

Specifies the user ID of the person who will be the primary administrator of DataGuide. This user ID must have SYSADM (or ALLOBJ authority if your information catalog is stored in a DB2 for OS/400 database) authority.

**/KA2**

Specifies the user ID of the person who will back up the primary administrator. This user ID must have database administrator authority.

**/MVSDB**

Valid only with /DBTYPE DB2; required with /DBTYPE DB2.

Specifies the name of the DB2 for MVS database.

**/TSTORGP**

Valid only with /DBTYPE DB2; required with /DBTYPE DB2.

Specifies the name of the storage group that you will use for tables.

**/XSTORGP**

Valid only with /DBTYPE DB2; required with /DBTYPE DB2.

Specifies the name of the storage group that you will use for indexes.

**/MVSUPPER**

Valid only with /DBTYPE DB2; optional with /DBTYPE DB2.

Indicates whether you want to save the property values of each object in uppercase.

- Y** The default. Specifies that the values are stored in the MVS database in uppercase, but you can enter the values in lowercase when you search for them using DataGuide.
- N** Specifies that the values are stored in the MVS database exactly as you enter them and that all DataGuide searches are case sensitive.

---

## Glossary

### A

**administrator.** A person responsible for managing the content and use of DataGuide.

**anchor.** A Grouping object that contains other objects, but is not contained by another Grouping object.

**application program interface (API).** See *DataGuide application programming interface*.

**Attachment.** The category for object types used to attach additional information to another DataGuide object. For example, you can attach comments to an object.

### B

**browse.** To display DataGuide objects that are grouped by subject. Contrast with *search*.

### C

**catalog.** See *information catalog* and *database catalog*.

**category.** A classification for DataGuide object types. The category designates the:

- Actions available to object types
- Relationships allowed between object types in the same or different categories.

Object types belong to one of the following categories:

- Attachment
- Contact
- Dictionary
- Elemental
- Grouping
- Program
- Support

**CeIDial sample data.** A sample information catalog (DGV5SAMP) available when you install DataGuide that can be used for installation verification. This sample information catalog is also used in the exercises in *Using DataGuide*.

**collection.** A container for objects. A collection can be used to gather objects of interest for easy access.

**Comments.** A classification for objects that annotate another object in DataGuide. For example, you may want to attach a Comments object to a chart object that contains notes about the data in the chart.

The Comments object type is shipped with DataGuide. You cannot add properties to it.

**commit.** To make changes to the DataGuide database permanent. Contrast with *roll back*.

**contact.** A reference for more information about an object. Further information might include the person who created the information that the object represents, or the department responsible for maintaining the information.

**Contact.** A category for the Contact object type and other object types that identify contacts.

**Contact object type.** A classification for objects that identify contacts.

### D

**database catalog.** A collection of tables that contains descriptions of database objects such as tables, views, and indexes.

**DataGuide application program interface (API).** The portion of DataGuide that processes application program requests for DataGuide services and functions.

**DataGuide Catalog window.** The main DataGuide window. It contains all the DataGuide objects available.

**DataGuide database.** The set of relational tables containing the metadata managed by DataGuide. See *information catalog*.

**DBCS.** Double-byte character set.

**decision-support system.** A system of applications that help users make decisions. This kind of system allows users to work with information presented in meaningful ways; for example, spreadsheets, charts, and reports.

**delete history.** A log of delete activity, the capture of which is turned on and off by the DataGuide administrator. The log can be transferred to a tag language file.



## derived data information catalog

**derived data.** Data that is copied or enhanced (perhaps by summarizing the data) from operational data sources into an informational database.

**descriptive data.** Data that identifies and describes an object, for example, the name of a table, the location of a spreadsheet, or the creator of a document. Also called metadata.

**Description view.** A view that lists the properties and property values for an object.

**Dictionary.** The category for object types that can be used to define terminology (for example, the "Glossary entries" object type in the sample information catalog).

**dictionary facility.** A collection of definitions or synonyms for the business terms you use in the information catalog. After it is created, the dictionary facility appears in every user's Catalog window as a saved search icon.

**double-byte character set (DBCS).** A set of characters in which each character is represented by two bytes. Languages such as Japanese, Chinese, and Korean, which contain more symbols than can be represented by 256 code points, require double-byte character sets. Contrast with *single-byte character set*.

**DP NAME.** An identification for an object type that uniquely identifies it for import operations. Also called the short name of an object type.

## E

**echo file.** A file produced by DataGuide when it imports a tag language file. This file contains all the tags that have been processed since either the beginning of the tag language file or the point when the last COMMIT tag was processed.

**Elemental.** The category for non-Grouping object types that are the building blocks for other DataGuide object types. Elemental object types are at the bottom of object type hierarchies. "Columns in relational tables," "Presentations {electronic and hardcopy}," and "Graphics and Images" are all examples of Elemental object types.

**export.** To copy metadata from DataGuide, translate the metadata into tag language, and put this output in a tag language file for a subsequent import operation.

**external name.** The 80-byte name for an object type. Also called object type name.

**extract control file.** A file that contains statements that control the operation of an extractor utility program.

**extract program.** A utility program that copies from a metadata source, such as an RDBMS catalog, translates the metadata into tag language, and places this output in a tag language file.

## F

**FAT.** File allocation table. A table used to allocate space on a disk for a file and to locate the file.

**FLGID.** See *object identifier*.

## G

**Grouping.** The category for object types that can contain other object types. Examples of Grouping object types available in the sample information catalog shipped with DataGuide are: "Tables or views in a relational database," which contains the Elemental object type "Columns in relational tables"; and "Multi-dimensional model," which contains another Grouping object type "Dimension."

## H

**HPFS.** High-performance file system. In OS/2, an installable file system that uses high-speed buffer storage, known as a cache, to provide fast access to large disk volumes. File names used with the HPFS can have as many as 254 characters.

## I

**icon.** A graphical representation of an object, object type, collection, new search, saved search, or subject.

**import.** To apply the contents of a tag language file to a DataGuide information catalog to initially populate the information catalog, change the information catalog contents, or copy the contents of another DataGuide information catalog to the information catalog.

**information catalog.** The database managed by DataGuide containing descriptive data that helps users identify and locate the data and information available to them in the organization. The information catalog is a component of the Information Warehouse framework.

## information resource object type

**information resource.** In Visual Warehouse, the definition of a resource from which a business view extracts data or to which a business view writes data. In general, information resources correspond to operational data stores. However, an information resource can correspond to a file to which a business view writes data for temporary storage.

**information source.** An item of data or information, such as a table or chart, that is represented by a DataGuide object.

**informational application.** A program or system that lets users retrieve and analyze their data.

**informational database.** A database that contains derived data and is intended for business decision making.

**input structure.** A self-defining data structure used to submit data to the DataGuide application program interface.

**instance.** See *object*.

**instance identifier.** A 10-digit numeric identifier generated by DataGuide for each object. The identifier is unique for that object within a given object type (an object of another object type may have the same identifier), and within a given DataGuide database (an object in another DataGuide database may have the same identifier).

