Powersoft®

Open Tools from Sybase, Inc.

PowerBuilder

Advanced PowerBuilder Utilities

User's Guide

Version 6

Power Builder®

AB0880

October 1997

Contents

About This Book

Subject

This book describes how to use the Advanced PowerBuilder Utilities, a set of analysis and development utilities that extend the PowerBuilder development environment.

Audience

This book is for experienced PowerBuilder developers. It assumes you:

- Are currently developing applications using PowerBuilder
- Understand SQL and how to use your site-specific DBMS

CHAPTER 1 Getting Started

About this chapter

This chapter describes the Advanced PowerBuilder Utilities and how to start them and set them up.

Contents

About the Advanced PowerBuilder Utilities

The Advanced PowerBuilder Utilities are a set of utilities that extend the PowerBuilder development environment. The utilities automate, and in some cases eliminate, the traditional tasks involved in application development.

These tools include four analysis utilities and two development utilities.

Analysis utilities

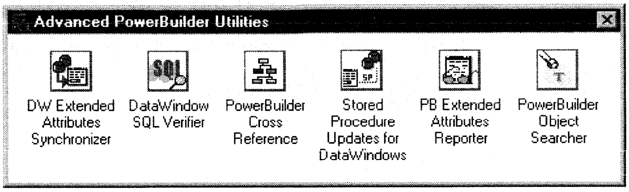
Utility	What it does
Cross Reference	Examines the relationships among the objects that make up a PowerBuilder application
DataWindow Extended Attribute Synchronizer (DWEAS)	Updates attributes of existing DataWindow objects from the PowerBuilder extended attribute tables
DataWindow SQL Verifier	Checks the syntax of the SQL used by existing DataWindow objects
PowerBuilder Extended Attribute Reporter (PEAR)	Generates reports on the contents of the PowerBuilder extended attribute tables

Development utilities

Utility	What it does
Stored Procedure Update	Creates PowerScript statements that use stored procedures to update the database through a DataWindow
Object Search	Scans one or more PowerBuilder libraries, looking for objects or text within objects

Using the launcher

When you start the Advanced PowerBuilder Utilities, the **launcher** displays:


Advanced PowerBuilder Utilities
DW Extended Attributes Synchronizer
DataWindow SQL Verifier
PowerBuilder Cross Reference
Stored Procedure Updates for DataWindows
PB Extended Attributes Reporter
PowerBuilder Object Searcher

On UNIX

On UNIX, start the Advanced PowerBuilder Utilities by typing **apu60.exe** from a shell prompt.

Starting a utility

To start one of the utilities, do one of the following:

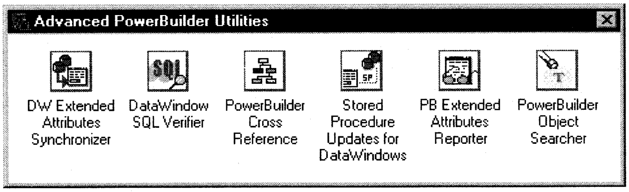
- Double-click the utility.
- Right-click the utility and select Run from the popup menu.

On Macintosh

On the Macintosh: select the utility, point at an empty area of the launcher, and press CONTROL+click.

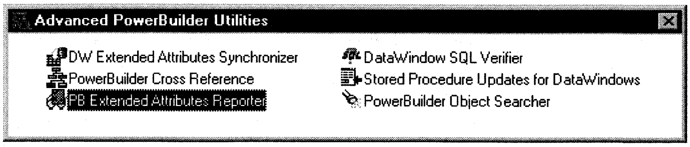
Changing views

The default launcher display is large icon view:


Advanced PowerBuilder Utilities
DW Extended Attributes Synchronizer
DataWindow SQL Verifier
PowerBuilder Cross Reference
Stored Procedure Updates for DataWindows
PB Extended Attributes Reporter
PowerBuilder Object Searcher

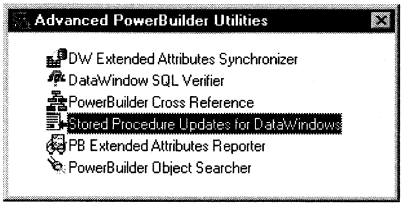
You can use the popup menu to select another view:

- Small icon view:



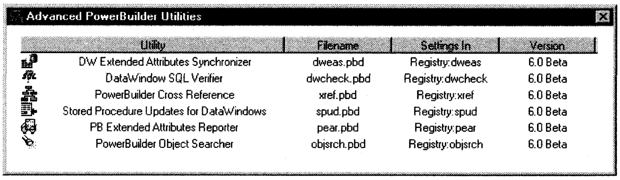
DW Extended Attributes Synchronizer
PowerBuilder Cross Reference
PB Extended Attributes Reporter
DataWindow SQL Verifier
Stored Procedure Updates for DataWindows
PowerBuilder Object Searcher

- List view:



DW Extended Attributes Synchronizer
DataWindow SQL Verifier
PowerBuilder Cross Reference
Stored Procedure Updates for DataWindows
PB Extended Attributes Reporter
PowerBuilder Object Searcher

- Report view:



Utility	Filename	Settings In	Version
DW Extended Attributes Synchronizer	dweas.pbd	Registry:dweas	6.0 Beta
DataWindow SQL Verifier	dwcheck.pbd	Registry:dwcheck	6.0 Beta
PowerBuilder Cross Reference	xref.pbd	Registry:xref	6.0 Beta
Stored Procedure Updates for DataWindows	spud.pbd	Registry:spud	6.0 Beta
PB Extended Attributes Reporter	pear.pbd	Registry:pear	6.0 Beta
PowerBuilder Object Searcher	objsrch.pbd	Registry:objsrch	6.0 Beta

Specifying application settings

You can use the launcher's popup menu to specify application settings.

About the settings These settings include database connection information and other utility-specific information. The utilities save application-specific settings in either the registry (Windows 95 and Windows NT) or an INI file (other platforms):

- **Windows 95 and Windows NT** The Advanced PowerBuilder Utilities save application settings in separate sections (one for each utility plus one for APU) of HKEY_CURRENT_USER\Software\Powersoft\PowerBuilder\6.0 in the registry
- **Other platforms** The Advanced PowerBuilder Utilities save application settings in separate INI files in the Advanced PowerBuilder Utilities directory

The Advanced PowerBuilder Utilities create and maintain these settings automatically.

Specifying settings You can modify application settings from the launcher or from within the individual utilities.

❖ **To modify a utility's application settings from the launcher:**

1 Point at a utility and right-click to display the popup menu.

2 Select Settings from the popup menu.

The utility's Application Settings dialog box displays.

3 Select a tab and modify settings as needed.

FOR INFO For more about the settings, see the individual utility chapters.

Starting utilities directly

You can bypass the launcher and set up individual icons for the utilities.

To do so, run the APU60.EXE program:

c:\program.files\powersoft\pb6\adk\apu\apu60.exe *utilityarg*

For this utility	Use this argument
Cross Reference	XREF
DataWindow Extended Attribute Synchronizer	DWEAS
DataWindow SQL Verifier	DWCHECK
PowerBuilder Extended Attribute Reporter	PEAR
Stored Procedure Update	SPUD
Object Search	OBJSRCH

To run Cross Reference directly, for example, specify this Run command:

```
c:\program files\powersoft\pb6\adk\apu\apu60.exe xref
```

On Macintosh

On the Macintosh, this capability is not enabled.

CHAPTER 2

Cross Reference

About this chapter

This chapter explains how to use Cross Reference.

Contents

About Cross Reference

Cross Reference reports the relationships among objects that make up a PowerBuilder application. These objects include windows, menus, DataWindows, user objects, global variables, shared variables, instance variables, global functions, window functions, user object functions, and PowerScript functions.

Why use Cross Reference

It is easy to see what objects make up an application using the Library painter, but it may be difficult to remember where each object was used in the application and the relationships among them.

With Cross Reference you can:

- **Determine effective use of objects** Cross Reference displays a list of objects in the application and the relationships among them. You can determine whether objects are being used effectively and whether there are new objects that can be created to reduce duplication of effort. For example, you may detect that only one object is using a global variable, thereby allowing you to remove the global variable. You may also notice that several descendent window objects are using two variations of the same global function. Those functions may be able to be combined and placed into the ancestor window, resulting in better performance and less code maintenance.

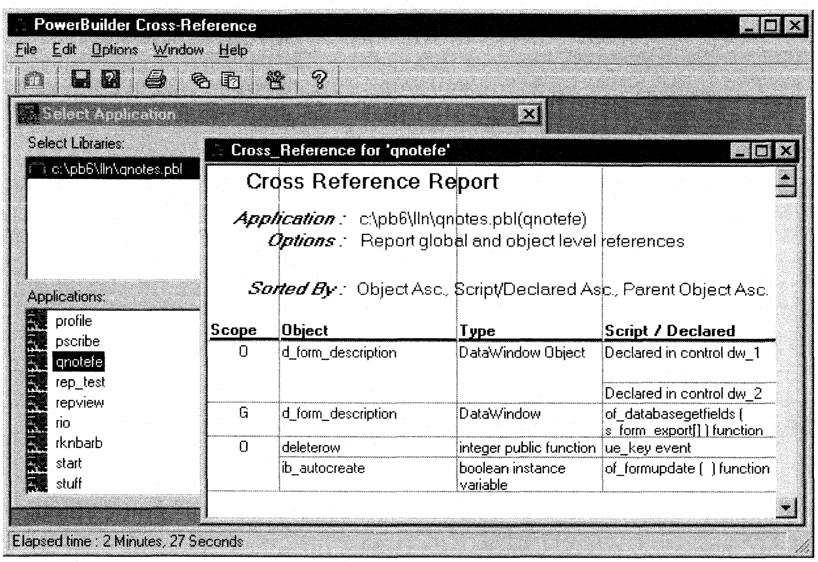
  (You can also display a list of the objects that are *not* referenced in the application.)

- **Determine where objects reside** Cross Reference also displays information about where DataWindow objects are used (as standard DataWindows, dropdown DataWindows, and nested reports), where external functions have been declared, and where global functions have been used in DataWindow objects.

How Cross Reference works

You select the application and which libraries in that application to analyze. Cross Reference analyzes your application.

The Cross Reference workspace looks like this:



Cross Reference Report

Application : c:\pb6\lln\qnotes.pbl(qnotefe)
Options : Report global and object level references

Sorted By : Object Asc., Script/Declared Asc., Parent Object Asc.

Scope	Object	Type	Script / Declared
O	d_form_description	DataWindow Object	Declared in control dw_1
			Declared in control dw_2
G	d_form_description	DataWindow	of_databasegetfields (s_form_export[]) function
O	deleterow	integer public function	ue_key event
	ib_autocreate	boolean instance variable	of_formupdate ( ) function

Elapsed time : 2 Minutes, 27 Seconds

Using Cross Reference

Using Cross Reference involves these steps:

- Specifying Cross Reference application settings
- Running Cross Reference
- Viewing the Cross Reference report

Starting Cross Reference
For how to start Cross Reference, see "Using the launcher" on page 3.

Specifying Cross Reference application settings

Before you run Cross Reference, you can specify connection information for the Cross Reference database and choose among several reporting options.

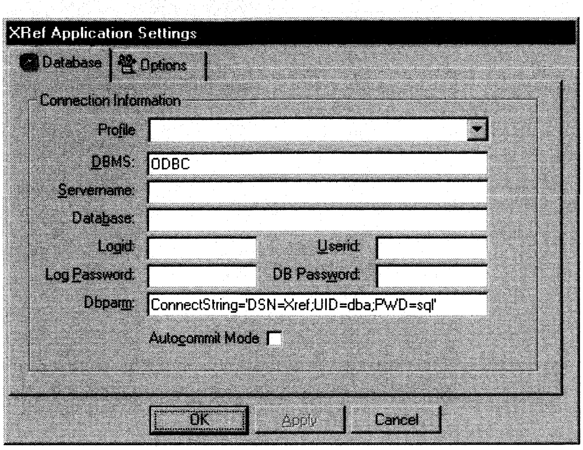
Specifying application settings before starting Cross Reference
You can also specify application settings from the launcher. For how to do this, see "Specifying application settings" on page 4.

❖ **To specify Cross Reference application settings:**

1 Select Options>Settings from the menu bar.

The XRef Application Settings dialog box displays.

2 Click the Database tab to specify connection information for the Cross Reference database:



DBMS: ODBC

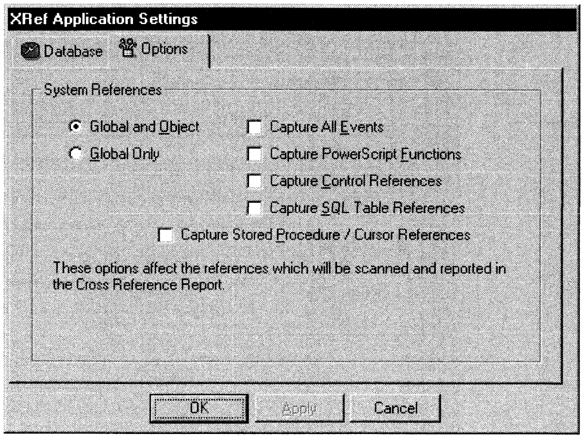
Dbparm: ConnectString='DSN=Xref;UID=dba;PWD=sql'

Autocommit Mode □

Database settings contain information on the current DBMS connection. You can specify:

Setting	Meaning
Profile	Name of the profile in the PB.INI file that specifies the database you will use
DBMS	Specific 3-character DBMS identifier
ServerName	Server name or connect string, if connecting to a remote DBMS. This value is dependent on the value entered in DBMS above
Database	Name of the database to connect to (for DBMSs that require it)
UserID	Identifier used to connect to the database
Database password	Password associated with the UserID
LogID	Identifier used to connect to the database
Logpassword	Password associated with the LogID
DBParm	DBMS-dependent keyword, typically used for ODBC
AutoCommit	Whether recoverable transaction processing is enabled

3 Click the Options tab to specify Cross Reference report options:



System References

● Global and Object
○ Global Only

□ Capture All Events
□ Capture PowerScript Functions
□ Capture Control References
□ Capture SQL Table References
□ Capture Stored Procedure / Cursor References

These options affect the references which will be scanned and reported in the Cross Reference Report.

You can specify these options for the Cross Reference report:

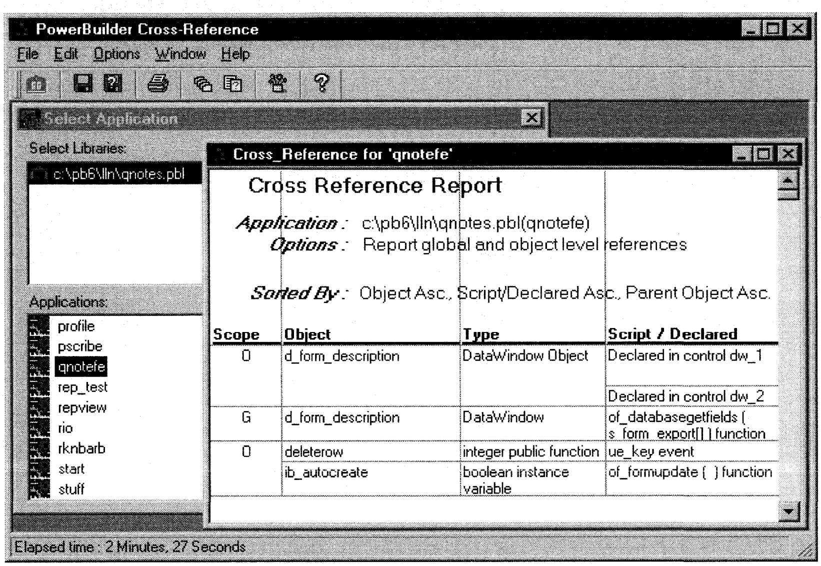
Setting	Reports on
Global & Object	Objects that contain shared variables, instance variables, or object functions that are referenced from *within* the selected object are analyzed in the Cross Reference report
Global Only	Global references only
Capture All Events	All events. If you cross-reference an event and Cross Reference finds no object references in the event, the report displays Script found for the event's object name
Capture PowerScript Functions	All PowerScript functions
Capture Control References	All control references
Capture SQL Table References	SQL table references
Capture Stored Procedure/Cursor References	Stored procedure and cursor usage

4 Click OK.

You see the Select Application window in the Cross Reference workspace.

