helpsystems

User Guide
Abstract

Contents

Introduction

ABSTRACT is a powerful programming environment from Help/Systems. It includes both documentation and development tools. The tools are integrated into a consistent interface that complies with SAA CUA guidelines. The software runs on all models of the iSeries computer system and is completely independent of all IBM licensed program products.

Through its documentation facilities, ABSTRACT provides a complete, centralized information source for all of your iSeries software. The relationships among the various entities in your applications are kept in an information repository. This data dictionary can be accessed both interactively and through batch programs to give you high level views of the control and data flows within your applications.

The ABSTRACT development environment is designed as a replacement for part of the IBM Programming Development Manager (PDM). Those who are familiar with the features and functions of PDM will find that ABSTRACT offers similar ease of programming features. At the same time, it greatly expands the power of PDM's development concepts through enhancements to user option files, integration with the documentation dictionary, and use of enhanced display station capabilities.

ABSTRACT has many extraordinary ease of use features as well. The primary working displays offer pop-up action bars and pull down menus. They make it easy to learn and remember the product functions. They also offer an easy path for assistance when the ABSTRACT user is uncertain of the functions which will accomplish a particular objective. Of course, advanced users can "short cut" the action bar/pull downs with function keys - even further enhancing their productivity during development.

In short, everything that you need for a complete programming environment is integrated into one consistent, easy to use interface. Once you install ABSTRACT and it analyzes your applications, you will be able to use it as your primary operations base - working from ABSTRACT displays from the time you begin your work until you leave for the day.

Audience

ABSTRACT is designed to assist all of the members of the programming and development team.

Programmers get a facility that enables them to perform edit-compile-debug cycles with a minimum number of transactions - efficiently using all of the available programming utilities and system functions, without the hassles of switching from one environment to another. ABSTRACT also makes it easy for one person to handle several cycles concurrently - overlapping the development of one set of programs with another.

Analysts get a complete, interactive, cross referencing tool that can deliver up-to-date information about application structure, data flows, and object references. Information can be accessed quickly and easily by using a few keystrokes at the workstation.

Software designers get an overview of existing systems so that new applications or enhancements can be integrated with existing modules quickly and easily.

Managers and auditors get full scale documentation that records how a completed application works. File layouts, job stream flowcharts, top-down and bottom up analysis of object usage, etc. can serve as the basis for a permanent library of application documentation that can be kept current and complete with a minimum of effort.

Benefits

There are many significant benefits to using the ABSTRACT development and documentation system. You will realize most of them as soon as you begin using the product, although some of them will become apparent only after you have become familiar with its features and functions.

Once the product is installed and has documented the applications in your environment, you will notice that ABSTRACT makes the job of becoming acquainted with unfamiliar systems much easier. It will create reports and views that give you a broad, conceptual idea of how things work - or the details of specific data elements and program objects. It is a tremendous advantage to anyone who is starting fresh on a given application, or who simply can't remember the specifics about how it all works together.

ABSTRACT will reduce the time you need to find and fix application problems. Through documentation about job stream flows, field usage, and object relationships, you will be able to trace exactly what happens, and where.

You will also witness improved productivity in new development. Significant gains can often be made simply by reusing existing programs and files, adapting them to new situations, providing new functions. Want to know which programs and logical files already exist that affect the customer file? ABSTRACT makes it easy to identify them, review their function, and begin the process of adapting them to new uses.

Through a reliable and consistent source of information, ABSTRACT reports will meet many requirements of your data processing auditors. Most manually produced documentation is incomplete when it is created and out of date shortly thereafter. ABSTRACT makes it easy to maintain a complete library of information and keep it up to date as your applications mature.

Primary Features

Every major ABSTRACT function can be accessed from one of the menus provided with the product. They can also be accessed through CL commands. This makes it easy to tailor your own menus to your specific requirements, and to embed ABSTRACT commands in your own CL programs.

The menu and display interface is simple and straightforward. It conforms to the guidelines defined by the SAA CUA (Common User Access) definition. Once acquainted with the functions of the product, advanced users will find a wealth of customization options that let them shape ABSTRACT into a highly personalized development environment.

Many of the advanced functions are accomplished through iSeries assembly-level MI (machine interface) programming. This unique capability means that ABSTRACT can provide greater function while using system resources as efficiently as possible.

In addition to its other features, ABSTRACT can be set to use the full capabilities of your 27x132 display station. The primary object list displays will use the full width of a *DS4 workstation to provide you with the greatest amount of information possible.

ABSTRACT defaults are maintained on a user by user basis. Each user can specify separate settings for their display characteristics and the default batch job description and job queue. Settings are automatically updated each time the product is used so that the most recently used settings are always available. The most recently used object list is remembered so that it can be automatically recalled when ABSTRACT is used again.

Initialization

The ABSTRACT documentation dictionary is created and maintained through a batch process known as *initialization*. The initialization function can be performed on an entire library or on specific objects - the granularity of each request can be specified when it is made. Each request can be run interactively or submitted to a batch subsystem.

In addition, ABSTRACT makes it easy to keep the documentation for several application sets independent. Whether you want to make a distinction between production and development applications, or to segregate the documentation for each application into different databases, ABSTRACT can accomplish your objective by allowing you to specify a data set name when the initialization request is made. Information in the various data sets will remain independent throughout the analysis, reporting, and development phases of ABSTRACT use.

The initialization process uses a combination of iSeries system functions and ABSTRACT routines to document the objects in your applications. Some of these functions analyze iSeries objects and the information contained in them - files, programs, commands, menus, etc.

Other functions require access to the program source code for your applications. ABSTRACT can determine the source code used to create a program by referencing the service data in the program object description. An initialization option will direct ABSTRACT to locate the source code for the program and automatically analyze it after the object level information has been documented. Once the source code has been located, ABSTRACT can determine the HLL program usage of data fields and program calls that will complete the documentation database. *This way, all programs within the library can be loaded, even if the source used to create them resides elsewhere.*

After the initialization phase is complete, the cross reference database will contain information about:

Database structure

physical/logical relationships
format definitions
field definitions

Flow of control

program-program transfers among CL and HLL programs/modules
user profile initial program
command processing program (CPP) access
menu access to programs and commands
subsystem routing entry programs
Trigger programs

Data flow

file object manipulations
program-file relationships
program-field relationships
database use by SEQUEL views
database use by OS/400 query definitions

Object usage

command parameter references within programs
within job descriptions (user profile, queues, etc.)

Repository Information

binding directories
service programs

This information is used by the analysis, reporting, and development functions of ABSTRACT.

Analysis and Reporting

The analysis functions of ABSTRACT allow you to use the workstation to determine how your software works from three primary points of view:

how and where the transfer of program control occurred
data flows between application programs and the database
object usage within the application

ABSTRACT reporting functions allow you to route the cross reference information to your printer or an output file. You can document specific parts of your application or create an entire library of information that can be saved for archival purposes.

Successful use of the analysis and reporting functions depends heavily on the proper completion of the initialization phase. If performed correctly, the initialization functions will load the documentation cross references with all the necessary information about your software.

Once loaded, the cross references can be accessed by the analysis, reporting, and development tools of ABSTRACT in a bi-directional fashion using either a top-down or a bottom-up approach.

Object usage displays and reports indicate how and where a given object (file, program, field, or system object) is used by the applications in a data set.

Object references displays and reports tell about the references that a particular object (usually a program) makes - the files it accesses, programs it calls, etc.

Development Tools

Perhaps the most powerful feature of ABSTRACT is the use of a PDM-like shell for displaying cross referencing information. With it, you can use all the features of IBM's Programming Development Manager (PDM) for editing source code and compiling and managing objects, in addition to viewing the cross reference information about the application set.

A significant part of the ABSTRACT potential lies in the option file that controls the function of the option codes that can be entered on the primary display. The option file is compatible with the PDM option file, so you can quickly and easily load ABSTRACT with the options you have configured into your current PDM file. Once you have done so a new array of function will be at your fingertips because of two significant enhancements:

ABSTRACT allows each option code to run several CL commands through the addition of a logical line end character. ABSTRACT makes it easy to delete (or create) work files prior to running a program, or to start debug and add breakpoints, or perform a host of other actions all from a single option code.

ABSTRACT provides 3 position alphanumeric option codes rather than the two positions allowed by PDM. Although it may seem trivial at first, you will quickly understand the advantage as you create mnemonic option codes that convey meaning much more clearly than before. In addition, all the option codes (and their description) can be determined by scrolling through a list at the top of the display. You no longer need to leave your working environment to examine the options available to you.

In addition to the PDM-like development environment that accelerates the edit-compile-debug cycle, ABSTRACT also provides two powerful tools to enhance the development process.

Automated program recompiler. You can quickly and easily create several programs from a single source file. You can also create only those programs that make use of a particular file. Once the database changes have been made, you can have ABSTRACT automatically recompile the programs that use the changed objects.

Database recreation facility. Once changes have been made to the DDS source code, you can use this tool to recreate a physical file and all the logical files based on it. Data will be preserved. Member relationships will be recreated exactly as in the existing file structures. It's never been easier to make changes to your database!

Before you begin

If you have read this far, you are well on your way to successfully installing and implementing the ABSTRACT software. The background in the preceding section will serve you well as you try to understand how ABSTRACT can help you in the daily development, analysis, and management of your applications systems.

Where to begin

The ABSTRACT User's Guide consists of seven more chapters that explain the functions of the product in detail.

Part 2, Initialization gives detailed instructions for loading the cross-reference files that ABSTRACT uses.

Part 3, Object Relation Concepts explains the concepts of the ABSTRACT 'object relations' commands. Object relations is a general concept that includes object references (what objects are referenced by a given program for example) and object usage (what objects use a program, file, or field).

Part 4, Object References and Object Usage explains the ABSTRACT commands that show object references and object usage.

Object reference information shows the objects that a given object (usually a program) references. A typical object-reference question would be What files are referenced by this program?

Object usage information shows the objects that make use of a given object. A typical object-usage question would be What programs use this file?.

Part 5, File Analysis describes the ABSTRACT file analysis features and shows how you can work with the members, formats, fields, access paths, and file layouts for the files within your applications.

Part 6, Exception Reports documents the ABSTRACT exception reports. Exception reports describe things that are *not quite right* within your application. For example, ABSTRACT can document all the instances where program source code has been changed since its last compilation into a program object.

Part 7, Programming Tools provides information on how to use the ABSTRACT programming tools for recreating the database and program objects within your applications.

Part 8, User-defined Option Files gives step-by-step instructions on how to use the ABSTRACT option files. The option file lets you customize the ABSTRACT environment to your particular needs - greatly speeding your work.

Document conventions

A number of conventions are used to make the ABSTRACT User's Guide easier to read.

Monospaced text is used to represent text typed exactly as it is seen on the computer. For example,

```
RSTLIB ABSTRACT DEV(*DKT01)
```

Italicized text represents the name of something that you supply as a parameter (**`RSTLIB`** ***`Library`*** **`DEV(`*****`DeviceName`*****`)`**).

Bold text is used for emphasis.

Boxed text is especially important and highlights items that you must pay attention to. Often, boxed text is something that must be done before anything else will work.

How to contact Help/Systems

This manual should help answer most questions that arise about the product's function and use.

If you need additional assistance, we encourage you to contact us directly. We will do our best to answer your questions, provide additional information, or help you solve a problem. Before you contact us, we request that you please have the following information on hand:

- The ABSTRACT User's Guide
- The release and modification level number of the OS/400 operating system loaded on your computer.
- The iSeries model number installed at your location.
- All pertinent documentation describing your problem. This will include items such as the joblog produced when the problem occurs, print keys of displays, report output, etc. A joblog can be produced from an interactive job simply by typing:

  ```
  DSPJOBLOG OUTPUT(*PRINT)
  ```

It is likely that you will be requested to forward the evidence of your problem by fax or e-mail. This is an excellent way for us to see exactly what you are describing and to solve your problem as expeditiously as possible.

Please contact Help/Systems during regular business hours (central time) at:

Help/Systems
6533 Flying Cloud Drive
Suite 200
Eden Prairie MN, IL 55344

Phone (952) 933-0609

Email: chi.support@helpsystems.com

It is usually best to begin your call by requesting assistance from the marketing representative you worked with in acquiring the software. If necessary, your call can be escalated to the technical assistance team along with pertinent details of your installation and support history.

Getting Started

As described above, ABSTRACT combines powerful documentation features, programming tools, and a development shell that makes it easy to perform the daily tasks that you face.

The PDM-like development facility can be used as soon as the software is installed on your system. You can use it in your routine edit-compile-debug cycles as well as in your daily system management operations.

Some analysis and reporting functions will be available as soon as the product is installed on your system. For example, the file analysis features that provide information about file objects (record layouts, database structure, access paths, etc.) do not depend on the cross reference dictionary and will provide useful information even before you run the initialization procedures to load your applications into a documentation data set.

Most of the ABSTRACT documentation functions however do rely on cross-reference information. Before these can give proper information, details of your applications must be loaded into the ABSTRACT data files. Refer to *Part 2, Initialization* for detailed instructions regarding the initialization phase and the types of requests you can make. After you have completed this phase, ABSTRACT will be able to give you an accurate description of how your applications work.

Using ABSTRACT

The Common User Access (CUA) component of IBM's System Application Architecture (SAA) defines interface components that should be the same across all applications. It lays the framework for the function of command function keys (F12 always cancels the current screen) and the appearance of display panels (options should always run underneath the display title) in addition to overall interaction between the user and the software. In this framework, CUA describes two different approaches for the design of applications. They are known as the *action-object* and *object-action* orientationY of an application.

An action-object orientation forces the user to choose an action and then decide what object to work on. Many iSeries applications use this action-object orientation. An initial menu is presented with a list of actions that can be performed, such as create, delete, modify, etc. After the user selects an action, they specify the object(s) they wish to work with through another display panel or prompt.

The object-action approach (used by ABSTRACT and PDM) allows the user to select a list of objects that will be manipulated, then choose any action that applies to object(s) in the list. The CUA guidelines recommend the use of this orientation wherever possible for a number of reasons, but primarily because it avoids *action modes*.

If the user wants to copy and then rename the same object, an action-object approach forces the user to select the copy action, enter the object name, exit the copy action, select the rename option, and enter the same object name again! The copy option and the rename option are examples of action modes. An interface that forces a user to constantly change action modes while working with the same objects interferes with the natural flow of the user's work. By comparison, in an object-action model the user can select the object, then choose the copy and rename options at the same time. The end result is that the user is able to perform the desired actions more quickly and naturally. Users spend less time thinking about how the interface works and more time being productive.

There are a number of other reasons that IBM encourages the development of object-action oriented applications. See the SAA: Common User Access Basic Interface Design Guide for more details.

Because of this, ABSTRACT was designed as an object-action application. The major cross-referencing commands all present 'object lists' and allow the user to carry out actions upon them. For this reason, they are often referred to as 'object list' commands.

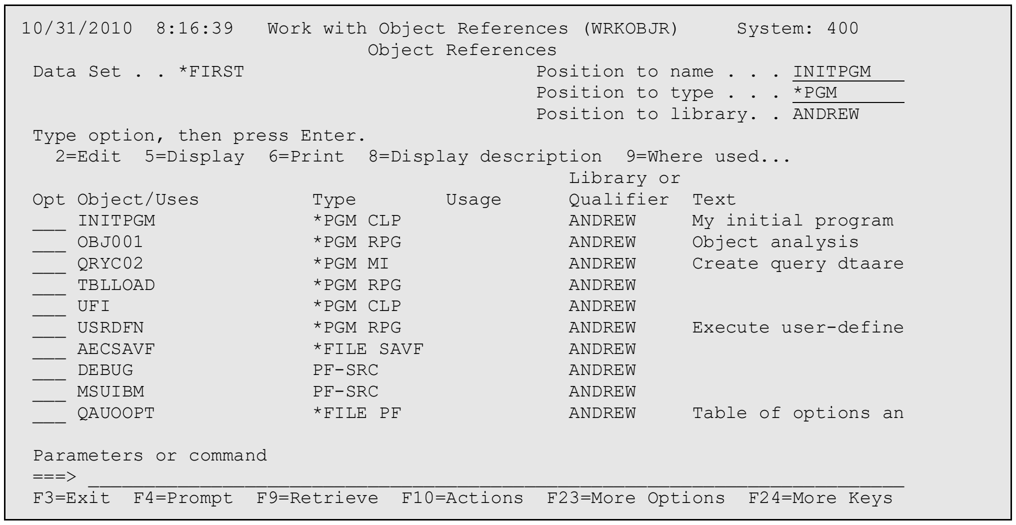
Overview

Just as the Work With Objects Using PDM (WRKOBJPDM) command is used to start the Program Development Manager, ABSTRACT has ‘work with’ commands that present object lists and provide easy access to ABSTRACT functions. Consider the examples below to learn how ABSTRACT can work with the objects in your application libraries.

By typing:

```
WRKOBJR LIB(ANDREW)
```

our sample user has asked ABSTRACT to display all information for the objects in library ANDREW. Any cross reference details about the references made by the objects in the ANDREW library can also be displayed in the list. For the sake of simplicity, the example display lists only the objects in the library, no relations are shown. This would also appear if the ANDREW library had never been loaded into the cross references through the ABSTRACT initialization procedures.



```
10/31/2010  8:16:39   Work with Object References (WRKOBJR)     System: 400
                               Object References
 Data Set . . *FIRST                          Position to name . . . INITPGM
                                              Position to type . . . *PGM
                                              Position to library. . ANDREW
 Type option, then press Enter.
   2=Edit  5=Display  6=Print  8=Display description  9=Where used...
                                                 Library or
 Opt Object/Uses          Type        Usage      Qualifier  Text
 ___ INITPGM              *PGM CLP               ANDREW     My initial program
 ___ OBJ001               *PGM RPG               ANDREW     Object analysis
 ___ QRYC02               *PGM MI                ANDREW     Create query dtaare
 ___ TBLLOAD              *PGM RPG               ANDREW
 ___ UFI                  *PGM CLP               ANDREW
 ___ USRDFN               *PGM RPG               ANDREW     Execute user-define
 ___ AECSAVF              *FILE SAVF             ANDREW
 ___ DEBUG                PF-SRC                 ANDREW
 ___ MSUIBM               PF-SRC                 ANDREW
 ___ QAUOOPT              *FILE PF               ANDREW     Table of options an

 Parameters or command
 ===> ________________________________________________________________________
 F3=Exit  F4=Prompt  F9=Retrieve  F10=Actions  F23=More Options  F24=More Keys
```

Every object in the list can be acted upon by typing an option number next to it. The available options and how to control them are described in more detail in *Part 8, User-Defined Option Files*.

Since ABSTRACT uses the object-action approach, every product command (and OS/400 system command) is available from any of the work with displays. Press F23 to scroll through the available options.

For example, option 10 can be used to load cross-reference data for an individual object - simply by entering the option number and pressing the Enter key. The LOADXREF command that option 10 executes is described in more detail in *Part 2, Initialization*.

The ABSTRACT ‘where-used’ command, Work With X-Ref Usage (WRKOBJUX), can be run by simply typing option 9 next to the object name for which the information is required. The object usage and object reference commands are described in *Part 4, Object References and Object Usage*.

Having a wide range of options (and function keys) available to you can sometimes be confusing, especially if you are unfamiliar with all the things you can do. ABSTRACT makes it easy for you to learn and use the product by providing three outstanding facilities:

Scrollable lists of options and function keys

Action bar and pull-down menus

Context sensitive and hypertext assisted help information

The next display is similar to the one above, except that the option and function key lines have been scrolled and a pull down menu (with an overlaying dialog window) has overlaid the original information.

```
: File   View   Options   ........................................          :
:........................ :                                      : .........:
Data S : 1 1. Subset list :      Subset Object References        :  INITPGM
       :                  :                                      :  *PGM
       : 1 1. Show parent : Object library. . ANDREW             :  ANDREW
Type o :   2. Show relati : Object name . . . *ALL               :
  12=W :                  : Object type . . . *ALL               :
       : 1 1. Show usage  : Object attribute  *ALL               :
Opt Ob :   *. Show full o :                                      :
___ IN :   3. Show full d : Object references to level . .  0    : al program
___ OB :                  :   Files  . . . . . . . . . . .  Y    : nalysis
___ QR : 2 1. Use 132 col :   Formats  . . . . . . . . . .  Y    : uery dtaare
___ TB :   2. Use 80 colu :   Fields . . . . . . . . . . .  Y    :
___ UF :                  :   Members  . . . . . . . . . .  Y    :
___ US : 2 1. Full screen :   Subroutines  . . . . . . . .  Y    : user-define
___ AE :   2. Show option :   Programs . . . . . . . . . .  Y    :
___ DE :................. :   Other object types . . . . .  Y    :
___ MSUIBM                : Show unique relations  . . . .  Y    :
___ QAUOOPT               : Reference explosion level  . .  0    :  options an
                          : References with usage  *ALL          :
Parameters or command     :                                      :
===> ____________________ : F12=Cancel                           :
F5=Refresh  F7=Previous O :......................................:
```

Using the object-action approach makes it easy to perform your normal activities without ever leaving the object list display. Simply choose the objects you want to work with and then go to work! *Part 3, Object Relations Concepts* explains the functions of the list displays.

ABSTRACT Menu

Although the object-action approach of the object list commands is the most powerful and efficient way to use ABSTRACT, you may feel more comfortable (especially at the beginning) by using a menu based action-object oriented approach.

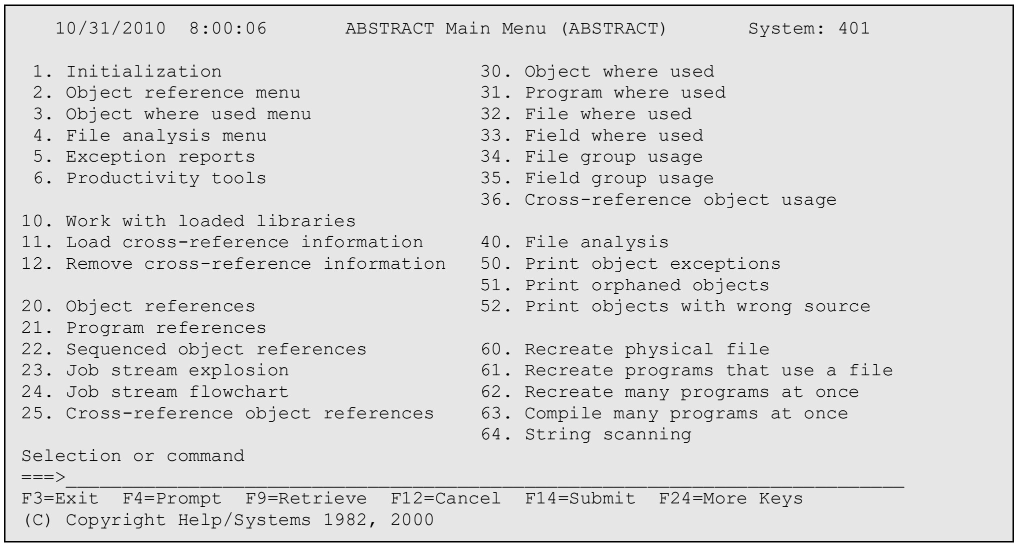
You may want to take a few minutes right now and step through the ABSTRACT menus. This will give you an opportunity to make a quick review of the functions available through the product. If the product has been installed successfully, sign on to an iSeries workstation and type the command:

```
ABSTRACT/ABSTRACT
```

The primary ABSTRACT menu should appear and look like the one below. You can access the menu from any command line simply by typing the ABSTRACT command (as you just did) or by typing the command:

```
GO ABSTRACT/ABSTRACT
```

Other ABSTRACT menus can be accessed using the GO command and specifying the menu name. If the ABSTRACT library is not on your library list, you must qualify the menu name with the ABSTRACT library in order for the GO command to be able to find it.


```
   10/31/2010  8:00:06       ABSTRACT Main Menu (ABSTRACT)       System: 401

 1. Initialization                         30. Object where used
 2. Object reference menu                  31. Program where used
 3. Object where used menu                 32. File where used
 4. File analysis menu                     33. Field where used
 5. Exception reports                      34. File group usage
 6. Productivity tools                     35. Field group usage
                                           36. Cross-reference object usage
10. Work with loaded libraries
11. Load cross-reference information       40. File analysis
12. Remove cross-reference information     50. Print object exceptions
                                           51. Print orphaned objects
20. Object references                      52. Print objects with wrong source
21. Program references
22. Sequenced object references            60. Recreate physical file
23. Job stream explosion                   61. Recreate programs that use a file
24. Job stream flowchart                   62. Recreate many programs at once
25. Cross-reference object references      63. Compile many programs at once
                                           64. String scanning
Selection or command
===>_______________________________________________________________________________
F3=Exit  F4=Prompt  F9=Retrieve  F12=Cancel  F14=Submit  F24=More Keys
(C) Copyright Help/Systems 1982, 2000
```

The ABSTRACT main menu is shown above. Once the menu appears on your display station, you can use it to enter OS/400 commands and to access ABSTRACT functions.

ABSTRACT automatically adds the ABSTRACT library to your library list; once you have entered any one of the ABSTRACT menus, all of the ABSTRACT commands will be available to you, no library name qualification is necessary.

Menu selections and commands

Every menu within ABSTRACT looks and acts in a manner consistent with other OS/400 menus. The date, time, and system name appear at the top of the display on either side of the menu name.

Menu options appear in the middle of the display. The description of the option is listed next to its option number, and the command that is run by the option (or its menu name) is listed to the right. Any option that has no command listed to its right uses the command listed above it. The same command is not listed multiple times.

To run a menu option, type its option number on the selection line at the bottom of the display and press the Enter key. When the option has completed its function, this menu will be redisplayed.

In addition to selecting options, you may also run any valid OS/400 command by typing the command on the selection line and pressing the Enter key.

Function Keys

Each ABSTRACT menu also provides a consistent set of function keys. These function keys also work on the 'object list' displays described in chapters 2 and 3. Notice that ABSTRACT provides several enhancements (cf. F14, F19) to the function key set provided by OS/400 system menus.

F3	Exit this ABSTRACT function
F4	Prompt the option or command that is on the command line
F9	Retrieve 'backwards' through previously executed commands
F12	Exit the current display and return to the previous function

F14 Submit the option or command on the command line to batch. The default job description defined for this user will be used

F15 Prompt the Submit Job (SBMJOB) command before submitting the option or the command on the command line

F16 Work with options in the user option file

F18 Change the default job description and user option file defined for the current user

F19 Retrieve ‘forwards’ through previously executed commands

F24 Display more function keys

Menu driver files

The ABSTRACT menu handling program runs the commands that are stored in a source member tied to the menu name. The MNUCMD file in the ABSTRACT library contains the source members for the ABSTRACT menus.

Source member format includes the title of the menu and the option number and command to be run for each option of the menu. Refer to the following example.

```
5738PW1 V2R1M1 920327  SEU SOURCE LISTING  03/14/92 13:01:49    PAGE  1

 SOURCE FILE . . . . . . . ABSTRACT/MNUCMD
 MEMBER . . . . . . . . . APINZQQ

 SEQNBR ....+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ......... ...+... 2
    1 APINZQQ,1
   100 0001 ?LOADXREF OBJ(*N/*ALL) MBR(*BOTH)             /*SBM*/
   101 0002 WRKLIBX                          /*INT*/
   200 0003 ?LOADXREF OBJ(*N/*ALL) ?*TYPE(*FILE) ?*MBR(*NONE)     /*SBM*/
   301 0004 ?LOADCMD
   302 0005 WRKCMDTRKD                        /*INT*/
   303 0006 ?LOADXREF OBJ(*N/*ALL) ?*TYPE(*PGM) MBR(*OBJ)       /*SBM*/
   500 0007 ?LOADMENU                         /*SBM*/
   501 0008 ?LOADVIEW                         /*SBM*/
   502 0009 ?LOADQRY
   503 0010 ?LOADXREF OBJ(*N/QRPGSRC) ?*TYPE(*SRCF) MBR(*ALL)     /*SBM*/
   900 0011 ?RMVXREF                         /*SBM*/
  1000 0012 RGZXREF                          /*INT*/
```

This source member controls the execution of the initialization menu APINZ. The command that each option will run is listed next to the option number. Selective prompting requests are allowed. Refer to the CL Programmer’s Guide for complete information about the selective prompting characters.

Positions 114-120 of each option record can specify a comment that indicates whether the command will be restricted to interactive execution (/*INT*/) or whether the Run in Batch flag (function key 18) controls when it will be submitted (/*SBM*/) to the batch job queue. The menu display will reflect the status of the option by appending in batch to the option description if it will be submitted. Function key 14 of the menu will always submit commands to the batch job queue unless they have an /*INT*/ indication in the control record.

You can customize the ABSTRACT menus to your environment by changing the command definitions, selective prompting values, and/or their default interactive or batch environment. Use SEU to change the appropriate menu source member. Once the member is updated, the menu will work in accordance with the changes you have made.

ABSTRACT Help Facility

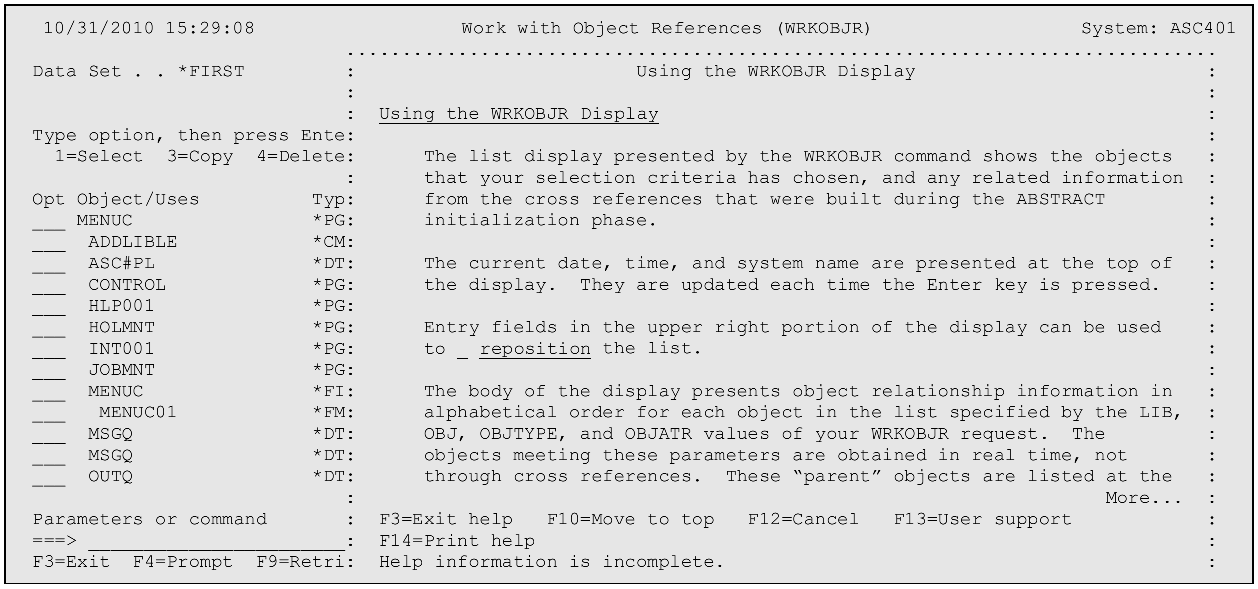
On-line assistance is available from each ABSTRACT display through the Help key on your keyboard. Text is presented using the User Interface Manager (UIM) capabilities of the iSeries. It is handled in the same way as the help text for other system facilities.

ABSTRACT help text is context sensitive. The assistance you receive is related to the location of the cursor at the time you pressed the help key. For best results, you should move the cursor on top of the item in question before you press the help key. Move the cursor to the top of the display if you want comprehensive assistance for the display. Move the cursor to the last line on the display if you want assistance covering the function keys.

Many definitions and in-depth topics can be accessed using the hypertext capabilities of the help system. With this feature, you can jump from a current help panel to a related one that provides additional information and then return to the previous panel or jump to another related topic.

You can search for topical information using the index search capabilities of the system as well. Once you press a function key to access the search index, you can browse information in a topical sequence, or locate the specific help you need by specifying keywords that describe the topic.

The panel below shows an example of the assistance.



```
 10/31/2010 15:29:08                  Work with Object References (WRKOBJR)              System: ASC401
                            ........................................................................
Data Set . . *FIRST         :                         Using the WRKOBJR Display                      :
                            :                                                                        :
                            :  Using the WRKOBJR Display                                             :
Type option, then press Ente:                                                                        :
  1=Select  3=Copy  4=Delete:      The list display presented by the WRKOBJR command shows the objects   :
                            :      that your selection criteria has chosen, and any related information  :
Opt Object/Uses          Typ:      from the cross references that were built during the ABSTRACT         :
___ MENUC               *PG:      initialization phase.                                                 :
___  ADDLIBLE           *CM:                                                                            :
___  ASC#PL             *DT:      The current date, time, and system name are presented at the top of   :
___  CONTROL            *PG:      the display.  They are updated each time the Enter key is pressed.    :
___  HLP001             *PG:                                                                            :
___  HOLMNT             *PG:      Entry fields in the upper right portion of the display can be used    :
___  INT001             *PG:      to _ reposition the list.                                             :
___  JOBMNT             *PG:                                                                            :
___  MENUC              *FI:      The body of the display presents object relationship information in   :
___   MENUC01           *FM:      alphabetical order for each object in the list specified by the LIB,  :
___  MSGQ               *DT:      OBJ, OBJTYPE, and OBJATR values of your WRKOBJR request.  The         :
___  MSGQ               *DT:      objects meeting these parameters are obtained in real time, not       :
___  OUTQ               *DT:      through cross references.  These "parent" objects are listed at the   :
                            :                                                               More...  :
Parameters or command       :  F3=Exit help   F10=Move to top   F12=Cancel   F13=User support          :
===> _______________________:  F14=Print help                                                        :
F3=Exit  F4=Prompt  F9=Retri:  Help information is incomplete.                                       :
```

The help key was pressed after the cursor was moved to the top of the underlying Work With Object References display. The text describes the display and how it can be used.

Once the help panel is displayed, you can navigate through the text using several different facilities.

Depending on your user profile, the help text may appear in a window overlaying your display. Use function key 20 to expand the text to a full 24x80 window. Press function key 20 again to return the text to a smaller window on your display.

Access additional pages of help text by pressing the roll keys. You can position the help window to a specific line by moving the cursor to it and pressing function key 10. The line of help text under the cursor will move to the top of the display.

You can print the help topic by pressing function key 14.

Additional information about using the help facility can be acquired by using function key 13. A series of panels describing the procedures that will allow you to get the most out of the help text will be presented to you.

Once you are finished with the help text, press function key 12 to return to the previous display (which may be another help panel) or use function key 3 to exit the help facility altogether and return to the underlying ABSTRACT display.

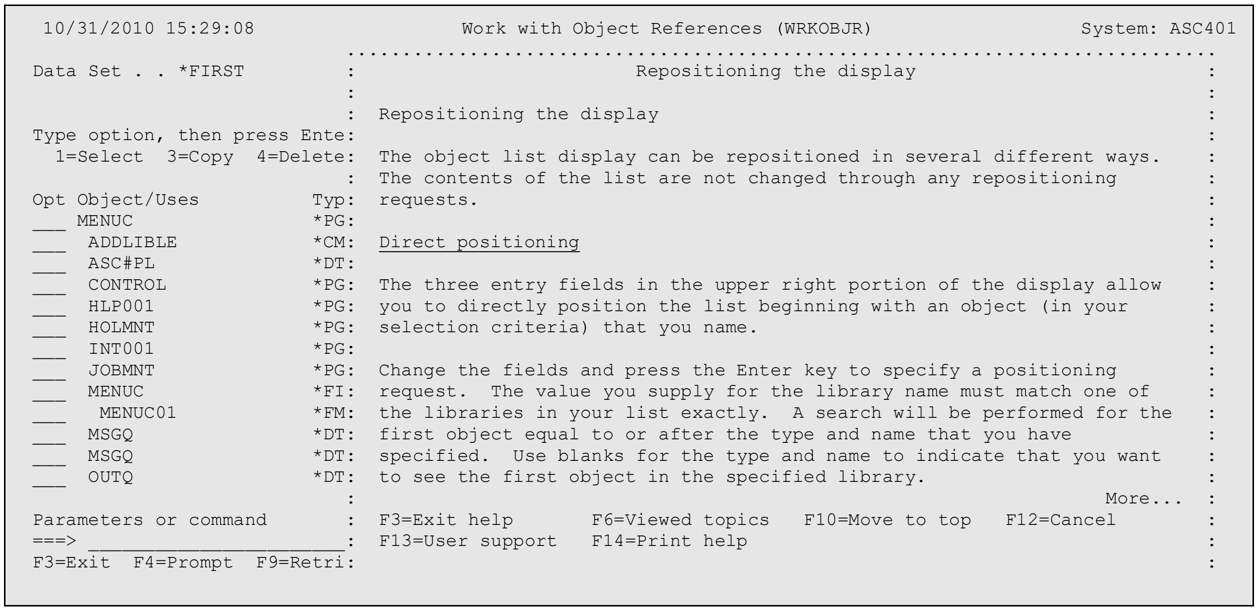
Hypertext

Hypertext selections in the help text will be highlighted, underlined and have an entry field preceding them. The word reposition on the previous display is an example of a hypertext entry. Hypertext makes it easy to navigate through the help text. You can select topics that interest you without wasting time with information you already know.

Activate a hypertext selection by moving the cursor prior to the hypertext keyword (use the TAB key) and pressing Enter. The help topic you selected will be displayed on your workstation. When you are finished with it, you can return to the initial topic by pressing F12.

If you choose several hypertext topic(s) after one another, you can return to previous ones by pressing function key 12 in succession and backing up to them. Alternatively, you can press function key 6 to view your current hypertext path and return to any point on the path.

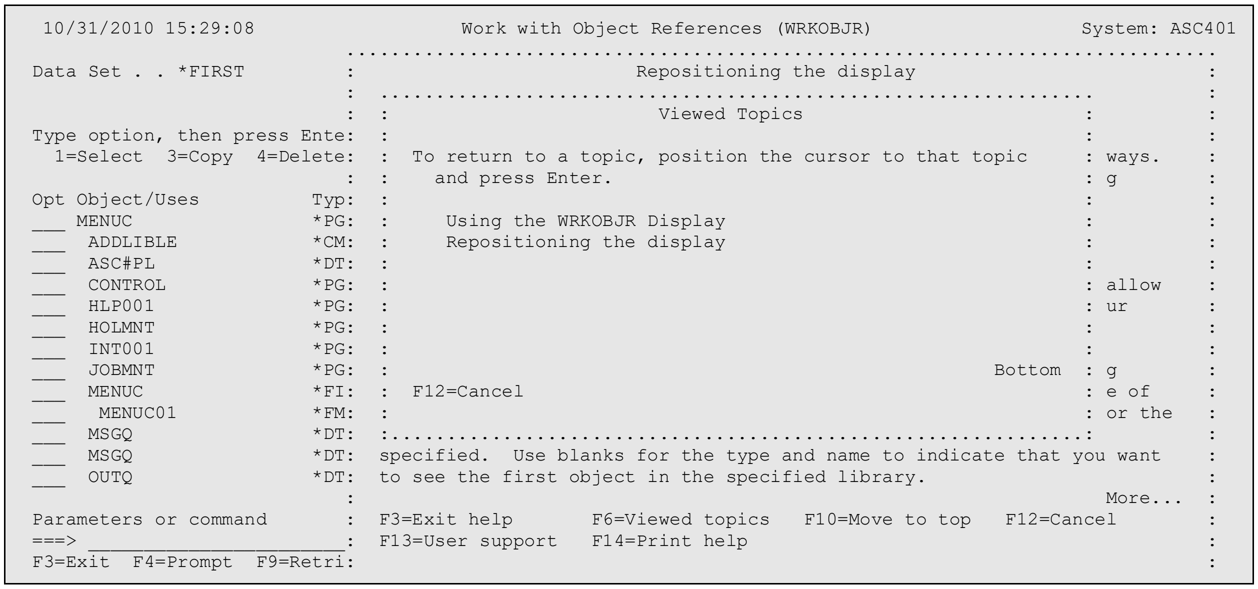
The display on the next page was accessed by moving the cursor to the reposition hypertext field on the previous display and pressing the Enter key. The help facility gives additional information about repositioning the display.



```
 10/31/2010 15:29:08                  Work with Object References (WRKOBJR)               System: ASC401
                            ............................................................................
Data Set . . *FIRST         :                        Repositioning the display                       :
                            :                                                                        :
                            :  Repositioning the display                                             :
Type option, then press Ente:                                                                        :
  1=Select  3=Copy  4=Delete:  The object list display can be repositioned in several different ways.    :
                            :  The contents of the list are not changed through any repositioning      :
Opt Object/Uses          Typ:  requests.                                                             :
___ MENUC               *PG:                                                                        :
___  ADDLIBLE           *CM:  Direct positioning                                                    :
___  ASC#PL             *DT:                                                                        :
___  CONTROL            *PG:  The three entry fields in the upper right portion of the display allow   :
___  HLP001             *PG:  you to directly position the list beginning with an object (in your      :
___  HOLMNT             *PG:  selection criteria) that you name.                                    :
___  INT001             *PG:                                                                        :
___  JOBMNT             *PG:  Change the fields and press the Enter key to specify a positioning       :
___  MENUC              *FI:  request.  The value you supply for the library name must match one of    :
___   MENUC01           *FM:  the libraries in your list exactly.  A search will be performed for the  :
___  MSGQ               *DT:  first object equal to or after the type and name that you have          :
___  MSGQ               *DT:  specified.  Use blanks for the type and name to indicate that you want   :
___  OUTQ               *DT:  to see the first object in the specified library.                     :
                            :                                                             More...  :
Parameters or command       :  F3=Exit help       F6=Viewed topics   F10=Move to top   F12=Cancel      :
===> _______________________:  F13=User support   F14=Print help                                    :
F3=Exit  F4=Prompt  F9=Retri:                                                                        :
```

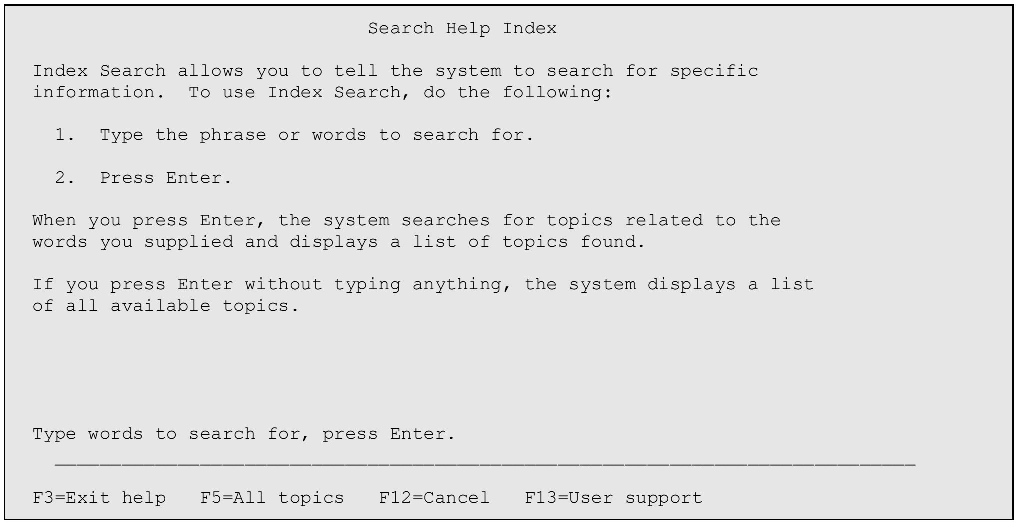
The next display illustrates the hypertext path thus far. (F6 was pressed from the display above.) It shows two entries: the initial topic and the hypertext topic accessed with the reposition entry.

After selecting F6 to view your current path, you can return to any display along the path by moving the cursor alongside the topic and pressing the Enter key. Press function key 12 to return to the help panel you just left. (In this case, the reposition panel)

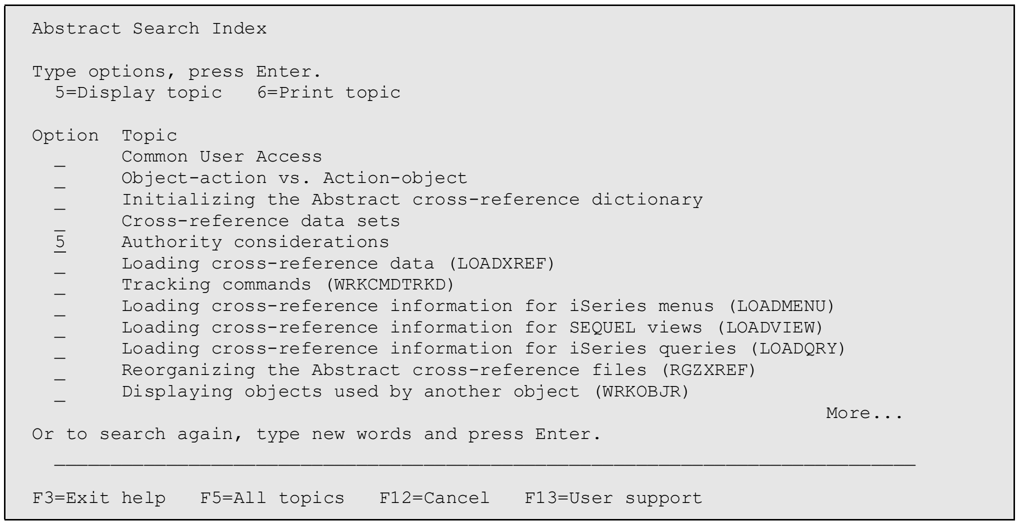

```
 10/31/2010 15:29:08                  Work with Object References (WRKOBJR)              System: ASC401
                            ..............................................................................
Data Set . . *FIRST         :                         Repositioning the display                      :
                            :  ...............................................................         :
                            :  :                           Viewed Topics                      :         :
Type option, then press Ente:  :                                                              :         :
  1=Select  3=Copy  4=Delete:  :  To return to a topic, position the cursor to that topic     : ways.   :
                            :  :    and press Enter.                                          : g       :
Opt Object/Uses          Typ:  :                                                              :         :
___ MENUC                *PG:  :     Using the WRKOBJR Display                                :         :
___  ADDLIBLE            *CM:  :     Repositioning the display                                :         :
___  ASC#PL              *DT:  :                                                              :         :
___  CONTROL             *PG:  :                                                              : allow   :
___  HLP001              *PG:  :                                                              : ur      :
___  HOLMNT              *PG:  :                                                              :         :
___  INT001              *PG:  :                                                              :         :
___  JOBMNT              *PG:  :                                                      Bottom  : g       :
___  MENUC               *FI:  :  F12=Cancel                                                  : e of    :
___   MENUC01            *FM:  :                                                              : or the  :
___  MSGQ                *DT:  :..............................................................:         :
___  MSGQ                *DT:  specified.  Use blanks for the type and name to indicate that you want    :
___  OUTQ                *DT:  to see the first object in the specified library.                         :
                            :                                                              More...  :
Parameters or command       :  F3=Exit help       F6=Viewed topics   F10=Move to top   F12=Cancel       :
===> _______________________:  F13=User support   F14=Print help                                        :
F3=Exit  F4=Prompt  F9=Retri:                                                                          :
```

Index search

The Index search facility of the Help system makes it easy to find and learn about an ABSTRACT topic. If you select option 4 from the pull-down Help menu or press F11 from any ABSTRACT help display, the search index display will appear. You can search for topics based on keywords that you specify.

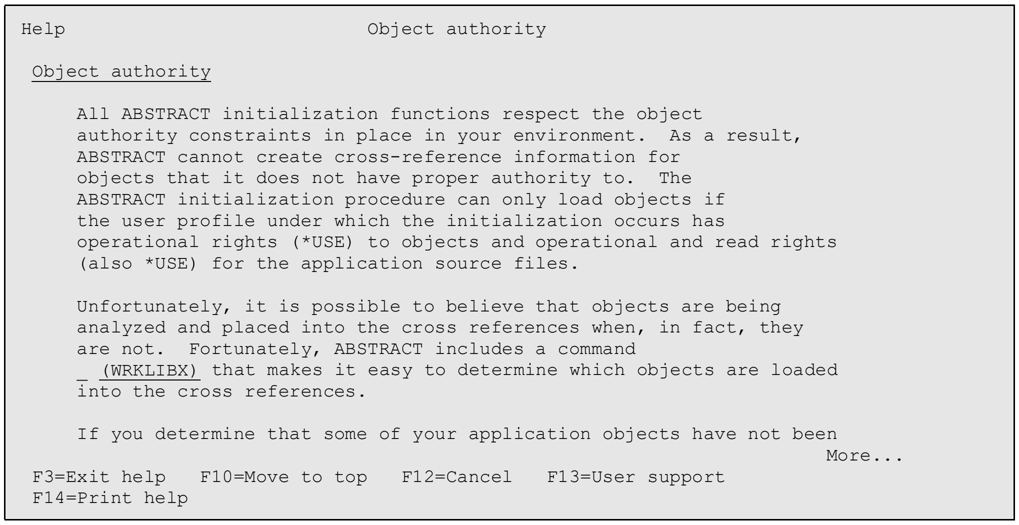

```
                              Search Help Index

  Index Search allows you to tell the system to search for specific
  information.  To use Index Search, do the following:

    1.  Type the phrase or words to search for.

    2.  Press Enter.

  When you press Enter, the system searches for topics related to the
  words you supplied and displays a list of topics found.

  If you press Enter without typing anything, the system displays a list
  of all available topics.




  Type words to search for, press Enter.
    ____________________________________________________________________________

  F3=Exit help   F5=All topics   F12=Cancel   F13=User support
```

The display tells how to use the search facility. Press Enter without specifying a topic or use F5 to get a complete list of all the topics available. If you make a request, a list of topics meeting your criteria will appear. Use it to choose the specific topic(s) you are interested in.



```
 Abstract Search Index

 Type options, press Enter.
   5=Display topic   6=Print topic

 Option  Topic
   _     Common User Access
   _     Object-action vs. Action-object
   _     Initializing the Abstract cross-reference dictionary
   _     Cross-reference data sets
   5     Authority considerations
   _     Loading cross-reference data (LOADXREF)
   _     Tracking commands (WRKCMDTRKD)
   _     Loading cross-reference information for iSeries menus (LOADMENU)
   _     Loading cross-reference information for SEQUEL views (LOADVIEW)
   _     Loading cross-reference information for iSeries queries (LOADQRY)
   _     Reorganizing the Abstract cross-reference files (RGZXREF)
   _     Displaying objects used by another object (WRKOBJR)
                                                                    More...
 Or to search again, type new words and press Enter.
   ____________________________________________________________________________

 F3=Exit help   F5=All topics   F12=Cancel   F13=User support
```

Display or print topics in the list by placing a ‘5’ or ‘6’ next to them and pressing the Enter key. You can perform another search by entering new keywords into the search line as you did before. Selected topics will appear on the display like this:



```
Help                           Object authority

 Object authority

     All ABSTRACT initialization functions respect the object
     authority constraints in place in your environment.  As a result,
     ABSTRACT cannot create cross-reference information for
     objects that it does not have proper authority to.  The
     ABSTRACT initialization procedure can only load objects if
     the user profile under which the initialization occurs has
     operational rights (*USE) to objects and operational and read rights
     (also *USE) for the application source files.

     Unfortunately, it is possible to believe that objects are being
     analyzed and placed into the cross references when, in fact, they
     are not.  Fortunately, ABSTRACT includes a command
     _ (WRKLIBX) that makes it easy to determine which objects are loaded
     into the cross references.

     If you determine that some of your application objects have not been
                                                                    More...
 F3=Exit help   F10=Move to top   F12=Cancel   F13=User support
 F14=Print help
```

Hypertext topics and additional function keys are available as before. Exit the Help facility and return to the ABSTRACT display by pressing function key 3 or by repeatedly pressing function key 12 to back up through the help panels you have visited.

More information about the OS/400 UIM help facility can be obtained by pressing function key 13 and accessing the extended help available for user support.

Although the ABSTRACT help facility is an advanced method to assist you while using the product, it shouldn’t be regarded as a replacement for the complete documentation that you will find in this User’s Guide. The best way to learn the product is by reference to this manual. The help features provide an exceptional way of reviewing ABSTRACT features or of obtaining assistance for a particular display or function.

The Next Step

For your convenience, the remaining chapters in this manual correspond to an option on the main menu. The functions of the menu are completely described by the corresponding chapter in the manual. This makes it easy to get acquainted with ABSTRACT, since you can progress through the User's Guide chapter by chapter as you work through the options available on each menu.

Part 2, Object Relation Concepts, covers concepts that apply to both options 2 and 3 of the main menu. *Part 3, Object References and Object Usage*, details the actual commands that options 2 and 3 run. All other option numbers directly correspond to a chapter in this user's guide.

Now begin with the next chapter, *Initialization*, for a description of the process you use to load information about your applications into the ABSTRACT database.

Initialization

Many of the ABSTRACT analysis and reporting functions rely on information stored in the cross reference files that comprise the documentation database. In order for these functions to give proper information, details of your application software must be correctly loaded into its data files. This process, called *initialization*, is accomplished by using the Load Cross Reference (LOADXREF) command.

This chapter will discuss the initialization process and describe the features and options of the LOADXREF command. This section presents background and concepts that you need to know and describes the function of the APINZ initialization menu. Subsequent sections detail the function of each command that affects the cross reference dictionary.

The initialization process uses a combination of iSeries system functions and ABSTRACT routines to document the objects in your applications. Some of these functions analyze iSeries objects and the information contained in them — files, programs, commands, menus, etc.

With the introduction of ILE (Integrated Language Environment), three new object types can also be analyzed: service programs, modules, and binding directories. A module object is actually an intermediate step in the creation of an ILE program. References in this manual to programs imply both program and module objects.

Other initialization functions require access to the program source code for your applications. ABSTRACT can determine the source code used to create a program by referencing the service data in the program object description. An initialization option can direct ABSTRACT to locate the source code for the program and automatically analyze it after the object level information has been processed. Even if source code has been moved since compiling objects, ABSTRACT will attempt to find the related source member by searching the list of libraries loaded in the cross reference. Once the source code has been located, ABSTRACT can determine the HLL program usage of data fields and program calls that will complete the documentation database. ILE source analysis occurs for single module programs when the program is loaded into the cross reference. For multi-module programs, source analysis takes place when each module is loaded into the cross reference.

Depending on your development procedures, the object service data may not indicate the correct library/file name for the source code corresponding to a given program object. Refer to the Change Service Data (CHGSVCDTA) command in Part 7 for instructions to correct this problem. It is also possible that the source member type does not correctly reflect the language of a source member. The Change SEU Type (CHGTYPE) command, also described in Part 7, can correct this problem quickly and easily.

Once you have identified an application that you wish to include in the ABSTRACT dictionary, you will most likely want to load all the information about it into the cross references. This is the usual way that you will use the product, and making a load request for the complete information is quick and easy with the LOADXREF command.

After the dictionary is loaded, you will want to keep it current by periodically refreshing it. One of the easiest ways to do this is to run the LOADXREF command through a job scheduler on a daily or weekly basis. The command can be tailored so that only objects that have changed since the last time you did a LOADXREF will be analyzed. Each time objects are analyzed, any existing references for an object are removed from the dictionary and new ones are added.

Database content

The complete initialization process involves many individual steps. When the LOADXREF command is used and TYPE(*ALL) is specified, these steps occur:

- Load database relationships
- Load file descriptions
- Cross reference physical files and trigger programs
- Cross reference commands and their CPPs
- Load program-file relationships
- Analyze CL program objects (refer to source if ALWRTVSRC(*NO))
- Load service programs
- Load module objects
- Load binding directories
- Load job descriptions (LOADJOBD)
- Analyze menu objects (LOADMENU)
- Analyze OS/400 query definitions (LOADQRY)
- Analyze subsystem descriptions (LOADSBSD)
- Analyze SEQUEL views (LOADVIEW)
- Locate HLL source and determine field usage and HLL CALLs

In addition to the steps above, the LOADUSRPRF command can be used to analyze the user profile definitions on your system. Other analysis commands (LOADJOBSCD, LOADMNUMGR) extract application relationships from other applications such as Help/Systems' ROBOT job scheduler.

> Successful field usage cross referencing (based on HLL source code) depends on accurate database and program relationship information. It is important to initialize any libraries containing database files that are used in a cross-application fashion before the other libraries within the application.

Due to its comprehensive nature, you can expect an initialization request that operates on an entire application library to require an extended amount of time to complete. It is quite possible that the initialization request may take several hours to load the hundreds or thousands of objects that might comprise your application.

Once the initialization phase has completed, the cross reference database will contain information about:

- Database structure
    - physical/logical relationships
    - format definitions
    - field definitions

- Flow of control
    - program-program transfers among CL and HLL programs
    - user profile initial program
    - command processing program (CPP) access
    - menu access to programs and commands
    - subsystem routing entry programs
- Data flow
    - file object manipulations
    - program-file relationships

program-field relationships
database use by SEQUEL views
database use by OS/400 query definitions

Object usage
command parameter references within programs
within job descriptions (user profile, queues, etc.)

The cross reference information noted above is contained in the following ABSTRACT files:

DSPDBR	File-file cross reference detailing the structure of the application database
FILEFLD	File-Field cross reference describing record layouts
PGMREF	Program-file cross reference showing file usage by program
OBJREF	Program-program cross reference detailing transfers of control within the application
FLDUPD, INTDICT, CBLDICT	Program-field cross reference which describes the usage of database fields by individual programs
CMDOVR, CMDFMT, CMDDBF	Database command usage cross reference describing use of specific commands by the application
CPYXRF	Copybook cross reference catalogs use of the COPY facility of RPG and COBOL compilers to include source from other members
SUBXRF	Subroutine cross reference contains names and use of each RPG subroutine

Cross reference data sets

ABSTRACT makes it easy to keep the documentation for several application sets independent. Whether you want to make a distinction between production and development applications, or to segregate the documentation for each application into different databases, ABSTRACT can accomplish your objective by allowing you to specify a data set name when the initialization request is made. Information in the various data sets will remain independent throughout the analysis, reporting, and development phases of ABSTRACT use.

The ABSTRACT commands default to a data set named *FIRST. This reserved name identifies the base data set used by the product. Additional data sets can be created by using the Add Data Set (ADDDTASET) command described at the end of this chapter. The current data set names can be listed by prompting one of the ABSTRACT commands (WRKOBJR, CLRDTASET, etc.) and entering a ‘?’into the DTASET parameter.

Empty data sets are automatically removed during reorganization. See the Reorganize Cross Reference (RGZXREF) command for more information.

Object authority

All ABSTRACT initialization functions respect the object authority constraints in place in your environment. As a result, ABSTRACT cannot create cross reference information for objects that it does not have proper authority to. The ABSTRACT initialization procedure can only load objects if the user profile under which the initialization occurs has operational rights (*USE) to objects and operational and read rights (also *USE) for the application source files.

Unfortunately, it is possible to *believe* that objects are being analyzed and placed into the cross references when, in fact, they are not. Fortunately, ABSTRACT includes a command (WRKLIBX) that makes it easy to determine which objects are loaded into the cross references.

> One of the easiest methods to avoid accidental exclusion of application objects from the cross reference dictionary is to run the initialization procedure from the security officer (QSECOFR) user profile.

If you determine that some of your application objects have not been placed into the cross reference files, you can take one of two actions. You can have someone with proper authority grant the necessary rights for the objects which are currently restricted to the user profile that will be running the initialization procedure. Alternatively, you can run the initialization procedure under a different user profile; one having the necessary authority to the application objects and source code.

Initialization options

If you wish, the initialization functions can be tailored to load information into the cross references in very specific ways. Depending on the type of initialization request, only a few or perhaps several thousand records may be added to the documentation database. The granularity of the initialization process is entirely up to you.

Objects can be loaded individually or generically. All objects in a library can be loaded with a single command, or several commands can be run to load certain types of objects within a given library.

Each time the initialization functions are executed, the system date is placed into the cross reference. This makes it possible to reload the cross reference files based upon a particular date or to reload all objects which have changed since they were originally loaded.

The date referencing options on the LOADXREF command allow only those objects created since a given date to be loaded. Alternatively, cross reference information can be updated for objects which have changed since the last time they were analyzed.

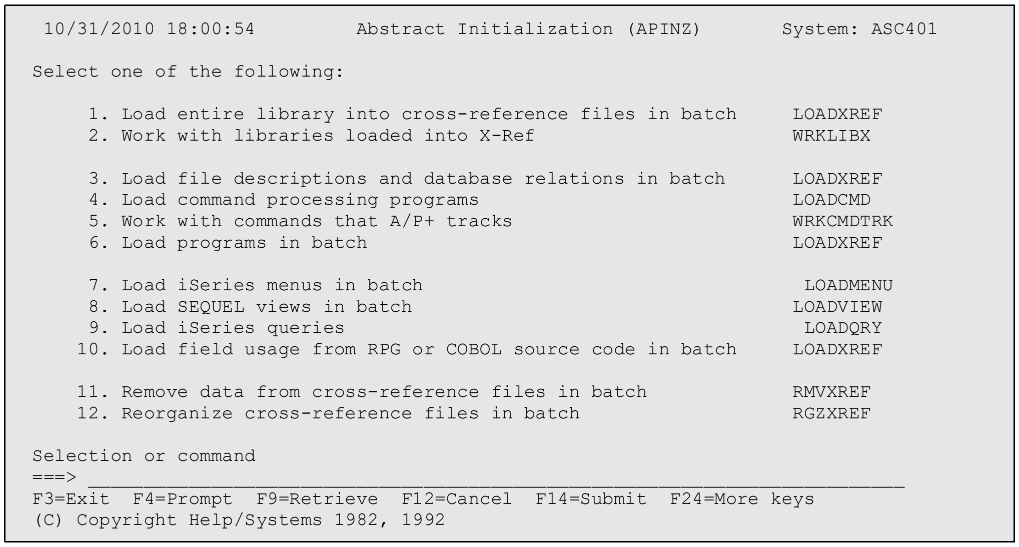
Initialization (APINZ) Menu

The initialization menu (APINZ) can be accessed from the ABSTRACT main menu through option 1, or by typing GO ABSTRACT/APINZ on a command entry line and pressing the Enter key.

The menu allows you to access most of the initialization functions of ABSTRACT and to load your applications into the cross reference files. It also allows you full access to your authorized command set through the command entry line at the bottom of the display.

The menu does not include specific references to the commands listed below. Refer to the command reference section on the following pages for details of each command.

LOADUSRPRF	Load User Profile objects
LOADJOBD	Load job descriptions
LOADSBSD	Load subsystem descriptions
LOADJOBSCD	Load job scheduler entries
LOADMNUMGR	Load menu manager definitions



```
 10/31/2010 18:00:54         Abstract Initialization (APINZ)       System: ASC401

Select one of the following:

     1. Load entire library into cross-reference files in batch     LOADXREF
     2. Work with libraries loaded into X-Ref                       WRKLIBX

     3. Load file descriptions and database relations in batch      LOADXREF
     4. Load command processing programs                            LOADCMD
     5. Work with commands that A/P+ tracks                         WRKCMDTRK
     6. Load programs in batch                                      LOADXREF

     7. Load iSeries menus in batch                                  LOADMENU
     8. Load SEQUEL views in batch                                  LOADVIEW
     9. Load iSeries queries                                         LOADQRY
    10. Load field usage from RPG or COBOL source code in batch     LOADXREF

    11. Remove data from cross-reference files in batch             RMVXREF
    12. Reorganize cross-reference files in batch                   RGZXREF

Selection or command
===> ______________________________________________________________________
F3=Exit  F4=Prompt  F9=Retrieve  F12=Cancel  F14=Submit  F24=More keys
(C) Copyright Help/Systems 1982, 1992
```

Full initialization

Once you have identified an application that you wish to include in the ABSTRACT dictionary, you will most likely want to load all the information about it into the cross references.

The starting point for loading any application is to use option 2 to invoke the WRKLIBX facility and begin an interactive procedure for creating and submitting initialization requests. WRKLIBX allows you to list the libraries in your application and submit an initialization request that automatically analyzes all objects in the correct sequence.

Partial initialization

If you want to load a subset of a library, or only run specific parts of the initialization process, you should perform the appropriate options (3 through 10) **in the order listed on the menu**.

If your application's file and program descriptions are not loaded into the ABSTRACT cross-references, RPG and COBOL source analysis functions will not work correctly. If user written commands are not properly cross referenced prior to CL analysis, references to them will not be included when the CL programs are processed.

Menu options

All the menu options except option numbers 2 and 5 will prompt (and then run) an ABSTRACT command. Depending on the menu item, the command will be submitted to the batch job queue or it will begin running immediately when you press the Enter key. Given that an initialization request can run for an extended period of time, this may not be an efficient use of your display device or interactive priority.

> Your request will always be submitted to batch if you type the menu option number and press F14 or F15. Use F15 if you want to prompt the Submit Job (SBMJOB) command before it is submitted.

Select option 1 to prompt the LOADXREF command. The command prompt will appear and you will have the opportunity to specify which objects should be loaded into the cross reference files. Refer to page 2-8 for more information about the LOADXREF command.

Use option 2 to begin the interactive process of working with the information in the cross references files. Refer to page 2-15 for more information about the WRKLIBX command.

Option 3 can be used to request specific initialization of database information. When the option is selected, the LOADXREF command prompt will appear. The prompt will allow entry of the object criteria to be loaded, and will be pre-filled with TYPE(*FILE) so that only the database descriptions will be initialized. Refer to page 2-8 for more information about the LOADXREF command.

Option 4 prompts the LOADCMD command and allows you to specify which of your user-written commands should be analyzed and placed into the command tracking file. ABSTRACT will automatically cross reference all command processing and validity checking programs and insert default entries into the command tracking file. Refer to page 2-23 for more information about the LOADCMD command.

Use option 5 to begin the interactive process of working with the information contained in the ABSTRACT command tracking files. Unlike the other options on this menu, option 5 does not result in any analysis of your applications software. Instead, you use it to specify what information about your commands will be placed into the cross references during the CL analysis phase. Refer to page 2-25 for more information about the WRKCMDTRKD command.

Option 6 will request analysis of the program objects in your applications by prompting the LOADXREF command with TYPE(*PGM) MBR(*OBJ) specified. Program relationships will be loaded for all programs. The source encapsulated into CL program objects will be used during the analysis unless ALWRTVSRC(*NO) was specified during the compilation. In the latter case and in the case of RPG and COBOL programs, the program object service data will be used to locate the source file and member. Refer to page 2-8 for more information about the LOADXREF command.

Use option 7 to indicate which iSeries menu objects should be placed into the cross references. The Load iSeries Menu Definition (LOADMENU) command will be prompted. Refer to page 2-35 for more information about the LOADMENU command.

Option 8 can be used to request that SEQUEL view definitions should be analyzed and database references included in the ABSTRACT files. The Load SEQUEL View Definition (LOADVIEW) command will be prompted. Refer to page 2-37 for more information about the LOADVIEW command.

Option 9 will prompt the Load iSeries Query Definition (LOADQRY) command so that database references made by query objects can be included in the cross reference files. Refer to page 2-41 for more information about the LOADQRY command.

Use option 10 to request analysis of RPG and COBOL source code. The LOADXREF command prompt will appear with TYPE(*SRCF) specified. You will be able to indicate the name and library of the source file to be analyzed. Refer to page 2-8 for more information about the LOADXREF command.

> Option 10 should be used only if ABSTRACT is unable to locate the HLL program source code. Option 6 will normally cause the HLL source to be analyzed, but may be unable to find source members no longer in the file and library identified by the service data. The joblog that results when option 6 runs will clearly identify the source which cannot be located and must be loaded through option 10. Refer to the Change Service Data (CHGSVCDTA) command for steps you can take to avoid this problem.

Option 11 will prompt the Remove Cross Reference (RMVXREF) command and allow you to remove information from the cross reference files. The Remove Cross Reference command has the same format as the Load Cross Reference (LOADXREF) command. Refer to page 2-8 for information about the LOADXREF/RMVXREF commands.

Option 12 will display a confirmation prompt for the reorganize function. The option will run the Reorganize Cross Reference (RGZXREF) command to remove deleted records and empty data sets from the files. Refer to page 2-52 for more information about the RGZXREF command.

Menu Function Keys

As with other ABSTRACT menus, the initialization menu provides several function key options. Notice that there are some differences (cf. F14, F19) from the function key set provided by OS/400 system menus.

F3	Exit this ABSTRACT function
F4	Prompt the option or command that is on the command line
F9	Retrieve ‘backwards’ through previously executed commands
F12	Exit the current display and return to the previous function
F14	Submit the option or command on the command line to batch. The default job description defined for this user will be used
F15	Prompt the Submit Job (SBMJOB) command before submitting the option or the command on the command line
F16	Work with options in the user option file
F18	Change the default job description and user option file defined for the current user
F19	Retrieve ‘forwards’ through previously executed commands
F24	Display more function keys

Load/Remove Cross Reference (LOADXREF/RMVX-REF)

The LOADXREF and RMVXREF commands are used to place and remove details of your application programs and files into the ABSTRACT cross reference files. The commands can be used within a job stream or from command entry. Many of the options on the APINZ menu execute the LOADX-REF command.

One way to access these commands is through the WRKLIBX command, described on page 2-15. WRKLIBX presents a display that lets you work with cross-reference information you have loaded or would like to load. This display is especially useful for setting up initial loads for applications that you want to work with. WRKLIBX lets you identify the libraries you want to load and offers an easy way to submit the LOADXREF commands needed to load everything in the correct sequence.

In most cases you will want to submit these commands to a batch subsystem rather than execute them interactively. In order to execute properly, the job performing the command must have the ABSTRACT library on its library list.

Any errors encountered during the load process will be logged to the APJRN file. Problems include: authority errors, missing source code, mismatch between member type and program attribute, and database files not loaded into cross-reference during source analysis. Since these errors can result in incomplete cross reference information, the LOADXREF job will set an abnormal completion code after it finishes processing all requested objects and the job log will list the specific errors encountered. These errors can also be reviewed using the WRKLIBX command. Errors should be corrected and the affected objects reloaded to ensure complete cross reference information.

The LOADXREF and RMVXREF commands share the same format and keyword definitions. For the sake of brevity, only the LOADXREF syntax is shown here. RMVXREF differs from LOADX-REF only in its function, not its form.