**I/O structure.** See *input structure* and *output structure*.

## K

**keyword.** An element of the DataGuide tag language that identifies the meaning of a data value imported into or exported out of a DataGuide information catalog.

**keyword search.** See *search*.

## L

**link.** A connection between two or more objects involved in a linked relationship.

**linked relationship.** A relationship between objects in an information catalog. Objects in a linked relationship are peers, rather than one an underlying object of the other.

For example, in the sample information catalog shipped with DataGuide, the object called **CeIDial Sales Information** is linked with various objects describing CeIDial advertisements for the year.

**log file.** A file produced by DataGuide when it imports a tag language file or exports objects in the DataGuide information catalog. This file records the times and dates when the import or export started and stopped and any error information for the process.

## M

**metadata.** Data about information sources. See *descriptive data*.

**multiple character wildcard.** A character used to represent any series of characters of any length. By default, the multiple character wildcard is an asterisk (\*). See also *wildcard* and *single character wildcard*.

## N

**New Search icon.** An icon in the DataGuide Catalog window that is used to begin a search.

**not-applicable symbol.** A character that indicates that a value for a required property was not provided when an object was created. The not-applicable symbol is a hyphen (-) by default, but you could have identified a different symbol when you created the information catalog.

## O

**object.** An item that represents a unit or distinct grouping of information. Each DataGuide object identifies and describes information, but does not contain the actual information. For example, an object can provide the name of a report, list its creation date, and describe its purpose.

**object identifier.** A 16-digit identifier for an object that is made up of its 6-digit object type identifier and its 10-digit instance identifier that is used with some API calls. See *object type identifier* and *instance identifier*.

**object type.** A classification for objects. An object type is used to reflect a type of business information, such as a table, report, or image.

## object type identifier search criteria

DataGuide provides a set of sample object types, which you can modify. You can also create additional object types to meet the needs of your organization.

**object type identifier.** A 6-digit numeric identifier generated by DataGuide for each object type. The identifier is unique within the DataGuide database.

**object type registration.** With the DataGuide application program interface, the basic information about an object type that you must define in the DataGuide information catalog before you can define the properties for the object type. This information includes the category, the name, the icon, and the name of the table containing the object information.

**operational data.** Data used to run the day-to-day operations of an organization.

**option.** In DataGuide tag language, a parameter of the ACTION tag that defines the action to be performed on objects or object types in the DataGuide database when the tag language file is imported.

**output structure.** A self-defining data structure produced by DataGuide when returning data produced by a DataGuide API call.

## P

**physical type name.** The name of the table in the DataGuide database that contains metadata for instances of a specific object type.

**populate.** To add object types, objects, or metadata to the DataGuide information catalog.

**Program category.** The category for the Programs object type.

**Programs object type.** A classification for objects that identify and describe applications capable of processing the actual information described by DataGuide objects. The Programs object type is shipped with DataGuide.

**property.** A characteristic or attribute that describes a unit of information. Each object type has a set of associated properties. For example, the "Graphics and Images" object type in the sample DataGuide information catalog includes the following properties:

- Name
- Description
- Image type

Image filename

For each object, a set of values are assigned to the properties.

**property name.** The 80-byte descriptive name of a property that is displayed in the DataGuide user interface. Contrast with *property short name*.

**property short name.** An 8-character name used by DataGuide to uniquely identify a property of an object or object type.

**property value.** The value of a property.

**PT NAME.** See *physical type name*.

## R

**RDBMS.** Relational database management system.

**RDBMS catalog.** A set of tables that contain descriptions of SQL objects, such as tables, views, and indexes, maintained by an RDBMS.

**relational database management system.** A software system, such as DB2 for OS/2, that manages and stores relational data.

**registration.** See *object type registration*.

**roll back.** To remove uncommitted changes to the DataGuide database. Contrast with *commit*.

## S

**saved search.** A set of search criteria that is saved for subsequent use. Appears as an icon in the Catalog window.

**SBCS.** Single-byte character set.

**search.** To request the display of DataGuide objects that meet specific criteria.

**search by subject.** See *browse*.

**search by term.** See *search*.

**search criteria.** Options and character strings used to specify how to perform a search. This can include object type names, property values, whether the search is for an exact match, and whether the search is case sensitive.

## single-byte character set (SBCS) work area

**single-byte character set (SBCS).** A character set in which each character is represented by a one-byte code. Contrast with *double-byte character set*.

**single character wildcard.** A character used to represent any single character. By default, the single character wildcard is a question mark (?). See also *wildcard* and *multiple character wildcard*.

**subject search.** See *browse*.

**Subjects icon.** An icon in the DataGuide Catalog window that when selected displays the Subjects window.

**Support.** The category for object types that provide additional information about your information catalog or enterprise (for example, the “DataGuide News” object type in the sample information catalog).

**support facility.** A collection of information you consider helpful for users of your information catalog, such as announcements of changes or updates to the information catalog. After it is created, the support facility appears in every user’s Catalog window as a saved search icon.

## T

**tag.** An element of the tag language. Tags indicate actions to be taken when the tag language file is imported to DataGuide.

**tag language.** A format for defining object types and objects, and actions to be taken on those object types and objects, in a DataGuide information catalog.

**tag language file.** A file containing DataGuide tag language that describes objects and object types to be added, updated, or deleted in the DataGuide information catalog when the file is imported. You can also use the tag language file for batch imports of objects into DataGuide.

A tag language file is produced by:

- Exporting objects from an information catalog

- Transferring a delete history log
- Extracting descriptive data from another database system using an extract program

**Tree view.** A view that displays hierarchically an object and the objects it contains.

## U

**unit of work.** A recoverable sequence of operations within an application process. A unit of work is the basic building block a database management system uses to ensure that a database is in a consistent state. A unit of work is ended when changes to the database are committed or rolled back.

**universal unique identifier (UUI).** A key for an object. The key is comprised of up to five properties, which, when concatenated in a designated order, uniquely identify the object during import and export functions.

**user.** A person who accesses the information available in DataGuide information catalogs but who is not an administrator.

Some DataGuide users, if they have been granted authority, can perform some object management tasks normally performed by DataGuide administrators.

## V

**View menu.** A menu used to change the way objects are displayed in a window.

## W

**wildcard.** A special character that is used as a variable when specifying property values in a search. See also *single character wildcard* and *multiple character wildcard*.

**work area.** See *DataGuide Catalog window*.

---

## Notices

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Subject to IBM's valid intellectual property or other legally protectable rights, any functionally equivalent product, program, or service may be used instead of the IBM product, program, or service. The evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, are the responsibility of the user.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing to:

IBM Director of Licensing  
IBM Corporation  
North Castle Drive  
Armonk, NY 10594-1785  
U.S.A.

Licensees of this program who wish to have information about it for the purpose of enabling: (1) the exchange of information between independently created programs and other programs (including this one) and (2) the mutual use of the information which has been exchanged, should contact:

IBM Corporation  
J74/G4  
555 Bailey Avenue  
P.O. Box 49023  
San Jose, CA 95161-9023  
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including, in some cases, payment of a fee.