Running Cross Reference

Once you have started Cross Reference and specified application settings, you see the Select Application window, with the application objects and the PBL files where they reside:



Cross Reference Report

Application : c:\pb6\lln\qnotes.pbl(qnotefe)
Options : Report global and object level references

Sorted By : Object Asc., Script/Declared Asc., Parent Object Asc.

Scope	Object	Type	Script / Declared
O	d_form_description	DataWindow Object	Declared in control dw_1
			Declared in control dw_2
G	d_form_description	DataWindow	of_databasegetfields (s_form_export[]) function
O	deleterow	integer public function	ue_key event
	ib_autocreate	boolean instance variable	of_formupdate ( ) function

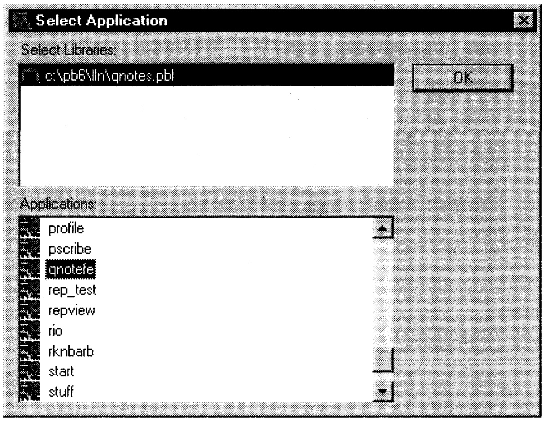
Elapsed time : 2 Minutes, 27 Seconds

What you do

Now you run Cross Reference.

❖ To run Cross Reference:

1 Select an application by clicking its name in the Applications box in the bottom half of the Select Application window:



If you see the word INVALID next to a library name, this means the application and libraries are specified in the PB.INI file, but the application no longer exists.

If the Select Application window is empty Cross Reference uses path information from the registry (Windows 95 and Windows NT) or the APU60.INI file (other platforms) to locate the PB.INI file (PowerBuilder Preferences on the Macintosh; PB.INI on UNIX) and access the list of applications displayed in the Select Application window. If the PB.INI file cannot be found, the Select Application window is empty. Check the location of the PB.INI file and modify path information as needed.

If a "Cannot read from drive" message displays Cross Reference checks to see if each application named in the PB.INI file exists. If this application resides on a drive that is unavailable (such as an empty diskette drive or an unattached network drive), file errors will interrupt the process and a message will display. Click Cancel to continue the checking process.

---

2 (Optional) Exclude libraries by clicking on them in the Select Libraries box in the top half of the window.

When you exclude a library, Cross Reference will use objects in that library as references, but will not look for references in that library.

3 Click OK.

What Cross Reference does

Cross Reference displays an empty report window and begins analysis. How long the analysis takes depends on the speed of your machine network, the number of objects in the selected application, and the application settings you specified.

Cross Reference works in two phases:

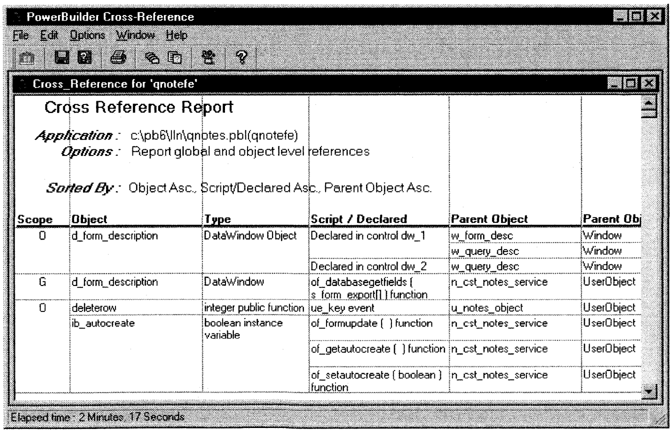
- **Phase 1** searches all objects in the application and creates a list of object references including variables, functions, and controls.
- **Phase 2** converts each object to source format and parses and searches the source.

  Phase 2 is an intensive process and may take some time for a large application. Depending on the level of analysis you specified, the parsing process views each object's scripts and variable declarations for object references:

  - If you selected *Global Only*, Cross Reference searches for primary objects (windows, DataWindows, menus, and global functions)
  - If you selected *Global & Object*, Cross Reference will also find for a given object the shared variables, instance variables, and object functions that are declared and referenced within the current object

Viewing the Cross Reference report

When Phase 2 is finished, the Cross Reference report window displays:



PowerBuilder Cross-Reference
File Edit Options Window Help
Cross_Reference for 'qnotefe'

Cross Reference Report

Application : c:\pb6\lln\qnotes.pbl(qnotefe)
Options : Report global and object level references

Sorted By : Object Asc., Script/Declared Asc., Parent Object Asc.

Scope	Object	Type	Script / Declared	Parent Object	Parent Obj
O	d_form_description	DataWindow Object	Declared in control dw_1	w_form_desc	Window
				w_query_desc	Window
			Declared in control dw_2	w_query_desc	Window
G	d_form_description	DataWindow	of_databasegetfields (s_form_export[]) function	n_cst_notes_service	UserObject
O	deleterow	integer public function	ue_key event	u_notes_object	UserObject
	ib_autocreate	boolean instance variable	of_formupdate ( ) function	n_cst_notes_service	UserObject
			of_getautocreate ( ) function	n_cst_notes_service	UserObject
			of_setautocreate (boolean) function	n_cst_notes_service	UserObject

Elapsed time : 2 Minutes, 17 Seconds

What you see

The detail section of the report window displays a row for each referenced item, with four columns:

Column	What it reports
Scope	Location of the object with respect to the application. It can be one of two values: ♦ **Global** The item is visible and accessible by the whole application ♦ **Object** The item is a reference to an object or a component of an object
Object	Name of the item referenced
Type	PowerBuilder data type and its scope (global, public, instance, and so on) for Object
Script/Declared	Name of the variable or event script that references Object
Parent Object	Name of the object that contains Script/Declared where Object is referenced
Parent Object Type	PowerBuilder data type of Parent Object
Parent Object Library	Filename of the library containing Parent Object

What you can do

You can save the Cross Reference report—in the Cross Reference database or in one of a variety of file formats, including HTML Table and Powersoft report. You can also see a list of unreferenced objects, sort and filter the results (with up to four levels of sorting), or change column order.

Saving Cross Reference results
You need to save your results before you exit Cross Reference. Any unsaved results are deleted when you exit.

❖ To show a list of unreferenced objects:

1. Select Options>Show Unreferenced from the menu bar.
2. To return to the report window, select Options>Show Cross Reference Report from the menu bar.

❖ To save the current report:

1. To save it in the Cross Reference database, select File>Save from the menu bar.
2. To save it in a flat file, select File>Save As from the menu bar.

❖ To display a saved report:

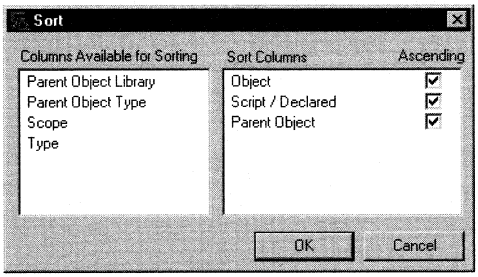
1 Select File>Open Saved Report from the menu bar.

2 Select the report.

❖ To sort the report:

1 Click the Sort button in the toolbar.

The Sort dialog box displays:



Columns Available for Sorting
- Parent Object Library
- Parent Object Type
- Scope
- Type

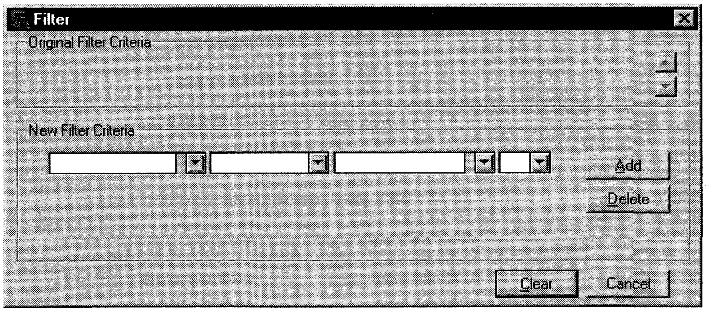
Sort Columns	Ascending
Object	☑
Script / Declared	☑
Parent Object	☑

2 Drag columns back and forth to specify the sort columns. Use the Ascending checkbox to specify the sort order (ascending or descending).

❖ To filter the report:

1 Click the Filter button in the toolbar.

The Filter dialog box displays:



2 Select a column, an operator, and a value.

❖ To change column order in the report:

♦ Click the column header area to select a column, drag the pointer to the place you want, and drop it.

❖ **To display columns ordered by parent object:**

- ♦ Select Options>Report by Parent from the menu bar.

❖ **To display columns ordered by referenced object:**

- ♦ Select Options>Report by Object from the menu bar.

About the Cross Reference database

Cross Reference saves results to the Cross Reference database XREF.DB. This section describes the three tables in the Cross Reference database.

App_classes table

Contents

Class information for the Cross Reference report. A row for every object in an application

Primary key

Application, pbl, object, nestedclass, parent, parenttypeof, and nestedin

Foreign key relationships

None

Columns

Column	Description
Application	Name of the application object
Pbl	Path and filename of the library containing the object
Object	Base object containing the class reference
Nestedclass	Nested class (for example controlname, function, event, variable, and so on)
Parent	Name of the parent of the nested class
Parenttypeof	Type of the parent class
Scope	Global or object
Nestedin	Name of the class the object is nested in
Isreferenced	Indicator that the class was referenced in the application

App_info table

Contents

Cross Reference report information for the application

Primary key

Application

Foreign key relationships

None

Columns

Column	Description
Application	The name of the application object
Report_globals	Whether the report included Global and Object references or Global references only
List_sql	Whether the report included SQL table references
List_sp	Whether the report included stored procedure references
List_ps_func	Whether the report included PowerScript functions
List_all_events	Whether the report captured all events
List_controls	Whether the report captured control references
Sort	String indicating sort order
Rep_date	Date the report was created
Rep_time	Time the report was created
Pbl	Path and filename of the PBL containing the application object

Xref_info table

Contents

Cross-reference information for all application objects

Primary key

All columns

Foreign key relationships

None

Columns

Column	Description
Object_ref	Object name
Object_ref_type	Object type
Event	Event or control where object is used
Referenced_in	Object containing the event or control where the referenced object is used
Ref_in_type	Object type of the object named in the previous column
Pbl	Filename of the PBL containing the referenced object

Column	Description
Application	Name of the application containing the referenced object
Scope	Global or object

CHAPTER 3

DataWindow Extended Attribute Synchronizer (DWEAS)

About this chapter

This chapter explains how to use the DataWindow Extended Attribute Synchronizer (DWEAS).

Contents

About DWEAS

The DataWindow Extended Attribute Synchronizer (DWEAS) is an interactive tool for updating the extended attributes of existing DataWindow objects with the current extended attributes from the Powersoft repository.

Why use DWEAS

The Powersoft repository contains extended attribute information for your database tables, including validation rules, display formats, edit styles, and text for column headings and labels.

When you create a DataWindow object, PowerBuilder uses information from the repository to create labels, headings, defaults, edit styles, display formats, and validation rules. But the relationship between an existing DataWindow and the repository is static: after a DataWindow is built, the DataWindow painter doesn't reference the repository. If an extended attribute changes in the repository, the DataWindow remains unchanged. The reason for this static design is that a dynamically linked repository can cause undesirable changes to DataWindow objects.

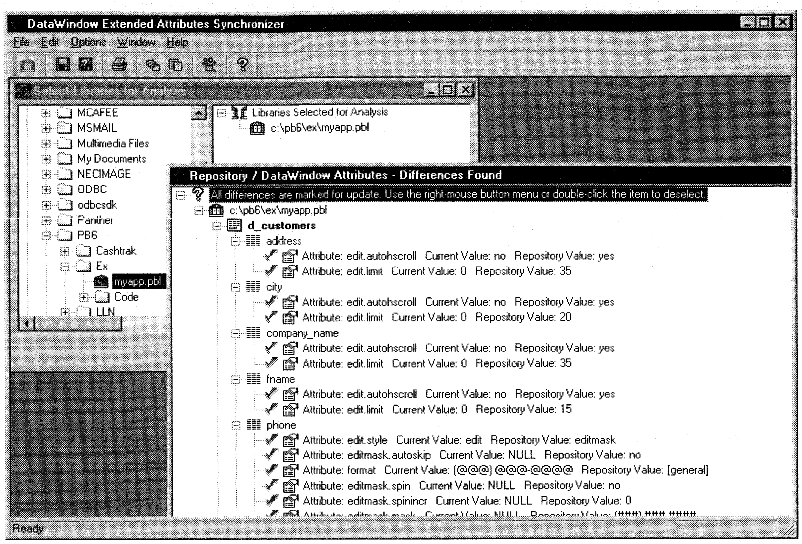
DWEAS allows you to selectively update DataWindow objects with current extended attributes from the repository.

How DWEAS works

DWEAS uses PowerBuilder's dynamic DataWindow facility to create new DataWindow objects that include the current extended attributes from the repository. Then DWEAS compares the brand-new DataWindow objects with the existing DataWindow objects and highlights any differences. You can then selectively mark differing extended attributes and update the existing DataWindow objects with the current repository values. Then you can save the updated DataWindow objects.

DWEAS examines a DataWindow's logical attributes only (column ID, validation information, edit style information, and so on). How DWEAS modifies and places the physical attributes (text color, font, font size, and so on) depends on the standards in the specific development environments.

The DWEAS workspace looks like this:


DataWindow Extended Attributes Synchronizer
File Edit Options Window Help
Select Libraries for Analysis
MCAFEE
MSMAIL
Multimedia Files
My Documents
NECIMAGE
ODBC
odbcsdk
Panther
PB6
Cashtrak
Ex
myapp.pbl
Code
LLN
Libraries Selected for Analysis
c:\pb6\ex\myapp.pbl
Repository / DataWindow Attributes - Differences Found
Ready

```
All differences are marked for update. Use the right-mouse button menu or double-click the item to deselect.
  c:\pb6\ex\myapp.pbl
    d_customers
      address
        Attribute: edit.autohscroll   Current Value: no   Repository Value: yes
        Attribute: edit.limit   Current Value: 0   Repository Value: 35
      city
        Attribute: edit.autohscroll   Current Value: no   Repository Value: yes
        Attribute: edit.limit   Current Value: 0   Repository Value: 20
      company_name
        Attribute: edit.autohscroll   Current Value: no   Repository Value: yes
        Attribute: edit.limit   Current Value: 0   Repository Value: 35
      fname
        Attribute: edit.autohscroll   Current Value: no   Repository Value: yes
        Attribute: edit.limit   Current Value: 0   Repository Value: 15
      phone
        Attribute: edit.style   Current Value: edit   Repository Value: editmask
        Attribute: editmask.autoskip   Current Value: NULL   Repository Value: no
        Attribute: format   Current Value: (@@@) @@@-@@@@   Repository Value: [general]
        Attribute: editmask.spin   Current Value: NULL   Repository Value: no
        Attribute: editmask.spinincr   Current Value: NULL   Repository Value: 0
        Attribute: editmask.mask   Current Value: NULL   Repository Value: (###) ###-####
```

Using DWEAS

Using DWEAS involves these steps:

- Specifying DWEAS application settings
- Running DWEAS
- Viewing the DWEAS report

Starting DWEAS
For how to start DWEAS, see "Using the launcher" on page 3.

Specifying DWEAS application settings

Before you run DWEAS, you can specify the connection parameters for your DBMS and the default libraries that DWEAS selects for analysis when DWEAS starts up.

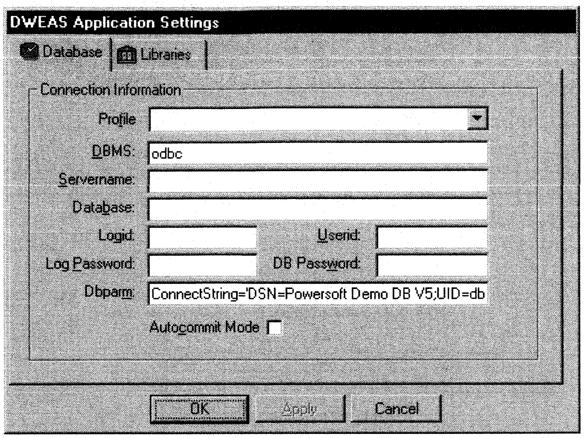
Specifying application setting before starting DWEAS
You can also specify application settings from the launcher. For how to do this, see "Specifying application settings" on page 4.