Deleted Objects

In normal processing, LOADXREF will automatically remove cross reference information for objects that are no longer on the system. This happens in all cases except when a generic object name is specified on the OBJ parameter. The same is true of the RMVXREF. When a generic object name is specified, processing only takes place for objects on the system that meet the generic object name entered.

Copybook Members

Embedded references to other source members are analyzed along with the members that reference them. When the reference is not qualified by library, ABSTRACT locates the copy members by looking first in the library for the current file, then through the jobs library list, and finally through the list of libraries loaded in the cross reference.

The following figure shows the command syntax diagram for the LOADXREF command. Five parameters may be supplied to the command, but only the first is required.

```
                    .-*CURLIB/------.   .-*ALL----------.
>>--LOADXREF----OBJ-+---------------+---+---------------+--------------------------------->
                    +-library-name/-+---'-generic*_name-'
                    '-*ALLLOD/------'
                                                                                 Required
==========================================================================================
                                                                                 Optional
        .-*ALL----.        .-*ALLOBJ-.          .-*PRV---.
>--TYPE-+---------+---DATE-+---------+---DTASET-+--------+------------------------------->
        +-*FILE---+        +-*CHG----+          +-*FIRST-+
        +-*PGM----+        '-date----'          '-name---'
        +-*SRCF---+
        +-*CMD----+
        +-*JOBD---+
        +-*MENU---+
        +-*VIEW---+
        +-*QRYDFN-+
        +-*SBSD---+
        +-*SRVPGM-+
        +-*MODULE-+
        +-*BNDDIR-+
        '-*NOFILE-'

                                   .-*SRCLIB------.
>--MBR-+----------------+---PGMLIB-+----------------------------------------------------><
       +-*BOTH----------+          '-library-name-'
       +-*OBJ-----------+
       +-*ALL-----------+
       +-*NONE----------+
       '-*generic*-name-'
```

OBJ Parameter

Indicates the objects to be analyzed. Information for each object included by the OBJ keyword will be placed into the cross references specified by the TYPE keyword. If a DDM source file is included by the OBJ parameter, a remote conversation will be started and the source members in the target source file will be analyzed as if they were located on the source machine.

<u>*CURLIB:</u> the library identified as the job's current library will be used to find objects matching the name criteria.

***ALLLOD:** all libraries currently loaded into the cross reference. All objects in the libraries currently loaded into the cross reference will be examined by the analysis routines. This will include libraries that have been entered on the WRKLIBX display even if the load function has not yet occurred for those libraries. Using this value makes it easy to update or refresh the cross reference information with a single command.

<u>*ALL:</u> requests that each object in the library be analyzed.

Name: specify the name of an object to be analyzed, or specify a generic name (one to nine characters suffixed with an asterisk) to indicate that all objects sharing the prefix will be analyzed.

TYPE Parameter

Identifies the type of cross reference information that you want be loaded for the objects meeting the name criteria above.

***ALL**: indicates that all cross references should be loaded for the objects. Each type of analysis listed below will occur when the *ALL option is selected. This includes the file, program and source and miscellaneous object (menu, view, etc.) cross references.

***FILE:** requests that the *file objects* meeting the OBJ keyword should be placed into the two file based cross references. The DSPDBR and DSPFFD commands will be executed for each file requested. Also cross references any trigger programs built over a physical file.

***PGM:** loads *program object* level information for the programs indicated by the OBJ keyword. The program-file cross reference will be loaded. CL programs will be analyzed. RPG and COBOL source members will only be analyzed if the MBR parameter is *OBJ.

***SRCF:** requests analysis of the *source files* matching the OBJ keyword and uses the MBR keyword to determine which members should be analyzed. **Use this option when you want to load field-level cross-reference information for a specific RPG or COBOL source member.**

Note that successful source analysis depends on previous *FILE and *PGM loading of the cross references. The in-depth field-level cross referencing available through ABSTRACT depends on prior loading of the file-field and file-program information for each source member. If you choose to use *SRCF loading, be certain that it is the final step in your loading process.

***CMD:** uses the LOADCMD command to load information about the validity checking program (VCP) and the command processing program (CPP) for the commands indicated by the OBJ keyword.

***JOBD:** uses the LOADJOBD command to load information about the job descriptions indicated by the OBJ keyword. User profile name, job queue, and output queue will be recorded for each job description.

***MENU:** uses the LOADMENU command to load information about the display and program menus indicated by the OBJ keyword.

***QRYDFN:** uses the LOADQRY command to load information about the iSeries query definition objects indicated by the OBJ keyword. The file usage cross reference will be updated.

***SBSD:** uses the LOADSBSD command to load information about the subsystem definitions indicated by the OBJ keyword. The routing programs used by the subsystem will be placed into the cross reference.

***VIEW:** uses the LOADVIEW command to load information about the SEQUEL views indicated by the OBJ keyword.

***SRVPGM:** loads service program object level information for the service programs indicated by the OBJ keyword. Service program information shows modules and service programs bound into the service program indicated by the OBJ keyword.

***MODULE:** loads module object level information for the modules indicated by the OBJ keyword. The information recorded for module objects is identical to that recorded for program objects.

***BNDDIR:** loads binding directory object level information for the binding directories indicated by the OBJ keyword. Binding directory information shows what modules and service programs have been added to a binding directory.

***NOFILE:** requests analysis of all object types except files. Some applications scatter programs and files across multiple libraries. In these cases, it is necessary to load all of the files from each of the libraries first (LOADXREF TYPE(*FILE)), then use this parameter (LOADXREF TYPE(*NOFILE)) to go back and reload all other object types from the same libraries.

DATE Parameter

Indicates what date criteria to use, if any, during the analysis.

<u>***ALLOBJ:**</u> Load all objects, regardless of their creation date, into the cross-reference files.

***CHG:** filters the objects selected by the OBJ keyword. The cross references are examined for each object requested. If the object creation date is more recent than the date of the cross reference information (or if cross reference information isn't found), it will be loaded into the cross references. If the object hasn't changed since its cross reference information was loaded, it will be skipped.

Date: Only objects created on or after this date (in system date format) will be passed to the analysis routines. When TYPE(*SRCF) loading is used, the member change date is examined and compared to this date.

DTASET Parameter

Allows you to maintain independent copies of cross-reference data. For example, data on an order-entry application could be stored in a data set called ORDENTRY. A separate customer look up application could be stored in a LOOKUP data set.

<u>***PRV:**</u> Use the value for this parameter from the previous session. If no previous session has occurred, use *FIRST.

***FIRST:** all information should be stored in the default data set.

Name: Names a separate data set, created by the Add Data Set (ADDDTASET) command, to contain the information. If the data set does not already exist, an error will occur.

MBR Parameter

If the TYPE parameter is specified as *ALL, *PGM, *FILE or *SRCF, this parameter controls how HLL program source members shall be accessed. If the parameter is left blank it will perform as follows:

TYPE(*ALL)	MBR(*BOTH)[a]
TYPE(*FILE)	MBR(*NONE)
TYPE(*SRCF)	MBR(*ALL)
All others	MBR(*OBJ)

a. Except when the Library parameter is *ALLLOD, then runs as MBR(*OBJ)

***BOTH**: requests the most complete analysis. TYPE(*ALL) must also be specified. Each time an RPG or COBOL program object is encountered, service information will be used to access the corresponding source members. In addition, all members in each source file included by the OBJ parameter will be analyzed. ABSTRACT will only analyze members once in cases where service data indicates a member in a source file within the library.

***OBJ:** object service data should be referenced for each RPG/COBOL program or module that meets the OBJ criteria to determine the source member to analyze. For ILE programs, source file information is retrieved from the program object. TYPE(*ALL) or TYPE(*PGM) must be specified. If the service data or program information is not correct, the ABSTRACT will search the libraries listed in the cross reference for the corresponding source member.

***ALL:** Each source member in all the source files meeting the OBJ qualification should be analyzed. TYPE(*ALL) or TYPE(*SRCF) is required.

***NONE:** source analysis should not occur. ABSTRACT will not analyze the program's source member. This is the only value that can be specified when the type parameter is *FILE.

*generic**-name: Only the members meeting both the OBJ name and MBR name criteria will be analyzed. TYPE(*ALL) or TYPE(*SRCF) is required.

PGMLIB Parameter

If the TYPE parameter is specified as *ALL or *SRCF, this parameter controls how ABSTRACT will cross reference the source members it analyzes. For complete cross referencing, ABSTRACT needs to record the library where the compiled program object resides. This library will be entered into all cross references originating from HLL program source members that are analyzed independently from program objects. No verification will be done by ABSTRACT to ensure that the objects actually exist in the indicated library.

<u>***SRCLIB:**</u> the library containing the source file will be assumed as the library containing the related program objects.

library-name: the specified library will be assumed to contain the related program object(s).

Examples

Note: *The level of sophistication provided by the LOADXREF and RMVXREF commands allows the same effect to be accomplished using several different methods. Each of the methods will have different performance characteristics if the number of objects actually analyzed is different. It may be wise to try different ways of accomplishing the same function (as below) in order to determine which best fits your environment.*

Load an entire library

If you are interested in loading data for an entire library, select option 1 of the ABSTRACT APINZ menu. This is the easiest way to load the ABSTRACT data files. It is equivalent to typing

```
LOADXREF OBJ(Library/*ALL) TYPE(*ALL) MBR(*BOTH)
```

where *Library* is the name of the library that you want to load.

Load a library subset

You may use the LOADXREF command for individual objects created after an entire library load. For example, to load a specific database file into the cross-reference files you might type:

```
LOADXREF OBJ(Library/File) TYPE(*FILE) MBR(*NONE)
```

This is equivalent to the function available through option 3 of the APINZ initialization menu.

Analysis of the program relations information for a COBOL or RPG program (function of APINZ option 3) can be accomplished with:

```
LOADXREF OBJ(Library/Program) TYPE(*PGM) MBR(*OBJ)
```

Date based analysis

Updating existing cross reference information can be accomplished by using the DATE parameter. If you know the last date the cross reference information was loaded, you should specify it on the LOADXREF command.

```
LOADXREF OBJ(RMSOB/*ALL) TYPE(*ALL) MBR(*BOTH) DATE(010192)
```

Only objects in the RMSOB library created on or after January 1, 1992 will be analyzed. Specifying a DATE parameter is usually faster than forcing ABSTRACT to load the entire library unless most of the objects in the library will be included by the date specified.

Using DATE(*CHG) causes objects that have never been placed into the cross references or those that have been modified since they were last loaded into the cross references to be analyzed.

```
LOADXREF OBJ(RMSOB/*ALL) TYPE(*ALL) MBR(*BOTH) DATE(*CHG)
```

While DATE(*CHG) this may seem to require the least amount of analysis time, the extra overhead required by searching the cross references to determine the last load date may sometimes outweigh the benefits.

Updating the cross references

Once libraries have been loaded into the cross reference files, you can easily update the information with a single command. Use *ALLLOD as a library qualifier as illustrated below:

```
   LOADXREF OBJ(*ALLLOD/*ALL) TYPE(*ALL)
```

All objects in each library loaded into the cross reference will be re–loaded so that current information is reflected.

```
   LOADXREF OBJ(*ALLLOD/*ALL) TYPE(*ALL) DATE(*CHG)
```

Each library in the current cross reference will be searched for objects that have been changed since they were last loaded. Only those objects will be re–loaded into the cross reference files.

Work With Library Cross Reference Data (WRKLIBX)

The WRKLIBX command gives you an easy way to load new applications and find out about the information loaded into ABSTRACT cross reference files. It uses a series of interactive displays to aid you in generating and refreshing the cross references. With it, the current information in the cross references can be accessed and requests for updating and removing information can be submitted to a batch subsystem.

You can invoke the WRKLIBX command from any command entry line, or by using option 2 from the APINZ menu. The command can also be run from an interactive program. Because WRKLIBX interacts with the workstation, it cannot be run in the batch environment. Other than the data set (DTASET) parameter, there are no parameters on the command.

```
                      .-*PRV---.
>>--WRKLIBX----DTASET-+--------+-----------------------------------------------------><
                      +-*FIRST-+
                      '-name---'
```

DTASET Parameter

Specifies the name of the data set you want to work with. You can accept the default or choose a separate data set, created by the Add Data Set (ADDDTASET) command, that contains the information. If the data set does not exist, an error will occur. The list of current data set names can be accessed by prompting the command and then prompting the data set parameter.

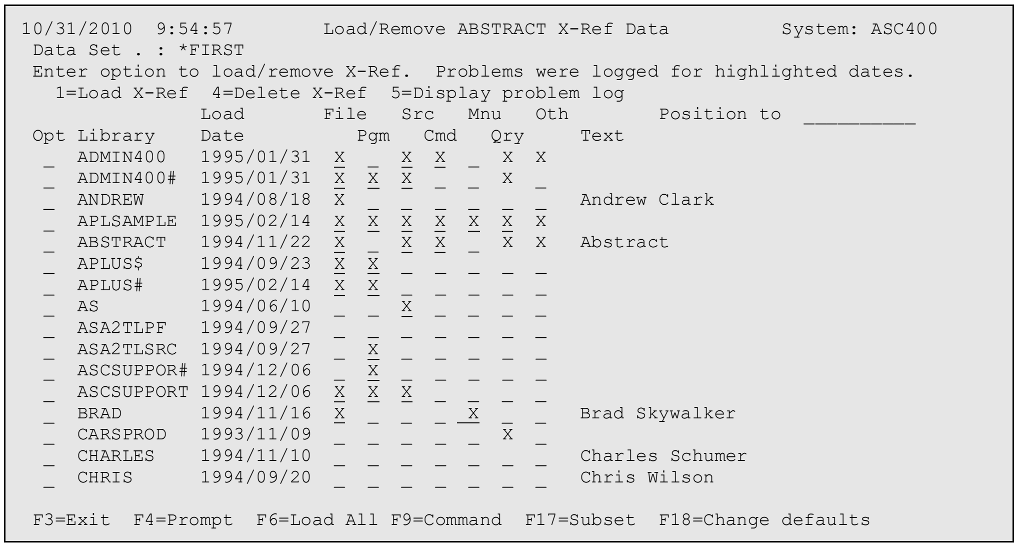
***PRV:** Use the value for this parameter from the previous session. If no previous session has occurred, use *FIRST.

***FIRST:** information from the first (default) data set should be accessed.

Name: names the data set that you want to work with.

The WRKLIBX Display

Once the command is processed, a display similar to the one below will appear. It shows all of the currently loaded libraries, when they were loaded, and what information is in place for them.



```
10/31/2010  9:54:57        Load/Remove ABSTRACT X-Ref Data          System: ASC400
 Data Set . : *FIRST
 Enter option to load/remove X-Ref.  Problems were logged for highlighted dates.
   1=Load X-Ref  4=Delete X-Ref  5=Display problem log
                Load       File   Src   Mnu   Oth        Position to  __________
 Opt Library    Date          Pgm   Cmd   Qry     Text
  _  ADMIN400   1995/01/31  X  _  X  X  _  X  X
  _  ADMIN400#  1995/01/31  X  X  X  _  _  X  _
  _  ANDREW     1994/08/18  X  _  _  _  _  _  _   Andrew Clark
  _  APLSAMPLE  1995/02/14  X  X  X  X  X  X  X
  _  ABSTRACT   1994/11/22  X  _  X  X  _  X  X   Abstract
  _  APLUS$     1994/09/23  X  X  _  _  _  _  _
  _  APLUS#     1995/02/14  X  X  _  _  _  _  _
  _  AS         1994/06/10  _  _  X  _  _  _  _
  _  ASA2TLPF   1994/09/27  _  _  _  _  _  _  _
  _  ASA2TLSRC  1994/09/27  _  X  _  _  _  _  _
  _  ASCSUPPOR# 1994/12/06  _  X  _  _  _  _  _
  _  ASCSUPPORT 1994/12/06  X  X  X  _  _  _  _
  _  BRAD       1994/11/16  X  _  _  _  X  _  _   Brad Skywalker
  _  CARSPROD   1993/11/09  _  _  _  _  _  X  _
  _  CHARLES    1994/11/10  _  _  _  _  _  _  _   Charles Schumer
  _  CHRIS      1994/09/20  _  _  _  _  _  _  _   Chris Wilson

 F3=Exit  F4=Prompt  F6=Load All F9=Command  F17=Subset  F18=Change defaults
```

Each library that has any information loaded will be displayed in alphabetical order. Current text for the library is listed at the right of the display if it is available. The date that objects within the library were last analyzed appears to the right of the library.

Eight columns in the middle of the display indicate the type of information loaded into the cross reference files. If database or device file objects from the library have been analyzed by the ABSTRACT file analysis routines, a ‘X’ will appear under the file column. Likewise, if programs objects within the library have been loaded, an ‘X’ will appear under the program column. If any RPG or COBOL source members in the library have been loaded, an ‘X’ will appear under the source column. If command, query, or menu objects have been loaded, an ‘X’ will appear under the respective column. If any other object types have been loaded (such as job descriptions or SEQUEL views), an ‘X’ appears under the ‘other’ column.

Entry fields

Initial load requests

This function offers the easiest way to load new application libraries into the cross-reference database. Simply enter all library names that make up the application into the library column and press F6 to load the listed libraries. Pressing F6 submits a batch job that will load all objects in the correct order to insure a successful load. Even if you wish to run the actual load step at a later time, entering the library names into the WRKLIBX display is the recommended starting point for running the initial load on a set of libraries. You can load the new libraries at a later time by using the LOADXREF command (page with the special value *ALLLOD for the library parameter.

Detailed cross reference information

Type an asterisk (‘*’) in place of one of the ‘X’ values in the middle of the display to request a detailed display of the library’s objects that are loaded into a specific type of cross reference. The resulting display, similar to that shown on page 2-19, will allow you to determine which objects are represented in the cross references and when they were loaded.

Error Log Requests

Any errors logged during the LOADXREF process were recorded in the APJRN file. If a library had any errors logged, the load date will be highlighted. The error log can be viewed by entering a 5 in the option column next to the library name. Errors should be corrected and the affected objects reloaded to ensure complete cross reference data.

Load/Remove Requests

In addition to reviewing current information, you can use the display above to make LOADXREF and/or RMVXREF requests and place them on a job queue. Also, the load date will be highlighted if any errors were encountered during the load process. Unless modified by using one or more of the function keys listed below, all requests will specify these parameter values:

```
OBJ(*ALL) TYPE(*ALL) DATE(*ALLOBJ) MBR(*BOTH)
```

Requests can be submitted simply by placing the proper value (1 or 4) into the fields in the middle of the display. Multiple requests may be made at one time by filling in several fields. The allowed values are:

1	Load cross reference information using LOADXREF
4	Remove cross reference information using RMVXREF

Place the value to the left of the library name to submit a request covering the entire library. All cross references for the objects in the library will be loaded and any source files found in the library or indicated by object service data will be examined as well. This is usually the <u>easiest</u> and most <u>trouble free</u> method of loading and removing cross reference information, but it may not be the optimal method in performance terms.

Values placed in the middle columns of the display will generate more specific load/remove requests. This can often be used to a significant advantage, resulting in decreased execution time of the analysis routines. If only some portions of the library have undergone modification it is usually not necessary to analyze the entire library.

If request values are placed under the 'Files' column, a request of TYPE(*FILE) will be submitted.
If placed under the 'Programs' column, a request of TYPE(*PGM) MBR(*OBJ) will be submitted.
If a request is made in the 'Source' column the command will be submitted with TYPE(*SRCF) MBR(*ALL) specified. Each member of the source files in the indicated library will be processed.

The command(s) are submitted using the library indicated on the line with the request. If you wish to analyze libraries other than those currently loaded into the cross references, roll to the end of the subfile and enter the library name on a blank line.

Function Keys

As with all ABSTRACT displays, use function key 3 to exit the WRKLIBX display.

F4 **Prompt**. Once you have entered requests into the display, you may press function key 4 to prompt each command before it is submitted. You can review the command and make any necessary changes before sending it to the job queue.

You can also use function key 4 to request a list of the libraries on your system so that you can choose which ones to analyze. To do this, use the roll keys to access the end of the subfile, position the cursor on a blank library line, and press function key 4. A display similar to the one shown on page 2-22 will appear.

F6 **Load all.** Use function key 6 to submit a request to load (or reload) all libraries listed on the display. A batch job is submitted that loads objects from all listed libraries. Objects are processed in the correct sequence to insure complete cross-reference data.

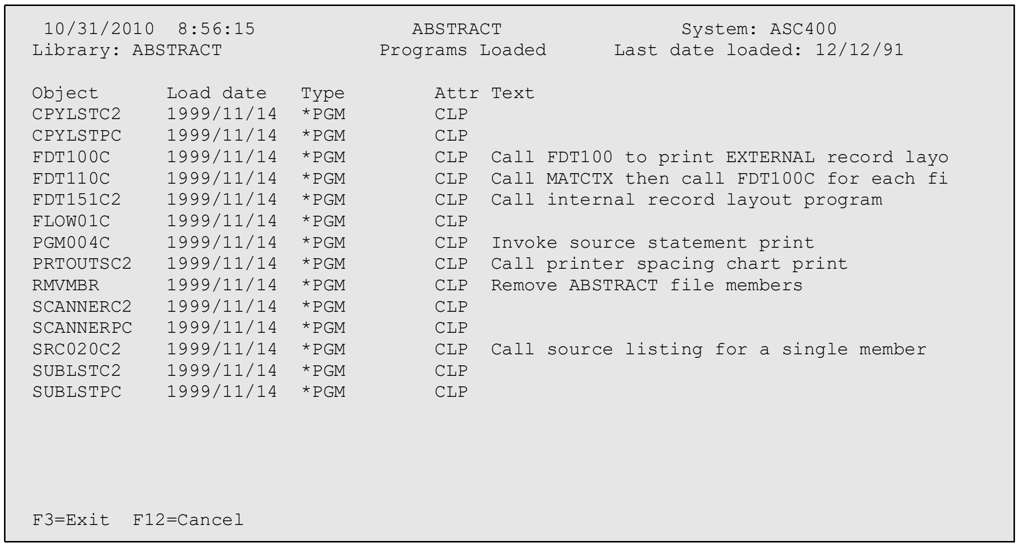
F9 **Command**. Use function key 9 to request a command entry line. A window like the one on page 2-19 will appear on the display. You will be able to enter new commands and recall commands that you have previously executed.

F17 **Subset**. By default, OBJ(*ALL) is specified when the load/remove request is submitted. By pressing function key 17 you can change the object subset criteria and specify a different set of objects to be loaded or removed. A window like the one on page 2-20 will appear on the display.

F18 **Change defaults**. ABSTRACT creates and maintains a data area containing several settings that are specific to your use of the product. Press function key 18 to change the job description that will be used for the submitted load/remove requests. A window like the one on page 2-21 will appear on the display.

Detailed cross reference contents

The following display shows the result of placing an asterisk (‘*’) on top of one of the ‘X’ markers on the primary display. It presents detailed information regarding the current contents of the cross reference files.

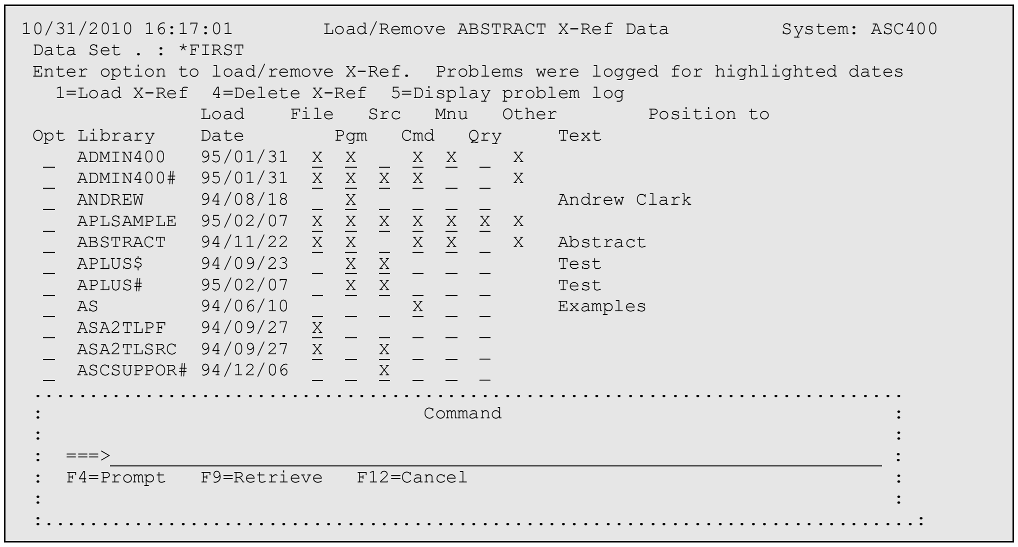

```
 10/31/2010  8:56:15              ABSTRACT                System: ASC400
Library: ABSTRACT              Programs Loaded      Last date loaded: 12/12/91

Object      Load date   Type        Attr Text
CPYLSTC2    1999/11/14  *PGM        CLP
CPYLSTPC    1999/11/14  *PGM        CLP
FDT100C     1999/11/14  *PGM        CLP  Call FDT100 to print EXTERNAL record layo
FDT110C     1999/11/14  *PGM        CLP  Call MATCTX then call FDT100C for each fi
FDT151C2    1999/11/14  *PGM        CLP  Call internal record layout program
FLOW01C     1999/11/14  *PGM        CLP
PGM004C     1999/11/14  *PGM        CLP  Invoke source statement print
PRTOUTSC2   1999/11/14  *PGM        CLP  Call printer spacing chart print
RMVMBR      1999/11/14  *PGM        CLP  Remove ABSTRACT file members
SCANNERC2   1999/11/14  *PGM        CLP
SCANNERPC   1999/11/14  *PGM        CLP
SRC020C2    1999/11/14  *PGM        CLP  Call source listing for a single member
SUBLSTC2    1999/11/14  *PGM        CLP
SUBLSTPC    1999/11/14  *PGM        CLP




F3=Exit  F12=Cancel
```

The top of the display indicates the library, type of cross reference, and last date any objects from the library were loaded into this cross reference. The middle of the display indicates each object from that library which is loaded into the cross reference, the date the information was loaded, and current object text if available.

Use the roll keys to roll through this information. F12 will return to the previous display. If F3 is pressed execution of the WRKLIBX command will be terminated.

Command entry window

A command entry window can be requested from the initial WRKLIBX display by pressing F9. A display similar to the one below will appear.

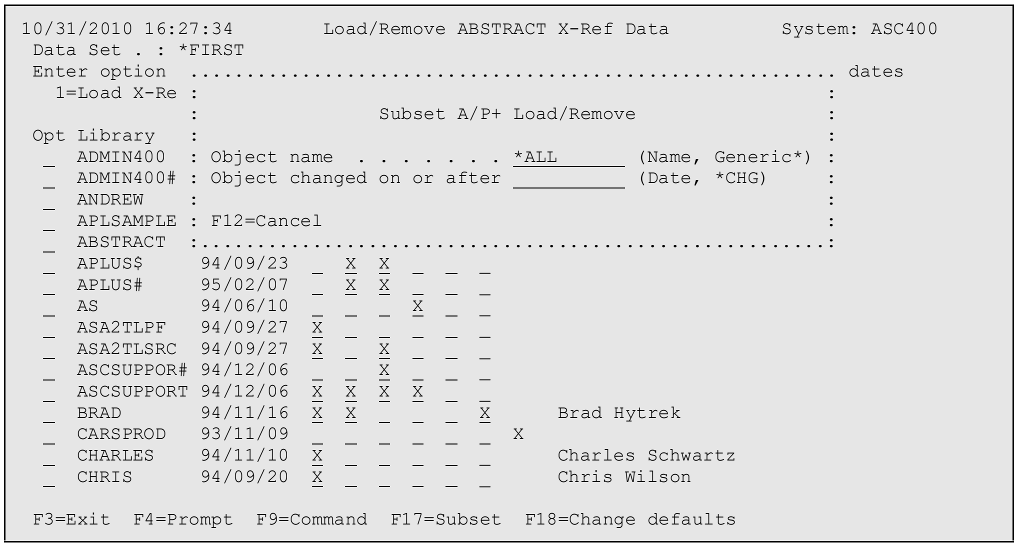


```
10/31/2010 16:17:01        Load/Remove ABSTRACT X-Ref Data          System: ASC400
 Data Set . : *FIRST
 Enter option to load/remove X-Ref.  Problems were logged for highlighted dates
   1=Load X-Ref  4=Delete X-Ref  5=Display problem log
               Load    File   Src   Mnu   Other        Position to
 Opt Library    Date        Pgm   Cmd   Qry     Text
  _  ADMIN400   95/01/31  X  X  _  X  X  _  X
  _  ADMIN400#  95/01/31  X  X  X  X  _  _  X
  _  ANDREW     94/08/18  _  X  _  _  _  _      Andrew Clark
  _  APLSAMPLE  95/02/07  X  X  X  X  X  X  X
  _  ABSTRACT   94/11/22  X  X  _  X  X  _  X   Abstract
  _  APLUS$     94/09/23  _  X  X  _  _  _      Test
  _  APLUS#     95/02/07  _  X  X  _  _  _      Test
  _  AS         94/06/10  _  _  _  X  _  _      Examples
  _  ASA2TLPF   94/09/27  X  _  _  _  _  _
  _  ASA2TLSRC  94/09/27  X  _  X  _  _  _
  _  ASCSUPPOR# 94/12/06  _  _  X  _  _  _
 ..............................................................................
 :                                 Command                                    :
 :                                                                            :
 :  ===>______________________________________________________________________ :
 :  F4=Prompt   F9=Retrieve   F12=Cancel                                      :
 :                                                                            :
 :............................................................................:
```

You may enter any command to which you are authorized, prompt the command with F4, use F9 to retrieve previously executed commands, or use F12 to remove the command entry window and return to the WRKLIBX display.

Changing the object subset

The values that will be used when you submit a Load/Remove Cross Reference command can be accessed and changed through the object subset window. Press F17 from the primary WRKLIBX display to cause the window to appear. It will look like the one below.



```
10/31/2010 16:27:34        Load/Remove ABSTRACT X-Ref Data          System: ASC400
 Data Set . : *FIRST
 Enter option  ....................................................... dates
   1=Load X-Re :                                                      :
               :                Subset A/P+ Load/Remove                :
 Opt Library   :                                                      :
  _  ADMIN400  : Object name  . . . . . . . *ALL______ (Name, Generic*) :
  _  ADMIN400# : Object changed on or after __________ (Date, *CHG)     :
  _  ANDREW    :                                                      :
  _  APLSAMPLE : F12=Cancel                                           :
  _  ABSTRACT  :......................................................:
  _  APLUS$     94/09/23  _  X  X  _  _  _
  _  APLUS#     95/02/07  _  X  X  _  _  _
  _  AS         94/06/10  _  _  _  X  _  _
  _  ASA2TLPF   94/09/27  X  _  _  _  _  _
  _  ASA2TLSRC  94/09/27  X  _  X  _  _  _
  _  ASCSUPPOR# 94/12/06  _  _  X  _  _  _
  _  ASCSUPPORT 94/12/06  X  X  X  X  _  _
  _  BRAD       94/11/16  X  X  _  _  _  X      Brad Hytrek
  _  CARSPROD   93/11/09  _  _  _  _  _  _  X
  _  CHARLES    94/11/10  X  _  _  _  _  _      Charles Schwartz
  _  CHRIS      94/09/20  X  _  _  _  _  _      Chris Wilson

 F3=Exit  F4=Prompt  F9=Command  F17=Subset  F18=Change defaults
```

Use this window to override the default values for the OBJ and DATE parameters. Refer to the LOADXREF command on page 2-8 for information about the values that you can use in these fields.

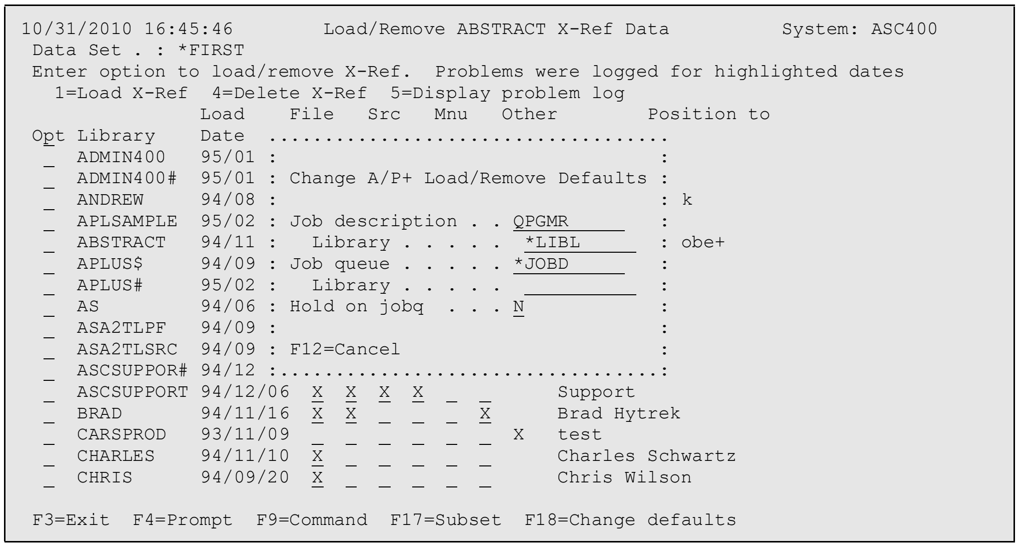
The current value will be shown in the entry fields that are presented on the display. Change them by typing the new values you want to use.

Press the Enter key to record your changes or press function key 12 to cancel the change and return to the previous display.

Changing the default job description

Initialization requests are submitted using the job description identified by the default values specific to your user profile. You can change the job description that will be used by pressing function key 18 from the primary WRKLIBX display. The job description determines both the job queue that will receive submitted jobs, and the initial library list of those jobs.

When function key 18 is pressed, a window similar to the one on the display shown on the next page will appear.



```
10/31/2010 16:45:46        Load/Remove ABSTRACT X-Ref Data          System: ASC400
 Data Set . : *FIRST
 Enter option to load/remove X-Ref.  Problems were logged for highlighted dates
   1=Load X-Ref  4=Delete X-Ref  5=Display problem log
               Load    File   Src   Mnu   Other        Position to
 Opt Library    Date  ..................................
  _  ADMIN400   95/01 :                                  :
  _  ADMIN400#  95/01 : Change A/P+ Load/Remove Defaults :
  _  ANDREW     94/08 :                                  : k
  _  APLSAMPLE  95/02 : Job description . . QPGMR        :
  _  ABSTRACT   94/11 :   Library . . . . .  *LIBL       : obe+
  _  APLUS$     94/09 : Job queue . . . . . *JOBD        :
  _  APLUS#     95/02 :   Library . . . . .  __________  :
  _  AS         94/06 : Hold on jobq  . . . N            :
  _  ASA2TLPF   94/09 :                                  :
  _  ASA2TLSRC  94/09 : F12=Cancel                       :
  _  ASCSUPPOR# 94/12 :..................................:
  _  ASCSUPPORT 94/12/06  X  X  X  X  _  _      Support
  _  BRAD       94/11/16  X  X  _  _  _  X      Brad Hytrek
  _  CARSPROD   93/11/09  _  _  _  _  _  _  X   test
  _  CHARLES    94/11/10  X  _  _  _  _  _      Charles Schwartz
  _  CHRIS      94/09/20  X  _  _  _  _  _      Chris Wilson

 F3=Exit  F4=Prompt  F9=Command  F17=Subset  F18=Change defaults
```

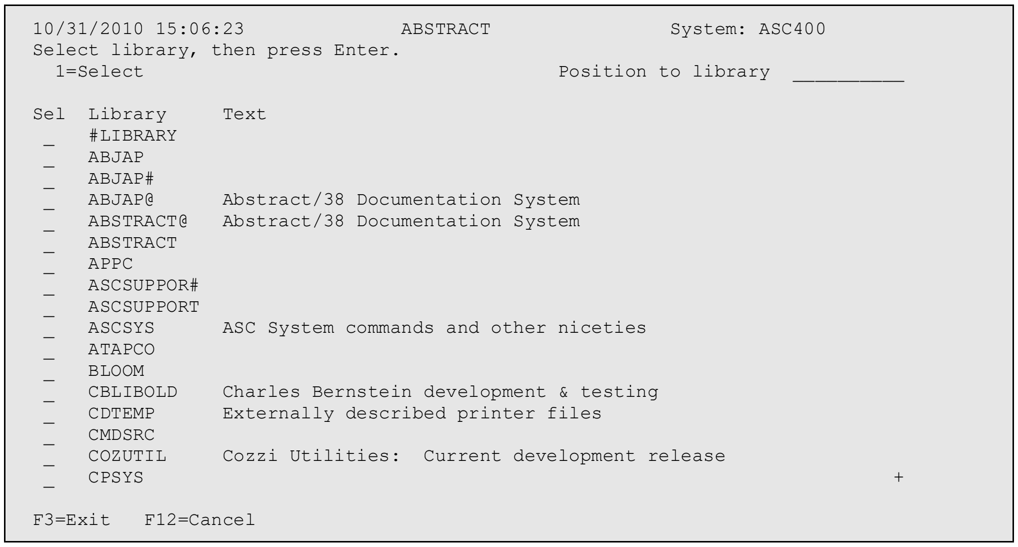
Type the name of the job description that should be used when submitting LOADXREF/RMVXREF commands. If you want submitted jobs to be automatically held on the job queue, type a Y into the 'Hold on jobq' field. Jobs that are placed on automatic hold must be explicitly released by the system operator or another user before they will execute.

Usually you will find that the job description that you normally use when compiling objects will be a good choice for your default value.

Press the Enter key to record your changes or press function key 12 to cancel the change and return to the previous display.

Library name prompting

If you are not certain of the library names available on your system, you can request a list by pressing function key 4 from the primary WRKLIBX display (see page 2-15) while no other load/remove options are pending. A display similar to the one below will appear.



```
10/31/2010 15:06:23              ABSTRACT                System: ASC400
Select library, then press Enter.
  1=Select                                     Position to library  __________

Sel  Library     Text
 _   #LIBRARY
 _   ABJAP
 _   ABJAP#
 _   ABJAP@      Abstract/38 Documentation System
 _   ABSTRACT@   Abstract/38 Documentation System
 _   ABSTRACT
 _   APPC
 _   ASCSUPPOR#
 _   ASCSUPPORT
 _   ASCSYS      ASC System commands and other niceties
 _   ATAPCO
 _   BLOOM
 _   CBLIBOLD    Charles Bernstein development & testing
 _   CDTEMP      Externally described printer files
 _   CMDSRC
 _   COZUTIL     Cozzi Utilities:  Current development release
 _   CPSYS                                                                +

F3=Exit   F12=Cancel
```

This display shows a list of library names which have not been placed into the cross reference files. OS/400 object text for the library is listed to the right of the library names. You may roll through the libraries using the roll keys or position the list by changing the ‘Position to’ field at the top of the display.

Add one or more of these libraries to the list on the primary display by placing a ‘1’ to the left of the library name(s) and pressing the Enter key. The initial WRKLIBX display will reappear with the libraries you selected listed at the bottom of the subfile. You may then specify a load request for them.

If you wish to end the option without submitting further requests, press function key 3. Use function key 12 to return to the previous display without selecting any additional libraries.

Load Command Information (LOADCMD) Command

The Load Command Information (LOADCMD) command locates command definition objects (*CMD) and adds entries into ABSTRACT cross references that record the command processing, validity checking, and prompt override programs for them. Validity checking programs (VCP) check to make sure that the user has supplied the correct number and type of parameters. Prompt override programs override what values are shown for certain parameters when the command is prompted. The values that are entered into these parameters are passed to the command processing program (CPP) for the command. See the CL Programmer's Guide for more information on user written commands.

LOADCMD also ensures that each command that it locates also has at least a default entry in the parameter tracking file. If it does not, LOADCMD adds a default entry into the file. These entries are used during CL program analysis to determine the level of command and object cross referencing. The tracking file entries can be viewed and changed by using the Work With Command Tracking Definitions (WRKCMDTRKD) command. Refer to page 2-25 for more information.

If your application does not include any command definitions, you need not execute this part of the initialization process.

The LOADCMD function is automatically run as part of the LOADXREF TYPE(*ALL) job stream. You can invoke the LOADCMD command from any command entry line, or by using option 4 from the APINZ menu. The command can also be run from within a batch or interactive program. The syntax for the LOADCMD command is shown below:

```
                   .-*CURLIB/------.   .-*ALL----------.          .-*PRV---.
>>--LOADCMD----CMD-+---------------+---+---------------+---DTASET-+--------+----------><
                   +-*USRLIBL/-----+   '-*generic_name-'          +-*FIRST-+
                   +-*ALLUSR/------+                              '-name---'
                   +-*ALL/---------+
                   '-*library_name-'
```

CMD Parameter

Specify the qualified name of the command(s) to be analyzed. A generic name (leading characters followed by an asterisk) can be specified to select objects with similar names. Specify a library name to be searched for the command(s) indicated or use one of the following special values:

***CURLIB:** Your job's current library will be searched for the commands that meet the command name part of the search criteria.

***USRLIBL:** The user portion of your job's library list will be searched for the commands that meet the command name part of the search criteria.

***ALLUSR:** All user libraries will be searched for the commands that meet the command name part of the search criteria.

***ALL:** All libraries on the system will be searched for the commands that meet the command name part of the search criteria. **Caution:** this includes library QSYS (which should probably not be analyzed)

The name portion of the CMD parameter specifies the commands that shall be analyzed in the library(s) selected.

***ALL:** Loads all commands in the specified library(s).

Name: specify the name of an individual command, or use a generic name (one to nine characters suffixed with an asterisk, e.g. SEQUEL*) to indicate that all commands sharing the prefix should be loaded.

DTASET Parameter

Specifies the name of the data set you want to load information into. You can accept the default or choose a separate data set, created by the Add Data Set (ADDDTASET) command, to contain the information. If the data set does not exist, an error will occur. The list of current data set names can be accessed by prompting the data set parameter.

***PRV:** Use the value for this parameter from the previous session. If no previous session has occurred, use *FIRST.

***FIRST:** all information should be stored in the default data set.

Name: the data set you want to use.

Examples

```
LOADCMD *ALL
```

Use this command to search your current library (*CURLIB) and load all cross reference all the commands in it against the validity and command processing programs they use.

```
LOADCMD PRDLIB/POST*
```

The library PRDLIB will be searched for commands beginning with POST. ABSTRACT will find commands such as POSTORDS, POSTPMTS, etc. The cross reference dictionary will be loaded so that references to these commands will be able to flow through to the programs that they invoke.

Work With Command Tracking Definitions (WRKCMDTRKD)

The Work With Command Tracking Definitions (WRKCMDTRKD) command provides you with an interactive facility that lets you tell ABSTRACT which commands that you want to track and the parameter values that you want to retain information about.

During CL analysis (option 6 from the APINZ menu), ABSTRACT will record program usage of commands listed in the tracking file, and also record references to the objects used by the command's parameters.

To do this, the command tracking file must contain two kinds of information:

the command name and an indication of whether ABSTRACT should keep track of CL program references to it, and

the names of command parameters and the type of OS/400 object referenced by a value for the parameter

ABSTRACT automatically tracks command usage and object references for these commands:

Allocate Object (ALCOBJ)
Add Logical File Member (ADDLFM)
Add Physical File Member (ADDPFM)
Call A Program (CALL)
Change Data With DFU (CHGDTA)
Clear Physical File Member (CLRPFM)
Create Logical File (CRTLF)
Create Physical File (CRTPF)
Deallocate Object (DLCOBJ)
Delete File (DLTF)
Display Data With DFU (DSPDTA)
Format Data (FMTDTA)
Override With Data Base File (OVRDBF)
Remove Member (RMVM)
Reorganize Physical File Member (RGZPFM)
Run Query (RUNQRY)
Start DFU (STRDFU)
Submit Job (SBMJOB) see notes at end of section
Submit Remote Command (SBMRMTCMD)
Transfer Control (TFRCTL)

Because they are handled by default, any WRKCMDTRKD entries for these commands will be ignored. ABSTRACT will not retain information about other commands unless command tracking definitions are created (and enabled) for them.

The Load Command Definitions (LOADCMD) command will guarantee that a default entry exists in the command tracking file for each command object that it processes (see page 2-23). Individual parameter definitions will not be placed into the file. If you want ABSTRACT to record object reference information for the various command parameters, you must add definitions for them to the command tracking file by using this command.

Access the Work With Command Tracking Definitions function by typing WRKCMDTRKD on a command entry line, or by using option 5 from the APINZ menu (see page 2-4). Because WRKCMDTRKD interacts with the workstation, it cannot be run in the batch environment.

The WRKCMDTRKD command has no parameters.

```
>>--WRKCMDTRKD---------------------------------------------------------------------><
```

When option 5 is selected from the APINZ menu, or the WRKCMDTRKD command is entered from a command line, a display similar to the one below will appear.

```
 10/31/2010 14:40:54   Work with Command Tracking Definitions     System: ASC401

Type option or command to track.              Position to command . . __________
  2=Edit  3=Copy  4=Delete  5=Display command source
                          Parameter
Opt Command    Parameter  Type       Track? Command Text
 _  ADDLIBLE   LIB        *LIB         Y    Add Library List Entry
 _  CHGDTAARA  DTAARA     *DTAARA      Y    Change Data Area
 _  CHGLIBL    LIBL       *LIB         Y    Change Library List
 _  CPYSPLF    TOFILE     *FILE        N    Copy Spooled File
 _  CRTDUPOBJ  NEWOBJ     OBJTYPE      Y    Create Duplicate Object
 _  CRTDUPOBJ  OBJ        OBJTYPE      Y    Create Duplicate Object
 _  GO         MENU       *MENU        Y    Go to Menu
 _  MOVOBJ     OBJ        OBJTYPE      Y    Move Object
 _  OPNQRYF    FILE       *FILE        Y    Open Query File
 _  OPNQRYF    FILE       *MBR         Y    Open Query File
 _  OPNQRYF    FILE       *FMT         Y    Open Query File
 _  OPNSQLF    VIEW       *DTAARA      Y    Open an SQL data path
 _  RTVDTAARA  DTAARA     *DTAARA      Y    Retrieve Data Area
 _  RUNQRY     QRY        *QRYDFN      Y    Run Query
 _  RUNQRY     QRYFILE    *FILE        Y    Run Query
 _  WRKOBJR    *ANY                    Y    Work with object references

 F3=Exit  F6=Create  F9=Command line  F22=More entries
```

The display shows the entries currently in the command tracking file and lets you change them to suit your needs. Each item line in the middle of the display names a command and describes one of its parameters. The command text appears to the right of the parameter definition.

Use the roll keys to access multiple pages on the display or the Position to value to locate a command directly.

Elements of the display

The ‘Command’ and ‘Parameter’ fields show the name of the command (and identify the parameter) that is being described. ABSTRACT will search for and record use of the command when analyzing the CL programs in your application.

It will also record the value of the listed parameter in its object reference database. Parameters other than those enabled in the command tracking file will be ignored.

A special value of ‘*ANY’ in the parameter name field indicates that no specific parameter referencing should occur, but that references to the command should still be recorded in the cross reference.

The ‘Parameter Type’ value records the type of object reference that occurs when a value is supplied for this command parameter. Usually, this value is an OS/400 object type (*FILE, *PGM, *DTAARA, etc.), but these special values can also be used:

*FMT	a file format name
*MBR	a file member name
*FLD	a field name
*SUB	a program subroutine name
*CMDSTR	a CL command string

*SQL	an SQL statement
*SEQUEL	an SQL statement that conforms to SEQUEL syntax

If the special value *CMDSTR is used, the parameter value will be analyzed as a command string. The value will be treated as if it were embedded in the program at the same line as the originating command and all references for the command string (if any) will also be logged. For instance, if command named SBMBCHJOB were being tracked, any references in a CL program to it would be logged. If *CMDSTR were specified for one of its parameters, the value of the indicated parameter would be scanned as if it were a separate command. If tracking definitions exist for the command string, its references will be placed into the cross reference as well.

Occasionally the object type referenced by a parameter depends on a value that is supplied by another parameter. In these cases, the name of the parameter holding the object type will be listed as the ‘Parameter Type’ value. (See the CRTDUPOBJ command above)

The ‘Track?’ field either enables (Y) or disables (N) the cross referencing of this command parameter. It is better to disable command tracking for a given command or parameter by placing an ‘N’ in this field instead of deleting the record from the file. If you later decide to track the command, you won’t need to figure out what the command/parameter definition should be.

Function keys

Press function key 3 to exit the WRKCMDTRKD command.

Press F6 to open the definition window so that you can create a tracking definition.

Use function key 9 to request a command entry line. A window like the one on page 2-20 will appear on the display. You will be able to enter new commands and recall commands that you have previously executed.

Function key 22 will display additional entries for the ‘Parameter’ field. You will use this display to describe mixed list parameters or parameters that require a reference to another parameter to determine the object’s library name. Refer to page 2-30 for more information.

Entry fields

Type any value in the ‘Position to’ field at the top of the display and press the Enter key. The list will be repositioned to the value specified.

Select a definition and perform one of the listed functions for it by positioning the cursor in the adjacent entry field, typing the option number, and pressing Enter.

Option 2 (edit) will present the definition window so that you can change the current tracking definition.

Option 3 (copy) will present a window that allows you to name a new command, parameter, or sequence. Use it when you want to use an existing definition as a starting point when creating additional definitions.

Option 4 (delete) will remove an entry from the list. No confirmation display will be presented.

Using the WRKCMDTRKD definitions

The following discussion and examples should help you understand how the values on this display affect the information ABSTRACT keeps. Use it as a guide as you create additional command tracking definitions. Refer to the display above for illustration.

No information

No information will be kept for any use of the Copy Spooled File (CPYSPLF) command because all definitions for it have been disabled by using a tracking flag value of ‘N’. All commands other than those listed on page 2-25 and those shown on the display will also be ignored. Any references to the RSTOBJ command, for example, will not be placed into the cross reference. Definitions for commands that are automatically analyzed will be ignored.

Simple command usage

CL program references to the WRKOBJR command will be recorded because an enabled entry exists for the command. No information about parameter value usage for the WRKOBJR command will be recorded during CL analysis since no enabled parameter definitions are present in the tracking file.

Later, when viewing ABSTRACT displays and reports, you will be able to determine that a given CL program references the WRKOBJR command and will be able to locate all uses of the WRKOBJR command, but you won’t be able to determine any of the objects that might be referenced through uses of the command. If the command has been loaded through the Load Command (LOADCMD) function, ABSTRACT will also be able to show the transfer of control through the command to its command processing program and any subsequent programs in the job stream.

Any enabled entry for a given command will provide this level of information. An enabled entry containing only a *ANY parameter name reference, is created for each command processed by the LOADCMD command if a command tracking definition does not already exist. If you want parameter level tracking for the command, you must enter a description for each parameter you want cross referenced.

Object reference through command parameters

Command parameter tracking provides an even greater level of cross referencing. With this kind of command tracking you will be able to determine the objects that a given CL program references and the commands that it uses in referencing them. You will also be able to use the ABSTRACT where used functions to locate parameter level usage for a given object.

Most command parameters are very simple to describe. All that is needed is the parameter name and the type of object that it references.

Some command parameters are more difficult to describe than the simple ones just illustrated. The type of object referenced by a parameter may be given by another parameter value (c.f. CRTDUPOBJ) or a parameter may allow a complex list of values like the FILE parameter of the OPNQRYF command. The definitional facility of WRKCMDTRKD will allow you to create appropriate descriptions of any command that you want ABSTRACT to analyze.

Simple parameters

Refer again to the display on page 2-26. When ABSTRACT finds a CL program that uses the CHGLIBL command, it will examine the LIB parameter and record a library name (*LIB) reference for each value used in the parameter. If the CL command is:

```
CHGLIBL (DATALIB PRODLIB QGPL QTEMP)
```

a reference for each library included in the list will be recorded. ABSTRACT will place each parameter value in the cross reference - whether the parameter allows just one name, or many.

ABSTRACT will correctly record parameter values, even if they represent qualified names. Consider the definition for the FILE parameter of the OPNQRYF command. It will cause an object reference entry for a *FILE object to be recorded, using the value(s) specified in the CL program.

For example, a CL program might include this command:

```
OPNQRYF FILE((PRDLIB/CUST) (PRDLIB/ORDER)) +
FORMAT(PRDLIB/JOINFMT) JFLD((1/CUSNO 2/CUSNO))
```

ABSTRACT will record not only the fact that the CL program uses the OPNQRYF command, but that *FILE objects named PRDLIB/CUST and PRDLIB/ORDER are referenced by the FILE parameter. No information about the FORMAT or JFLD keywords will be retained because they are not described on the WRKCMDTRKD display.

In the same way, ABSTRACT will create a record indicating the *DTAARA objects used by each CL program referencing the CHGDTAARA, OPNSQLF, or RTVDTAARA commands.

Indirect object type

Not all commands are as straight-forward as the ones we have just considered. The CRTDUPOBJ command, for example, has two parameters that are not specific to a single object type. The type of object referenced by the OBJ and NEWOBJ parameters can only be known by examining the value in the OBJTYPE parameter. To record this, the ‘Parameter Type’ column must specify the parameter name that contains the object type value. Refer to the display on page 2-26 for an example of the definition for the CRTDUPOBJ command.

If, when performing CL analysis, ABSTRACT encounters the command:

```
CRTDUPOBJ OBJ(CUST) FROMLIB(PRDLIB) +
OBJTYPE(*FILE) TOLIB(QTEMP) NEWOBJ(CUST2)
```

then it can correctly determine that the OBJ and NEWOBJ parameters are referring to *FILE objects because the command tracking definition for CRTDUPOBJ indicates that the OBJTYPE value determines the type of object referenced by OBJ and NEWOBJ.

Indirect qualification

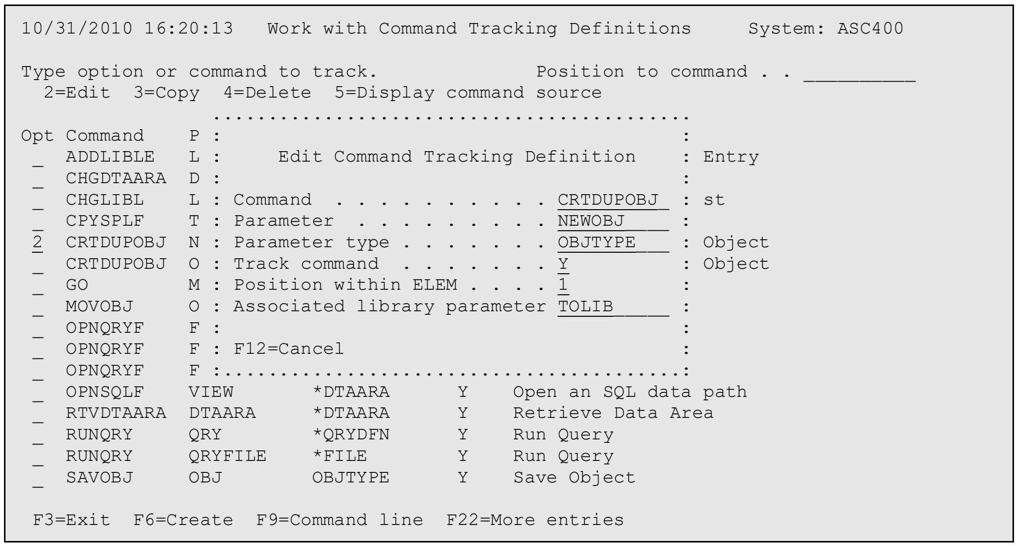
Notice that the definition for the CRTDUPOBJ command has separate parameters to indicate the library for the OBJ/NEWOBJ values. If ABSTRACT is to get the correct qualifiers for the objects named by the OBJ/NEWOBJ parameter values, it must be informed where to look for them.

Just as you can specify that the object type for a command parameter is determined by the value of another parameter (refer to the example above) the WRKCMDTRKD display also allows you to define an indirect library reference for a command parameter.

Press F22 from the WRKCMDTRKD display to view the extended definition display. Alternatively, you can view the complete definition for an individual parameter by selecting it with option 2=Edit. Either action will allow you to see additional values that describe command parameters with indirect qualification, or with mixed lists.

Notice on the edit display that two additional entry fields are provided. One of them lets you indicate the position of this parameter type within a mixed list of elements (see Mixed lists below). The other lets you to specify the name of the parameter that qualifies the object name in this parameter definition.

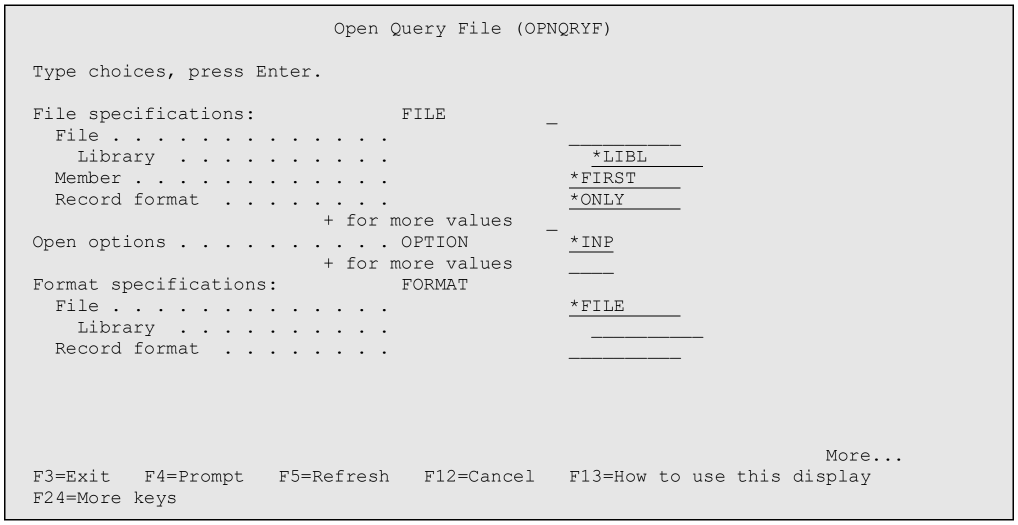
The example below tells ABSTRACT to record references to the NEWOBJ parameter of the CRTDUPOBJ command. When a reference to the command is located, the value of the OBJTYPE parameter will be stored in the cross reference records to identify the type of object indicated by the NEWOBJ parameter.



```
10/31/2010 16:20:13   Work with Command Tracking Definitions     System: ASC400

Type option or command to track.              Position to command . . __________
  2=Edit  3=Copy  4=Delete  5=Display command source
                 .........................................
Opt Command    P :                                       :
 _  ADDLIBLE   L :     Edit Command Tracking Definition    : Entry
 _  CHGDTAARA  D :                                       :
 _  CHGLIBL    L : Command  . . . . . . . . . . CRTDUPOBJ  : st
 _  CPYSPLF    T : Parameter  . . . . . . . . . NEWOBJ     :
 2  CRTDUPOBJ  N : Parameter type . . . . . . . OBJTYPE    : Object
 _  CRTDUPOBJ  O : Track command  . . . . . . . Y          : Object
 _  GO         M : Position within ELEM . . . . 1          :
 _  MOVOBJ     O : Associated library parameter TOLIB      :
 _  OPNQRYF    F :                                       :
 _  OPNQRYF    F : F12=Cancel                            :
 _  OPNQRYF    F :.......................................:
 _  OPNSQLF    VIEW       *DTAARA      Y    Open an SQL data path
 _  RTVDTAARA  DTAARA     *DTAARA      Y    Retrieve Data Area
 _  RUNQRY     QRY        *QRYDFN      Y    Run Query
 _  RUNQRY     QRYFILE    *FILE        Y    Run Query
 _  SAVOBJ     OBJ        OBJTYPE      Y    Save Object

 F3=Exit  F6=Create  F9=Command line  F22=More entries
```

Mixed lists

Some commands let you specify several kinds of objects (not just several objects of a given type) with a single parameter. Contrast the LIBL parameter of CHGLIBL with the FILE parameter of OPNQRYF. The command prompt display for the OPNQRYF command shows the complex nature of the FILE parameter.



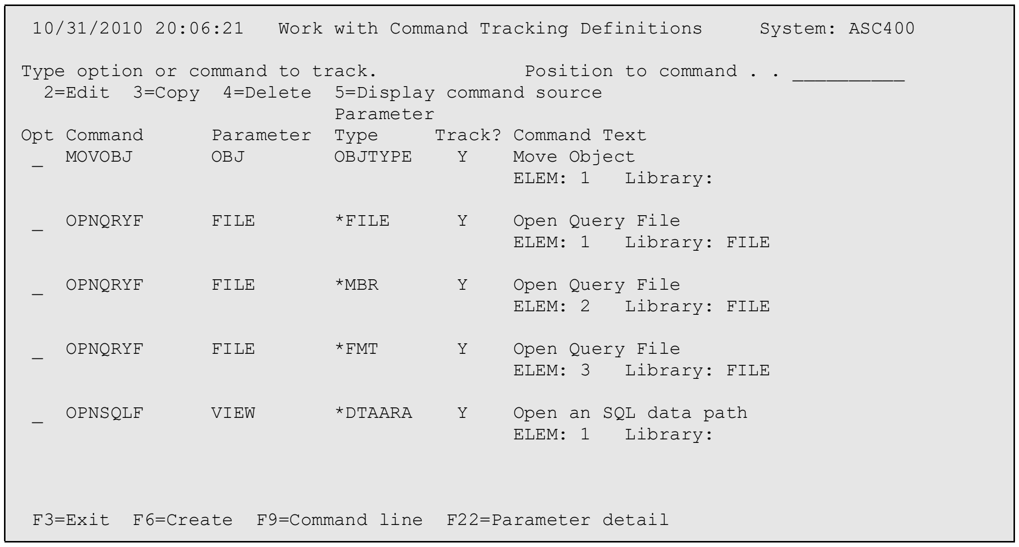
```
                            Open Query File (OPNQRYF)

 Type choices, press Enter.

 File specifications:            FILE          _
   File . . . . . . . . . . . . .                __________
     Library  . . . . . . . . . .                  *LIBL
   Member . . . . . . . . . . . .                *FIRST
   Record format  . . . . . . . .                *ONLY
                         + for more values     _
 Open options . . . . . . . . . . OPTION         *INP
                         + for more values       ____
 Format specifications:          FORMAT
   File . . . . . . . . . . . . .                *FILE
     Library  . . . . . . . . . .                  __________
   Record format  . . . . . . . .                __________




                                                                       More...
 F3=Exit   F4=Prompt   F5=Refresh   F12=Cancel   F13=How to use this display
 F24=More keys
```

In each case you are allowed to specify several values, but the LIBL parameter of the CHGLIBL command lets you enter only a simple list - only one *kind* of value. On the other hand, the FILE parameter of the OPNQRYF command lets you enter a qualified file name, a member name, and a format name *for each element in the list.*

The first element in OPNQRYF's FILE list is a qualified file name. The second element refers to member name to be used from the file. The third (and final) element in the list is the name of the record format to be accessed from the file.

ABSTRACT needs to know which part of the list you want tracked. You do not need to track each part of the list if you are only interested in one or two elements. The element number for the item you are tracking is shown on the extended display and in the definition window.

The display below shows the complete definition for the OPNQRYF list. Since the file name is the first element of the FILE list, the description for the *FILE portion of the parameter indicates element position 1. The *MBR entry indicates element position 2. The *FMT entry indicates element position 3.



```
 10/31/2010 20:06:21   Work with Command Tracking Definitions     System: ASC400

Type option or command to track.             Position to command . . __________
  2=Edit  3=Copy  4=Delete  5=Display command source
                            Parameter
Opt Command      Parameter  Type     Track? Command Text
 _  MOVOBJ       OBJ        OBJTYPE    Y    Move Object
                                            ELEM: 1   Library:

 _  OPNQRYF      FILE       *FILE      Y    Open Query File
                                            ELEM: 1   Library: FILE

 _  OPNQRYF      FILE       *MBR       Y    Open Query File
                                            ELEM: 2   Library: FILE

 _  OPNQRYF      FILE       *FMT       Y    Open Query File
                                            ELEM: 3   Library: FILE

 _  OPNSQLF      VIEW       *DTAARA    Y    Open an SQL data path
                                            ELEM: 1   Library:



 F3=Exit  F6=Create  F9=Command line  F22=Parameter detail
```

Given the descriptions above, ABSTRACT will analyze occurrences of the OPNQRYF command and correctly cross reference the file, member, and form names used by the command. You will be able to use ABSTRACT displays and reports to find file, member, and format usage as it occurs in your CL programs through the OPNQRYF command.

Mixed lists can seem both complicated and confusing. Describing them to ABSTRACT can appear difficult, especially at first. It is important to realize that they are quite rare among OS/400 commands, and practically never used in user written commands.

Verifying the accuracy of your entries

Once you have made entries in the WRKCMDTRKD file, you will want to be certain that they are correct, and that ABSTRACT is able to acquire the information that you want.

The best way to do this is to create (or find) a CL program that uses the commands that you want to check and load it into the cross references using a command like this:

LOADXREF OBJ(*Library/Program*) TYPE(*PGM) MBR(*OBJ)

Once the command has completed (it should not take long) you can find out what information ABSTRACT was able to get using the command definitions you have specified on the WRKCMDTRKD display. Use a command like:

WRKOBJRS *Library/Program* *PGM EXPLVL(0) OUTPUT(*PRINT)

to find out what ABSTRACT was able to store into the cross reference files based on the command definition and source code you provided.

The definitions on page 2-26 could be verified using the CL source code below. Notice that parameters and command names referenced in the command tracking description file is tested using a sample CL command in the source.

```
SEQNBR *...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 .
    1- PGM
    2-       ADDLFM   FILE(DATALIB/JOINLGL) MBR(LGLMBR) +
    3              DTAMBRS((DATALIB/PHYDTA1 (PHYMBR1 PHYMBR2)) +
    4                  (DATALIB/PHYDTA2 (PHYMBR3 PHYMBR4)))
    6-
    7-       CHGLIBL  LIBL(PRDLIB SEQUEL QGPL QTEMP)
  100-
  401-       CRTDUPOBJ OBJ(ORIGINAL) FROMLIB(PRODLIB) +
  402             NEWOBJ(NEWOBJ) TOLIB(QTEMP) OBJTYPE(*DTAARA)
  403-
  500-       OPNQRYF  FILE((PRDLIB/PHYDTA1 PHY1MBR1) +
  501              (PRDLIB/PHYDTA2 PHY2MBR2 PHYD2FMT)) +
  502              FORMAT(PRDLIB/FMTFILE) JFLD((PHYDTA1/CUSNO +
  503              2/CUSNO))
  504-
  700-       OPNSQLF  VIEW(ANDREWVIEW)
  701-
  702-       WRKOBJR  ANDREW
  800- ENDPGM
                * * * * *  E N D  O F  S O U R C E  * * * * *
```

If the definitions on page 2-26 were present in the command tracking file when the CL program code above was analyzed, the Work With Sequenced Object References (WRKOBJRS) listing for the program would look like the one on the next page.

The program specified on the command is shown at the head of the object list. The name of the program (WRKCMDTRK), its attribute (*PGM CLP), library (ANDY), and object text are all printed on the listing. All of the objects referenced by this program are listed underneath it and indented to the right.

The first several referenced objects have no sequence number printed under the Extended Description column. The information for these entries was obtained when the Display Program References (DSPPGMREF) command processed the CMDTRKTEST program during the LOADXREF initialization step. Each of the object references indicates the object involved, its associated library, and the type of usage made (somewhere) in the program.

```
  10/31/2010 17:11:02                         Abstract                        Page . . . :   1
                                   Sequenced Object References

Object name . . : CMDTRKTEST                                               System . . : ASC400
Object type . . : *PGM                                                     Data Set . : *FIRST
Object library  : ANDY
                                   Library or
Object/Uses  Type       Usage      Qualifier   Extended Description    Text
CMDTRKTEST   *PGM CLP              ANDY                                Test CMDTRK descriptions
  JOINLGL    *FILE                 DATALIB
  PHYDTA1    *FILE                 PRDLIB
  PHYDTA2    *FILE                 PRDLIB
  JOINLGL    *FILE      ADDLFM     DATALIB     Seq#(0.02) Mbr(LGLMBR)
  CHGLIBL    *CMD       CHGLIBL    QSYS        Seq#(0.07) Lib(*LIBL)   Change Library List
  PRDLIB     *LIB       CHGLIBL    QSYS        Seq#(0.07)
  QGPL       *LIB PROD  CHGLIBL    QSYS        Seq#(0.07)              General Purpose Library
  QTEMP      *LIB       CHGLIBL    QSYS        Seq#(0.07)
  SEQUEL     *LIB PROD  CHGLIBL    QSYS        Seq#(0.07)              SEQUEL/400 Data Manageme
  CRTDUPOBJ  *CMD       CRTDUPOBJ  QSYS        Seq#(4.01) Lib(*LIBL)   Create Duplicate Object
  NEWOBJ     *DTAARA    CRTDUPOBJ  QTEMP       Seq#(4.01)
  ORIGINAL   *DTAARA    CRTDUPOBJ  PRODLIB     Seq#(4.01)
  OPNQRYF    *CMD       OPNQRYF    QSYS        Seq#(5) Lib(*LIBL)      Open Query File
  PHYDTA1    *FILE PF   OPNQRYF    QTEMP       Seq#(5) Lib(PRDLIB)     Sample Data File
  PHYDTA2    *FILE      OPNQRYF    PRDLIB      Seq#(5)
  PHYD2FMT   *FMT       OPNQRYF    PHYDTA2     Seq#(5)
  PHY1MBR1   *MBR       OPNQRYF    PHYDTA1     Seq#(5)
  PHY2MBR2   *MBR       OPNQRYF    PHYDTA2     Seq#(5)
  ANDREWVIEW SQLVIEW    OPNSQLF    ANDY        Seq#(7) Lib(*LIBL)
  OPNSQLF    *CMD       OPNSQLF    SEQUEL      Seq#(7) Lib(*LIBL)      Open a SEQUEL file
  WRKOBJR    *CMD       WRKOBJR    APLUS       Seq#(7.02) Lib(*LIBL)   Work with object referen
```

The rest of the listing reports information discovered during the CL analysis phase of the LOADX-REF process. Notice that the source sequence number generating the entry is listed in the Extended Description column. Since the report is printed in order by source sequence, it should be very easy to locate missing (or incorrect) entries. Refer to the WRKCMDTRK definitions and the source code on the previous page to see how each command and parameter value is documented by ABSTRACT.