---

## Trademarks

The following terms are trademarks of the IBM Corporation in the United States or other countries or both:

AIX	DB2 Universal Database
BookManager	IBM
DATABASE 2	IMS
DataGuide	MVS
DataJoiner	OS/2
DataPropagator	OS/390
DB2	OS/400
DB2 Connect	RACF
DB2 OLAP Server	WebExplorer

1-2-3 and Lotus are trademarks of Lotus Development Corporation in the United States and/or other countries.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries.



Microsoft, Windows, Windows NT , and the Windows logo are registered trademarks of Microsoft Corporation.

Other company, product, and service names, may be trademarks or service marks of others.

---

## Bibliography

To get copies of the books listed here, or to get more information about a particular library, see your IBM representative.

You can view the DATABASE 2 or Universal Database publications from the DB2 Web site (<http://www.software.ibm.com/data/db2/>) or order them for a fee.

### **DATABASE 2 publications:**

*DATABASE 2 Administration Guide for common servers* (S20H-4580)

*DATABASE 2 Messages Reference for common servers* (S20H-4808)

*DATABASE 2 Problem Determination Guide for common servers* (S20H-4779)

*DATABASE 2 (for MVS) Version 3 Messages and Codes* (SC26-4892)

*DATABASE 2 (for MVS) Version 4 Messages and Codes* (SC26-3268)

*DB2/400 SQL Programming Version 3* (SC21-3611)

### **DB2 Universal Database publications**

*IBM DB2 Universal Database Administration Guide* (S10J-8157)

*IBM DB2 Universal Database Messages Reference* (S10J-8168)

*IBM DB2 Universal Database Troubleshooting Guide* (S10J-8169)

### **Visual Warehouse information:**

*Planning and Installing Visual Warehouse and DataGuide* (SC26-3496)

*Managing Visual Warehouse* (GC26-8822)

*Programming Guide and Reference* (SC26-3368)

This book is available softcopy on the CD-ROM. You can also order this publication from IBM for an additional fee.

*Using DataGuide* This book is available online in the **Visual Warehouse** folder.

*Visual Warehouse and DataGuide Messages & Reason Codes* This book is available online in the **DataGuide** folder.

Visual Warehouse and DataGuide online help

### **Web Computing information**

*A Comprehensive Guide to Virtual Private Networks, Volume I: IBM Firewall, Server and Client Solutions*, SG24-5201

---

## Index

### A

access information catalog, users can't 92  
ACTION tag  
    OBJINST keyword 118, 132  
    OBJTYPE keyword 123  
    planning for extract program 65  
    RELATION keyword 127  
    sequence 150  
    tag language reference 118, 128  
    tips 149  
ADD option  
    ACTION.OBJINST 119  
    ACTION.OBJTYPE 124, 137  
    ACTION.RELATION 127  
administrator  
    resetting with CLEARKA command 94  
ALTERKA command, changing DataGuide  
    administrators with 18  
APPEND option 124  
Application data sample object type 107  
ASCII text 65  
Attachment category  
    Comments object type defined 112  
    definition of 22  
    relationships  
        modifying 54  
        summary of 22  
ATTACHMENT keyword 146  
Audio clips sample object type 110

### B

backing up DataGuide database 92  
backup administrator, identifying 18  
blanks removed from variable values 117  
Business subject areas sample object type 107

### C

C (CHAR) 30  
category  
    Attachment  
        Comments object types defined 112  
        copying comments 52  
        creating comments 51  
        definition of 22

category (*continued*)

Attachment (*continued*)

    deleting comments 53  
    relationships with other categories 22  
    updating comments 52

Contact

    definition of 22  
    object type in sample information catalog 111  
    People to contact object type in sample  
        information catalog 111  
    relationships with other categories 22

definition of 21

Dictionary

    creating dictionary facility 62  
    definition of 22  
    Glossary entries object type in sample information  
        catalog 111  
    object type in sample information catalog 111  
    relationships with other categories 22

Elemental

    Audio clips object type in sample information  
        catalog 110  
    Charts object type in sample information  
        catalog 110  
    definition of 21  
    Documents object type in sample information  
        catalog 110  
    Images or graphics object type in sample  
        information catalog 110  
    Internet documents object type in sample  
        information catalog 110  
    Lotus Approach queries object type in sample  
        information catalog 110  
    object types in sample information catalog 109  
    Presentations object type in sample information  
        catalog 111  
    relationships with other categories 22  
    Spreadsheets object type in sample information  
        catalog 111  
    Text-based reports object type in sample  
        information catalog 111  
    Video clips object type in sample information  
        catalog 111

Grouping

    Application data object type in sample information  
        catalog 107  
    Business subject areas object type in sample  
        information catalog 107



category (*continued*)

- Grouping (*continued*)
  - Columns or fields object type in sample information catalog 107
  - Databases object type in sample information catalog 107
  - definition of 21
  - Dimensions within a multi-dimensional database object type in sample information catalog 107
  - Elements object type in sample information catalog 108
  - Files object type in sample information catalog 108
  - IMS database definitions (DBD) object type in sample information catalog 108
  - IMS program control blocks (PCB) object type in sample information catalog 108
  - IMS program specification blocks (PSB) object type in sample information catalog 108
  - IMS segments object type in sample information catalog 108
  - Members within a multi-dimensional database object type in sample information catalog 108
  - Multi-dimensional databases object type in sample information catalog 109
  - object types in sample information catalog 106
  - Records object type in sample information catalog 109
  - Relational tables and views object type in sample information catalog 109
  - relationships with other categories 22
  - Subschemas object type in sample information catalog 109
  - Transformations object type in sample information catalog 109
- Program
  - definition of 22
  - relationships with other categories 22
- Program, Programs object type defined 112
- Support
  - creating support facility 63
  - DataGuide news object type in sample information catalog 112
  - definition of 22
  - object types in sample information catalog 112
  - Online news services object type in sample information catalog 112
  - Online publications object type in sample information catalog 112
  - relationships with other categories 22
- CATEGORY keyword 30, 137
- CelDial business scenario
  - object type property specifications 112
  - predefined object type descriptions 106
  - sample information catalog provided with DataGuide, creating 100
- CHAR data type for object type property 27
- character data type
  - on PROPERTY tag 143
  - optional property 35
  - property of DL 30
- Charts sample object type 110
- checkpoint tags 69
- checkpoint, commit 76
- checkpt\_id identifier 129
- CHKPID keyword 129
- CLEARKA command for resetting logged-on administrator user ID 94
- code
  - extended, finding what it means 76
  - reason, finding what it means 76
- Columns or fields sample object type 107
- command
  - ALTERKA, for changing DataGuide administrators 18
  - CLEARKA, for resetting logged-on administrator user ID 94
  - DGUIDE, for opening an information catalog 152
  - DGWDEMO, for creating sample DB2 UDB for Windows NT information catalog 100
  - IMPORT 30, 34
- comment
  - attaching to objects 54
  - copying for an object 52
  - creating for an object 51
  - deleting from an object 53
  - detaching from objects 54
  - updating for an object 52
- comment status list
  - setting values for users 19
  - shown in Create Comment window 19
- COMMENT tag
  - planning for extract programs 65
  - tag language reference 128
  - when to use 151
- Comments object type 112
- commit call 69
- commit checkpoint 76, 129
- COMMIT tag
  - planning for extract program 65