❖ **To specify DWEAS application settings:**

1 Select Options>Settings from the menu bar.

The DWEAS Application Settings dialog box displays.

2 Click the DBMS tab to specify connection information for the database to be analyzed:

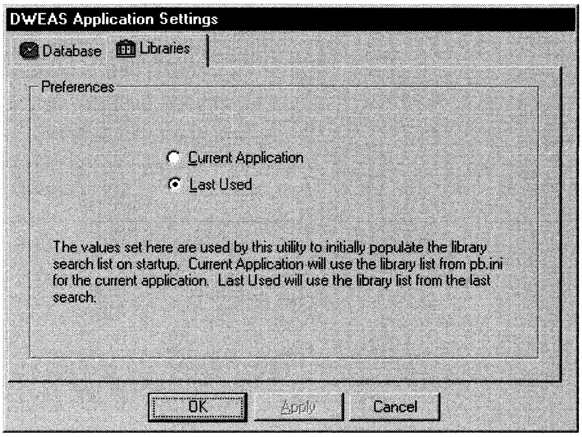


Profile:
DBMS: odbc
Servername:
Database:
Logid:
Userid:
Log Password:
DB Password:
Dbparm: ConnectString='DSN=Powersoft Demo DB V5;UID=db
Autocommit Mode □

Database settings contain information on the current DBMS connection. You can specify:

Setting	Meaning
Profile	Name of the profile in the PB.INI file that specifies the database you will use
DBMS	Specific 3-character DBMS identifier
ServerName	Server name or connect string, if connecting to a remote DBMS. This value is dependent on the value entered in DBMS above
Database	Name of the database to connect to (for DBMSs that require it)
UserID	Identifier used to connect to the database
Database password	Password that is associated with the UserID
LogID	Identifier used to connect to the database
Logpassword	Password that is associated with the LogID
DBParm	DBMS-dependent keyword, typically used for ODBC
AutoCommit	Specifies whether recoverable transaction processing is enabled

3 Click the Libraries tab to specify the default libraries selected for analysis that will display at startup in the right frame of the Select Libraries for Analysis window:



○ Current Application
● Last Used

The values set here are used by this utility to initially populate the library search list on startup. Current Application will use the library list from pb.ini for the current application. Last Used will use the library list from the last search.

You can specify:

Setting	Meaning
Current Application	DWEAS displays the libraries specified in the PB.INI file for PowerBuilder's current application
Last Used	DWEAS displays the libraries you last searched in

Running DWEAS

What you do

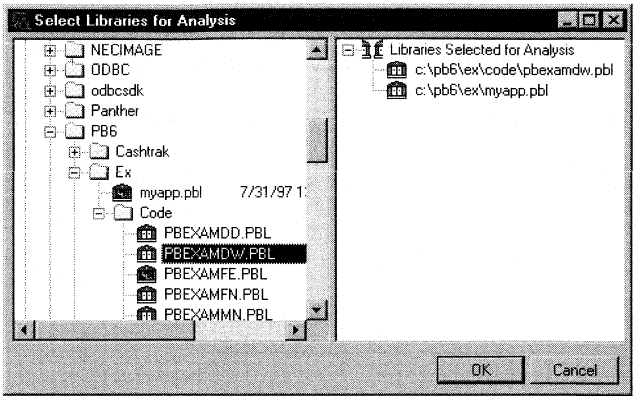
Once you have started DWEAS and specified application settings, you select libraries from the Select Libraries for Analysis window.

❖ To run DWEAS:

1 In the left frame of the Select Libraries for Analysis window, select the PowerBuilder libraries you need by:

- Dragging a library or a folder from the left frame and dropping it in the right frame

- Double-clicking a library or folder to display libraries in the right frame



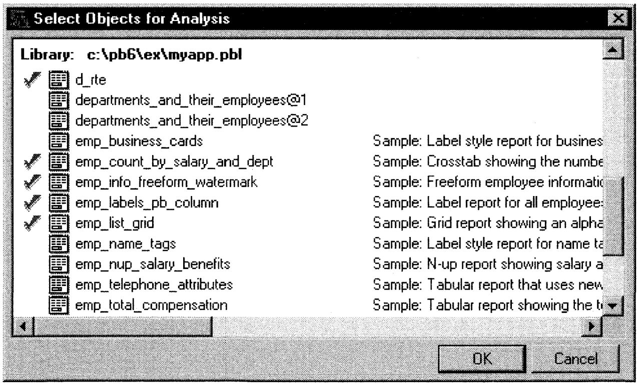
NECIMAGE
ODBC
odbcsdk
Panther
PB6
Cashtrak
Ex
myapp.pbl 7/31/97 1:
Code
PBEXAMDD.PBL
PBEXAMDW.PBL
PBEXAMFE.PBL
PBEXAMFN.PBL
PBEXAMMN.PBL

Libraries Selected for Analysis
c:\pb6\ex\code\pbexamdw.pbl
c:\pb6\ex\myapp.pbl

2 Click OK.

The Select Objects for Analysis window displays the DataWindows in the libraries you specified.

3 In the Select Objects for Analysis window:
Click a DataWindow to select that DataWindow.
or
Use CTRL+click or SHIFT+click to select multiple DataWindows.



Library: c:\pb6\ex\myapp.pbl

✓	d_rte	
	departments_and_their_employees@1	
	departments_and_their_employees@2	
	emp_business_cards	Sample: Label style report for busines
✓	emp_count_by_salary_and_dept	Sample: Crosstab showing the numbe
✓	emp_info_freeform_watermark	Sample: Freeform employee informatic
✓	emp_labels_pb_column	Sample: Label report for all employee:
✓	emp_list_grid	Sample: Grid report showing an alpha
	emp_name_tags	Sample: Label style report for name ta
	emp_nup_salary_benefits	Sample: N-up report showing salary a
	emp_telephone_attributes	Sample: Tabular report that uses new
	emp_total_compensation	Sample: Tabular report showing the t

4 Click OK.

What DWEAS does

DWEAS loads and analyzes the DataWindow objects. How long the analysis takes depends on the speed of your machine, network, DBMS, and the number of columns in the DataWindow objects.

For each object, DWEAS:

1 Loads the DataWindow object definition.

2 Creates an instance of the DataWindow object.

3 Accesses the SQL SELECT statement used to generate the DataWindow object.

4 Internally generates a Dynamic DataWindow object based on the SQL SELECT statement from step 3.

5 Analyzes the extended-attribute differences for the columns of the DataWindow objects. This includes display formatting, validation rules, and edit style information.

6 Displays the findings in the Repository/DataWindow Attributes - Differences Found window.

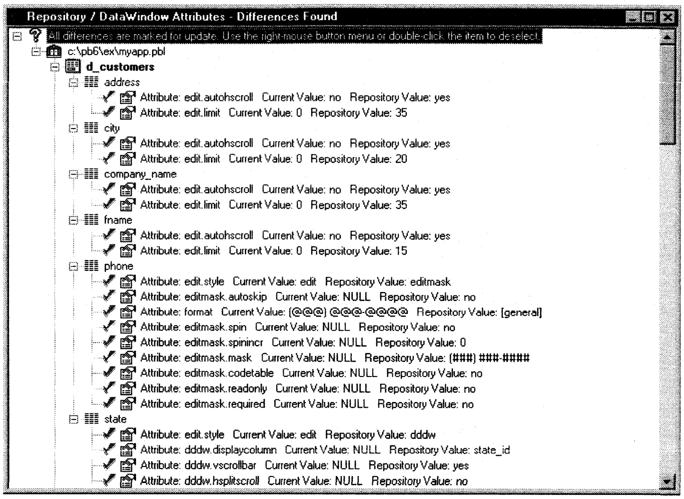
Viewing the DWEAS report

This section describes how to:

- Identify differences between the current and repository attribute values
- Select attributes

What you see

The Repository/DataWindow Attributes - Differences Found window to displays the column identification information as well as the current and repository attribute values. The current values are the attributes as they exist in the selected DataWindow object, and the repository values are the attributes as they exist in the repository:



```
All differences are marked for update. Use the right-mouse button menu or double-click the item to deselect.
  c:\pb6\ex\myapp.pbl
    d_customers
      address
        ✓ Attribute: edit.autohscroll  Current Value: no  Repository Value: yes
        ✓ Attribute: edit.limit  Current Value: 0  Repository Value: 35
      city
        ✓ Attribute: edit.autohscroll  Current Value: no  Repository Value: yes
        ✓ Attribute: edit.limit  Current Value: 0  Repository Value: 20
      company_name
        ✓ Attribute: edit.autohscroll  Current Value: no  Repository Value: yes
        ✓ Attribute: edit.limit  Current Value: 0  Repository Value: 35
      fname
        ✓ Attribute: edit.autohscroll  Current Value: no  Repository Value: yes
        ✓ Attribute: edit.limit  Current Value: 0  Repository Value: 15
      phone
        ✓ Attribute: edit.style  Current Value: edit  Repository Value: editmask
        ✓ Attribute: editmask.autoskip  Current Value: NULL  Repository Value: no
        ✓ Attribute: format  Current Value: (@@@) @@@-@@@@  Repository Value: [general]
        ✓ Attribute: editmask.spin  Current Value: NULL  Repository Value: no
        ✓ Attribute: editmask.spinincr  Current Value: NULL  Repository Value: 0
        ✓ Attribute: editmask.mask  Current Value: NULL  Repository Value: (###) ###-####
        ✓ Attribute: editmask.codetable  Current Value: NULL  Repository Value: no
        ✓ Attribute: editmask.readonly  Current Value: NULL  Repository Value: no
        ✓ Attribute: editmask.required  Current Value: NULL  Repository Value: no
      state
        ✓ Attribute: edit.style  Current Value: edit  Repository Value: dddw
        ✓ Attribute: dddw.displaycolumn  Current Value: NULL  Repository Value: state_id
        ✓ Attribute: dddw.vscrollbar  Current Value: NULL  Repository Value: yes
        ✓ Attribute: dddw.hsplitscroll  Current Value: NULL  Repository Value: no
```

The extended attributes are selected by default for updating with the repository values.

What you can do

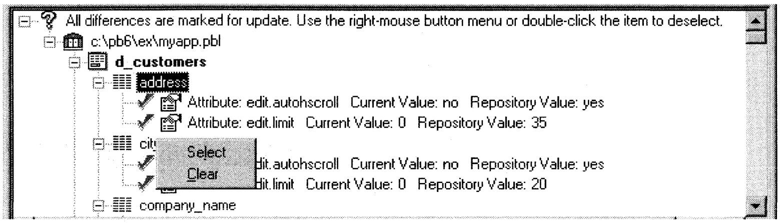
You can clear and select extended attributes as needed to specify the extended attributes you want updated with the repository values. You can also sort and filter the report.

❖ To clear or select extended attributes for updating with the repository values:

1 Double-click an extended attribute name to clear or select the extended attribute.

or

At any level in the tree (library, DataWindow, column, or extended attribute), right-click, and select Clear or Select from the popup menu to clear or select all extended attributes at that level:



All differences are marked for update. Use the right-mouse button menu or double-click the item to deselect.
c:\pb6\ex\myapp.pbl
d_customers
address
Attribute: edit.autohscroll Current Value: no Repository Value: yes
Attribute: edit.limit Current Value: 0 Repository Value: 35
cit
it.autohscroll Current Value: no Repository Value: yes
it.limit Current Value: 0 Repository Value: 20
company_name

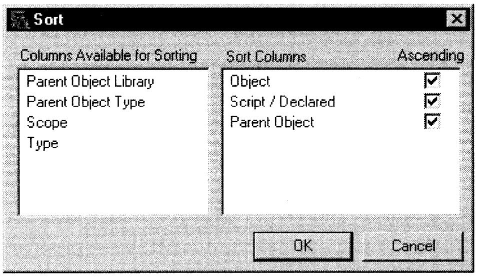
2 Continue specifying DataWindow attributes for updating as needed.

3 Save the updated DataWindows using File>Save.
or
Save the updated DataWindows with new names using File>Save As.

❖ **To sort the report:**

1 Click the Sort button in the toolbar.

The Sort dialog box displays:



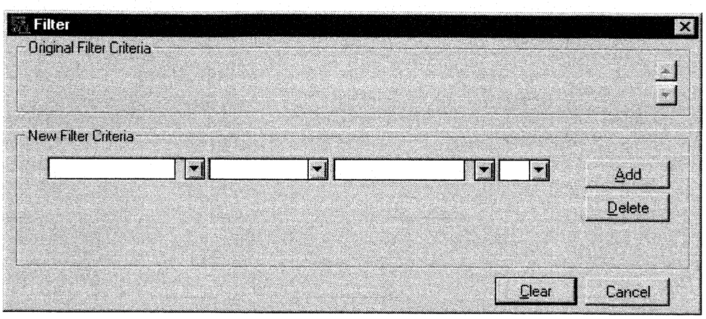
Columns Available for Sorting	Sort Columns	Ascending
Parent Object Library	Object	☑
Parent Object Type	Script / Declared	☑
Scope	Parent Object	☑
Type		

2 Drag columns back and forth to specify the sort columns. Use the Ascending checkbox to specify the sort order (ascending or descending).

❖ To filter the report:

1 Click the Filter button in the toolbar.

The Filter dialog box displays:



2 Select a column, an operator, and a value.

CHAPTER 4 DataWindow SQL Verifier

About this chapter

This chapter explains how to use the DataWindow SQL Verifier.

Contents

About DataWindow SQL Verifier

The DataWindow SQL Verifier checks the validity of SQL statements used by the DataWindow objects you select in the libraries you specify.

Why use the DataWindow SQL Verifier

When DataWindow objects designed for one DBMS are migrated to another DBMS, it may be necessary to check the SQL statements created when the DataWindows were built. For example, one DBMS may require that all SQL statements be enclosed in quotation marks and another may flag quotation marks as invalid. You can use the DataWindow SQL Verifier to test each DataWindow against the target DBMS.

You can also use the DataWindow SQL Verifier to determine whether columns have been added or deleted from database tables.

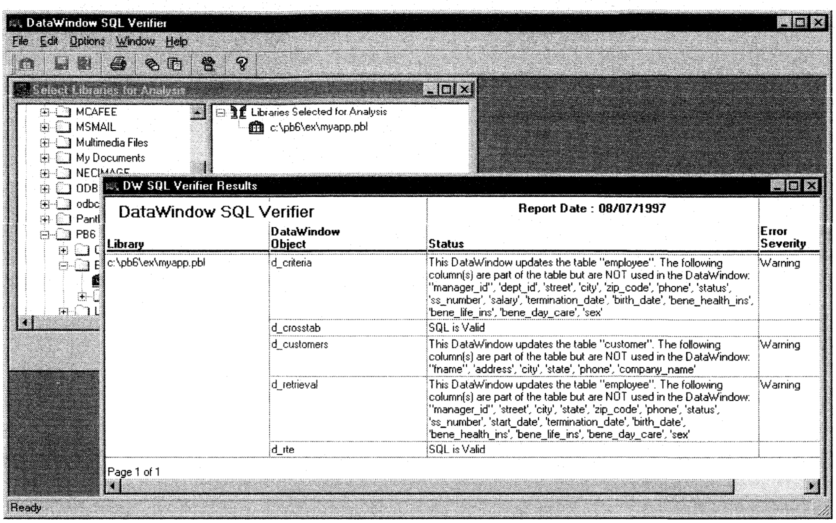
How the DataWindow SQL Verifier works

The DataWindow SQL Verifier works with the DataWindows you select in the libraries you specify. The DataWindow SQL Verifier extracts the SQL statement from each of the DataWindow objects, executes it against the current DBMS, and generates a report identifying any problems.

Typical problems are invalid column and table names. The DataWindow SQL Verifier also ensures that updatable columns in the DataWindow object match column names in the target DBMS.

For analysis of the DataWindow structure
For a more detailed analysis of the DataWindow at a structural level, see Chapter 3, "DataWindow Extended Attribute Synchronizer (DWEAS)".