One final explanation about the report is in order. Notice that the Extended Description column sometimes, but not always, contains a 'Lib(...)' reference. The content of the 'Lib(...)' reference indicates how the object listed under the "Object" column was originally qualified in the source code.

The 'Lib(...)' reference will be generated when ABSTRACT is unable to locate an object as originally qualified in the source code, yet it is able to find an object matching the indicated name in a different library. A 'Lib(...)' reference will also be generated each time an unqualified (*LIBL) reference occurs in the source.

The library name listed in the "Library" column indicates the library in which ABSTRACT located the object using its search algorithm. The object attribute and text description are retrieved from this object.

For example, the CHGLIBL, CRTDUPOBJ, and OPNQRYF commands were all unqualified (or qualified by *LIBL) in the original source, but located in the QSYS library by the WRKOBJRS processing program. The ANDREWVIEW object used in sequence 7.00 was also unqualified, but located in library ANDY when the WRKOBJRS report was created. The PHYDTA1 file was qualified by PRTLIB in the source code, but couldn't be found there when the WRKOBJRS command was run. A PHYDTA1 file was located in the QTEMP library and used to acquire an attribute and text value.

Once you are satisfied that the command usage and parameter definitions are being cross referenced the way that you want, you should use the Load Cross Reference Data (LOADXREF) command to process your application library and load the information into the cross reference files.

SBMJOB tracking notes

SBMJOB is often used to submit calls to other programs. ABSTRACT can track program references between such programs as long as the request or command string is explicit in the SBMJOB command. If the command string is built elsewhere in the program, ABSTRACT will not be able to determine a relation between the programs.

Load iSeries Menu (LOADMENU) Command

The Load Menu (LOADMENU) command allows you to load native iSeries menus into the ABSTRACT cross-reference dictionary. Once loaded, ABSTRACT will treat them similar to CL programs - menu options will explode out and down just as CALLs from within a CL program.

ABSTRACT documents both display file and program menus. Display file menus are documented so that the command for each option can be recorded along with the option number that runs it. The menu's associated message file is located and processed as if it were a CL program. Both standard and user created command tracking definitions are used to record the references that occur within the menu.

Program menus are documented as a single CALL-type transfer from the menu object to the menu program. CL analysis of the menu program will create the documentation for it.

If your application does not include any iSeries menu (*MENU) objects, you need not execute this part of the initialization process.

The LOADMENU command is automatically run as part of the LOADXREF TYPE(*ALL) job stream. You can invoke the LOADMENU command from any command entry line, or by using option 7 from the APINZ menu. The command can also be run from within a batch or interactive program. The syntax for the LOADMENU command is shown below:

```
                     .-*CURLIB/------.   .-*ALL----------.          .-*PRV---.
>>--LOADMENU----MENU-+---------------+---+---------------+---DTASET-+--------+---------><
                     +-*USRLIBL/-----+   '-*generic_name-'          +-*FIRST-+
                     +-*ALLUSR/------+                              '-name---'
                     +-*ALL/---------+
                     '-*library_name-'
```

MENU Parameter

Specify the qualified name of the menu(s) to be analyzed. A generic name (leading characters followed by an asterisk) can be specified to select objects with similar names. Specify a library name to be searched for the menu(s) indicated or use one of the following special values:

<u>*CURLIB:</u> Your job's current library will be searched for the menus that meet the menu name part of the search criteria.

***USRLIBL:** The user portion of your job's library list will be searched for the menus that meet the menu name part of the search criteria.

***ALLUSR:** All user libraries will be searched for the menus that meet the menu name part of the search criteria.

***ALL:** All libraries on the system will be searched for the menus that meet the menu name part of the search criteria. **Caution:** this includes libraries QSYS and QUSRSYS (which should probably <u>not</u> be analyzed)

The name portion of the MENU parameter specifies the menus that shall be analyzed.

<u>*ALL:</u> Loads all menus in the specified library(s).

Name: specify the name of an individual menu, or use a generic name (one to nine characters suffixed with an asterisk, e.g. MENU*) to indicate that all menus sharing the prefix should be loaded.

DTASET Parameter

Specifies the name of the data set you want to load information into. You can accept the default or choose a separate data set, created by the Add Data Set (ADDDTASET) command, to contain the information. If the data set does not exist, an error will occur. The list of current data set names can be accessed by prompting the data set parameter.

***PRV:** Use the value for this parameter from the previous session. If no previous session has occurred, use *FIRST.

***FIRST:** all information should be stored in the default data set.

Name: the data set you want to use.

Examples

```
LOADMENU *ALL
```

Use this command to search your current library (*CURLIB) and load cross references with all the menus in it.

```
LOADMENU PRDLIB/MENU*
```

The library PRDLIB will be searched for menus beginning with MENU. ABSTRACT will find menus such as MENU01, MENU02, etc. The cross reference dictionary will be loaded so that you will be able to view the function of the menus through the object relations displays.

```
LOADMENU *ALLUSR/*ALL
```

All the menus in all user libraries will be loaded into the cross references.

Load SEQUEL View (LOADVIEW) Command

If you have licensed the SEQUEL data retrieval and manipulation product from Help/Systems, ABSTRACT can document the content and usage of the SEQUEL views on your system.

SEQUEL views are similar to SQL/400 views in that they define a result table of rows and columns based on fields from files in your application database. The LOADVIEW command will use SEQUEL's Display View Description (DSPVIEWD) command to cross reference the files used by each view. This information will be presented on ABSTRACT object relation displays and reports.

> If your application does not include any SEQUEL view objects, you need not execute this part of the initialization process.

The LOADVIEW command is automatically run as part of the LOADXREF TYPE(*ALL) job stream. You can invoke the LOADVIEW command from any command entry line, or by using option 8 from the APINZ menu. The command can also be run from within a batch or interactive program. The syntax for the LOADVIEW command is shown below:

```
                     .-*CURLIB/------.   .-*ALL----------.          .-*PRV---.
>>--LOADVIEW----VIEW-+---------------+---+---------------+---DTASET-+--------+---------><
                     +-*USRLIBL/-----+   '-*generic_name-'          +-*FIRST-+
                     +-*ALLUSR/------+                              '-name---'
                     +-*ALL/---------+
                     '-*library_name-'
```

VIEW Parameter

Specify the qualified name of the view data area(s) to be analyzed. A generic name (leading characters followed by an asterisk) can be specified to select objects with similar names. Specify a library name to be searched for the view(s) indicated or use one of the following special values:

***CURLIB:** Your job's current library will be searched for the views that meet the object name part of the search criteria.

***USRLIBL:** The user portion of your job's library list will be searched for the views that meet the object name part of the search criteria.

***ALLUSR:** All user libraries will be searched for the views that meet the object name part of the search criteria.

***ALL:** All libraries on the system will be searched for the views that meet the object name part of the search criteria.

The name portion of the VIEW parameter specifies the views that will be analyzed.

***ALL:** Loads all views in the specified library(s).

Name: specify the name of an individual view, or use a generic name (one to nine characters suffixed with an asterisk, e.g. CUST*) to indicate that all views sharing the prefix should be loaded.

DTASET Parameter

Specifies the name of the data set you want to load information into. You can accept the default or choose a separate data set, created by the Add Data Set (ADDDTASET) command, to contain the information. If the data set does not exist, an error will occur. The list of current data set names can be accessed by prompting the data set parameter.

***PRV:** Use the value for this parameter from the previous session. If no previous session has occurred, use *FIRST.

***FIRST:** all information should be stored in the default data set.

Name: the data set you want to use.

Examples

```
LOADVIEW *ALL
```

Use this command to search your current library (*CURLIB) and load cross references with all the views in it.

```
LOADVIEW PRDLIB/CUST*
```

The library PRDLIB will be searched for views beginning with CUST. ABSTRACT will find views such as CUSTV1, CUSTLIST, etc. The cross reference dictionary will be loaded so that you will be able to see the files that each view uses through the object relations displays and reports.

```
LOADVIEW *ALLUSR/*ALL
```

All the views in all user libraries will be loaded into the cross references.

Load SEQUEL Report (LOADREPORT) Command

If you have licensed the SEQUEL data retrieval and manipulation product from Help/Systems, ABSTRACT can document relations between SEQUEL reports and the SEQUEL views used by those reports.

SEQUEL reports are built over SEQUEL views and use the view to retrieve data for processing by the report definition. The report definition gives users extensive formatting control over view results as well as calculation capabilities not available in SQL. The LOADREPORT command will cross reference the view used by each report. This information will be presented on ABSTRACT object relation displays and reports.

If your application does not include any SEQUEL report objects, you need not execute this part of the initialization process.

The LOADREPORT command is automatically run as part of the LOADXREF TYPE(*ALL) job stream. You can also run the LOADREPORT command from any command entry line. The command can also be run from within a batch or interactive program. The syntax for the LOADREPORT command is shown below:

```
                         .-*CURLIB/------.   .-*ALL----------.          .-*PRV---.
>>--LOADREPORT----REPORT-+---------------+---+---------------+---DTASET-+--------+------>
                         +-*USRLIBL/-----+   '-*generic_name-'          +-*FIRST-+
                         +-*ALLUSR/------+                              '-name---'
                         +-*ALL/---------+
                         '-*library_name-'

           .-*USRSPC-.
>--RPTTYPE-+---------+-----------------------------------------------------------------><
           '-*DTAARA-'
```

REPORT Parameter

Specify the qualified name of the report object(s) to be analyzed. A generic name (leading characters followed by an asterisk) can be specified to select objects with similar names. Specify a library name to be searched for the view(s) indicated or use one of the following special values:

***CURLIB:** Your job's current library will be searched for the views that meet the object name part of the search criteria.

***USRLIBL:** The user portion of your job's library list will be searched for the views that meet the object name part of the search criteria.

***ALLUSR:** All user libraries will be searched for the views that meet the object name part of the search criteria.

***ALL:** All libraries on the system will be searched for the views that meet the object name part of the search criteria.

The name portion of the VIEW parameter specifies the views that will be analyzed.

*ALL: Loads all reports in the specified library(s).

Name: specify the name of an individual view, or use a generic name (one to nine characters suffixed with an asterisk, e.g. CUST*) to indicate that all views sharing the prefix should be loaded.

DTASET Parameter

Specifies the name of the data set you want to load information into. You can accept the default or choose a separate data set, created by the Add Data Set (ADDDTASET) command, to contain the information. If the data set does not exist, an error will occur. The list of current data set names can be accessed by prompting the data set parameter.

*PRV: Use the value for this parameter from the previous session. If no previous session has occurred, use *FIRST.

***FIRST:** all information should be stored in the default data set.

Name: the data set you want to use.

RPTTYPE Parameter

Specifies the type of SEQUEL report you wish to load.

*USRSPC: Load information for user space type reports. User space reports are the type of report created by current versions of SEQUEL.

*DTAARA: Load information for data area type reports. Data area reports are the type of report created by the original versions of SEQUEL.

Examples

```
LOADREPORT *ALL
```

Use this command to search your current library (*CURLIB) and load cross references with all the reports in it.

```
LOADREPORT PRDLIB/CUST*
```

The library PRDLIB will be searched for reports beginning with CUST. ABSTRACT will find reports such as CUSTVR1, CUSTLISTR, etc. The cross reference dictionary will be loaded so that you will be able to see the view that each report uses through the object relations displays and reports.

Load iSeries Query Definition (LOADQRY) Command

If you have licensed the Query/400 data retrieval product from IBM, ABSTRACT can document the content and usage of the query definitions on your system.

Query definitions are similar to SQL/400 views in that they define a result table of rows and columns based on fields from files in your application database. The LOADQRY command will search each query definition and cross reference the files and fields used by them. This information will be presented on ABSTRACT object relation displays and reports.

If your application does not include any query definitions, you need not execute this part of the initialization process.

The LOADQRY command is automatically run as part of the LOADXREF TYPE(*ALL) job stream. You can invoke the LOADQRY command from any command entry line, or by using option 9 from the APINZ menu. The command can also be run from within a batch or interactive program. The syntax for the LOADQRY command is shown below:

```
                   .-*CURLIB/------.   .-*ALL----------.          .-*PRV---.
>>--LOADQRY----QRY-+---------------+---+---------------+---DTASET-+--------+----------><
                   +-*USRLIBL/-----+   '-*generic_name-'          +-*FIRST-+
                   +-*ALLUSR/------+                              '-name---'
                   +-*ALL/---------+
                   '-*library_name-'
```

QRY Parameter

Specify the qualified name of the query definition(s) to be analyzed. A generic name (leading characters followed by an asterisk) can be specified to select objects with similar names. Specify a library name to be searched for the query(s) indicated or use one of the following special values:

CURLIB: Your job's current library will be searched for queries that meet the object name part of the search criteria.

***USRLIBL:** The user portion of your job's library list will be searched for queries that meet the object name part of the search criteria.

***ALLUSR:** All user libraries will be searched for queries that meet the object name part of the search criteria.

***ALL:** All libraries on the system will be searched for queries that meet the object name part of the search criteria.

The name portion of the QRY parameter specifies the queries that will be analyzed.

ALL: Loads all queries in the specified library(s).

Name: specify the name of an individual query, or use a generic name (one to nine characters suffixed with an asterisk, e.g. CUST*) to indicate that all queries sharing the prefix should be loaded.

DTASET Parameter

Specifies the name of the data set you want to load information into. You can accept the default or choose a separate data set, created by the Add Data Set (ADDDTASET) command, to contain the information. If the data set does not exist, an error will occur. The list of current data set names can be accessed by prompting the command and then prompting the data set parameter.

<u>***PRV:**</u> Use the value for this parameter from the previous session. If no previous session has occurred, use *FIRST.

***FIRST:** all information should be stored in the default data set.

Name: the data set you want to use.

Examples

```
LOADQRY *ALL
```

Use this command to search your current library (*CURLIB) and load all the queries in it into the cross references.

```
LOADQRY PRDLIB/CUST*
```

The library PRDLIB will be searched for queries beginning with CUST. ABSTRACT will find queries such as CUSTQ1, CUSTQRY, etc. The cross reference dictionary will be loaded so that you will be able to view the files that each query uses through the object relations displays and reports.

```
LOADQRY *ALLUSR/*ALL
```

All the queries in all user libraries will be loaded into the cross references.

Load Job Description (LOADJOBD) Command

This command accesses job description objects and places the following information into the cross reference files:

Job queue and output queue
Print device
User profile

Although it is not specifically listed on the APINZ menu, it is automatically run as part of the LOADXREF TYPE(*ALL) job stream.

You can invoke the LOADJOBD command from any command entry line. The command can also be run from within a batch or interactive program. The syntax for the command is shown below:

```
                      .-*CURLIB/------.   .-*ALL----------.           .-*PRV---.
>>--LOADJOBD----JOBD-+---------------+---+---------------+---DTASET-+--------+---------><
                      +-*USRLIBL/-----+   '-*generic_name-'           +-*FIRST-+
                      +-*ALLUSR/------+                               '-name---'
                      +-*ALL/---------+
                      '-*library_name-'
```

JOBD Parameter

Specify the qualified name of the job description(s) to be analyzed. A generic name (leading characters followed by an asterisk) can be specified to select objects with similar names. Specify a library name to be searched or use one of the following special values:

<u>*LIBL:</u> Your job's library list will be searched for job descriptions that meet the object name part of the search criteria.

***CURLIB:** Your job's current library will be searched for job descriptions that meet the object name part of the search criteria.

***USRLIBL:** The user portion of your job's library list will be searched for job descriptions that meet the object name part of the search criteria.

***ALLUSR:** All user libraries will be searched for job descriptions that meet the object name part of the search criteria.

***ALL:** All libraries on the system will be searched for job descriptions that meet the object name part of the search criteria.
The name portion of the JOBD parameter specifies the job descriptions that will be analyzed.

<u>*ALL:</u> Loads all job descriptions in the specified library(s).

Name: specify the name of an individual job description, or use a generic name (one to nine characters suffixed with an asterisk, e.g. CUST*) to indicate that all job descriptions sharing the prefix should be loaded.

DTASET Parameter

Specifies the name of the data set you want to load information into. You can accept the default or choose a separate data set, created by the Add Data Set (ADDDTASET) command, to contain the information. If the data set does not exist, an error will occur. The list of current data set names can be accessed by prompting the command and then prompting the data set parameter.

<u>*PRV:</u> Use the value for this parameter from the previous session. If no previous session has occurred, use *FIRST.

***FIRST:** all information should be stored in the default data set.

Name: the data set you want to use.

Examples

```
LOADJOBD *ALL/*ALL
```

Use this command to load all the job descriptions on your system into the cross references.

```
LOADJOBD PRODLIB/*ALL DTASET(PRODUCTION)
```

The job descriptions in the PRODLIB library will be analyzed. The cross reference information in the PRODUCTION data set will be updated.

Load Subsystem Description (LOADSBSD) Command

This command accesses subsystem description objects and places the routing data information into the cross reference files.

Although it is not specifically listed on the APINZ menu, the LOADSBSD command is automatically run as part of the LOADXREF TYPE(*ALL) job stream.

You can invoke the LOADSBSD command from any command entry line. The command can also be run from within a batch or interactive program. The syntax for the command is shown below:

```
                     .-*CURLIB/------.   .-*ALL----------.          .-*PRV---.
>>--LOADSBSD----SBSD-+---------------+---+---------------+---DTASET-+--------+---------><
                     +-*USRLIBL/-----+   '-*generic_name-'          +-*FIRST-+
                     +-*ALLUSR/------+                              '-name---'
                     +-*ALL/---------+
                     '-*library_name-'
```

SBSD Parameter

Specify the qualified name of the subsystem description(s) to be analyzed. A generic name (leading characters followed by an asterisk) can be specified to select objects with similar names. Specify a library name to be searched or use one of the following special values:

<u>*LIBL:</u> Your job's library list will be searched for subsystem descriptions that meet the object name part of the search criteria.

***CURLIB:** Your job's current library will be searched for subsystem descriptions that meet the object name part of the search criteria.

***USRLIBL:** The user portion of your job's library list will be searched for subsystem descriptions that meet the object name part of the search criteria.

***ALLUSR:** All user libraries will be searched for subsystem descriptions that meet the object name part of the search criteria.

***ALL:** All libraries on the system will be searched for subsystem descriptions that meet the object name part of the search criteria.

The name portion of the SBSD parameter specifies the subsystem descriptions that will be analyzed.

<u>*ALL:</u> Loads all subsystem descriptions in the specified library(s).

Name: specify the name of an individual subsystem description, or use a generic name (one to nine characters suffixed with an asterisk, e.g. CUST*) to indicate that all subsystem descriptions sharing the prefix should be loaded.

DTASET Parameter

Specifies the name of the data set you want to load information into. You can accept the default or choose a separate data set, created by the Add Data Set (ADDDTASET) command, to contain the information. If the data set does not exist, an error will occur. The list of current data set names can be accessed by prompting the command and then prompting the data set parameter.

<u>*PRV:</u> Use the value for this parameter from the previous session. If no previous session has occurred, use *FIRST.

***FIRST:** all information should be stored in the default data set.

Name: the data set you want to use.

Examples

```
LOADSBSD *ALL/*ALL
```

Use this command to load all the subsystem descriptions on your system into the cross references.

```
LOADSBSD PRODLIB/*ALL DTASET(PRODUCTION)
```

The subsystem descriptions in the PRODLIB library will be analyzed. The cross reference information in the PRODUCTION data set will be updated.

Load User Profile (LOADUSRPRF) Command

This command accesses user profile objects and places the following information into the cross reference files:

Initial program or menu
Message queue and output queue
Job description
Attention handling program

Note: *This command is not run automatically in the LOADXREF TYPE(*ALL) job stream. If you want ABSTRACT to analyze your user profile definitions, you must specifically run the LOADUSRPRF command.*

You can invoke the LOADUSRPRF command from any command entry line. The command can also be run from within a batch or interactive program. The syntax for the LOADUSRPRF command is shown below:

```
                            .-*ALL----------.           .-*PRV---.
>>--LOADUSRPRF----USRPRF-+---------------+---DTASET-+--------+------------------------><
                            '-*generic_name-'           +-*FIRST-+
                                                        '-name---'
```

USRPRF Parameter

Specify the name of the user profile(s) to be analyzed. A generic name (leading characters followed by an asterisk) can be specified to select objects with similar names.

***ALL:** All the user profiles on your system will be analyzed.

Name: specify the name of an individual user profile or use a generic name (one to nine characters suffixed with an asterisk, e.g. CUST*) to indicate that all profiles sharing the prefix should be loaded.

DTASET Parameter

Specifies the name of the data set you want to load information into. You can accept the default or choose a separate data set, created by the Add Data Set (ADDDTASET) command, to contain the information. If the data set does not exist, an error will occur. The list of current data set names can be accessed by prompting the command and then prompting the data set parameter.

***PRV:** Use the value for this parameter from the previous session. If no previous session has occurred, use *FIRST.

***FIRST:** all information should be stored in the default data set.

Name: the data set you want to use.

Examples

```
LOADUSRPRF *ALL
```

Use this command to load all the user profiles on your system into the cross references.

```
LOADUSRPRF ORDER* DTASET(ORDERAPP)
```

The user profiles beginning with the characters ORDER will be analyzed. The cross reference information in the ORDERAPP data set will be updated.

Load Job Scheduler (LOADJOBSCD) Command

This command accesses information in your job scheduler and places it into the cross reference files.

Note: *This command is not run automatically in the LOADXREF TYPE(*ALL) job stream. If you want ABSTRACT to analyze information in your job scheduler, you must specifically run the LOAD-JOBSCD command.*

You can invoke the LOADJOBSCD command from any command entry line. The command can also be run from within a batch or interactive program. The syntax for the LOADJOBSCD command is shown below:

```
                      .-*PILOT--.          .-PILOT-.          .-*PRV---.
>>--LOADJOBSCD----PRDID-+---------+---PRDLIB-+-------+---DTASET-+--------+------------><
                      '-*ROBOT -'          '-name--'          +-*FIRST-+
                                                              '-name---'
```

PRDID Parameter

Specify the name of the job scheduling product information to be analyzed.

***SCHEDULER:** Job scheduling definitions from the SCHEDULER[1] product will be loaded into the cross references.

***PILOT:** Job scheduling definitions from the PILOT product will be loaded into the cross references.

***ROBOT:** Job scheduling definitions from the ROBOT[2] product will be loaded into the cross references.

***JOBSCDE:** Job scheduling definitions from the OS/400 job scheduler (WRKJOBSCDE) will be loaded into the cross references.

PRDLIB Parameter

Specify the name of the library containing the job scheduling definitions. For the OS/400 job scheduler, this value is not necessary to find the job definitions and is only used to record a ‘dummy’ parent library for them on subsequent object relation requests.

SCHEDULER: The SCHEDULER library will be searched for job scheduling definitions.

DTASET Parameter

Specifies the name of the data set you want to load information into. You can accept the default or choose a separate data set, created by the Add Data Set (ADDDTASET) command, to contain the information. If the data set does not exist, an error will occur. The list of current data set names can be accessed by prompting the command and then prompting the data set parameter.

1. OpCenter SCHEDULER and PILOT are trademarks of Help/Systems
2. ROBOT is a trademark of Help/Systems, Inc.

***PRV:** Use the value for this parameter from the previous session. If no previous session has occurred, use *FIRST.

***FIRST:** all information should be stored in the default data set.

Name: the data set you want to use.

Examples

```
LOADJOBSCD PRDID(*PILOT) PRDLIB(PILOT20)
```

Use this command to load all the PILOT job scheduling definitions in the PILOT20 library into the cross references

Load Menu Manager (LOADMNUMGR) Command

This command accesses information in your menu managing application and places it into the cross reference files.

Note: *This command is not run automatically in the LOADXREF TYPE(*ALL) job stream. If you want ABSTRACT to analyze information in your menu manager, you must specifically run the LOADMNUMGR command.*

You can invoke the LOADMNUMGR command from any command entry line. The command can also be run from within a batch or interactive program. The syntax for the LOADMNUMGR command is shown below:

```
                                                                .-*PRV---.
>>--LOADMNUMGR----PRDID-+-*PRMS-+---PRDLIB-+-name-+---DTASET-+--------+----------------><
                                                                +-*FIRST-+
                                                                '-name---'
```

PRDID Parameter

Specify the name of the menu managing product information to be analyzed.

***PRMS:** Menu definitions from the CA-PRMS[1] product will be loaded into the cross references.

PRDLIB Parameter

Specify the name of the library containing the menu definitions.

DTASET Parameter

Specifies the name of the data set you want to load information into. You can accept the default or choose a separate data set, created by the Add Data Set (ADDDTASET) command, to contain the information. If the data set does not exist, an error will occur. The list of current data set names can be accessed by prompting the command and then prompting the data set parameter.

***PRV:** Use the value for this parameter from the previous session. If no previous session has occurred, use *FIRST.

***FIRST:** all information should be stored in the default data set.

Name: the data set you want to use.

Examples

```
LOADMNUMGR PRDLIB(RMSMNUMGR)
```

Use this command to load all the PRMS menu definitions in the RMSMNUMGR library into the cross references.

1. CA-PRMS is a trademark of Computer Associates, Inc.

Reorganize Cross-Reference Files (RGZXREF)

This command will compress the cross reference files and remove deleted records. Each time a LOADxxxx command is run, existing cross reference records that match the object criteria are deleted before the new ones are added in. Physical files in the ABSTRACT dictionary are created with the reuse deleted records attribute so that the deleted records are reused. If you choose to change these files so that deleted records are not reused, you should use this command on a regular basis to remove the empty space that will build up over time. Unless you change the files to REUSEDLT(*NO) you will not need to use this command.

Access the ABSTRACT reorganize function by selecting option 12 from the APINZ menu, or by executing the RGZXREF command. The command can also be run from within a batch or interactive program.

```
                         .-*PRV---.
>>--RGZXREF----+-DTASET-+--------+----------------------------------------------------><
                         +-*FIRST-+
                         '-name---'
```

DTASET Parameter

Specifies the name of the data set you want to load information into. You can accept the default or choose a separate data set, created by the Add Data Set (ADDDTASET) command, to contain the information. If the data set does not exist, an error will occur. The list of current data set names can be accessed by prompting the command and then prompting the data set parameter.

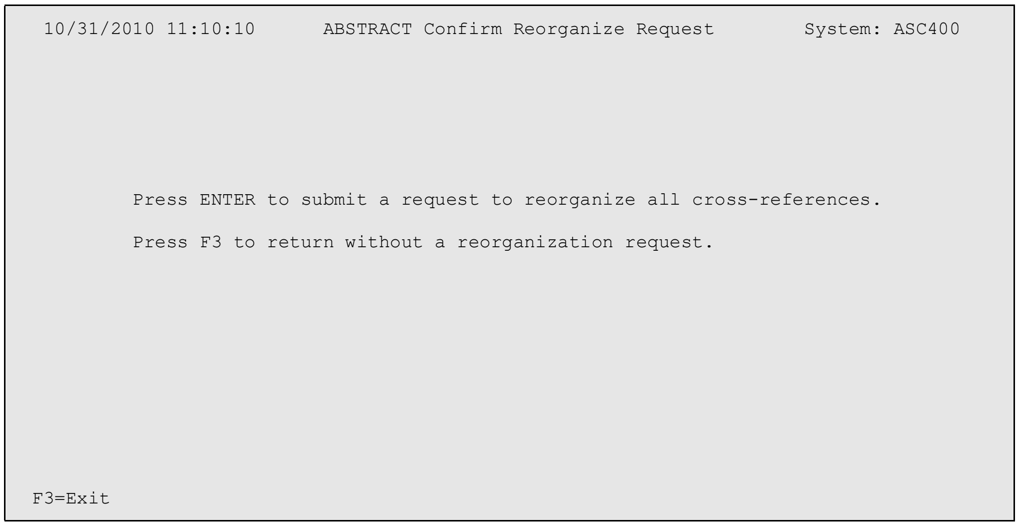
***PRV:** Use the value for this parameter from the previous session. If no previous session has occurred, use *FIRST.

***FIRST:** information in the first (default) data set should be reorganized.

Name: the data set you want to reorganize.

Confirmation Display

If the command is run interactively, or option 11 is selected from the APINZ menu, a display will appear informing the user that a re-organization request is about to be submitted to the batch subsystem. The display looks like the one on the next page.



```
 10/31/2010 11:10:10      ABSTRACT Confirm Reorganize Request        System: ASC400




         Press ENTER to submit a request to reorganize all cross-references.

         Press F3 to return without a reorganization request.







F3=Exit
```

When this display appears, you will have the option to continue or to cancel the reorganize request.

Because the cross references can grow to be quite large, RGZXREF may take an extended time to complete. While it is running, it will lock other users out of the ABSTRACT dictionary files. You should only execute this command during non peak-usage times.

Press Enter to submit the reorganization job to the batch subsystem identified by the job queue listed in the default job description. A completion message will be sent to your workstation when the job finishes. Function key 18 can be pressed from the menu to view and change the job description that will be used.

Press F3 to exit without submitting a reorganization request.

Add Cross-Reference Data Set (ADDDTASET)

ABSTRACT makes it easy to keep the documentation for several application sets independent. Whether you want to make a distinction between production and development applications, or to segregate the documentation for each application into different databases, ABSTRACT can accomplish your objective by allowing you to specify a data set name when the initialization request is made. Information in the various data sets will remain independent throughout the analysis, reporting, and development phases of ABSTRACT use. Separate data sets for cross reference information make the job of documenting and analyzing separate applications considerably easier since you won't become confused about which objects belong to which application.

Use this command to create a cross reference data set. Once a data set has been created, you can place information into it using the various ABSTRACT LOADxxx commands. Then you can use the analysis and reporting functions to access the information in it without getting it mixed with information in other cross reference data sets.

Access the Add Cross Reference Data Set (ADDDTASET) command from any command entry line. The command can also be run from within a batch or interactive program.

```
>>--ADDDTASET----+-DTASET-+-dataset_name-+------------------------------------------><
```

DTASET Parameter

Specifies the name of the data set you want to create. If the data set already exists, an error will occur.

dataset-name: specify a valid name for the data set you are creating. Members with this name will be added to the ABSTRACT cross reference files. Refer to the Control Language Reference Guide for details of naming rules for file members.

Example

```
ADDDTASET DEVELOPMNT
```

This command will create a new data set, called DEVELOPMNT, that can be used by subsequent LOADXREF and analysis commands.

Clear Cross-Reference Data Set (CLRDTASET)

The Clear Data Set (CLRDTASET) command can be used to remove all the information that is loaded into a particular ABSTRACT data set. It is equivalent to, yet faster than, a Remove Cross Reference (RMVXREF) command that specifies all objects in all libraries (see page 2-56). When the command completes, the data set will still exist, but it will not contain any data.

The command can be entered from a command entry line or run within a program. Because the command may require an extended amount of time to complete, you should usually submit it to a batch subsystem for execution.

The syntax for the CLRDTASET command is shown below:

```
                             .-*FIRST-------.
>>--CLRDTASET----+-DTASET-+--------------+-------------------------------------------><
                             '-dataset_name-'
```

DTASET Parameter

Specifies the name of the data set you want to clear. If the data set does not exist, an error will occur.

***FIRST:** the first (default) data set will be erased.

Name: specify a valid name for the data set you want cleared.

Example

```
CLRDTASET *FIRST
```

This command will remove all the cross reference information loaded into the first data set. When finished, the default data set will still exist, yet it will have no information in it.

Remove Cross-Reference Data Set (RMVDTASET)

The Remove Data Set (RMVDTASET) command can be used to remove a data set (and the information it contains) from the cross reference dictionary. It is the logical opposite of the Add Data Set command. When the command completes, the data set will not exist.

The command can be entered from a command entry line or run within a program. Because the command may require an extended amount of time to complete, you should usually submit it to a batch subsystem for execution.

The syntax for the RMVDTASET command is shown below:

```
>>--RMVDTASET----+-DTASET-+-dataset_name-+------------------------------------------><
```

DTASET Parameter

Specifies the name of the data set you want to remove. If the data set does not exist, an error will occur.

dataset-name: specify a valid name for the data set you want deleted.

Example

```
RMVDTASET PAYROLL
```

This command will remove the PAYROLL data set from the cross reference dictionary. Any information loaded into the data set will be lost.

Using ABSTRACT List Displays

ABSTRACT provides four major functions: object references (objects used by another object), object usage (object where used), file analysis, and program creation. Functions that provide object reference and object usage information are also known as object relation commands. Most of the ABSTRACT cross-referencing functions are provided by its object relation commands.

All of the object relationship functions are presented on list displays. These CUA compliant displays provide you with a true object-action mechanism of working with the various elements of your applications software.

Every one of the list displays functions in a similar manner. A consistent set of options and function keys are available regardless of the type of information presented on your display. This makes ABSTRACT easy to use - once you master the concepts involved with the list displays, you will feel at home on any of the various ABSTRACT panels.

This chapter will explain the features and functions of the list displays. The examples use displays generated using the Work With Object References (WRKOBJR) command, although the other object relationship displays work in a similar fashion. Detailed information about functions that are specific to individual commands is located in Part 4, Object Relations.

The list displays are accessed by using any of the ABSTRACT Work With ... commands, or by selecting an equivalent option from the object references (APOBJR) or object usage (APOBJU) menus. The commands that present list displays are:

WRKOBJR	Work With Object References
WRKOBJRS	Work With Sequenced Object References
WRKOBJRX	Work With Cross Referenced Object References
WRKOBJU	Work With Object Usage
WRKOBJUX	Work With Cross Referenced Object Usage
WRKFGU	Work With File Group Usage
WRKFGR	Work With File Group References
WRKFLDGU	Work With Field Group Usage

Each time you use one of the list displays, ABSTRACT saves the values that describe your environment preferences. They will be used automatically if *PRV (previous value) is specified on the command parameters the next time you use ABSTRACT. *PRV is the default for most parameters of the object relations commands.

If, for example, you exit the list display and have omitted format references from the list (DSPFMT(*NO)), they will remain suppressed the next time you use an object relation command if you specify (or accept the default) *PRV for the DSPFMT Keyword.

Elements Of The List Display

The list display presented by the WRKOBJR command shows the objects that your selection criteria has chosen, and any related information from the cross references that were built during the ABSTRACT initialization phase. An example of the type of display presented by WRKOBJR is shown below.

```
210/31/2010 16:09:26   Work with Object References (WRKOBJR)     System: ASC400

Data Set . . *FIRST                          Position to name . . . MENUC
                                             Position to type . . . *PGM
                                             Position to library. . PILOT
Type option, then press Enter.
  1=Select  3=Copy  4=Delete  7=Rename  11=Move
                                                Library or
Opt Object/Uses          Type        Usage      Qualifier  Text
___ MENUC                *PGM CLP               PILOT      Pilot menu driver
___  ADDLIBLE            *CMD        ADDLIBLE   QSYS       Add LibraryList
___  ASC#PL              *DTAARA     RTVDTAARA  PILOT
___  CONTROL             *PGM CLP    CALL       PILOT      Monitor control inf
___   CONTROL            *FILE DSPF  InpOut     PILOT      Control information
___    CONTRL01          *FMT
___   EXTEND             *DTAARA     Unknown    PILOT
___   HLP001             *PGM RPG    CALL       PILOT      Help text processor
___    HELPTXT           *FILE PF    Inp        PILOT
___      LINTX           Chr 78      Inp
___    HLP001            *FILE DSPF  InpOut     PILOT

Parameters or command
===> _____________________________________________________________________
F3=Exit  F4=Prompt  F9=Retrieve  F10=Actions  F23=More Options  F24=More Keys
```

The current date, time, and system name are presented at the top of the display. They are updated each time the Enter key is pressed.

The ABSTRACT data set named at the upper left is the one used in accessing the cross reference information. It was specified (or defaulted) on the WRKOBJR command. You can change the data set by pressing function key 3 and making another WRKxxxx request that specifies a different data set name.

Entry fields in the upper right portion of the display can be used in repositioning the list. Refer to the following section for more information.

If the list has been restricted to a single object type (programs, files, etc.) the title and column headings on the display will indicate the type of object references and the position to type field at the upper right will not be entry capable.

Parent objects

The body of the display presents object relationship information in alphabetical order for each object in the list specified by the LIB, OBJ, OBJTYPE, and OBJATR values of your request. These parent objects are listed at the leftmost (not indented) positions of the list. They are ordered alphabetically by type, just as in a DSPLIB listing.

Depending on the type of function you have requested, the list of parent objects may be originating from the cross reference dictionary or may be real objects (obtained in real time) from the library(s) you have specified on the command.

Use the table below as a guide for the source of the list of objects:

Real-time Object List	Cross Reference Object List
WRKOBJR	WRKOBJRX
WRKOBJU	WRKOBJUX
WRKOBJRS	WRKFGR
WRKFGU	
WRKFLDGU	

If you are using one of the real-time object relation commands, you may have specified *ALLUSR or *ALL for the library name. Libraries within the list are presented in alphabetical order. If *LIBL or *USRLIBL was specified, libraries in the list are presented in the order that they occur within your current library list.

Indented objects

Items that are related to the objects meeting your selection criteria are shown (indented) on the display beneath their parent object. Indented objects are ordered alphabetically by type, just as in a DSPLIB listing. The object type, usage, and text are also shown for each parent and related item.

Object reference displays (like the one above) show the objects that are referenced by parent objects in the list. The indented object is used by the parent.

Object usage displays show the counterpart to the reference display - the indented objects are those that reference the parent object. The parent objects are used by those indented beneath them.

For instance, the Work With Object References (WRKOBJR) display above shows that a CL program named MENUC in the PILOT library references a data area named ASC#PL and calls a CL program named CONTROL (also in the PILOT library). The MENUC program may reference additional objects too. The roll keys could be used to access following pages in the list.

The CL program named CONTROL (called by MENUC) uses a display file named CONTROL and performs both input and output to a particular format (CONTROL01) in it. The CONTROL program also accesses a data area named EXTEND and calls an RPG program named HLP001.

The RPG program named HLP001 gets input from a physical file named HELPTXT. It accesses the LINTX field from the TEXT format in the file. Other fields in the file (not shown in the list) are not referenced by the program.

Object information

ABSTRACT can display object type, attribute, text, usage and an extended description for each item in the object list. Unfortunately, due to the constraints of your workstation, you will not be able to see all of this information at one time. ABSTRACT provides three different views of the object list in 80 column mode and two types of display in the 132 column mode.

Object resolution

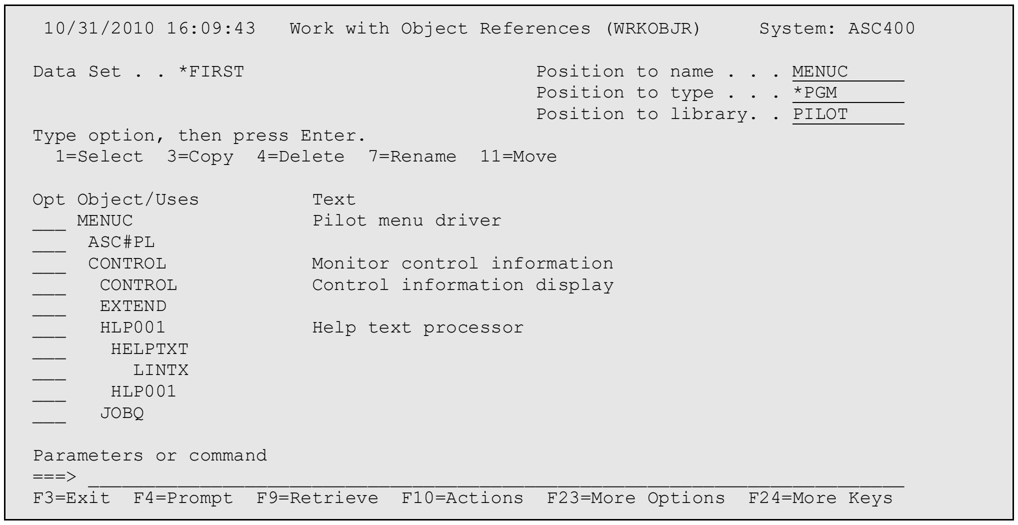
The object subtype (RPG, PF, etc.), text, and the library shown in the Library column for the referenced items on the display is acquired on a real-time basis according to the following search algorithm:

If the cross reference item is qualified, search the qualifying library name.
If the item cannot be found in the indicated library, or if it is unqualified or qualified by *LIBL, search the library of the parent object shown on the display.
If not found, search the current job's library list.
Finally, search the cross reference files (FIFO) for any qualified reference to the named object.
Search the library indicated by the qualifier listed in the cross reference file.

If an existing object still cannot be found following this search, the library name and text columns are left blank on the display, and the object type column will not include an attribute.

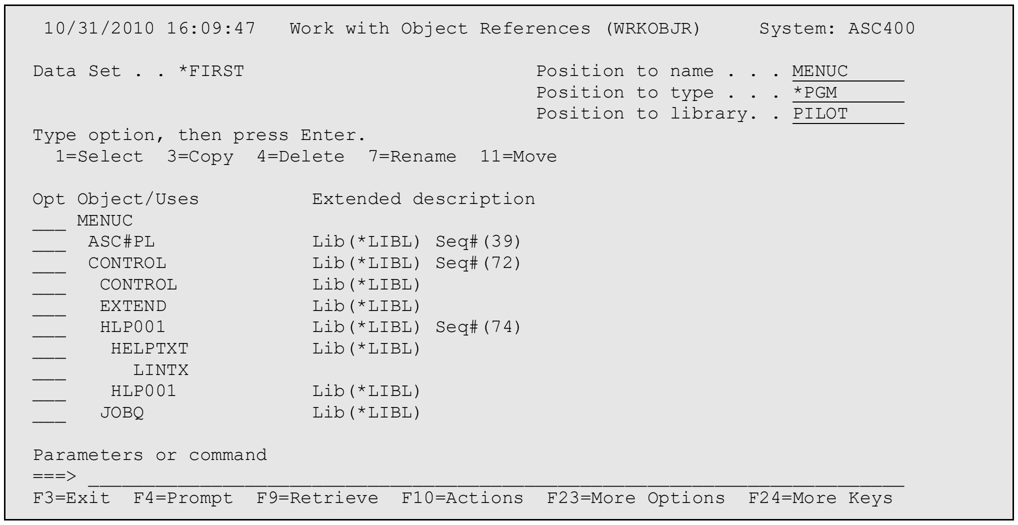
Information display modes

Press function key 11 to switch among the different panel types. The panel on page 3-2 provides an example of an 80 column display showing object type, usage, and a portion of the text. This is always the initial display format that is shown. The object type and usage columns with the complete, 50 character text description can be accessed with function key 11 as shown at the top of the next page.



```
 10/31/2010 16:09:43   Work with Object References (WRKOBJR)     System: ASC400

Data Set . . *FIRST                           Position to name . . . MENUC
                                              Position to type . . . *PGM
                                              Position to library. . PILOT
Type option, then press Enter.
  1=Select  3=Copy  4=Delete  7=Rename  11=Move

Opt Object/Uses          Text
___ MENUC                Pilot menu driver
___  ASC#PL
___  CONTROL             Monitor control information
___   CONTROL            Control information display
___   EXTEND
___   HLP001             Help text processor
___    HELPTXT
___      LINTX
___    HLP001
___   JOBQ

Parameters or command
===> __________________________________________________________________________
F3=Exit  F4=Prompt  F9=Retrieve  F10=Actions  F23=More Options  F24=More Keys
```

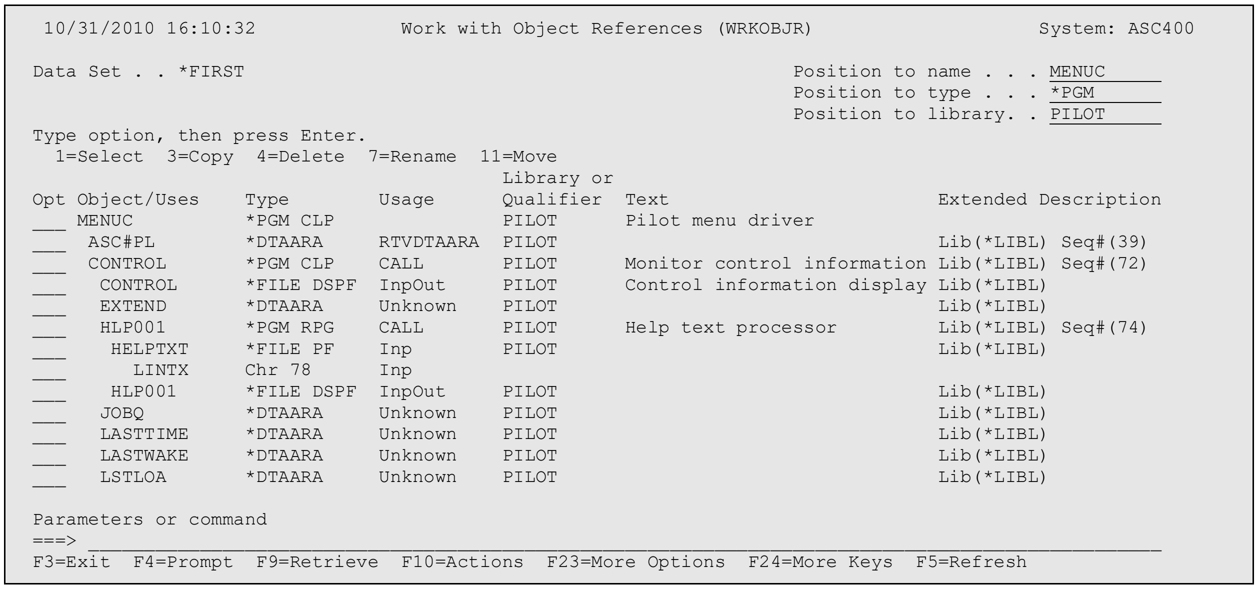
The complete extended description for items in the list can be viewed by pressing function key 11 again. A display similar to the one below will provide more information about the items on the display.



```
 10/31/2010 16:09:47   Work with Object References (WRKOBJR)     System: ASC400

Data Set . . *FIRST                           Position to name . . . MENUC
                                              Position to type . . . *PGM
                                              Position to library. . PILOT
Type option, then press Enter.
  1=Select  3=Copy  4=Delete  7=Rename  11=Move

Opt Object/Uses          Extended description
___ MENUC
___  ASC#PL              Lib(*LIBL) Seq#(39)
___  CONTROL             Lib(*LIBL) Seq#(72)
___   CONTROL            Lib(*LIBL)
___   EXTEND             Lib(*LIBL)
___   HLP001             Lib(*LIBL) Seq#(74)
___    HELPTXT           Lib(*LIBL)
___      LINTX
___    HLP001            Lib(*LIBL)
___   JOBQ               Lib(*LIBL)

Parameters or command
===> __________________________________________________________________________
F3=Exit  F4=Prompt  F9=Retrieve  F10=Actions  F23=More Options  F24=More Keys
```

If you are fortunate enough to be using a 27x132 capable display, you can see more information than will fit on a 24x80 display. If you have selected the 132 column display option (F18), function key

11 can still be used to access the complete text or extended description, but most of each will be shown on the standard display as indicated on the next page.



```
 10/31/2010 16:10:32           Work with Object References (WRKOBJR)                System: ASC400

Data Set . . *FIRST                                                  Position to name . . . MENUC
                                                                     Position to type . . . *PGM
                                                                     Position to library. . PILOT
Type option, then press Enter.
  1=Select  3=Copy  4=Delete  7=Rename  11=Move
                                          Library or
Opt Object/Uses    Type        Usage      Qualifier  Text                       Extended Description
___ MENUC          *PGM CLP               PILOT      Pilot menu driver
___  ASC#PL        *DTAARA     RTVDTAARA  PILOT                                 Lib(*LIBL) Seq#(39)
___  CONTROL       *PGM CLP    CALL       PILOT      Monitor control information Lib(*LIBL) Seq#(72)
___   CONTROL      *FILE DSPF  InpOut     PILOT      Control information display Lib(*LIBL)
___   EXTEND       *DTAARA     Unknown    PILOT                                 Lib(*LIBL)
___   HLP001       *PGM RPG    CALL       PILOT      Help text processor        Lib(*LIBL) Seq#(74)
___    HELPTXT     *FILE PF    Inp        PILOT                                 Lib(*LIBL)
___      LINTX     Chr 78      Inp
___    HLP001      *FILE DSPF  InpOut     PILOT                                 Lib(*LIBL)
___   JOBQ         *DTAARA     Unknown    PILOT                                 Lib(*LIBL)
___   LASTTIME     *DTAARA     Unknown    PILOT                                 Lib(*LIBL)
___   LASTWAKE     *DTAARA     Unknown    PILOT                                 Lib(*LIBL)
___   LSTLOA       *DTAARA     Unknown    PILOT                                 Lib(*LIBL)

Parameters or command
===> _________________________________________________________________________________________________
F3=Exit  F4=Prompt  F9=Retrieve  F10=Actions  F23=More Options  F24=More Keys  F5=Refresh
```

Extended description

To provide the greatest amount of information possible, the ABSTRACT display includes a column labelled Extended description. It contains information that is specific to the type of item referenced. Examples on these pages show how the extended description is presented for each of these types of reference:

Lib( )	If the ABSTRACT object resolution search (page 3-3) located an object in a library other than that indicated by the CL source, the original library qualifier will be listed
Seq( )	The source sequence number in the CL source for the item reference. If the reference occurs several times, multiple sequence numbers will be listed.
Src( )	The source file ABSTRACT analyzed to acquire the information.
Mbr( )	The member named in the CL command (ADDPFM, FMTDTA, OVRDBF, ...) or the HLL source member used during the analysis.
CblFld( )	Long field name for fields used in COBOL programs
FldAtr( )	Alternate descriptions for the field if program described and there is more than one form of the field.
Buffer( )	The starting and ending positions for a field within the indicated format.

Option and command entry

Next to each item in the list is an entry field that can receive an option code and run one or more commands. The option codes that are available for use will be displayed above the item list if the full screen mode (F18) has not been selected. You can cycle through the list by repetitively pressing function key 23.

You can choose several options for several different items on the list. Options will be processed in turn against the objects indicated for them. Press function key 12 to cancel a request and return to the list display without executing any additional options that may be pending.

A command entry line at the bottom of the display allows you to run commands or supply parameter values for the options you have selected. As with option definition statements (see Part 8) multiple commands can be entered with a single transaction if they are separated by a semicolon and a space. Substitution variables can be included in the commands in order to access values from selected objects (1=Select) in the list. Press function key 4 to prompt the command.

Commands you enter on the command line and options you select can be submitted to the batch subsystem using the default job description by pressing function key 14 after typing the command or selecting an option.

Refer to the section below for a complete description of the standard options and how they work.

Function Keys

You can also use various function keys to perform particular actions from the display. The function keys available for each display are listed at the bottom of the display, provided that the Full screen mode has not been selected. You can cycle through the list by repetitively pressing function key 24.

Option and function key definitions

The following tables give a brief description of the standard and reserved options and the function keys available from the object list display. They are explained in detail in the following sections of this chapter.

Reserved options cannot be changed and will always function as indicated, regardless of the contents of the option file that you are using. *Standard options* are defined in the QAUOOPT option file shipped with ABSTRACT. You can investigate (and change) the definition of these options using the option management functions described in Part 8.

The function keys listed in the table will always work as indicated. The key definitions cannot be changed. Function keys 2, 6, and 22 may be defined through options F02, F06, and F22 respectively. Refer to Part 8 for additional information.

Reserved Option	**Description**
1=Select	Chooses items in the list. Once selected, these items can be acted upon by a command (using substitution variables) entered on the command line, or by a function selected through the action bar.
3=Copy	Copy the item(s) using the Create Duplicate Object (CRTDUPOBJ) command.
4=Delete	Delete the object(s) using the appropriate Delete command
7=Rename	Rename the selected object(s) using the Rename Object (RNMOBJ) command.
11=Move	Move the object(s) to a new library.

Standard Option	Description
2=Edit	The Start SEU (STRSEU) command will be run using the source code indicated by the object you have selected. If the selected item is an object (not a source member) its object service data will be used to determine the source file and member that should be edited. If the object is a SEQUEL view or report, the Design View (DSNVIEW) or Design Report (DSNREPORT) command will be used to edit the object. If the object is a message file, the Work With Message Descriptions (WRKMSGD) command will be used.
5=Display	The appropriate display command (DSPPGM, DSPCMD, DSPJOBD, etc.) is run against the object to retrieve information from it. The SEQUEL Data Display (DISPLAY) command will be used for database file objects and SEQUEL views. The Display File Description (DSPFD) command will be used for non-database file objects.
8=Display Description	The object service information is displayed for the selected object. ABSTRACT will use the SEQUEL Display View Description (DSPVIEWD) and Display Report Description (DSPRPTD) commands against view and report data areas.
9=Where used	The ABSTRACT Work With Cross Referenced Object Usage (WRKOBJUX) command will be used to try to locate references to the selected object.
10=Load X-ref	The ABSTRACT Load Cross Reference (LOADXREF) command will be used to place information about the object into the cross reference files.
12=Work with	Work with the object using an appropriate OS/400 WRKxxx command. Source files will be processed by PDM's Work With Members Using PDM (WRKMBRPDM) command. Non-source file objects will be processed by the Analyze File (ANZFILE) command.
13=Change	The appropriate (based on object type) change command (CHGPGM, CHGCMD, etc.) will be prompted so you can change the object. Database files will be changed through a Change Physical File (CHGPF) or Change Logical File (CHGLF) command. If a SEQUEL view is selected, the Change View (CHGVIEW) command will be used.
14=Recompile	Use the ABSTRACT recompilers to recreate a program, file, or command. This function will be automatically submitted to the batch subsystem if the 'Run in batch' value is set to Y.
15=Copy file	Prompt the Copy File (CPYF) command using the file you have selected as the source of the copy.
16=Run	Run the object using an appropriate function. Program, command, DFU, Query, SEQUEL View and Report, and Menu objects can be processed by this option. This function will be automatically submitted to the batch subsystem if the 'Run in batch' value is set to Y.
17=Change w/SDA	Start the IBM Screen Design Aid utility using the display file or menu you selected.
18=Change w/DFU	Start the IBM Data File Utility using the DFU program, or using a default DFU application (if you selected a file object)
19=Object References	Use the ABSTRACT Work With Object References (WRKOBJR) command to process and fully explode the object using DSPLVL(10) EXPLVL(10) values.
25=Find string	Use the string searching capabilities of IBM's Programming Development Manager to search the members in a file.
29=Sequenced Ref	Use the ABSTRACT Work With Sequenced X-ref References (WRKOBJRS) command to present the references made by this object in order by source sequence number. Objects in the list will be fully exploded using DSPLVL(10) EXPLVL(10) values.
39=WRKFLDGU	Display the "where-used" information for the selected field and all fields that are based on it.
49=WRKFGU	Display the "where-used" information for the selected file and all files that are based on it.
90=Signoff	Use the SIGNOFF command to end your interactive session.

Function Key	**Description**
F1=Help	Shows additional information about a display, command, or message. The Help key is available on all displays, yet is usually not shown.
F3=Exit	End the current task and return to the display where you began without processing any options or changes you have entered on the display.
F4=Prompt	Use the command prompter to get assistance for an option you have selected or a command you have typed on the command line.
F5=Refresh	Changes input fields on the display back to their original values and rebuilds the current object list.
F7=Backward	Step one object backward in the list. The object list is repositioned so that the object preceding the one listed at the upper right corner of the display (prior to pressing function key 7) occupies that position.
F8=Forward	Step one object forward in the list. The object list is repositioned so that the object following the one listed at the upper right corner of the display (prior to pressing function key 8) occupies that position.
F9=Retrieve	Access the commands you have used in LIFO (last in first out) order.
F10=Actions	Places the action bar at the top of the display. Once it is available you can access pull-down menus from it.
F11=Alt.Display	Toggles the display between forms showing partial or complete object text and extended description information.
F12=Cancel	Ends the current option and returns to the previous display.
F13=Repeat	All option lines between a pair of selected options are filled with the value indicated by the pair. If an ending point has not been selected, the option is repeated for all objects following the selected one.
F14=Submit	Submit the selected option(s) to the batch subsystem.
F15=Prompted Submit	Prompt the Submit Job (SBMJOB) command prior to submitting the selected option(s) to the batch subsystem.
F16=Options	Displays the Work With User Defined Options display. Refer to *Part 8, Working With Options*, for more information about this function.
F17=Subset	Change the criteria that select objects in the list.
F18=Defaults	Specify the default (or standard) settings used during your session.
F19=Next	Opposite of F9=Retrieve, function key 19 accesses commands in FIFO (first in, first out) order. It can be very handy if you press function key 9 too many times in your attempt to retrieve a specific command.
F20=Top	Moves the object positioned under the cursor to the top of the display.
F21=Print	Uses the OUTPUT(*PRINT) parameter and the current subset criteria to create a printout of the object relationship information.
F23=More options	Cycles the list of available options.
F24=More keys	Cycles the list of available function keys.

Working with previous commands

Each time you execute an option or run command(s) from the command line, they are recorded and can be later retrieved from your job message queue. ABSTRACT allows you to work with commands that you have previously run through two function keys;

F9=RetrieveF19=Next

The list displays, like ABSTRACT menus can access previously executed commands and retrieve them so they can be run again. Press function key 9 to retrieve *backwards* through the list of previously executed commands. Use function key 19 to process the list *forwards*. If you are working backwards in the list and go too far, simply press function key 19 to begin working your way forwards again.

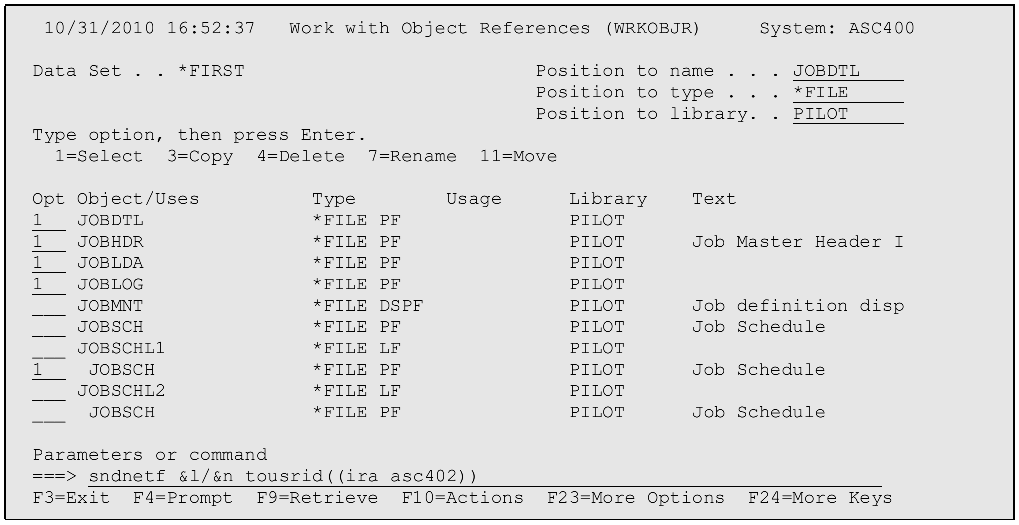
Repeating an option

If you want to run the same option for several items in the list, you can use the option repeat feature. Simply type the option number next to the first object that you want to run the option against and press function key 13 to repeat the option to the end of the list.

If you want the option repeat to stop before the end of the list, place another instance of the option next to the last object you want to include prior to pressing function key 13. The option will be repeated from the first item you have selected up to and including the last item you selected.

Example

The sample display on the next page demonstrates the substitution capabilities of the command line. Using option 1=Select for items on the display will cause the command to be run for each item selected. Substitution values (see Part 8) can be used for each object. In the example below, each file selected will be sent to the user IRA at location ASC402 using the Send Network File (SNDNETF) command. If function key 14 were pressed, the commands would be submitted to batch using the default job description.



```
 10/31/2010 16:52:37   Work with Object References (WRKOBJR)     System: ASC400

Data Set . . *FIRST                          Position to name . . . JOBDTL
                                             Position to type . . . *FILE
                                             Position to library. . PILOT
Type option, then press Enter.
  1=Select  3=Copy  4=Delete  7=Rename  11=Move

Opt Object/Uses          Type        Usage      Library    Text
1__ JOBDTL               *FILE PF               PILOT
1__ JOBHDR               *FILE PF               PILOT      Job Master Header I
1__ JOBLDA               *FILE PF               PILOT
1__ JOBLOG               *FILE PF               PILOT
___ JOBMNT               *FILE DSPF             PILOT      Job definition disp
___ JOBSCH               *FILE PF               PILOT      Job Schedule
___ JOBSCHL1             *FILE LF               PILOT
1__  JOBSCH              *FILE PF               PILOT      Job Schedule
___ JOBSCHL2             *FILE LF               PILOT
___  JOBSCH              *FILE PF               PILOT      Job Schedule

Parameters or command
===> sndnetf &l/&n tousrid((ira asc402))
F3=Exit  F4=Prompt  F9=Retrieve  F10=Actions  F23=More Options  F24=More Keys
```

Repositioning the display

The object list display can be repositioned in several different ways. The contents of the list are not changed through any repositioning requests. List contents can only by changed by choosing a different subset (F17) or by making a different object relationship request.

Roll keys

Use the roll keys to scroll backwards and forwards through the list one page at a time. As you progress forward through the list, it will be extended from the preceding page until the list is completed.

F7=Previous object   F8=Next object

You can also shift the display by repositioning to the next (or previous) parent object in the list. Press function key 8 to redisplay the list beginning with the object after (alphabetically by type) the first item currently on the display. Press function key 7 to step backwards through the list.

F20=Move to top

Use function key 20 to move the item under the cursor to the top line of the item list. This is an especially effective way to see the references for an item in the list without having them split across a display page boundary.

Direct positioning

The three entry fields in the upper right portion of the display allow you to directly position the list beginning with an object (in your selection criteria) that you name.

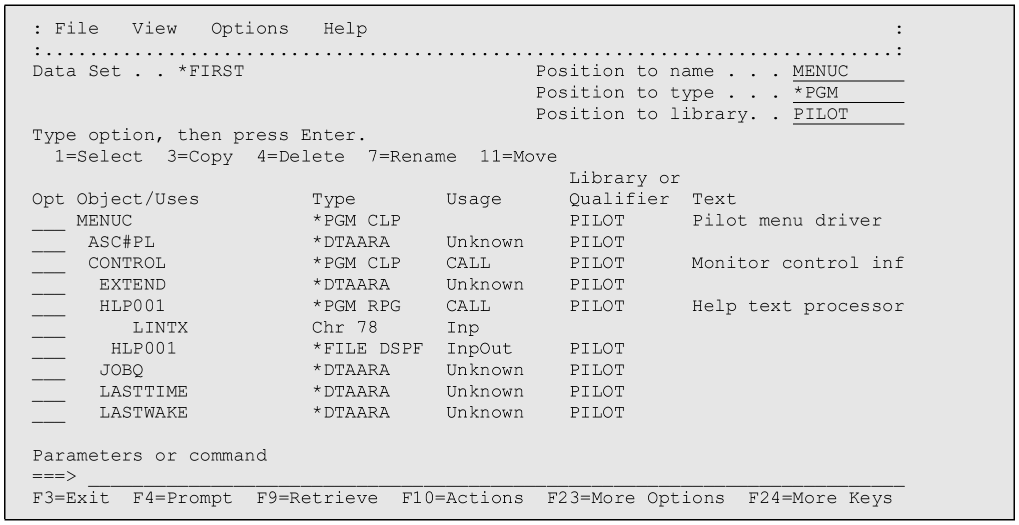
Change the fields and press the Enter key to specify a positioning request. The value you supply for the library name must match one of the libraries in your list exactly. A search will be performed for the first object equal to or after the type and name that you have specified. Use blanks for the type and name to indicate that you want to see the first object in the specified library.

Using the action bar

ABSTRACT allows you to request a CUA-compliant action bar at the top of the display. The action bar will help you to learn about the functions of ABSTRACT and assist you when you are not certain which option to use or function key to press in order to carry out a particular action.

The action bar allows you to access pull-down menus that perform the major functions of ABSTRACT. Once you see a pull down menu on the display, you can select an option from it that will present another menu, or carry out an action.

Using the action bar is easy. Simply remember to select one or more objects using option 1=Select and then press function key 10 to get assistance. A window like the one below will appear.



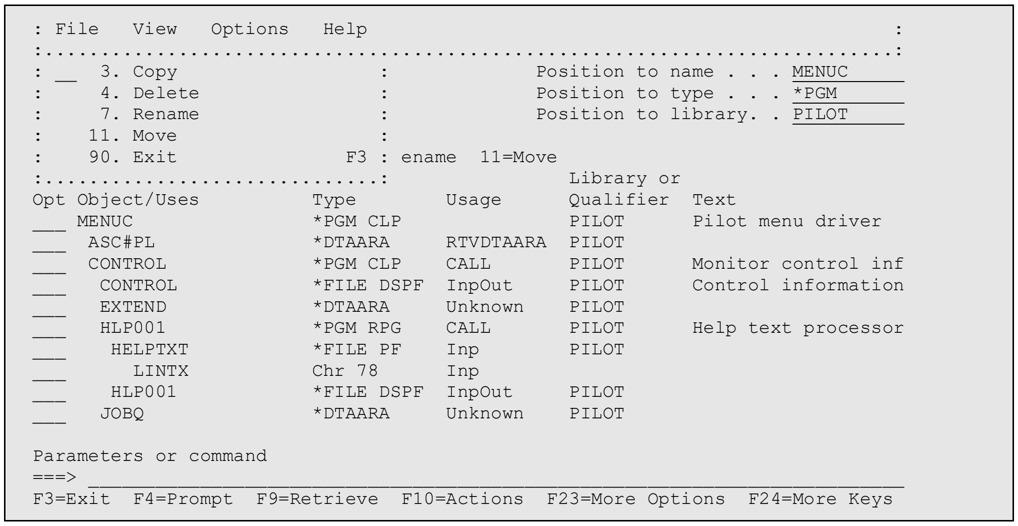
```
: File   View   Options   Help                                              :
:...........................................................................:
Data Set . . *FIRST                          Position to name . . . MENUC
                                             Position to type . . . *PGM
                                             Position to library. . PILOT
Type option, then press Enter.
  1=Select  3=Copy  4=Delete  7=Rename  11=Move
                                                Library or
Opt Object/Uses          Type        Usage      Qualifier  Text
___ MENUC                *PGM CLP               PILOT      Pilot menu driver
___  ASC#PL              *DTAARA     Unknown    PILOT
___  CONTROL             *PGM CLP    CALL       PILOT      Monitor control inf
___   EXTEND             *DTAARA     Unknown    PILOT
___   HLP001             *PGM RPG    CALL       PILOT      Help text processor
___      LINTX           Chr 78      Inp
___    HLP001            *FILE DSPF  InpOut     PILOT
___   JOBQ               *DTAARA     Unknown    PILOT
___   LASTTIME           *DTAARA     Unknown    PILOT
___   LASTWAKE           *DTAARA     Unknown    PILOT

Parameters or command
===> _____________________________________________________________________
F3=Exit  F4=Prompt  F9=Retrieve  F10=Actions  F23=More Options  F24=More Keys
```

Once the action bar has appeared, use the TAB key or the cursor control keys to move the cursor to the desired category (File, View, Options, or Help) and press the Enter key. Although you will be able to position the cursor to (and type into) fields outside the scope of the action bar and pull-down menus, once the action bar appears on the display, input to other fields will be ignored until the action bar is removed. Press function key 12 or function key 10 to remove the action bar and return to the underlying display.

Select the File action if you want to apply an action to the objects you have selected. Choose the View option if you want to change the contents of the object list on the display. The Options action will present the Work With Options display described in Part 8 and allow you to select one or more options for execution. Alternatively, you can use the Help action to access the ABSTRACT help facility.

File actions

The display below shows the pull down menu that can be accessed by positioning the cursor on the File action within the action bar. It allows you to work with the objects you have selected and perform operations against them. Actions on the File menu are limited to the reserved options: Copy, Delete, Move, and Rename.



```
: File   View   Options   Help                                               :
:............................................................................:
: __  3. Copy                :             Position to name . . . MENUC
:     4. Delete              :             Position to type . . . *PGM
:     7. Rename              :             Position to library. . PILOT
:    11. Move                :
:    90. Exit             F3 : ename  11=Move
:............................:                Library or
Opt Object/Uses          Type        Usage      Qualifier  Text
___ MENUC                *PGM CLP               PILOT      Pilot menu driver
___  ASC#PL              *DTAARA     RTVDTAARA  PILOT
___  CONTROL             *PGM CLP    CALL       PILOT      Monitor control inf
___   CONTROL            *FILE DSPF  InpOut     PILOT      Control information
___   EXTEND             *DTAARA     Unknown    PILOT
___   HLP001             *PGM RPG    CALL       PILOT      Help text processor
___    HELPTXT           *FILE PF    Inp        PILOT
___      LINTX           Chr 78      Inp
___    HLP001            *FILE DSPF  InpOut     PILOT
___   JOBQ               *DTAARA     Unknown    PILOT

Parameters or command
===> _________________________________________________________________________
F3=Exit  F4=Prompt  F9=Retrieve  F10=Actions  F23=More Options  F24=More Keys
```

Each option number on the pull-down File menu corresponds to an option available from the main object list display and works in the same fashion. Choose an option from the menu to apply the listed action against objects that were selected by 1=Select before the action bar was displayed.

Selecting an object and then choosing an item from the File pull-down is equivalent to selecting the corresponding option from the object list display. Additional dialog and confirmation displays will be presented once you choose one of the File options. Refer to the following sections describing each action for details about these displays.

Use option 90 to exit the ABSTRACT function and return to the display you were using before starting ABSTRACT. Selecting option 90 from the File menu is equivalent to pressing function key 3 from the object list display.

View actions

If the action bar (function key 10) is present on the display, press the TAB key once and then press Enter to change the look or contents of the object list. The View menu will be pulled down so that you can see and change the settings for the display.

The View menu combines elements of the F17=Subset and F18=Defaults displays in a simple, easy to understand format. It does not provide all of the function available on them, but will allow you to control some of the primary characteristics of the object list display. The pull-down View menu looks like the one below.

```
: File   View   Options   Help                                            :
:.........................................................................:
Data S : _ 1. Subset list...    F17 :        Position to name . . . MENUC
       :                            :        Position to type . . . *PGM
       : 2 1. Show parent only      :        Position to library. . PILOT
Type o :   2. Show relations        :
  1=Se :                            :   11=Move
       : 1 1. Show usage and text   :           Library or
Opt Ob :   2. Show full object text : sage      Qualifier  Text
___ ME :   3. Show full description :           PILOT      Pilot menu driver
___  A :                            : nknown    PILOT
___  C : 1 1. Use 132 columns       : ALL       PILOT      Monitor control inf
___    :   2. Use 80 column display : nknown    PILOT
___    :                            : ALL       PILOT      Help text processor
___    : 2 1. Full screen           : np
___    :   2. Show options and keys : npOut     PILOT
___    :............................: nknown    PILOT
___   LASTTIME          *DTAARA     Unknown    PILOT
___   LASTWAKE          *DTAARA     Unknown    PILOT

Parameters or command
===> _____________________________________________________________________
F3=Exit  F4=Prompt  F9=Retrieve  F10=Actions  F23=More Options  F24=More Keys
```

The View menu gives you control over five aspects of the display and shows their current settings. Change any or all of the items on the display to modify the way the object list display is presented.