- COMMIT tag (*continued*)
  - tag language reference 129, 130
  - when to use 151
- committing to database 69
- concurrent access 92
- Contact category
  - definition of 22
  - object type
    - People to contact, provided in sample information catalog 111
    - sample information catalog, provided in 111
  - relationships
    - Attachment, adding 54
    - Attachment, removing 54
    - modifying 50
    - Program, adding 58
    - Program, removing 62
    - summary of 22
- CONTACT keyword 146
- CONTAIN keyword 146
- Contains relationship, modifying 46
- customized extract programs 65

## D

- data
  - corrupt, what to do 93
  - inconsistent, what to do 93
  - recovering 94
- data type
  - C (CHAR) 30, 35, 143
  - L (LONG VARCHAR) 30, 35, 143
  - T (TIMESTAMP) 30, 35, 143
  - V (VARCHAR) 30, 35, 143
  - valid 117
- database
  - alias name 13
  - backup 93
  - local, registering 13
  - remote, registering 13
  - rollback 151
  - supported by DataGuide 3
- DATABASE 2 for AIX information catalog, defining 8
- DATABASE 2 for OS/2 2
  - Directory Tool 13
  - information catalog, defining 3
  - log files 91
- DATABASE 2 for OS/390 information catalog, defining 5
- DATABASE 2 for OS/400 information catalog, defining 7
- DATABASE 2 for Windows 95 information catalog, defining 10
- DATABASE 2 for Windows NT information catalog, defining 10
- DATABASE 2 Parallel Edition for AIX information catalog 2
- Databases sample object type 107
- DataGuide
  - opening information catalog from command line 152
  - opening information catalog from user interface 16
- DataGuide administrator
  - changing in Manage DataGuide Users window 18
  - changing with ALTERKA command 18
- DataGuide news sample object type 112
- DB2 Connect 3
- DB2 for AIX information catalog, defining 8
- DB2 for OS/2 2
  - Directory Tool 13
  - information catalog, defining 3
  - log files 91
- DB2 for OS/390 information catalog, defining 5
- DB2 for OS/400 information catalog, defining 7
- DB2 for Windows 95 information catalog, defining 10
- DB2 OLAP Server/Essbase
  - metadata
    - identifying for exchange 78
    - scheduling for exchange 80
- DB2 PE information catalog 2
- DB2 UDB EEE information catalog 2
- DB2 UDB for Windows NT information catalog, defining 10
- DB2 Universal Database Enterprise-Extended Edition information catalog 2
- DBCS 116
- delete history
  - creating 70
  - importing 71
- DELETE option
  - ACTION.OBJINST 119
  - ACTION.OBJTYPE 125
  - ACTION.RELATION 128
  - OBJINST keyword 132
- DELETE\_EXT option on ACTION.OBJTYPE 125
- DELETE\_TREE\_ALL option
  - ACTION.OBJINST 120
  - OBJINST keyword 132
- DELETE\_TREE\_REL option
  - ACTION.OBJINST 121

DELETE\_TREE\_REL option (*continued*)  
 OBJINST keyword 132  
 descriptive data, extracting 64  
 desktop applications, extract programs for 99  
 DGMDISC command, for converting DataGuide tag language to MDIS metadata  
 , specifying 85  
 database user ID, specifying 85  
 input DataGuide tag language file, specifying 85  
 output MDIS file, specifying 85  
 password, specifying 85  
 syntax of 85  
 DGUIDE command for invoking DataGuide 85, 87  
 ADMIN keyword 86, 88, 153  
 DGNAME keyword, for specifying information catalog 86, 88, 153  
 IMPORT keywords  
 ICOPATH 153  
 LOGFILE 153  
 RESTART 154  
 LOGFILE, MDIS\_IMPORT keyword 86, 88  
 MDIS\_EXPORT keywords  
 LOGFILE 86, 88  
 OBJECT 89  
 OBJTYPE 89  
 PASSWORD keyword 86, 89, 153  
 TRACE keyword 87, 89, 154  
 USERID keyword 87, 90, 154  
 DGV5SAMP sample information catalog  
 object types defined in 106  
 DGV5SAMP sample information catalog, creating 100  
 Dictionary category  
 creating dictionary facility 62  
 definition of 22  
 object type  
 Glossary entries, provided in sample information catalog 111  
 sample information catalog, provided in 111  
 relationships  
 Attachment, adding 54  
 Attachment, removing 54  
 Program, adding 58  
 Program, removing 62  
 summary of 22  
 dictionary facility, creating 62  
 Dimensions within a multi-dimensional database sample  
 object type 107  
 disk space, monitoring 91  
 DISKCNTRL tag  
 extract program 65

DISKCNTRL tag (*continued*)  
 tag language reference 130  
 tips 149  
 DL keyword  
 defining optional properties 35  
 defining properties 30  
 tag language reference 143  
 Documents sample object type 110  
 double-byte character set (DBCS) 116  
 DT keyword  
 defining optional properties 35  
 defining properties 30  
 tag language reference 143

## E

echo (ECH) file 74  
 definition of 74  
 example of 74  
 problem diagnosis 95  
 reading 74  
 restarting 151  
 solving import problems with 74  
 Elemental category  
 definition of 21  
 object types  
 Audio clips, provided in sample information catalog 110  
 Charts, provided in sample information catalog 110  
 Documents, provided in sample information catalog 110  
 Images or graphics, provided in sample information catalog 110  
 Internet documents, provided in sample information catalog 110  
 Lotus Approach queries, provided in sample information catalog 110  
 Presentations, provided in sample information catalog 111  
 sample information catalog, provided in 109  
 Spreadsheets, provided in sample information catalog 111  
 Text-based reports, provided in sample information catalog 111  
 Video clips, provided in sample information catalog 111  
 relationships  
 Attachment, adding 54  
 Attachment, removing 54  
 Contact, adding 50

## Elemental category (*continued*)

### relationships (*continued*)

- Contact, removing 50
- Contains, adding 46
- Contains, removing 46
- linked, adding 48
- linked, removing 48
- Program, adding 58
- Program, removing 62
- summary of 22

## Elements sample object type 108

### EUI vii

See also user interface (EUI), performing DataGuide tasks with

### examples

- combining two information catalogs 68
- echo file 74
- log file 75
- messages with reason codes, using 76
- Programs object, required for
  - Class 57
  - Identifier 57
  - Name 57
  - Qualifier 1, 2, 3 57
- properties 66
- tag language file 31
- trace file 96
- UUI 67

### export

- options
  - default in settings notebook 72
  - specifying during export 72

### exporting

- icon files from information catalog 72
- metadata from information catalog 72
- solving problems from 75

## extended code, finding what it means 76

### external name

- changing 34
- for object type, rules for specifying 24
- for property, rules for specifying 26
- of object type 30
- of object type property 31