The DataWindow SQL Verifier workspace looks like this:



DataWindow SQL Verifier

Report Date : 08/07/1997

Library	DataWindow Object	Status	Error Severity
c:\pb6\ex\myapp.pbl	d_criteria	This DataWindow updates the table "employee". The following column(s) are part of the table but are NOT used in the DataWindow: "manager_id", 'dept_id', 'street', 'city', 'zip_code', 'phone', 'status', 'ss_number', 'salary', 'termination_date', 'birth_date', 'bene_health_ins', 'bene_life_ins', 'bene_day_care', 'sex'	Warning
	d_crosstab	SQL is Valid	
	d_customers	This DataWindow updates the table "customer". The following column(s) are part of the table but are NOT used in the DataWindow: "fname", 'address', 'city', 'state', 'phone', 'company_name'	Warning
	d_retrieval	This DataWindow updates the table "employee". The following column(s) are part of the table but are NOT used in the DataWindow: "manager_id", 'street', 'city', 'state', 'zip_code', 'phone', 'status', 'ss_number', 'start_date', 'termination_date', 'birth_date', 'bene_health_ins', 'bene_life_ins', 'bene_day_care', 'sex'	Warning
	d_rte	SQL is Valid	

Page 1 of 1

Using the DataWindow SQL Verifier

This section describes how to use the DataWindow SQL Verifier:

- Specifying DataWindow SQL Verifier application settings
- Running the DataWindow SQL Verifier
- Viewing the DataWindow SQL Verifier report

Starting the DataWindow SQL Verifier
For how to start the DataWindow SQL Verifier, see "Using the launcher" on page 3.

Specifying DataWindow SQL Verifier application settings

Before you run the DataWindow SQL Verifier, you may need to specify application settings, particularly a connection and connection parameters to another DBMS. The DataWindow SQL Verifier performs the analysis of the SQL in the DataWindows against the current DBMS.

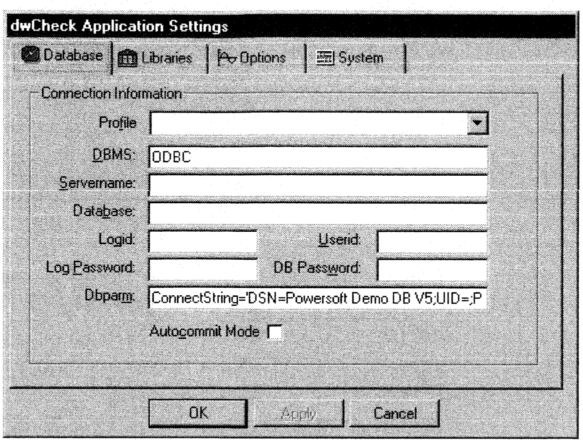
Specifying application setting before starting the DataWindow SQL Verifier
You can also specify application settings from the launcher. For how to do this, see "Specifying application settings" on page 4.

❖ To specify DataWindow SQL Verifier application settings:

1 Select Options>Settings from the menu bar.

The dwCheck Application Settings dialog box displays.

2 Click the Database tab to specify connection information for the database:



dwCheck Application Settings

Connection Information

Profile:

DBMS: ODBC

Servername:

Database:

Logid:

Userid:

Log Password:

DB Password:

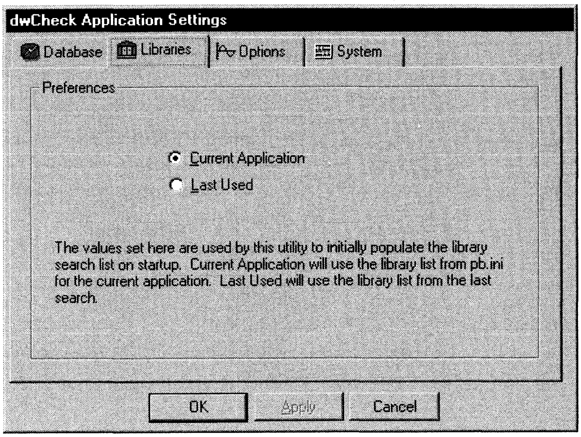
Dbparm: ConnectString='DSN=Powersoft Demo DB V5;UID=;P

Autocommit Mode □

Database settings contain information on the current DBMS connection. You can specify:

Setting	Meaning
Profile	Name of the profile in the PB.INI file that specifies the database you will use
DBMS	Specific 3-character DBMS identifier
ServerName	Server name or connect string, if connecting to a remote DBMS. This value is dependent on the value entered in DBMS above
Database	Name of the database to connect to (for DBMSs that require it)
UserID	Identifier used to connect to the database
Database password	Password associated with the UserID
LogID	Identifier used to connect to the database
Logpassword	Password associated with the LogID
DBParm	DBMS-dependent keyword, typically used for ODBC
AutoCommit	Whether recoverable transaction processing is enabled

3 Click the Libraries tab to specify at startup the libraries selected for analysis (in the right frame of the Select Libraries for Analysis window):



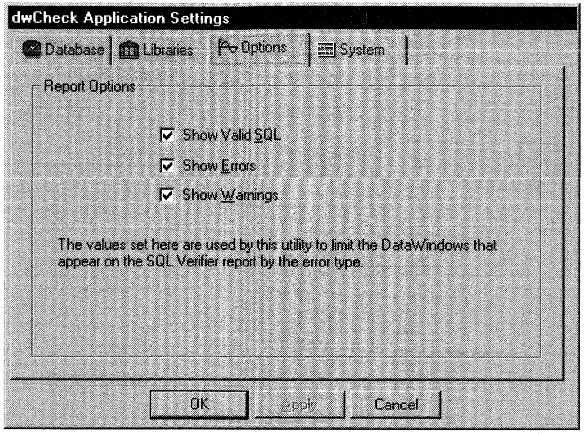
● Current Application
○ Last Used

The values set here are used by this utility to initially populate the library search list on startup. Current Application will use the library list from pb.ini for the current application. Last Used will use the library list from the last search.

You can specify:

Setting	Meaning
Current Application	The DataWindow SQL Verifier displays the libraries specified in the PB.INI file for PowerBuilder's current application
Last Used	The DataWindow SQL Verifier displays the libraries you last searched in

4 Click the Options tab to specify restrictions on the error messages displayed by the DataWindow SQL Verifier:



Report Options

☑ Show Valid SQL

☑ Show Errors

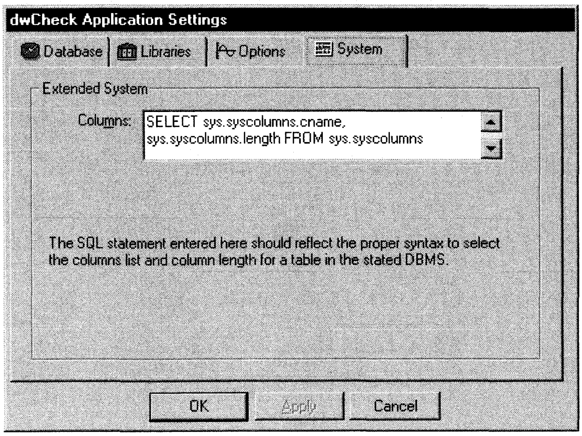
☑ Show Warnings

The values set here are used by this utility to limit the DataWindows that appear on the SQL Verifier report by the error type.

You can specify:

Setting	Meaning
Show Valid SQL	DataWindows with valid SQL are displayed
Show Errors	DataWindow with errors are displayed
Show Warnings	DataWindows with warnings are displayed

5 Click the System tab to specify a SQL SELECT statement that the DataWindow SQL Verifier uses to select a list of columns for a specified table:



SELECT sys.syscolumns.cname, sys.syscolumns.length FROM sys.syscolumns

The SQL statement entered here should reflect the proper syntax to select the columns list and column length for a table in the stated DBMS.

To specify the SQL statement, type the SQL statement in the Columns edit box. The SQL statement must return one column: the column name. Additionally, the statement must include a retrieval argument for table name and the argument must be *:table_name*.

6 Click OK.

Running the DataWindow SQL Verifier

What you do

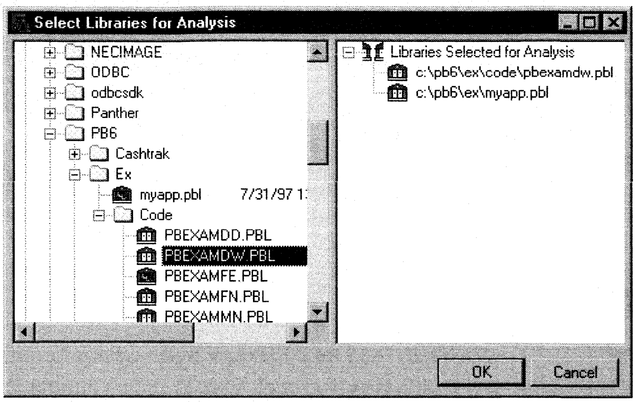
Once you have started the DataWindow SQL Verifier and specified application settings, you select libraries and then DataWindows.

❖ **To run the DataWindow SQL Verifier:**

1 In the left frame of the Select Libraries for Analysis window, select PowerBuilder libraries by:

- Dragging a library or a folder containing libraries to the right frame.

- ♦ Double-clicking the library name or folder name to display libraries in the right frame.



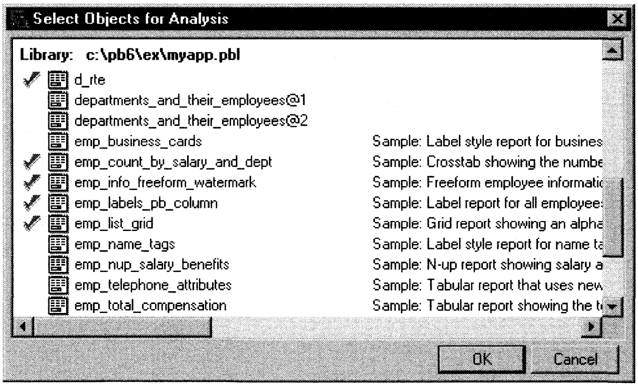
```
NECIMAGE
ODBC
odbcsdk
Panther
PB6
  Cashtrak
  Ex
    myapp.pbl      7/31/97 1:
    Code
      PBEXAMDD.PBL
      PBEXAMDW.PBL
      PBEXAMFE.PBL
      PBEXAMFN.PBL
      PBEXAMMN.PBL
```

```
Libraries Selected for Analysis
  c:\pb6\ex\code\pbexamdw.pbl
  c:\pb6\ex\myapp.pbl
```

2 Click OK.

The Select Objects for Analysis window displays the DataWindows in the libraries you specified.

3 In the Select Objects for Analysis window:
Click a DataWindow to select that DataWindow.
or
Use CTRL+click or SHIFT+click to select multiple DataWindows.



Library: c:\pb6\ex\myapp.pbl

	Object	Description
✓	d_rte	
	departments_and_their_employees@1	
	departments_and_their_employees@2	
	emp_business_cards	Sample: Label style report for busines
✓	emp_count_by_salary_and_dept	Sample: Crosstab showing the numbe
✓	emp_info_freeform_watermark	Sample: Freeform employee informatic
✓	emp_labels_pb_column	Sample: Label report for all employee:
✓	emp_list_grid	Sample: Grid report showing an alpha
	emp_name_tags	Sample: Label style report for name ta
	emp_nup_salary_benefits	Sample: N-up report showing salary a
	emp_telephone_attributes	Sample: Tabular report that uses new
	emp_total_compensation	Sample: Tabular report showing the t

4 Click OK.

What the DataWindow SQL Verifier does

The DataWindow SQL Verifier displays an empty results window and then begins analysis of each DataWindow object. How long the analysis takes depends on the speed of your machine and network and the number of DataWindow objects.

The DataWindow SQL Verifier scans the DataWindow objects, displaying each DataWindow object name on the status line and in the results window. Then for each DataWindow object, the DataWindow SQL Verifier:

1 Retrieves and examines the SQL statement.

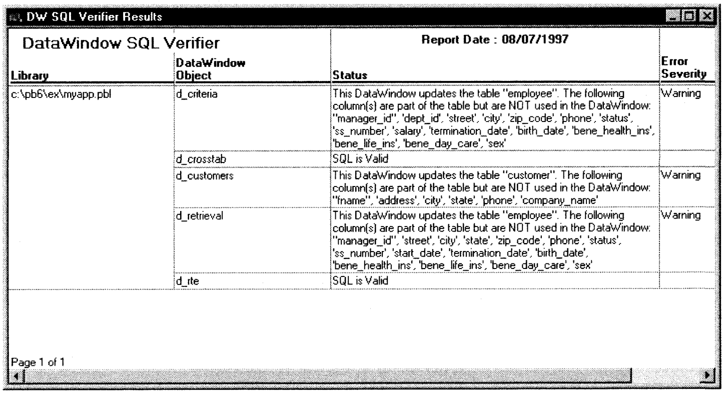
2 Submits the SQL statement to the DBMS and checks the result for errors.

3 Displays the status in the results window.

Viewing the DataWindow SQL Verifier report

What you see

When the DataWindow SQL Verifier completes the analysis, a report displays with a row for each DataWindow object analyzed:



DataWindow SQL Verifier

Report Date : 08/07/1997

Library	DataWindow Object	Status	Error Severity
c:\pb6\ex\myapp.pbl	d_criteria	This DataWindow updates the table "employee". The following column(s) are part of the table but are NOT used in the DataWindow: "manager_id", 'dept_id', 'street', 'city', 'zip_code', 'phone', 'status', 'ss_number', 'salary', 'termination_date', 'birth_date', 'bene_health_ins', 'bene_life_ins', 'bene_day_care', 'sex'	Warning
	d_crosstab	SQL is Valid	
	d_customers	This DataWindow updates the table "customer". The following column(s) are part of the table but are NOT used in the DataWindow: "fname", 'address', 'city', 'state', 'phone', 'company_name'	Warning
	d_retrieval	This DataWindow updates the table "employee". The following column(s) are part of the table but are NOT used in the DataWindow: "manager_id", 'street', 'city', 'state', 'zip_code', 'phone', 'status', 'ss_number', 'start_date', 'termination_date', 'birth_date', 'bene_health_ins', 'bene_life_ins', 'bene_day_care', 'sex'	Warning
	d_rte	SQL is Valid	

Page 1 of 1

The rows are divided into four columns:

Column	What it displays
Library	The name of the PowerBuilder Library file (PBL) that contains the referenced DataWindow object. This column is blank when the library is the same as the preceding one
DataWindow Object	The name of the referenced DataWindow object

Column	What it displays
Status	The result of the check of the DataWindow SQL. One of the following messages displays: **SQL Is Valid** The SQL was executed without producing any errors **Script Source DataWindow** The DataWindow is script-driven and contains no SQL statement **Missing column** Column named in the DataWindow no longer exists in the database **Unable to create DataWindow** DataWindow could not be generated: possible version mismatch: *Incomplete update message* Your DataWindow updates the database but does not access all database columns. This may not be a problem, however, it could indicate that columns have been added to the table since the DataWindow was created *DBMS-specific error message* An error occurred while executing the SQL statement; the message returned by the DBMS is displayed here
Error Severity	Severity of the error: ♦ **Warning** Not all columns are in use; there is a potential for update errors ♦ **Error** SQL error ♦ **Valid** SQL has no errors

What you can do

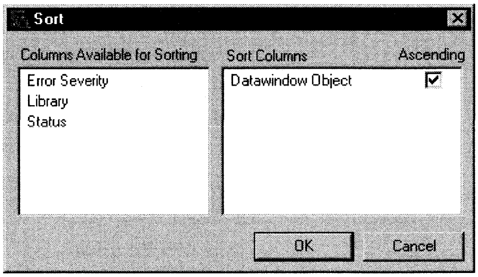
You can sort, filter, and print the DataWindow SQL Verifier report.

Print reports before exiting
If you have generated DataWindow SQL Verifier reports that you want to keep, be sure to print them before exiting. When you exit the DataWindow SQL Verifier, all results are deleted.

❖ To sort a DataWindow SQL Verifier report:

1 Click the Sort button on the toolbar.

The Sort dialog box displays:

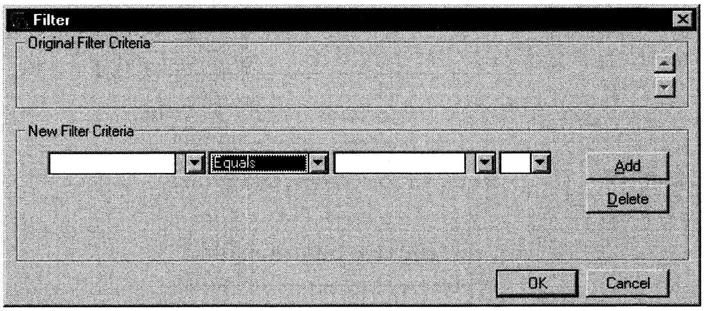


2 Drag columns back and forth to specify the sort columns. Use the Ascending checkbox to specify the sort order (ascending or descending).