The display above shows that ABSTRACT is currently set to show object relations with object usage and text columns. Information will be presented (if possible) on a 27x132 display. Option summary and function key text will also be presented on the display.

Place a ‘1’ next to the first item on the menu to request the list subset window and change the objects that appear as “parent” objects in the list. Once the list subset window appears you will also be able to customize the list contents, choosing the types of objects you want included on the display. The list subset window can also be accessed by pressing function key 17 from the object list display. Refer to page 3-18 for a complete description of the capabilities of the list subset window.

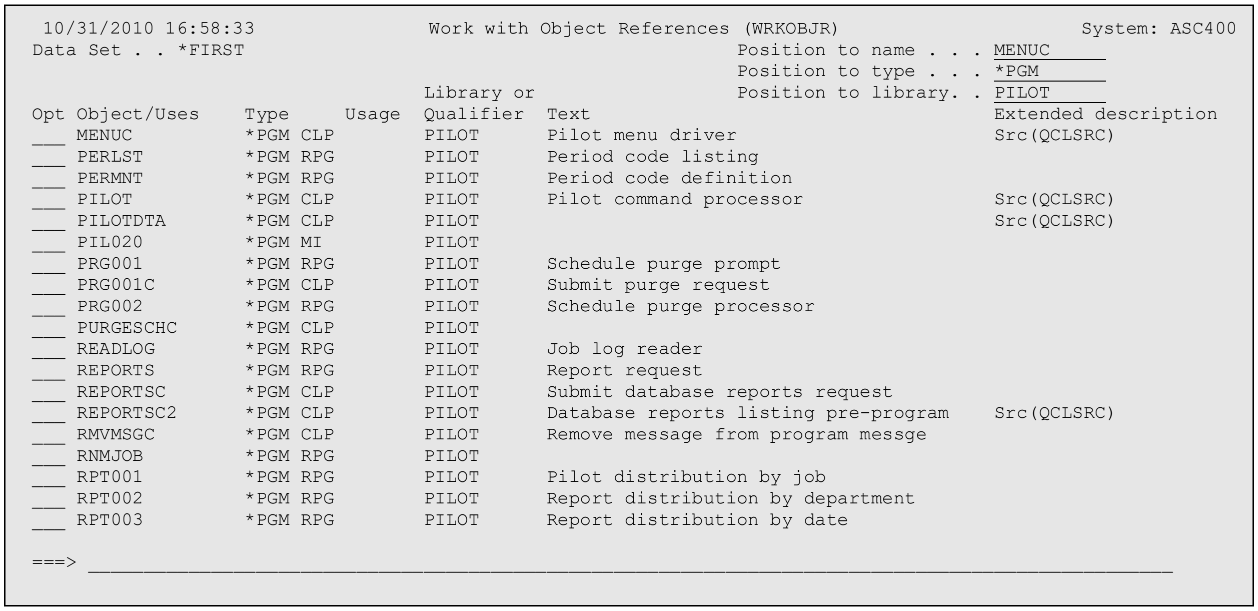
Choose the parent only selection if you want no object relationships to be presented on the display. This is equivalent to specifying DSPLVL(0) on the WRKOBJR command or through the list subset window. Conversely, selecting show object relations is equivalent to specifying DSPLVL(10).

The next section of the View menu allows you to specify the type of information presented in the object list. Choosing one of these options is equivalent to pressing function key 11 (see page) until the display shows the desired type of information. Two or three choices are available, depending on whether you are using 80 or 132 column capability.

The final section of the View menu controls whether option and function key summaries are presented above and below the object list. Selecting the Full screen option allows an additional six lines of the object list to appear on the display by suppressing these summary lines.

Parent objects only

The sample display below is an example of selecting parent only, 132 column, and full screen options. It lists 19 of the objects in the PILOT library, omitting option and function key summaries to provide extra lines on the display.


```
 10/31/2010 16:58:33               Work with Object References (WRKOBJR)                           System: ASC400
Data Set . . *FIRST                                           Position to name . . . MENUC
                                                              Position to type . . . *PGM
                                   Library or                 Position to library. . PILOT
Opt Object/Uses     Type     Usage  Qualifier  Text                                   Extended description
___ MENUC           *PGM CLP        PILOT      Pilot menu driver                      Src(QCLSRC)
___ PERLST          *PGM RPG        PILOT      Period code listing
___ PERMNT          *PGM RPG        PILOT      Period code definition
___ PILOT           *PGM CLP        PILOT      Pilot command processor                Src(QCLSRC)
___ PILOTDTA        *PGM CLP        PILOT                                             Src(QCLSRC)
___ PIL020          *PGM MI         PILOT
___ PRG001          *PGM RPG        PILOT      Schedule purge prompt
___ PRG001C         *PGM CLP        PILOT      Submit purge request
___ PRG002          *PGM RPG        PILOT      Schedule purge processor
___ PURGESCHC       *PGM CLP        PILOT
___ READLOG         *PGM RPG        PILOT      Job log reader
___ REPORTS         *PGM RPG        PILOT      Report request
___ REPORTSC        *PGM CLP        PILOT      Submit database reports request
___ REPORTSC2       *PGM CLP        PILOT      Database reports listing pre-program   Src(QCLSRC)
___ RMVMSGC         *PGM CLP        PILOT      Remove message from program messge
___ RNMJOB          *PGM RPG        PILOT
___ RPT001          *PGM RPG        PILOT      Pilot distribution by job
___ RPT002          *PGM RPG        PILOT      Report distribution by department
___ RPT003          *PGM RPG        PILOT      Report distribution by date

===> _____________________________________________________________________________________________________
```

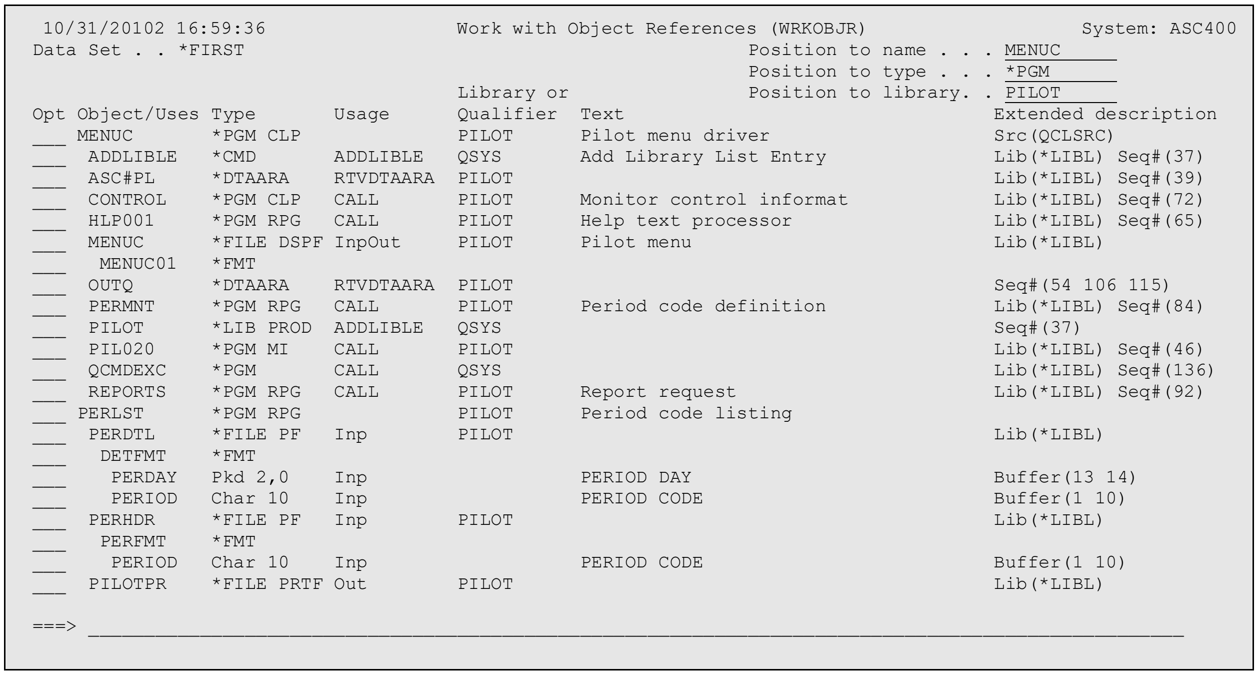
The display shows information similar to what you would see if you used the OS/400 Work With Object (WRKOBJ) command. Each object meeting the criteria specified on the name parameters is listed as a parent. No indented entries are provided because display level zero (DSPLVL(0)) was specified, eliminating all object relations from the display.

Parents plus relations

The next display uses the same object name criteria as the one above but shows object relations for the parent objects on the display. Based on the observation that there are at least two levels of indentation shown on the display, we know that DSPLVL(2) or higher was used. This allowed the ABSTRACT list display to continue providing object relationships for parent objects until the indentation level was reached or no more relationships needed to be displayed.

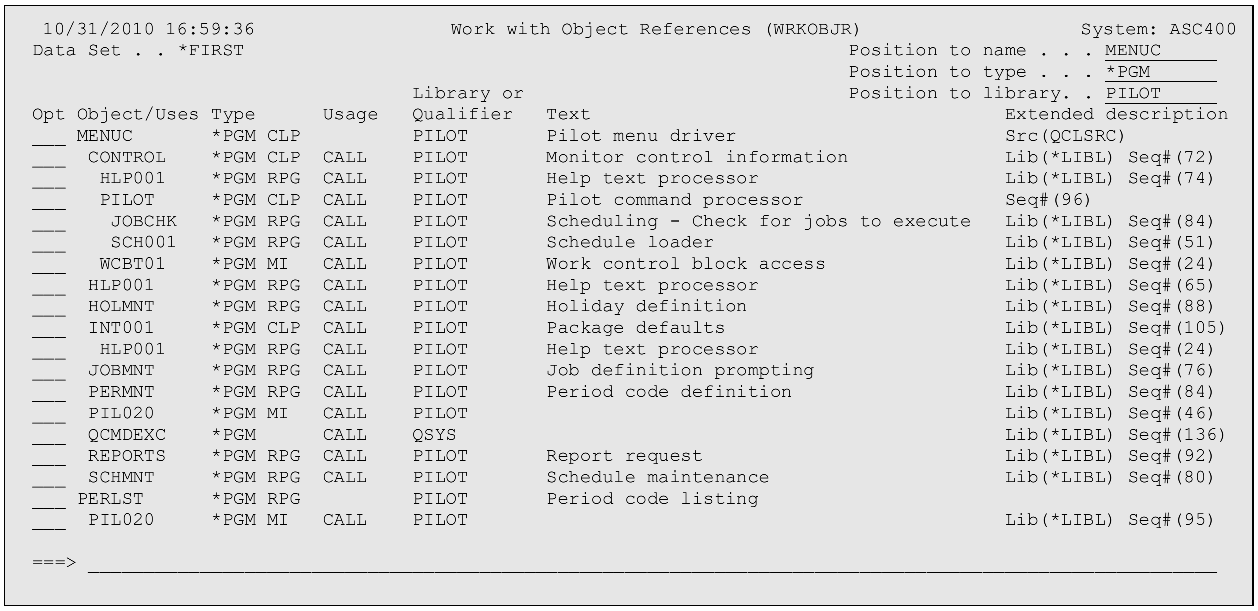
The extra information on the display has greatly expanded the list. Whereas the display above shows 17 parent objects, the display showing object relationships lists only two - MENUC and PERLST.

Although the list display can become quite complex when high levels of indentation are present, it is easy to simplify the display and omit certain types of relationships. Refer to page 3-18 for instructions about these controls.



```
 10/31/20102 16:59:36               Work with Object References (WRKOBJR)                   System: ASC400
Data Set . . *FIRST                                        Position to name . . . MENUC
                                                           Position to type . . . *PGM
                                      Library or           Position to library. . PILOT
Opt Object/Uses Type       Usage      Qualifier  Text                                  Extended description
___ MENUC       *PGM CLP              PILOT      Pilot menu driver                     Src(QCLSRC)
___  ADDLIBLE   *CMD       ADDLIBLE   QSYS       Add Library List Entry                Lib(*LIBL) Seq#(37)
___  ASC#PL     *DTAARA    RTVDTAARA  PILOT                                            Lib(*LIBL) Seq#(39)
___  CONTROL    *PGM CLP   CALL       PILOT      Monitor control informat              Lib(*LIBL) Seq#(72)
___  HLP001     *PGM RPG   CALL       PILOT      Help text processor                   Lib(*LIBL) Seq#(65)
___  MENUC      *FILE DSPF InpOut     PILOT      Pilot menu                            Lib(*LIBL)
___   MENUC01   *FMT
___  OUTQ       *DTAARA    RTVDTAARA  PILOT                                            Seq#(54 106 115)
___  PERMNT     *PGM RPG   CALL       PILOT      Period code definition                Lib(*LIBL) Seq#(84)
___  PILOT      *LIB PROD  ADDLIBLE   QSYS                                             Seq#(37)
___  PIL020     *PGM MI    CALL       PILOT                                            Lib(*LIBL) Seq#(46)
___  QCMDEXC    *PGM       CALL       QSYS                                             Lib(*LIBL) Seq#(136)
___  REPORTS    *PGM RPG   CALL       PILOT      Report request                        Lib(*LIBL) Seq#(92)
___ PERLST      *PGM RPG              PILOT      Period code listing
___  PERDTL     *FILE PF   Inp        PILOT                                            Lib(*LIBL)
___   DETFMT    *FMT
___    PERDAY   Pkd 2,0    Inp                   PERIOD DAY                            Buffer(13 14)
___    PERIOD   Char 10    Inp                   PERIOD CODE                           Buffer(1 10)
___  PERHDR     *FILE PF   Inp        PILOT                                            Lib(*LIBL)
___   PERFMT    *FMT
___    PERIOD   Char 10    Inp                   PERIOD CODE                           Buffer(1 10)
___  PILOTPR    *FILE PRTF Out        PILOT                                            Lib(*LIBL)

===> ___________________________________________________________________________________________________
```

The display on the next page[1] is a simplified version of the information above that suppresses all information except program relationships. The Reference explosion level has been incremented to allow iterative program explosion to occur (c.f. CONTROL, PILOT, and INT001).



```
 10/31/2010 16:59:36               Work with Object References (WRKOBJR)                    System: ASC400
Data Set . . *FIRST                                                 Position to name . . . MENUC
                                                                    Position to type . . . *PGM
                                  Library or                        Position to library. . PILOT
Opt Object/Uses Type      Usage   Qualifier   Text                                        Extended description
___ MENUC       *PGM CLP          PILOT       Pilot menu driver                           Src(QCLSRC)
___  CONTROL    *PGM CLP  CALL    PILOT       Monitor control information                 Lib(*LIBL) Seq#(72)
___   HLP001    *PGM RPG  CALL    PILOT       Help text processor                         Lib(*LIBL) Seq#(74)
___   PILOT     *PGM CLP  CALL    PILOT       Pilot command processor                     Seq#(96)
___    JOBCHK   *PGM RPG  CALL    PILOT       Scheduling - Check for jobs to execute      Lib(*LIBL) Seq#(84)
___    SCH001   *PGM RPG  CALL    PILOT       Schedule loader                             Lib(*LIBL) Seq#(51)
___   WCBT01    *PGM MI   CALL    PILOT       Work control block access                   Lib(*LIBL) Seq#(24)
___  HLP001     *PGM RPG  CALL    PILOT       Help text processor                         Lib(*LIBL) Seq#(65)
___  HOLMNT     *PGM RPG  CALL    PILOT       Holiday definition                          Lib(*LIBL) Seq#(88)
___  INT001     *PGM CLP  CALL    PILOT       Package defaults                            Lib(*LIBL) Seq#(105)
___   HLP001    *PGM RPG  CALL    PILOT       Help text processor                         Lib(*LIBL) Seq#(24)
___  JOBMNT     *PGM RPG  CALL    PILOT       Job definition prompting                    Lib(*LIBL) Seq#(76)
___  PERMNT     *PGM RPG  CALL    PILOT       Period code definition                      Lib(*LIBL) Seq#(84)
___  PIL020     *PGM MI   CALL    PILOT                                                   Lib(*LIBL) Seq#(46)
___  QCMDEXC    *PGM      CALL    QSYS                                                    Lib(*LIBL) Seq#(136)
___  REPORTS    *PGM RPG  CALL    PILOT       Report request                              Lib(*LIBL) Seq#(92)
___  SCHMNT     *PGM RPG  CALL    PILOT       Schedule maintenance                        Lib(*LIBL) Seq#(80)
___ PERLST      *PGM RPG          PILOT       Period code listing
___  PIL020     *PGM MI   CALL    PILOT                                                   Lib(*LIBL) Seq#(95)

===> ______________________________________________________________________________________________________
```

1. Additional lines on the display (HOLMNT, JOBMNT, etc.) were removed from the previous example for illustration purposes.

Options action

Choose the Options item on the action bar to view, run, and change the options defined in the option file. Selecting it is equivalent to pressing function key 16 from the object list display. A pop-up window similar to the one below will appear.

```
 : File   View   Options   Help                                          :
 :........... ...........................................................:
 Data Set . .  :          Work with User-Defined Options               :C
               :                                                       :
               : Type option, then press Enter.                        :T
 Type option,  :   1=Run  2=Change  3=Copy  4=Delete  7=Rename         :
   1=Select  3 :                                                       :
               : Sel Opt  Description                                  :
 Opt Object/Us :  _    2  STRSEU SRCFILE(&SRCLIB/&SRCFIL) SRCMBR(&SRCMB :
 ___ MENUC     :  _    2  SEQUEL/DSNREPORT REPORT(&L/&N) /* Edit */    :iver
 ___  ASC#PL   :  _    2  SEQUEL/DSNVIEW VIEW(&L/&N) /* Edit */        :
 ___  CONTROL  :  _    2  STRSEU SRCFILE(&L/&F) SRCMBR(&N) /* Edit */  :ol inf
 ___  HLP001   :  _    2  WRKMSGD MSGF(&L/&N) /* Edit */               :cessor
 ___  HOLMNT   :  _    5  DSP&S &L/&N /* Display */                    :ition
 ___  INT001   :  _    5  SEQUEL/DISPLAY 'SELECT * FROM &L/&N' /* Displ :lts
 ___  JOBMNT   :  _    5  SEQUEL/REPORT REPORT(&L/&N); DSPSPLF &N SPLNB :n prom
 ___  MENUC    :                                                       :
 ___   MENUC01 : Option file . . . QAUOOPT                             :
 ___  MSGQ     :   Library . . . . ABSTRACT                            :
               : Option member . . QAUOOPT                             :
 Parameters or :                                                       :
 ===> ________ : F1=Help  F4=Prompt  F5=Refresh  F6=Create  F12=Cancel  :
 F3=Exit  F4=P :.......................................................: Keys
```

If you have used 1=Select to select one or more objects from the primary list display prior to requesting the action bar, you can run user-defined options against them. Once the option window is displayed, you can place a 1=Run selection next to each option that you want to run for the objects you chose prior to accessing the option list. Each option will be run in turn for the selected objects.

For more information about working with the option definitions and the user option file, refer to *Part 8, Working With User-defined Options*.

Press function key 12 to return to the object list display, or use function key 3 to close the option window and return to the action bar.

Help actions

The Help action verb allows you to access the ABSTRACT help facility. Choose the Help action to display the pull-down menu shown below. You will be able to use any of the five options on it.

```
 : File   View   Options   Help                                          :
 :.......................................................................:
 Data Set . . *FIRST     : 1 1. Help for help... : ion to name . . . MENUC
                         :   2. Extended help... : ion to type . . . *PGM
                         :   3. Keys help...     : ion to library. . PILOT
 Type option, then press :   4. Help index...    :
   1=Select  3=Copy  4=D :   5. About...         :
                         :.......................: brary or
 Opt Object/Uses          Type        Usage      Qualifier  Text
 ___ MENUC                *PGM CLP               PILOT      Pilot menu driver
 ___  ASC#PL              *DTAARA     Unknown    PILOT
 ___  CONTROL             *PGM CLP    CALL       PILOT2     Monitor control inf
 ___  HLP001              *PGM RPG    CALL       PILOT      Help text processor
 ___  HOLMNT              *PGM RPG    CALL       PILOT      Holiday definition
 ___  INT001              *PGM CLP    CALL       PILOT2     Package defaults
 ___  JOBMNT              *PGM RPG    CALL       PILOT      Job definition prom
 ___  MENUC               *FILE DSPF  InpOut     PILOT      Pilot menu
 ___  MSGQ                *DTAARA     Unknown    APLUS      Default workstation
 ___  OUTQ                *DTAARA     Unknown    ABSTRACT

 Parameters or command
 ===> __________________________________________________________________________
 F3=Exit  F4=Prompt  F9=Retrieve  F10=Actions  F23=More Options  F24=More Keys
```

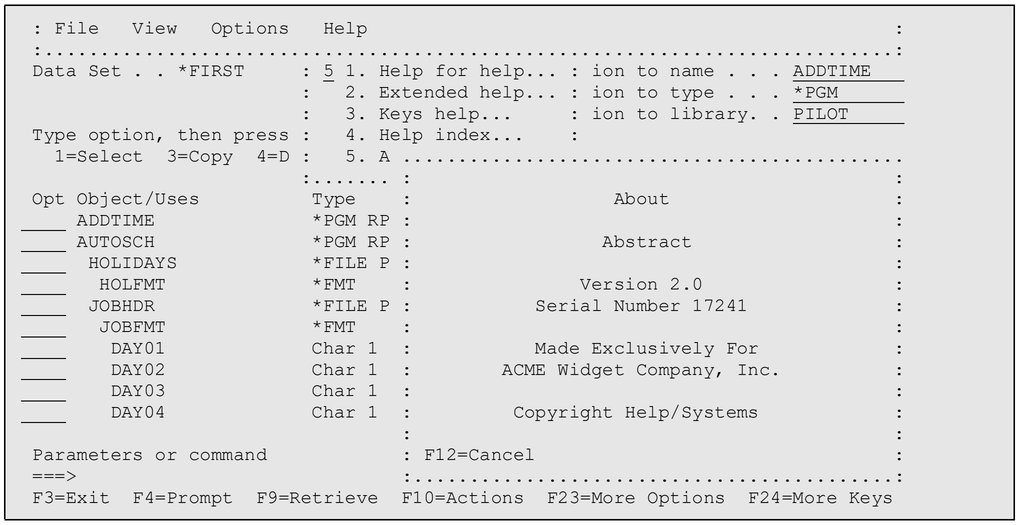
Select the first option to access information about the help facility. When you choose it, the iSeries user support display will appear so that you can learn about the help facility, index search capabilities, etc.

Option 2 of the Help menu will access the extended help for the list panel. Selecting the option is equivalent to moving the cursor to the top line of the display and pressing the help key. Refer to the Introduction for more information about ABSTRACT help text.

Help menu option 3 will present help text describing the function keys available on the list display. You can also acquire help text about the function keys by positioning the cursor to the last line on the display and pressing the Help key.

Option 4 will begin the index search feature. You will be able to find topics associated with keywords that you specify. The index search facility is described in the Help for help menu of option 1 and also in the Introduction section of this user's guide.

Help option 5 will present a display that tells about the product.

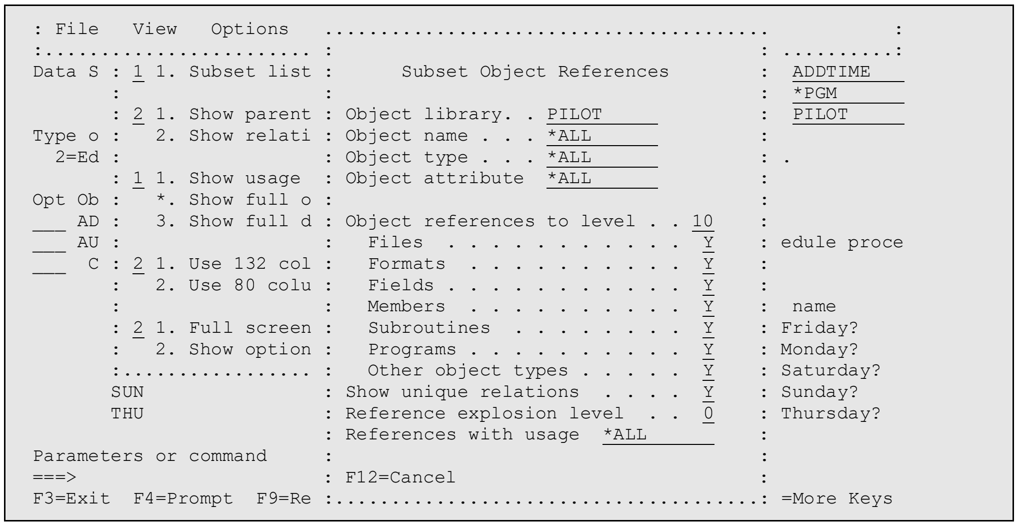


```
 : File   View   Options   Help                                         :
 :......................................................................:
 Data Set . . *FIRST     : 5 1. Help for help... : ion to name . . . ADDTIME
                         :   2. Extended help... : ion to type . . . *PGM
                         :   3. Keys help...     : ion to library. . PILOT
 Type option, then press :   4. Help index...    :
   1=Select  3=Copy  4=D :   5. A ..........................................
                         :....... :                                     :
 Opt Object/Uses          Type    :                  About                :
____ ADDTIME              *PGM RP :                                       :
____ AUTOSCH              *PGM RP :                 Abstract              :
____  HOLIDAYS            *FILE P :                                       :
____   HOLFMT             *FMT    :               Version 2.0             :
____  JOBHDR              *FILE P :           Serial Number 17241         :
____   JOBFMT             *FMT    :                                       :
____    DAY01             Char 1  :           Made Exclusively For        :
____    DAY02             Char 1  :        ACME Widget Company, Inc.      :
____    DAY03             Char 1  :                                       :
____    DAY04             Char 1  :         Copyright Help/Systems        :
                                  :                                       :
 Parameters or command            : F12=Cancel                            :
 ===>                             :.......................................:
 F3=Exit  F4=Prompt  F9=Retrieve  F10=Actions  F23=More Options  F24=More Keys
```

Return to the help menu from any of these displays by pressing F12. Exit the help menu and action bar by continuing to press F12 until you return to the list display.

Changing the list contents

The contents of the object list can be changed through the subset window. Press F17 or select the first item on the View menu of the action bar to view and/or change the subset criteria.



```
: File   View   Options   .......................................           :
:.......................  :                                       : .........:
Data S : 1 1. Subset list :      Subset Object References         :  ADDTIME
       :                  :                                       :  *PGM
       : 2 1. Show parent : Object library. . PILOT               :  PILOT
Type o :   2. Show relati : Object name . . . *ALL                :
  2=Ed :                  : Object type . . . *ALL                : .
       : 1 1. Show usage  : Object attribute  *ALL                :
Opt Ob :   *. Show full o :                                       :
___ AD :   3. Show full d : Object references to level . . 10     :
___ AU :                  :   Files  . . . . . . . . . . .  Y     : edule proce
___  C : 2 1. Use 132 col :   Formats  . . . . . . . . . .  Y     :
       :   2. Use 80 colu :   Fields . . . . . . . . . . .  Y     :
       :                  :   Members  . . . . . . . . . .  Y     :  name
       : 2 1. Full screen :   Subroutines  . . . . . . . .  Y     : Friday?
       :   2. Show option :   Programs . . . . . . . . . .  Y     : Monday?
       :................. :   Other object types . . . . .  Y     : Saturday?
       SUN                : Show unique relations  . . . .  Y     : Sunday?
       THU                : Reference explosion level  . .  0     : Thursday?
                          : References with usage  *ALL           :
Parameters or command     :                                       :
===>                      : F12=Cancel                            :
F3=Exit  F4=Prompt  F9=Re :.......................................: =More Keys
```

The subset window on the display shows the object name criteria used in selecting the parent objects on the display and also indicates which references will be included in the list.

Specify the objects to be included in the list by setting the appropriate values for library, name, type, and attribute at the top of the display. These values correspond to the LIB, OBJ, OBJTYP, and OBJATR command parameters. Refer to the parameter descriptions for complete information about each field.

The lower part of the window allows you to specify the relationships that you want to be displayed. The categories in the window correspond to DSPxxxx parameters on the command. Refer to the command descriptions in Part 4 for a discussion of these parameters. Select a ‘Y’ or an ‘N’ for each of the different categories. Specify a numeric value for the reference and explosion levels. The “References with usage” selection can specify either *ALL, *NONCMD, *CMD, *IO, *INP, *OUT, *UPD, or a single reference type (CALL, OVRDBF, Inp, etc.) that should be included in the list. Refer to the parameter descriptions for a complete description of each category.

Once you have made your selections, press the Enter key. The object list will be redisplayed according to the values in the subset window. Press function key 12 from the display above to return to the object list without modifying the subset criteria.

Object reference level

Use the object reference level value to specify the maximum depth of the relationship display. This value controls the number of indented levels that appear under the leftmost (parent) objects. It is the value that is specified for the DSPLVL parameter when the object relation command is started. Use a value between 0 and 10 for the reference level field. If you specify zero, the display will be similar to the one on page 3-14 showing only the parent objects in the list. A higher number will allow up to 10 levels of indenting to appear in the object list. The display on page 3-15 shows an indented list that is 3 levels deep.

Reference explosion level

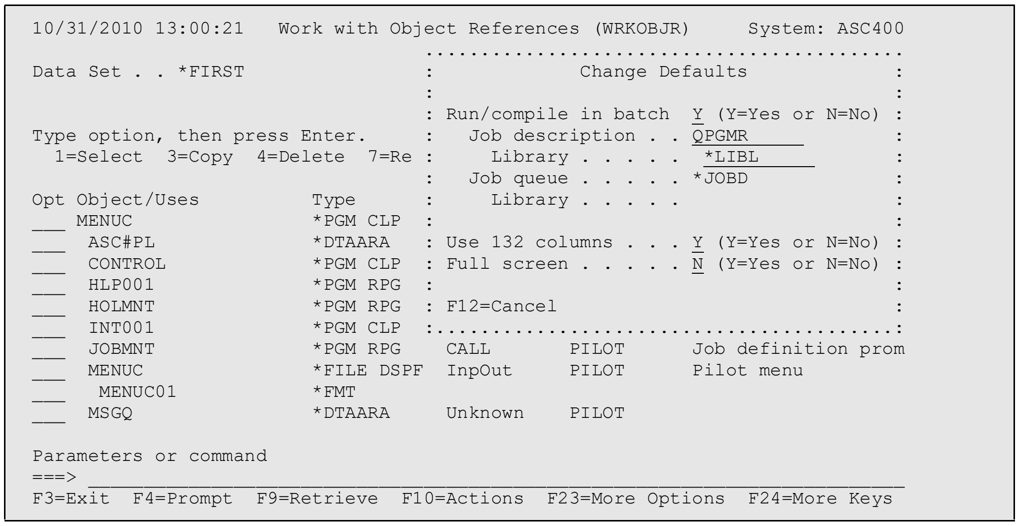
Use the second level number, the explosion level, to specify the maximum number of times that an additional object relation explosion should occur. This value controls the number of recursive iterations for the list explosion. If the explosion level is greater than zero, the object relations explosion will be attempted for each indented item in the list until no additional relations can be found, or the explosion level is reached.

The value in this field is specified on the EXPLVL parameter when the object relation command is started. Use a value between 0 and 10 for the explosion level. If you specify zero, all references for the parent object are given, but references for the underlying objects are not. A display like the first example on page 3-15 showing relations for only the parent objects in the list. No relations are given for any of the programs shown as referenced by MENUC for example.

A higher explosion number will allow up to 10 levels of recursion to occur, permitting a display like the one on page 3-2 showing three reference generations. This display (unlike the one on page 3-15) shows reference information for the indented programs such as CONTROL and HLP001 as if they had been parent objects.

Changing the defaults

Press function key 18 to make the Change Defaults window appear on your display. You can use it to specify the job description that should be used when ABSTRACT submits work to the batch subsystem, and configure display options to meet your preferences.



```
10/31/2010 13:00:21   Work with Object References (WRKOBJR)     System: ASC400
                                  ........................................
Data Set . . *FIRST               :            Change Defaults           :
                                  :                                      :
                                  : Run/compile in batch  Y (Y=Yes or N=No) :
Type option, then press Enter.    :   Job description . . QPGMR          :
  1=Select  3=Copy  4=Delete  7=Re :     Library . . . . .  *LIBL         :
                                  :   Job queue . . . . . *JOBD          :
Opt Object/Uses          Type      :     Library . . . . .                :
___ MENUC                *PGM CLP  :                                      :
___  ASC#PL              *DTAARA   : Use 132 columns . . . Y (Y=Yes or N=No) :
___  CONTROL             *PGM CLP  : Full screen . . . . . N (Y=Yes or N=No) :
___  HLP001              *PGM RPG  :                                      :
___  HOLMNT              *PGM RPG  : F12=Cancel                           :
___  INT001              *PGM CLP  :......................................:
___  JOBMNT              *PGM RPG    CALL       PILOT      Job definition prom
___  MENUC               *FILE DSPF  InpOut     PILOT      Pilot menu
___   MENUC01            *FMT
___  MSGQ                *DTAARA     Unknown    PILOT

Parameters or command
===> ___________________________________________________________________________
F3=Exit  F4=Prompt  F9=Retrieve  F10=Actions  F23=More Options  F24=More Keys
```

Change elements of the display by typing over them and pressing the Enter key. Use function key 12 to exit the window.

The first entry field determines whether your selections for the 14=Compile and 16=Run options will be submitted automatically to the batch subsystem. Specify ‘N’ to run these options interactively. If you do, they will be submitted to batch only if function key 14 is used.

The job description and job queue specified on the display will be used when submitting batch jobs, whether they are submitted automatically (see above) or via function key 14. Indicate the name and library of the job description you want to use or type *USRPRF to use the job description named in your user profile. Likewise, specify the job queue or use *JOBD to reference the queue named by the indicated job description.

The ‘132 column’ option will be enabled if you are using a 132 column capable display. Specify a ‘Y’ value to indicate that the full 27x132 capability of your workstation should be used. Use ‘N’ to request 24x80 displays. This option can also be controlled from the View menu of the action bar.

The ‘Full screen’ option controls whether or not the option and function key summaries will be displayed. Specify a ‘Y’ to suppress them and use as much of the display as possible for the object list. Use a ‘N’ to indicate that you do not want to use ABSTRACT in full screen mode.

Printing the list contents

The contents of the object list can be printed by pressing function key 21. The OUTPUT(*PRINT) option will be used with the current Work With command to print the entire list defined by the current subset criteria. Refer to the example on the next page.

```
 10/31/2010 11:13:31                           Abstract
                                           Object References
Object name . . : MENUC                                                                System . . : ASC400
Object type . . : *ALL                                                                 Data Set . : *FIRST
Object library  : PILOT
                                           Library or
   Object/Uses         Type         Usage      Qualifier  Text                          Extended Description

 1 MENUC              *PGM CLP               PILOT      Pilot menu driver
 2 .ASC#PL            *DTAARA      RTVDTAARA  PILOT                                     Lib(*LIBL) Seq#(39)
 2 .CONTROL           *PGM CLP     CALL       PILOT      Monitor control information   Lib(*LIBL) Seq#(72)
 3 ..CONTROL          *FILE DSPF   InpOut     PILOT      Control information display
 4 ...CONTRL01        *FMT
 3 ..EXTEND           *DTAARA      RTVDTAARA  PILOT                                     Lib(*LIBL) Seq#(56)
 3 ..HLP001           *PGM RPG     CALL       PILOT      Help text processor           Lib(*LIBL) Seq#(74)
 4 ...HELPTXT         *FILE PF     Inp        PILOT                                     Lib(*LIBL)
 5 ....TEXT           *FMT
 6 .....LINTX         Char 78      Inp                                                  Buffer(14 91)
 4 ...HLP001          *FILE DSPF   InpOut     PILOT                                     Lib(*LIBL)
 5 ....HELP           *FMT
 5 ....HELPC          *FMT
 3 ..PILOT            *PGM CLP     CALL       PILOT      Pilot command processor       Seq#(96)
 4 ...JOBCHK          *PGM RPG     CALL       PILOT      Scheduling - Check for jobs   Lib(*LIBL) Seq#(84)
 5 ....JOBHDR         *FILE PF     Upd        PILOT      Job Master Header             Lib(*LIBL)
 6 .....JOBFMT        *FMT
 7 ......EXEDAY       Pkd 2,0      Inp                   LAST DAY RUN                  Buffer(148 149)
 7 ......EXEMON       Pkd 2,0      Inp                   LAST MONTH RUN                Buffer(146 147)
 7 ......EXETIM       Pkd 6,0      Out                   TIME LAST RUN                 Buffer(152 155)
 7 ......EXEYER       Pkd 2,0      Inp                   LAST YEAR RUN                 Buffer(150 151)
 7 ......JOBNAM       Char 10      Inp                   JOB NAME                      Buffer(1 10)
 7 ......PCODE        Char 10      Out                   PERIOD CODE                   Buffer(171 180)
 7 ......SBMDAY       Pkd 2,0      Inp                   LAST DAY SUBMITTED            Buffer(138 139)
 7 ......SBMMON       Pkd 2,0      Inp                   LAST MONTH SUBMITTED          Buffer(136 137)
 7 ......SBMTIM       Pkd 6,0      Out                   TIME LAST SUBMITTED           Buffer(142 145)
 5 ....JOBSCHL4       *FILE LF     Upd        PILOT                                     Lib(*LIBL)
 6 .....SCHFMT        *FMT
 7 ......EXEDAY       Pkd 2,0      Inp                   EXECUTE DAY                   Buffer(23 24)
 7 ......EXEMON       Pkd 2,0      Inp                   EXECUTE MONTH                 Buffer(21 22)
 7 ......EXETIM       Pkd 6,0      Out                   EXECUTE TIME                  Buffer(27 30)
 7 ......EXEYER       Pkd 2,0      Inp                   EXECUTE YEAR                  Buffer(25 26)
 7 ......JOBNAM       Char 10      Inp                   JOB NAME                      Buffer(1 10)
 7 ......JOBSQN       Char 6       Out                                                  Buffer(57 62)
 7 ......SCHCDE       Char 1       Out                   SCHEDULE CODE                 Buffer(11 11)
 7 ......SCHDAY       Pkd 2,0      Inp                   SCHEDULE DAY                  Buffer(14 15)
 7 ......SCHMON       Pkd 2,0      Inp                   SCHEDULE MONTH                Buffer(12 13)
 7 ......SCHYER       Pkd 2,0      Inp                   SCHEDULE YEAR                 Buffer(16 17)
 4 ...LASTTIME        *DTAARA      Unknown    PILOT                                     Lib(*LIBL)
 4 ...LASTWAKE        *DTAARA      RTVDTAARA  PILOT                                     Lib(*LIBL) Seq#(33)
 4 ...LSTLOA          *DTAARA      RTVDTAARA  PILOT                                     Lib(*LIBL) Seq#(58)
 4 ...PILOT           *LIB PROD    ADDLIBLE   QSYS                                      Seq#(20)
 4 ...SCH001          *PGM RPG     CALL       PILOT      Schedule loader               Lib(*LIBL) Seq#(51)
 5 ....EXTEND         *DTAARA      Unknown    PILOT                                     Lib(*LIBL)
 5 ....HOLIDAYS       *FILE PF     Inp        PILOT                                     Lib(*LIBL)
```

Object References

Four commands present object reference information on the list display:

WRKOBJR	Work With Object References
WRKOBJRS	Work With Sequenced Object References
WRKOBJRX	Work With Cross Referenced Object References
WRKFGR	Work With File Group References

Output from the commands can also be routed to the printer or an output file. Together, these commands tell what objects are referenced by the parent objects in the list. The displays and reports from these commands provide the following types of information:

Program references to files: What files are used by a given program? How are they used? Which formats of the files are accessed? Of all the fields within these files, which ones are referenced by the program? How are these fields used?

Program-program transfers: What programs or user written commands are invoked from a specified program? How does the invocation stack continue to develop as programs are invoked?

File-file relationships: What physical file(s) are referenced by a given logical?

Program references to other objects: What other objects are used by the program? What commands cause them to be referenced?

Copybook and Subroutine References: What subroutines and copybooks are referenced in the application's source file?

This section of the chapter will explain how the object list display and report can be used to acquire and interpret this information.

File References

There are several types of file references that may occur within your application. ABSTRACT will record them and make the information available to you through the list displays. Use the information and examples below to tailor the display to suit your specific requirements.

Program references to files

Program references to files result from external data structure definitions as well as usage for I/O purposes.

Obtain information about the files referenced by programs in your applications as follows:

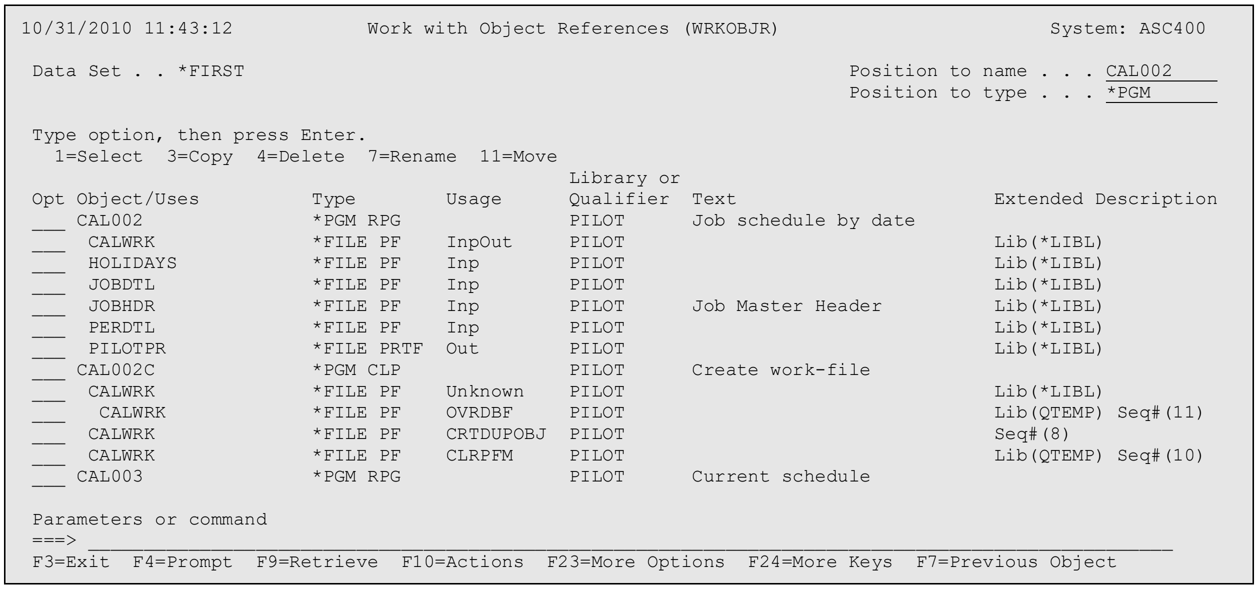
Specify an object naming request that includes the program(s) of interest, and make certain that the object type value includes program objects. Do this by using the LIB, OBJ, and OBJTYPE parameters of the reference command, or the subset window available through the action bar or function key 17.

The relationship display must show at least one level of indentation. Use a value of 1 or higher for the DSPLVL parameter or the object reference level value of the subset window. Alternatively, you can specify a Y value for the Show relations entry of the view menu of the action bar.

Files must be included in the relationship display. Specify DSPFILE(*YES) when using the command, or a Y value for the Files entry of the subset window.

The display below presents an example of file references by programs in the PILOT library. Additional references have been removed from the display. The command used to access the display had the form:

```
WRKOBJR LIB(Library) OBJ(*ALL) OBJTYPE(*PGM) DSPLVL(10)
     DSPFILE(*YES) DSPPGM(*NO) DSPFMT(*NO) DSPFLD(*NO)
     DSPMBR(*NO) DSPSUB(*NO) DSPOTR(*NO) DSPUSG(*ALL)
```



```
10/31/2010 11:43:12          Work with Object References (WRKOBJR)                System: ASC400

 Data Set . . *FIRST                                                Position to name . . . CAL002
                                                                    Position to type . . . *PGM

 Type option, then press Enter.
   1=Select   3=Copy   4=Delete   7=Rename   11=Move
                                             Library or
 Opt Object/Uses          Type        Usage      Qualifier  Text                       Extended Description
 ___ CAL002               *PGM RPG               PILOT      Job schedule by date
 ___  CALWRK              *FILE PF    InpOut     PILOT                                 Lib(*LIBL)
 ___  HOLIDAYS            *FILE PF    Inp        PILOT                                 Lib(*LIBL)
 ___  JOBDTL              *FILE PF    Inp        PILOT                                 Lib(*LIBL)
 ___  JOBHDR              *FILE PF    Inp        PILOT      Job Master Header          Lib(*LIBL)
 ___  PERDTL              *FILE PF    Inp        PILOT                                 Lib(*LIBL)
 ___  PILOTPR             *FILE PRTF  Out        PILOT                                 Lib(*LIBL)
 ___ CAL002C              *PGM CLP               PILOT      Create work-file
 ___  CALWRK              *FILE PF    Unknown    PILOT                                 Lib(*LIBL)
 ___   CALWRK             *FILE PF    OVRDBF     PILOT                                 Lib(QTEMP) Seq#(11)
 ___  CALWRK              *FILE PF    CRTDUPOBJ  PILOT                                 Seq#(8)
 ___  CALWRK              *FILE PF    CLRPFM     PILOT                                 Lib(QTEMP) Seq#(10)
 ___ CAL003               *PGM RPG               PILOT      Current schedule

 Parameters or command
 ===> ______________________________________________________________________________________________
 F3=Exit  F4=Prompt  F9=Retrieve  F10=Actions  F23=More Options  F24=More Keys  F7=Previous Object
```

The display shows three parent programs that currently exist in the PILOT library: CAL002, CAL002C and CAL003. File references for CAL003 (if any) are not shown because there is not room on the display. Changing the display to full screen would provide an additional six lines of references.

The CAL002 RPG program references five database files and one printer file. Only the CALWRK data can be changed by the program. The extended description for each of the files indicates that the job's library list at run time will determine the actual file used. ABSTRACT found files matching the indicated names in the PILOT library (cf the Library column) and used them to determine file attributes and text.

The CAL002C CL program makes three references, all to a file named CALWRK. The first CALWRK reference is unqualified with usage recorded as "Unknown". Because the display level allows more than one level of indentation, an additional reference is shown. Together, these two references indicate an Override With Database File at sequence 11 will redirect subsequent program usage of CALWRK (the first reference) to QTEMP/CALWRK (the second reference).

File references due to command usage

Sequence number 8 in the CAL002C program issues a Create Duplicate Object (CRTDUPOBJ) command against the PILOT/CALWRK file. Lack of a LIB( ) reference in the extended description for it indicates that the reference was qualified with the library indicated in the Library column. Sequence 10 in the program issues a CLRPFM command against QTEMP/CALWRK. ABSTRACT was unable to locate a CALWRK file in QTEMP, so the attributes (and text) shown on the display were obtained from the file in the PILOT library (as noted by the Library column.

These references occur because of the command tracking capabilities of ABSTRACT. References to commands with *FILE parameters that are being recorded (because of standard or user defined command tracking definitions) will appear as long as the object usage value does not exclude them. Command tracking definitions are described in *Part 2, Initialization*. The Work With Command Tracking Definitions (WRKCMTRKD) command will allow you to create the definitions that track file references.

You can suppress references to files made by command usage within a program (c.f. OVRDBF, CRTDUPOBJ, CLRPFM) by using the value *NONCMD for the DSPUSG parameter or the References with usage entry of the subset window. The resulting object display will show only the types of data usage associated with the files: Input, Output, Unknown, etc.

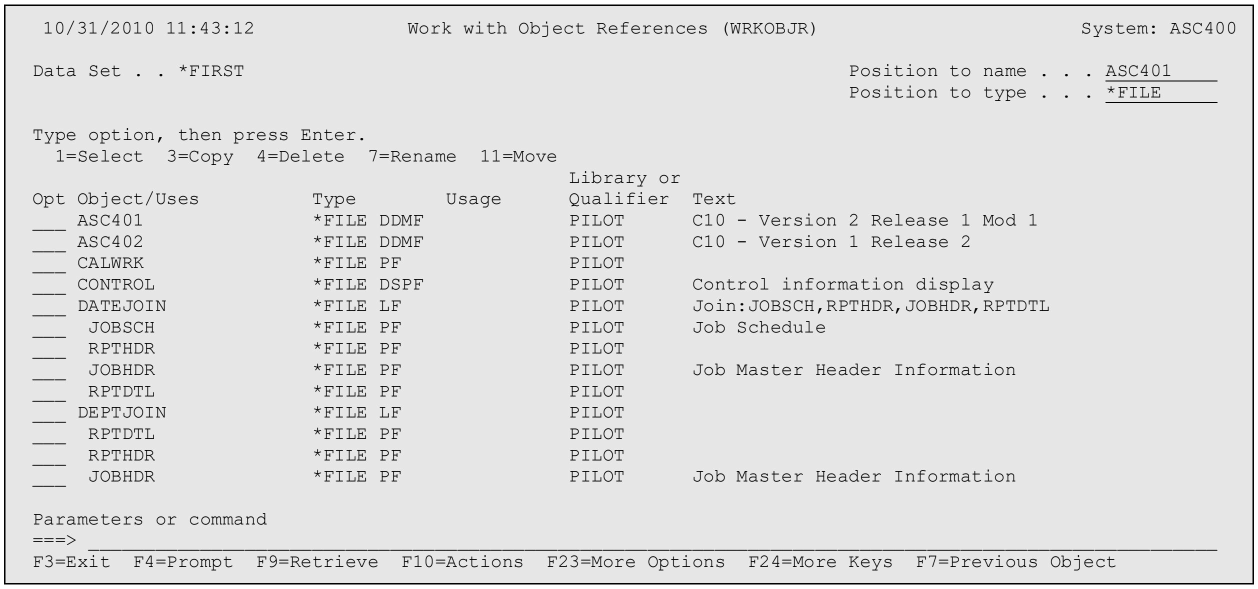
You can see the physical files that are referenced by logical files on the display by setting the explosion level (and perhaps the indentation level) to a higher value.

Logical file references to physical files

Logical files reference at least one physical file each. Join logical files and multiple format logical files may reference up to 32 physical files. These relationships can be viewed by requesting an object list that includes logical files as parent objects, at least one level of indentation, and displays file references. Refer to the section above for details about creating requests that specify these factors.

The next display presents an example of logical-physical file references within the PILOT library. Additional references have been removed from the display. The command used to access the display had the form:

```
WRKOBJR LIB(Library) OBJ(*ALL) OBJTYPE(*FILE) DSPLVL(10)
     DSPFILE(*YES) DSPPGM(*NO) DSPFMT(*NO) DSPFLD(*NO)
     DSPMBR(*NO) DSPSUB(*NO) DSPOTR(*NO) DSPUSG(*ALL)
```



```
 10/31/2010 11:43:12           Work with Object References (WRKOBJR)                System: ASC400

Data Set . . *FIRST                                                Position to name . . . ASC401____
                                                                   Position to type . . . *FILE_____

Type option, then press Enter.
  1=Select  3=Copy  4=Delete  7=Rename  11=Move
                                                Library or
Opt Object/Uses          Type        Usage      Qualifier  Text
___ ASC401               *FILE DDMF             PILOT      C10 - Version 2 Release 1 Mod 1
___ ASC402               *FILE DDMF             PILOT      C10 - Version 1 Release 2
___ CALWRK               *FILE PF               PILOT
___ CONTROL              *FILE DSPF             PILOT      Control information display
___ DATEJOIN             *FILE LF               PILOT      Join:JOBSCH,RPTHDR,JOBHDR,RPTDTL
___  JOBSCH              *FILE PF               PILOT      Job Schedule
___  RPTHDR              *FILE PF               PILOT
___  JOBHDR              *FILE PF               PILOT      Job Master Header Information
___  RPTDTL              *FILE PF               PILOT
___ DEPTJOIN             *FILE LF               PILOT
___  RPTDTL              *FILE PF               PILOT
___  RPTHDR              *FILE PF               PILOT
___  JOBHDR              *FILE PF               PILOT      Job Master Header Information

Parameters or command
===> _________________________________________________________________________________________
F3=Exit  F4=Prompt  F9=Retrieve  F10=Actions  F23=More Options  F24=More Keys  F7=Previous Object
```

All the files in the PILOT library are presented as parent objects on the display. References to underlying physical file(s) are included for each logical file on the display. The library name shown in the Library column is the actual library for the physical file - no search is performed.

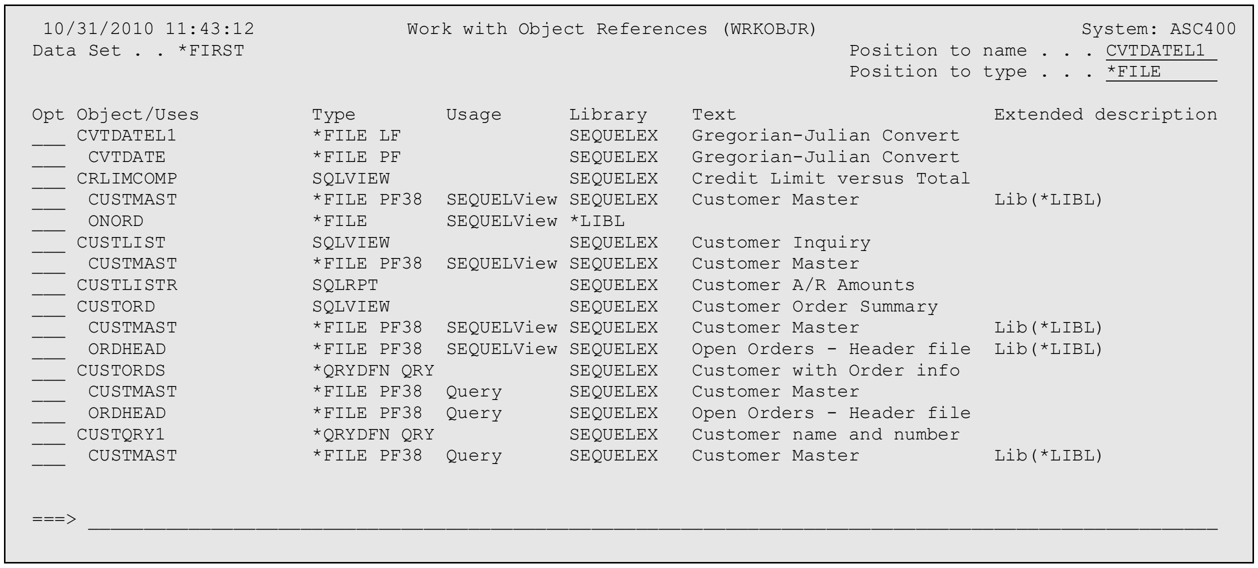
For additional information about the file analysis and reporting capabilities of ABSTRACT, refer to part 5 in this user's guide.

Other references to files

Other types of objects may refer to files within your applications. **Query/400** query definitions and **SEQUEL**[1] **views** both reference file data and are cross referenced by ABSTRACT. **Menu objects** reference the display file associated with the menu. Reference information for these other types of objects can be viewed in the list by requesting an object list that includes them as parent objects, at least one level of indentation, and displays file references. Refer to the sections above for details about creating requests that specify these factors.

The following full screen display presents an example of file references from both Query/400 query definitions and SEQUEL views within the SEQUELEX library. The command used to access the display had the form:

```
WRKOBJR LIB(Library) OBJ(*ALL) OBJTYPE(*ALL) DSPLVL(10)
        DSPFILE(*YES) DSPPGM(*NO) DSPFMT(*NO) DSPFLD(*NO)
        DSPMBR(*NO) DSPSUB(*NO) DSPOTR(*NO) DSPUSG(*ALL)
```



```
 10/31/2010 11:43:12            Work with Object References (WRKOBJR)                 System: ASC400
Data Set . . *FIRST                                                Position to name . . . CVTDATEL1
                                                                   Position to type . . . *FILE

Opt Object/Uses           Type        Usage      Library    Text                         Extended description
___ CVTDATEL1             *FILE LF               SEQUELEX   Gregorian-Julian Convert
___  CVTDATE              *FILE PF               SEQUELEX   Gregorian-Julian Convert
___ CRLIMCOMP             SQLVIEW                SEQUELEX   Credit Limit versus Total
___  CUSTMAST             *FILE PF38  SEQUELView SEQUELEX   Customer Master              Lib(*LIBL)
___  ONORD                *FILE       SEQUELView *LIBL
___ CUSTLIST              SQLVIEW                SEQUELEX   Customer Inquiry
___  CUSTMAST             *FILE PF38  SEQUELView SEQUELEX   Customer Master
___ CUSTLISTR             SQLRPT                 SEQUELEX   Customer A/R Amounts
___ CUSTORD               SQLVIEW                SEQUELEX   Customer Order Summary
___  CUSTMAST             *FILE PF38  SEQUELView SEQUELEX   Customer Master              Lib(*LIBL)
___  ORDHEAD              *FILE PF38  SEQUELView SEQUELEX   Open Orders - Header file    Lib(*LIBL)
___ CUSTORDS              *QRYDFN QRY            SEQUELEX   Customer with Order info
___  CUSTMAST             *FILE PF38  Query      SEQUELEX   Customer Master
___  ORDHEAD              *FILE PF38  Query      SEQUELEX   Open Orders - Header file
___ CUSTQRY1              *QRYDFN QRY            SEQUELEX   Customer name and number
___  CUSTMAST             *FILE PF38  Query      SEQUELEX   Customer Master              Lib(*LIBL)


===> ____________________________________________________________________________________________________
```

The display shows the SEQUEL views and query definitions in the SEQUELEX library and the files they reference. Unqualified references, which will be resolved when the query or view is run, are noted with a Lib(*LIBL) reference in the Extended Description column on the display. Other references, such as those for the CUSTORDS query and the CUSTLIST view, are fully qualified by the parent object.

As with the other object list displays, objects that cannot be found during the ABSTRACT search will show no attribute or text information. The file named ONORD, having an unqualified reference by the CRLIMCOMP view, could not be found so no attribute, text, or library information is provided on the display.

Field References

Field level reference information can be acquired for the programs in your applications as indicated below. It is important to note that ABSTRACT shows only the fields actually referenced by the pro-

1. SEQUEL is a licensed program product available from SEQUEL Software.

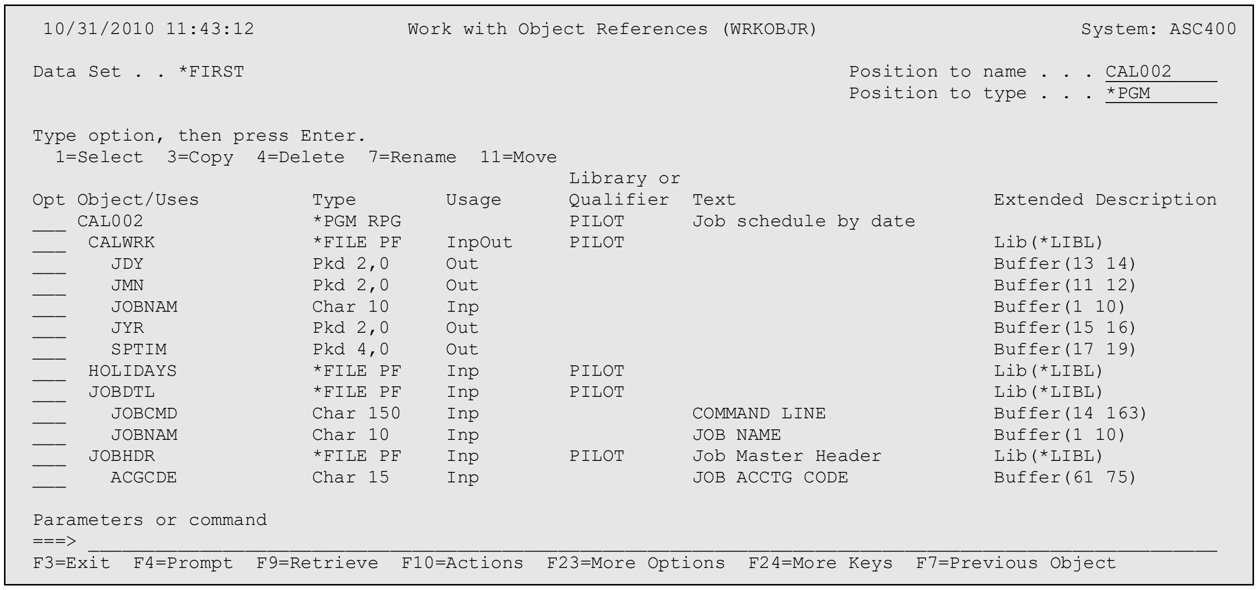
gram, not all of the fields in the file that is used. Note also, that when a program does a WRITE to a file, all fields are implicitly referenced and will be listed by ABSTRACT.

Specify an object naming request that includes the program(s) of interest, and make certain that the object type value includes program objects. Do this by using the LIB, OBJ, and OBJTYPE parameters of the reference command, or the subset window available through the action bar or function key 17.

The relationship display must show at least three levels of indentation. Use a value of 3 (or higher) for the DSPLVL parameter or the object reference level value of the subset window. Alternatively, you can specify a Y value for the Show relations entry of the view menu of the action bar.

Fields must be included in the relationship display. Files should also be included. Specify DSPFLD(*YES) and DSPFILE(*YES) when using the command, or a Y value for the Fields and Files entries of the subset window.

The display below presents an example of field references by programs in the PILOT library. Except for the file references, additional references have been removed from the display.



```
 10/31/2010 11:43:12           Work with Object References (WRKOBJR)                  System: ASC400

Data Set . . *FIRST                                                   Position to name . . . CAL002
                                                                      Position to type . . . *PGM

Type option, then press Enter.
  1=Select  3=Copy  4=Delete  7=Rename  11=Move
                                                  Library or
Opt Object/Uses          Type        Usage        Qualifier  Text                           Extended Description
___ CAL002               *PGM RPG                 PILOT      Job schedule by date
___   CALWRK             *FILE PF    InpOut       PILOT                                     Lib(*LIBL)
___     JDY              Pkd 2,0     Out                                                    Buffer(13 14)
___     JMN              Pkd 2,0     Out                                                    Buffer(11 12)
___     JOBNAM           Char 10     Inp                                                    Buffer(1 10)
___     JYR              Pkd 2,0     Out                                                    Buffer(15 16)
___     SPTIM            Pkd 4,0     Out                                                    Buffer(17 19)
___   HOLIDAYS           *FILE PF    Inp          PILOT                                     Lib(*LIBL)
___   JOBDTL             *FILE PF    Inp          PILOT                                     Lib(*LIBL)
___     JOBCMD           Char 150    Inp                     COMMAND LINE                   Buffer(14 163)
___     JOBNAM           Char 10     Inp                     JOB NAME                       Buffer(1 10)
___   JOBHDR             *FILE PF    Inp          PILOT      Job Master Header              Lib(*LIBL)
___     ACGCDE           Char 15     Inp                     JOB ACCTG CODE                 Buffer(61 75)

Parameters or command
===> ____________________________________________________________________________________________
F3=Exit  F4=Prompt  F9=Retrieve  F10=Actions  F23=More Options  F24=More Keys  F7=Previous Object
```

Acquire a field reference list like this by using a command like:

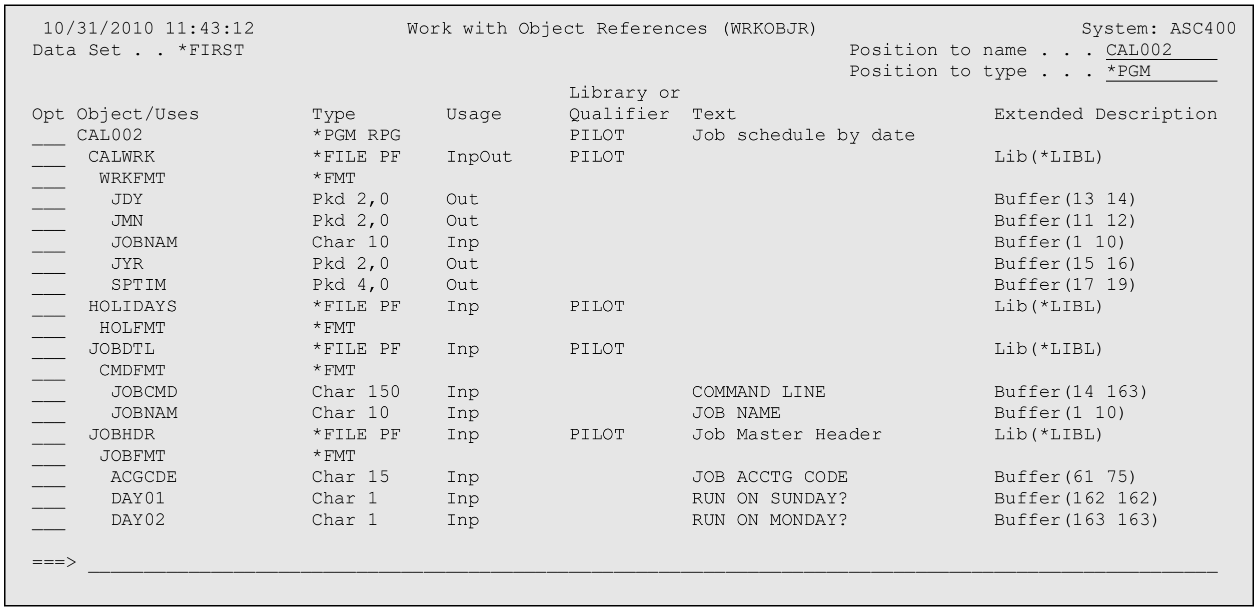
```
WRKOBJR LIB(Library) OBJ(*ALL) OBJTYPE(*PGM) DSPLVL(10)
      DSPFILE(*YES) DSPFLD(*YES) DSPPGM(*NO) DSPFMT(*NO)
      DSPMBR(*NO) DSPSUB(*NO)  DSPOTR(*NO) DSPUSG(*ALL)
```

The display shows several field references made by the CAL002 program in the PILOT library. References are listed beneath the file in which they occur. Field attributes, text, and buffer positions are obtained from the file in the library indicated by the Library column. Field usage is displayed as either Inp (input) or Out (update or output) depending on the information recorded during the initialization step.

The information shown on the display records actual field usage by the program. For example, no fields in the HOLIDAYS file are actually referenced by the program, and only two fields (JOBNAM, JOBCMD) from the JOBDTL file are used. Notice that some fields can be updated by the program without all fields in the file being changed. Refer to the usage information for the CALWRK file.

Although file information could be suppressed from the display (by specifying DSPFILE(*NO) for example), doing so would make it impossible to know from which file the fields were being accessed.

Format references can be added to the display by specifying DSPFMT(*YES) on the reference command, or by using a Y as the Formats value on the subset display. This can sometimes make it easier to tell exactly which fields are being used, especially in a multi-format logical file. The previous display looks like the one below when formats are added to the list. Note that the full screen form is used to display the most information possible.


```
 10/31/2010 11:43:12            Work with Object References (WRKOBJR)                    System: ASC400
Data Set . . *FIRST                                                  Position to name . . . CAL002____
                                                                     Position to type . . . *PGM______
                                                 Library or
Opt Object/Uses          Type        Usage      Qualifier  Text                      Extended Description
___ CAL002               *PGM RPG               PILOT      Job schedule by date
___  CALWRK              *FILE PF    InpOut     PILOT                                Lib(*LIBL)
___   WRKFMT             *FMT
___    JDY               Pkd 2,0     Out                                             Buffer(13 14)
___    JMN               Pkd 2,0     Out                                             Buffer(11 12)
___    JOBNAM            Char 10     Inp                                             Buffer(1 10)
___    JYR               Pkd 2,0     Out                                             Buffer(15 16)
___    SPTIM             Pkd 4,0     Out                                             Buffer(17 19)
___  HOLIDAYS            *FILE PF    Inp        PILOT                                Lib(*LIBL)
___   HOLFMT             *FMT
___  JOBDTL              *FILE PF    Inp        PILOT                                Lib(*LIBL)
___   CMDFMT             *FMT
___    JOBCMD            Char 150    Inp                   COMMAND LINE              Buffer(14 163)
___    JOBNAM            Char 10     Inp                   JOB NAME                  Buffer(1 10)
___  JOBHDR              *FILE PF    Inp        PILOT      Job Master Header         Lib(*LIBL)
___   JOBFMT             *FMT
___    ACGCDE            Char 15     Inp                   JOB ACCTG CODE            Buffer(61 75)
___    DAY01             Char 1      Inp                   RUN ON SUNDAY?            Buffer(162 162)
___    DAY02             Char 1      Inp                   RUN ON MONDAY?            Buffer(163 163)

===> _____________________________________________________________________________________________
```

Program References

ABSTRACT initialization will record transfers from one program to another (CALL, TFRCTL) and program invocation through user written commands (CPP/VCP) and both display file and program menu objects. All of this information is available through the object referencing and usage lists.

The WRKOBJR and WRKOBJRX commands can display a multiple level explosion in addition to a simple program-program or command-program list. The WRKOBJRS is restricted to a single level listing; the called programs in the list will not be further exploded.

Specify an object naming request that includes the program(s), menu(s), and/or command(s) of interest, and make certain that the object type value includes the program, menu, and/or command objects you want to see. Do this by using the LIB, OBJ, and OBJTYPE parameters of the reference command, or the subset window available through the action bar or function key 17.

The relationship display must show at least one level of indentation. Use a value of 1 or higher for the DSPLVL parameter or the object reference level value of the subset window. Alternatively, you can specify a Y value for the Show relations entry of the view menu of the action bar.