### external name property 25

### EXTNAME keyword

- creating object types 30, 31
- on OBJECT tag 137, 141
- on PROPERTY tag 143
- optional property 35
- updating object type 34

## extract program

- creating new objects 66
  - customized 65, 69
  - desktop applications 99
  - input and output 65
  - installing files 65
  - steps for running documented in README files 99
- ## extracting descriptive data 64

## F

### file

- echo 74
    - See also echo (ECH) file
  - insufficient space for, what to do 91
  - log 75
    - See also log (LOG) file
  - lost, what to do 92
  - trace 95
    - See also trace (TRC) file
- ## Files sample object type 108
- filling your information catalog 37
  - formatting tag language files 66

## G

## Glossary entries sample object type 111

### Grouping category

- definition of 21
- object types
  - Application data, provided in sample information catalog 107
  - Business subject areas, provided in sample information catalog 107
  - Columns or fields, provided in sample information catalog 107
  - Databases, provided in sample information catalog 107
  - Dimensions within a multi-dimensional database, provided in sample information catalog 107
  - Elements, provided in sample information catalog 108
  - Files, provided in sample information catalog 108
  - IMS database definitions (DBD), provided in sample information catalog 108
  - IMS program control blocks (PCB), provided in sample information catalog 108
  - IMS program specification blocks (PSB), provided in sample information catalog 108
  - IMS segments, provided in sample information catalog 108

## Grouping category (*continued*)

### object types (*continued*)

- Members within a multi-dimensional database, provided in sample information catalog 108
- Multi-dimensional databases, provided in sample information catalog 109
- Records, provided in sample information catalog 109
- Relational tables and views, provided in sample information catalog 109
- sample information catalog, provided in 106
- Subschemas, provided in sample information catalog 109
- Transformations, provided in sample information catalog 109

### relationships

- Attachment, adding 54
- Attachment, removing 54
- Contact, adding 50
- Contact, removing 50
- Contains, adding 46
- Contains, removing 46
- linked, adding 48
- linked, removing 48
- Program, adding 58
- Program, removing 62
- summary of 22

grouping objects by subject 44

## H

- HANDLES keyword 59
- hide system generated properties 25
- history, delete 71
  - See also* delete history
- home page, URL for DataGuide 98

## I

### ICOFIELD keyword

- creating object types 30
- tag language reference 137, 141
- updating object type 34

### icon representing object type

- changing 33
- changing for OS/2 32
- changing for Windows 32
- exporting 72
- identifying 25

### ICWFILE keyword

- creating object types 30
- tag language reference 137, 141
- updating object type 34

Images or graphics sample object type 110

IMPORT command 30, 34

### importing

- delete history 71
- restarting from checkpoint 76
- solving problems from 74, 75
- tag language files 71

IMS database definitions (DBD) sample object type 108

IMS program control blocks (PCB) sample object type 108

IMS program specification blocks (PSB) sample object type 108

IMS segments sample object type 108

### information catalog

- combining with another information catalog
  - merging object types before 68
  - with tag language files 64

corrupt data in, what to do 93

DB2 for AIX, defining 8

DB2 for OS/2, defining 3

DB2 for OS/390, defining 5

DB2 for OS/400, defining 7

DB2 for Windows 95, defining 10

DB2 UDB for Windows NT, defining 10

defining 3

establishing object types in 22

### exporting

- icon files for objects 72

- metadata from 72

exporting MDIS metadata from 85, 87

extract programs for populating 98

filling 64

### importing

- delete history tag language files 70, 71

- restarting from checkpoint 76

- tag language files 71

importing MDIS metadata into 85, 87

inconsistent data in, what to do 93

local, registering 13

maximum allowed object types 23

migrating 12

### objects

- copying in 39

- creating in, using DataGuide tag language 38

- creating in, using DataGuide windows 37

- deleting from, using DataGuide tag language 42

- deleting from, using DataGuide windows 42

information catalog (*continued*)

- objects (*continued*)
    - updating in, using DataGuide tag language 41
    - updating in, using DataGuide windows 40
  - opening from command line 152
  - opening from user interface 16
  - planning 21
  - populating with objects 37
  - refreshing 64
  - remote, registering 13
  - sample provided with DataGuide
    - creating 100
    - object types defined in 106
    - predefined program objects 113
  - setting up 1, 20
  - users can't access, what to do 92
- input to customized extract program 65
- installing extract programs 65
- instance identifier property 25
- INSTANCE tag
  - planning for extract program 65
  - tag language reference 131, 136
  - with ACTION.OBJINST(ADD) 131
  - with ACTION.OBJINST(DELETE\_TREE\_ALL) 132
  - with ACTION.OBJINST(DELETE\_TREE\_REL) 132
  - with ACTION.OBJINST(DELETE) 132
  - with ACTION.OBJINST(MERGE) 131
  - with ACTION.OBJINST(UPDATE) 134
  - with ACTION.RELATION(ADD) 135
  - with ACTION.RELATION(DELETE) 135
- INSTIDNT property 25
- Internet documents sample object type 110
- invocation parameters
  - opening the Invocation Parameters window to specify 55
  - recommended value for programs 58

## K

keyword

- ATTACHMENT 146
- CATEGORY 30, 137
- CHKPID 129
- CONTACT 146
- CONTAIN 146
- context sensitive 150
- DL
  - defining optional properties 35
  - defining properties 30
  - tag language reference 143

keyword (*continued*)

- DT
  - defining optional properties 35
  - defining properties 30
  - tag language reference 143
- EXTNAME 35
  - creating object types 30, 31
  - on OBJECT tag 137, 141
  - on PROPERTY tag 143
  - updating object type 34
- HANDLES 59
- ICOFIELD 30
  - optional keyword on OBJECT 137, 141
  - updating object type 34
- ICWFIELD
  - creating object types 30
  - optional keyword on OBJECT 137, 141
  - updating object type 34
- LINK 146
- NAME 59
- NULLS 31
- OBJTYPE 123
- PARMLIST 59
- PHYNAME 29, 137
- RELATION 127
- RELTYPE 146, 147
- SEQUENCE 130
- SHRTDESC 59
- SHRTNAME
  - as a property 39
  - creating object types 30
  - optional property 35
- SOURCEKEY 133
  - ACTION.OBJINST(DELETE) 134
  - ACTION.RELATION 135
  - associating contacts 50
  - deleting object 62
  - using parentheses 46
- SOURCETYPE
  - associating contacts 50
  - defining grouping 46
  - defining linked relationship 48
  - RELTYPE 146
- STARTCMD 59
- TARGETKEY
  - ACTION.RELATION 135
  - associating contacts 50
  - using parentheses 46
- TARGETTYPE
  - associating contacts 50
  - defining grouping 46

keyword (*continued*)