❖ To filter a DataWindow SQL Verifier report:

1 Click the Filter button on the toolbar.

The Filter dialog box displays:



2 Select a column, an operator, and a value.

CHAPTER 5

PowerBuilder Extended Attribute Reporter (PEAR)

About this chapter

This chapter explains how to use the PowerBuilder Extended Attribute Reporter (PEAR).

Contents

About PEAR

The PowerBuilder Extended Attribute Reporter (PEAR) reports on extended attributes contained in the PowerBuilder repository.

Why use PEAR

PEAR displays extended attributes for one or more chosen database tables, including labels, headers, justification, case, display format, validation rules, edit styles, and initial value. You can then determine whether or not you are using the extended attributes appropriately.

How Pear works

PEAR accesses your database system tables to identify:

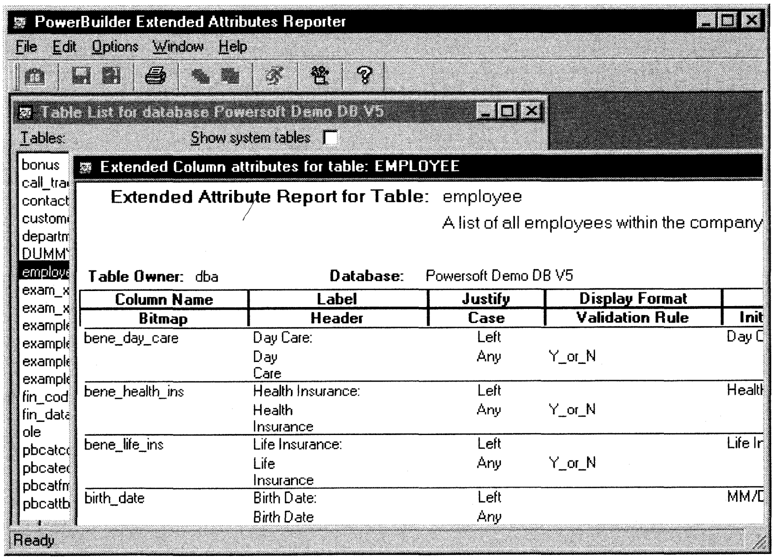
- Tables that can be displayed
- Columns in tables selected for reporting

PEAR then accesses the Powersoft repository and displays a report of the extended attributes for all columns in the selected tables.

PEAR uses DBMS-specific SQL syntax to access the system tables that contain database table and column definitions. You may need to modify the default SQL syntax.

FOR INFO For information about modifying the SQL syntax, see "Specifying database parameter defaults" on page 53.

The PEAR workspace looks like this:



Extended Attribute Report for Table: employee

A list of all employees within the company

Table Owner: dba **Database:** Powersoft Demo DB V5

Column Name	Label	Justify	Display Format	
Bitmap	**Header**	**Case**	**Validation Rule**	**Init**
bene_day_care	Day Care: Day Care	Left Any	 Y_or_N	Day C
bene_health_ins	Health Insurance: Health Insurance	Left Any	 Y_or_N	Healt
bene_life_ins	Life Insurance: Life Insurance	Left Any	 Y_or_N	Life Ir
birth_date	Birth Date: Birth Date	Left Any		MM/D

Using PEAR

This section describes how to use PEAR:

- Specifying DBMS connection information
- Running PEAR
- Viewing a PEAR report
- Specifying database parameter defaults

Starting PEAR
For how to start PEAR, see "Using the launcher" on page 3.

Specifying DBMS connection information

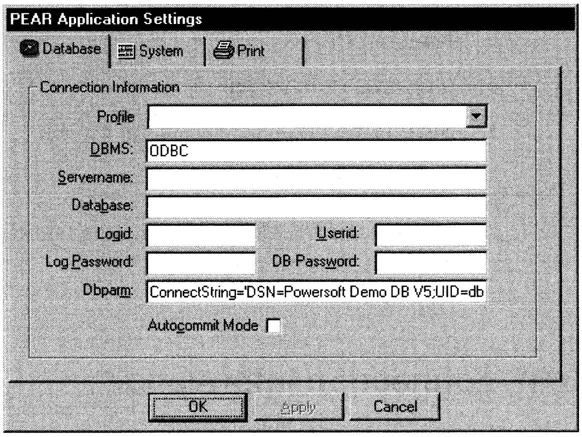
If you start PEAR and a database has not been set in the application settings, the PEAR Application Settings dialog box displays. When you enter new information in these fields and click OK, PEAR updates the application settings (provided the logon is successful).

Specifying application setting before starting Pear
You can also specify application settings from the launcher. For how to do this, see "Specifying application settings" on page 4.

❖ To specify database connection information:

1 Select Options>Settings from the menu bar.

The PEAR Application Settings dialog box displays:



Profile:

DBMS: ODBC

Servername:

Database:

Logid:

Userid:

Log Password:

DB Password:

Dbparm: ConnectString='DSN=Powersoft Demo DB V5;UID=db

Autocommit Mode □

You can specify:

Setting	Meaning
Profile	Name of the profile in the PB.INI file that specifies the database you will use
DBMS	Specific 3-character DBMS identifier
Servername	Server name or connect string, if connecting to a remote DBMS. This value is dependent on the value entered in DBMS above
Database	Name of the database to connect to (for DBMSs that require it)
UserID	Identifier used to connect to the database
Database password	Password used (if needed) to connect to the database
LogID	Identifier used to connect to the database
Logpassword	Log password used to connect to the database
DBParm	DBMS-dependent keyword, typically used for ODBC
AutoCommit	Specifies whether recoverable transaction processing is enabled

2 Make sure that the profile name is correct.

3 Click OK.

Running PEAR

Once you have started PEAR, you see the Table List for Database window where you select one or more tables.

❖ **To run PEAR:**

1 In the Table List for Database window, select a single table by clicking it.
or
Select many tables by clicking them.
or
Select all tables by clicking Select All.

2 (Optional) Add system tables to the list by selecting the System Tables checkbox.

3 Specify whether to view only or automatically print by selecting either the Run & Preview (default) or the Run, View, & Print radio button.

4 Click OK.

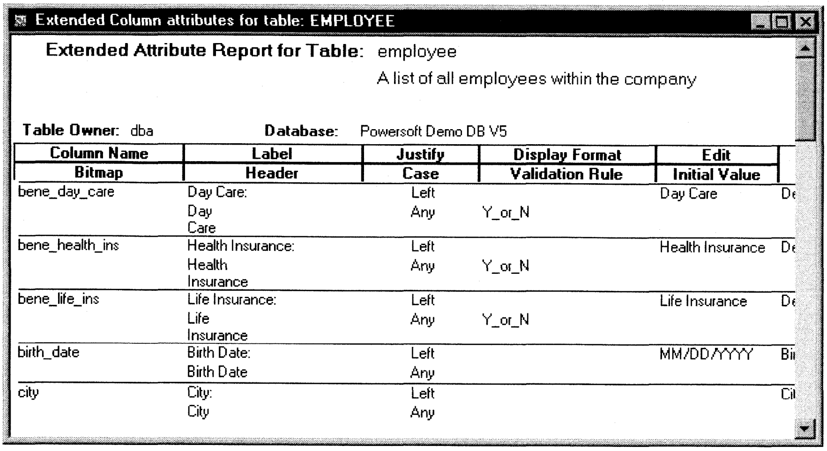
If there is an error retrieving information, you may need to modify the default SQL syntax and then run PEAR again.

FOR INFO For information about modifying the SQL syntax, see "Specifying database parameter defaults" on page 53.

Viewing a PEAR report

What you see

For each table you selected, PEAR displays the Extended Column Attributes window with extended attribute information for each column:



Extended Column attributes for table: EMPLOYEE

Extended Attribute Report for Table: employee

A list of all employees within the company

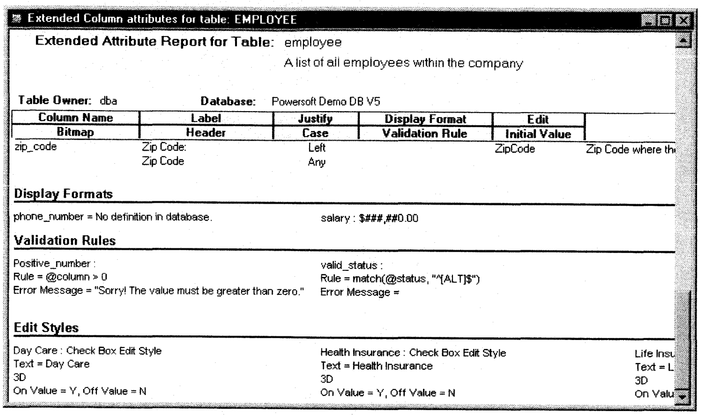
Table Owner: dba **Database:** Powersoft Demo DB V5

Column Name	Label	Justify	Display Format	Edit	
Bitmap	**Header**	**Case**	**Validation Rule**	**Initial Value**	
bene_day_care	Day Care:	Left		Day Care	De
	Day Care	Any	Y_or_N		
bene_health_ins	Health Insurance:	Left		Health Insurance	De
	Health Insurance	Any	Y_or_N		
bene_life_ins	Life Insurance:	Left		Life Insurance	De
	Life Insurance	Any	Y_or_N		
birth_date	Birth Date:	Left		MM/DD/YYYY	Bi
	Birth Date	Any			
city	City:	Left			Ci
	City	Any			

The detail section of the report contains two rows for each column in the table that was analyzed. The rows are divided into these columns:

Column	What it reports
Column name	Name of the column
Bitmap	Bitmap, if defined for the column
Label	Label for use with freeform DataWindows
Header	Heading for use with tabular DataWindows
Justify	Justification (left, right, center)
Case	Case (upper, lower, any)
Display Format	Name of display format, if defined for the column
Validation Rule	Name of validation rule, if defined for the column
Edit	Edit style, if defined for the column
Initial Value	Initial value, if defined for the column
Comments	Comments, if defined for the column

The last section of the report provides a summary of display formats, validation rules, and edit styles:



Extended Column attributes for table: EMPLOYEE

Extended Attribute Report for Table: employee

A list of all employees within the company

Table Owner: dba **Database:** Powersoft Demo DB V5

Column Name	Label	Justify	Display Format	Edit	
Bitmap	**Header**	**Case**	**Validation Rule**	**Initial Value**	
zip_code	Zip Code:	Left		ZipCode	Zip Code where th
	Zip Code	Any			

Display Formats

phone_number = No definition in database.

salary : $###,##0.00

Validation Rules

Positive_number :
Rule = @column > 0
Error Message = "Sorry! The value must be greater than zero."

valid_status :
Rule = match(@status, "^[ALT]$")
Error Message =

Edit Styles

Day Care : Check Box Edit Style
Text = Day Care
3D
On Value = Y, Off Value = N

Health Insurance : Check Box Edit Style
Text = Health Insurance
3D
On Value = Y, Off Value = N

Life Insu
Text = L
3D
On Valu

What you can do

You can print the PEAR report by selecting File>Print from the menu bar.

❖ **To specify the paper size for printing:**

1. Select Options>Settings from the menu bar and click the Print tab.

   The PEAR Application Settings dialog box displays.

2. In the PEAR Application Settings dialog box displays, click the Print tab.

3. Select the paper size.

Print reports before exiting
If you have generated PEAR reports that you want to keep, be sure to print them before exiting. When you exit PEAR, all results are deleted.

Specifying database parameter defaults

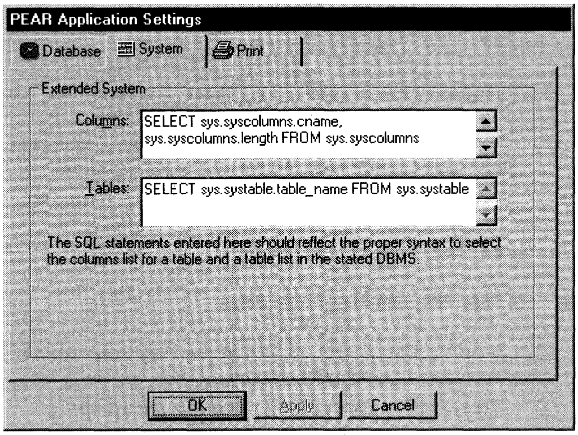
Database parameter defaults contain DBMS-specific SQL SELECT statements used by the Table List for Database window and the Extended Attribute window to access table and column information from the DBMS system tables. In some cases, your DBMS's SQL syntax may be different from the defaults, but you can change the database parameter defaults.

❖ To modify database parameter defaults:

1 Select Options>Settings from the menu bar.

The PEAR Application Settings dialog box displays.

2 Click the System tab:



Columns: SELECT sys.syscolumns.cname, sys.syscolumns.length FROM sys.syscolumns

Tables: SELECT sys.systable.table_name FROM sys.systable

The SQL statements entered here should reflect the proper syntax to select the columns list for a table and a table list in the stated DBMS.

3 Modify SQL statements as needed by typing in the Columns and Tables boxes:

Setting	Contains	PEAR uses this to
Tables	DBMS-specific SQL SELECT statement that returns a list of tables defined to the database. This statement must return one column: the table name	Populate the Tables for Database window
Columns	DBMS-specific SQL SELECT statement that returns a list of columns for a particular table. This statement must return one column: the column name. The statement must also include a retrieval argument for table name and the argument must be *:table_name*	Populate the Extended Attributes window

You can specify additional WHERE criteria to further restrict the tables and columns displayed. For example, you might add restrictions to the SQL= specification to omit DBMS and PowerBuilder system tables.

For example, for Sybase Tables:

```
SELECT sysobjects.name FROM sysobjects WHERE
type = 'U' order by sysobjects.name
```

For example, for Sybase Columns:

```
SELECT syscolumns.name FROM syscolumns WHERE
syscolumns.tbname = :table_name
```

4 Click OK.

CHAPTER 6

Stored Procedure Update

About this chapter

This chapter explains how to use Stored Procedure Update.

Contents

About Stored Procedure Update

Stored Procedure Update generates PowerScript statements you can use to override default DataWindow behavior and update the database through stored procedures.

Your DBMS must support stored procedures
To use Stored Procedure Update, your DBMS must support stored procedures.

Why use Stored Procedure update

Database stored procedures allow you to define procedural SQL statements in a database for use by all applications. Using stored procedures to perform database updates allows you to enhance database security, integrity, and performance. Since stored procedures provide for conditional execution, you can also use them to enforce additional business rules.

What you do You implement database update through stored procedures by:

- Revoking UPDATE, INSERT, and DELETE rights from all users
- Selectively granting stored procedure EXECUTE rights to the appropriate users as stored procedures are created

Although a DataWindow can use stored procedures as a retrieval data source, the DataWindow Update function always updates the database by dynamically generating INSERT, DELETE, and UPDATE SQL statements. This can be an problem when standards require that stored procedures perform all database updates.

PowerBuilder provides the SQLPreview event, which is invoked just before PowerBuilder submits a SQL statement to the database. This event allows you to override the default DataWindow update capability by creating a script that updates the database through a stored procedure.

What Stored Procedure Update does Stored Procedure Update automatically creates PowerScript statements that you can use in a DataWindow's SQLPreview event to perform database updates through stored procedures.

How Stored Procedure Update works

Stored Procedure Update generates PowerScript statements based on information you provide. These statements override default DataWindow database update processing and invoke your stored procedures instead. You then paste or import the PowerScript statements into your DataWindow's SQLPreview event.