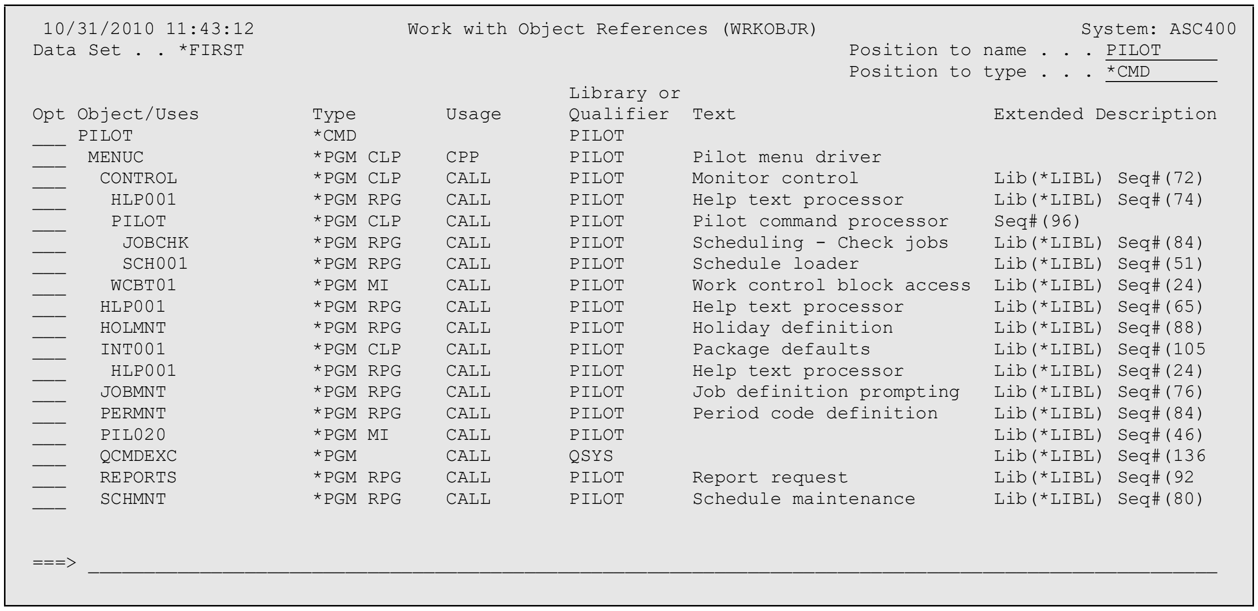
If you want WRKOBJR to show a multiple level explosion, you should set the explosion level value to the number of invocation levels you want to see (up to 10). Specify this value using the EXPLVL parameter or the Reference explosion level entry of the subset window. The DSPLVL value should be set to a value at least as high as the explosion level.

Programs must be included in the relationship display. Specify DSPPGM(*YES) when using the command, or a Y value for the Programs entry of the subset window.

If your application includes user written commands and you want to see references to them and subsequent invocation of their command processing and validity checking programs, include the commands in the list by using DSPOTR(*YES) or a Y value for the Other object types entry of the subset window. Use the same value if you want to see references through iSeries menus.

The following display presents an example of program transfers starting with the PILOT command in the PILOT library. Additional references have been removed from the display. The command used to acquire the display had the form:

WRKOBJR LIB(*Library*) OBJ(Command-name) OBJTYPE(*CMD)
DSPLVL(10) EXPLVL(10) DSPPGM(*YES)
DSPFILE(*NO) DSPFLD(*NO) DSPFMT(*NO)
DSPMBR(*NO) DSPSUB(*NO) DSPOTR(*NO) DSPUSG(*ALL)



```
 10/31/2010 11:43:12            Work with Object References (WRKOBJR)                  System: ASC400
Data Set . . *FIRST                                                   Position to name . . . PILOT
                                                                      Position to type . . . *CMD
                                                  Library or
Opt Object/Uses           Type        Usage       Qualifier  Text                         Extended Description
___ PILOT                 *CMD                    PILOT
___  MENUC                *PGM CLP    CPP         PILOT      Pilot menu driver
___   CONTROL             *PGM CLP    CALL        PILOT      Monitor control              Lib(*LIBL) Seq#(72)
___    HLP001             *PGM RPG    CALL        PILOT      Help text processor          Lib(*LIBL) Seq#(74)
___    PILOT              *PGM CLP    CALL        PILOT      Pilot command processor      Seq#(96)
___     JOBCHK            *PGM RPG    CALL        PILOT      Scheduling - Check jobs      Lib(*LIBL) Seq#(84)
___     SCH001            *PGM RPG    CALL        PILOT      Schedule loader              Lib(*LIBL) Seq#(51)
___    WCBT01             *PGM MI     CALL        PILOT      Work control block access    Lib(*LIBL) Seq#(24)
___   HLP001              *PGM RPG    CALL        PILOT      Help text processor          Lib(*LIBL) Seq#(65)
___   HOLMNT              *PGM RPG    CALL        PILOT      Holiday definition           Lib(*LIBL) Seq#(88)
___   INT001              *PGM CLP    CALL        PILOT      Package defaults             Lib(*LIBL) Seq#(105
___    HLP001             *PGM RPG    CALL        PILOT      Help text processor          Lib(*LIBL) Seq#(24)
___   JOBMNT              *PGM RPG    CALL        PILOT      Job definition prompting     Lib(*LIBL) Seq#(76)
___   PERMNT              *PGM RPG    CALL        PILOT      Period code definition       Lib(*LIBL) Seq#(84)
___   PIL020              *PGM MI     CALL        PILOT                                   Lib(*LIBL) Seq#(46)
___   QCMDEXC             *PGM        CALL        QSYS                                    Lib(*LIBL) Seq#(136
___   REPORTS             *PGM RPG    CALL        PILOT      Report request               Lib(*LIBL) Seq#(92
___   SCHMNT              *PGM RPG    CALL        PILOT      Schedule maintenance         Lib(*LIBL) Seq#(80)


===> ____________________________________________________________________________________________________
```

The display shows the object list in the full screen mode, presenting the greatest number of items possible on one panel.

The display shows that the PILOT/PILOT command references a program named MENUC in the PILOT library. ABSTRACT located the program and determined that it is a CL program with the text shown on the display.

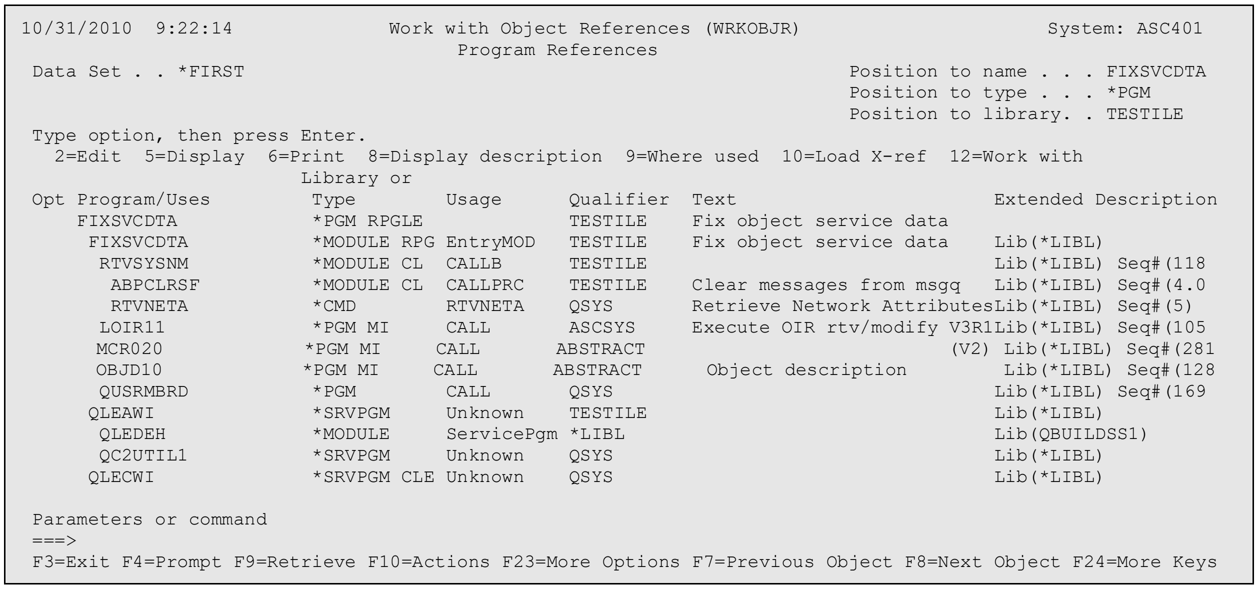
The MENUC program makes unqualified (*LIBL) CALLs to these programs: CONTROL, HLP001, HOLMNT, INT001, JOBMNT, PERMNT, PIL020, QCMDEXC, REPORTS, and SCHMNT. The sequence number where the CALL command occurs in MENUC is listed in the extended description for the program. ABSTRACT located each of the programs (except QCMDEXC) in the PILOT library and displays the corresponding program type and text for them.

The display also shows that the CONTROL program makes unqualified CALLs to programs HLP001 and WCBT01, and a qualified CALL to the PILOT/PILOT program at sequence 96.

The PILOT/PILOT program CALLs programs JOBCHK and SCH001 at statements 84 and 51 respectively.

The following display presents an example of the ILE FIXSVCDTA program in the TESTILE library. Additional references have been removed from the display. The command used to acquire the display had the form:

```
WRKOBJR LIB(Library) OBJ(Command-name) OBJTYPE(*PGM)
     DSPLVL(10) EXPLVL(10) DSPPGM(*YES)
     DSPFILE(*NO) DSPFLD(*NO) DSPFMT(*NO)
     DSPMBR(*NO) DSPSUB(*NO) DSPOTR(*YES) DSPUSG(*ALL)
```



```
10/31/2010  9:22:14             Work with Object References (WRKOBJR)                      System: ASC401
                                      Program References
 Data Set . . *FIRST                                                        Position to name . . . FIXSVCDTA
                                                                            Position to type . . . *PGM
                                                                            Position to library. . TESTILE
 Type option, then press Enter.
   2=Edit  5=Display  6=Print  8=Display description  9=Where used  10=Load X-ref  12=Work with
                         Library or
 Opt Program/Uses         Type        Usage      Qualifier  Text                       Extended Description
     FIXSVCDTA            *PGM RPGLE             TESTILE    Fix object service data
      FIXSVCDTA           *MODULE RPG EntryMOD   TESTILE    Fix object service data    Lib(*LIBL)
       RTVSYSNM           *MODULE CL  CALLB      TESTILE                               Lib(*LIBL) Seq#(118
        ABPCLRSF          *MODULE CL  CALLPRC    TESTILE    Clear messages from msgq   Lib(*LIBL) Seq#(4.0
        RTVNETA           *CMD        RTVNETA    QSYS       Retrieve Network AttributesLib(*LIBL) Seq#(5)
       LOIR11             *PGM MI     CALL       ASCSYS     Execute OIR rtv/modify V3R1Lib(*LIBL) Seq#(105
       MCR020            *PGM MI     CALL       ABSTRACT                           (V2) Lib(*LIBL) Seq#(281
       OBJD10            *PGM MI     CALL       ABSTRACT     Object description         Lib(*LIBL) Seq#(128
       QUSRMBRD           *PGM        CALL       QSYS                                  Lib(*LIBL) Seq#(169
      QLEAWI              *SRVPGM     Unknown    TESTILE                               Lib(*LIBL)
       QLEDEH             *MODULE     ServicePgm *LIBL                                 Lib(QBUILDSS1)
       QC2UTIL1           *SRVPGM     Unknown    QSYS                                  Lib(*LIBL)
      QLECWI              *SRVPGM CLE Unknown    QSYS                                  Lib(*LIBL)

 Parameters or command
 ===>
 F3=Exit F4=Prompt F9=Retrieve F10=Actions F23=More Options F7=Previous Object F8=Next Object F24=More Keys
```

The display shows that the FIXSVCDTA program contains a program entry module named the same as the program, FIXSVCDTA. The FIXSVCDTA module does a bound call (CALLB in RPGLE) to module RTVSYSNM. RTVSYSNM, in turn, does a bound call (CALLPRC in CLLE) to module ABPCLRSF. Notice that the FIXSVCDTA program also contains references to service program QLEAWI. Within that service program are references to module QLEDEH and service program QC2UTIL1. Within the FIXSVCDTA module, there is also an external call to program LOIR11. Note that ILE programs can do both bound and external calls.

Copybook and Source Member References

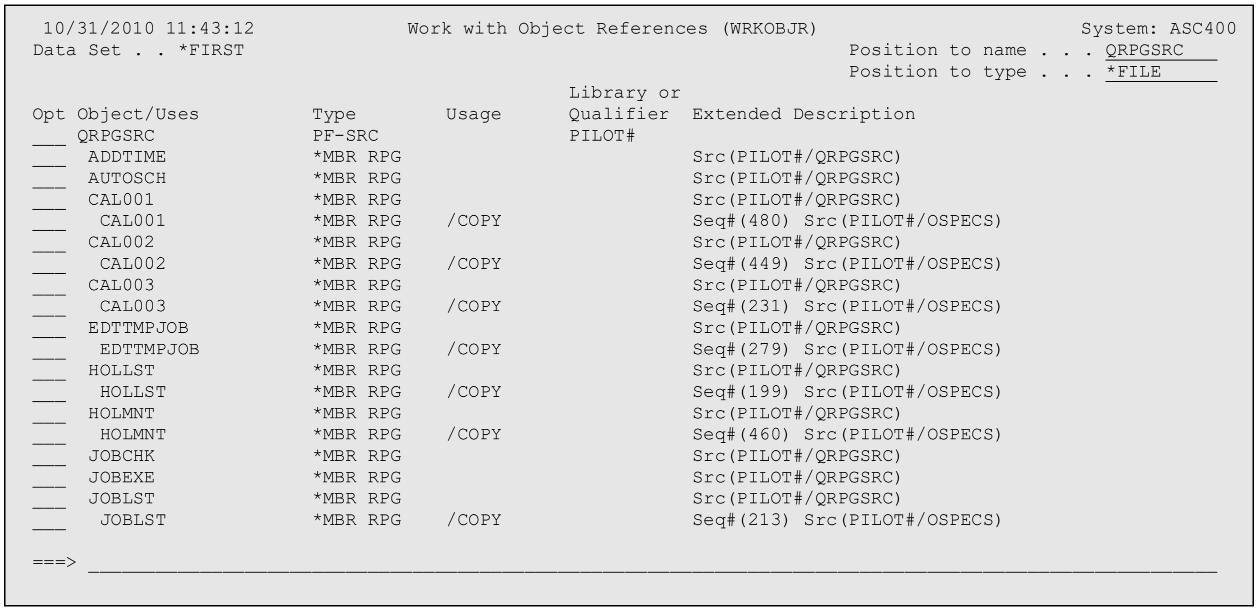
The COBOL and RPG compilers allow you to embed references to other source members within the primary source member that constitutes a program. ABSTRACT initialization will record the references to these copybooks from the source file members that contain them. As with other cross reference information, copybook references can be acquired by using the object list display and report.

Specify an object naming request that includes the source file(s) containing the program source members you are interested in. Do this by using the LIB, OBJ, and OBJTYPE(*FILE) parameters of the reference command, or the subset window available through the action bar or function key 17.

The relationship display must show at least two levels of indentation. Use a value of 2 or higher for the DSPLVL parameter or the object reference level value of the subset window. Alternatively, you can specify a Y value for the Show relations entry of the view menu of the action bar.

Members must be included in the relationship display. Specify DSPMBR(*YES) when using the command, or a Y value for the Members entry of the subset window.

The display below presents an example of copybook references within members of the QRPGSRC source file in the PILOT# library. The full screen form of the object list is used and one of the alternate forms of the display (function key 11) have been selected so that the entire extended description can be viewed.



```
 10/31/2010 11:43:12             Work with Object References (WRKOBJR)                  System: ASC400
Data Set . . *FIRST                                                 Position to name . . . QRPGSRC
                                                                    Position to type . . . *FILE
                                                 Library or
Opt Object/Uses          Type        Usage      Qualifier  Extended Description
___ QRPGSRC              PF-SRC                 PILOT#
___  ADDTIME             *MBR RPG                          Src(PILOT#/QRPGSRC)
___  AUTOSCH             *MBR RPG                          Src(PILOT#/QRPGSRC)
___  CAL001              *MBR RPG                          Src(PILOT#/QRPGSRC)
___   CAL001             *MBR RPG    /COPY                 Seq#(480) Src(PILOT#/OSPECS)
___  CAL002              *MBR RPG                          Src(PILOT#/QRPGSRC)
___   CAL002             *MBR RPG    /COPY                 Seq#(449) Src(PILOT#/OSPECS)
___  CAL003              *MBR RPG                          Src(PILOT#/QRPGSRC)
___   CAL003             *MBR RPG    /COPY                 Seq#(231) Src(PILOT#/OSPECS)
___  EDTTMPJOB           *MBR RPG                          Src(PILOT#/QRPGSRC)
___   EDTTMPJOB          *MBR RPG    /COPY                 Seq#(279) Src(PILOT#/OSPECS)
___  HOLLST              *MBR RPG                          Src(PILOT#/QRPGSRC)
___   HOLLST             *MBR RPG    /COPY                 Seq#(199) Src(PILOT#/OSPECS)
___  HOLMNT              *MBR RPG                          Src(PILOT#/QRPGSRC)
___   HOLMNT             *MBR RPG    /COPY                 Seq#(460) Src(PILOT#/OSPECS)
___  JOBCHK              *MBR RPG                          Src(PILOT#/QRPGSRC)
___  JOBEXE              *MBR RPG                          Src(PILOT#/QRPGSRC)
___  JOBLST              *MBR RPG                          Src(PILOT#/QRPGSRC)
___   JOBLST             *MBR RPG    /COPY                 Seq#(213) Src(PILOT#/OSPECS)

===> ____________________________________________________________________________________________
```

The display can be acquired by using a command like the one below. All of the copybook references within a source file in your application will be listed as shown previously.

```
WRKOBJR LIB(Library) OBJ(File-name) OBJTYPE(*FILE)
        DSPLVL(10) DSPMBR(*YES) DSPFILE(*NO) DSPFLD(*NO)
        DSPFMT(*NO) DSPPGM(*NO) DSPSUB(*NO) DSPOTR(*NO)
        DSPUSG(*ALL) EXPLVL(0)
```

Each member in the source file that includes copybook or subroutine references is listed on the display. Source copy requests are indented below the members that make the request. Using the display above for example, it is apparent that neither the ADDTIME nor the AUTOSCH members include a statement that copies source from another member. (They do however include subroutine references.)

Many of the other members on the display include a copy request. The copy member and source file are shown on the display, along with the statement sequence number where the reference occurs and a /COPY usage value. In the display above, member CAL001 references a copy member named CAL001 in the PILOT#/OSPECS file at sequence 480. Other copybook references on the display show a similar structure.

RPG Subroutine References

RPG subroutines within a program member are also documented by ABSTRACT. The initialization phase will record the references to subroutines so that you can track them through the reference and usage lists. Once the source has been loaded into the cross reference files, you can display references to subroutines using the WRKOBJR, WRKOBJRS, and WRKOBJRX commands.

Specify an object naming request that includes the source file(s) that contain the program source members you are interested in. Do this by using the LIB, OBJ, and OBJTYPE(*FILE) parameters of the reference command, or the subset window available through the action bar or function key 17.

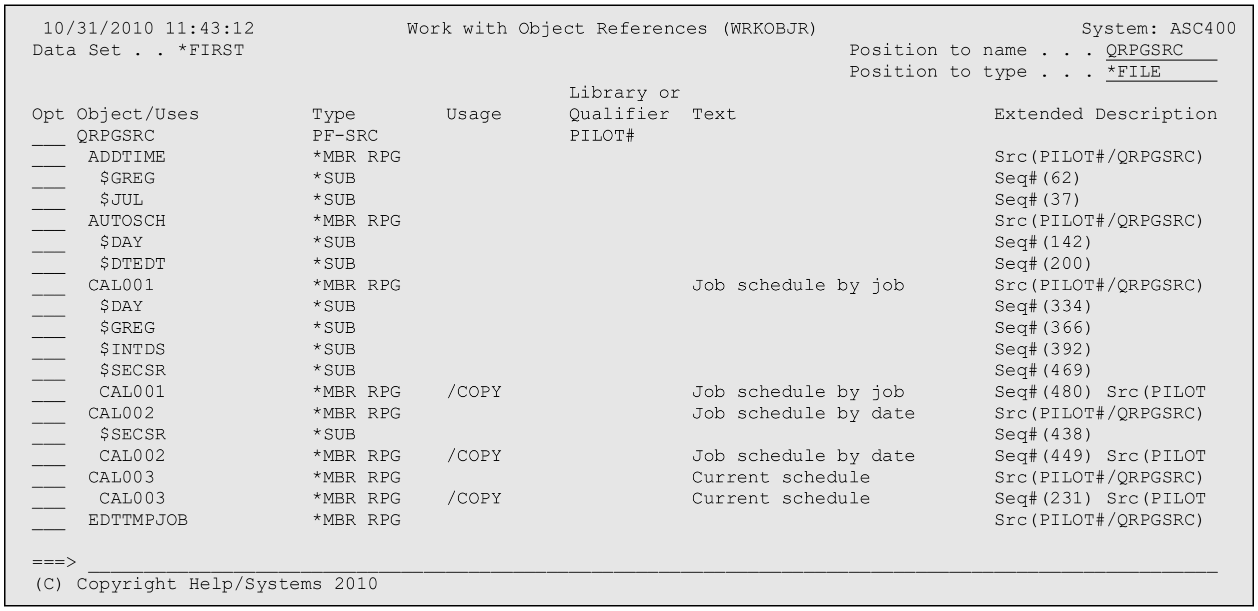
The relationship display must show at least two levels of indentation. Use a value of 2 or higher for the DSPLVL parameter or the object reference level value of the subset window. Alternatively, you can specify a Y value for the Show relations entry of the view menu of the action bar.

Subroutines must be included in the relationship display. Specify DSPSUB(*YES) when using the command, or a Y value for the Subroutines entry of the subset window.

A command to list the subroutine references within an RPG source file looks like this:

```
WRKOBJR LIB(Library) OBJ(File-name) OBJTYPE(*FILE)
     DSPLVL(10) DSPSUB(*YES)DSPFILE(*NO) DSPFLD(*NO)
     DSPFMT(*NO) DSPPGM(*NO)DSPMBR(*NO) DSPOTR(*NO)
     DSPUSG(*ALL) EXPLVL(0)
```

The display below presents an example of subroutine references within members of the QRPGSRC source file in the PILOT# library. Copybook references are also included (DSPMBR(*YES)) on the full screen form of the object list display.



```
 10/31/2010 11:43:12            Work with Object References (WRKOBJR)                      System: ASC400
Data Set . . *FIRST                                                    Position to name . . . QRPGSRC
                                                                       Position to type . . . *FILE
                                                     Library or
Opt Object/Uses          Type        Usage      Qualifier  Text                     Extended Description
___ QRPGSRC              PF-SRC                 PILOT#
___  ADDTIME             *MBR RPG                                                   Src(PILOT#/QRPGSRC)
___   $GREG              *SUB                                                       Seq#(62)
___   $JUL               *SUB                                                       Seq#(37)
___  AUTOSCH             *MBR RPG                                                   Src(PILOT#/QRPGSRC)
___   $DAY               *SUB                                                       Seq#(142)
___   $DTEDT             *SUB                                                       Seq#(200)
___  CAL001              *MBR RPG                          Job schedule by job      Src(PILOT#/QRPGSRC)
___   $DAY               *SUB                                                       Seq#(334)
___   $GREG              *SUB                                                       Seq#(366)
___   $INTDS             *SUB                                                       Seq#(392)
___   $SECSR             *SUB                                                       Seq#(469)
___   CAL001             *MBR RPG    /COPY                 Job schedule by job      Seq#(480) Src(PILOT
___  CAL002              *MBR RPG                          Job schedule by date     Src(PILOT#/QRPGSRC)
___   $SECSR             *SUB                                                       Seq#(438)
___   CAL002             *MBR RPG    /COPY                 Job schedule by date     Seq#(449) Src(PILOT
___  CAL003              *MBR RPG                          Current schedule         Src(PILOT#/QRPGSRC)
___   CAL003             *MBR RPG    /COPY                 Current schedule         Seq#(231) Src(PILOT
___  EDTTMPJOB           *MBR RPG                                                   Src(PILOT#/QRPGSRC)

===> ________________________________________________________________________________________
(C) Copyright Help/Systems 2010
```

Each member in the source file that includes subroutine or copybook references is listed on the display. Subroutine references within the program member are indented below the members that include them. The reference includes both the subroutine name and the sequence number in the source member where it occurs.

The display above shows that member ADDTIME includes two subroutines: $GREG (referenced at sequence 62), and $JUL (referenced at sequence 37). These are the only two subroutines in the program. Subroutine references for other program members have a similar structure on the display.

Other Object References

ABSTRACT initialization will record references to objects in your applications based on the definitions in the command tracking file. Refer to the description of the Work With Command Tracking Definitions (WRKCMDTRKD) command in Part 2 for more information about these definitions and how you can create and change them.

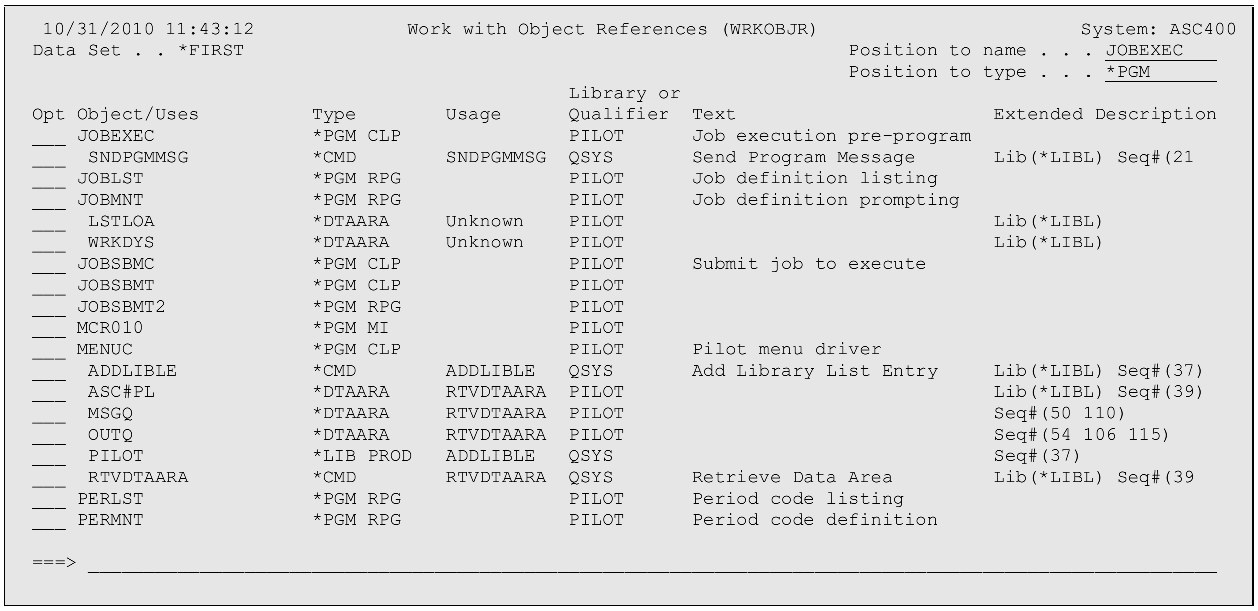
The object reference commands will present the information acquired due to the command tracking definitions on the list display and report in a format similar to that for other types of references. These references are made by CL programs and menu objects in your applications and can be listed by choosing them as parent objects in the list.

> Specify an object naming request that includes the program(s), and/or menu(s) of interest. Do this by using the LIB, OBJ, and OBJTYPE parameters of the reference command, or the subset window available through the action bar or function key 17.
>
> The relationship display must show at least <u>one</u> level of indentation. Use a value of 1 or higher for the DSPLVL parameter or the object reference level value of the subset window. Alternatively, you can specify a Y value for the Show relations entry of the view menu of the action bar.
>
> <u>Other objects</u> must be included in the relationship display. Specify DSPOTR(*YES) when using the command, or a Y value for the Other object types entry of the subset window.

If your application includes user written <u>commands</u> and you want to see the <u>programs</u> they use, specify a non-zero explosion level value (making sure that the display level is at least as large) and include the programs in the list by using DSPPGM(*YES) or a Y value for the Programs entry of the subset window.

The following display presents an example of other object references within the PILOT library. The command used to acquire the display had the form:

```
WRKOBJR LIB(Library) OBJ(*ALL) OBJTYPE(*PGM)
     DSPLVL(10) DSPOTR(*YES) DSPFILE(*NO) DSPFLD(*NO)
     DSPFMT(*NO) DSPPGM(*NO) DSPMBR(*NO) DSPSUB(*NO)
     DSPOTR(*NO) DSPUSG(*ALL)
```



```
 10/31/2010 11:43:12              Work with Object References (WRKOBJR)                 System: ASC400
Data Set . . *FIRST                                                 Position to name . . . JOBEXEC
                                                                    Position to type . . . *PGM
                                                Library or
Opt Object/Uses          Type        Usage      Qualifier  Text                    Extended Description
___ JOBEXEC              *PGM CLP               PILOT      Job execution pre-program
___  SNDPGMMSG           *CMD        SNDPGMMSG  QSYS       Send Program Message    Lib(*LIBL) Seq#(21
___ JOBLST               *PGM RPG               PILOT      Job definition listing
___ JOBMNT               *PGM RPG               PILOT      Job definition prompting
___  LSTLOA              *DTAARA     Unknown    PILOT                              Lib(*LIBL)
___  WRKDYS              *DTAARA     Unknown    PILOT                              Lib(*LIBL)
___ JOBSBMC              *PGM CLP               PILOT      Submit job to execute
___ JOBSBMT              *PGM CLP               PILOT
___ JOBSBMT2             *PGM RPG               PILOT
___ MCR010               *PGM MI                PILOT
___ MENUC                *PGM CLP               PILOT      Pilot menu driver
___  ADDLIBLE            *CMD        ADDLIBLE   QSYS       Add Library List Entry  Lib(*LIBL) Seq#(37)
___  ASC#PL              *DTAARA     RTVDTAARA  PILOT                              Lib(*LIBL) Seq#(39)
___  MSGQ                *DTAARA     RTVDTAARA  PILOT                              Seq#(50 110)
___  OUTQ                *DTAARA     RTVDTAARA  PILOT                              Seq#(54 106 115)
___  PILOT               *LIB PROD   ADDLIBLE   QSYS                               Seq#(37)
___  RTVDTAARA           *CMD        RTVDTAARA  QSYS       Retrieve Data Area      Lib(*LIBL) Seq#(39
___ PERLST               *PGM RPG               PILOT      Period code listing
___ PERMNT               *PGM RPG               PILOT      Period code definition

===> ____________________________________________________________________________________________
```

As with other reference displays, objects meeting the name criteria are presented as parent objects without indentation. Items that are referenced by the parent objects are indented underneath them and described by the object name, attribute, usage, text and description columns of the display.

References to commands are selected when the other object switch is specified. The display above shows references to three commands. The SNDPGMMSG command, used by the JOBEXEC program at sequence 21 is unqualified in the program. ABSTRACT located it in the QSYS library and acquired the text shown on the display from the current command object definition. The same is true for the ADDLIBLE and RTVDTAARA commands used by the MENUC program. In order for these references to be recorded, these commands had tracking definitions enabled for them when the initialization process was run for the PILOT library.

Data areas that are referenced by CL and other HLL programs are presented when other objects are selected. Where possible, ABSTRACT will provide meaningful usage information about the data area. (c.f. the qualified references to ASC#PL, MSGQ, and OUTQ data areas by the MENUC program) In other cases, Unknown usage will be listed. Unqualified references to the LSTLOA and WRKDYS data areas by the JOBMNT RPG program are listed as Unknown because the program reference (DSPPGMREF) information does not indicate how the data area is used.

Additional objects are recorded as indicated by the parameter part of the command tracking definitions. The display above shows that the PILOT library was referenced by the QSYS/ADDLIBLE command at statement 37 in the MENUC program.

Object Usage

Four commands present object usage information on the list display:

WRKOBJU	Work With Object Usage
WRKOBJUX	Work With Cross Referenced Object Usage
WRKFGU	Work With File Group Usage
WRKFLDGU	Work With Field Group Usage

Output from the commands can also be routed to the printer or an output file. The displays and reports from these commands provide the following types of information:

File usage by program: What programs use the files within the application? How are they used?

Field usage: Given a specific field, which ones are referenced by the programs in the application? How are they used? What program(s) change the values for a particular field within a specific file?

Program usage: What programs, user written commands, or menus invoke a particular program? What job stream stacks can cause its execution?

Physical File usage: What logical file(s) are built over a given physical file?

Object usage: Where are specific objects referenced within the application?

Copybook and Subroutine Usage: What programs use a given subroutine or copy member?

This section of the chapter will explain how the object list display and report can be used to acquire and interpret this information.

File Usage

File usage can occur in:

High level language programs
Parameter references by commands in a CL program
Menu objects
Query definitions
SEQUEL views

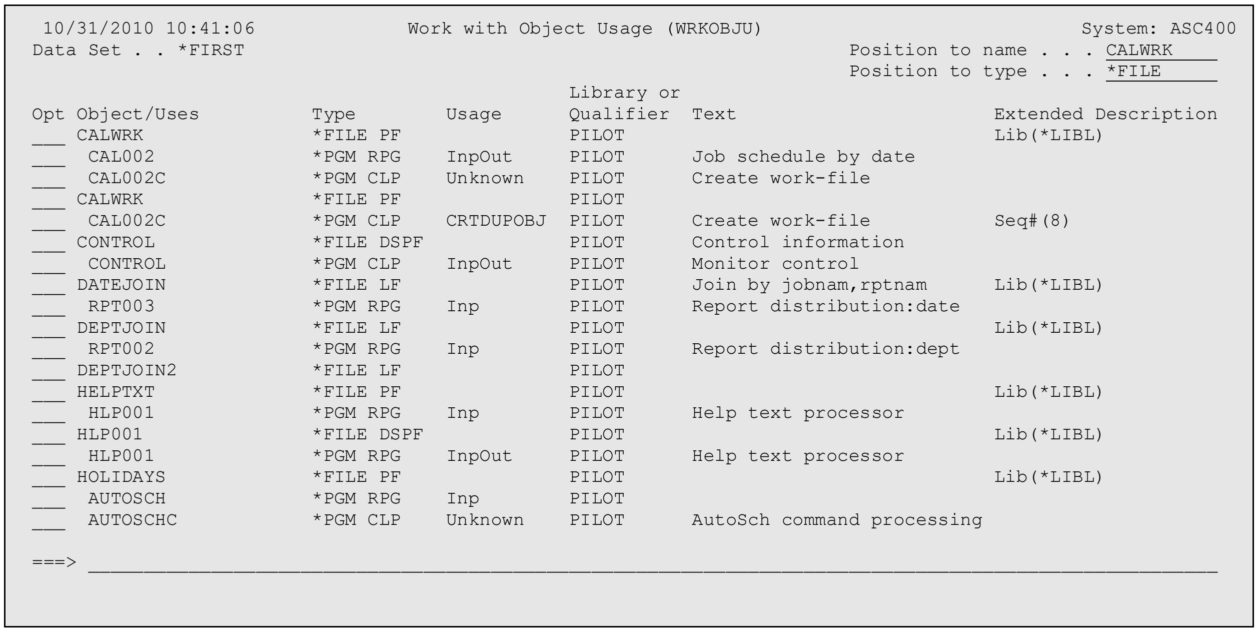
ABSTRACT will record all of these and make the information available to you through the list displays. Obtain usage information about the files in your applications by using any of the usage commands.

Specify an object naming request that includes the file(s) of interest, and make certain that the object type value includes file objects. Do this by using the LIB, OBJ, and OBJTYPE parameters of the usage command, or the subset window available through the action bar or function key 17.
The relationship display must show at least one level of indentation. Use a value of 1 or higher for the DSPLVL parameter or the Object usage level value of the subset window. Alternatively, you can specify a Y value for the Show relations entry of the view menu of the action bar.

Programs, Files, and/or Other objects must be included in the relationship display. Specify DSP-PGM(*YES), DSPFILE(*YES), and/or DSPOTR(*YES) when using the command, or a Y value for the Programs, Files, and/or Other object types entries of the subset window.

The display below presents an example of file usage in the PILOT application. The command used to access the display had the form:

```
WRKOBJU LIB(Library) OBJ(*ALL) OBJTYPE(*FILE) DSPLVL(10)
        DSPFILE(*YES) DSPPGM(*YES) DSPOTR(*YES) DSPFMT(*NO)
        DSPFLD(*NO) DSPMBR(*NO) DSPSUB(*NO) DSPUSG(*ALL)
```



```
 10/31/2010 10:41:06           Work with Object Usage (WRKOBJU)                    System: ASC400
Data Set . . *FIRST                                                Position to name . . . CALWRK
                                                                   Position to type . . . *FILE
                                                 Library or
Opt Object/Uses          Type        Usage       Qualifier  Text                       Extended Description
___ CALWRK               *FILE PF                PILOT                                 Lib(*LIBL)
___  CAL002              *PGM RPG    InpOut      PILOT      Job schedule by date
___  CAL002C             *PGM CLP    Unknown     PILOT      Create work-file
___ CALWRK               *FILE PF                PILOT
___  CAL002C             *PGM CLP    CRTDUPOBJ   PILOT      Create work-file           Seq#(8)
___ CONTROL              *FILE DSPF              PILOT      Control information
___  CONTROL             *PGM CLP    InpOut      PILOT      Monitor control
___ DATEJOIN             *FILE LF                PILOT      Join by jobnam,rptnam      Lib(*LIBL)
___  RPT003              *PGM RPG    Inp         PILOT      Report distribution:date
___ DEPTJOIN             *FILE LF                PILOT                                 Lib(*LIBL)
___  RPT002              *PGM RPG    Inp         PILOT      Report distribution:dept
___ DEPTJOIN2            *FILE LF                PILOT
___ HELPTXT              *FILE PF                PILOT                                 Lib(*LIBL)
___  HLP001              *PGM RPG    Inp         PILOT      Help text processor
___ HLP001               *FILE DSPF              PILOT                                 Lib(*LIBL)
___  HLP001              *PGM RPG    InpOut      PILOT      Help text processor
___ HOLIDAYS             *FILE PF                PILOT                                 Lib(*LIBL)
___  AUTOSCH             *PGM RPG    Inp         PILOT
___  AUTOSCHC            *PGM CLP    Unknown     PILOT      AutoSch command processing

===> ____________________________________________________________________________________________________
```

In accordance with the request, all programs, files and other objects that use (or might use) the files in the PILOT library are presented on the display. Parent objects in the list are obtained from the current contents of the library - text and attribute information matches the current object definition for them.

Usage of the file is indicated by an indented entry underneath the entry that describes the file. The indented entry indicates the type of object making the reference to the file, and specifies how the file is used.

The object lists produced by the WRKOBJU command show both qualified and unqualified usage of the selected files. Unqualified usage, having a LIB(*LIBL) reference in the extended description, precedes usage that is explicitly qualified for the library indicated by the object subset criteria. In some cases, this causes two parent entries to appear in the list: some entries are listed underneath the unqualified (*LIBL) parent file, others are listed underneath the qualified file entry.

For instance, programs CAL002 and CAL002C make unqualified (*LIBL) references to the CAL-WRK physical file. Depending on the library list of a job during execution, the CALWRK file in PILOT may be used by these programs. The first CALWRK entry on the display indicates this.

The second entry for the CALWRK file does not have a LIB(*LIBL) reference in its extended description. This indicates a qualified reference, made explicitly to the CALWRK file in the PILOT library. Underneath this entry we find that program CAL002 issues a Create Duplicate Object (CRT-DUPOBJ) command at sequence number 8. Regardless of the library list, the CAL002 program will use the PILOT/CALWRK file in sequence 8.

In many cases, only one type of entry (qualified or unqualified) will appear. The example display reports a qualified reference to the PILOT/CONTROL display file by PILOT/CONTROLC, a CL program. No other references to the CONTROL display file are recorded in the cross references. Unqualified input references are made to the DATEJOIN and DEPTJOIN logical files by programs RPT003 and RPT002.

Files in the list that are not used will not have any entries listed underneath them. Thus there are no recorded references (either qualified or unqualified) to the DEPTJOIN2 file in the PILOT library.

The report on the next page shows usage information for other files in the PILOT library and demonstrates additional relationships that did not exist in the previous example. Parent objects are listed at the leftmost margin of the report and are denoted with level number 1.

Note that like the previous display example, there are two sets of entries for the JOBSCH and JOBSCHL3 files - those that make an unqualified (*LIBL) reference, and those that explicitly name the file by qualification with the PILOT library.

Logical files that are built on physical files in the list are indented underneath the physical file. There are five logical files listed underneath the JOBSCH physical file on the sample report: DATEJOIN, JOBSCHL1, JOBSCHL2, JOBSCHL3 and JOBSCHL4.

Override definitions are presented in the object lists with both the from-file and to-file serving as parent objects. ABSTRACT allows you to search for file usage using either parameter as a starting point. The from-file part of the override is shown without an attribute or library reference.

The programs that issue the override command are listed beneath the from-file or to-file reference. The extended description completes the information for the override reference. ABSTRACT will suppress duplicate references if an override references the same name as both the from-file and the to-file parameters.

```
 10/31/2010 13:33:16                                 Abstract                           Page . . . :   1
                                                   Object Usage
Object name . . : JOBSCH*                                                                 System . . : ASC400
Object type . . : *FILE                                                                   Data Set . : *FIRST
Object library  : PILOT
                                            Library or
   Object/Used by  Type         Usage      Qualifier  Extended Description              Text

 1 JOBSCH          *FILE PF                PILOT      Lib(*LIBL)                        Job Schedule
 2 .AUTOSCH        *PGM RPG     InpOut     PILOT
 2 .AUTOSCHC       *PGM CLP     Unknown    PILOT                                        AutoSch command pgm
 2 .CAL003         *PGM RPG     Inp        PILOT                                        Current schedule
 2 .EDTTMPJOB      *PGM RPG     Out        PILOT
 2 .JOBSBMT        *PGM CLP     Unknown    PILOT
 2 .JOBSBMT2       *PGM RPG     Out        PILOT
 2 .JOBSCH         *FILE
 3 ..AUTOSCHC      *PGM CLP     OVRDBF     PILOT      TOFILE(JOBSCH) Seq#(35)           AutoSch command pgm
 2 .JOBSCHVIEW     SQLVIEW      SEQUELView QUERYLIB                                     Job view by date
 2 .PRG001        *PGM RPG     Inp         PILOT                                           Schedule purge prompt
 2 .PRG002         *PGM RPG     Upd        PILOT                                        Schedule purge
 2 .PURGESCHC      *PGM CLP     Unknown    PILOT
 2 .SCHMNT         *PGM RPG     OutUpd     PILOT                                        Schedule maintenance
 2 .SCHTMPJOB      *PGM CLP     Unknown    PILOT
 2 .SCH001         *PGM RPG     OutUpd     PILOT                                        Schedule loader
 1 JOBSCH          *FILE PF                PILOT                                        Job Schedule
 2 .AUTOSCHC       *PGM CLP     Unknown    PILOT                                        AutoSch command pgm
 2 .DATEJOIN       *FILE LF                PILOT                                        Join by jobnam,rptnam
 2 .JOBSBMT        *PGM CLP     Unknown    PILOT
 2 .JOBSCH         *FILE
 3 ..JOBSBMT       *PGM CLP     OVRDBF     PILOT      TOFILE(PILOT/JOBSCH) Seq#(70)
 3 ..PURGESCHC     *PGM CLP     OVRDBF     PILOT      TOFILE(PILOT/JOBSCH) Seq#(22)
 3 ..SCHTMPJOB     *PGM CLP     OVRDBF     PILOT      TOFILE(PILOT/JOBSCH) Seq#(37)
 2 .JOBSCHL1       *FILE LF                PILOT
 2 .JOBSCHL2       *FILE LF                PILOT
 2 .JOBSCHL3       *FILE LF                PILOT
 2 .JOBSCHL4       *FILE LF                PILOT
 2 .JOBVIEW        SQLVIEW      SEQUELView QUERYLIB
 2 .PILOTDTA       *PGM CLP     Out        PILOT
 2 .PURGESCHC      *PGM CLP     Unknown    PILOT
 2 .SCHTMPJOB      *PGM CLP     Unknown    PILOT
 1 JOBSCHL1        *FILE LF                PILOT
 2 .JOBSCHQRY      *QRYDFN QRY  Query      QUERYLIB                                     Simple schedule query
 1 JOBSCHL2        *FILE LF                PILOT      Lib(*LIBL)
 2 .JOBMNT         *PGM RPG     Upd        PILOT                                        Job definition
 1 JOBSCHL3        *FILE LF                PILOT      Lib(*LIBL)
 2 .JOBEXE         *PGM RPG     OutUpd     PILOT                                        Job Execution
 2 .JOBEXEC        *PGM CLP     Unknown    PILOT                                        Job execution pre-pgm
 2 .JOBMNT         *PGM RPG     Upd        PILOT                                        Job definition
 2 .PRG001         *PGM RPG     Inp        PILOT                                        Schedule purge prompt
 2 .RNMJOB         *PGM RPG     Upd        PILOT
 2 .SCHMNT         *PGM RPG     Inp        PILOT                                        Schedule maintenance
 1 JOBSCHL3        *FILE LF                PILOT
 2 .JOBEXEC        *PGM CLP     Unknown    PILOT                                        Job execution pre-pgm
 2 .JOBSCHL3       *FILE
 3 ..JOBEXEC       *PGM CLP     OVRDBF     PILOT      TOFILE(PILOT/JOBSCHL3) Seq#(11)   Job execution pre-pgm
 1 JOBSCHL4        *FILE LF                PILOT      Lib(*LIBL)
 2 .JOBCHK         *PGM RPG     Upd        PILOT                                        Scheduling - Check
```

The report demonstrates several examples of override usage. Listed under the unqualified (*LIBL) entry for the JOBSCH file is another JOBSCH file entry that has the AUTOSCHC program beneath it. This entry indicates the command:

```
OVRDBF FROMFILE(JOBSCH) TOFILE(*LIBL/JOBSCH)
```

is used at statement 35 within the AUTOSCHC program. Similar override commands (with PILOT qualified to-file values) are listed further down in the report for programs JOBSBMT, PURGESCHC, and SCHTMPJOB.

Other references on the report list file usage for **SEQUEL views** and **Query/400 objects**. Command parameter usage of files will be listed in a similar form, showing the program (or menu) that referenced the file and the command that caused the usage.

File Group References and Usage

In many cases, you will be interested in information about the usage of your application data regardless of the specific file name through which it occurs. ABSTRACT provides file group reference and usage commands to help you find all the references to physical data — whether the reference occurs through a physical file or one of the logical files that are built over it, or an override that specifies any of them as a target.

The commands begin by locating the file you name in the LIB and FILE parameters and acquiring the names of all related database files. If you supply a physical file, ABSTRACT determines the logical files that use it. If you supply a logical file name, ABSTRACT determines all the physicals contributing to the logical and then locates all the logical files built over this group of physicals. This operation takes place in real-time, without reference to the cross reference files.

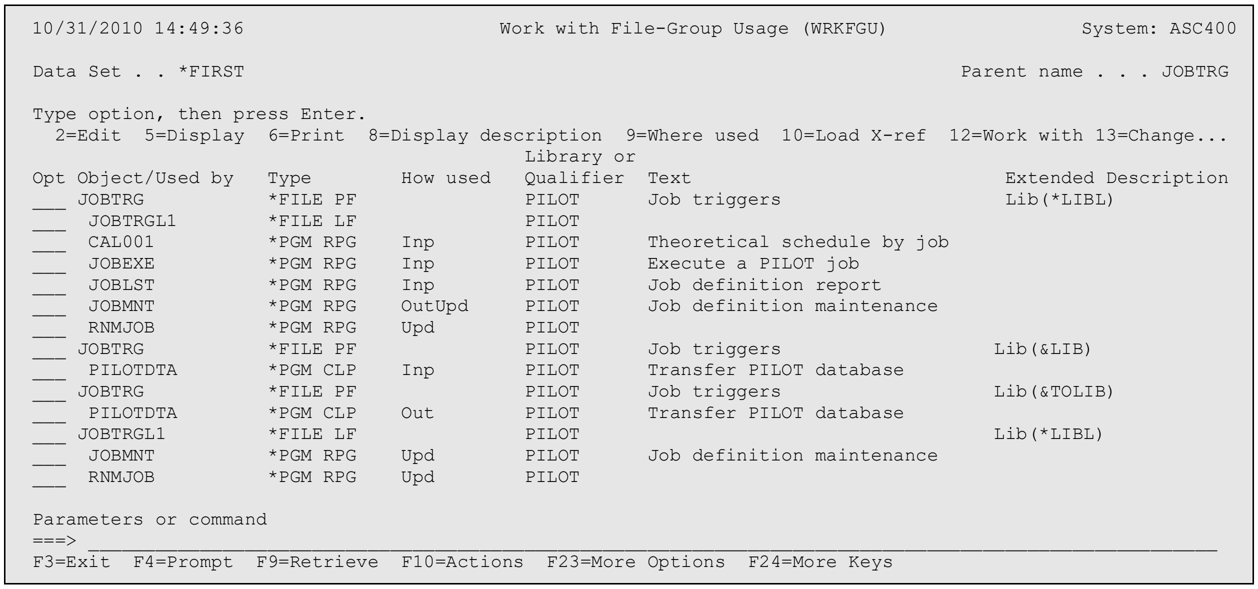
After the files in the group have been identified, the reference or usage information from the cross reference dictionary is presented on a list display. Depending on your needs, you can view the files in the group as parents (WRKFGU) or as dependents on a references list (WRKFGR).

The WRKFGR and WRKFGU commands will present both qualified and unqualified usage of the files in the group.

Display the file group usage list by using a command of the form:

```
WRKFGU LIB(library) FILE(File-name) DSPLVL(10)
      DSPFILE(*YES) DSPPGM(*YES) DSPOTR(*YES)
      DSPFMT(*NO) DSPFLD(*NO) DSPMBR(*NO)
      DSPSUB(*NO) DSPUSG(*ALL)
```

The next display shows usage information for the file group that contains the JOBTRGL1 file in the PILOT library. Each file in the group appears as a parent object on the display. Qualified and unqualified references to the physical file or any of the logicals built on it appear on the display.



```
10/31/2010 14:49:36                  Work with File-Group Usage (WRKFGU)                System: ASC400

Data Set . . *FIRST                                                                Parent name . . . JOBTRG

Type option, then press Enter.
  2=Edit  5=Display  6=Print  8=Display description  9=Where used  10=Load X-ref  12=Work with 13=Change...
                                                   Library or
Opt Object/Used by   Type       How used   Qualifier  Text                          Extended Description
___ JOBTRG           *FILE PF              PILOT      Job triggers                  Lib(*LIBL)
___  JOBTRGL1        *FILE LF              PILOT
___  CAL001          *PGM RPG   Inp        PILOT      Theoretical schedule by job
___  JOBEXE          *PGM RPG   Inp        PILOT      Execute a PILOT job
___  JOBLST          *PGM RPG   Inp        PILOT      Job definition report
___  JOBMNT          *PGM RPG   OutUpd     PILOT      Job definition maintenance
___  RNMJOB          *PGM RPG   Upd        PILOT
___ JOBTRG           *FILE PF              PILOT      Job triggers                 Lib(&LIB)
___  PILOTDTA        *PGM CLP   Inp        PILOT      Transfer PILOT database
___ JOBTRG           *FILE PF              PILOT      Job triggers                 Lib(&TOLIB)
___  PILOTDTA        *PGM CLP   Out        PILOT      Transfer PILOT database
___ JOBTRGL1         *FILE LF              PILOT                                   Lib(*LIBL)
___  JOBMNT          *PGM RPG   Upd        PILOT      Job definition maintenance
___  RNMJOB          *PGM RPG   Upd        PILOT

Parameters or command
===> ___________________________________________________________________________________________
F3=Exit  F4=Prompt  F9=Retrieve  F10=Actions  F23=More Options  F24=More Keys
```

Refer to the discussion regarding file usage information that begins on page 3-33 for an explanation of the WRKFGU display and report.

The following report contrasts with the WRKFGU display above by presenting the same information in an object references format. Here, parent objects are those that refer to the elements of the file group. The report is considerably longer than the display above because it includes all references from each parent, not just the references to the files in the file group.

```
 10/31/2010 14:47:34                                   Abstract              Page . . . :   1
                                                 File Group References
File library  . : PILOT                                                       System . . : ASC400
File name . . . : JOBTRGL1                                                    Data Set . : *FIRST

                                            Library or
   Object/uses  Type         Usage    Qualifier  Text                                  Description
   CAL001       *PGM RPG              PILOT      Theoretical schedule by job report
 1 .CALHDR      *FILE PF     Inp      PILOT                                            Lib(*LIBL)
 1 .HOLIDAYS    *FILE PF     Inp      PILOT      Holiday file                          Lib(*LIBL)
 1 .JOBDTL      *FILE PF     Inp      PILOT                                            Lib(*LIBL)
 1 .JOBDTLV     *FILE PF     Inp      PILOT                                            Lib(*LIBL)
 1 .JOBHDR      *FILE PF     Inp      PILOT                                            Lib(*LIBL)
 1 .JOBLDA      *FILE PF     Inp      PILOT                                            Lib(*LIBL)
 1 .JOBTRG      *FILE PF     Inp      PILOT      Job triggers                          Lib(*LIBL)
 1 .PERDTL      *FILE PF     Inp      PILOT      Period code detail                    Lib(*LIBL)
 1 .PERHDR      *FILE PF     Inp      PILOT      Period code header                    Lib(*LIBL)
 1 .PILOTPR     *FILE PRTF   Out      PILOT                                            Lib(*LIBL)
 1 .PLTDATE     *FILE PF     Inp      PILOT      PILOT date file                       Lib(*LIBL)
 1 .PLTDATL1    *FILE LF     Inp      PILOT                                            Lib(*LIBL)
 1 .PLTDATL2    *FILE LF     Inp      PILOT                                            Lib(*LIBL)
 1 .PERHDR      *FILE PF     Inp      PILOT      Period code header                    Lib(*LIBL)
 1 .PLTDDM      *FILE PF     Inp      PILOT                                            Lib(*LIBL)
 1 .PLTDFTS     *FILE PF     Inp      PILOT      Job Header defaults                   Lib(*LIBL)
 1 .PLTLIBL     *FILE PF     Inp      PILOT                                            Lib(*LIBL)
 1 .RPTDTL      *FILE PF     OutUpd   PILOT      Report distribution detail            Lib(*LIBL)
 1 .RPTHDR      *FILE PF     OutUpd   PILOT      Report distribution header            Lib(*LIBL)
   PILOTDTA     *PGM CLP              PILOT      Transfer PILOT database
 1 .*ALL        *FILE        Unknown  *LIBL                                            Lib(*LIBL)
 1 .CALHDR      *FILE PF     Unknown  PILOT                                            Lib(*LIBL)
 1 .CALHDR      *FILE PF     Inp      PILOT                                            Lib(&LIB)
 1 .CALHDR      *FILE PF              PILOT                                            Lib(&TOLIB)
 1 .HOLIDAYS    *FILE PF     Unknown  PILOT      Holiday file                          Lib(*LIBL)
 1 .HOLIDAYS    *FILE PF     Inp      PILOT      Holiday file                          Lib(&LIB)
 1 .HOLIDAYS    *FILE PF              PILOT      Holiday file                          Lib(&TOLIB)
 1 .JOBDTL      *FILE PF     Inp      PILOT                                            Lib(&LIB)
 1 .JOBDTL      *FILE PF     Out      PILOT                                            Lib(&TOLIB)
 1 .JOBDTLV     *FILE PF     Inp      PILOT                                            Lib(&LIB)
 1 .JOBDTLV     *FILE PF     Out      PILOT                                            Lib(&TOLIB)
 1 .JOBHDR      *FILE PF     Unknown  PILOT                                            Lib(*LIBL)
 1 .JOBHDR      *FILE PF     Inp      PILOT                                            Lib(&LIB)
 1 .JOBHDR      *FILE PF              PILOT                                            Lib(&TOLIB)
 1 .JOBLDA      *FILE PF     Inp      PILOT                                            Lib(&LIB)
 1 .JOBLDA      *FILE PF     Out      PILOT                                            Lib(&TOLIB)
 1 .JOBLOG      *FILE PF     Inp      PILOT                                            Lib(&LIB)
 1 .JOBLOG      *FILE PF     Out      PILOT                                            Lib(&TOLIB)
 1 .JOBSCH      *FILE PF     Inp      PILOT                                            Lib(&LIB)
 1 .JOBSCH      *FILE PF     Out      PILOT                                            Lib(&TOLIB)
 1 .JOBSCHV     *FILE PF     Inp      PILOT                                            Lib(&LIB)
 1 .JOBSCHV     *FILE PF     Out      PILOT                                            Lib(&TOLIB)
 1 .JOBTRG      *FILE PF     Inp      PILOT      Job triggers                          Lib(&LIB)
 1 .JOBTRG      *FILE PF     Out      PILOT      Job triggers                          Lib(&TOLIB)
 1 .LOGFILE     *FILE PF     Inp      PILOT      Saved job logs by job                 Lib(&LIB)
 1 .LOGFILE     *FILE PF     Out      PILOT      Saved job logs by job                 Lib(&TOLIB)
 1 .PERDTL      *FILE PF     Inp      PILOT      Period code detail                    Lib(&LIB)
 1 .PERDTL      *FILE PF     Out      PILOT      Period code detail                    Lib(&TOLIB)
 1 .PERHDR      *FILE PF     Unknown  PILOT      Period code header                    Lib(*LIBL)
 1 .PERHDR      *FILE PF     Inp      PILOT      Period code header                    Lib(&LIB)
 1 .PERHDR      *FILE PF              PILOT      Period code header                    Lib(&TOLIB)
 1 .PLTDATE     *FILE PF     Inp      PILOT      PILOT date file                       Lib(&LIB)
 1 .PLTDATE     *FILE PF     Out      PILOT      PILOT date file                       Lib(&TOLIB)
 1 .PLTDDM      *FILE PF     Inp      PILOT                                            Lib(&LIB)
 1 .PLTDDM      *FILE PF     Out      PILOT                                            Lib(&TOLIB)
 1 .PLTDFTS     *FILE PF     Inp      PILOT      Job Header defaults                   Lib(&LIB)
 1 .PLTDFTS     *FILE PF     Out      PILOT      Job Header defaults                   Lib(&TOLIB)
 1 .PLTLIBL     *FILE PF     Inp      PILOT                                            Lib(&LIB)
 1 .PLTLIBL     *FILE PF     Out      PILOT                                            Lib(&TOLIB)
 1 .RPTDTL      *FILE PF     Inp      PILOT      Report distribution detail            Lib(&LIB)
```

Field Usage

You can acquire field usage information that has been cataloged by ABSTRACT by using the WRK-OBJUX command. Because fields are not true objects, you will not be able to use the WRKOBJU command to document their usage. To obtain usage information about the fields in your applications:

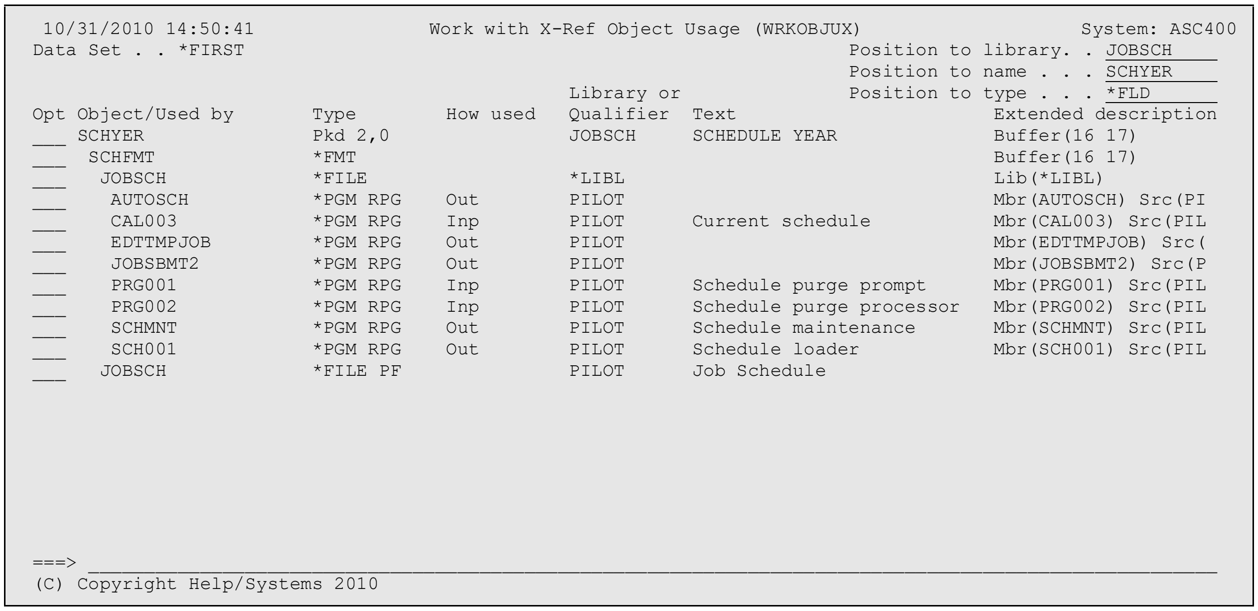
specify the field you are interested in, qualify it with the file where it is located, and specify OBJTYPE(*FLD) on the usage command, or the subset window available through the action bar or function key 17;

show at least three levels of indentation. Use a value of 3 or higher for the DSPLVL parameter or the Object usage level value of the subset window. Alternatively, specify a Y value for the Show relations entry of the view menu of the action bar;

include programs in the relationship display. Files and formats may also be chosen, but are not required for useful information. Specify DSPPGM(*YES), DSPFILE(*YES) and DSP-FMT(*YES) when using the command, or Y values for the Programs, Files and Formats entries of the subset window.

The display below presents an example of field usage in the PILOT application. The command used to access the display had the form:

```
WRKOBJUX LIB(file-name) OBJ(field-name) OBJTYPE(*FLD) DSPLVL(10)
      DSPPGM(*YES) DSPFILE(*YES) DSPFMT(*YES) DSPFLD(*NO)
      DSPMBR(*NO) DSPSUB(*NO) DSPOTR(*NO) DSPUSG(*ALL)
```



```
 10/31/2010 14:50:41            Work with X-Ref Object Usage (WRKOBJUX)                System: ASC400
Data Set . . *FIRST                                                Position to library. . JOBSCH
                                                                   Position to name . . . SCHYER
                                               Library or          Position to type . . . *FLD
Opt Object/Used by        Type        How used   Qualifier  Text                     Extended description
___ SCHYER                Pkd 2,0                JOBSCH     SCHEDULE YEAR            Buffer(16 17)
___  SCHFMT               *FMT                                                       Buffer(16 17)
___   JOBSCH              *FILE                  *LIBL                               Lib(*LIBL)
___    AUTOSCH            *PGM RPG    Out        PILOT                               Mbr(AUTOSCH) Src(PI
___    CAL003             *PGM RPG    Inp        PILOT      Current schedule         Mbr(CAL003) Src(PIL
___    EDTTMPJOB          *PGM RPG    Out        PILOT                               Mbr(EDTTMPJOB) Src(
___    JOBSBMT2           *PGM RPG    Out        PILOT                               Mbr(JOBSBMT2) Src(P
___    PRG001             *PGM RPG    Inp        PILOT      Schedule purge prompt    Mbr(PRG001) Src(PIL
___    PRG002             *PGM RPG    Inp        PILOT      Schedule purge processor Mbr(PRG002) Src(PIL
___    SCHMNT             *PGM RPG    Out        PILOT      Schedule maintenance     Mbr(SCHMNT) Src(PIL
___    SCH001             *PGM RPG    Out        PILOT      Schedule loader          Mbr(SCH001) Src(PIL
___   JOBSCH              *FILE PF               PILOT      Job Schedule








===> __________________________________________________________________________________________
(C) Copyright Help/Systems 2010
```

The display lists the attributes of the requested field(s), and text and buffer positions within the indicated file. As with other usage lists, both qualified and unqualified references are provided. All of the references shown on the sample display above are made through unqualified use of the JOBSCH file.

Two types of references can appear on a field usage entry: Input (Inp) and Output (Out). If the field is modified by the program that uses it and the file is opened for record update or addition, the field is marked as having an output reference. If the field is referenced by the program and the conditions for an output reference are not met, it is marked as an input reference. If the field is not referenced by the program or its display or communications files then it will not appear on the field or field group usage/reference lists unless it exists in a record that is processed with a WRITE operation.

Field Group Usage

In many cases, you will be interested in information about the usage of your application data regardless of the specific file name through which it occurs. ABSTRACT provides field group usage commands to help you find all the references to physical data - whether the reference occurs through a physical file or one of the logical files that are built over it, or an override that specifies any of them as a target.

Like the file group commands, field group commands begin by locating the file you name in the LIB and FILE parameters and acquiring the names of all related database files. If you supply a physical file, ABSTRACT determines the logical files that use it. If you supply a logical file name, ABSTRACT determines all the physicals contributing to the logical and then locates all the logical files built over this group of physicals. This operation takes place in real-time, without reference to the cross reference files.

After the files in the group have been identified, the field usage information from the cross reference dictionary is presented on a list display. Information is presented for each of the files in the group based on the position of the field you selected on the command.

Like the WRKOBJU command, the WRKFLDGU command will present both qualified and unqualified usage of the files in the group. This contrasts with the WRKOBJUX command that shows only usage that exactly matches the qualified names that you supply.

Display the field group usage list by using a command of the form:

```
WRKFLDGU LIB(Library) FILE(File-name) FIELD(field-name)
      DSPLVL(10) DSPPGM(*YES) DSPFILE(*YES) DSPFMT(*YES)
      DSPFLD(*NO) DSPMBR(*NO) DSPSUB(*NO) DSPUSG(*ALL)
      DSPOTR(*NO)
```

The printout on the next page shows field group usage information for the SCHYER field of the JOBSCH file group in the PILOT library. Qualified and unqualified references to the physical file or any of the logicals built on it appear on the report.

Each field that shares the positions used by the SCHYER field in the JOBSCH file is listed as a parent object on the report with level number 1 to its left.

Because DSPFMT(*YES) was specified on the usage request, each format is shown for each use of the fields.

Indented underneath the format entry is the name of the file through which the reference to the field occurs. The file entry indicates whether the programs underneath it access the file with a qualified or unqualified reference. Each of the references on the sample report are made by RPG programs (which cannot make qualified file references).

The sample report on the next page is a powerful demonstration of how useful field group reporting can be. It shows very clearly that only six of the programs in the entire application can modify the value of the SCHYER field in the JOBSCH file. Five of them perform the modifications through the JOBSCH file, the sixth uses the JOBSCHL3 file. There may be many other programs in the application that use these *files*, but the eleven programs listed on the report are the only ones that reference the positions occupied by the SCHYER *field*. (Refer to the file group commands for information about programs that use the database regardless of file access).

```
 10/31/2010 16:48:08                        Abstract                            Page . . . :   1
                                       Field Group Usage
Object name . . : JOBSCH                                                            System . . : ASC400
Object type . . : *FILE                                                             Data Set . : *FIRST
Object library  : PILOT
                                   Library or
Object/Used by   Type        Usage    Qualifier  Extended Description                    Text

 1 SCHYER        Pkd 2,0              DATEJOIN   Buffer(75 76)                           SCHEDULE YEAR
 2 .DTEFMT       *FMT                            Buffer(75 76)
 3 ..DATEJOIN    *FILE                *LIBL      Lib(*LIBL)
 4 ...RPT003     *PGM RPG    Inp      PILOT      Mbr(RPT003) Src(PILOT#/QRPGSRC)         Report distribution
 1 SCHYER        Pkd 2,0              JOBSCH     Buffer(16 17)                           SCHEDULE YEAR
 2 .SCHFMT       *FMT                            Buffer(16 17)
 3 ..JOBSCH      *FILE                *LIBL      Lib(*LIBL)
 4 ...AUTOSCH    *PGM RPG    Out      PILOT      Mbr(AUTOSCH) Src(PILOT#/QRPGSRC)
 4 ...CAL003     *PGM RPG    Inp      PILOT      Mbr(CAL003) Src(PILOT#/QRPGSRC)         Current schedule
 4 ...EDTTMPJOB  *PGM RPG    Out      PILOT      Mbr(EDTTMPJOB) Src(PILOT#/QRPGSRC)
 4 ...JOBSBMT2   *PGM RPG    Out      PILOT      Mbr(JOBSBMT2) Src(PILOT#/QRPGSRC)
 4 ...PRG001     *PGM RPG    Inp      PILOT      Mbr(PRG001) Src(PILOT#/QRPGSRC)         Schedule purge pmt
 4 ...PRG002     *PGM RPG    Inp      PILOT      Mbr(PRG002) Src(PILOT#/QRPGSRC)         Schedule purge pgm
 4 ...SCHMNT     *PGM RPG    Out      PILOT      Mbr(SCHMNT) Src(PILOT#/QRPGSRC)         Schedule maint
 4 ...SCH001     *PGM RPG    Out      PILOT      Mbr(SCH001) Src(PILOT#/QRPGSRC)         Schedule loader
 1 SCHYER        Pkd 2,0              JOBSCHL1   Buffer(16 17)                           SCHEDULE YEAR
 1 SCHYER        Pkd 2,0              JOBSCHL2   Buffer(16 17)                           SCHEDULE YEAR
 1 SCHYER        Pkd 2,0              JOBSCHL3   Buffer(16 17)                           SCHEDULE YEAR
 2 .SCHFMT       *FMT                            Buffer(16 17)
 3 ..JOBSCHL3    *FILE                *LIBL      Lib(*LIBL)
 4 ...JOBEXE     *PGM RPG    Out      PILOT      Mbr(JOBEXE) Src(PILOT#/QRPGSRC)         Job Execution
 4 ...PRG001     *PGM RPG    Inp      PILOT      Mbr(PRG001) Src(PILOT#/QRPGSRC)         Schedule purge pmt
 4 ...SCHMNT     *PGM RPG    Inp      PILOT      Mbr(SCHMNT) Src(PILOT#/QRPGSRC)         Schedule maint
 1 SCHYER        Pkd 2,0              JOBSCHL4   Buffer(16 17)                           SCHEDULE YEAR
 2 .SCHFMT       *FMT                            Buffer(16 17)
 3 ..JOBSCHL4    *FILE                *LIBL      Lib(*LIBL)
 4 ...JOBCHK     *PGM RPG    Inp      PILOT      Mbr(JOBCHK) Src(PILOT#/QRPGSRC)         Scheduling - Check
```

Refer to the discussion regarding field usage information that begins on page 3-39 for an explanation of the WRKFLDGU display and report.