TARGETYPE (*continued*)

defining linked relationship 48

RELTYPE 146

TYPE

creating object types 29

creating objects 39

creating optional property 35

deleting object types 36

deleting objects 42

OBJTYPE(ADD) 137

OBJTYPE(APPEND) 140

OBJTYPE(DELETE) 140, 142

OBJTYPE(MERGE) 137

OBJTYPE(UPDATE) 141, 142

RELTYPE 146

updating object type 34

unsupported for national languages 116

UUI\_short\_name 46, 48, 50

UUICLASS 59

UUIIDENT 59

UUIQUAL1, 2, 3 59

UUISEQ 30, 143

## L

L (LONG VARCHAR) 30

last changed by 25

last changed date and time property 25

LINK keyword 146

linked relationship, creating 48

list, comment status 19

*See also* comment status list

log (LOG) file

DataGuide 91

DB2 for OS/2 91

definition of 74

example of 75

location of 75

reading 75

solving export problems with 75

solving import problems with 75

logging on to DataGuide

from command line 152

from user interface 16

LONG VARCHAR data type for object type property 27

long variable character data type

on PROPERTY tag 143

optional property 35

property of DL 30

Lotus Approach queries sample object type 110

## M

maximum

LONG VARCHAR properties for object type 27

object types for an information catalog 23

properties for an object type 25

recommended length of UUI 28

MDIS

described 83

predefined object types that map to 100

Columns or fields 107

Databases 107

Dimensions within a multi-dimensional

database 107

Elements 108

Files 108

IMS database definitions (DBD) 108

IMS program control blocks (PCB) 108

IMS program specification blocks (PSB) 108

IMS segments 108

Members within a multi-dimensional

database 108

Multi-dimensional databases 109

Records 109

Relational tables and views 109

Subschemas 109

Transformations 109

URL for Web site 83

MDISDGC command, for converting MDIS metadata to

DataGuide tag language

database user ID, specifying 84

input MDIS file, specifying 84

log file, specifying 84

output DataGuide tag language file, specifying 84

password, specifying 84

syntax of 84

Members within a multi-dimensional database sample

object type 108

MERGE option

ACTION.OBJINST 122

ACTION.OBJTYPE 125, 137

messages, using 94

metadata

DataGuide, exchanging with other information

catalogs 64

exchanging with other products 77

DB2 OLAP Server/Essbase 77

establishing environment for 77

identifying DB2 OLAP objects to exchange 78

- metadata (*continued*)
  - exchanging with other products (*continued*)
    - identifying Visual Warehouse metadata to exchange 80
    - non-IBM, MDIS-conforming products 83
    - scheduling in Visual Warehouse 82
    - Visual Warehouse 77
  - exporting from DataGuide 72
  - extracting from other products 64
  - MDIS 83
    - tag language files, converting from 84
    - tag language files, converting to 85
  - synchronizing with DB2 OLAP Server 78
  - synchronizing with Visual Warehouse 80
- Metadata Interchange Specification 83
  - See also* MDIS
- migrating an information catalog 12
- Multi-dimensional databases sample object type 109

## N

- name
  - external
    - of object type 30
    - of object type property 31
  - short
    - of object type 29
    - of object type property 30
- NAME keyword 59
- NAME property 25
- names
  - reserved words that cannot be used for 115
  - rules for specifying
    - object type (external) name 24
    - property name 26
    - short, for object type 24
    - short, for property 27
- national language support (NLS)
  - unsupported tags and keywords 116
- NetBIOS
  - node 13
- NL tag
  - planning for extract programs 65
  - tag language reference 136
- not-applicable symbol
  - default value of 28
  - for DB2 for AIX information catalog, selecting 9
  - for DB2 for MVS information catalog, selecting 6
  - for DB2 for OS/2 information catalog, selecting 4
  - for DB2 for OS/400 information catalog, selecting 7

- not-applicable symbol (*continued*)
  - for DB2 for Windows 95 information catalog, selecting 11
  - for DB2 UDB for Windows NT information catalog, selecting 11
  - identified when creating information catalog 28
  - specifying for use during MDIS export 89
- NULLS keyword 31, 143

## O

- object
  - adding contacts
    - steps for, using DataGuide tag language 50
    - steps for, using DataGuide windows 49
  - adding to another information catalog 64
  - attaching comments to 54
  - copying, steps for 39
  - creating 66
    - steps for, using DataGuide tag language 38
    - steps for, using DataGuide windows 37
  - definition of 37
  - deleting
    - steps for, using DataGuide tag language 42
    - steps for, using DataGuide windows 42
  - detaching comments from 54
  - exchanging with object from another information catalog 64
  - grouping with other objects
    - preparation for 44
    - steps for, using DataGuide tag language 46
    - steps for, using DataGuide windows 44
  - linking with other objects
    - steps for, using DataGuide tag language 48
    - steps for, using DataGuide windows 47
  - properties 66
  - relationship
    - Attachment 54
    - Contact 50
    - Contains 46
    - link 48
  - removing contacts
    - steps for, using DataGuide tag language 50
    - steps for, using DataGuide windows 49
  - updating
    - steps for, using DataGuide tag language 41
    - steps for, using DataGuide windows 40
- OBJECT tag
  - planning for extract program 65
  - tag language reference 136, 142



OBJECT tag (*continued*)

- with ACTION.OBJTYPE(ADD) 137
- with ACTION.OBJTYPE(APPEND) 139
- with ACTION.OBJTYPE(DELETE\_EXT) 140
- with ACTION.OBJTYPE(DELETE) 140
- with ACTION.OBJTYPE(MERGE) 137
- with ACTION.OBJTYPE(UPDATE) 140

object type

- associating programs with
  - preparation for 55
  - steps for, using DataGuide tag language 58
  - steps for, using DataGuide windows 55

Attachment category

- Comments object type defined 112

category

- definition of 21

Comments

- copying 52
- creating 51
- deleting 53
- updating 52

Contact category

- People to contact, provided in sample information catalog 111
- sample information catalog, provided in 111

copying program association for 60

creating

- preparation for 23
- steps for, using DataGuide windows 23
- steps for, using tag language 29

definition of 21

deleting

- steps for, using DataGuide windows 36
- steps for, using tag language 36

Dictionary category

- creating dictionary facility with 62
- Glossary entries, provided in sample information catalog 111
- sample information catalog, provided in 111

disassociating programs with

- steps for, using DataGuide tag language 62
- steps for, using DataGuide windows 62

Elemental category

- Audio clips, provided in sample information catalog 110
- Charts, provided in sample information catalog 110
- Documents, provided in sample information catalog 110
- Images or graphics, provided in sample information catalog 110

object type (*continued*)

Elemental category (*continued*)

- Internet documents, provided in sample information catalog 110

- Lotus Approach queries, provided in sample information catalog 110

- Presentations, provided in sample information catalog 111

- sample information catalog, provided in 109

- Spreadsheets, provided in sample information catalog 111

- Text-based reports, provided in sample information catalog 111

- Video clips, provided in sample information catalog 111

establishing in information catalog 22

exporting 72

Grouping category

- Application data, provided in sample information catalog 107

- Business subject areas, provided in sample information catalog 107

- Columns or fields, provided in sample information catalog 107

- Databases, provided in sample information catalog 107

- Dimensions within a multi-dimensional database, provided in sample information catalog 107