Two ways to invoke stored procedures You can use Stored Procedure Update to invoke stored procedures in either of two ways:

- Name stored procedures directly
- Name PowerBuilder user object functions that use remote procedure calls (RPCs) to invoke database stored procedures (such a user object must be a Standard Class user object of type Transaction)

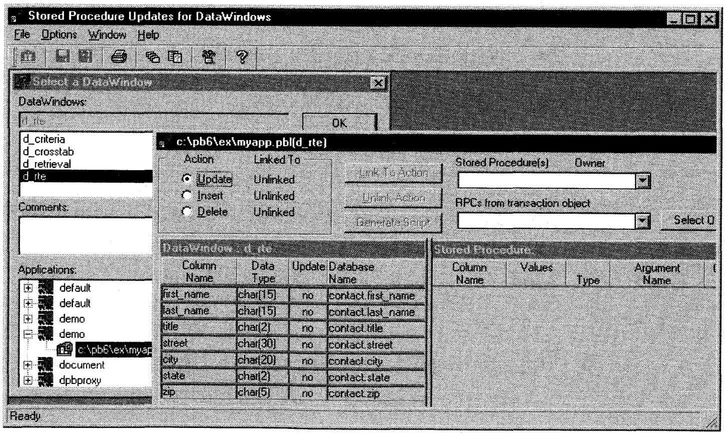
  FOR INFO For more information on implementing remote procedure calls in user objects, see *Application Techniques.*

Recommendations Don't create stored procedures until all DataWindows have been tested completely. This will save you the trouble of revising stored procedures due to DataWindow design changes

Your stored procedures should contain WHERE clauses that name all columns in the table. This will enhance database integrity by preventing lost updates.

Error handling assumptions Stored Procedure Update generates PowerScript code that checks the Transaction object's SQLCODE attribute to determine whether the database update succeeded. Your stored procedure or user object function must use SQLCODE to indicate success or failure. You can do this by using a database function such as Sybase's RAISERROR or by setting the SQLCODE attribute explicitly in the user object function.

The Stored Procedure Update workspace looks like this:



DataWindows: d_rte

d_criteria
d_crosstab
d_retrieval
d_rte

Comments:

Applications:

- default
- default
- demo
- demo
  - c:\pb6\ex\myap
- document
- dpbproxy

c:\pb6\ex\myapp.pbl(d_rte)

Action	Linked To
● Update	Unlinked
○ Insert	Unlinked
○ Delete	Unlinked

Stored Procedure(s) Owner

RPCs from transaction object

DataWindow : d_rte

Column Name	Data Type	Update	Database Name
first_name	char(15)	no	contact.first_name
last_name	char(15)	no	contact.last_name
title	char(2)	no	contact.title
street	char(30)	no	contact.street
city	char(20)	no	contact.city
state	char(2)	no	contact.state
zip	char(5)	no	contact.zip

Stored Procedure

Column Name	Values	Type	Argument Name

Using Stored Procedure Update

This section describes how to use Stored Procedure Update:

- Specifying Stored Procedure Update application settings
- Running Stored Procedure Update
- Updating the DataWindow's SQLPreview event

Starting Stored Procedure Update
For how to start Stored Procedure Update, see "Using the launcher" on page 3.

Specifying Stored Procedure Update application settings

Before you run Stored Procedure Update, you can adjust the default application settings for the following:

- Transaction object named in statements generated by Stored Procedure Update
- Error file used as a template for error handling statements generated by Stored Procedure Update
- Whether to copy PowerScript statements to the Windows clipboard
- Whether to create a file with the PowerScript statements
- RPC object and library
- Connection profile

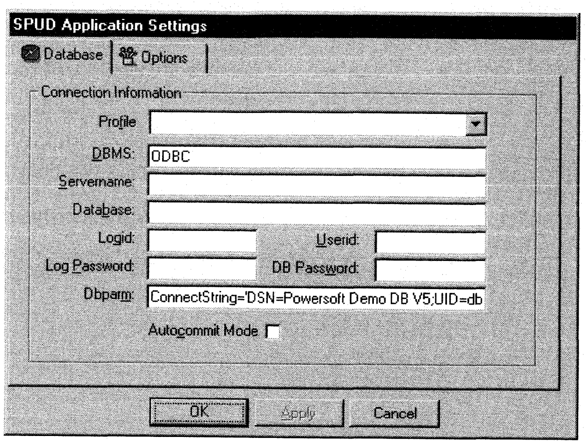
Specifying application setting before starting Stored Procedure Update
You can also specify application settings from the launcher. For how to do this, see "Specifying application settings" on page 4.

❖ To specify Stored Procedure Update settings:

1 Click the Settings button.
or
Select Options>Settings from the menu bar.

The SPUD Application Settings dialog box displays.

2 Click the Database tab to select a database connection profile or other connection information:



SPUD Application Settings

Connection Information

Profile:

DBMS: ODBC

Servername:

Database:

Logid:

Userid:

Log Password:

DB Password:

Dbparm: ConnectString='DSN=Powersoft Demo DB V5;UID=db

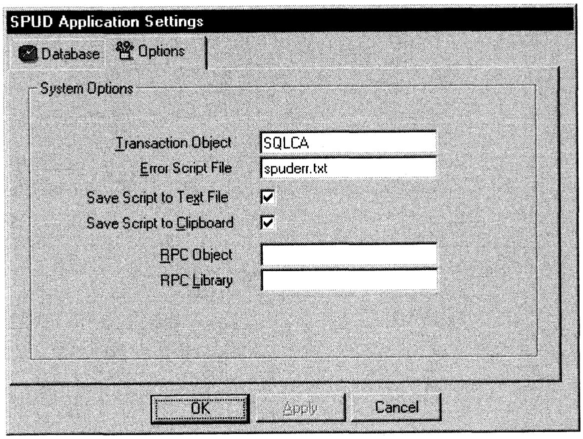
Autocommit Mode ☐

You can specify:

Setting	Meaning
Profile	Name of the profile in the PB.INI file that specifies the database you will use
DBMS	Specific 3-character DBMS identifier
Servername	Server name or connect string, if connecting to a remote DBMS. This value is dependent on the value entered in DBMS above
Database	Name of the database to connect to (for DBMSs that require it)
UserID	Identifier used to connect to the database
Database password	Password used (if needed) to connect to the database
LogID	Identifier used to connect to the database
LogPassword	Log password used to connect to the database
DBParm	DBMS-dependent keyword, typically used for ODBC
AutoCommit	Specifies whether recoverable transaction processing is enabled

3 Click the Options tab to specify:

- The Transaction object that Stored Procedure Update names in the PowerScript statements it generates
- The error filename
- Whether to generate PowerScript statements to the clipboard, to a text file, or both
- The RPC object displayed by default (this is typically the last RPC object used by Stored Procedure Update)
- The RPC library containing the default RPC object



Transaction Object: SQLCA
Error Script File: spuderr.txt
Save Script to Text File ☑
Save Script to Clipboard ☑
RPC Object:
RPC Library:

4 Click OK.

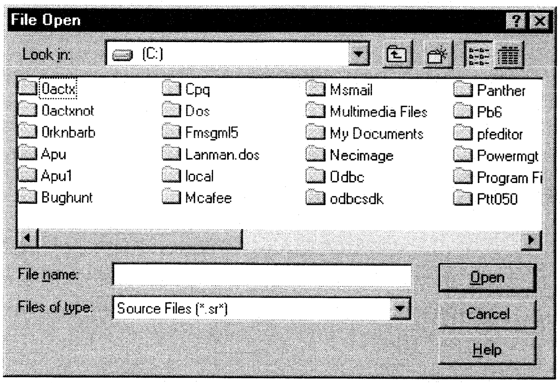
About the error file

The error file named in the Options tab is used by Stored Procedure Update for error handling. You may want to modify the PowerScript error handling statements in the file.

❖ To modify error handling statements:

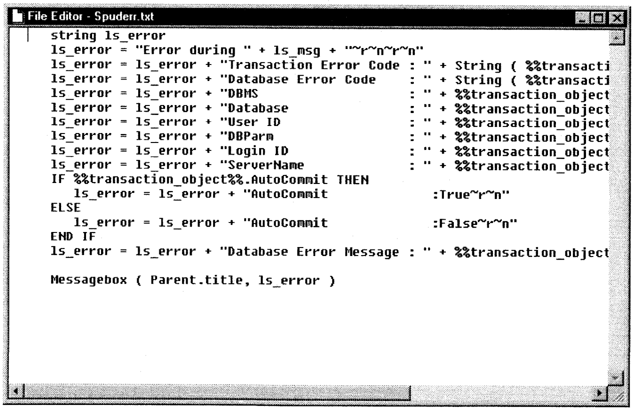
1. If you are in PowerBuilder, access the PowerBuilder File Editor by clicking the Edit button on the PowerBar or by pressing SHIFT+F6 (otherwise, use the Windows Notepad or some other ASCII text editor).

   The File Open dialog box displays:



2. Navigate to the directory containing the Advanced PowerBuilder Utilities, select the SPUDERR.TXT file, and click Open.

   The File Editor displays the requested file:



```
string ls_error
ls_error = "Error during " + ls_msg + "~r~n~r~n"
ls_error = ls_error + "Transaction Error Code : " + String ( %%transacti
ls_error = ls_error + "Database Error Code    : " + String ( %%transacti
ls_error = ls_error + "DBMS                   : " + %%transaction_object
ls_error = ls_error + "Database               : " + %%transaction_object
ls_error = ls_error + "User ID                : " + %%transaction_object
ls_error = ls_error + "DBParm                 : " + %%transaction_object
ls_error = ls_error + "Login ID               : " + %%transaction_object
ls_error = ls_error + "ServerName             : " + %%transaction_object
IF %%transaction_object%%.AutoCommit THEN
   ls_error = ls_error + "AutoCommit              :True~r~n"
ELSE
   ls_error = ls_error + "AutoCommit              :False~r~n"
END IF
ls_error = ls_error + "Database Error Message : " + %%transaction_object

Messagebox ( Parent.title, ls_error )
```

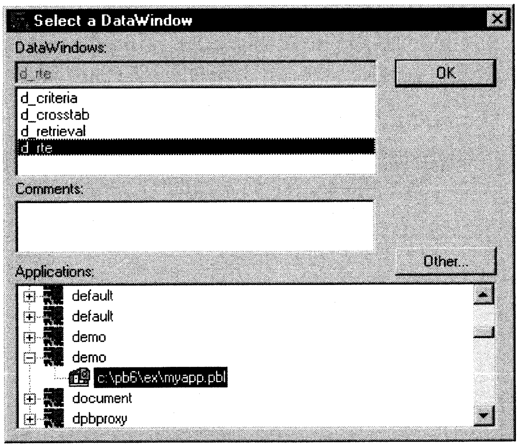
3. Review and modify error-handling statements as appropriate.

Use %%transaction_object%% for flexibility
For statements and functions that require the Transaction object as a qualifier or parameter, you can type **%%transaction_object%%** and Stored Procedure Update will substitute at script creation time with the appropriate Transaction object name, as specified in the Options section.

4 Save the changes.

Running Stored Procedure Update

When you start Stored Procedure Update, the Select A DataWindow dialog box displays, allowing you to select a DataWindow for which to create database update statements. This dialog box lists the applications defined in the PB.INI file:



DataWindows: d_rte

d_criteria
d_crosstab
d_retrieval
d_rte

Applications:

default
default
demo
demo
c:\pb6\ex\myapp.pbl
document
dpbproxy

Selecting a DataWindow

You can select:

- An application, PBL, and DataWindow
- A DataWindow in a PBL not defined in the PB.INI file

If the Applications list is empty Stored Procedure Update uses path information from the registry (Windows 95 and Windows NT) or the APU60.INI file (other platforms) to locate the PB.INI file (PowerBuilder Preferences on the Macintosh; PB.INI on UNIX) and access the list of applications displayed in the Select Application window. If the PB.INI file cannot be found, the Select Application window is empty. Check the location of the PB.INI file and modify path information as needed.

If a "Cannot read from drive" message displays Stored Procedure Update checks to see if each application named in the PB.INI file exists. If this application resides on a drive that is unavailable (such as an empty diskette drive or an unattached network drive), file errors will interrupt the process. Click Cancel to continue the checking process.

❖ **To select an application, PBL, and DataWindow:**

1 Use the Applications TreeView control at the bottom of the window to select the application containing the PBL you want.

2 Select the PBL containing the DataWindow to be analyzed.

3 In the DataWindows listbox, select the DataWindow to be analyzed.

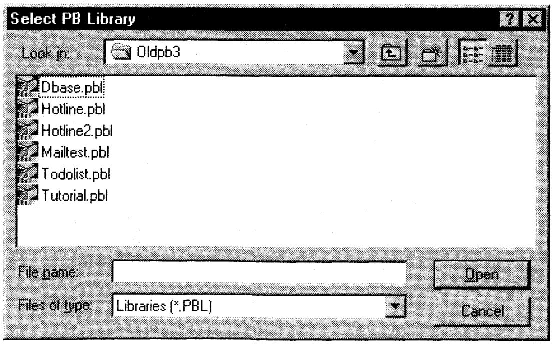
4 Click OK.

❖ **To select a DataWindow from a PBL not defined in PB.INI:**

1 Click the Other button.

The Select PB Library dialog box displays.

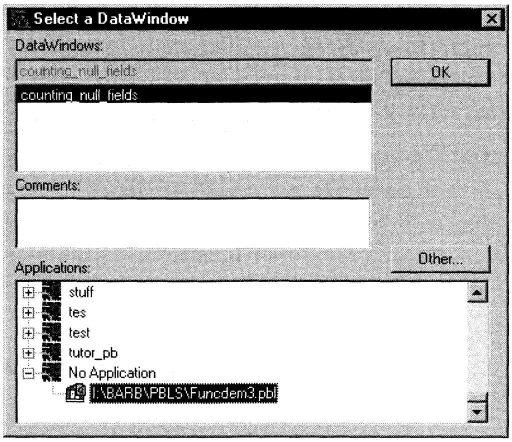
2 Navigate to the drive and directory for the PBL containing the DataWindow to be analyzed:



3 Select the PBL containing the DataWindow to be analyzed.

4 Click Open.

The Select a DataWindow dialog box displays with the selected PBL under the No Application item:


Select a DataWindow

DataWindows:

counting_null_fields

counting_null_fields

Comments:

Applications:

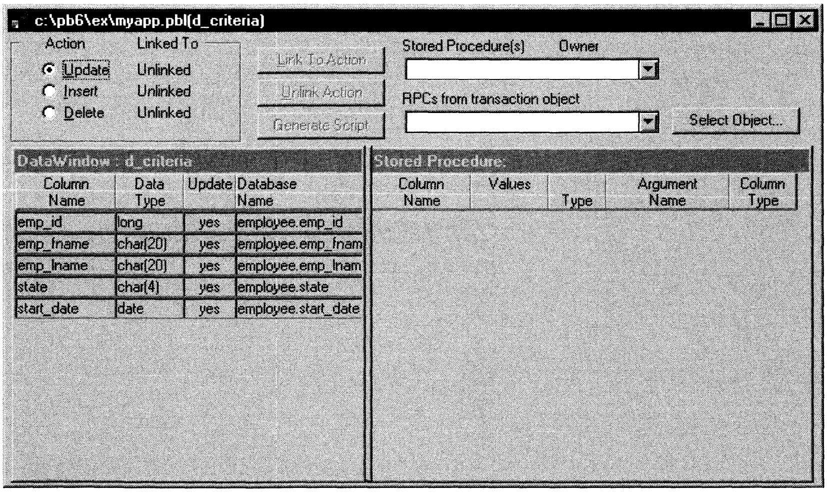
- stuff
- tes
- test
- tutor_pb
- No Application
  - I:\BARB\PBLS\Funcdem3.pbl

5 In the DataWindows listbox, select the DataWindow to be analyzed.

6 Click OK.

Generating PowerScript statements for database update

After you have selected a DataWindow, you see the update criteria window:


c:\pb6\ex\myapp.pbl(d_criteria)

Action	Linked To
● Update	Unlinked
○ Insert	Unlinked
○ Delete	Unlinked

Stored Procedure(s) Owner

RPCs from transaction object

DataWindow : d_criteria

Column Name	Data Type	Update	Database Name
emp_id	long	yes	employee.emp_id
emp_fname	char(20)	yes	employee.emp_fnam
emp_lname	char(20)	yes	employee.emp_lnam
state	char(4)	yes	employee.state
start_date	date	yes	employee.start_date

Stored Procedure:

Column Name	Values	Type	Argument Name	Column Type

You use the update criteria window to:

- Select a database update action (UPDATE, INSERT, DELETE)
- Select a user object containing functions that perform remote procedure calls (optional)
- Select a stored procedure or remote procedure call that performs the specified action

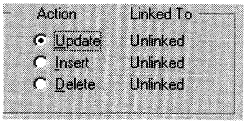
Stored procedures and remote procedure calls (RPCs)
All references to stored procedures apply equally to RPCs.