Program Usage

Program usage can occur as a result of:

CALL/TFRCTL commands in CL or HLL programs
RPG calls to programs through named variables and constants
Parameter references by commands in a CL program
Menu objects
Command references to CPP/VCP programs
Routing entries in subsystem descriptions
Initial program references of user profile definitions
PILOT and ROBOT Job scheduler definitions
CA-PRMS Menu Manager definitions

ABSTRACT will record all of these and make the information available to you through the list displays. The WRKOBJU command can display a multiple level where-used explosion that shows the job stream that can cause a program to be run in addition to a simple program usage.

To obtain usage information about the programs in your applications:

Specify an object naming request that includes the program(s) of interest. Do this by using the LIB, OBJ, and OBJTYPE(*PGM) parameters of the usage command, or the subset window available through the action bar or function key 17.

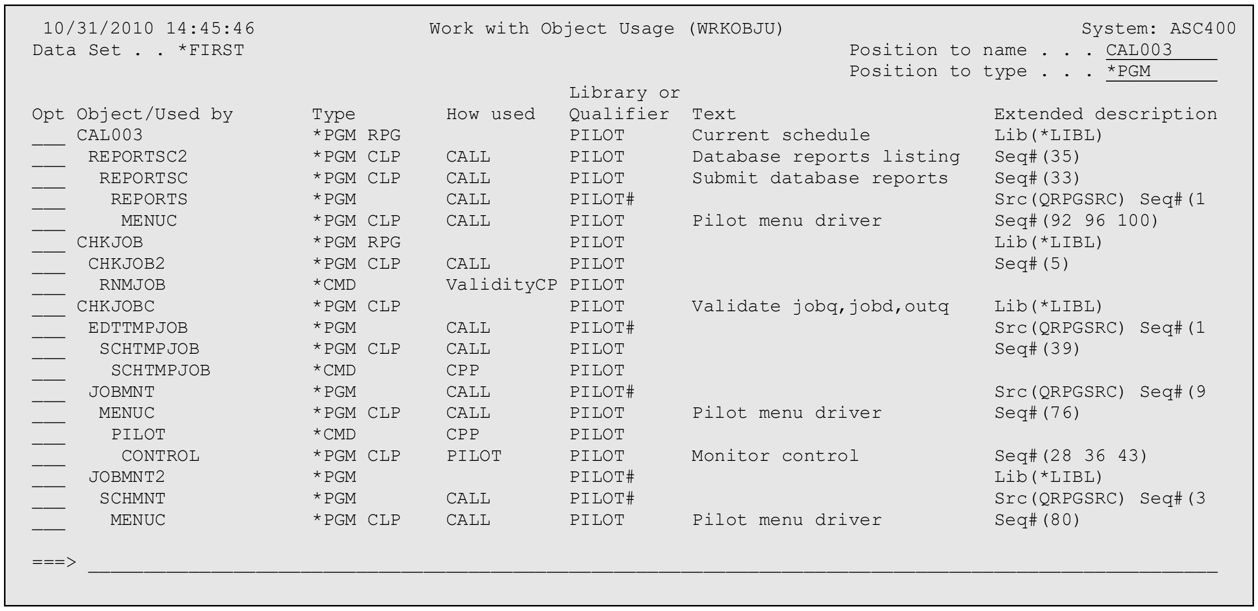
Show at least one level of indentation. Use a value of 1 or higher for the DSPLVL parameter or the Object usage level value of the subset window. Alternatively, you can specify a Y value for the Show relations entry of the view menu of the action bar.

If you want WRKOBJR to show a multiple level explosion, you should set the explosion level value to the number of invocation levels you want to see (up to 10). Specify this value using the EXPLVL parameter or the Object usage explosion level entry of the subset window. The DSPLVL value should be set to a value at least as high as the explosion level.

Programs must be included in the relationship display. Specify DSPPGM(*YES) when using the command, or a Y value for the Programs entry of the subset window.

If your application includes user written commands and you want to see references to them and subsequent invocation of their command processing and validity checking programs, include the commands in the list by using DSPOTR(*YES) or a Y value for the Other object types entry of the subset window. Use the same value if you want to see references through iSeries menus.

The following display presents examples of program usage in the PILOT library.



```
 10/31/2010 14:45:46                Work with Object Usage (WRKOBJU)                          System: ASC400
Data Set . . *FIRST                                                      Position to name . . . CAL003____
                                                                         Position to type . . . *PGM______
                                                Library or
Opt Object/Used by       Type        How used   Qualifier  Text                       Extended description
___ CAL003               *PGM RPG               PILOT      Current schedule           Lib(*LIBL)
___  REPORTSC2           *PGM CLP    CALL       PILOT      Database reports listing   Seq#(35)
___   REPORTSC           *PGM CLP    CALL       PILOT      Submit database reports    Seq#(33)
___    REPORTS           *PGM        CALL       PILOT#                                Src(QRPGSRC) Seq#(1
___     MENUC            *PGM CLP    CALL       PILOT      Pilot menu driver          Seq#(92 96 100)
___ CHKJOB               *PGM RPG               PILOT                                 Lib(*LIBL)
___  CHKJOB2             *PGM CLP    CALL       PILOT                                 Seq#(5)
___   RNMJOB             *CMD        ValidityCP PILOT
___ CHKJOBC              *PGM CLP               PILOT      Validate jobq,jobd,outq    Lib(*LIBL)
___  EDTTMPJOB           *PGM        CALL       PILOT#                                Src(QRPGSRC) Seq#(1
___   SCHTMPJOB          *PGM CLP    CALL       PILOT                                 Seq#(39)
___    SCHTMPJOB         *CMD        CPP        PILOT
___  JOBMNT              *PGM        CALL       PILOT#                                Src(QRPGSRC) Seq#(9
___   MENUC              *PGM CLP    CALL       PILOT      Pilot menu driver          Seq#(76)
___    PILOT             *CMD        CPP        PILOT
___     CONTROL          *PGM CLP    PILOT      PILOT      Monitor control            Seq#(28 36 43)
___  JOBMNT2             *PGM                   PILOT#                                Lib(*LIBL)
___   SCHMNT             *PGM        CALL       PILOT#                                Src(QRPGSRC) Seq#(3
___    MENUC             *PGM CLP    CALL       PILOT      Pilot menu driver          Seq#(80)

===> _____________________________________________________________________________________________________
```

The command used to acquire the display had the form:

```
WRKOBJU LIB(library) OBJ(*ALL) OBJTYPE(*PGM) DSPLVL(10)
        EXPLVL(3) DSPPGM(*YES) DSPOTR(*YES)
        DSPFILE(*NO) DSPFLD(*NO) DSPFMT(*NO)
        DSPMBR(*NO) DSPSUB(*NO) DSPUSG(*ALL)
```

Parent objects on the display exist in the PILOT library. Each indented entry uses the entry above it through the command listed in the How used column. The Library column indicates the location of the CL program object or source library containing the HLL source code. The extended description column on the display shows the sequence number(s) in the program where the parent object is used.

The CAL003 RPG program, for example, is called by the REPORTSC2 CL program. The unqualified (*LIBL) CALL occurs at sequence 35 in REPORTSC2. The REPORTSC2 program is called by the REPORTSC program that is called by REPORTS which is called by the MENUC program at sequence numbers 92, 96 and 100.

The display also shows that the CHKJOB program is CALLed by program CHKJOB2 which serves as a validity checking program for the RNMJOB command in the PILOT library.

If a program can be invoked by more than one program in the application, each possible caller will have the same indentation level. Program CHKJOBC on the previous sample display can be CALLed by the EDTTMPJOB, JOBMNT,and JOBMNT2 programs. The EDTTMPJOB program is CALLed by the CL program SCHTMPJOB which is used as a command processing program for the SCHTMPJOB command.

Copybook Usage

ABSTRACT initialization will record the references to source copybook members from the program source file members that contain them. Source copy usage is accessed (like field usage) through the WRKOBJUX command.

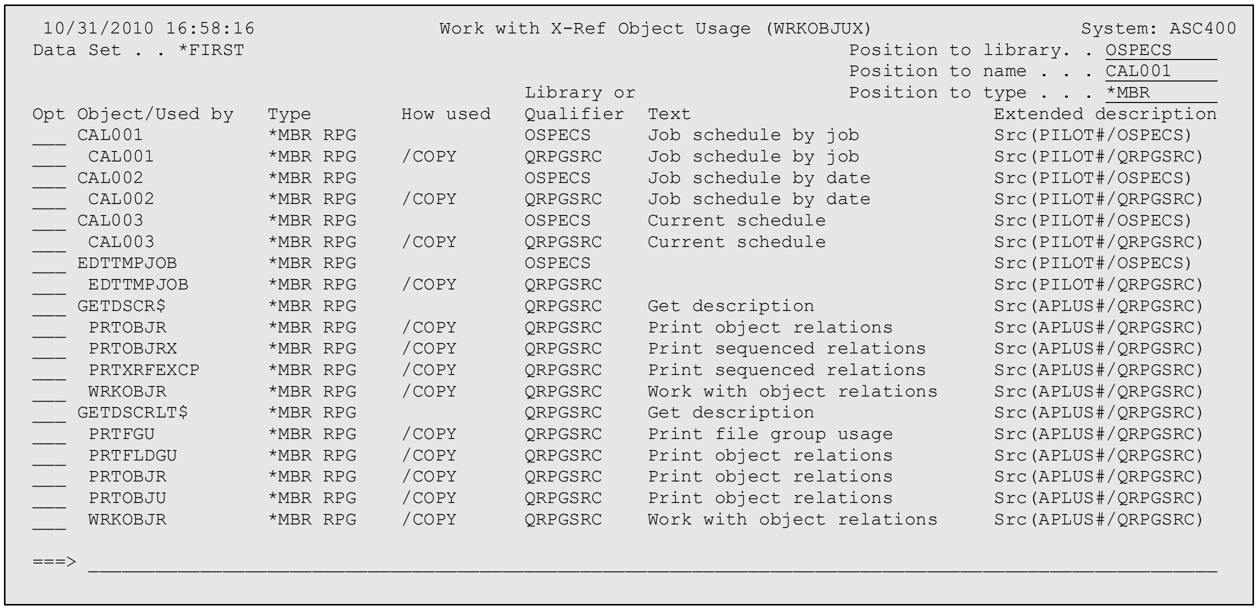
Specify an object naming request that includes the member name(s) you are interested in. Qualify the members with the file that includes them (or *ALL). Do this by using the LIB, OBJ, and OBJTYPE(*MBR) parameters of the WRKOBJUX command, or the subset window available through the action bar or function key 17.

The relationship display must show at least one level of indentation. Use a value of 1 or higher for the DSPLVL parameter or the object reference level value of the subset window. Alternatively, you can specify a Y value for the Show relations entry of the view menu of the action bar.

Members must be included in the relationship display. Specify DSPMBR(*YES) when using the command, or a Y value for the Members entry of the subset window.

The display below presents an example of the copybook usage recorded within the cross reference dictionary. The command used to acquire the display had the form:

```
WRKOBJUX LIB(*ALL) OBJ(*ALL) OBJTYPE(*MBR)
     DSPLVL(10) DSPMBR(*YES) DSPFILE(*NO)
     DSPFLD(*NO) DSPFMT(*NO) DSPPGM(*NO)
     DSPSUB(*NO) DSPOTR(*NO) DSPUSG(*ALL)
```



```
 10/31/2010 16:58:16                Work with X-Ref Object Usage (WRKOBJUX)                  System: ASC400
Data Set . . *FIRST                                                 Position to library. . OSPECS____
                                                                    Position to name . . . CAL001____
                                            Library or              Position to type . . . *MBR______
Opt Object/Used by   Type        How used   Qualifier  Text                        Extended description
___ CAL001           *MBR RPG               OSPECS     Job schedule by job         Src(PILOT#/OSPECS)
___  CAL001          *MBR RPG    /COPY      QRPGSRC    Job schedule by job         Src(PILOT#/QRPGSRC)
___ CAL002           *MBR RPG               OSPECS     Job schedule by date        Src(PILOT#/OSPECS)
___  CAL002          *MBR RPG    /COPY      QRPGSRC    Job schedule by date        Src(PILOT#/QRPGSRC)
___ CAL003           *MBR RPG               OSPECS     Current schedule            Src(PILOT#/OSPECS)
___  CAL003          *MBR RPG    /COPY      QRPGSRC    Current schedule            Src(PILOT#/QRPGSRC)
___ EDTTMPJOB        *MBR RPG               OSPECS                                 Src(PILOT#/OSPECS)
___  EDTTMPJOB       *MBR RPG    /COPY      QRPGSRC                                Src(PILOT#/QRPGSRC)
___ GETDSCR$         *MBR RPG               QRPGSRC    Get description             Src(APLUS#/QRPGSRC)
___  PRTOBJR         *MBR RPG    /COPY      QRPGSRC    Print object relations      Src(APLUS#/QRPGSRC)
___  PRTOBJRX        *MBR RPG    /COPY      QRPGSRC    Print sequenced relations   Src(APLUS#/QRPGSRC)
___  PRTXRFEXCP      *MBR RPG    /COPY      QRPGSRC    Print sequenced relations   Src(APLUS#/QRPGSRC)
___  WRKOBJR         *MBR RPG    /COPY      QRPGSRC    Work with object relations  Src(APLUS#/QRPGSRC)
___ GETDSCRLT$       *MBR RPG               QRPGSRC    Get description             Src(APLUS#/QRPGSRC)
___  PRTFGU          *MBR RPG    /COPY      QRPGSRC    Print file group usage      Src(APLUS#/QRPGSRC)
___  PRTFLDGU        *MBR RPG    /COPY      QRPGSRC    Print object relations      Src(APLUS#/QRPGSRC)
___  PRTOBJR         *MBR RPG    /COPY      QRPGSRC    Print object relations      Src(APLUS#/QRPGSRC)
___  PRTOBJU         *MBR RPG    /COPY      QRPGSRC    Print object relations      Src(APLUS#/QRPGSRC)
___  WRKOBJR         *MBR RPG    /COPY      QRPGSRC    Work with object relations  Src(APLUS#/QRPGSRC)

===> _____________________________________________________________________________________________________
```

Each of the parent objects on the display is used by the items that are indented and listed beneath it. The source file containing the members in the list is shown in the Library column and fully qualified in the Extended description column.

The display shows that the CAL001 member in the file PILOT#/OSPECS is copied by the CAL001 member in file PILOT#/QRPGSRC. The sequence number(s) where the source copy request is made can be obtained by changing the mode of the display with function key 11.

Multiple copy references appear on the display without further indentation. Each of the source members listed below the GETDSCR$ parent makes a source copy reference to it.

RPG Subroutine Usage

As with copybook information, ABSTRACT initialization will record the references to RPG subroutines from the program source file members that contain them. RPG subroutine usage is accessed (like field usage) through the WRKOBJUX command.

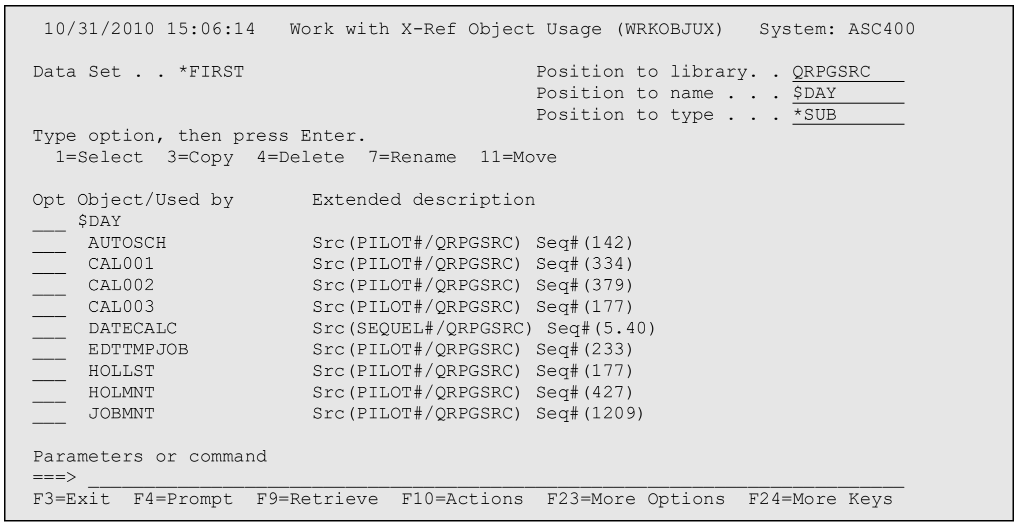
Specify an object naming request that includes the RPG subroutine name(s) you are interested in. Qualify the subroutine name with the file that includes them (or *ALL). Do this by using the LIB, OBJ, and OBJTYPE(*SUB) parameters of the WRKOBJUX command, or the subset window available through the action bar or function key 17.

The relationship display must show at least one level of indentation. Use a value of 1 or higher for the DSPLVL parameter or the object reference level value of the subset window. Alternatively, you can specify a Y value for the Show relations entry of the view menu of the action bar.

Members must be included in the relationship display. Specify DSPMBR(*YES) when using the command, or a Y value for the Members entry of the subset window.

The display below presents an example of the subroutine usage recorded within the cross reference dictionary. The command used to acquire the display had the form:

```
WRKOBJUX LIB(file-name) OBJ(subroutine) OBJTYPE(*SUB)
      DSPLVL(10) DSPMBR(*YES) DSPFILE(*NO)
      DSPFLD(*NO) DSPFMT(*NO) DSPPGM(*NO)
      DSPSUB(*NO) DSPOTR(*NO) DSPUSG(*ALL)
```



```
 10/31/2010 15:06:14   Work with X-Ref Object Usage (WRKOBJUX)   System: ASC400

Data Set . . *FIRST                        Position to library. . QRPGSRC
                                           Position to name . . . $DAY
                                           Position to type . . . *SUB
Type option, then press Enter.
  1=Select  3=Copy  4=Delete  7=Rename  11=Move

Opt Object/Used by       Extended description
___ $DAY
___  AUTOSCH             Src(PILOT#/QRPGSRC) Seq#(142)
___  CAL001              Src(PILOT#/QRPGSRC) Seq#(334)
___  CAL002              Src(PILOT#/QRPGSRC) Seq#(379)
___  CAL003              Src(PILOT#/QRPGSRC) Seq#(177)
___  DATECALC            Src(SEQUEL#/QRPGSRC) Seq#(5.40)
___  EDTTMPJOB           Src(PILOT#/QRPGSRC) Seq#(233)
___  HOLLST              Src(PILOT#/QRPGSRC) Seq#(177)
___  HOLMNT              Src(PILOT#/QRPGSRC) Seq#(427)
___  JOBMNT              Src(PILOT#/QRPGSRC) Seq#(1209)

Parameters or command
===> ________________________________________________________________________
F3=Exit  F4=Prompt  F9=Retrieve  F10=Actions  F23=More Options  F24=More Keys
```

An alternate form of the display has been chosen by selecting function key 11 to present the full extended description for the items in the list.

Each of the source members listed uses the $DAY subroutine at the sequence number indicated in the list. For instance, the DATECALC member in the SEQUEL#/QRPGSRC file uses the $DAY subroutine at sequence number 5.40 and the JOBMNT member of PILOT#/QRPGSRC references $DAY at statement 1209.

Object Reference and Usage Commands

This chapter will describe each of the ABSTRACT object relation commands and show how you can use them to get the information you need about your applications. Since each of the commands presents information using a list display, you should be familiar with the material in *Part 3, Object List Displays*.

There are several object relation commands. They can be divided into two groups: object reference and object usage. Object referencing and object usage are logical opposites of one another. For example, if a file is referenced by a program and listed underneath it on one of the reference displays, the program will be listed underneath the file on the usage display - indicating that the file is used by the program.

The ABSTRACT object reference commands present information showing how objects are referenced by a parent object. They are:

Work With Object References (WRKOBJR)
Work With Cross-Reference Object References (WRKOBJRX)
Work With Sequenced Object References (WRKOBJRS)
Work With File Group References (WRKFGR)

The ABSTRACT object usage commands provide information about the objects that use a given object. Object usage commands are:

Work With Cross-Reference Object Usage (WRKOBJUX)
Work With Object Usage (WRKOBJU)
Work With File Group Usage (WRKFGU)
Work With Field Group Usage (WRKFLDGU)

Each command is similar; once you learn how to use the WRKOBJR command, for example, you will be able to use any of the other object relation commands.

The following sections in this chapter present the object relation menus and the commands that they run.

Object Reference Menu (APOBJR)

The object reference menu (APOBJR) can be accessed from the ABSTRACT main menu through option 2, or by typing:

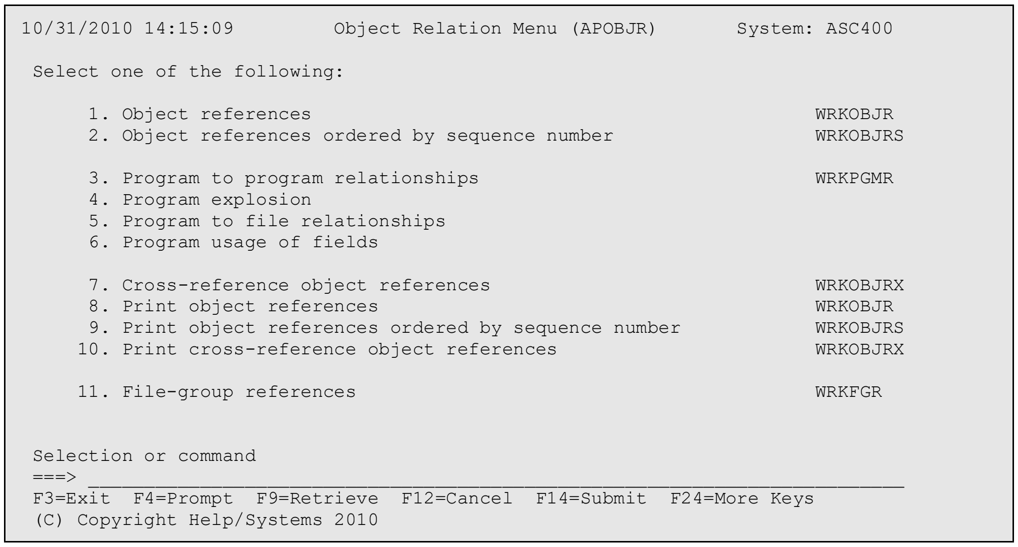
```
GO ABSTRACT/APOBJR
```

on a command entry line and pressing the Enter key.

The menu allows you to use the object reference commands and functions of ABSTRACT and to display and print the output. Each item on the menu presents object reference information - that is, it tells about the objects that make up your application, showing the references made by each object you have selected.

The menu also allows you to have full access to your authorized command set through the command entry line at the bottom of the display. Like the other ABSTRACT menus, it can access the option file and default job queue setting through function keys.

The menu it will look like the example below.



```
10/31/2010 14:15:09         Object Relation Menu (APOBJR)       System: ASC400

 Select one of the following:

      1. Object references                                             WRKOBJR
      2. Object references ordered by sequence number                  WRKOBJRS

      3. Program to program relationships                              WRKPGMR
      4. Program explosion
      5. Program to file relationships
      6. Program usage of fields

      7. Cross-reference object references                             WRKOBJRX
      8. Print object references                                       WRKOBJR
      9. Print object references ordered by sequence number            WRKOBJRS
     10. Print cross-reference object references                       WRKOBJRX

     11. File-group references                                         WRKFGR


 Selection or command
 ===> _________________________________________________________________________
 F3=Exit  F4=Prompt  F9=Retrieve  F12=Cancel  F14=Submit  F24=More Keys
 (C) Copyright Help/Systems 2010
```

Each of the menu options will prompt, and then run, an ABSTRACT command. Depending on the menu item and the value of the Run in Batch flag, the command may be submitted to the batch job queue or begin running immediately when you press the Enter key. Given that some print requests can run for an extended period of time, this may not be an efficient use of your display device or interactive priority.

Your request will always be submitted to the batch subsystem if you type the menu option number and press F14 or F15. Refer to the section describing the ABSTRACT menu driver files in the Introduction for information about changing the batch/interactive environment for each menu option.

Select option 1 to prompt the Work With Object References (WRKOBJR) command. It allows you to work with objects in your application and view the cross reference information that is currently loaded for them. Refer to page 4-6 for complete information about this function.

Use option 2 to begin the Work With Sequenced Object References (WRKOBJRS) command. This command is similar to WRKOBJR in that it also provides object reference functions, but differs in that referenced items appear in order according to the sequence number that generated the reference (rather than alphabetically). Refer to page 4-14 for a description of the WRKOBJRS command.

Menu options 3 through 6 also run the Work With Object References program although they use a modified version of the WRKOBJR command. The Work With Program Relations (WRKPGMR) command renames some WRKOBJR parameters and pre–specifies others to make the prompt seem more applicable for program object relations. Refer to the description of the WRKOBJR command beginning on page 4-6 for complete details about these options.

Option 7 invokes the Work With X-ref Object References (WRKOBJRX) command. Like the WRKOBJR command, it displays object reference information. Instead of drawing the list of parent objects by direct library access, it gets them from the cross reference dictionary.

Menu options 8, 9, and 10 run the object reference commands and direct their output to the printer.

Option 11 runs the Work With File Group References (WRKFGR) command. It displays information about the files in a file group — the complete set of logicals and physicals related to the file you specify. The WRKFGR command is described beginning on page 4-29.

Function Keys

Like the other ABSTRACT menus, the object relations menus (APOBJR, APOBJU) provide a consistent set of function keys:

F3	Exit this ABSTRACT function
F4	Prompt the option or command that is on the command line
F9	Retrieve ‘backwards’ through previously executed commands
F12	Exit the current display and return to the previous function
F14	Submit the option or command on the command line to batch using the default job description defined for this user
F15	Prompt the Submit Job (SBMJOB) command before submitting the option or the command on the command line
F16	Work with options in the user option file
F18	Change the default job description and user option file defined for the current user
F19	Retrieve ‘forwards’ through previously executed commands
F24	Display more function keys

Object Usage (APOBJU) Menu

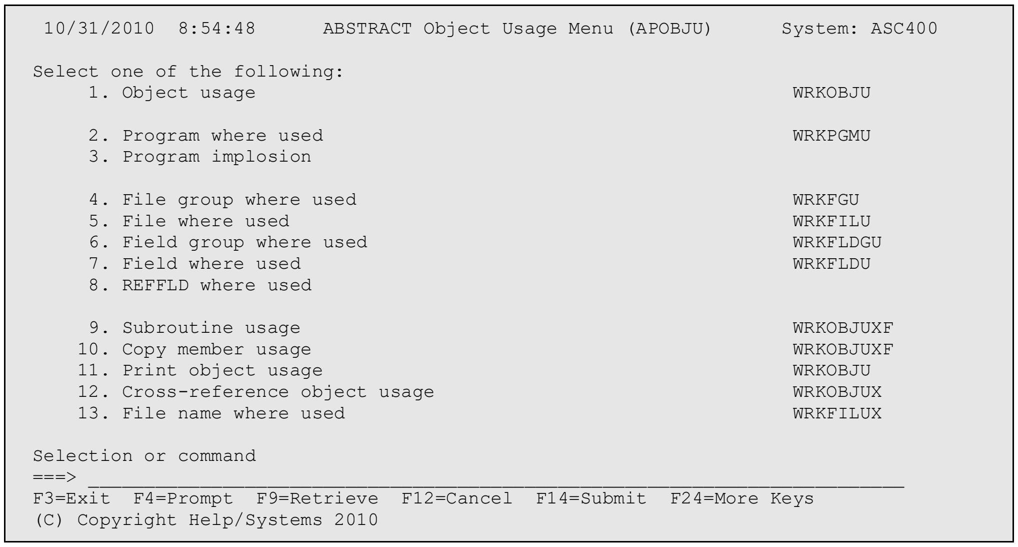
This menu accesses the ABSTRACT object usage functions. Each item on the menu displays object usage information — that is, it tells where and how an object within your application is used.

Acquire the menu by selecting option 3 from the primary (ABSTRACT) menu or by typing

```
GO ABSTRACT/APOBJU
```

on a command entry line and pressing the Enter key.

As with other ABSTRACT menus, the APOBJU menu provides full access to your authorized command set through the command entry line at the bottom of the display.


```
 10/31/2010  8:54:48      ABSTRACT Object Usage Menu (APOBJU)      System: ASC400

Select one of the following:
     1. Object usage                                                WRKOBJU

     2. Program where used                                          WRKPGMU
     3. Program implosion

     4. File group where used                                       WRKFGU
     5. File where used                                             WRKFILU
     6. Field group where used                                      WRKFLDGU
     7. Field where used                                            WRKFLDU
     8. REFFLD where used

     9. Subroutine usage                                            WRKOBJUXF
    10. Copy member usage                                           WRKOBJUXF
    11. Print object usage                                          WRKOBJU
    12. Cross-reference object usage                                WRKOBJUX
    13. File name where used                                        WRKFILUX

Selection or command
===> _________________________________________________________________________
F3=Exit  F4=Prompt  F9=Retrieve  F12=Cancel  F14=Submit  F24=More Keys
 (C) Copyright Help/Systems 2010
```

Each of the menu options will prompt and then run the ABSTRACT command shown at the right of the menu.

Your request will always be submitted to the batch subsystem if you type the menu option number and press F14 or F15. Refer to the section describing the ABSTRACT menu driver files in the Introduction for information about changing the batch/interactive environment for each menu option.

Select option 1 to invoke the primary object usage command, Work With Object Usage (WRKOBJU). It allows you to work with objects in your application and view the cross reference information that is currently loaded for them. Refer to page 4-6 for complete information about this function.

Menu options 2, 3 and 5 also run the Work With Object Usage program although they use a modified version of the WRKOBJU command. The Work With Program Usage (WRKPGMU) and Work With File Usage (WRKFILU) commands rename some WRKOBJR parameters and pre-specify others to make the prompt seem more applicable for the relationships you want to see. Refer to the description of the WRKOBJU command beginning on page 4-44 for complete details about these options.

Use menu options 4 to access the file group function. Beginning with the file you specify, ABSTRACT will locate the related logical or physical files and show usage relationships for all the files in the group. The Work With File Group (WRKFGU) command is described beginning on page 4-52.

Option 6 will invoke the field group function detailed on page 4-59 and following. Like the file group, the field group may include several files - each file that uses the field positions occupied by the field and file that you specify on the command.

Menu options 7 through 10 run the Work With X-ref Object Usage (WRKOBJUX) program to provide usage information. Since fields, subroutines, and copy members aren't true iSeries "objects", you can't use the object-based display of the WRKOBJU list. Instead, you need to access the cross reference information directly through the WRK...X commands.

Options 7 through 10 use a modified version of the WRKOBJUX command. The actual commands used (WRKFLDU, WRKOBJUXF) rename some WRKOBJUX parameters and pre-specify others to make the prompts seem more applicable for their indicated purpose. See page 4-36 for a description of the WRKOBJUX and the various customized commands that go with it.

Option 11 on the menu lets you to print usage information without accessing one of the list displays.

Menu option 12 prompts the Work With X-ref Object Usage command so that you can search the cross reference files directly, without first starting with an iSeries object list. This can be especially useful when you want to find out about object no longer on the system, or to do a quick lookup of some cross reference information.

Use option 13 when you want information about a file that no longer (or perhaps never did) exists on your system. It is a customized version of the WRKOBJUX command that searches all cross reference records and pre-specifies the object type.

Turn to page 4-3 for a description of the active function keys and their use.

Work With Object References (WRKOBJR) Command

The Work With Object References (WRKOBJR) command shows reference information for objects that exist on your system. It searches for objects currently on your system that match the object naming criteria you supply. You get the cross reference information obtained during the initialization (see chapter 1) that shows which objects are used by the ones you have selected. Output can be directed to the display, printer, or a database file.

The WRKOBJR command is similar to two other sets of object reference commands:

Work With Sequenced Object References (WRKOBJRS) presents cross reference information in source sequence order, rather than the alphabetic order (by name) used by WRKOBJR.

Work With X-Ref References (WRKOBJRX) presents the list in alphabetic sequence, but uses the information in the cross reference files (not the objects in your libraries) as the parent objects in the list.

The WRKOBJR command can show program-program relationships, program explosions, program-file and program-field relationships, and program usage of commands. Refer to *Chapter 2, Using ABSTRACT List Displays* for a description of the features and functions of the lists created by WRKOBJR.

Only the libraries to which you have *USE authority will be searched for the objects. Objects that you have no authority for will be excluded from the list.

If you route the information to the display, ABSTRACT options can be used to process items in the list. You must have the appropriate authority to the objects referenced by the option and *USE authority for each command that is run.

The WRKOBJR report is similar to the display. Objects on the display and report are presented using an indented list. The list is ordered alphabetically by object type. The format of the output file is similar, except that a level number field reports the indenting level of the objects in the file.

The syntax for the WRKOBJR command is given below. The parameters and their meanings follow.

```
                   .-*CURLIB/------.       .-*ALL----------.
>>--WRKOBJR----LIB-+---------------+---OBJ-+---------------+----------------------------->
                   +-*PRV/---------+       +-object_name---+
                   +-*LIBL/--------+       '-*generic_name-'
                   +-*USRLIBL/-----+
                   +-*ALLUSR/------+
                   +-*ALL/---------+
                   '-library_name/-'

           .-*ALL--------.          .-*ALL-------------.
>--OBJTYPE-+-------------+---OBJATR-+------------------+--------------------------------->
           +-object_type-+          +-object_attribute-+
           +-*DBF--------+          '-generic_name-----'
           '-*SRCF-------'
                                                                                Required
==========================================================================================
                                                                                Optional

          .-*PRV----.           .-*PRV-.          .-*PRV-.          .-*PRV-.
>--DSPLVL-+---------+---DSPFILE-+------+---DSPFMT-+------+---DSPFLD-+------+------------->
          '-numeric-'           +-*YES-+          +-*YES-+          +-*YES-+
                                '-*NO--'          '-*NO--'          '-*NO--'

          .-*PRV-.          .-*PRV-.          .-*PRV-.          .-*PRV-.
>--DSPMBR-+------+---DSPSUB-+------+---DSPPGM-+------+---DSPOTR-+------+----------------->
          +-*YES-+          +-*YES-+          +-*YES-+          +-*YES-+
          '-*NO--'          '-*NO--'          '-*NO--'          '-*NO--'

          .-*PRV-.          .----0----.          .-*ALL---.          .-*PRV---.
>--DSPUNQ-+------+---EXPLVL-+---------+---DSPUSG-+--------+---DTASET-+--------+---------->
          +-*YES-+          '-numeric-'          +-text---+          +-*FIRST-+
          '-*NO--'                               +-*NOCMD-+          '-name---'
                                                 +-*CMD---+
                                                 +-*IO----+
                                                 +-*INP---+
                                                 +-*OUT---+
                                                 '-*UPD---'

          .-*--------.           .-*CURLIB-.                     .-*FIRST-.
>--OUTPUT-+----------+---OUTFILE-+---------+---FILENAME---OUTMBR-+--------+-------------->
          +-*PRINT---+           +-*LIBL---+                     '-member-'
          '-*OUTFILE-'           '-library-'

          .-*REPLACE-.
>--MBROPT-+----------+------------------------------------------------------------------><
          '-*ADD-----'
```

LIB Parameter

Specifies the libraries that are searched for the objects indicated by the other parameters on the command. If no library is indicated, *CURLIB is assumed. The possible library values are:

***CURLIB:** The current library is searched. If no library is specified as the current library for the job, the QGPL library is used.

***PRV:** The library used during the previous session, regardless of which of the ABSTRACT WRKxxx commands was used.

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***USRLIBL:** Only the libraries in the user portion of the library list are searched.

***ALLUSR:** All non-system libraries, including all user-defined libraries and the QGPL library, are searched. Libraries with names starting with the letter Q, other than QGPL, are not included.

***ALL:** All libraries in the system, including QSYS, are searched.

library-name: Only the specific library is searched for the objects.

OBJ Parameter

Specifies the object name. You can use this parameter to work with all the objects or a subset of objects in the indicated library.

<u>***ALL:**</u> All objects in the specified library(s) are selected.

object-name: Only objects with this specific name will be included in the list.

***generic*-name:** Specify a partial object name qualified by an asterisk (*) to select several objects meeting the criteria. The following generic forms are allowed: ABC*, *ABC, *ABC*, A*B, abc*, **ALL. Refer to Appendix G, General Functions, in Volume 1 of the <u>CL Reference</u> manual for more information about generic names.

OBJTYPE Parameter

Specifies the object type. You can use this parameter to work with all object types or a subset of objects. An expanded description of the OBJTYPE parameter and a list of valid OS/400 object types can be found in Appendix A, Expanded Parameter Descriptions, of Volume 1 of the <u>CL Reference</u> manual.

<u>***ALL:**</u> All objects regardless of their type are included.

object-type: Specify any valid system object type to display a list of all objects of that particular type.

***DBF:** All data base files will be included. These are *FILE objects with an attribute beginning with PF, LF, or DDMF.

OBJATR Parameter

Specifies the object attribute. You can use this parameter to work with all object attributes or only specific ones. An expanded description of the OBJATR parameter and a list of valid OS/400 object attributes can be found in Appendix A, Expanded Parameter Descriptions, of Volume 1 of the <u>CL Reference</u> manual.

<u>***ALL:**</u> All objects in the specified library(s) are selected regardless of their attribute definition.

***BLANK:** Only objects with no attribute will be included.

object-attribute: Only objects with this specific attribute will be included in the list. PF-DTA can be used to select only non-source physical files. PF-SRC can be used to select only source physical files.

***generic*-name:** Specify a partial object attribute qualified by an asterisk (*) to select several objects meeting the criteria. The following generic forms are allowed: ABC*, *ABC, *ABC*, A*B, abc*, **ALL. For example, *RPG* will select RPG, RPG36, RPG38, and SQLRPG objects. Refer to Appendix G, General Functions, in Volume 1 of the CL Reference manual for more information about generic names.

DSPLVL Parameter

Specifies the maximum depth of the relationship display. This value controls the number of indented levels that appear under the leftmost (parent) objects.

***PRV:** The level used for this parameter during the previous session.

numeric: Specify a value between zero (0) and ten (10) that indicates the maximum level of indentation to appear on the object list display.

DSPFILE Parameter

Specify whether cross reference information indicating references to file (*FILE) objects should appear on the display.

***PRV:** Use the value for this parameter from the previous session.

***YES:** All files referenced by objects in the list will be included on the display.

***NO:** None of the files referenced by an object in the list will appear.

DSPFMT Parameter

Specify whether cross reference information indicating references to file formats (*FMT) should appear on the display.

***PRV:** Use the value for this parameter from the previous session.

***YES:** All file formats referenced by objects in the list will be included on the display.

***NO:** None of the files referenced by an object in the list will appear.

DSPFLD Parameter

Specify whether cross reference information indicating references to file fields (*FLD) should appear on the display.

***PRV:** Use the value for this parameter from the previous session.

***YES:** All fields referenced by an object in the list will be included on the display.

***NO:** None of the files referenced by an object in the list will appear.

DSPMBR Parameter

Specify whether cross reference information indicating references to file members (*MBR) should appear on the display.

***PRV:** Use the value for this parameter from the previous session.

***YES:** All members referenced by an object in the list will be included on the display.

***NO:** None of the file members referenced by an object in the list will appear.

DSPSUB Parameter

Specify whether cross reference information indicating program references to subroutines (*SUB) should appear on the display.

***PRV:** Use the value for this parameter from the previous session.

***YES:** All subroutines referenced by a program in the list will be included on the display.

***NO:** No program references to subroutines will appear.

DSPPGM Parameter

Specify whether cross reference information indicating references to programs (*PGM) should appear on the display.

***PRV:** Use the value for this parameter from the previous session.

***YES:** All programs referenced by an object in the list will be included on the display.

***NO:** None of the programs referenced by an object in the list will appear.

DSPOTR Parameter

Specify whether cross reference information indicating references to miscellaneous object types (other than those above) should appear on the display. This will includes references to commands (*CMD), data areas (*DTAARA), job descriptions (*JOBD), etc.

***PRV:** Use the value for this parameter from the previous session.

***YES:** All miscellaneous object types referenced by an object in the list will be included on the display.

***NO:** None of the miscellaneous object types referenced by an object in the list will appear.

DSPUNQ Parameter

Specify whether duplicate cross reference records for an object should be suppressed.

***PRV:** Use the value for this parameter from the previous session.

***YES:** If a given qualified object is referenced more than once, only one line will appear on the display or report. All references will be collapsed into a single line. The USAGE column will indicate the first type of use. The SEQ() values in the EXTENDED DESCRIPTION column will list the locations where the object was referenced.

***NO:** Each reference will appear on a separate line. This is especially useful when a program works with an object in several ways, such as opening a file twice.

EXPLVL Parameter

Specifies the number of recursive iterations for the list explosion. This value determines the depth of the explosion tree before additional relationships are ignored.

<u>0:</u> No explosion recursion occurs for objects on the display. All references for the parent object are given, but references for the underlying objects are not.

numeric: Specify a value between zero (0) and ninety-nine (99) that indicates the maximum number of reference generations that should be shown for each parent object on the display.

DSPUSG Parameter

Lets you filter the references in the list. Only references with the specific type of usage you select will appear beneath a parent object.

<u>*ALL:</u> No usage selection will be used to filter references from the list.

text: Only references with usage values matching this text will be included. (c.f. CALL, Input, ...)

***CMD:** Only objects referenced through CL commands will be included in the list.

***NONCMD:** Command references in CL programs will be suppressed from the list. Only non-command (Unknown and input/output types) references will be included.

***IO:** Files with input, output, or update (including delete) usage will be included in the list. Other objects will be omitted.

***INP:** Files with input usage will be included in the list. Other objects will be omitted.

***OUT:** Files with output usage will be included in the list. Other objects will be omitted.

***UPD:** Files with update (or delete) usage will be included in the list. Other objects will be omitted.

***UPDOUT:** Files with output or update (including delete) usage will be included in the list. Other objects will be omitted.

DTASET Parameter

Specify the cross reference data set to be used in acquiring the information.

<u>*PRV:</u> Use the value for this parameter from the previous session. If no previous session has occurred, use *FIRST.

***FIRST:** information from the first (default) data set should be accessed.

Name: Specify any existing data set. Use the command prompter to find the names available to you.

OUTPUT Parameter

Indicates where the results of the command should be routed. Specify:

***:** Information will be displayed at the workstation.

***PRINT:** Information will be spooled to the ABSTRACTPRT printer file for printing.

***OUTFILE:** Information will be sent to a database file.

OUTFILE Parameter

Specifies the name of the database file to receive output from the command. The file name must be specified and you must have proper data rights to add records to it. You can specify one of two special values for the library name. If the outfile does not exist prior to execution, the command will create it unless *LIBL is specified for the file's library.

***CURLIB:** The job's current library will be used to locate the file. If it is not found, the output file will be created in the current library.

***LIBL:** The job's library list will be searched for the indicated file.

A "pattern" outfile in the ABSTRACT library named APOUTFIL supplies information regarding the size, allocation parameters, and maximum number of members allowed. You can alter the characteristics of the created output files by using the Change Physical File (CHGPF) command on the pattern file.

OUTMBR Parameter

Specifies the name of the database file member(s) to which the command output is directed. Specify:

***FIRST:** output is directed to the first member in the file. If this value is specified and the member does not exist, a member will be created with the same name as the file specified in the OUTFILE parameter.

member–name: output is directed to the named member in the file. If the member does not exist, it will be added to the file.

MBROPT Parameter

If the output file exists before the command is run, this keyword indicates whether records in the file will be cleared before the new information is placed into the file.

***REPLACE:** existing records will be cleared from the output member prior to inserting new information.

***ADD:** records currently in the member are retained and new records will be added to them.

Examples

```
WRKOBJR PILOT
```

This command will present a list showing all the objects in the PILOT library that you have *USE authority to. Reference information for the objects will be displayed according to the values in use when you last ended ABSTRACT.

```
WRKOBJR *USRLIBL OBJTYPE(*PGM) DSPPGM(*YES) EXPLVL(2)
```

Only the programs on the user portion of your library list will appear as parent objects on the object list display. References to the programs they use will appear and these programs will also be exploded to the second generation. Files, fields, and other objects used by items in the list may or may not appear, depending on the values in use when you last ended ABSTRACT.

Work With Sequenced Object References (WRKOBJRS)

The Work With Sequenced Object References (WRKOBJRS) command shows object references for objects that exist on your system. They search for objects currently on your system that match the object naming criteria you supply. Once found, they access the cross reference information obtained during the initialization (see chapter 1) to display the objects that are used by those you have selected.

The referenced objects are presented in the order they are used by the parent, according to the sequence number that generated the reference.

The WRKOBJRS command is similar to two other sets of object reference commands:

> Work With Object References (WRKOBJR) presents cross reference information on the object list display in alphabetic order (by name), rather than the source sequence order used by WRKOBJRS.
>
> Work With X-Ref References (WRKOBJRX) also presents the list in alphabetic sequence, but uses the information in the cross reference files (not the objects in your libraries) as the starting point for the parent objects in the list.

The WRKOBJRS command can show program-program relationships, program-file and program-field relationships, and program usage of commands. Refer to *Chapter 2, Using ABSTRACT List Displays* for a description of the features and functions of the list display used by the WRKOBJRS command.

Only the libraries to which you have *USE authority will be searched for the objects you name on the WRKOBJRS command. Objects that you have no authority to will be excluded from the list that is presented.

If you route the information to the display, ABSTRACT options can be used to process items in the list. You must have the appropriate authority to the objects referenced by the option and *USE authority for each command that is run.

The WRKOBJRS report is similar to the display. Objects on the display and report are presented using an indented list. The list is ordered alphabetically by object type. The format of the output file is similar, except that a level number field reports the indenting level of the objects in the file.

The syntax for the WRKOBJRS command is given below. The parameters and their meanings follow.

```
                    .-*CURLIB/------.       .-*ALL----------.
>>--WRKOBJRS----LIB-+---------------+---OBJ-+---------------+--------------------------->
                    +-*PRV/---------+       +-object_name---+
                    +-*LIBL/--------+       '-*generic_name-'
                    +-*USRLIBL/-----+
                    +-*ALLUSR/------+
                    +-*ALL/---------+
                    '-library_name/-'

           .-*ALL--------.          .-*ALL-------------.
>--OBJTYPE-+-------------+---OBJATR-+------------------+-------------------------------->
           +-object_type-+          +-object_attribute-+
           +-*DBF--------+          '-generic_name-----'
           '-*SRCF-------'
                                                                                Required
=========================================================================================
                                                                                Optional

          .-*PRV----.           .-*PRV-.          .-*PRV-.          .-*PRV-.
>--DSPLVL-+---------+---DSPFILE-+------+---DSPFMT-+------+---DSPFLD-+------+------------>
          '-numeric-'           +-*YES-+          +-*YES-+          +-*YES-+
                                '-*NO--'          '-*NO--'          '-*NO--'

          .-*PRV-.          .-*PRV-.          .-*PRV-.          .-*PRV-.
>--DSPMBR-+------+---DSPSUB-+------+---DSPPGM-+------+---DSPOTR-+------+---------------->
          +-*YES-+          +-*YES-+          +-*YES-+          +-*YES-+
          '-*NO--'          '-*NO--'          '-*NO--'          '-*NO--'

          .-*PRV-.          .----0----.          .-*ALL---.          .-*PRV---.
>--DSPUNQ-+------+---EXPLVL-+---------+---DSPUSG-+--------+---DTASET-+--------+--------->
          +-*YES-+          '-numeric-'          +-text---+          +-*FIRST-+
          '-*NO--'                               +-*NOCMD-+          '-name---'
                                                 +-*CMD---+
                                                 +-*IO----+
                                                 +-*INP---+
                                                 +-*OUT---+
                                                 '-*UPD---'

          .-*--------.           .-*CURLIB-.                     .-*FIRST-.
>--OUTPUT-+----------+---OUTFILE-+---------+---FILENAME---OUTMBR-+--------+------------->
          +-*PRINT---+           +-*LIBL---+                     '-member-'
          '-*OUTFILE-'           '-library-'

          .-*REPLACE-.
>--MBROPT-+----------+-----------------------------------------------------------------><
          '-*ADD-----'
```

LIB Parameter

Specifies the libraries that are searched for the objects indicated by the other parameters on the command. If no library is indicated, *CURLIB is assumed. The possible library values are:

<u>*CURLIB:</u> The current library is searched. If no library is specified as the current library for the job, the QGPL library is used.

***PRV:** The library used during the previous session, regardless of which of the ABSTRACT WRKxxx commands was used.

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***USRLIBL:** Only the libraries in the user portion of the library list are searched.

***ALLUSR:** All non-system libraries, including all user-defined libraries and the QGPL library, are searched. Libraries with names starting with the letter Q, other than QGPL, are not included.

***ALL:** All libraries in the system, including QSYS, are searched.

library-name: Only the specific library is searched for the objects.

OBJ Parameter

Specifies the object name. You can use this parameter to work with all the objects or a subset of objects in the indicated library.

***ALL:** All objects in the specified library(s) are selected.

object-name: Only objects with this specific name will be included in the list.

***generic*-name:** Specify a partial object name qualified by an asterisk (*) to select several objects meeting the criteria. The following generic forms are allowed: ABC*, *ABC, *ABC*, A*B, abc*, **ALL. Refer to Appendix G, General Functions, in Volume 1 of the CL Reference manual for more information about generic names.

OBJTYPE Parameter

Specifies the object type. You can use this parameter to work with all object types or a subset of objects. An expanded description of the OBJTYPE parameter and a list of valid OS/400 object types can be found in Appendix A, Expanded Parameter Descriptions, of Volume 1 of the CL Reference manual.

***ALL:** All objects regardless of their type are included.

object-type: Specify any valid system object type to display a list of all objects of that particular type.

***DBF:** All data base files will be included. These are *FILE objects with an attribute beginning with PF, LF, or DDMF.

OBJATR Parameter

Specifies the object attribute. You can use this parameter to work with all object attributes or only specific ones. An expanded description of the OBJATR parameter and a list of valid OS/400 object attributes can be found in Appendix A, Expanded Parameter Descriptions, of Volume 1 of the CL Reference manual.

***ALL:** All objects in the specified library(s) are selected regardless of their attribute definition.

***BLANK:** Only objects with no attribute will be included.

object-attribute: Only objects with this specific attribute will be included in the list. PF-DTA can be used to select only non-source physical files. PF-SRC can be used to select only source physical files.

***generic*-name:** Specify a partial object attribute qualified by an asterisk (*) to select several objects meeting the criteria. The following generic forms are allowed: ABC*, *ABC, *ABC*, A*B, abc*, **ALL. For example, *RPG* will select RPG, RPG36, RPG38, and SQLRPG objects. Refer to Appendix G, General Functions, in Volume 1 of the CL Reference manual for more information about generic names.

DSPLVL Parameter

Specifies the maximum depth of the relationship display. This value controls the number of indented levels that appear under the leftmost (parent) objects.

***PRV:** The level used for this parameter during the previous session.

numeric: Specify a value between zero (0) and ten (10) that indicates the maximum level of indentation to appear on the object list display.

DSPFILE Parameter

Specify whether cross reference information indicating references to file (*FILE) objects should appear on the display.

***PRV:** Use the value for this parameter from the previous session.

***YES:** All files referenced by objects in the list will be included on the display.

***NO:** None of the files referenced by an object in the list will appear.

DSPFMT Parameter

Specify whether cross reference information indicating references to file formats (*FMT) should appear on the display.

***PRV:** Use the value for this parameter from the previous session.

***YES:** All file formats referenced by objects in the list will be included on the display.

***NO:** None of the files referenced by an object in the list will appear.

DSPFLD Parameter

Specify whether cross reference information indicating references to file fields (*FLD) should appear on the display.

***PRV:** Use the value for this parameter from the previous session.

***YES:** All fields referenced by an object in the list will be included on the display.

***NO:** None of the files referenced by an object in the list will appear.

DSPMBR Parameter

Specify whether cross reference information indicating references to file members (*MBR) should appear on the display.

PRV: Use the value for this parameter from the previous session.

***YES:** All members referenced by an object in the list will be included on the display.

***NO:** None of the file members referenced by an object in the list will appear.

DSPSUB Parameter

Specify whether cross reference information indicating program references to subroutines (*SUB) should appear on the display.

PRV: Use the value for this parameter from the previous session.

***YES:** All subroutines referenced by a program in the list will be included on the display.

***NO:** No program references to subroutines will appear.

DSPPGM Parameter

Specify whether cross reference information indicating references to programs (*PGM) should appear on the display.

PRV: Use the value for this parameter from the previous session.

***YES:** All programs referenced by an object in the list will be included on the display.

***NO:** None of the programs referenced by an object in the list will appear.

DSPOTR Parameter

Specify whether cross reference information indicating references to miscellaneous object types (other than those above) should appear on the display. This will includes references to commands (*CMD), data areas (*DTAARA), job descriptions (*JOBD), etc.

PRV: Use the value for this parameter from the previous session.

***YES:** All miscellaneous object types referenced by an object in the list will be included on the display.

***NO:** None of the miscellaneous object types referenced by an object in the list will appear.

DSPUNQ Parameter

Specify whether duplicate cross reference records for an object should be suppressed.

PRV: Use the value for this parameter from the previous session.

***YES:** If a given qualified object is referenced more than once, only one line will appear on the display or report. All references will be collapsed into a single line. The USAGE column will indicate the first type of use. The SEQ() values in the EXTENDED DESCRIPTION column will list the locations where the object was referenced.

***NO:** Each reference will appear on a separate line. This is especially useful when a program works with an object in several ways, such as opening a file twice.

EXPLVL Parameter

Specifies the number of recursive iterations for the list explosion. This value determines the depth of the explosion tree before additional relationships are ignored.

<u>0:</u> No explosion recursion occurs for objects on the display. All references for the parent object are given, but references for the underlying objects are not.

numeric: Specify a value between zero (0) and ninety-nine (99) that indicates the maximum number of reference generations that should be shown for each parent object on the display.

DSPUSG Parameter

Lets you filter the references in the list. Only references with the specific type of usage you select will appear beneath a parent object.

<u>*ALL:</u> No usage selection will be used to filter references from the list.

text: Only references with usage values matching this text will be included. (c.f. CALL, Input, ...)

***CMD:** Only objects referenced through CL commands will be included in the list.

***NONCMD:** Command references in CL programs will be suppressed from the list. Only non-command (Unknown and input/output types) references will be included.

***IO:** Files with input, output, or update (including delete) usage will be included in the list. Other objects will be omitted.

***INP:** Files with input usage will be included in the list. Other objects will be omitted.

***OUT:** Files with output usage will be included in the list. Other objects will be omitted.

***UPD:** Files with update (or delete) usage will be included in the list. Other objects will be omitted.

***UPDOUT:** Files with output or update (including delete) usage will be included in the list. Other objects will be omitted.

DTASET Parameter

Specify the cross reference data set to be used in acquiring the information.

<u>*PRV:</u> Use the value for this parameter from the previous session.

***FIRST:** information from the first (default) data set should be accessed.

Name: Specify any existing data set. Use the command prompter to find the names available to you.

OUTPUT Parameter

Indicates where the results of the command should be routed. Specify:

***:** Information will be displayed at the workstation.

***PRINT:** Information will be spooled to the ABSTRACTPRT printer file for printing.

***OUTFILE:** Information will be sent to a database file.

OUTFILE Parameter

Specifies the name of the database file to receive output from the command. The file name must be specified and you must have proper data rights to add records to it. You can specify one of two special values for the library name. If the outfile does not exist prior to execution, the command will create it unless *LIBL is specified for the file's library.

***CURLIB:** The job's current library will be used to locate the file. If it is not found, the output file will be created in the current library.

***LIBL:** The job's library list will be searched for the indicated file.

A "pattern" outfile in the ABSTRACT library named APOUTFIL supplies information regarding the size, allocation parameters, and maximum number of members allowed. You can alter the characteristics of the created output files by using the Change Physical File (CHGPF) command on the pattern file.

OUTMBR Parameter

Specifies the name of the database file member(s) to which the command output is directed. Specify:

***FIRST:** output is directed to the first member in the file. If this value is specified and the member does not exist, a member will be created with the same name as the file specified in the OUTFILE parameter.

member–name: output is directed to the named member in the file. If the member does not exist, it will be added to the file.

MBROPT Parameter

If the output file exists before the command is run, this keyword indicates whether records in the file will be cleared before the new information is placed into the file.

***REPLACE:** existing records will be cleared from the output member prior to inserting new information.

***ADD:** records currently in the member are retained and new records will be added to them.

Examples

```
WRKOBJRS PILOT
```

This command will present a list showing all the objects in the PILOT library that you have *USE authority to. Reference information for the objects will be displayed in sequenced order according to the values in use when you last ended ABSTRACT.

```
WRKOBJRS PILOT OBJTYPE(*PGM) DSPPGM(*YES) DSPFILE(*YES)
```

Only the programs in the PILOT library will appear as parent objects on the object list display. References to the programs and files they use will appear in order of reference. Fields and other objects used by the programs in the list may or may not appear, depending on the values in use when you last ended ABSTRACT.

Work With X-ref Object References (WRKOBJRX)

The Work With X-ref Object References (WRKOBJRX) command shows object reference information for objects present in the cross reference dictionary.

Unlike the WRKOBJR and WRKOBJRS commands, it searches the cross reference data set for objects that match the object naming criteria you supply. Once found, it uses the cross reference information obtained during the initialization (see chapter 1) to display objects that are used by the ones you have selected. The referenced objects listed for each parent are presented in alphabetic order.

The WRKOBJRX command can be especially useful when you want to find out about the contents of the cross reference dictionary. Usually however, you will prefer to use the WRKOBJR command because it lets you work with the real objects currently on your system.

The syntax for the WRKOBJRX command is given below. The parameters and their meanings follow.

```
                    .-*CURLIB/------.       .-*ALL----------.
>>--WRKOBJRX----LIB-+---------------+---OBJ-+---------------+--------------------------->
                    +-*PRV/---------+       +-object_name---+
                    +-*LIBL/--------+       '-*generic_name-'
                    +-*USRLIBL/-----+
                    +-*ALLUSR/------+
                    +-*ALL/---------+
                    '-library_name/-'

           .-*ALL--------.          .-*ALL-------------.
>--OBJTYPE-+-------------+---OBJATR-+------------------+-------------------------------->
           +-object_type-+          +-object_attribute-+
           +-*DBF--------+          '-generic_name-----'
           '-*SRCF-------'
                                                                                Required
=========================================================================================
                                                                                Optional

          .-*PRV----.           .-*PRV-.          .-*PRV-.          .-*PRV-.
>--DSPLVL-+---------+---DSPFILE-+------+---DSPFMT-+------+---DSPFLD-+------+------------>
          '-numeric-'           +-*YES-+          +-*YES-+          +-*YES-+
                                '-*NO--'          '-*NO--'          '-*NO--'

          .-*PRV-.          .-*PRV-.          .-*PRV-.          .-*PRV-.
>--DSPMBR-+------+---DSPSUB-+------+---DSPPGM-+------+---DSPOTR-+------+---------------->
          +-*YES-+          +-*YES-+          +-*YES-+          +-*YES-+
          '-*NO--'          '-*NO--'          '-*NO--'          '-*NO--'

          .-*PRV-.          .----0----.          .-*ALL---.          .-*PRV---.
>--DSPUNQ-+------+---EXPLVL-+---------+---DSPUSG-+--------+---DTASET-+--------+--------->
          +-*YES-+          '-numeric-'          +-text---+          +-*FIRST-+
          '-*NO--'                               +-*NOCMD-+          '-name---'
                                                 +-*CMD---+
                                                 +-*IO----+
                                                 +-*INP---+
                                                 +-*OUT---+
                                                 '-*UPD---'

          .-*--------.           .-*CURLIB-.                     .-*FIRST-.
>--OUTPUT-+----------+---OUTFILE-+---------+---FILENAME---OUTMBR-+--------+------------->
          +-*PRINT---+           +-*LIBL---+                     '-member-'
          '-*OUTFILE-'           '-library-'

          .-*REPLACE-.
>--MBROPT-+----------+-----------------------------------------------------------------><
          '-*ADD-----'
```

If you route the information to the display, ABSTRACT options can be used to process items in the list. You must have the appropriate authority to the objects referenced by the option and *USE authority for each command that is run.

The WRKOBJRX report is similar to the display. Objects on the display and report are presented using an indented list. The list is ordered alphabetically by object type. The format of the output file is similar, except that a level number field reports the indenting level of the objects in the file.

LIB Parameter

Specifies the library name(s) to use in searching the cross reference dictionary. If no library is indicated, *CURLIB is assumed. The possible library values are:

<u>*CURLIB:</u> The current library name value is used in the search. If no library is specified as the current library for the job, the QGPL library is used.

***PRV:** The library used during the previous session, regardless of which of the ABSTRACT WRKxxx commands was used.

***ALL:** All libraries in the cross reference should be included.

library-name: The cross reference dictionary is searched for the specific library you name.

OBJ Parameter

Specifies the object name. You can use this parameter to work with all the objects or a subset of objects in the indicated library.

<u>*ALL:</u> All cross reference records for the specified library(s) are selected.

object-name: Only objects with this specific name will be included in the list.

***generic*-name:** Specify a partial object name qualified by an asterisk (*) to select several objects meeting the criteria. The following generic forms are allowed: ABC*, *ABC, *ABC*, A*B, abc*, **ALL. Refer to Appendix G, General Functions, in Volume 1 of the <u>CL Reference</u> manual for more information about generic names.

OBJTYPE Parameter

Specifies the object type. You can use this parameter to work with all object types or a subset of objects. An expanded description of the OBJTYPE parameter and a list of valid OS/400 object types can be found in Appendix A, Expanded Parameter Descriptions, of Volume 1 of the <u>CL Reference</u> manual.

<u>*ALL:</u> All objects regardless of their type are included.

object-type: Specify any valid system object type to display a list of all objects of that particular type.

DSPLVL Parameter

Specifies the maximum depth of the relationship display. This value controls the number of indented levels that appear under the leftmost (parent) objects.

<u>*PRV:</u> The level used for this parameter during the previous session.

numeric: Specify a value between zero (0) and ten (10) that indicates the maximum level of indentation to appear on the object list display.

DSPFILE Parameter

Specify whether cross reference information indicating references to file (*FILE) objects should appear on the display.

<u>*PRV:</u> Use the value for this parameter from the previous session.

***YES:** All files referenced by objects in the list will be included on the display.

***NO:** None of the files referenced by an object in the list will appear.

DSPFMT Parameter

Specify whether cross reference information indicating references to file formats (*FMT) should appear on the display.

<u>*PRV:</u> Use the value for this parameter from the previous session.

***YES:** All file formats referenced by objects in the list will be included on the display.

***NO:** None of the files referenced by an object in the list will appear.

DSPFLD Parameter

Specify whether cross reference information indicating references to file fields (*FLD) should appear on the display.

<u>*PRV:</u> Use the value for this parameter from the previous session.

***YES:** All fields referenced by an object in the list will be included on the display.

***NO:** None of the files referenced by an object in the list will appear.

DSPMBR Parameter

Specify whether cross reference information indicating references to file members (*MBR) should appear on the display.

<u>*PRV:</u> Use the value for this parameter from the previous session.

***YES:** All members referenced by an object in the list will be included on the display.

***NO:** None of the file members referenced by an object in the list will appear.

DSPSUB Parameter

Specify whether cross reference information indicating program references to subroutines (*SUB) should appear on the display.

<u>*PRV:</u> Use the value for this parameter from the previous session.

***YES:** All subroutines referenced by a program in the list will be included on the display.

***NO:** No program references to subroutines will appear.

DSPPGM Parameter

Specify whether cross reference information indicating references to programs (*PGM) should appear on the display.

<u>*PRV:</u> Use the value for this parameter from the previous session.

***YES:** All programs referenced by an object in the list will be included on the display.

***NO:** None of the programs referenced by an object in the list will appear.

DSPOTR Parameter

Specify whether cross reference information indicating references to miscellaneous object types (other than those above) should appear on the display. This will includes references to commands (*CMD), data areas (*DTAARA), job descriptions (*JOBD), etc.

<u>*PRV:</u> Use the value for this parameter from the previous session.

***YES:** All miscellaneous object types referenced by an object in the list will be included on the display.

***NO:** None of the miscellaneous object types referenced by an object in the list will appear.

DSPUNQ Parameter

Specify whether duplicate cross reference records for an object should be suppressed.

<u>*PRV:</u> Use the value for this parameter from the previous session.

***YES:** If a given qualified object is referenced more than once, only one line will appear on the display or report. All references will be collapsed into a single line. The USAGE column will indicate the first type of use. The SEQ() values in the EXTENDED DESCRIPTION column will list the locations where the object was referenced.

***NO:** Each reference will appear on a separate line. This is especially useful when a program works with an object in several ways, such as opening a file twice.

EXPLVL Parameter

Specifies the number of recursive iterations for the list explosion. This value determines the depth of the explosion tree before additional relationships are ignored.

<u>0:</u> No explosion recursion occurs for objects on the display. All references for the parent object are given, but references for the underlying objects are not.

numeric: Specify a value between zero (0) and ninety-nine (99) that indicates the maximum number of reference generations that should be shown for each parent object on the display.

DSPUSG Parameter

Lets you filter the references in the list. Only references with the specific type of usage you select will appear beneath a parent object.

ALL: No usage selection will be used to filter references from the list.

text: Only references with usage values matching this text will be included. (c.f. CALL, Input, ...)

***CMD:** Only objects referenced through CL commands will be included in the list.

***NONCMD:** Command references in CL programs will be suppressed from the list. Only non-command (Unknown and input/output types) references will be included.

***IO:** Files with input, output, or update (including delete) usage will be included in the list. Other objects will be omitted.

***INP:** Files with input usage will be included in the list. Other objects will be omitted.

***OUT:** Files with output usage will be included in the list. Other objects will be omitted.

***UPD:** Files with update (or delete) usage will be included in the list. Other objects will be omitted.

***UPDOUT:** Files with output or update (including delete) usage will be included in the list. Other objects will be omitted.

DTASET Parameter

Specify the cross reference data set to be used in acquiring the information.

PRV: Use the value for this parameter from the previous session.

***FIRST:** information from the first (default) data set should be accessed.

Name: Specify any existing data set. Use the command prompter to find the names available to you.

OUTPUT Parameter

Indicates where the results of the command should be routed. Specify:

***:** Information will be displayed at the workstation.

***PRINT:** Information will be spooled to the ABSTRACTPRT printer file for printing.

***OUTFILE:** Information will be sent to a database file.

OUTFILE Parameter

Specifies the name of the database file to receive output from the command. The file name must be specified and you must have proper data rights to add records to it. You can specify one of two special values for the library name. If the outfile does not exist prior to execution, the command will create it unless *LIBL is specified for the file's library.

<u>*CURLIB:</u> The job's current library will be used to locate the file. If it is not found, the output file will be created in the current library.

***LIBL:** The job's library list will be searched for the indicated file.

A "pattern" outfile in the ABSTRACT library named APOUTFIL supplies information regarding the size, allocation parameters, and maximum number of members allowed. You can alter the characteristics of the created output files by using the Change Physical File (CHGPF) command on the pattern file.

OUTMBR Parameter

Specifies the name of the database file member(s) to which the command output is directed. Specify:

<u>*FIRST:</u> output is directed to the first member in the file. If this value is specified and the member does not exist, a member will be created with the same name as the file specified in the OUTFILE parameter.

member–name: output is directed to the named member in the file. If the member does not exist, it will be added to the file.

MBROPT Parameter

If the output file exists before the command is run, this keyword indicates whether records in the file will be cleared before the new information is placed into the file.

<u>*REPLACE:</u> existing records will be cleared from the output member prior to inserting new information.

***ADD:** records currently in the member are retained and new records will be added to them.

Examples

```
WRKOBJRX PILOT DSPLVL(0)
```

All objects from the PILOT library that are loaded into the cross references will be presented on the display. No reference information will be shown since a zero reference level was specified.

```
WRKOBJRX *CURLIB OBJTYPE(*PGM) DSPPGM(*YES) DSPFILE(*YES)
     DSPFLD(*YES) DSPMBR(*YES) DSPSUB(*YES) DSPOTR(*YES)
     DSPUSG(*ALL) DSPLVL(10) EXPLVL(0) OUTPUT(*PRINT)
```

All reference information for the job's current library (*CURLIB) will be printed.

Work With File Group References (WRKFGR)

The WRKFGR command makes it easy for you to find all the objects that reference one or more of the files in a file group. Since it is an object reference command, the results show the referencing objects as parent items in the list, and the member(s) of the file group indented beneath them, along with other referenced objects.

The command begins by locating the file you name in the LIB and FILE parameters and acquiring the names of all related database files. If you supply a physical file, ABSTRACT determines the logical files that use it. If you supply a logical file name, ABSTRACT determines all the physicals contributing to the logical and then locates all the logical files built over this group. This operation takes place in real-time, without reference to the cross reference files.

After the files in the group have been identified, the cross reference dictionary is searched to find parent objects that make references to one or more of them. Each parent object is presented on a list display with all the references that it makes.

The WRKFGR command is very similar to the Work With File Group Usage (WRKFGU) command, except that it shows object references rather than object usage. With WRKFGR, parent items in the list are the programs, queries, etc. that reference the file. With WRKFGU, parent items are files from the file group and the indented items in the list are the programs, queries, etc. that use them.

If you route the information to the display, ABSTRACT options can be used to process items in the list. You must have the appropriate authority to the objects referenced by the option and *USE authority for each command that is run.

The WRKFGR report is similar to its display. Objects on the display and report are presented using an indented list. The list is ordered alphabetically. The format of the output file is similar, except that a level number field reports the indenting level of the objects in the file.