- Elements, provided in sample information catalog 108

- Files, provided in sample information catalog 108

- IMS database definitions (DBD), provided in sample information catalog 108

- IMS program control blocks (PCB), provided in sample information catalog 108

- IMS program specification blocks (PSB), provided in sample information catalog 108

- IMS segments, provided in sample information catalog 108

- Members within a multi-dimensional database, provided in sample information catalog 108

- Multi-dimensional databases, provided in sample information catalog 109

- Records, provided in sample information catalog 109

- Relational tables and views, provided in sample information catalog 109

- sample information catalog, provided in 106

- Subschemas, provided in sample information catalog 109

- object type (*continued*)
  - Grouping category (*continued*)
    - Transformations, provided in sample information catalog 109
  - icon representing
    - changing 33
    - changing for OS/2 32
    - changing for Windows 32
    - identifying 25
  - identifying descriptive data for extraction 99
  - limits for an information catalog 23
  - merging to facilitate adding to another information catalog 68
  - name (external name), rules for specifying 24
  - Program category, Programs object type defined 112
  - property
    - adding 25
    - adding during update 32
    - data types for 27
    - five common properties defined by DataGuide 25
    - maximum allowed LONG VARCHAR properties 27
    - maximum recommended length of UUI 28
    - part of UUI 27
    - rules for UUI 28
    - steps for adding, using DataGuide tag language 30
    - steps for adding, using DataGuide windows 26
  - relationship
    - Program, adding 58
    - Program, removing 62
  - relationships between 22
  - short name, rules for specifying 24
  - Support category
    - creating support facility with 63
    - DataGuide news, provided in sample information catalog 112
    - Online news services, provided in sample information catalog 112
    - Online publications, provided in sample information catalog 112
    - sample information catalog, provided in 112
  - updating
    - steps for, using DataGuide windows 32
    - steps for, using tag language 34
  - updating program association for
    - steps for, using DataGuide windows 60
- object type identifier property 25
- objects
  - exporting 72
  - importing 71
- OBJTYPID property 25
- online information and messages 94
- Online news services sample object type 112
- Online publications sample object type 112
- option
  - ACTION.RELATION 135
  - ADD
    - ACTION.OBJINST 119
    - ACTION.OBJTYPE 124, 137
    - ACTION.RELATION 127, 135
  - APPEND 124
  - DELETE 135
    - ACTION.OBJINST 119
    - ACTION.OBJTYPE 125
    - ACTION.RELATION 128
    - on OBJINST 132
  - DELETE\_EXT
    - ACTION.OBJTYPE 125
  - DELETE\_TREE\_ALL
    - ACTION.OBJINST 120
    - on OBJINST 132
  - DELETE\_TREE\_REL
    - ACTION.OBJINST 121
    - on OBJINST 132
  - MERGE
    - ACTION.OBJINST 122
    - ACTION.OBJTYPE 125, 137
  - UPDATE
    - ACTION.OBJINST 122, 134
    - ACTION.OBJTYPE 126
- output from customized extract program 65

**P**

- parameters, invocation
  - opening the Invocation Parameters window to specify 55
  - recommended value for programs 58
- parentheses, use of 46, 50
- PARMLIST keyword 59
- People to contact sample object type 111
- PHYNAME keyword 29, 137
- populating your information catalog 37
- Presentations sample object type 111
- primary administrator, identifying 18

- problems with DataGuide
  - closes unexpectedly 95
  - preventing 91
  - recovering from system failure 94
  - solving 91, 97
  - tools for solving 93
- Program category
  - definition of 22
  - relationships
    - Attachment, adding 54
    - Attachment, removing 54
    - summary of 22
- Program category, Programs object type defined 112
- program, extract
  - available from DataGuide Web site 98
  - steps for preparing to run 99
  - steps for running documented in README files 99
  - supplied with DataGuide 98
- programs
  - associating with object types
    - using DataGuide tag language 58
    - using DataGuide windows 55
  - copying associating with object types 60
  - disassociating with object types
    - using DataGuide tag language 62
    - using DataGuide windows 62
  - starting from object types, invocation parameters for 58
  - updating association with object types, using DataGuide windows 60
- Programs object type 112
- Programs that can be invoked from DataGuide objects 112
- property
  - adding
    - overview 25
    - steps for, using DataGuide tag language 30
    - steps for, using DataGuide windows 26
  - data types
    - CHAR 27
    - LONG VARCHAR 27
    - TIMESTAMP 27
    - VARCHAR 27
  - definition of 21
  - external name (NAME) 25
  - five common properties defined by DataGuide
    - hiding 25
    - instance identifier 25
    - last changed by 25
    - last changed date and time 25
    - object type identifier 25
- property (*continued*)
  - instance identifier (INSTIDNT) 25
  - last changed by (UPDATEBY) 25
  - last changed date and time (UPDATIME) 25
  - maximum recommended length of UUI 28
  - NAME 31
  - name (external), rules for specifying 26
  - object 66
  - object type identifier (OBJTYPID) 25
  - optional 35
  - part of UUI 27
  - Programs object, required for
    - Class 57
    - Identifier 57
    - Qualifier 1, 2, 3 57
  - rules for UUI 28
  - short name, rules for specifying 27
  - specifications for Attachment category object type, Comments 112
  - specifications for Contact category sample object type, People to contact 111
  - specifications for Dictionary category sample object type, Glossary entries 111
  - specifications for Elemental category sample object types
    - Audio clips 110
    - Charts 110
    - Documents 110
    - Images or graphics 110
    - Internet documents 110
    - Lotus Approach queries 110
    - Presentations 111
    - Spreadsheets 111
    - Text-based reports 111
    - Video clips 111
  - specifications for Grouping category sample object types
    - Application data 107
    - Business subject areas 107
    - Columns or fields 107
    - Databases 107
    - Dimensions within a multi-dimensional database 107
    - Elements 108
    - Files 108
    - IMS database definitions (DBD) 108
    - IMS program control blocks (PCB) 108
    - IMS program specification blocks (PSB) 108
    - IMS segments 108
    - Members within a multi-dimensional database 108

- property (*continued*)
  - specifications for Grouping category sample object types (*continued*)
    - Multi-dimensional databases 109
    - Records 109
    - Relational tables and views 109
    - Subschemas 109
    - Transformations 109
  - specifications for Program category object type
    - Programs that can be invoked from DataGuide objects 112
  - specifications for Support category sample object types
    - DataGuide news 112
    - Online news services 112
    - Online publications 112
- PROPERTY tag 65, 143, 146

## R

- reading syntax diagrams 118
- Records sample object type 109
- recovery, data 94
- register
  - information catalog 13
  - server node 13
- related publications 165
- Relational tables and views sample object type 109
- relationship
  - adding with tag language 46
  - between object types 22
  - between objects, modifying
    - Attachment 54
    - Contact 50
    - Contains 46
    - linked 48
  - deleting with tag language 46
- RELTYPE tag 65, 146, 147
- reserved words 115
- restarting the echo file 151
- restrictions for extract program 69
- rolling back data 151