- Map DataWindow fields to stored procedure arguments
- Link the database update action to the stored procedure using the specified mapping
- Generate PowerScript statements that enable the DataWindow to update the database through stored procedures

❖ To generate PowerScript statements that enable the DataWindow to update the database through stored procedures:

1 If you are using a Transaction object other than SQLCA, select Options>Settings and click the Options tab to modify the Transaction object named in the PowerScript statements, as described in "Specifying Stored Procedure Update application settings" on page 60. If using a nonstandard Transaction object for SQLCA, you can still use SQLCA here.

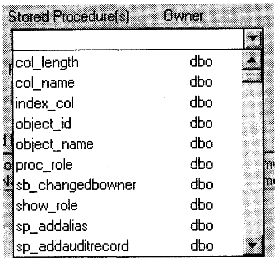
2 Select an action by clicking the radio button you want in the Action box:



Action	Linked To
● Update	Unlinked
○ Insert	Unlinked
○ Delete	Unlinked

3 Click the down arrow next to the Stored Procedure dropdown listbox.

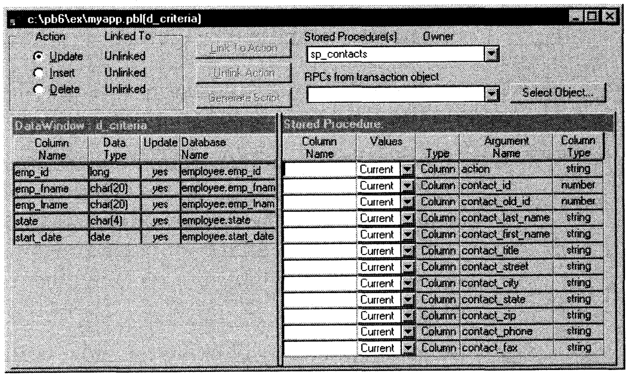
The list displays all stored procedures for the current database:



Stored Procedure(s)	Owner
col_length	dbo
col_name	dbo
index_col	dbo
object_id	dbo
object_name	dbo
proc_role	dbo
sb_changedbowner	dbo
show_role	dbo
sp_addalias	dbo
sp_addauditrecord	dbo

4 Scroll through the list and click on the stored procedure you want to use for the action selected in step 1.

The stored procedure's arguments display in the Stored Procedure box:



c:\pb6\ex\myapp.pbl(d_criteria)

Action	Linked To
● Update	Unlinked
○ Insert	Unlinked
○ Delete	Unlinked

Stored Procedure(s): sp_contacts

Owner:

RPCs from transaction object:

DataWindow : d_criteria

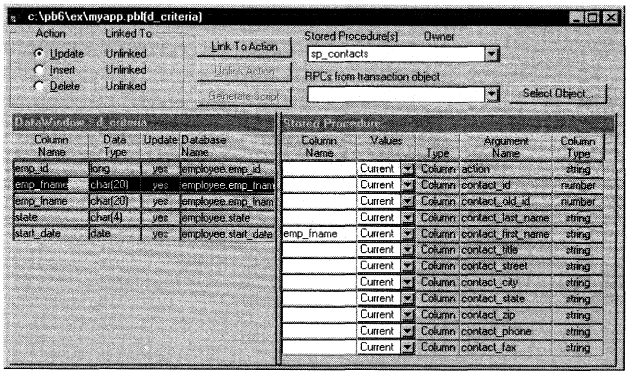
Column Name	Data Type	Update	Database Name
emp_id	long	yes	employee.emp_id
emp_fname	char(20)	yes	employee.emp_fnam
emp_lname	char(20)	yes	employee.emp_lnam
state	char(4)	yes	employee.state
start_date	date	yes	employee.start_date

Stored Procedure

Column Name	Values	Type	Argument Name	Column Type
	Current	Column	action	string
	Current	Column	contact_id	number
	Current	Column	contact_old_id	number
	Current	Column	contact_last_name	string
	Current	Column	contact_first_name	string
	Current	Column	contact_title	string
	Current	Column	contact_street	string
	Current	Column	contact_city	string
	Current	Column	contact_state	string
	Current	Column	contact_zip	string
	Current	Column	contact_phone	string
	Current	Column	contact_fax	string

5 Map DataWindow column names to stored procedure arguments by dragging DataWindow fields and dropping them over the associated stored procedure argument name.

Stored Procedure Update warns you of data type mismatches. If necessary, you can click in the Column Type column and change the stored procedure's data type specification (for example, to map a long to a decimal) to force a match:



c:\pb6\ex\myapp.pbl(d_criteria)

Action	Linked To
● Update	Unlinked
○ Insert	Unlinked
○ Delete	Unlinked

Stored Procedure(s): sp_contacts

RPCs from transaction object:

DataWindow - d_criteria

Column Name	Data Type	Update	Database Name
emp_id	long	yes	employee.emp_id
emp_fname	char(20)	yes	employee.emp_fnam
emp_lname	char(20)	yes	employee.emp_lnam
state	char(4)	yes	employee.state
start_date	date	yes	employee.start_date

Stored Procedure

Column Name	Values	Type	Argument Name	Column Type
	Current	Column	action	string
	Current	Column	contact_id	number
	Current	Column	contact_old_id	number
	Current	Column	contact_last_name	string
emp_fname	Current	Column	contact_first_name	string
	Current	Column	contact_title	string
	Current	Column	contact_street	string
	Current	Column	contact_city	string
	Current	Column	contact_state	string
	Current	Column	contact_zip	string
	Current	Column	contact_phone	string
	Current	Column	contact_fax	string

You can also type directly into the Column Name column. Stored Procedure Update uses the following rules to determine mapping type:

- If the value you type *matches a column name*, Stored Procedure Update uses the column value for argument mapping
- If the value you type *is enclosed in quotation marks*, Stored Procedure Update assumes a literal and uses the literal value for column mapping
- If the value you type *has no quotes and the data type is not numeric*, Stored Procedure Update assumes that it is a PowerScript variable, for which you will provide a value in the SQLPreview event

Stored Procedure Update passes NULL values for arguments for which no mapping is specified.

Original value versus Current value

The Values dropdown listbox displays two items: Current and Original. Current passes the current DataWindow column value to the stored procedure. Original passes the original DataWindow column value. You should choose Original if the column is used in a WHERE clause for UPDATE or DELETE.

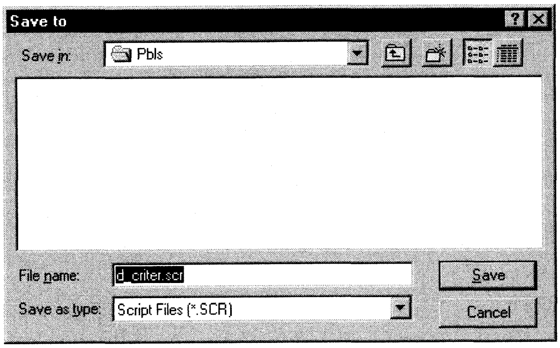
6 Click Link to Action.

7 Repeat steps 1 through 5 for all appropriate actions.

8 Click the Generate Script button.

9 *If you did not link stored procedures to all three actions* using the Update, Insert, and Delete radio buttons, Stored Procedure Update displays a warning message box. Click OK.

If you specified that Stored Procedure Update should save PowerScript statements to a file, the Save To dialog box displays, prompting for a filename:



Save to

Save in: Pbls

File name: d_criter.scr

Save as type: Script Files (*.SCR)

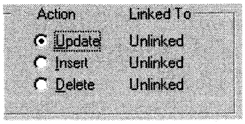
Specify the filename and click Save.

Stored Procedure Update creates PowerScript statements that use the specified stored procedures to update the database and writes them to the selected destinations (clipboard and/or file), as specified in the SPUD Application Settings dialog box.

❖ **To generate PowerScript statements based on RPCs encapsulated in a standard class user object of type transaction:**

1 If you are using a Transaction object other than SQLCA, select Options>Settings and click the Options tab to modify the Transaction object named in the PowerScript statements, as described in "Specifying Stored Procedure Update application settings" on page 60. If using a nonstandard Transaction object for SQLCA, you can still use SQLCA here.

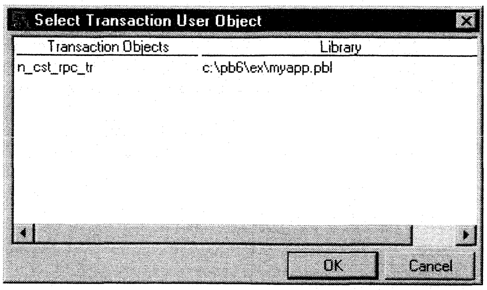
2 Select an action by clicking a radio button in the Action box:



Action	Linked To
● Update	Unlinked
○ Insert	Unlinked
○ Delete	Unlinked

3 Click Select Object.

The Select Transaction User Object dialog box displays all user objects of type transaction:

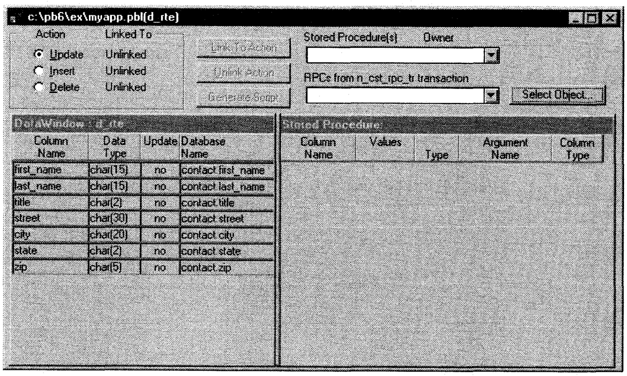


Transaction Objects	Library
n_cst_rpc_tr	c:\pb6\ex\myapp.pbl

4 Select the user object containing remote procedure calls to update your DataWindow.

5 Click OK.

The update criteria window displays and the RPCs from Transaction object dropdown listbox is enabled:



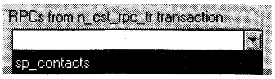
Action	Linked To
● Update	Unlinked
○ Insert	Unlinked
○ Delete	Unlinked

Column Name	Data Type	Update	Database Name
first_name	char(15)	no	contact.first_name
last_name	char(15)	no	contact.last_name
title	char(2)	no	contact.title
street	char(30)	no	contact.street
city	char(20)	no	contact.city
state	char(2)	no	contact.state
zip	char(5)	no	contact.zip

Column Name	Values	Type	Argument Name	Column Type

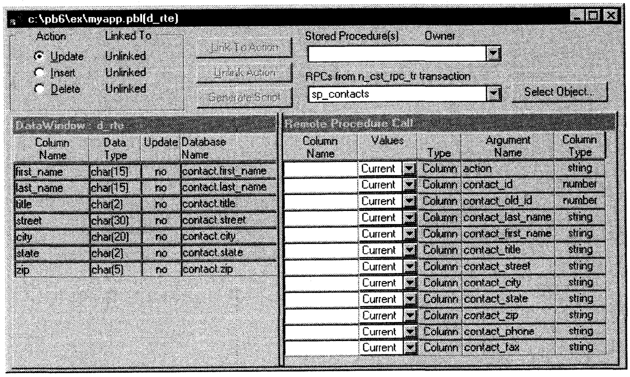
6 Click the down arrow next to the RPCs from Transaction object dropdown listbox.

A dropdown list displays all functions in the selected user object:



7 Scroll through the list and click on the function that executes the stored procedure for the action selected in step 3.

The stored procedure's arguments display in the Remote Procedure Call box:



c:\pb6\ex\myapp.pbl(d_rte)

Action	Linked To
● Update	Unlinked
○ Insert	Unlinked
○ Delete	Unlinked

Stored Procedure(s):

Owner:

RPCs from n_cst_rpc_tr transaction: sp_contacts

DataWindow : d_rte

Column Name	Data Type	Update	Database Name
first_name	char(15)	no	contact.first_name
last_name	char(15)	no	contact.last_name
title	char(2)	no	contact.title
street	char(30)	no	contact.street
city	char(20)	no	contact.city
state	char(2)	no	contact.state
zip	char(5)	no	contact.zip

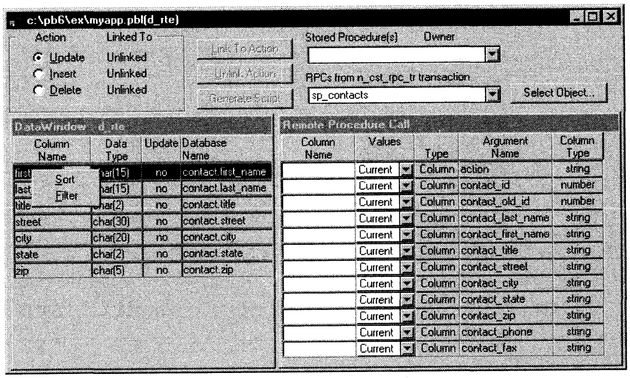
Remote Procedure Call

Column Name	Values	Type	Argument Name	Column Type
	Current	Column	action	string
	Current	Column	contact_id	number
	Current	Column	contact_old_id	number
	Current	Column	contact_last_name	string
	Current	Column	contact_first_name	string
	Current	Column	contact_title	string
	Current	Column	contact_street	string
	Current	Column	contact_city	string
	Current	Column	contact_state	string
	Current	Column	contact_zip	string
	Current	Column	contact_phone	string
	Current	Column	contact_fax	string

8 Continue as described in steps 5 through 9 for the procedure about generating PowerScript statements that update the database through stored procedures.

Sorting and filtering the column display

You can organize the update criteria window by sorting and filtering the column display:



c:\pb6\ex\myapp.pbl(d_rte)

Action	Linked To
● Update	Unlinked
○ Insert	Unlinked
○ Delete	Unlinked

Stored Procedure(s):

Owner:

RPCs from n_cst_rpc_tr transaction: sp_contacts

DataWindow : d_rte

Column Name	Data Type	Update	Database Name
first	ar(15)	no	contact.first_name
last	ar(15)	no	contact.last_name
title	ar(2)	no	contact.title
street	char(30)	no	contact.street
city	char(20)	no	contact.city
state	char(2)	no	contact.state
zip	char(5)	no	contact.zip

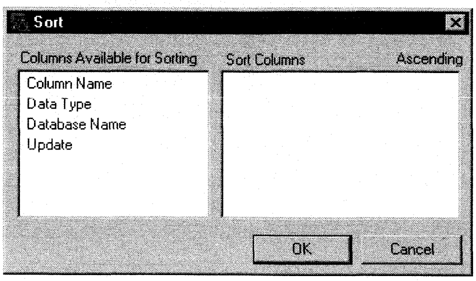
Remote Procedure Call

Column Name	Values	Type	Argument Name	Column Type
	Current	Column	action	string
	Current	Column	contact_id	number
	Current	Column	contact_old_id	number
	Current	Column	contact_last_name	string
	Current	Column	contact_first_name	string
	Current	Column	contact_title	string
	Current	Column	contact_street	string
	Current	Column	contact_city	string
	Current	Column	contact_state	string
	Current	Column	contact_zip	string
	Current	Column	contact_phone	string
	Current	Column	contact_fax	string

❖ To sort the column display:

1 Point to one of the columns, right-click, and select Sort from the popup menu.

The Sort dialog box displays:

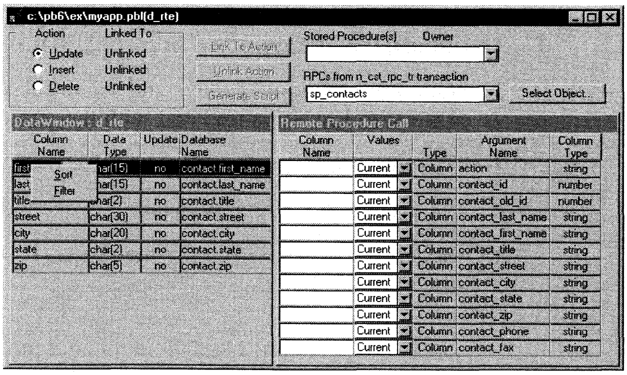


Column Name
Data Type
Database Name
Update

2 Drag columns back and forth to specify the sort columns. Use the Ascending checkbox to specify the sort order (ascending or descending).