The syntax for the WRKFGR command is given below. The parameters and their meanings follow.

```
                  .-*CURLIB/------.
>>--WRKFGR----LIB-+---------------+---FILE-+-file_name-+-------------------------------->
                  +-*PRV/---------+
                  +-*ALL/---------+
                  '-library_name/-'
                                                                                Required
=========================================================================================
                                                                                Optional

          .-*PRV----.           .-*PRV-.          .-*PRV-.          .-*PRV-.
>--DSPLVL-+---------+---DSPFILE-+------+---DSPFMT-+------+---DSPFLD-+------+------------>
          '-numeric-'           +-*YES-+          +-*YES-+          +-*YES-+
                                '-*NO--'          '-*NO--'          '-*NO--'

          .-*PRV-.          .-*PRV-.          .-*PRV-.          .-*PRV-.
>--DSPMBR-+------+---DSPSUB-+------+---DSPPGM-+------+---DSPOTR-+------+---------------->
          +-*YES-+          +-*YES-+          +-*YES-+          +-*YES-+
          '-*NO--'          '-*NO--'          '-*NO--'          '-*NO--'

          .-*PRV-.          .----0----.          .-*ALL---.          .-*PRV---.
>--DSPUNQ-+------+---EXPLVL-+---------+---DSPUSG-+--------+---DTASET-+--------+--------->
          +-*YES-+          '-numeric-'          +-text---+          +-*FIRST-+
          '-*NO--'                               +-*NOCMD-+          '-name---'
                                                 +-*CMD---+
                                                 +-*IO----+
                                                 +-*INP---+
                                                 +-*OUT---+
                                                 '-*UPD---'

          .-*--------.           .-*CURLIB-.                     .-*FIRST-.
>--OUTPUT-+----------+---OUTFILE-+---------+---FILENAME---OUTMBR-+--------+------------->
          +-*PRINT---+           +-*LIBL---+                     '-member-'
          '-*OUTFILE-'           '-library-'

          .-*REPLACE-.
>--MBROPT-+----------+-----------------------------------------------------------------><
          '-*ADD-----'
```

LIB Parameter

Specifies the library containing the file named in the FILE parameter. If no library is indicated, *CURLIB is assumed. The possible library values are listed below. You must have *USE authority to the library indicated by this parameter.

<u>*CURLIB:</u> The current library is searched. If no library is specified as the current library for the job, the QGPL library is used.

***PRV:** The library used during the previous session, regardless of which of the ABSTRACT WRKxxx commands was used.

***LIBL:** All libraries in the user and system portions of the job’s library list are searched.

library-name: Only the specific library is searched for the objects.

FILE Parameter

Specifies the name of the file in the file group. It can be either a physical or logical database file. You must have *USE authority to it.

DSPLVL Parameter

Specifies the maximum depth of the relationship display. This value controls the number of indented levels that appear under the leftmost (parent) objects.

<u>*PRV:</u> The level used for this parameter during the previous session.

numeric: Specify a value between zero (0) and ten (10) that indicates the maximum level of indentation to appear on the object list display.

DSPFILE Parameter

Specify whether cross reference information indicating references to file (*FILE) objects should appear on the display.

<u>*PRV:</u> Use the value for this parameter from the previous session.

***YES:** All files referenced by objects in the list will be included on the display.

***NO:** None of the files referenced by an object in the list will appear.

DSPFMT Parameter

Specify whether cross reference information indicating references to file formats (*FMT) should appear on the display.

<u>*PRV:</u> Use the value for this parameter from the previous session.

***YES:** All file formats referenced by objects in the list will be included on the display.

***NO:** None of the files referenced by an object in the list will appear.

DSPFLD Parameter

Specify whether cross reference information indicating references to file fields (*FLD) should appear on the display.

<u>*PRV:</u> Use the value for this parameter from the previous session.

***YES:** All fields referenced by an object in the list will be included on the display.

***NO:** None of the files referenced by an object in the list will appear.

DSPMBR Parameter

Specify whether cross reference information indicating references to file members (*MBR) should appear on the display.

***PRV:** Use the value for this parameter from the previous session.

***YES:** All members referenced by an object in the list will be included on the display.

***NO:** None of the file members referenced by an object in the list will appear.

DSPSUB Parameter

Specify whether cross reference information indicating program references to subroutines (*SUB) should appear on the display.

***PRV:** Use the value for this parameter from the previous session.

***YES:** All subroutines referenced by a program in the list will be included on the display.

***NO:** No program references to subroutines will appear.

DSPPGM Parameter

Specify whether cross reference information indicating references to programs (*PGM) should appear on the display.

***PRV:** Use the value for this parameter from the previous session.

***YES:** All programs referenced by an object in the list will be included on the display.

***NO:** None of the programs referenced by an object in the list will appear.

DSPOTR Parameter

Specify whether cross reference information indicating references to miscellaneous object types (other than those above) should appear on the display. This will includes references to commands (*CMD), data areas (*DTAARA), job descriptions (*JOBD), etc.

***PRV:** Use the value for this parameter from the previous session.

***YES:** All miscellaneous object types referenced by an object in the list will be included on the display.

***NO:** None of the miscellaneous object types referenced by an object in the list will appear.

DSPUNQ Parameter

Specify whether duplicate cross reference records for an object should be suppressed.

***PRV:** Use the value for this parameter from the previous session.

***YES:** If a given qualified object is referenced more than once, only one line will appear on the display or report. All references will be collapsed into a single line. The USAGE column will indicate the first type of use. The SEQ() values in the EXTENDED DESCRIPTION column will list the locations where the object was referenced.

***NO:** Each reference will appear on a separate line. This is especially useful when a program works with an object in several ways, such as opening a file twice.

EXPLVL Parameter

Specifies the number of recursive iterations for the list explosion. This value determines the depth of the explosion tree before additional relationships are ignored.

<u>0:</u> No explosion recursion occurs for objects on the display. All references for the parent object are given, but references for the underlying objects are not.

numeric: Specify a value between zero (0) and ninety-nine (99) that indicates the maximum number of reference generations that should be shown for each parent object on the display.

DSPUSG Parameter

Lets you filter the references in the list. Only references with the specific type of usage you select will appear beneath a parent object.

<u>*ALL:</u> No usage selection will be used to filter references from the list.

text: Only references with usage values matching this text will be included. (c.f. CALL, Input, ...)

***CMD:** Only objects referenced through CL commands will be included in the list.

***NONCMD:** Command references in CL programs will be suppressed from the list. Only non-command (Unknown and input/output types) references will be included.

***IO:** Files with input, output, or update (including delete) usage will be included in the list. Other objects will be omitted.

***INP:** Files with input usage will be included in the list. Other objects will be omitted.

***OUT:** Files with output usage will be included in the list. Other objects will be omitted.

***UPD:** Files with update (or delete) usage will be included in the list. Other objects will be omitted.

***UPDOUT:** Files with output or update (including delete) usage will be included in the list. Other objects will be omitted.

DSPUSG Parameter

Specify criteria that allows you to select references for a specific type of command. Only references matching the selected usage value will appear in the list beneath a parent object.

<u>*ALL:</u> No usage selection will be used to filter references from the list.

text: Only references with usage values matching this text will be included. (c.f. CALL, Input, ...)

***NONCMD:** Command references in CL programs will be suppressed from the list. Only non-command references will be included. The list will include CL program references for objects with Unknown or input/output attributes, in addition to all high level language references.

DTASET Parameter

Specify the cross reference data set to be used in acquiring the information.

<u>*PRV:</u> Use the value for this parameter from the previous session.

***FIRST:** information from the first (default) data set should be accessed.

Name: Specify any existing data set. Use the command prompter to find the names available to you.

OUTPUT Parameter

Indicates where the results of the command should be routed. Specify:

<u>*:</u> Information will be displayed at the workstation.

***PRINT:** Information will be spooled to the ABSTRACTPRT printer file for printing.

***OUTFILE:** Information will be sent to a database file.

OUTFILE Parameter

Specifies the name of the database file to receive output from the command. The file name must be specified and you must have proper data rights to add records to it. You can specify one of two special values for the library name. If the outfile does not exist prior to execution, the command will create it unless *LIBL is specified for the file's library.

<u>*CURLIB:</u> The job's current library will be used to locate the file. If it is not found, the output file will be created in the current library.

***LIBL:** The job's library list will be searched for the indicated file.

A "pattern" outfile in the ABSTRACT library named APOUTFIL supplies information regarding the size, allocation parameters, and maximum number of members allowed. You can alter the characteristics of the created output files by using the Change Physical File (CHGPF) command on the pattern file.

OUTMBR Parameter

Specifies the name of the database file member(s) to which the command output is directed. Specify:

<u>*FIRST:</u> output is directed to the first member in the file. If this value is specified and the member does not exist, a member will be created with the same name as the file specified in the OUTFILE parameter.

member–name: output is directed to the named member in the file. If the member does not exist, it will be added to the file.

MBROPT Parameter

If the output file exists before the command is run, this keyword indicates whether records in the file will be cleared before the new information is placed into the file.

<u>*REPLACE:</u> existing records will be cleared from the output member prior to inserting new information.

***ADD:** records currently in the member are retained and new records will be added to them.

Examples

Sample displays and printed output from the file group usage commands is presented and described in *Part 3, Object Relations*.

```
WRKFGR PILOT JOBSCH DSPLVL(0)
```

The PILOT/JOBSCH file will be located and all associated physical and logical files will be determined. Objects that reference one or more of the files in the list will be presented on the object list display. The DSPLVL(0) parameter causes all reference information to be suppressed.

```
WRKFGR PILOT JOBSCH DSPFILE(*YES)
```

The PILOT/JOBSCH file will be located and all associated physical and logical files will be determined. Objects that reference one or more of the files in the list will be presented on the object list display in addition to all files that they reference.

Work With X-ref Object Usage (WRKOBJUX)

The Work With X-ref Object Usage (WRKOBJUX) command shows information for objects in the cross reference dictionary. It is the opposite of the WRKOBJRX command. It present lists showing object usage - how and where objects within your application get used. The command contrasts with WRKOBJR, WRKOBJRX, and WRKOBJRS which show the references made by your application objects.

Unlike the WRKOBJU command, it searches the cross reference data set (instead of an actual library) for objects that match the naming criteria you supply. The cross reference information obtained during the initialization (see chapter 1) is the source for objects that are shown as parent objects on the display or printout.

You will find this command to be especially useful when you need information about items that are not true objects. Fields, subroutines, and copy members cannot participate as parent items in a list that shows only actual OS/400 objects so they won't appear when you use the WRKOBJU command. WRKOBJUX will show information about these items because they base their work on cross reference items not library objects.

```
                    .-*CURLIB/------.       .-*ALL----------.
>>--WRKOBJUX----LIB-+---------------+---OBJ-+---------------+---------------------------->
                    +-*PRV/---------+       +-object_name---+
                    +-*ALL/---------+       '-*generic_name-'
                    '-*library_name-'

           .-*ALL--------.
>--OBJTYPE-+-------------+--------------------------------------------------------------->
           '-object_type-'
                                                                                Required
==========================================================================================
                                                                                Optional

          .-*PRV----.           .-*PRV-.          .-*PRV-.          .-*PRV-.
>--DSPLVL-+---------+---DSPFILE-+------+---DSPFMT-+------+---DSPFLD-+------+------------->
          '-numeric-'           +-*YES-+          +-*YES-+          +-*YES-+
                                '-*NO--'          '-*NO--'          '-*NO--'

          .-*PRV-.          .-*PRV-.          .-*PRV-.          .-*PRV-.
>--DSPMBR-+------+---DSPSUB-+------+---DSPPGM-+------+---DSPOTR-+------+----------------->
          +-*YES-+          +-*YES-+          +-*YES-+          +-*YES-+
          '-*NO--'          '-*NO--'          '-*NO--'          '-*NO--'

          .-*PRV-.          .----0----.          .-*ALL---.          .-*PRV---.
>--DSPUNQ-+------+---EXPLVL-+---------+---DSPUSG-+--------+---DTASET-+--------+---------->
          +-*YES-+          '-numeric-'          +-text---+          +-*FIRST-+
          '-*NO--'                               +-*NOCMD-+          '-name---'
                                                 +-*CMD---+
                                                 +-*IO----+
                                                 +-*INP---+
                                                 +-*OUT---+
                                                 '-*UPD---'

          .-*--------.           .-*CURLIB-.                     .-*FIRST-.
>--OUTPUT-+----------+---OUTFILE-+---------+---FILENAME---OUTMBR-+--------+-------------->
          +-*PRINT---+           +-*LIBL---+                     '-member-'
          '-*OUTFILE-'           '-library-'

          .-*REPLACE-.
>--MBROPT-+----------+------------------------------------------------------------------><
          '-*ADD-----'
```

If you route the information to the display, ABSTRACT options can be used to process items in the list. You must have the appropriate authority to the objects referenced by the option and *USE authority for each command that is run.

The WRKOBJUX report is similar to its display. Objects on the display and report are presented using an indented list. The list is ordered alphabetically by object type. The format of the output file is similar, except that a level number field reports the “indenting level” of the objects in the file.

LIB Parameter

Specifies the qualifier name to use in searching the cross reference dictionary. If no library is indicated, *CURLIB is assumed. The possible library values are:

__*CURLIB:__ The current library name value is used in the search. If no library is specified as the current library for the job, the QGPL library is used.

***PRV:** The library used during the previous session, regardless of which of the ABSTRACT WRKxxx commands was used.

***ALL:** All libraries in the cross reference should be included.

library-name: The cross reference dictionary is searched for the specific library you name.

If The WRKOBJRX command specifies *FLD, *SUB, or *MBR as the object type, this parameter specifies the file qualifier for the field, subroutine, or copy member named by the OBJ parameter.

OBJ Parameter

Specifies the object name. You can use this parameter to work with all the objects or a subset of objects in the indicated library.

__*ALL:__ All cross reference records for the specified library(s) are selected.

object-name: Only objects with this specific name will be included in the list.

***generic*-name:** Specify a partial object name qualified by an asterisk (*) to select several objects meeting the criteria. The following generic forms are allowed: ABC*, *ABC, *ABC*, A*B, abc*, **ALL. Refer to Appendix G, General Functions, in Volume 1 of the CL Reference manual for more information about generic names.

OBJTYPE Parameter

Specifies the object type. You can use this parameter to work with all object types or a subset of objects. An expanded description of the OBJTYPE parameter and a list of valid OS/400 object types can be found in Appendix A, Expanded Parameter Descriptions, of Volume 1 of the CL Reference manual.

__*ALL:__ All objects regardless of their type are included.

object-type: Specify any valid system object type to display a list of all objects of that particular type.

DSPLVL Parameter

Specifies the maximum depth of the relationship display. This value controls the number of indented levels that appear under the leftmost (parent) objects.

__*PRV:__ The level used for this parameter during the previous session.

numeric: Specify a value between zero (0) and ten (10) that indicates the maximum level of indentation to appear on the object list display.

DSPFILE Parameter

Specify whether cross reference information indicating references to file (*FILE) objects should appear on the display.

PRV: Use the value for this parameter from the previous session.

YES: All files referenced by objects in the list will be included on the display.

NO: None of the files referenced by an object in the list will appear.

DSPFMT Parameter

Specify whether cross reference information indicating references to file formats (*FMT) should appear on the display.

PRV: Use the value for this parameter from the previous session.

YES: All file formats referenced by objects in the list will be included on the display.

NO: None of the files referenced by an object in the list will appear.

DSPFLD Parameter

Specify whether cross reference information indicating references to file fields (*FLD) should appear on the display.

PRV: Use the value for this parameter from the previous session.

YES: All fields referenced by an object in the list will be included on the display.

NO: None of the files referenced by an object in the list will appear.

DSPMBR Parameter

Specify whether cross reference information indicating references to file members (*MBR) should appear on the display.

PRV: Use the value for this parameter from the previous session.

YES: All members referenced by an object in the list will be included on the display.

NO: None of the file members referenced by an object in the list will appear.

DSPSUB Parameter

Specify whether cross reference information indicating program references to subroutines (*SUB) should appear on the display.

PRV: Use the value for this parameter from the previous session.

YES: All subroutines referenced by a program in the list will be included on the display.

NO: No program references to subroutines will appear.

DSPPGM Parameter

Specify whether cross reference information indicating references to programs (*PGM) should appear on the display.

<u>*PRV:</u> Use the value for this parameter from the previous session.

***YES:** All programs referenced by an object in the list will be included on the display.

***NO:** None of the programs referenced by an object in the list will appear.

DSPOTR Parameter

Specify whether cross reference information indicating references to miscellaneous object types (other than those above) should appear on the display. This will includes references to commands (*CMD), data areas (*DTAARA), job descriptions (*JOBD), etc.

<u>*PRV:</u> Use the value for this parameter from the previous session.

***YES:** All miscellaneous object types referenced by an object in the list will be included on the display.

***NO:** None of the miscellaneous object types referenced by an object in the list will appear.

DSPUNQ Parameter

Specify whether duplicate cross reference records for an object should be suppressed.

<u>*PRV:</u> Use the value for this parameter from the previous session.

***YES:** If a parent object is referenced more than once, only one line will appear on the display or report. All references will be collapsed into a single line. The USAGE column will indicate the first type of use. The SEQ() values in the EXTENDED DESCRIPTION column will list the locations where the object was referenced.

***NO:** Each reference will appear on a separate line. This is especially useful when a program works with an object in several ways, such as opening a file twice.

EXPLVL Parameter

Specifies the number of recursive iterations for the list explosion. This value determines the depth of the explosion tree before additional relationships are ignored.

<u>0:</u> No explosion recursion occurs for objects on the display. All references for the parent object are given, but references for the underlying objects are not.

numeric: Specify a value between zero (0) and ninety-nine (99) that indicates the maximum number of reference generations that should be shown for each parent object on the display.

DSPUSG Parameter

Lets you filter the references in the list. Only objects using the parent in the manner you select will appear in the list.

ALL: No usage selection will be used to filter references from the list.

text: Only objects using the parent with usage values matching this text will be included. (c.f. CALL, Input, ...)

***CMD:** Only object usage via CL commands will be shown in the list.

***NONCMD:** Command references in CL programs will be suppressed from the list. Only non-command (Unknown and input/output types) uses will be included.

***IO:** Programs using files for input, output, or update (including delete) usage will be included in the list. Other objects will be omitted.

***INP:** Programs using files for input will be included in the list. Other objects will be omitted.

***OUT:** Programs using files for output will be included in the list. Other objects will be omitted.

***UPD:** Programs using files for update (or delete) will be included in the list. Other objects will be omitted.

***UPDOUT:** Files with output or update (including delete) usage will be included in the list. Other objects will be omitted.

DTASET Parameter

Specify the cross reference data set to be used in acquiring the information.

PRV: Use the value for this parameter from the previous session.

***FIRST:** information from the first (default) data set should be accessed.

Name: Specify any existing data set. Use the command prompter to find the names available to you.

OUTPUT Parameter

Indicates where the results of the command should be routed. Specify:

***:** Information will be displayed at the workstation.

***PRINT:** Information will be spooled to the ABSTRACTPRT printer file for printing.

***OUTFILE:** Information will be sent to a database file.

OUTFILE Parameter

Specifies the name of the database file to receive output from the command. The file name must be specified and you must have proper data rights to add records to it. You can specify one of two special values for the library name. If the outfile does not exist prior to execution, the command will create it unless *LIBL is specified for the file's library.

***CURLIB:** The job's current library will be used to locate the file. If it is not found, the output file will be created in the current library.

***LIBL:** The job's library list will be searched for the indicated file.

A "pattern" outfile in the ABSTRACT library named APOUTFIL supplies information regarding the size, allocation parameters, and maximum number of members allowed. You can alter the characteristics of the created output files by using the Change Physical File (CHGPF) command on the pattern file.

OUTMBR Parameter

Specifies the name of the database file member(s) to which the command output is directed. Specify:

<u>*FIRST:</u> output is directed to the first member in the file. If this value is specified and the member does not exist, a member will be created with the same name as the file specified in the OUTFILE parameter.

member–name: output is directed to the named member in the file. If the member does not exist, it will be added to the file.

MBROPT Parameter

If the output file exists before the command is run, this keyword indicates whether records in the file will be cleared before the new information is placed into the file.

<u>*REPLACE:</u> existing records will be cleared from the output member prior to inserting new information.

***ADD:** records currently in the member are retained and new records will be added to them.

Examples

```
WRKOBJUX LIB(PILOT) OBJ(HLP001) OBJTYPE(*PGM)
```

All objects that use the PILOT/HLP001 program will be presented on the display. The list will include any programs that make a <u>qualified</u> CALL or TFRCTL and any commands that use it as a command processing program (CPP) or validity checking program (VCP).

```
WRKOBJUX LIB(*ALL) OBJ(HLP001) OBJTYPE(*PGM)
```

This command works like the previous one except that all instances of the HLP001 program in the cross references will be used as parent objects on the list display. Consequently, all usage information for HLP001 will be displayed. Qualified references will appear underneath the specific qualified HLP001 program, unqualified references will appear underneath a *LIBL instance of the program.

```
WRKOBJUX HELPTXT LINTX OBJTYPE(*FLD) DSPLVL(10)
     DSPPGM(*YES) DSPFILE(*YES)
```

This command obtains usage information for the LINTX field in the HELPTXT file. All objects that reference the LINTX field in the HELPTXT file will appear on the display. Field, subroutine and copy member usage can be obtained using WRKOBJRX by using its name as the OBJ parameter, and either *ALL (to acquire all references of the named object) or the file name containing it as the LIB parameter.

```
WRKOBJUX *ALL CONVERT OBJTYPE(*SUB) DSPPGM(*YES)
```

All references to the CONVERT subroutine will be shown on the object list, regardless of the source file that they occur in.

Work With Object Usage (WRKOBJU)

The Work With Object Usage (WRKOBJU) command shows usage information for objects that exist on your system. It searches for objects currently on your system that match the object naming criteria you supply. Once found, it accesses the cross reference information obtained during the initialization (see chapter 1) to display the objects that reference them.

WRKOBJU is the opposite of the WRKOBJR command. It presents lists showing object usage - how and where objects within your application gets used. They contrast with the object reference commands (WRKOBJR, WRKOBJRX, WRKOBJRS) which show the references made by your application objects.

Unlike the WRKOBJUX command, will present both qualified and unqualified usage of the parent objects in the list. WRKOBJUX shows only usage that exactly matches the qualified names that you supply.

Because the WRKOBJU command begins with real objects in your libraries, it will not be able to present usage information for fields, subroutines, or copy members. Use the WRKOBJUX command to find usage information for these quasi objects.

The WRKOBJU command can show program-program relationships, program explosions, program-file and program-field relationships, and program usage of commands. Refer to *Chapter 2, Using ABSTRACT List Displays* for a description of the features and functions of the list display used by the WRKOBJU command.

Only the libraries to which you have *USE authority will be searched for the objects you name on the WRKOBJU command. Objects that you have no authority for will be excluded from the list that is presented.

If you route the information to the display, ABSTRACT options can be used to process items in the list. You must have the appropriate authority to the objects referenced by the option and *USE authority for each command that is run.

The WRKOBJU report is similar to its display. Objects on the display and report are presented using an indented list. The list is ordered alphabetically by object type. The format of the output file is similar, except that a level number field reports the indenting level of the objects in the file.

The syntax for the WRKOBJU command is given below. The parameters and their meanings follow.

```
                   .-*CURLIB/------.       .-*ALL----------.
>>--WRKOBJU----LIB-+---------------+---OBJ-+---------------+----------------------------->
                   +-*PRV/---------+       +-object_name---+
                   +-*LIBL/--------+       '-*generic_name-'
                   +-*USRLIBL/-----+
                   +-*ALLUSR/------+
                   +-*ALL/---------+
                   '-library_name/-'

           .-*ALL--------.          .-*ALL-------------.
>--OBJTYPE-+-------------+---OBJATR-+------------------+--------------------------------->
           +-object_type-+          +-object_attribute-+
           +-*DBF--------+          '-generic_name-----'
           '-*SRCF-------'
                                                                                Required
==========================================================================================
                                                                                Optional

          .-*PRV----.           .-*PRV-.          .-*PRV-.          .-*PRV-.
>--DSPLVL-+---------+---DSPFILE-+------+---DSPFMT-+------+---DSPFLD-+------+------------->
          '-numeric-'           +-*YES-+          +-*YES-+          +-*YES-+
                                '-*NO--'          '-*NO--'          '-*NO--'

          .-*PRV-.          .-*PRV-.          .-*PRV-.          .-*PRV-.
>--DSPMBR-+------+---DSPSUB-+------+---DSPPGM-+------+---DSPOTR-+------+----------------->
          +-*YES-+          +-*YES-+          +-*YES-+          +-*YES-+
          '-*NO--'          '-*NO--'          '-*NO--'          '-*NO--'

          .-*PRV-.          .----0----.          .-*ALL---.          .-*PRV---.
>--DSPUNQ-+------+---EXPLVL-+---------+---DSPUSG-+--------+---DTASET-+--------+---------->
          +-*YES-+          '-numeric-'          +-text---+          +-*FIRST-+
          '-*NO--'                               +-*NOCMD-+          '-name---'
                                                 +-*CMD---+
                                                 +-*IO----+
                                                 +-*INP---+
                                                 +-*OUT---+
                                                 '-*UPD---'

          .-*--------.           .-*CURLIB-.                     .-*FIRST-.
>--OUTPUT-+----------+---OUTFILE-+---------+---FILENAME---OUTMBR-+--------+-------------->
          +-*PRINT---+           +-*LIBL---+                     '-member-'
          '-*OUTFILE-'           '-library-'

          .-*REPLACE-.
>--MBROPT-+----------+------------------------------------------------------------------><
          '-*ADD-----'
```

LIB Parameter

Specifies the libraries that are searched for the objects indicated by the other parameters on the command. If no library is indicated, *CURLIB is assumed. The possible library values are:

<u>*CURLIB:</u> The current library is searched. If no library is specified as the current library for the job, the QGPL library is used.

***PRV:** The library used during the previous session, regardless of which of the ABSTRACT WRKxxx commands was used.

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***USRLIBL:** Only the libraries in the user portion of the library list are searched.

***ALLUSR:** All non-system libraries, including all user-defined libraries and the QGPL library, are searched. Libraries with names starting with the letter Q, other than QGPL, are not included.

***ALL:** All libraries in the system, including QSYS, are searched.

library-name: Only the specific library is searched for the objects.

OBJ Parameter

Specifies the object name. You can use this parameter to work with all the objects or a subset of objects in the indicated library.

<u>*ALL:</u> All objects in the specified library(s) are selected.

object-name: Only objects with this specific name will be included in the list.

***generic*-name:** Specify a partial object name qualified by an asterisk (*) to select several objects meeting the criteria. The following generic forms are allowed: ABC*, *ABC, *ABC*, A*B, abc*, **ALL. Refer to Appendix G, General Functions, in Volume 1 of the <u>CL Reference</u> manual for more information about generic names.

OBJTYPE Parameter

Specifies the object type. You can use this parameter to work with all object types or a subset of objects. An expanded description of the OBJTYPE parameter and a list of valid OS/400 object types can be found in Appendix A, Expanded Parameter Descriptions, of Volume 1 of the <u>CL Reference</u> manual.

<u>*ALL:</u> All objects regardless of their type are included.

object-type: Specify any valid system object type to display a list of all objects of that particular type.

***DBF:** All data base files will be included. These are *FILE objects with an attribute beginning with PF, LF, or DDMF.

OBJATR Parameter

Specifies the object attribute. You can use this parameter to work with all object attributes or only specific ones. An expanded description of the OBJATR parameter and a list of valid OS/400 object attributes can be found in Appendix A, Expanded Parameter Descriptions, of Volume 1 of the <u>CL Reference</u> manual.

<u>*ALL:</u> All objects in the specified library(s) are selected regardless of their attribute definition.

***BLANK:** Only objects with no attribute will be included.

object-attribute: Only objects with this specific attribute will be included in the list. PF-DTA can be used to select only non-source physical files. PF-SRC can be used to select only source physical files.

***generic*-name:** Specify a partial object attribute qualified by an asterisk (*) to select several objects meeting the criteria. The following generic forms are allowed: ABC*, *ABC, *ABC*, A*B, abc*, **ALL. For example, *RPG* will select RPG, RPG36, RPG38, and SQLRPG objects. Refer to Appendix G, General Functions, in Volume 1 of the CL Reference manual for more information about generic names.

DSPLVL Parameter

Specifies the maximum depth of the relationship display. This value controls the number of indented levels that appear under the leftmost (parent) objects.

***PRV:** The level used for this parameter during the previous session.

numeric: Specify a value between zero (0) and ten (10) that indicates the maximum level of indentation to appear on the object list display.

DSPFILE Parameter

Specify whether cross reference information indicating references to file (*FILE) objects should appear on the display.

***PRV:** Use the value for this parameter from the previous session.

***YES:** All files referenced by objects in the list will be included on the display.

***NO:** None of the files referenced by an object in the list will appear.

DSPFMT Parameter

Specify whether cross reference information indicating references to file formats (*FMT) should appear on the display.

***PRV:** Use the value for this parameter from the previous session.

***YES:** All file formats referenced by objects in the list will be included on the display.

***NO:** None of the files referenced by an object in the list will appear.

DSPFLD Parameter

Specify whether cross reference information indicating references to file fields (*FLD) should appear on the display.

***PRV:** Use the value for this parameter from the previous session.

***YES:** All fields referenced by an object in the list will be included on the display.

***NO:** None of the files referenced by an object in the list will appear.

DSPMBR Parameter

Specify whether cross reference information indicating references to file members (*MBR) should appear on the display.

<u>*PRV:</u> Use the value for this parameter from the previous session.

***YES:** All members referenced by an object in the list will be included on the display.

***NO:** None of the file members referenced by an object in the list will appear.

DSPSUB Parameter

Specify whether cross reference information indicating program references to subroutines (*SUB) should appear on the display.

<u>*PRV:</u> Use the value for this parameter from the previous session.

***YES:** All subroutines referenced by a program in the list will be included on the display.

***NO:** No program references to subroutines will appear.

DSPPGM Parameter

Specify whether cross reference information indicating references to programs (*PGM) should appear on the display.

<u>*PRV:</u> Use the value for this parameter from the previous session.

***YES:** All programs referenced by an object in the list will be included on the display.

***NO:** None of the programs referenced by an object in the list will appear.

DSPOTR Parameter

Specify whether cross reference information indicating references to miscellaneous object types (other than those above) should appear on the display. This will includes references to commands (*CMD), data areas (*DTAARA), job descriptions (*JOBD), etc.

<u>*PRV:</u> Use the value for this parameter from the previous session.

***YES:** All miscellaneous object types referenced by an object in the list will be included on the display.

***NO:** None of the miscellaneous object types referenced by an object in the list will appear.

DSPUNQ Parameter

Specify whether duplicate cross reference records for an object should be suppressed.

<u>*PRV:</u> Use the value for this parameter from the previous session.

***YES:** If a given qualified object is referenced more than once, only one line will appear on the display or report. All references will be collapsed into a single line. The USAGE column will indicate the first type of use. The SEQ() values in the EXTENDED DESCRIPTION column will list the locations where the object was referenced.

***NO:** Each reference will appear on a separate line. This is especially useful when a program works with an object in several ways, such as opening a file twice.

EXPLVL Parameter

Specifies the number of recursive iterations for the list explosion. This value determines the depth of the explosion tree before additional relationships are ignored.

<u>**0:**</u> No explosion recursion occurs for objects on the display. All references for the parent object are given, but references for the underlying objects are not.

numeric: Specify a value between zero (0) and ninety-nine (99) that indicates the maximum number of reference generations that should be shown for each parent object on the display.

DSPUSG Parameter

Specify criteria that allows you to select references for a specific type of command. Only references matching the selected usage value will appear in the list beneath a parent object.

<u>***ALL:**</u> No usage selection will be used to filter references from the list.

text: Only references with usage values matching this text will be included. (c.f. CALL, Input, ...)

***NONCMD:** Command references in CL programs will be suppressed from the list. Only non-command references will be included. The list will include CL program references for objects with Unknown or input/output attributes, in addition to all high level language references.

***IO:** Programs using files for input, output, or update (including delete) usage will be included in the list. Other objects will be omitted.

***INP:** Programs using files for input will be included in the list. Other objects will be omitted.

***OUT:** Programs using files for output will be included in the list. Other objects will be omitted.

***UPD:** Programs using files for update (or delete) will be included in the list. Other objects will be omitted.

***UPDOUT:** Files with output or update (including delete) usage will be included in the list. Other objects will be omitted.

DTASET Parameter

Specify the cross reference data set to be used in acquiring the information.

<u>***PRV:**</u> Use the value for this parameter from the previous session.

***FIRST:** information from the first (default) data set should be accessed.

Name: Specify any existing data set. Use the command prompter to find the names available to you.

OUTPUT Parameter

Indicates where the results of the command should be routed. Specify:

***:** Information will be displayed at the workstation.

***PRINT:** Information will be spooled to the ABSTRACTPRT printer file for printing.

***OUTFILE:** Information will be sent to a database file.

OUTFILE Parameter

Specifies the name of the database file to receive output from the command. The file name must be specified and you must have proper data rights to add records to it. You can specify one of two special values for the library name. If the outfile does not exist prior to execution, the command will create it unless *LIBL is specified for the file's library.

***CURLIB:** The job's current library will be used to locate the file. If it is not found, the output file will be created in the current library.

***LIBL:** The job's library list will be searched for the indicated file.

A "pattern" outfile in the ABSTRACT library named APOUTFIL supplies information regarding the size, allocation parameters, and maximum number of members allowed. You can alter the characteristics of the created output files by using the Change Physical File (CHGPF) command on the pattern file.

OUTMBR Parameter

Specifies the name of the database file member(s) to which the command output is directed. Specify:

***FIRST:** output is directed to the first member in the file. If this value is specified and the member does not exist, a member will be created with the same name as the file specified in the OUTFILE parameter.

member–name: output is directed to the named member in the file. If the member does not exist, it will be added to the file.

MBROPT Parameter

If the output file exists before the command is run, this keyword indicates whether records in the file will be cleared before the new information is placed into the file.

***REPLACE:** existing records will be cleared from the output member prior to inserting new information.

***ADD:** records currently in the member are retained and new records will be added to them.

Examples

```
WRKOBJU PILOT
```

Qualified and unqualified usage information for all the objects in the PILOT library that you have *USE authority to will be displayed.

```
WRKOBJU PRODLIB CUSTMAST OBJTYPE(*FILE) +
     DSPPGM(*YES) DSPFILE(*YES)
```

Qualified and unqualified references to the CUSTMAST file in the PRODLIB library will appear on the display. Any programs that use the file in PRODLIB or with a *LIBL reference will be listed, along with any logical files built over it.

Work With File Group Usage (WRKFGU)

In many cases you will need to know the objects in your application that use the database, regardless of the specific file name that is referenced. ABSTRACT provides file group usage to help you find all the references to physical data - whether the reference occurs through a physical file or one of the logical files that are built over it.

The command begins by locating the file you name in the LIB and FILE parameters and acquiring the names of all related database files. If you supply a physical file, ABSTRACT determines the logical files that use it. If you supply a logical file name, ABSTRACT determines all the physicals contributing to the logical and then locates all the logical files built over this group. This operation takes place in real-time, without reference to the cross reference files.

After the files in the group have been identified, the usage information from the cross reference dictionary is presented on a list display. Information is presented for each of the files in the group.

The WRKFGU command is very similar to the Work With File Group References (WRKFGR) command, except that it shows object usage rather than object references. With WRKFGU, parent items are files from the file group and the indented items in the list are the programs, queries, etc. that use them. With WRKFGR, parent items in the list are the programs, queries, etc. that reference the file.

The WRKFGU command will present both qualified and unqualified usage of the files in the group. This contrasts with the WRKOBJUX command that shows only usage that exactly matches the qualified names that you supply.

If you route the information to the display, ABSTRACT options can be used to process items in the list. You must have the appropriate authority to the objects referenced by the option and *USE authority for each command that is run.

The WRKFGU report is similar to its display. Objects on the display and report are presented using an indented list. The list is ordered alphabetically. The format of the output file is similar, except that a level number field reports the indenting level of the objects in the file.

The syntax for the WRKFGU command is given below. The parameters and their meanings follow.

```
                 .-*CURLIB------.
>>--WRKFGU----LIB-+--------------+---FILE-file_name------------------------------------->
                 +-*PRV---------+
                 +-*ALL/--------+
                 '-library_name-'
                                                                               Required
=========================================================================================
                                                                               Optional

         .-*PRV----.           .-*PRV-.           .-*PRV-.           .-*PRV-.
>--DSPLVL-+---------+---DSPFILE-+------+---DSPFMT-+------+---DSPFLD-+------+------------>
         '-numeric-'           +-*YES-+           +-*YES-+           +-*YES-+
                               '-*NO--'           '-*NO--'           '-*NO--'

         .-*PRV-.           .-*PRV-.           .-*PRV-.           .-*PRV-.
>--DSPMBR-+------+---DSPSUB-+------+---DSPPGM-+------+---DSPOTR-+------+---------------->
         +-*YES-+           +-*YES-+           +-*YES-+           +-*YES-+
         '-*NO--'           '-*NO--'           '-*NO--'           '-*NO--'

         .-*PRV-.           .----0----.           .-*ALL---.           .-*PRV---.
>--DSPUNQ-+------+---EXPLVL-+---------+---DSPUSG-+--------+---DTASET-+--------+--------->
         +-*YES-+           '-numeric-'           +-text---+           +-*FIRST-+
         '-*NO--'                                 +-*NOCMD-+           '-name---'
                                                  +-*CMD---+
                                                  +-*IO----+
                                                  +-*INP---+
                                                  +-*OUT---+
                                                  '-*UPD---'

         .-*--------.           .-*CURLIB-.                      .-*FIRST-.
>--OUTPUT-+----------+---OUTFILE-+---------+---FILENAME---OUTMBR-+--------+------------->
         +-*PRINT---+           +-*LIBL---+                      '-member-'
         '-*OUTFILE-'           '-library-'

         .-*REPLACE-.
>--MBROPT-+----------+-------------------------------------------------------------><
         '-*ADD-----'
```

LIB Parameter

Specifies the library containing the file named in the FILE parameter. If no library is indicated, *CURLIB is assumed. The possible library values are listed below. You must have *USE authority to the library indicated by this parameter.

<u>*CURLIB:</u> The current library is searched. If no library is specified as the current library for the job, the QGPL library is used.

***PRV:** The library used during the previous session, regardless of which of the ABSTRACT WRKxxx commands was used.

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

library-name: Only the specific library is searched for the objects.

FILE Parameter

Specifies the name of the file in the file group. It can be either a physical or logical database file. You must have *USE authority to it.

DSPLVL Parameter

Specifies the maximum depth of the relationship display. This value controls the number of indented levels that appear under the leftmost (parent) objects.

<u>*PRV:</u> The level used for this parameter during the previous session.

numeric: Specify a value between zero (0) and ten (10) that indicates the maximum level of indentation to appear on the object list display.

DSPFILE Parameter

Specify whether cross reference information indicating references to file (*FILE) objects should appear on the display.

<u>*PRV:</u> Use the value for this parameter from the previous session.

***YES:** All files referenced by objects in the list will be included on the display.

***NO:** None of the files referenced by an object in the list will appear.

DSPFMT Parameter

Specify whether cross reference information indicating references to file formats (*FMT) should appear on the display.

<u>*PRV:</u> Use the value for this parameter from the previous session.

***YES:** All file formats referenced by objects in the list will be included on the display.

***NO:** None of the files referenced by an object in the list will appear.

DSPFLD Parameter

Specify whether cross reference information indicating references to file fields (*FLD) should appear on the display.

<u>*PRV:</u> Use the value for this parameter from the previous session.

***YES:** All fields referenced by an object in the list will be included on the display.

***NO:** None of the files referenced by an object in the list will appear.

DSPMBR Parameter

Specify whether cross reference information indicating references to file members (*MBR) should appear on the display.

<u>*PRV:</u> Use the value for this parameter from the previous session.

***YES:** All members referenced by an object in the list will be included on the display.

***NO:** None of the file members referenced by an object in the list will appear.

DSPSUB Parameter

Specify whether cross reference information indicating program references to subroutines (*SUB) should appear on the display.

<u>*PRV:</u> Use the value for this parameter from the previous session.

***YES:** All subroutines referenced by a program in the list will be included on the display.

***NO:** No program references to subroutines will appear.

DSPPGM Parameter

Specify whether cross reference information indicating references to programs (*PGM) should appear on the display.

<u>*PRV:</u> Use the value for this parameter from the previous session.

***YES:** All programs referenced by an object in the list will be included on the display.

***NO:** None of the programs referenced by an object in the list will appear.

DSPOTR Parameter

Specify whether cross reference information indicating references to miscellaneous object types (other than those above) should appear on the display. This will includes references to commands (*CMD), data areas (*DTAARA), job descriptions (*JOBD), etc.

<u>*PRV:</u> Use the value for this parameter from the previous session.

***YES:** All miscellaneous object types referenced by an object in the list will be included on the display.

***NO:** None of the miscellaneous object types referenced by an object in the list will appear.

DSPUNQ Parameter

Specify whether duplicate cross reference records for an object should be suppressed.

<u>*PRV:</u> Use the value for this parameter from the previous session.

***YES:** If a given qualified object is referenced more than once, only one line will appear on the display or report. All references will be collapsed into a single line. The USAGE column will indicate the first type of use. The SEQ() values in the EXTENDED DESCRIPTION column will list the locations where the object was referenced.

***NO:** Each reference will appear on a separate line. This is especially useful when a program works with an object in several ways, such as opening a file twice.

EXPLVL Parameter

Specifies the number of recursive iterations for the list explosion. This value determines the depth of the explosion tree before additional relationships are ignored.

0: No explosion recursion occurs for objects on the display. All references for the parent object are given, but references for the underlying objects are not.

numeric: Specify a value between zero (0) and ninety-nine (99) that indicates the maximum number of reference generations that should be shown for each parent object on the display.

DSPUSG Parameter

Lets you filter the references in the list. Only references with the specific type of usage you select will appear beneath a parent object.

***ALL:** No usage selection will be used to filter references from the list.

text: Only references with usage values matching this text will be included. (c.f. CALL, Input, ...)

***CMD:** Only objects referenced through CL commands will be included in the list.

***NONCMD:** Command references in CL programs will be suppressed from the list. Only non-command (Unknown and input/output types) references will be included.

***IO:** Files with input, output, or update (including delete) usage will be included in the list. Other objects will be omitted.

***INP:** Files with input usage will be included in the list. Other objects will be omitted.

***OUT:** Files with output usage will be included in the list. Other objects will be omitted.

***UPD:** Files with update (or delete) usage will be included in the list. Other objects will be omitted.

***UPDOUT:** Files with output or update (including delete) usage will be included in the list. Other objects will be omitted.

DSPUSG Parameter

Specify criteria that allows you to select references for a specific type of command. Only references matching the selected usage value will appear in the list beneath a parent object.

***ALL:** No usage selection will be used to filter references from the list.

text: Only references with usage values matching this text will be included. (c.f. CALL, Input, ...)

***NONCMD:** Command references in CL programs will be suppressed from the list. Only non-command references will be included. The list will include CL program references for objects with Unknown or input/output attributes, in addition to all high level language references.

DTASET Parameter

Specify the cross reference data set to be used in acquiring the information.

***PRV:** Use the value for this parameter from the previous session.

***FIRST:** information from the first (default) data set should be accessed.

Name: Specify any existing data set. Use the command prompter to find the names available to you.

OUTPUT Parameter

Indicates where the results of the command should be routed. Specify:

***:** Information will be displayed at the workstation.

***PRINT:** Information will be spooled to the ABSTRACTPRT printer file for printing.

***OUTFILE:** Information will be sent to a database file.

OUTFILE Parameter

Specifies the name of the database file to receive output from the command. The file name must be specified and you must have proper data rights to add records to it. You can specify one of two special values for the library name. If the outfile does not exist prior to execution, the command will create it unless *LIBL is specified for the file's library.

***CURLIB:** The job's current library will be used to locate the file. If it is not found, the output file will be created in the current library.

***LIBL:** The job's library list will be searched for the indicated file.

A "pattern" outfile in the ABSTRACT library named APOUTFIL supplies information regarding the size, allocation parameters, and maximum number of members allowed. You can alter the characteristics of the created output files by using the Change Physical File (CHGPF) command on the pattern file.

OUTMBR Parameter

Specifies the name of the database file member(s) to which the command output is directed. Specify:

***FIRST:** output is directed to the first member in the file. If this value is specified and the member does not exist, a member will be created with the same name as the file specified in the OUTFILE parameter.

member–name: output is directed to the named member in the file. If the member does not exist, it will be added to the file.

MBROPT Parameter

If the output file exists before the command is run, this keyword indicates whether records in the file will be cleared before the new information is placed into the file.

<u>*REPLACE:</u> existing records will be cleared from the output member prior to inserting new information.

***ADD:** records currently in the member are retained and new records will be added to them.

Examples

Sample displays and printed output from the file group usage commands is presented and described in *Part 3, Object Relations*.

```
WRKFGU PILOT JOBSCH DSPLVL(0)
```

The PILOT/JOBSCH file will be located and all associated physical and logical files will be determined. They will be presented on the object list display without any usage information due to the DSPLVL(0) parameter.

```
WRKFGU PILOT JOBSCH DSPPGM(*YES)
```

Qualified and unqualified references to the files in the JOBSCH file group will be presented on the object list display.

Work With Field Group Usage (WRKFLDGU)

The Work With Field Group Usage (WRKFLDGU) command displays usage information for database fields regardless of the specific logical or physical file through which the reference occurs. Like the WRKFGU command, field group usage begins by determining all of the files that participate in the database structure containing a file you specify.

Once the file list is complete, ABSTRACT determines which field(s) in each of the files overlap the field specified on the WRKFLDGU command. These fields will be used as parent objects on a display (or report) that shows where and how they are used.

If you route the information to the display, ABSTRACT options can be used to process items in the list. You must have the appropriate authority to the objects referenced by the option and *USE authority for each command that is run.

The WRKFLDGU report is similar to its display. Objects on the display and report are presented using an indented list. The list is ordered alphabetically. The format of the output file is similar, except that a level number field reports the indenting level of the objects in the file.

LIB Parameter

Specifies the library containing the file named in the FILE parameter. If no library is indicated, *CURLIB is assumed. The possible library values are listed below. You must have *USE authority to the library indicated by this parameter.

<u>*CURLIB:</u> The current library is searched. If no library is specified as the current library for the job, the QGPL library is used.

***PRV:** The library used during the previous session, regardless of which of the ABSTRACT WRKxxx commands was used.

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

library-name: Only the specific library is searched for the objects.

FILE Parameter

Specifies the name of the file in the file group. It can be either a physical or logical database file. You must have *USE authority to it.

FIELD Parameter

Indicates the name of a field in the file specified by the FILE and LIB parameters. Usage information will be presented for the field in the file and for related fields in the file group.

<u>*ALL:</u> Usage information for each field in the file will be presented.

DSPLVL Parameter

Specifies the maximum depth of the relationship display. This value controls the number of indented levels that appear under the leftmost (parent) objects.

<u>*PRV:</u> The level used for this parameter during the previous session.

numeric: Specify a value between zero (0) and ten (10) that indicates the maximum level of indentation to appear on the object list display.

DSPFILE Parameter

Specify whether cross reference information indicating references to file (*FILE) objects should appear on the display.

<u>*PRV:</u> Use the value for this parameter from the previous session.

***YES:** All files referenced by objects in the list will be included on the display.

***NO:** None of the files referenced by an object in the list will appear.

DSPFMT Parameter

Specify whether cross reference information indicating references to file formats (*FMT) should appear on the display.

<u>*PRV:</u> Use the value for this parameter from the previous session.

***YES:** All file formats referenced by objects in the list will be included on the display.

***NO:** None of the files referenced by an object in the list will appear.

DSPFLD Parameter

Specify whether cross reference information indicating references to file fields (*FLD) should appear on the display.

<u>*PRV:</u> Use the value for this parameter from the previous session.

***YES:** All fields referenced by an object in the list will be included on the display.

***NO:** None of the files referenced by an object in the list will appear.

DSPMBR Parameter

Specify whether cross reference information indicating references to file members (*MBR) should appear on the display.

<u>*PRV:</u> Use the value for this parameter from the previous session.

***YES:** All members referenced by an object in the list will be included on the display.

***NO:** None of the file members referenced by an object in the list will appear.

DSPSUB Parameter

Specify whether cross reference information indicating program references to subroutines (*SUB) should appear on the display.

<u>*PRV:</u> Use the value for this parameter from the previous session.

***YES:** All subroutines referenced by a program in the list will be included on the display.

***NO:** No program references to subroutines will appear.

DSPPGM Parameter

Specify whether cross reference information indicating references to programs (*PGM) should appear on the display.

<u>***PRV:**</u> Use the value for this parameter from the previous session.

***YES:** All programs referenced by an object in the list will be included on the display.

***NO:** None of the programs referenced by an object in the list will appear.

DSPOTR Parameter

Specify whether cross reference information indicating references to miscellaneous object types (other than those above) should appear on the display. This will includes references to commands (*CMD), data areas (*DTAARA), job descriptions (*JOBD), etc.

<u>***PRV:**</u> Use the value for this parameter from the previous session.

***YES:** All miscellaneous object types referenced by an object in the list will be included on the display.

***NO:** None of the miscellaneous object types referenced by an object in the list will appear.

DSPUNQ Parameter

Specify whether duplicate cross reference records for an object should be suppressed.

<u>***PRV:**</u> Use the value for this parameter from the previous session.

***YES:** If a given qualified object is referenced more than once, only one line will appear on the display or report. All usage information will be collapsed into a single line. The USAGE column will indicate the first type of use.

***NO:** Each reference will appear on a separate line. This is especially useful when a program works with an object in several ways, such as opening a file twice.

EXPLVL Parameter

Specifies the number of recursive iterations for the list explosion. This value determines the depth of the explosion tree before additional relationships are ignored.

<u>**0:**</u> No explosion recursion occurs for objects on the display. All references for the parent object are given, but references for the underlying objects are not.

numeric: Specify a value between zero (0) and ninety-nine (99) that indicates the maximum number of reference generations that should be shown for each parent object on the display.

DSPUSG Parameter

Lets you filter the references in the list. Only references with the specific type of usage you select will appear beneath a parent object.

<u>***ALL:**</u> No usage selection will be used to filter references from the list.

text: Only references with usage values matching this text will be included. (c.f. CALL, Input, ...)

***CMD:** Only objects referenced through CL commands will be included in the list.

***NONCMD:** Command references in CL programs will be suppressed from the list. Only non-command (Unknown and input/output types) references will be included.

***IO:** Files with input, output, or update (including delete) usage will be included in the list. Other objects will be omitted.

***INP:** Files with input usage will be included in the list. Other objects will be omitted.

***OUT:** Files with output usage will be included in the list. Other objects will be omitted.

***UPD:** Files with update (or delete) usage will be included in the list. Other objects will be omitted.

***UPDOUT:** Files with output or update (including delete) usage will be included in the list. Other objects will be omitted.

DTASET Parameter

Specify the cross reference data et to be used in acquiring the information.

<u>***PRV:**</u> Use the value for this parameter from the previous session.

***FIRST:** information from the first (default) data set should be accessed.

Name: Specify any existing data set. Use the command prompter to find the names available to you.

OUTPUT Parameter

Indicates where the results of the command should be routed. Specify:

<u>***:**</u> Information will be displayed at the workstation.

***PRINT:** Information will be spooled to the ABSTRACTPRT printer file for printing.

***OUTFILE:** Information will be sent to a database file.

OUTFILE Parameter

Specifies the name of the database file to receive output from the command. The file name must be specified and you must have proper data rights to add records to it. You can specify one of two special values for the library name. If the outfile does not exist prior to execution, the command will create it unless *LIBL is specified for the file's library.

<u>*CURLIB:</u> The job’s current library will be used to locate the file. If it is not found, the output file will be created in the current library.

***LIBL:** The job’s library list will be searched for the indicated file.

A “pattern” outfile in the ABSTRACT library named APOUTFIL supplies information regarding the size, allocation parameters, and maximum number of members allowed. You can alter the characteristics of the created output files by using the Change Physical File (CHGPF) command on the pattern file.

OUTMBR Parameter

Specifies the name of the database file member(s) to which the command output is directed. Specify:

<u>*FIRST:</u> output is directed to the first member in the file. If this value is specified and the member does not exist, a member will be created with the same name as the file specified in the OUTFILE parameter.

member–name: output is directed to the named member in the file. If the member does not exist, it will be added to the file.

MBROPT Parameter

If the output file exists before the command is run, this keyword indicates whether records in the file will be cleared before the new information is placed into the file.

<u>*REPLACE:</u> existing records will be cleared from the output member prior to inserting new information.

***ADD:** records currently in the member are retained and new records will be added to them.

Examples

Sample displays and printed output from the file group usage commands is presented and described in Chapter 2.

```
WRKFLDGU PILOT JOBSCH JOBNAM DSPLVL(0)
```

The PILOT/JOBSCH file will be located and all associated physical and logical files will be determined. Each field that is related to the JOBNAM field (shares its physical positions) will be presented on the object list display without any usage information due to the DSPLVL(0) parameter.

```
WRKFLDGU PILOT JOBSCH JOBNAM DSPPGM(*YES)
```

Program usage information for the JOBNAM field in the PILOT/JOBSCH file will be presented along with usage information for all related fields in the file group.

File Analysis

The file analysis features of ABSTRACT provide you with information about database, display, and printer files. This chapter will discuss the file analysis functions and describe how they can be used to get on-line and printed information about the files in your application.

The ABSTRACT file analysis functions include:

Analyze File (ANZFILE) capability to provide real-time information about database and device files. The field definitions for each format in the file, member information, database relationships, access path description, etc. are available in addition to a formatted display showing current file attributes. Cross reference information describing program usage for fields in the file is also available.

Printed record layouts. This facility is used to submit requests to the batch subsystem for printed copies of database file formats.

Program described file references. During the initialization phase, RPG and COBOL definitions for program described files are stored in the ABSTRACT dictionary. This information can be printed or displayed on-line.

Printed representations of display file and printer file definitions.

The following sections in this chapter present the file analysis menu (APFILE) and its capabilities, and describe each function in detail.

File Analysis (APFILE) Menu

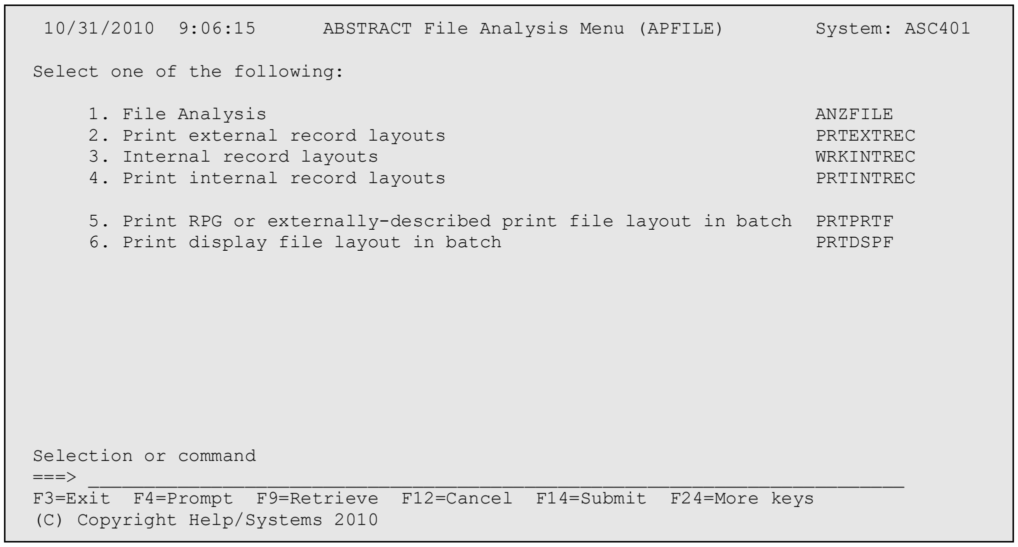
The file analysis menu (APFILE) can be accessed from the ABSTRACT main menu through option 4, or by typing:

```
GO ABSTRACT/APFILE
```

on a command entry line and pressing the Enter key.

The menu allows you to access all the file analysis functions of ABSTRACT and to display and print the output. It also allows you full access to your authorized command set through the command entry line at the bottom of the display. Like the other ABSTRACT menus, it can access the option file and default job queue setting through function keys.

Once you have started the menu, it will look like the one below.



```
 10/31/2010  9:06:15      ABSTRACT File Analysis Menu (APFILE)        System: ASC401

Select one of the following:

     1. File Analysis                                                 ANZFILE
     2. Print external record layouts                                 PRTEXTREC
     3. Internal record layouts                                       WRKINTREC
     4. Print internal record layouts                                 PRTINTREC

     5. Print RPG or externally-described print file layout in batch  PRTPRTF
     6. Print display file layout in batch                            PRTDSPF








Selection or command
===> _________________________________________________________________________
F3=Exit  F4=Prompt  F9=Retrieve  F12=Cancel  F14=Submit  F24=More keys
(C) Copyright Help/Systems 2010
```

Menu options

Menu options 1, 4, 5, and 6 will prompt (and then run) an ABSTRACT command. Depending on the menu item, the command will be submitted to the batch job queue or it will begin running immediately when you press the Enter key following the prompt display. Given that some requests can run for an extended period of time, this may not be an efficient use of your display device or interactive priority.

Your request will always be submitted to the batch subsystem if you type the menu option number and press F14. Refer to the section describing the ABSTRACT menu driver files in the Introduction for information about changing the batch/interactive environment for each menu option.

Select option 1 to prompt the ANZFILE command. The command prompt will appear and you will have the opportunity to specify the name of the database or device file that you want to examine. Refer to page 5-4 for more information about the ANZFILE command.

Use option 2 when you want to print the external record format definition for database files in your application. An interactive prompt will appear allowing you to choose the files you want documented. You can indicate files generically, or individually using the prompt. Once you have filled in the prompt, the request is automatically submitted to the batch subsystem for execution.

Options 3 and 4 can be used to learn about the program described specifications of your RPG and COBOL source code. Option 3 presents an interactive prompt that allows you to select a file and display/print all the program descriptions for it. See page 5-18 for a description of this function. Option 4 prints an internal record layout without passing through the interactive displays by using the Print Internal Record Layout (PRTINTREC) command.

Menu options 5 and 6 result in a printed representation of display or printout specifications. ABSTRACT can analyze RPG output specifications and both display and printer Data Definition Statements (DDS) and create a printed representation of them.

Menu Function Keys

As with other ABSTRACT menus, the file analysis menu provides several function key options. Notice that there are some differences (cf. F14, F19) from the function key set provided by OS/400 system menus.

F3	Exit the ABSTRACT environment
F4	Prompt the option or command that is on the command line
F9	Retrieve ‘backwards’ through previously executed commands
F12	Exit the menu display and return to the previous function
F14	Submit the option or command on the command line to batch using the default job description defined for this user
F15	Prompt the Submit Job (SBMJOB) command before submitting the option or the command on the command line
F16	Work with options in the user option file
F18	Change the default job description and user option file defined for the current user
F19	Retrieve ‘forwards’ through previously executed commands
F24	Display more function keys

Analyze File (ANZFILE) Command

The ABSTRACT Analyze File (ANZFILE) command will access the current file definition for database and device files on your system. You can use it to display the following information:

Field descriptions and buffer locations for each record format in the file

Member information

Database relationships

Access path descriptions

File attribute information

Object description data such as creation date, save/restore and usage information.

Program usage of fields

Where-used information for the file and its fields.

ABSTRACT cross references are not used by the file analysis programs except to deliver where-used information. You need not run the LOADXREF command prior to using the ANZFILE command.

The ANZFILE file analysis displays can be accessed through option 1 of the APFILE menu, or by entering the ANZFILE command on a command entry line.

Option 12=Work With in the standard ABSTRACT/QAUOOPT option file also runs the ANZFILE command. This means that the file analysis display can be accessed quickly and easily from any of the ABSTRACT object lists simply by placing option 12 next to a file item and pressing the Enter key.

```
                     .-*LIBL/-------.                      .-*PRV---.
>>--ANZFILE----FILE-+--------------+---file_name---DTASET-+--------+-------------------><
                     '-library_name-'                      '-*FIRST-'
```

FILE Parameter

Specifies the name of the file. The file must exist in the library indicated and you must have operational rights to it. The library name must be specified explicitly or must be the special value *LIBL.

<u>*LIBL:</u> All libraries in the user and system portions of the job’s library list are searched.

DTASET Parameter

Specifies the name of the data set you want to work with. You can accept the default or choose a separate data set, created by the Add Data Set (ADDDTASET) command, that contains the information. If the data set does not exist, an error will occur. The list of current data set names can be accessed by prompting the command and then prompting the data set parameter.

<u>*PRV:</u> Use the value for this parameter from the previous session. If no previous session has occurred, use *FIRST.

***FIRST:** information from the first (default) data set should be accessed.

Examples

```
ANZFILE FILE(QSYS/QADBXREF)
```

This command will locate the QADBXREF file in the QSYS library and present the initial display showing record format and member information.

```
ANZFILE CUSTMAST
```

Your current library list will be searched for a file named CUSTMAST. If one is found, the record format and member information for it will be displayed. Otherwise, an error message will be issued.

The File Analysis Display

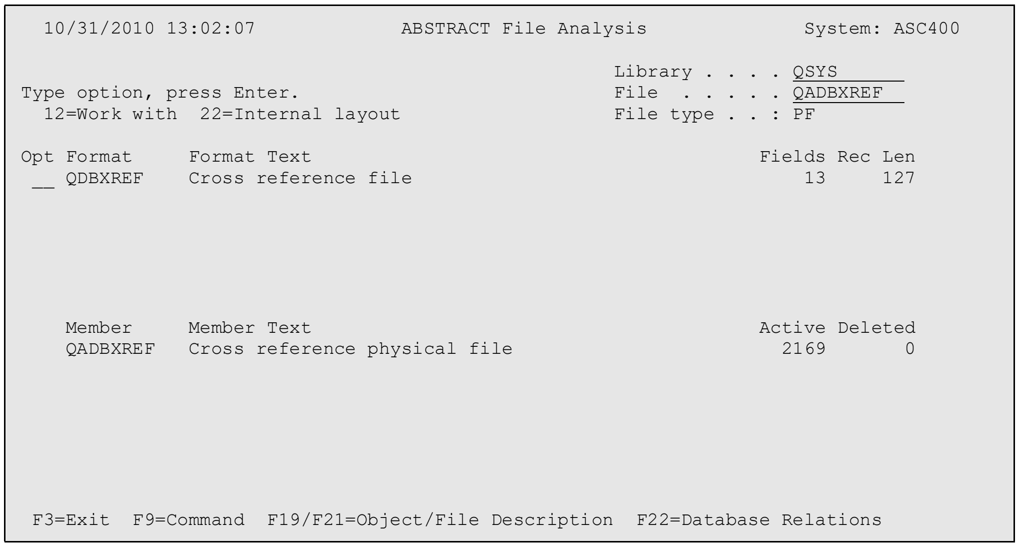
Once the ANZFILE command has been entered, ABSTRACT will access the indicated file, acquire information about it, and present it to you through a series of interactive displays. If you have selected a Distributed Data Management (DDM) file, ABSTRACT will attempt to reference the remote definition for the file. If you have chosen a program described file linked to an IDDU definition, ABSTRACT will display the external definition for the file.

The first display that is presented shows information about the record formats in the file, and member information if you have selected a database file. If you have chosen a physical file, you will see a display similar to the one on the next page.

The file name and its attribute are listed at the upper right corner of the display.

The display is divided in half. The top half of the screen lists all the externally defined record formats contained in the file. Format text, field count and record length in bytes is displayed for each format. If the top half of the display is full, press the roll keys to view more formats. Format information will be presented for both database and device files.

If you have selected a database file, the lower half of the display will list all of its members, if any, and show the member name, text and record counts. If the file has more members than can fit on the display page, you can position the cursor in the bottom half of the display and use the roll keys.



```
  10/31/2010 13:02:07            ABSTRACT File Analysis            System: ASC400

                                                     Library . . . . QSYS
Type option, press Enter.                            File  . . . . . QADBXREF
  12=Work with  22=Internal layout                   File type . . : PF

Opt Format     Format Text                                       Fields Rec Len
 __ QDBXREF    Cross reference file                                  13     127




    Member     Member Text                                       Active Deleted
    QADBXREF   Cross reference physical file                       2169       0





 F3=Exit  F9=Command  F19/F21=Object/File Description  F22=Database Relations
```

Entry Fields

Change the file you are working with by typing over the file and/or library name and pressing Enter. Information for the new file will replace the existing display.

Type option 12=Work with in the 'Opt' field to work with the external field definitions for one of the record formats on the display. A display similar to the one on page 5-6 will appear. Option 22 may be used to view program descriptions for this file. Refer to the description of the internal record layouts option on page 5-18.

Function Keys

Press F9 to request a pop-up command window. You will be able to enter new command requests and recall commands previously issued. Press F12 to exit the command window and return to the file analysis display.

Use F19 to view the current object description information. The Display Object Description (DSPOBJD) command will be used to present a display similar to the one on page 5-18 showing the full information about the file. Press F20 to see service (compile time) information about the file.

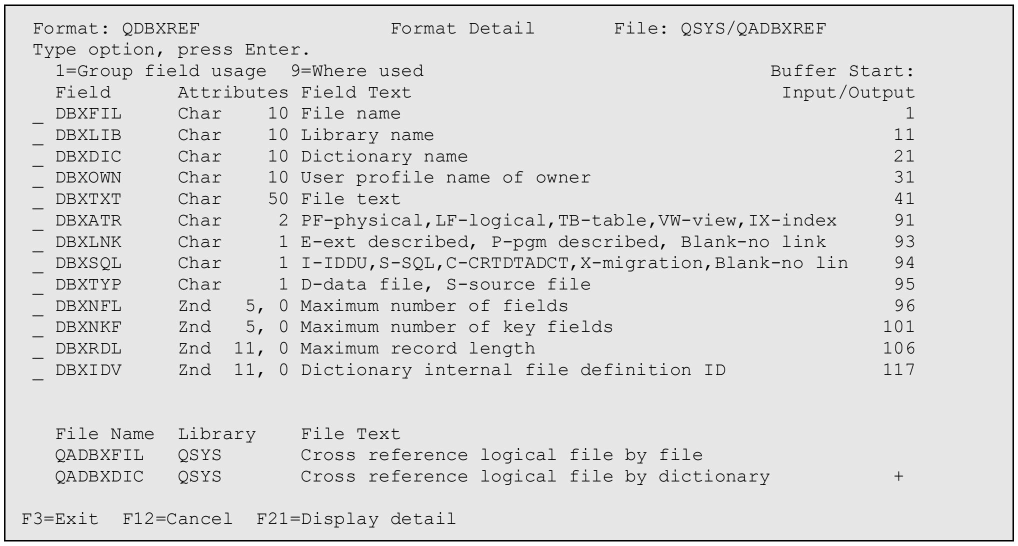
Press F21 to access a display showing the current file attributes. If you have selected a database file (physical or logical) the display will be similar to the formatted file attributes display described beginning on page 5-11. If you have selected a device file, the Display File Description (DSPFD) command will be run and the OS/400 formatted display will appear.

F22 can be used to display the database relationships involving this file. Refer to page 5-8 for an explanation.

Press F3 or F12 to exit the ANZFILE display.

Record Format Description

The display below shows the external definition of the record format. It can be accessed by choosing the 12=Work with option for one of the formats shown on the primary file analysis display.



```
 Format: QDBXREF                Format Detail       File: QSYS/QADBXREF
 Type option, press Enter.
   1=Group field usage  9=Where used                              Buffer Start:
   Field      Attributes Field Text                                Input/Output
 _ DBXFIL     Char    10 File name                                            1
 _ DBXLIB     Char    10 Library name                                        11
 _ DBXDIC     Char    10 Dictionary name                                     21
 _ DBXOWN     Char    10 User profile name of owner                          31
 _ DBXTXT     Char    50 File text                                           41
 _ DBXATR     Char     2 PF-physical,LF-logical,TB-table,VW-view,IX-index    91
 _ DBXLNK     Char     1 E-ext described, P-pgm described, Blank-no link     93
 _ DBXSQL     Char     1 I-IDDU,S-SQL,C-CRTDTADCT,X-migration,Blank-no lin   94
 _ DBXTYP     Char     1 D-data file, S-source file                          95
 _ DBXNFL     Znd   5, 0 Maximum number of fields                            96
 _ DBXNKF     Znd   5, 0 Maximum number of key fields                       101
 _ DBXRDL     Znd  11, 0 Maximum record length                              106
 _ DBXIDV     Znd  11, 0 Dictionary internal file definition ID             117


   File Name  Library    File Text
   QADBXFIL   QSYS       Cross reference logical file by file
   QADBXDIC   QSYS       Cross reference logical file by dictionary          +

F3=Exit  F12=Cancel  F21=Display detail
```

The top portion of the display identifies the file, library, and format you have selected.

Each field in the format will be shown in positional order along with its attributes, length and text, if any. The input buffer starting position for the field is shown at the right margin of the display. Use the roll keys to access additional pages of information.

The lower part of the display shows the files that share this record format description. These files have the same fields (in the same positions) as are listed on the display. Additional files may be viewed by or moving the cursor into the lower section with the arrow or Tab keys and then using the roll keys.

Field referencing options

Select option 1 or option 9 to request information from the ABSTRACT cross reference files about one or more fields on the display.

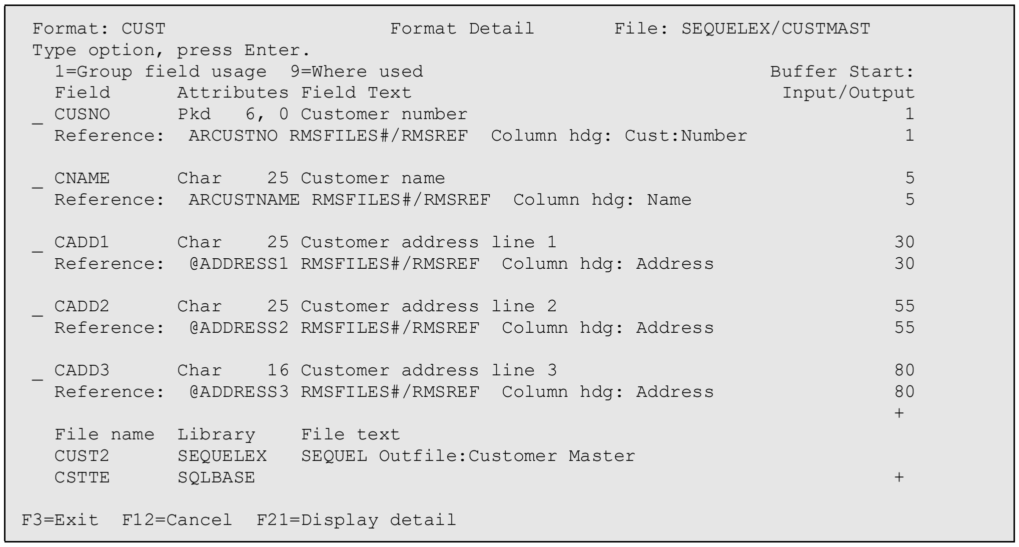
Option 1=Group field usage will run the Work With Field Group Usage (WRKFLDGU) command and present a display showing all usage of the selected field. References through the physical file, logical files built on it, and files overridden to it will be displayed. Refer to the description of the Work With Field Group Usage command for detailed information about this function.