## S

- sample information catalog
  - object types defined in 106
  - predefined program objects 113
- sample information catalog, creating 100

- SEQUENCE keyword 130
- setting up information catalog 1, 20
- settings notebook
  - default export options specified in 72
  - specifying whether to display 5 common properties 25
- short name
  - for object type, rules for specifying 24
  - for property, rules for specifying 27
  - of object type 29
  - of object type property 30
- short\_name 41
- SHRTDESC keyword 59
- SHRTNAME keyword
  - as a property 39
  - creating object types 30
  - optional property 35
  - PROPERTY tag 143
- SOURCEKEY keyword
  - ACTION.OBJINST(DELETE) 134
  - ACTION.RELATION 135
  - associating contacts 50
  - deleting object 62
  - tag language reference 133
  - using parentheses 46
- SOURCETYPE keyword
  - associating contacts 50
  - defining grouping 46
  - defining linked relationship 48
  - RELTYPE 146
- space, monitoring 91
- Spreadsheets sample object type 111
- STARTCMD keyword 59
- status list, comment 19
  - See also* comment status list
- Subschemas sample object type 109
- Support category
  - creating support facility 63
  - definition of 22
  - object types
    - DataGuide news, provided in sample information catalog 112
    - Online news services, provided in sample information catalog 112
    - Online publications, provided in sample information catalog 112
    - sample information catalog, provided in 112
  - relationships
    - Attachment, adding 54
    - Attachment, removing 54
    - Program, adding 58

- Support category *(continued)*
- relationships *(continued)*
  - Program, removing 62
  - summary of 22
- support facility, creating 63
- symptom
  - closes, unexpectedly 95
  - corrupt data 93
  - DataGuide failing 91
  - inconsistent data 93
  - lost files 92
  - users can't access information catalog 92
- syntax diagrams 118
- syntax rules for tag language 115
- system administration 2
- system failure, recovering from 94
- system registry information for DataGuide for Windows 93
- system-generated properties, hiding 25

## T

- T (TIMESTAMP) 30, 35
- TAB tag
  - planning for extract programs 65
  - tag language reference 147
- tag language
  - defining information 149
  - editing using word processor 65
  - extract programs for producing
    - supplied with DataGuide 98
    - writing customized 65
  - file
    - converting from MDIS 84
    - converting to MDIS 85
    - cutting and pasting online templates into viii
    - formatting 66
    - how DataGuide reads 116
    - importing into information catalog 71
    - MDIS-conforming, importing and exporting 85, 87
    - for creating object types 29, 66
    - for deleting object types 36
    - for merging object types 68
    - for updating object types 34
    - merging object types 68
    - overview 115
    - performing DataGuide tasks with vii
    - reference 115, 147
    - syntax rules 115

- tag language *(continued)*
- templates to cut and paste provided online viii
- tags
  - ACTION
    - OBJINST keyword 132
    - planning for extract program 65
    - sequence 150
    - tag language reference 118, 128
    - tips 149
  - COMMENT
    - planning for extract programs 65
    - tag language reference 128
    - when to use 151
  - COMMIT
    - during imports 69
    - planning for extract program 65
    - tag language reference 129, 130
    - when to use 151
  - contextual use of 149
  - DISKCNTRL
    - extract program 65
    - tag language reference 130
    - tips 149
  - INSTANCE 65, 131, 136
  - NL
    - planning for extract programs 65
    - tag language reference 136
  - NULLS 143
  - OBJECT 65
  - PROPERTY 65, 143, 146
  - RELTYPE 65
  - TAB
    - planning for extract programs 65
    - tag language reference 147
  - to define information 149
  - unsupported for national languages 116
- TARGETKEY keyword
  - ACTION.RELATION 135
  - associating contacts 50
  - using parentheses 46
- TARGETTYPE keyword
  - associating contacts 50
  - defining grouping 46
  - defining linked relationship 48
  - RELTYPE 146
- Text-based reports sample object type 111
- timestamp data type
  - on PROPERTY tag 143
  - optional property 35
  - property of DL 30

- TIMESTAMP data type for object type property 27
- trace (TRC) file
  - definition of 95
  - example of 96
  - interpreting 96
  - location of 95
  - rename after DataGuide closes unexpectedly 95
- Transformations sample object type 109
- troubleshooting DataGuide 91, 97
- TYPE keyword
  - creating
    - object types 29
    - objects 39
    - optional property 35
  - deleting object types 36
  - deleting objects 42
  - OBJTYPE(ADD) 137
  - OBJTYPE(APPEND) 140
  - OBJTYPE(DELETE) 140, 142
  - OBJTYPE(MERGE) 137
  - OBJTYPE(UPDATE) 141, 142
  - RELTYPE 146
  - updating object type 34

## U

- universal unique identifier (UII)
  - definition of 27
  - object type requirement 27
  - parts of 28
  - position of property in 67
  - property values 28, 134
  - rules for properties 28
  - UII\_short\_name
    - specifying when associating contacts 50
    - specifying when defining grouping 46
    - specifying when defining linked relationship 48
- UPDATE option
  - ACTION.OBJINST 122, 134
  - ACTION.OBJTYPE 126
- UPDATEBY property 25
- UPDATIME property 25
- URL, DataGuide home page 98
- user configuration files 92
- user ID
  - authorizing to DataGuide for Windows 2
  - changing DataGuide administrator with ALTERKA command 18
  - resetting logged-on administrator with CLEARKA command 94

- user interface (EUI), performing DataGuide tasks
  - with vii
  - Using DataGuide*, creating sample data for 100
- UII
  - definition of 27
  - object type requirement 27
  - parts of 28
  - position of property in 67
  - property values 28, 134
  - rules for properties 28
  - short\_name 41
  - UII\_short\_name
    - specifying when associating contacts 50
    - specifying when defining grouping 46
    - specifying when defining linked relationship 48
  - UII\_property\_value 133
  - UII\_short\_name keyword 50
  - UII\_short\_name value 133
  - UIICLASS keyword 59
  - UIIDENT keyword 59
  - UIEQUAL1, 2, 3 keywords 59
  - UISEQ keyword 30, 143
  - UISEQ option 67

## V

- V (VARCHAR) 30
- VARCHAR data type for object type property 27
- variable character data type
  - on PROPERTY tag 143
  - optional property 35
  - property of DL 30
- variable values 117
- Video clips sample object type 111
- Visual Warehouse
  - metadata
    - identifying for exchange 80
    - scheduling for exchange 82

## W

- Web site, URL for DataGuide 98
- writing customized extract programs 65
- writing tag language files 115



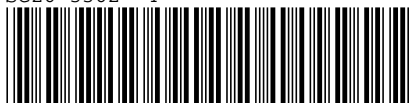
# IBM

Program Number: 5639-VW5



Printed in the United States of America  
on recycled paper containing 10%  
recovered post-consumer fiber.

SC26-3362- 4





*Spine information:*

**IBM**

IBM Visual Warehouse for Windows Managing DataGuide

Version 5 Release 2