❖ To filter the column display:

1 Point to one of the columns, right-click, and select Filter from the popup menu:



● Update Unlinked
○ Insert Unlinked
○ Delete Unlinked

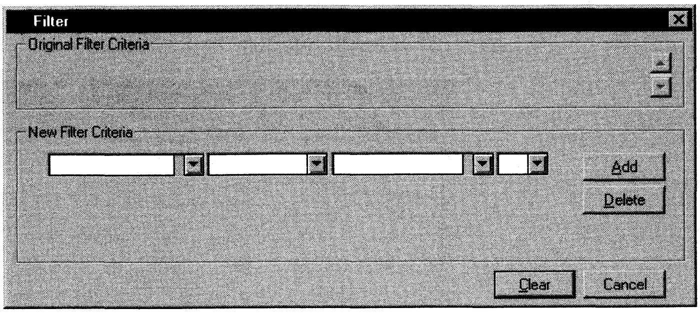
sp_contacts

Column Name	Data Type	Update	Database Name
first	har(15)	no	contact.first_name
last	har(15)	no	contact.last_name
title	har(2)	no	contact.title
street	char(30)	no	contact.street
city	char(20)	no	contact.city
state	char(2)	no	contact.state
zip	char(5)	no	contact.zip

Column Name	Values	Type	Argument Name	Column Type
	Current	Column	action	string
	Current	Column	contact_id	number
	Current	Column	contact_old_id	number
	Current	Column	contact_last_name	string
	Current	Column	contact_first_name	string
	Current	Column	contact_title	string
	Current	Column	contact_street	string
	Current	Column	contact_city	string
	Current	Column	contact_state	string
	Current	Column	contact_zip	string
	Current	Column	contact_phone	string
	Current	Column	contact_fax	string

2 Select Filter from the popup menu.

The Filter dialog box displays:



3 Select the column, an operator, and a value.

Updating the DataWindow's SQLPreview event

You use the PowerScript painter to paste or insert the PowerScript statements that Stored Procedure Update generates into the DataWindow control's SQLPreview event.

❖ **To update the DataWindow control's SQLPreview event:**

1 Open the Window painter and select the window that contains the DataWindow to be updated with the generated PowerScript statements.

2 Display the PowerScript painter for the DataWindow control.

3 Select the SQLPreview event.

4 Select Edit>Paste from the menu bar (if statements were generated to the clipboard).
or
Select File>Import from the menu bar and select the generated script file (if statements were generated to a file).

PowerBuilder inserts the generated PowerScript statements.

5 Review the PowerScript statements and modify them as needed.

CHAPTER 7 Object Search

About this chapter

This chapter explains how to use Object Search.

Contents

About Object Search

Object Search scans a list of PowerBuilder libraries, looking for objects or text within objects.

Why use Object Search

When working with a single application, tracking objects and variables presents no special difficulty. But as you maintain more and more applications, it becomes difficult to track object and variable usage across applications.

Object Search allows you to search through one or more PBLs looking for an object or text string. When you search through a group of PBLs, they do not have to be defined within a single application.

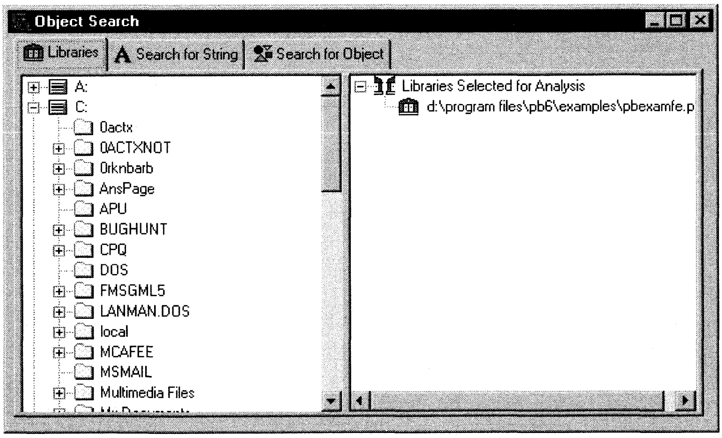
How Object Search works

Object Search examines PowerBuilder objects, searching for either a text string or an object name. You can restrict the search by object type and (for objects) you can also search for either an exact match or a pattern match.

The Object Search utility looks in:

- Every line of source code, including events, object functions, and global functions
- Structures
- Global, instance, and shared variable declarations
- External function declarations

The Object Search workspace looks like this:



Using Object Search

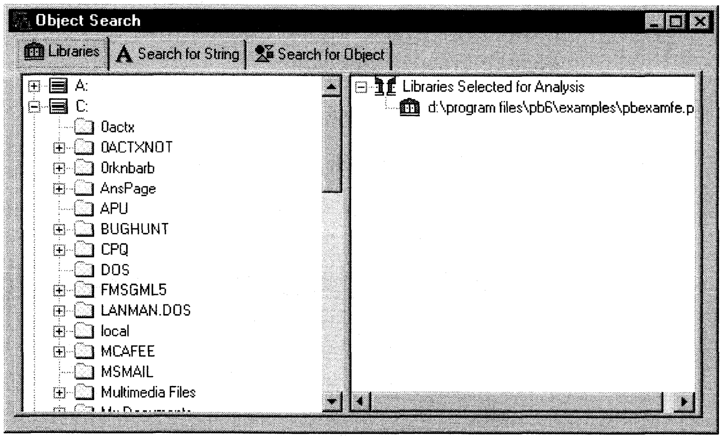
Using Object Search involves these steps:

- Running Object Search
- Viewing the Object Search report
- Populating the library search list

Starting Object Search
For how to start Object Search, see "Using the launcher" on page 3.

Running Object Search

After you have started Object Search, you see the Object Search window:



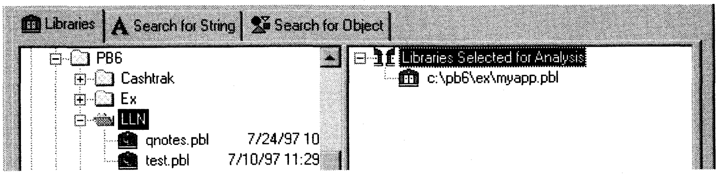
Selecting libraries

You can search for a string or object in one or more libraries.

❖ **To select libraries:**

1. Navigate to the appropriate folder.
2. Select the PowerBuilder libraries you need by:
   - Dragging a library or a folder from the left frame and dropping it in the right frame

- Double-clicking a library or folder to display libraries in the right frame



3 Repeat steps 1 and 2 until you've selected all the libraries you need.

Specifying the library display
You can specify the libraries selected for analysis to display at Object Search startup.

FOR INFO See "Populating the library search list" on page 81.

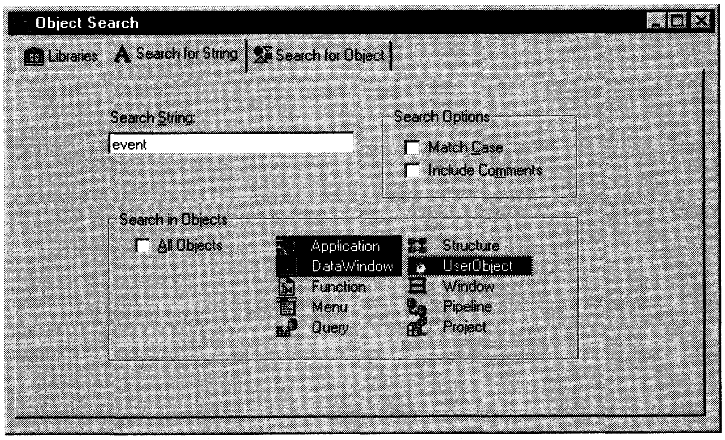
Searching for a string

You can search through objects for a specified string. For example, if you change an instance variable name, you need to know where your PowerScript code references that name.

❖ **To search for a string:**

1 Click the Search for String tab.

2 Type a string in the Search String box, optionally specifying whether to use a case-sensitive search and search comments:



Search String: event

3 Select object types to search in or the All Objects checkbox.

4 Click the Search button.

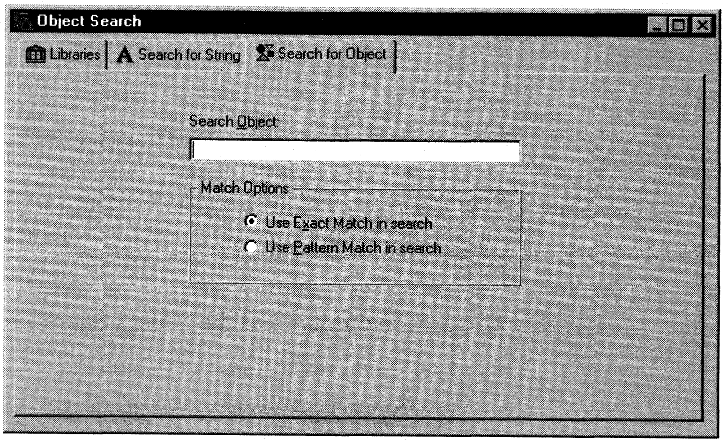
The Object Search utility scans the selected objects in the specified libraries, looking for the specified string. When the Object Search utility finishes searching, it displays the Object Search report.

Searching for an object

You can search by object name. For example, you may need to know which PowerBuilder library contains an object so that you can include it in your application's search path.

❖ **To search for an object:**

1 Click the Search for Object tab:



Search Object:

Match Options

● Use Exact Match in search

○ Use Pattern Match in search

2 Type the object name in the Search Object box.

3 Specify whether Object Search looks for an exact match or a pattern match:

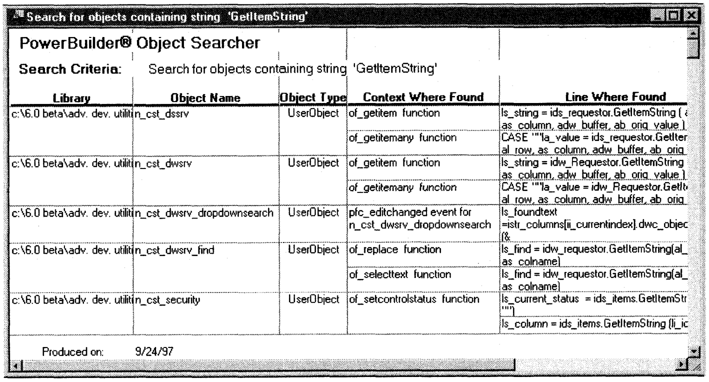
- **Exact match** Object Search displays objects whose name matches exactly

- **Pattern match** Object Search displays objects whose name contains the specified search string

4 Click Search.

Viewing the Object Search report

When Object Search finishes searching for strings or objects, it displays the Object Search report:



PowerBuilder® Object Searcher

Search Criteria: Search for objects containing string 'GetItemString'

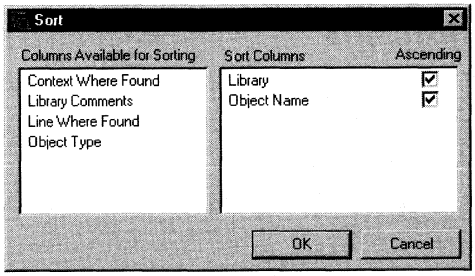
Library	Object Name	Object Type	Context Where Found	Line Where Found
c:\6.0 beta\adv. dev. utiliti	n_cst_dssrv	UserObject	of_getitem function	ls_string = ids_requestor.GetItemString (a as column, adw buffer, ab orig value)
			of_getitemany function	CASE ""la_value = ids_requestor.GetIter al row, as column, adw buffer, ab orig
c:\6.0 beta\adv. dev. utiliti	n_cst_dwsrv	UserObject	of_getitem function	ls_string = idw_Requestor.GetItemString as column, adw buffer, ab orig value)
			of_getitemany function	CASE ""la_value = idw_Requestor.GetIt al row, as column, adw buffer, ab orig
c:\6.0 beta\adv. dev. utiliti	n_cst_dwsrv_dropdownsearch	UserObject	pfc_editchanged event for n_cst_dwsrv_dropdownsearch	ls_foundtext =istr_columns[ii_currentindex].dwc_objec (&
c:\6.0 beta\adv. dev. utiliti	n_cst_dwsrv_find	UserObject	of_replace function	ls_find = idw_requestor.GetItemString(al_ as colname)
			of_selecttext function	ls_find = idw_requestor.GetItemString(al_ as colname)
c:\6.0 beta\adv. dev. utiliti	n_cst_security	UserObject	of_setcontrolstatus function	ls_current_status = ids_items.GetItemStr "")
				ls_column = ids_items.GetItemString (li_ic

Produced on: 9/24/97

Some searches may return a large number of rows. You can create more meaningful output by sorting and filtering the contents of the Object Search report.

❖ To sort the contents of the Object Search report:

1 Click the Sort button on the toolbar:



Columns Available for Sorting:

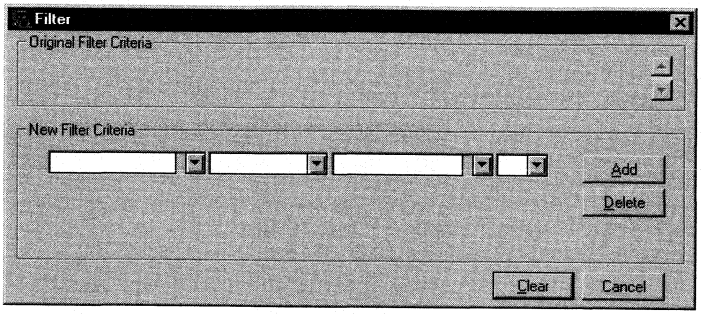
Context Where Found
Library Comments
Line Where Found
Object Type

Sort Columns	Ascending
Library	☑
Object Name	☑

2 Drag columns back and forth to specify the sort columns. Use the Ascending checkbox to specify the sort order (ascending or descending).

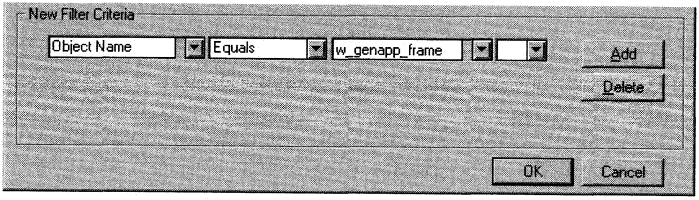
❖ **To filter the contents of the Object Search report:**

1 Click the Filter button on the toolbar:



2 Specify filter options as necessary.

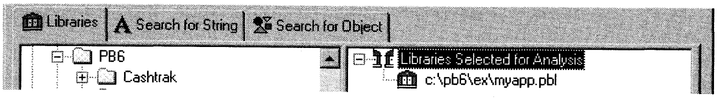
For example, you could use the following specification to display only those rows in w_genapp_frame:



Object Name | Equals | w_genapp_frame

Populating the library search list

You can specify the libraries selected for analysis (in the right-hand frame of the Object Search main window) to display at startup the libraries you last searched in or the libraries for the current PowerBuilder application:



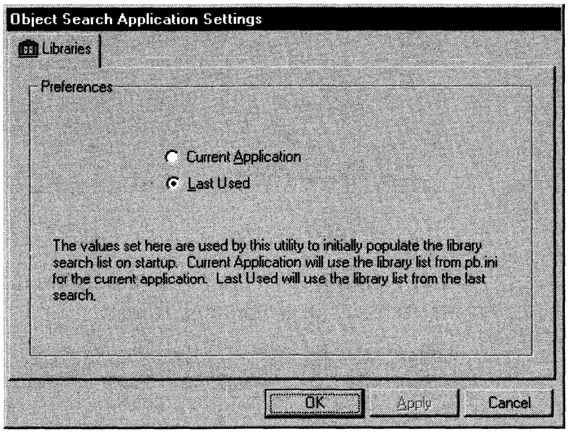
c:\pb6\ex\myapp.pbl

Specifying application setting before starting Object Search
You can also specify application settings from the launcher. For how to do this, see "Specifying application settings" on page 4.

❖ To specify the library search list at Object Search startup:

1 Select Options>Settings from the menu bar.
or
Click the Change Settings button on the toolbar.

The Object Search Applications Settings dialog box displays:



Object Search Application Settings

○ Current Application
● Last Used

The values set here are used by this utility to initially populate the library search list on startup. Current Application will use the library list from pb.ini for the current application. Last Used will use the library list from the last search.

2 Specify how you want the library list to display at startup. You can specify:

Setting	Meaning
Current Application	The Object Search utility displays the libraries specified in the PB.INI file for PowerBuilder's current application
Last Used	The Object Search utility displays the libraries you last searched in

3 Click OK.

4 For your choice to take effect, start the Object Search utility again.

Index