Option 9=Where used can be used to display the references to the selected field through this file only. Unlike the where-used capabilities of option 1, choosing this option restricts the search to references made through the file listed at the top of the display.

Function keys

Use F12 to return to the primary file analysis display, or press F3 to exit the function completely.

F21 can be used to drop the subfile and view additional information about the fields on the display. Column heading, join reference, field rename or concatenation, and field reference information will be presented if it is available. The output buffer starting position appears under the input position. Input and output positions will usually be the same, although they may differ for device files.



```
 Format: CUST                    Format Detail       File: SEQUELEX/CUSTMAST
 Type option, press Enter.
   1=Group field usage  9=Where used                              Buffer Start:
   Field      Attributes Field Text                                Input/Output
 _ CUSNO      Pkd   6, 0 Customer number                                      1
   Reference:  ARCUSTNO RMSFILES#/RMSREF  Column hdg: Cust:Number             1

 _ CNAME      Char    25 Customer name                                        5
   Reference:  ARCUSTNAME RMSFILES#/RMSREF  Column hdg: Name                  5

 _ CADD1      Char    25 Customer address line 1                             30
   Reference:  @ADDRESS1 RMSFILES#/RMSREF  Column hdg: Address               30

 _ CADD2      Char    25 Customer address line 2                             55
   Reference:  @ADDRESS2 RMSFILES#/RMSREF  Column hdg: Address               55

 _ CADD3      Char    16 Customer address line 3                             80
   Reference:  @ADDRESS3 RMSFILES#/RMSREF  Column hdg: Address               80
                                                                              +
   File name  Library    File text
   CUST2      SEQUELEX   SEQUEL Outfile:Customer Master
   CSTTE      SQLBASE                                                         +

F3=Exit  F12=Cancel  F21=Display detail
```

In its dropped form, the display shows extended field information and its output buffer starting position. Field reference information is preceded by the Reference: label and is presented in reference field, library/file form. Column heading information is preceded by a Column hdg: label.

As before, the files that share the format are displayed at the bottom of the display.

Options 1 and 9 can be used as indicated above to reference the field where used functions for fields on the display.

Press F21 again to revert to the previous form of the record format display. Use F12 to return to the primary file analysis display, or press F3 to exit the function completely.

Database Relationships

The database relationships display depicts database structure and access path information. Use it to determine the relationships among the physical and logical database files in your application. Access the display by pressing F22 from the main file analysis display.

In acquiring the database relationship structure, ABSTRACT begins by locating the physical file(s) that contain the data you have referenced through the FILE parameter on the ANZFILE command. If you have selected a logical file, ABSTRACT locates each of its underlying physical files.

Once the physical file(s) are identified, ABSTRACT acquires the current member list for each file, and proceeds to show the logical files (and members) built over each of them. Access path information is also shown for each file represented on the display.

The display below shows an example of the database relationship information that ABSTRACT can provide.

```
 Member: QADBXREF         Database Relations      File  : QSYS/QADBXREF
 8=Access path description  9=Where used

    Physical/
Opt  Logical file  Library     Member      Access Path
 _  QADBXREF       QSYS        QADBXREF    DBXLIB,DBXFIL
 _    QADBXDIC     QSYS        QADBXDIC    DBXDIC,DBXIDV(Sign)
 _    QADBXFIL     QSYS        QADBXFIL    DBXFIL,DBXLIB                 ...
 _    QSQCOLUMNS   SQLBASE     SYSCOLUMNS  Arrival Sequence              ...
 _    QSQTABLES    SQLBASE     SYSTABLES   Arrival Sequence              ...
 _    SYSCOLUMNS   SQLBASE     SYSCOLUMNS  Arrival Sequence
 _    SYSINDEXES   SQLBASE     SYSINDEXES  Arrival Sequence
 _    SYSINDEXES   SQLBASE     SYSINDEXES  Arrival Sequence
 _    SYSKEYS      SQLBASE     SYSKEYS     Arrival Sequence
 _    SYSTABLES    SQLBASE     SYSTABLES   Arrival Sequence
 _    SYSVIEWDEP   SQLBASE     SYSVIEWDEP  Arrival Sequence
 _    SYSVIEWDEP   SQLBASE     SYSVIEWDEP  Arrival Sequence
 _    SYSVIEWS     SQLBASE     SYSVIEWS    Arrival Sequence
 _    TABLE2       SQLBASE     TABLE2      Arrival Sequence
 _    Unused
 _    QSQCOLUMNS   QSYS                    Arrival Sequence              ...
 _    QSQTABLES    QSYS                    Arrival Sequence              ...

F3=Exit  F7/F17 Group field/file usage F12=Previous  F20/F21=Print list
```

The display lists the database structure in which the QADBXREF file participates. Each member of the physical files in the structure will be represented at the leftmost position of the file name column. Logical file members that are built on the physical file member are listed underneath it and indented to the right. Access path information is listed to the right of the file and member name.

Use the roll keys to access subsequent pages of the display.

The example above shows the database structure containing the file named QADBXREF in library QSYS. The QADBXREF member in the file is referenced by each of the thirteen logical files listed underneath it. If the QADBXREF file had multiple members, a similar structure would appear for each of them.

The bottom of the display lists two files under a heading of Unused. These logical files are built over the QADBXREF physical file but do not currently use any of its members. If a logical file member were added to these files using the Add Logical File Member (ADDLFM) command, the DTAMBRS parameter would determine the member(s) of QADBXREF used by the new logical member.

The display also shows that QADBXREF has two key fields (DBXLIB,DBXFIL) which are the same fields used by QSYS/QADBXFIL. The QADBXDIC file also has two key fields, one of which is signed numeric. Many of the files listed are not keyed, as indicated by the Arrival Sequence notation in the access path column. Records can be retrieved from them in arrival sequence only.

Several of the files on the display show an ellipsis (...) at the right. This indicates that the complete access path description is not shown and includes more information than can be presented on a single line. Select the file with option 8=Access path description to see the entire access path definition.

Display options

Option 8=Access path description can be used to view the complete access path definition for the files on the display. Key field definitions and select/omit criteria will be presented on a display similar to the one below.

Use option 9=Where used to find the references to the file you have selected. The Work With X-ref Object Usage (WRKOBJUX) command will be run. All references to the file, both qualified and unqualified, will be shown. Refer to the description for the WRKOBJUX command for details of its function.

Function keys

F7 can be used to access the field group usage for the file named at the top of the display. The Work With Field Group Usage (WRKFLDGU) command will be run and all references to the fields defined in the file, through any physical or logical file shown on the display, will appear. Refer to the description for the WRKFLDGU command for details of its function.

Use F17 to access the file group usage for the file named at the top of the display. The Work With File Group Usage (WRKFGU) command will be run and all references to the database structure shown on the display will appear. Refer to the description for the WRKFGU command for details of its function.

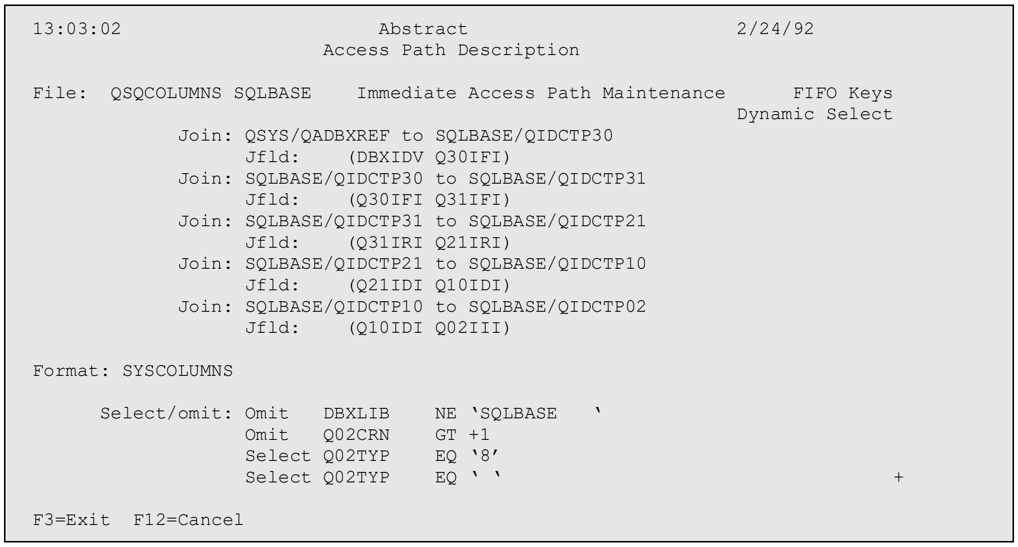
Press F12 to return to the primary file analysis display or use F3 to exit the function altogether.

Use F20 to print out the database relations with extended descriptions if the access path is longer than one line or if select/omit criteria exist.

Press F21 to print the database relations exactly as they appear on the display.

Access Path Description

An expanded information display can be used to show the entire access path and all select/omit criteria for the file. Select this feature by positioning the cursor on one of the files in the database structure display choosing option 8, and pressing the Enter key. A display similar to the one below will appear.



```
13:03:02                     Abstract                        2/24/92
                        Access Path Description

File:  QSQCOLUMNS SQLBASE    Immediate Access Path Maintenance      FIFO Keys
                                                              Dynamic Select
             Join: QSYS/QADBXREF to SQLBASE/QIDCTP30
                   Jfld:    (DBXIDV Q30IFI)
             Join: SQLBASE/QIDCTP30 to SQLBASE/QIDCTP31
                   Jfld:    (Q30IFI Q31IFI)
             Join: SQLBASE/QIDCTP31 to SQLBASE/QIDCTP21
                   Jfld:    (Q31IRI Q21IRI)
             Join: SQLBASE/QIDCTP21 to SQLBASE/QIDCTP10
                   Jfld:    (Q21IDI Q10IDI)
             Join: SQLBASE/QIDCTP10 to SQLBASE/QIDCTP02
                   Jfld:    (Q10IDI Q02III)

Format: SYSCOLUMNS

      Select/omit: Omit   DBXLIB    NE 'SQLBASE   '
                   Omit   Q02CRN    GT +1
                   Select Q02TYP    EQ '8'
                   Select Q02TYP    EQ ' '                               +

F3=Exit  F12=Cancel
```

The display describes the key fields, join specifications, and select/omit criteria that the file uses. Join specifications are presented at the top of the display. Key field and select/omit definitions are presented for each format that includes them.

The display above presents the access path information for the QSQCOLUMNS file in a library named SQLBASE. It is a join logical file with an immediate maintenance access path (vs. delayed or rebuild) that uses dynamic selection. QSQCOLUMNS joins five files together using one pair of fields in each join specification.

The SYSCOLUMNS record format does not include any key definitions (they would have been listed first) but does include select/omit specifications.

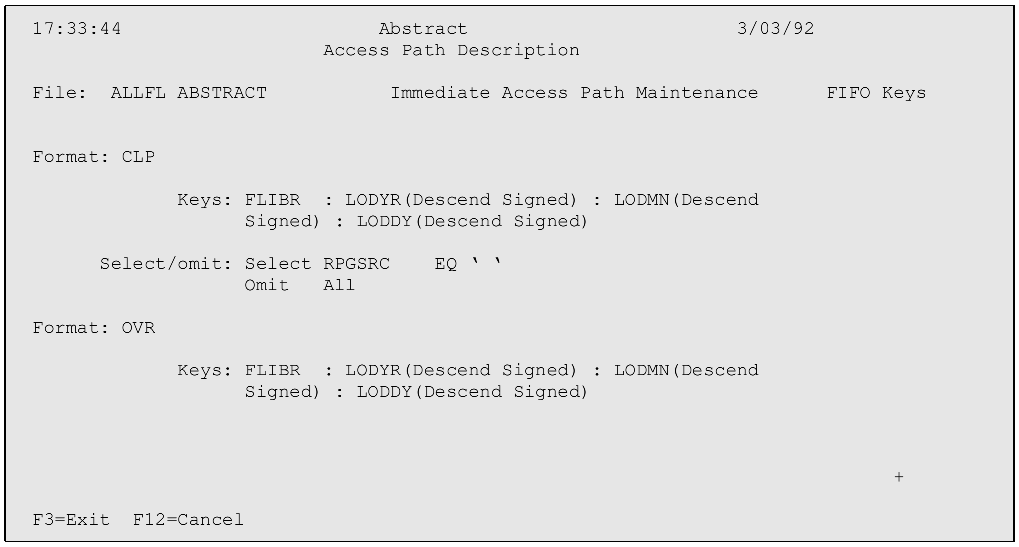
The select/omit criteria for the format are interpreted as follows:

Omit records with field DBXLIB not equal to 'SQLBASE'
Omit records with field Q02CRN greater than 1
Select records with field Q02TYP equal '8' or ' '

If additional pages of information are available, you can access them with the roll keys.

Press F12 to return to the database structure display (page 5-8) or use F3 to exit the file analysis function completely.

The display below demonstrates the extended access path display for another file. It shows key field definitions for two formats (note the Descending and Signed attributes) and describes the select/omit criteria on the format named CLP.

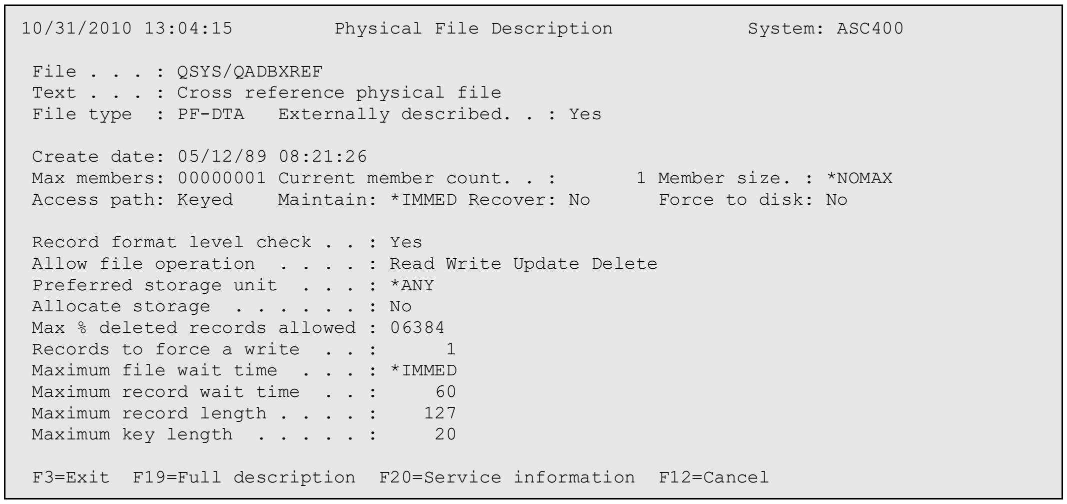


```
17:33:44                      Abstract                        3/03/92
                          Access Path Description

File:  ALLFL ABSTRACT           Immediate Access Path Maintenance      FIFO Keys


Format: CLP

             Keys: FLIBR  : LODYR(Descend Signed) : LODMN(Descend
                   Signed) : LODDY(Descend Signed)

      Select/omit: Select RPGSRC    EQ ' '
                   Omit   All

Format: OVR

             Keys: FLIBR  : LODYR(Descend Signed) : LODMN(Descend
                   Signed) : LODDY(Descend Signed)



                                                                        +

F3=Exit  F12=Cancel
```

File Attributes

The file attributes display provides file level information about the file you have selected for analysis. It is accessed by pressing F21 from the primary file analysis display.

The content of the display depends on the type of file chosen. Displays for physical and logical files will be similar to the one below, showing the most important attributes of the file. Refer to the appropriate create command (CRTPF, CRTLF) for a complete description of the information on the display.



```
10/31/2010 13:04:15         Physical File Description           System: ASC400

 File . . . : QSYS/QADBXREF
 Text . . . : Cross reference physical file
 File type  : PF-DTA   Externally described. . : Yes

 Create date: 05/12/89 08:21:26
 Max members: 00000001 Current member count. . :       1 Member size. : *NOMAX
 Access path: Keyed    Maintain: *IMMED Recover: No      Force to disk: No

 Record format level check . . : Yes
 Allow file operation  . . . . : Read Write Update Delete
 Preferred storage unit  . . . : *ANY
 Allocate storage  . . . . . . : No
 Max % deleted records allowed : 06384
 Records to force a write  . . :      1
 Maximum file wait time  . . . : *IMMED
 Maximum record wait time  . . :     60
 Maximum record length . . . . :    127
 Maximum key length  . . . . . :     20

 F3=Exit  F19=Full description  F20=Service information  F12=Cancel
```

If you have selected a device file (display, printer, communications, etc.), F21 from the primary analysis display will not provide a display like the one above. Instead, it will present the result of the Display File Description (DSPFD) command.

Two types of object descriptive information can be obtained from the formatted attributes display above: object descriptive information and the file description.

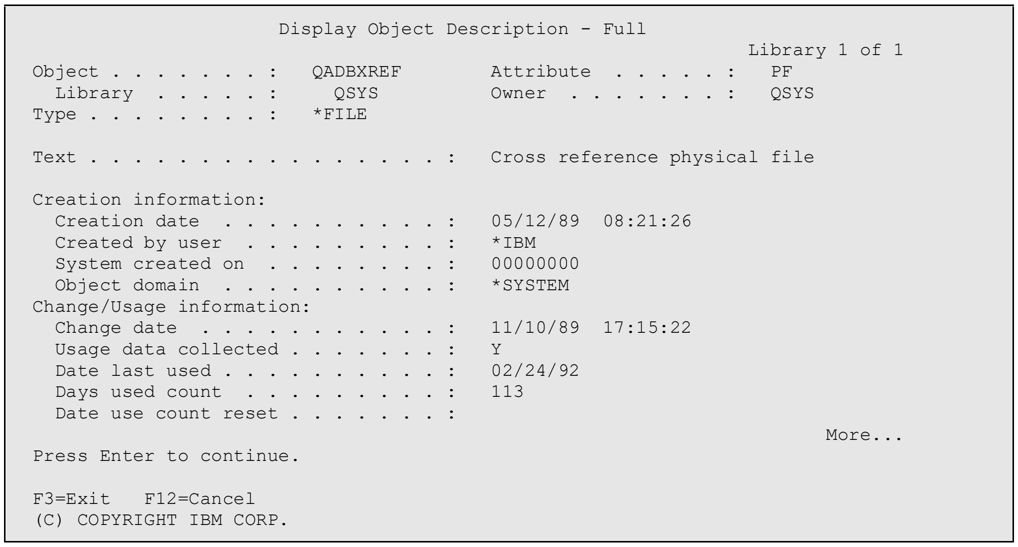
Use F19 to view the full object description. The Display Object Description (DSPOBJD) command will be used to present a display similar to the one below. It shows the full object description information about the file. Press F20 to see service (compile time) information about the file.

Press F21 to request output from the Display File Description (DSPFD) command for this file. The standard OS/400 output will be presented on the display.

Press F12 to return to the initial file analysis display or use F3 to exit the file analysis function.

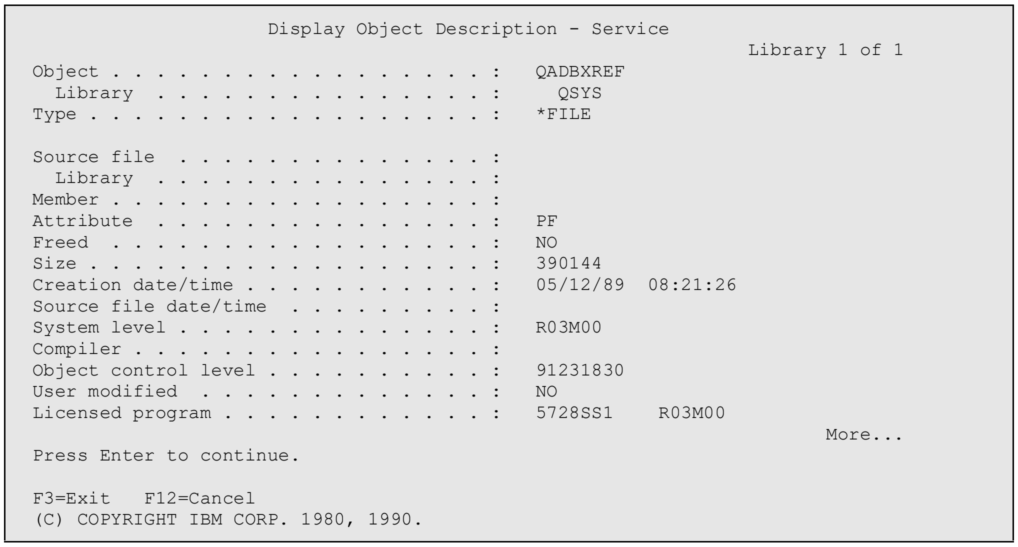
Object Description

Two object description displays can be accessed for the file you have selected. F19 from either the initial file analysis display (page 5-4) or from the file attributes display above will present the full object description. This display, similar to the one on the next page, provides creation, change/usage, storage, and save/restore information about the file.



```
                      Display Object Description - Full
                                                          Library 1 of 1
Object . . . . . . . :   QADBXREF        Attribute  . . . . . :   PF
  Library  . . . . . :     QSYS          Owner  . . . . . . . :   QSYS
Type . . . . . . . . :   *FILE

Text . . . . . . . . . . . . . . . . :   Cross reference physical file

Creation information:
  Creation date  . . . . . . . . . . :   05/12/89  08:21:26
  Created by user  . . . . . . . . . :   *IBM
  System created on  . . . . . . . . :   00000000
  Object domain  . . . . . . . . . . :   *SYSTEM
Change/Usage information:
  Change date  . . . . . . . . . . . :   11/10/89  17:15:22
  Usage data collected . . . . . . . :   Y
  Date last used . . . . . . . . . . :   02/24/92
  Days used count  . . . . . . . . . :   113
  Date use count reset . . . . . . . :
                                                                 More...
Press Enter to continue.

F3=Exit   F12=Cancel
(C) COPYRIGHT IBM CORP.
```

The object service information, accessed using F20 from the initial display or the file attributes panel, provides the compile-time attributes of the file object. Source file, member, change date, and compiler information are provided as shown below:



```
                     Display Object Description - Service
                                                          Library 1 of 1
Object . . . . . . . . . . . . . . . . :   QADBXREF
  Library  . . . . . . . . . . . . . . :     QSYS
Type . . . . . . . . . . . . . . . . . :   *FILE

Source file  . . . . . . . . . . . . . :
  Library  . . . . . . . . . . . . . . :
Member . . . . . . . . . . . . . . . . :
Attribute  . . . . . . . . . . . . . . :   PF
Freed  . . . . . . . . . . . . . . . . :   NO
Size . . . . . . . . . . . . . . . . . :   390144
Creation date/time . . . . . . . . . . :   05/12/89  08:21:26
Source file date/time  . . . . . . . . :
System level . . . . . . . . . . . . . :   R03M00
Compiler . . . . . . . . . . . . . . . :
Object control level . . . . . . . . . :   91231830
User modified  . . . . . . . . . . . . :   NO
Licensed program . . . . . . . . . . . :   5728SS1    R03M00
                                                                 More...
Press Enter to continue.

F3=Exit   F12=Cancel
(C) COPYRIGHT IBM CORP. 1980, 1990.
```

Refer to the CL Reference Manual - Volume 4 for a complete description of the Display Object Description (DSPOBJD) command and its output.

Exit these displays by pressing F12 or F3. F12 will return to the previous display, F3 will exit the file analysis function.

External Record Layouts

This function provides a printed representation of the external record format descriptions for database files. Both physical and logical files can be documented. No cross references are needed as the information is accessed from the file(s) when your request is made.

Two types of reports are available:

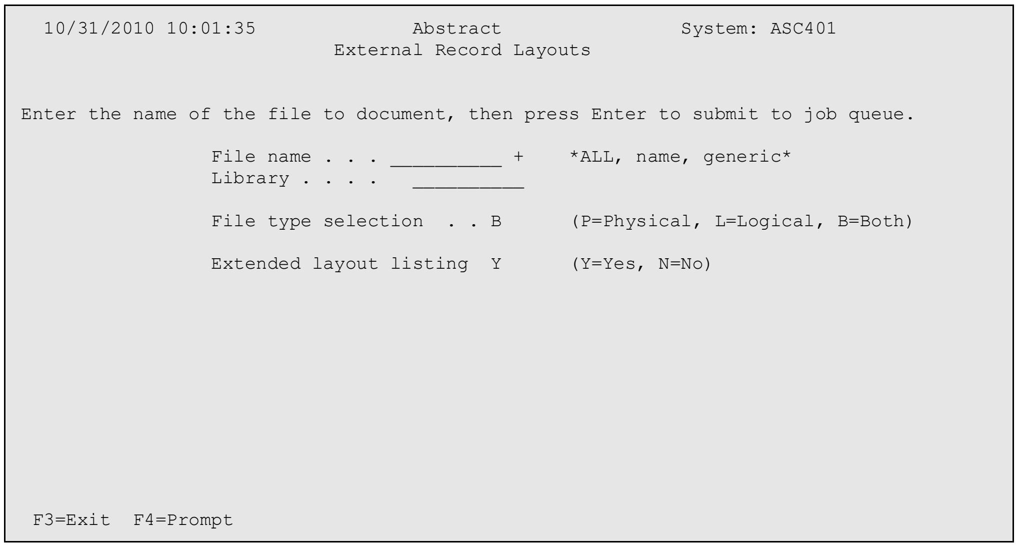
The *extended* listing will detail all reference, editing, column heading and text for each field.

The *abbreviated* report will list only field names, attributes and text for the fields in the chosen files.

Using the initial prompt display presented when the menu option is selected, you can choose to print record layout descriptions for one, several, or all of the files in a particular library.

All record layout print requests will be submitted to the batch job queue for execution. Your current job description, accessed using F18 from the menu, will be used to submit the job.

Request the record layout print function by selecting option 2 of the file analysis (APFILE) menu. A display similar to the one below will appear.


```
  10/31/2010 10:01:35              Abstract                System: ASC401
                            External Record Layouts


Enter the name of the file to document, then press Enter to submit to job queue.

                 File name . . . __________ +    *ALL, name, generic*
                 Library . . . .   __________

                 File type selection  . . B      (P=Physical, L=Logical, B=Both)

                 Extended layout listing  Y      (Y=Yes, N=No)











 F3=Exit  F4=Prompt
```

Use this prompt to specify the files you want to document. It allows you specify three types of information: file specification, qualifier, and listing request.

File

Type the name and library of the file you want to document. If you want to include all files in the library, type *ALL. You may generate a list of files and select items from it by specifying a generic file name (e.g. CUST*). The list will be similar to the one on page 5-15.

Type qualifier

Choose whether you want to document physical files, logical files, or both by placing a P, L, or B next to the file type selector on the display.

Listing request

Identify whether you want the abbreviated (field name, type, and text) printout or the longer, more complete report by specifying Y or N in the extended layout list prompt.

Function Keys

Make your request and press Enter to submit the request to the batch subsystem. If you have selected a generic file name, or pressed F4, an intermediate display will allow you to choose the specific files you want documented.

Press F4 to request a prompt that displays all of the files meeting the name/type criteria you have selected. You will be able to select individual files from the list.

Use F3 to exit the function and return to the APFILE menu without submitting any requests.

File selection prompt

The prompt on the next page will appear if you have pressed F4 from the initial display, or if you have specified a generic file name request. It allows you to indicate which of the files meeting your criteria should be documented.

The display lists all the files meeting the name and type (physical, logical, both) specification you have selected.

Select files on the display by placing a 1 next to them. Once the files you want to document have been selected, press the Enter key to submit the request to the batch job queue.

```
                              File Selection
 Type option, then press Enter.
   1=Select
           File Name        File Name        File Name        File Name
       _   QACJINFO     _   QADSPFLR     _   QAEZUJOB     _   QAFDSELO
       _   QADALO       _   QADSPJRN     _   QAFDACCP     _   QAFDSPOL
       _   QADBFDEP     _   QADSPJR2     _   QAFDBASI     _   QAFDTAP
       _   QADBLDEP     _   QADSPJR3     _   QAFDBSC      _   QAIFTRCF
       _   QADBLDNC     _   QADSPOBJ     _   QAFDCMN      _   QAJBACG
       _   QADBLPKG     _   QADSPPGM     _   QAFDCSEQ     _   QANFDNTF
       _   QADBPKG      _   QADSPPTF     _   QAFDDDM      _   QAOBJAUT
       _   QADBXDIC     _   QADSPUPA     _   QAFDDKT      _   QAOJSAVO
       _   QADBXFIL     _   QADSPUPB     _   QAFDDSP      _   QAOPOUFL
       _   QADBXRDBD    _   QADSPUPO     _   QAFDICF      _   QAOSDIRB
       _   QADBXREF     _   QADXERLG     _   QAFDJOIN     _   QAOSDIRF
       _   QADBXRMTNM   _   QADXJRNL     _   QAFDLGL      _   QAOSDIRO
       _   QADPGMAD     _   QADXXRC1     _   QAFDMBR      _   QAOSDIRX
       _   QADSHLR      _   QADXXRC2     _   QAFDMBRL     _   QAOSDSTO
       _   QADSPDBR     _   QAEZOUTQ     _   QAFDPHY      _   QAOSILIN
       _   QADSPDE      _   QAEZRCVA     _   QAFDPRT      _   QAOSILOT
       _   QADSPDOC     _   QAEZRCVD     _   QAFDRFMT     _   QAOSIQDL
       _   QADSPFFD     _   QAEZSTS      _   QAFDSAV      _   QAOSIRCV        +

  F3=Exit  F12=Cancel
```

If you want to exit the display without making a layout request, press F12 or F3. Use F12 to return to the previous display and choose a different file set. Press F3 to exit the record layout request prompt and return to the menu without submitting a request.

Record Layout Output

The output created by the record layout print request is designed to be printed on 8.5x11 paper at 10 characters per inch. You can change the length of the form by specifying appropriate LPI, PAGE-SIZE and OVRFLW values on a Change Printer File (CHGPRTF) command. To make the change to all the printer files used by ABSTRACT issue the command:

```
   CHGPRTF ABSTRACT/*ALL LPI( ) PAGESIZE( ) OVRFLW( )
```

There are two parts to the output from each record layout print request. A table of contents, listing each file documented and its page number, precedes the record layout definitions for the files you have selected. It looks like the sample at the top of the next page.

The record layout information following the table of contents presents the file and field information you have requested. File level information depends on the file type (physical/logical) and describes file and format text, access path information, and select/omit criteria. Refer to the example on the next page.

Table of contents

```
17:11:00               Abstract
 10/31/2010         Record Format Description
                      Table Of Contents

--------------------------------------------------------------------------------

 Page File name       Format   File Description

  1. QSYS/QADBFDEP     QDBFDEP  Cross reference dependency file
  2. QSYS/QADBLDEP     QDBFDEP  Dependency logical file by dependency
  3. QSYS/QADBLDNC     QDBFDEP  Dependency logical multiple-format file
  4.                   QDBNC2
  5. QSYS/QADBLPKG     QDBPKG   SQL Package logical file
  6. QSYS/QADBPKG      QDBPKG   SQL Package physical file
  7. QSYS/QADBXDIC     QDBXREF  Cross reference logical file by dictionar
  8. QSYS/QADBXFIL     QDBXREF  Cross reference logical file by file
  9. QSYS/QADBXRDBD    QDBXRDBD  RDB Directory physical file
 10. QSYS/QADBXREF     QDBXREF  Cross reference physical file
 11. QSYS/QADBXRMTNM   QDBXRDBD  RDB Directory logical file
```

Record Layout Information

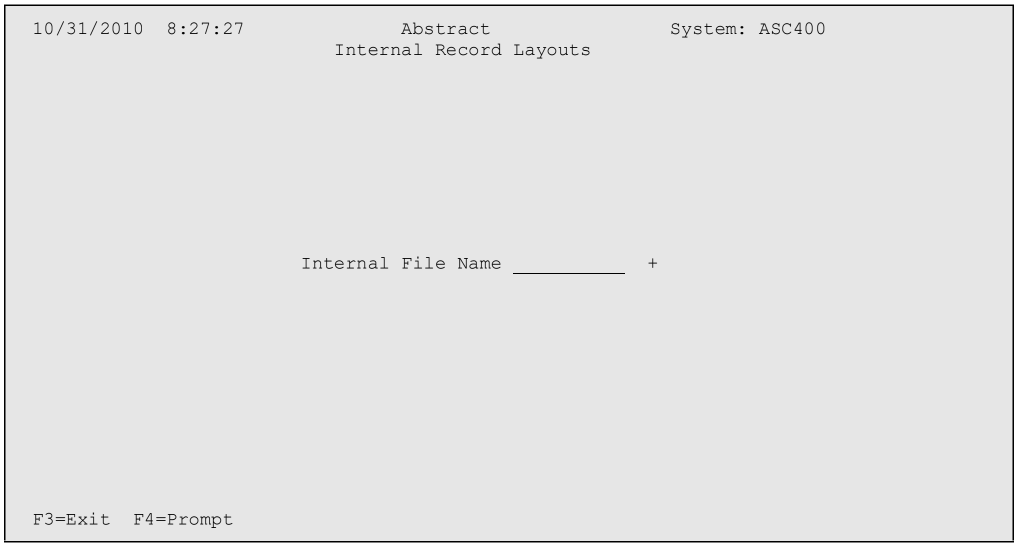
```
17:11:00                          Abstract                             Page  8
 10/31/2010                  Record Format Description

Log. File: QSYS/QADBXFIL     Format: QDBXREF  Over: QSYS/QADBXREF
Shares format: QSYS/QADBXREF
File Text: Cross reference logical file by file

Format Text: Cross reference file

Immediate Access Path Maintenance

FIFO key(s): DBXFIL : DBXLIB

Dynamic Select/Omit Testing
Select/Omit: Omit  DBXFIL  EQ ‛    ‛

---------------------------------------------------------------------------------

Field   Descriptions                                      Type Length Start Stop

DBXFIL   File name                                        Char    10      1   10
         Column Heading: FILE:NAME

DBXLIB   Library name                                     Char    10     11   20
         Column Heading: LIBRARY:NAME

DBXDIC   Dictionary name                                  Char    10     21   30
         Column Heading: DICTIONARY:NAME

DBXOWN   User profile name of owner                       Char    10     31   40
         Column Heading: FILE:OWNER

DBXTXT   File text                                        Char    50     41   90
         Column Heading: FILE:TEXT

DBXATR   PF-physical,LF-logical,TB-table,VW-view,IX-ind    Char     2     91   92
         Column Heading: FILE:ATTR

DBXLNK   E-ext described, P-pgm described, Blank-no lin    Char     1     93   93
         Column Heading: LINK:STATUS

DBXSQL   I-IDDU,S-SQL,C-CRTDTADCT,X-migration,Blank-no     Char     1     94   94
         Column Heading: DICTIONARY:TYPE

DBXTYP   D-data file, S-source file                       Char     1     95   95
         Column Heading: FILE:TYPE

DBXNFL   Maximum number of fields                         Znd   5, 0     96  100
         Column Heading: NBR:OF:FLDS

DBXNKF   Maximum number of key fields                     Znd   5, 0    101  105
         Column Heading: NBR:KEY:FLDS

DBXRDL   Maximum record length                            Znd  11, 0    106  116
         Column Heading: RECORD:LENGTH
```

Internal Record Layouts

The internal record layout function ABSTRACT provides information about your application's use and definition of *program described* files. If programs in your application use record or field definitions that are different from the external definitions in the database, ABSTRACT will acquire and store the program descriptions in its dictionary.

The displays that describe the program described data in your application are especially useful because they make it easy to locate the programs that use the data, and document inconsistent descriptions among them.

You can access the information about your program described data by selecting option 3 from the APFILE menu or by using the Work With Internal Record definitions (WRKINTREC) command. A display similar to the one below will appear. You will use it to indicate the program definitions that you are interested in displaying.

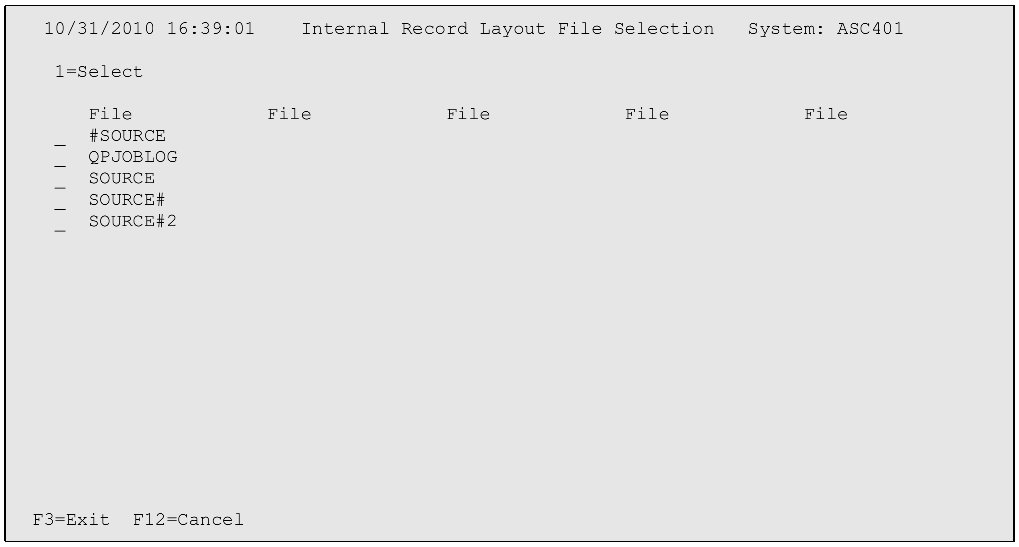

```
 10/31/2010  8:27:27             Abstract                System: ASC400
                             Internal Record Layouts









                          Internal File Name __________  +












 F3=Exit  F4=Prompt
```

Use this display to indicate the name of the file you wish to display. Type the program described file name and press Enter, or press F4 to acquire a list of available definitions. If you elect to specify a file name, a display similar to the one on page 5-20 will appear.

The file names used by this option do not necessarily correspond to physical file names which may exist on your system, but rather to the names declared by the program source code. You will find this option most useful if the names declared by your programs are consistent - always referencing a specific file with a given name.

Press F3 to end the program described layout display.

If F4 is pressed from the previous display, all the currently known program described file names will be shown. A display similar to the one below will appear.



```
 10/31/2010 16:39:01    Internal Record Layout File Selection   System: ASC401

  1=Select

     File            File            File            File            File
  _  #SOURCE
  _  QPJOBLOG
  _  SOURCE
  _  SOURCE#
  _  SOURCE#2















F3=Exit  F12=Cancel
```

The display lists each program described file name cataloged during the source analysis phase of the initialization process.

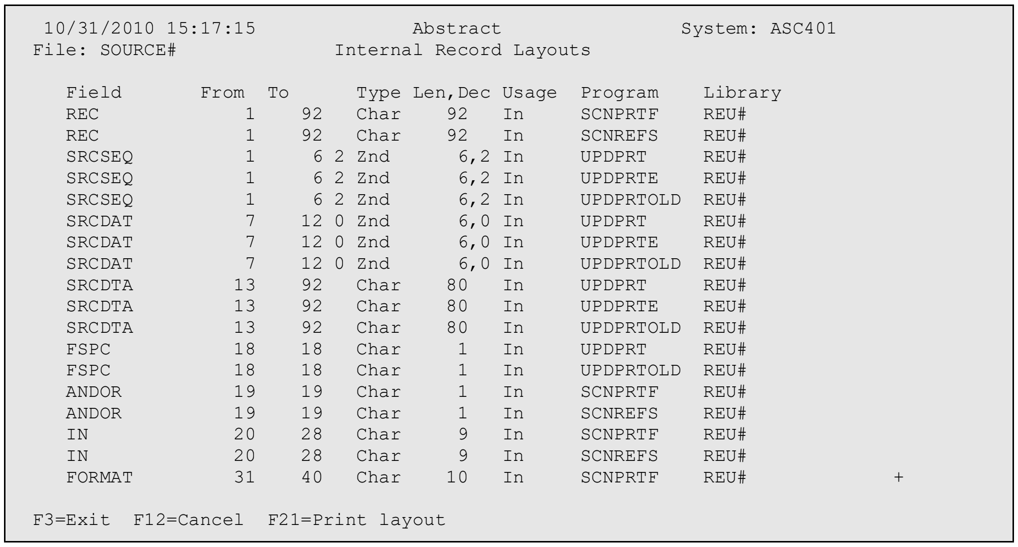
Select one of the names on the display by typing a ‘1’ next to it and pressing the Enter key. A display similar to the one on the following page will appear showing all of the program described definitions for the selected file.

Press F3 to exit the internal record layout function, or use F12 to return to the previous display.

Internal record layout display

The following display depicts the program described record layout descriptions for the indicated file. All program specifications which use this file name are merged onto the display.

Fields are displayed in positional order. Buffer positions as well as length and attributes are displayed for each field. The program which provides the description is shown at the right side of the display, and its use of the field is also indicated.



```
 10/31/2010 15:17:15              Abstract                System: ASC401
File: SOURCE#              Internal Record Layouts

   Field       From  To      Type Len,Dec Usage  Program    Library
   REC            1    92    Char    92   In     SCNPRTF    REU#
   REC            1    92    Char    92   In     SCNREFS    REU#
   SRCSEQ         1     6 2  Znd      6,2 In     UPDPRT     REU#
   SRCSEQ         1     6 2  Znd      6,2 In     UPDPRTE    REU#
   SRCSEQ         1     6 2  Znd      6,2 In     UPDPRTOLD  REU#
   SRCDAT         7    12 0  Znd      6,0 In     UPDPRT     REU#
   SRCDAT         7    12 0  Znd      6,0 In     UPDPRTE    REU#
   SRCDAT         7    12 0  Znd      6,0 In     UPDPRTOLD  REU#
   SRCDTA        13    92    Char    80   In     UPDPRT     REU#
   SRCDTA        13    92    Char    80   In     UPDPRTE    REU#
   SRCDTA        13    92    Char    80   In     UPDPRTOLD  REU#
   FSPC          18    18    Char     1   In     UPDPRT     REU#
   FSPC          18    18    Char     1   In     UPDPRTOLD  REU#
   ANDOR         19    19    Char     1   In     SCNPRTF    REU#
   ANDOR         19    19    Char     1   In     SCNREFS    REU#
   IN            20    28    Char     9   In     SCNPRTF    REU#
   IN            20    28    Char     9   In     SCNREFS    REU#
   FORMAT        31    40    Char    10   In     SCNPRTF    REU#             +

F3=Exit  F12=Cancel  F21=Print layout
```

ABSTRACT attempts to point out possible conflicts in definition of the record by displaying apparent conflicts in high intensity. Overlapping fields will have buffer positions highlighted and conflicting field attributes (Ex.: Character and Zoned) will have the attributes or decimal positions highlighted.

ABSTRACT displays up to ten characters of each variable name. Extended COBOL data names will be printed if F21 is pressed. COBOL group level data names are displayed along with elementary items. Group items have attributes 'GRP'.

F21 may be pressed to request a printout of the program described record layout for this file. The printed version is similar to the display except that the entire field name is shown for COBOL fields (up to 30 characters). Field conflicts are annotated on the printout with asterisks. Refer to the description of the Print Internal Record Layout (PRTINTREC) command for complete information about the printed form of the output.

Press F12 to return to the previous display and to select another file. Use function key to exit the function and return to the menu.

Printed Record Layout

The output on the next page shows an example of the report obtained by pressing F21. The report is produced interactively and is equivalent to using the Print Internal Record Layouts (PRTINTREC) command (or APFILE menu option 4) and specifying the file you selected on the primary display above.

```
15:18:38                               Abstract                        Page: 1
10/31/2010                     Internal File Record Layouts                 File:SOURCE#

File    Field   From To  Dec Type Len,Dec Usage Program   Library  Long field name(COBOL)

SOURCE#

        REC        1  92     Char    92   In    SCNPRTF   REU#
        REC        1  92     Char    92   In    SCNREFS   REU#
     ** SRCSEQ     1   6  2  Znd      6,2 In    UPDPRT    REU#
        SRCSEQ     1   6  2  Znd      6,2 In    UPDPRTE   REU#
        SRCSEQ     1   6  2  Znd      6,2 In    UPDPRTOLD REU#
        SRCDAT     7  12  0  Znd      6,0 In    UPDPRT    REU#
        SRCDAT     7  12  0  Znd      6,0 In    UPDPRTE   REU#
        SRCDAT     7  12  0  Znd      6,0 In    UPDPRTOLD REU#
        SRCDTA    13  92     Char    80   In    UPDPRT    REU#
        SRCDTA    13  92     Char    80   In    UPDPRTE   REU#
        SRCDTA    13  92     Char    80   In    UPDPRTOLD REU#
     ** FSPC      18  18     Char     1   In    UPDPRT    REU#
        FSPC      18  18     Char     1   In    UPDPRTOLD REU#
        ANDOR     19  19     Char     1   In    SCNPRTF   REU#
        ANDOR     19  19     Char     1   In    SCNREFS   REU#
        IN        20  28     Char     9   In    SCNPRTF   REU#
        IN        20  28     Char     9   In    SCNREFS   REU#
        FORMAT    31  40     Char    10   In    SCNPRTF   REU#
        FORMAT    31  40     Char    10   In    SCNREFS   REU#
        NAME      31  40     Char    10   In    SCNPRTF   REU#
        NAME      31  40     Char    10   In    SCNREFS   REU#
        REF       41  41     Char     1   In    SCNPRTF   REU#
        REF       41  41     Char     1   In    SCNREFS   REU#
        LEN       42  46     Char     5   In    SCNPRTF   REU#
        LEN       42  46     Char     5   In    SCNREFS   REU#
        TYPE      47  47     Char     1   In    SCNPRTF   REU#
        TYPE      47  47     Char     1   In    SCNREFS   REU#
        DEC       48  49     Char     2   In    SCNPRTF   REU#
        DEC       48  49     Char     2   In    SCNREFS   REU#
        LINE      51  53     Char     3   In    SCNPRTF   REU#
        LINE      51  53     Char     3   In    SCNREFS   REU#
     ** DEVICE    52  58     Char     7   In    UPDPRT    REU#
        DEVICE    52  58     Char     7   In    UPDPRTOLD REU#
     ** POS       54  56     Char     3   In    SCNPRTF   REU#
        POS       54  56     Char     3   In    SCNREFS   REU#
        FNC       57  92     Char    36   In    SCNREFS   REU#
        FUNCT     57  92     Char    36   In    SCNPRTF   REU#
        FUNCTN    57  92     Char    36   In    SCNREFS   REU#
```

The report shows all the program described definitions for each reference to the selected file name. The report is produced by the Print Internal Record Layout (PRTINTREC) command.

Print Printer File Layouts (PRTPRTF) Command

The Print Printer File Layout (PRTPRTF) command prints a representation of the output that can be obtained from either RPG-described or external, DDS described printer files. To do so, it analyzes the current source definition for the printout and creates a simulation of the output it defines.

This function can be accessed using option 5 from the file analysis (APFILE) menu, or by using the command from a command entry prompt. The syntax for the command is shown below:

```
                       .-*LIBL/--------.                    .-*ALL-.
>>--PRTPRTF----SRCFILE-+---------------+---file_name---MBR-+------+--------------------->
                       '-library_name/-'                    '-name-'

                                                                                Required
==========================================================================================
                                                                                Optional

           .-*NO--.          .-*PRV---.
>--DSPFLDD-+------+---DTASET-+--------+----------------------------------------------><
           '-*YES-'          +-*FIRST-+
                             '-name---'
```

SRCFILE Parameter

Indicates the source file to be examined. The source file you specify will be searched for the member listed by the MBR parameter.

***LIBL:** your current library list will be searched for the source file.

MBR Parameter

Identifies the member(s) to be examined in the indicated source file. The member must contain DDS (for externally described print files) or RPG source code. Name a specific member or use the *ALL special value.

***ALL:** Analyze all source members in the specified file.

DSPFLDD Parameter

Display descriptions for each field in the report.

***NO:** do not display field descriptions.

***YES:** The fields used on the report will follow its printed representation. Field name, usage, length, position, and text will be indicated.

DTASET Parameter

Specify the cross reference data set to be used in acquiring the information.

***PRV:** Use the value for this parameter from the previous session. If no previous session has occurred, use *FIRST.

***FIRST:** information from the first (default) data set should be accessed.

Name: Specify any existing data set. Use the command prompter to find the names available to you.

Examples

```
PRTPRTF SRCFILE(ABSTRACT#/QRPGSRC) MBR(FDT100)
```

This command will locate the FDT100 member in the ABSTRACT#/QRPGSRC source file, analyze its output specifications, and print a report layout description. No field descriptions will be included on the report.

```
PRTPRTF SRCFILE(QDDSSRC) MBR(*ALL) DSPFLDD(*YES)
```

This command will print report layout descriptions for all RPG or DDS printer files in the first QDDSSRC file on the library list. Field descriptions will be included on the report.

The PRTPRTF Report

The printed form of the report will represent program fields as a series of 9's if numeric, or X's if alphanumeric. Undefined fields will be represented as a single '?' on the report. If conditioned or unconditioned overlap occurs in the output specifications, the overlapping information will be overprinted. Sample output is shown below.

```
 Program- ABSTRACT#/QRPGSRC     FDT100                        Abstract                                    Page    1
 16:35:48                                              Printer Spacing Chart                            10/31/2001

1.......10........20........30........40........50........60........70........80.....    ....120.......130..

Forms Length- 66--Overflow Line- 60---------------------------QPRINT  ---------------    -Record Length-  85

  99:99:99                             Abstract
  99/99/99                     Record Format Description
                                   Table Of Contents

  XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

   Page  File name             Format     File Description

  9,999. XXXXXXXXXXXXXXXXXXXXX XXXXXXXXXX XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
 Program- ABSTRACT#/QRPGSRC     FDT100                        Abstract                             Page    2
 16:35:48                                              Printer Spacing Chart                      10/31/2010

1.......10........20........30........40........50........60........70........80.....    ....120.......130..

Forms Length- 66--Overflow Line- 60---------------------------QSYSPRT---------------------Record Length-  85

  99:99:99                           Abstract                           Page Z999
  99/99/99                    Record Format Description


  Log. File:
  Phy. File: XXXXXXXXXXXXXXXXXXXXX  Format: XXXXXXXXXX Over: XXXXXXXXXXXXXXXXXXXXX
  Shares format: XXXXXXXXXXXXXXXXXXXXX
  File Text: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

  Format Text: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
  Created From: XXXXXXXXXXXXXXXXXXXXX XXXXXXXXXX

  Join Files:  XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
  JFld:        XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
  JDupseq:     XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

  Immediate Access Path Maintenance
  Delayed Access Path Maintenance
  Arrival Sequence Order
  LIFO key(s):
  FIFO key(s):
  Unique key(s):

               XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

  Dynamic Select/Omit Testing
  Select/Omit: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
  XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

  Field      Descriptions                               Type Length  Start Stop

  XXXXXXXXXX XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX XXXX XXXXX   9999099990
             Reference: XXXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXXX
             Column Heading: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
             Edit code: X  Edit extender: X
             Edit word: XXXXXXXXXXXXXXXXXXXX
             XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
             Alias: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
             Jfield: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
```

The example output shown above depicts the output specifications of an RPG program. Its corresponding field descriptions are listed below. The source member containing the specifications is named at the top of the output (ABSTRACT#/QRPGSRC FDT100). The RPG program includes two printer files, QPRINT and QSYSPRT. Specifications for the two files are listed separately. The ruler line and the form description line at the top of the report are not part of the program's output, rather they are placed there by ABSTRACT as an aid to the interpretation and use of the report.

Field Descriptions

The field descriptions for the items on the report will be printed if DSPFLDD(*YES) is specified when the command is run. As with the layout chart, the field descriptions are printed for each printer file referenced by the program.

The field descriptions corresponding to the layout examples on the previous page are presented below:

```
 Program- ABSTRACT#/QRPGSRC     FDT100                   Abstract                           Page    3
 16:35:48                                          Printer Spacing Chart                    10/31/2010

1.......10........20........30........40........50........60........70........80.....    ....120.......130..

Forms Length- 66--Overflow Line- 60---------------------------QPRINT  ---------------    -Record Length-  85


Indicators Field  Len   End Edtcde/Edtwrd
           TYME     6 0  10 ‘  :  :  ‘
                         45 ‘Abstract’
           UDATE    6 0  10 Y
                         53 ‘Record Format Descriptio’
                         54 ‘n’
                         50 ‘Table Of Contents’
           DASH    81    83
                          7 ‘Page’
                         18 ‘File name’
                         37 ‘Format’
                         58 ‘File Description’
           TCPAGE   4 0   8 ‘ ,   .’
       15  FILE    21    30
           FMTNM   10    41
       12  FTEXT1  41    83
 Program- ABSTRACT#/QRPGSRC     FDT100                        Abstract                              Page    3
 16:35:48                                              Printer Spacing Chart                        10/31/2010

1.......10........20........30........40........50........60........70........80.....    ....120.......130..

Forms Length- 66--Overflow Line- 60--------------------------QSYSPRT----------------    -Record Length-  85

Indicators Field  Len   End Edtcde/Edtwrd
           TYME     6 0  10 '  :  :  '
                         45 'Abstract'
                         78 'Page'
           PAGE     4 0  83 Z
           UDATE    6 0  10 Y
                         53 'Record Format Descriptio'
                         54 'n'
       10                12 'Phy. File:'
      N10                12 'Log. File:'
           FILE    21    34
                         43 'Format:'
           FMTNM   10    54
                         12 'File Text:'
           FTEXT   50    63
                         14 'Format Text:'
           FMTTXT  50    65
                         15 'Created From:'
           SRCFIL  21    37
           FSRCM   10    48
                         13 'Join Files:'
           OUTDS   60    75
                          7 'JFld:'
           OUTDS   60    75
                         10 'JDupseq:'
           OUTDS   60    75
       20                24 'Arrival Sequence Order'
       22                23 'Immediate Access Path'
       22                35 ' Maintenance'
       21                21 'Rebuild Access Path'
       23                21 'Delayed Access Path'
   N20N22                33 ' Maintenance'
    22 21                16 'Unique key(s):'
    23 21                14 'LIFO key(s):'
N22N23 21                14 'FIFO key(s):'
           OUTDS   60    75
       21                22 'Dynamic Select/Omit '
       21                29 'Testing'
       21                14 'Select/Omit:'
           OUTDS   60    75
           DASH    81    83
                          7 'Field'
                         25 'Descriptions'
                         64 'Type'
                         71 'Length'
                         78 'Start'
                         83 'Stop'
           FLDNAM  10    12
           FLDTXT  46    59
           FLDLNC   5    70
           FLDSTR   4 0  78 '    0'
           FLDSTP   4 0  83 '    0'
           FLDTYP   4    64
                         23 'Reference:'
           REFFLD  21    45
           REFFIL  21    67
                         28 'Column Heading:'
           COLHDG  53    82
                         23 'Edit code:'
           FLDEC1   1    25
       24                41 'Edit extender:'
       24  FLDEC2   1    43
                         23 'Edit word:'
           FLDEDW  20    44
           OUTDS   60    73
                         19 'Alias:'
           FALIAS  30    50
                         20 'Jfield:'
           OUTDS   60    81
```

Print Display File Layout (PRTDSPF) Command

The Print Display File Layout (PRTDSPF) command analyzes externally described display files (member type DSPF) and prints a representation of the formats they contain. Like its printer layout counterpart, the PRTDSPF function analyzes the DDS source code for the display file to create a simulation of the file's output.

The print display layout function can be accessed using option 6 from the file analysis (APFILE) menu, or by using the command from a command entry prompt. The syntax for the command is shown below:

```
                       .-*LIBL/--------.                    .-*ALL-.
>>--PRTDSPF----SRCFILE-+---------------+---file_name---MBR-+------+--------------------->
                       '-library_name/-'                    '-name-'

                                                                                 Required
==========================================================================================
                                                                                 Optional

           .-*NO--.          .-*PRV---.
>--DSPFLDD-+------+---DTASET-+--------+-------------------------------------------------><
           '-*YES-'          +-*FIRST-+
                             '-name---'
```

SRCFILE Parameter

Indicates the source file to be examined. The source file you specify will be searched for the member listed by the MBR parameter.

<u>*LIBL:</u> your current library list will be searched for the source file.

MBR Parameter

Specifies the member(s) that should be examined in the indicated source file. The member must contain DDS display file source code.

<u>*ALL</u>: Analyze all source members in the specified file.

Name: specify a single member for analysis.

DSPFLDD Parameter

Display descriptions for each field in the report.

<u>*NO</u>: do not display field descriptions.

***YES:** The fields used on the report will follow its printed representation. Field name, usage, length, position, and text will be indicated.

DTASET Parameter

Specify the cross reference data set to be used in acquiring the information.

***PRV:** Use the value for this parameter from the previous session. If no previous session has occurred, use *FIRST.

***FIRST:** information from the first (default) data set should be accessed.

Name: Specify any existing data set. Use the command prompter to find the names available to you.

The PRTDSPF Report

Sample display file output is shown below. One page is devoted to each display format in the source member (in this example, just one format). Subfile control records are displayed with their subfile. Fields are printed with all conditional display attributes turned on. Fields with non-display attributes will not print. Reverse image fields appear as normal fields.

```
10/31/2010                        Abstract                              Page  1
11:08:32                     Display File Layout               Format: QINSTAPPR

File: ABSTRACT/QINSTAPP


   .... ....1.... ....2.... ....3.... ....4.... ....5.... ....6.... ....7.... ....8
 **********************************************************************************
 1                                 Help/Systems                                   *
 2                             Product Installation                               *
 3                                                                                *
 4 Choose the products that will be installed on your system.                     *
 5  1=Select                                                                      *
 6                                                                                *
 7 Sel Product name   Description                                                 *
 8  B  All products                                                               *
 9                                                                                *
10  B  SEQUEL      Database query and report generator                            *
11  B  ABSTRACT     Development/documentation environment                         *
12  B  STATUS      Job and resource accounting tool                               *
13  B  PILOT      Automated job scheduling and report distribution                *
14  B  REU       Interactive RPG report layout aid                                *
15                                                                                *
16                                                                                *
17                                                                                *
18                                                                                *
19                                                                                *
20                                                                                *
21                                                                                *
22                                                                                *
23 F3=Exit                                                                        *
24                                                                                *
 **********************************************************************************
```

Field Descriptions

The field descriptions for the items on the display will be printed if DSPFLDD(*YES) is specified when the command is run.

The field descriptions corresponding to the layout example above are presented below:

```
Format: QINSTAPPR

Command keys: CA03(03)   CA12(12)

Seq Indicatrs Field Name Use  Length Line  Pos Field Text
  1                                     1   28 'Advanced Systems Concepts'
  2                                     2   31 'Product Installation'
  3                                     4    2 'Choose the products that will be i-
  4                                            nstalled on your system.'
  5                                     5    4 '1=Select'
  6                                     7    2 'Sel'
  7                                     7    7 'Product name'
  8                                     7   24 'Description'
  9           INST_ALL    B      1      8    3
 10                                     8    7 'All'
 11                                     8   11 'products'
 12           INST_SQ     B      1     10    3
 13                                    10    7 'SEQUEL'
 14                                    10   24 'Database query and report generato-
 15                                            r'
 16           INST_AP     B      1     11    3
 17                                    11    7 'ABSTRACT'
 18                                    11   24 'Development/documentation environm-
 19                                            ent'
 20           INST_ST     B      1     12    3
 21                                    12    7 'STATUS'
 22                                    12   24 'Job and resource accounting tool'
 23           INST_PL     B      1     13    3
 24                                    13    7 'PILOT'
 25                                    13   24 'Automated job scheduling and repor-
 26                                            t distribution'
 27           INST_RU     B      1     14    3
 28                                    14    7 'REU'
 29                                    14   24 'Interactive RPG report layout aid'
 30                                    23    2 'F3=Exit'
```

Exception Reports

In addition to keeping track of the relationships between the objects in your applications, ABSTRACT can print other reports that you might find useful in developing and maintaining your software.

These reports are categorized under the broad title of exception reports because most of them can inform you of problems in your application or the cross reference dictionary that you should correct.

This chapter will discuss the types of reports that ABSTRACT can provide to aid you in discovering and correcting problems. Each report is accessible using the Print Object Exception (PRTOBJEXCP) command, or through the exception reporting menu, APEXCP.

Object Exception (APEXCP) Menu

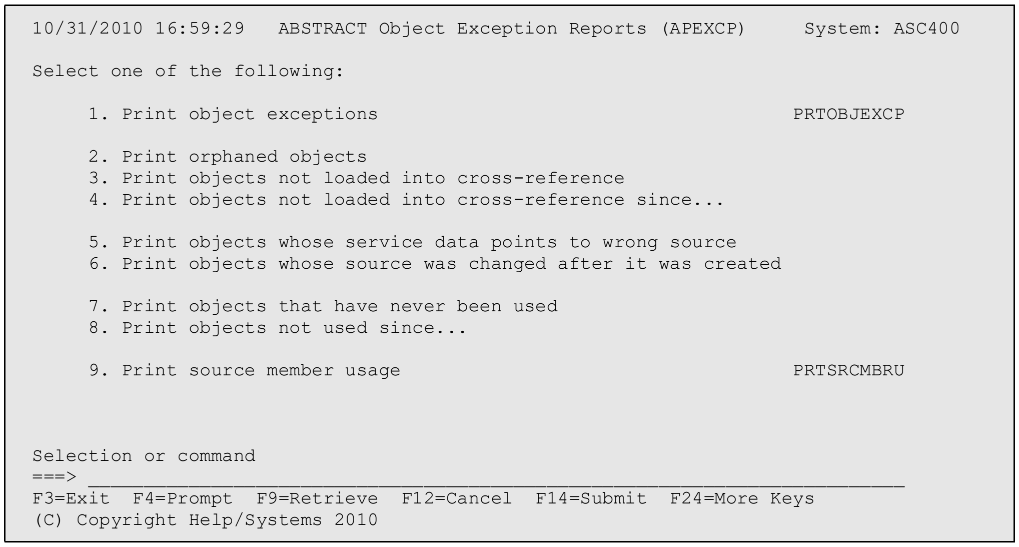
The exception reporting menu (APEXCP) can be accessed from the ABSTRACT main menu through option 5, or by typing:

```
GO ABSTRACT/APEXCP
```

on a command entry line and pressing the Enter key.

The menu allows you to access all the exception reporting capabilities of ABSTRACT. It also allows you full access to your authorized command set through the command entry line at the bottom of the display. Like the other ABSTRACT menus, it can access the option file and default job queue setting through function keys.

Once you have started the menu, it will look like the one below.



```
10/31/2010 16:59:29   ABSTRACT Object Exception Reports (APEXCP)     System: ASC400

Select one of the following:

     1. Print object exceptions                                     PRTOBJEXCP

     2. Print orphaned objects
     3. Print objects not loaded into cross-reference
     4. Print objects not loaded into cross-reference since...

     5. Print objects whose service data points to wrong source
     6. Print objects whose source was changed after it was created

     7. Print objects that have never been used
     8. Print objects not used since...

     9. Print source member usage                                   PRTSRCMBRU



Selection or command
===> _______________________________________________________________________________
F3=Exit  F4=Prompt  F9=Retrieve  F12=Cancel  F14=Submit  F24=More Keys
(C) Copyright Help/Systems 2010
```

Each report request can be run interactively or submitted to the job queue for batch execution.

The first eight options use the Print Object Exceptions (PRTOBJEXCP) command to create the listed report. When a menu option is selected, an appropriate PRTOBJEXCP command prompt will be displayed. Parameter defaults are automatically selected to create the type of report indicated.

The final menu option runs the Print Source Member Usage (PRTSRCMBRU) command to create a listing that shows which objects have been created from the members in a source file.

If your default setting for the run/compile in batch value (see page 3-19) is a Y, your report request will be submitted to the batch job queue. Otherwise, it will begin running interactively. Given that some requests can run for an extended period of time, this may not be an efficient use of your display device or interactive priority.

Your request will always be submitted to the batch subsystem if you type the menu option number and press F14 or F15. Refer to the section describing the ABSTRACT menu driver files in the Introduction for information about changing the batch/interactive environment for each menu option.

Menu options

Refer to the description of the PRTOBJEXCP and PRTSRCMBRU commands in the following section for complete information about the types of exception reports you can request.

Use option 1 to prompt the PRTOBJEXCP command with OPTION(*ALL) pre-filled. This will allow you to request that all exception related information will be printed on the report.

Options 2 through 8 will pre-fill the PRTOBJEXCP command prompt with appropriate values to select a particular type of report. These are:

Orphaned Objects: objects in an application library that are not referenced by any object loaded into the ABSTRACT cross reference dictionary. Menu option 2 selects the orphaned object report by specifying OPTION(*ORPHAN) on the PRTOBJEXCP command.

Undocumented Objects: objects in an application library that have never been loaded into the ABSTRACT cross reference dictionary, or that haven't been loaded since a given date. Menu options 3 and 4 create this type of report by selecting the last load date (LSTLOD) parameter. Option 3 specifies LSTLOD(*NOLOD) to select only objects that have never been loaded. Menu option 4 will prompt for the last load date value.

Object-source conflicts: Menu option 5 documents objects in your application library with service (compile-time) information that references a source member that cannot be located on your system. The OPTION(*NOSRC) parameter of the PRTOBJEXCP command is used.

Option 6 prints a report showing objects that reference a source member which has changed since the object was created. OPTION(*SRCCHG) is pre-filled on the command prompt.

Unused objects: Menu option 7 prints a listing of application objects that have never been used, selecting LSTUSD(*NOUSE) on the command prompt.

Option 8 will list only objects that haven't been used since a given date by prompting you for the LSTUSD value to be used in creating the report.

Source usage: Menu option 9 documents the way that source members are used. You specify the source file and a list of libraries to be searched. ABSTRACT prints a listing that shows the programs that have been created from the members in the source file. Members for which no programs could be found are listed at the end of the report.

Menu Function Keys

As with other ABSTRACT menus, the file analysis menu provides several function key options. Notice that there are some differences (cf. F14, F19) from the function key set provided by OS/400 system menus.

F3	Exit the ABSTRACT environment
F4	Prompt the option or command that is on the command line
F9	Retrieve ‘backwards’ through previously executed commands
F12	Exit the menu display and return to the previous function
F14	Submit the option or command on the command line to batch using the default job description defined for this user
F15	Prompt the Submit Job (SBMJOB) command before submitting the option or the command on the command line
F16	Work with options in the user option file
F18	Change the default job description and user option file defined for the current user
F19	Retrieve ‘forwards’ through previously executed commands
F24	Display more function keys

Print Object Exceptions (PRTOBJEXCP) Command

The Print Object Exception (PRTOBJEXCP) command allows you to create reports that describe potential problems with your application or ABSTRACT cross reference data. Use options one through eight of the APEXCP exception menu to execute the command from a command entry prompt.

Several types of exception reports can be generated by the PRTOBJEXCP command:

Orphaned objects: objects in an application library that are not referenced by any object loaded into the ABSTRACT cross reference dictionary. Presumably, if it is not referenced, it is not used by your application.

Undocumented objects: objects in an application library that have never been loaded into the ABSTRACT cross reference dictionary, or that haven't been loaded since a given date.

Object-source conflicts: objects in your application library with service (compile-time) information that references a source member that cannot be located on your system, or that references a source member changed since the object was created.

Most object-source conflicts result from a development strategy that calls for objects (and source) to be moved to the production library once development in a test library has been completed. Once the source code is moved, the corresponding object's service data no longer points to it. The ABSTRACT Change Service Description (CHGSVCDTA) command in chapter 7 can be used to update the object's service description to correct this problem.

Other object-source conflicts occur because of careless source management practices that allow source code to be changed without requiring subsequent compilation. The ABSTRACT program or file recompiler can be used to recreate these objects.

Unused objects: application objects that have never been used, or haven't been used since a given date. The OS/400 operating system updates an object's last used date whenever it is used. Refer to the Display Object Description (DSPOBJD) and Change Object Description (CHGOBJD) commands in the CL Reference Manual for a complete description of the object usage information and how it can be managed.

Report requests can be made so that one, some, or all of these types of reports are run with a single PRTOBJEXCP command.

The syntax for the PRTOBJEXCP command is shown below:

```
**PRTOBJEXCP****************************************************************************

                      .-*LIBL/-------.   .-*ALL---------.
>>--PRTOBJEXCP----OBJ-+--------------+---+--------------+------------------------------>
                      '-library_name-'   +-generic_name-+
                                         '-object_name--'

           .-*ALL--------.          .-*ALL--------.          .-*ALLOBJ---.
>--OBJTYPE-+-------------+---OBJATR-+-------------+---OPTION-+-----------+------------->
           +-object_type-+          '-object_name-'          +-*ALL------+
           +-*DBF--------+                                   +-*NONE-----+
           +-*FLD--------+                                     .-3 max---.
           +-MBR---------+                                     v         |
           '-*SUB--------'                                     +-*ORPHAN-+
                                                               +-*NOSRC--+
                                                               '-*SRCCHG-'


          .-*NOUSE-----.          .-*NOLOD-----.          .-*PRV---.
>--LSTUSE-+------------+---LSTLOD-+------------+---DTASET-+--------+------------------><
          '-date_value-'          '-date_value-'          +-*FIRST-+
                                                          '-name---'
```

The OBJ, OBJTYP, and OBJATR parameter values are used to determine the objects to be included on the report. The set of objects identified by these values will be examined for the attributes that create an exception. If there is a problem of the type indicated by the OPTION, LSTUSE, and LSTLOD values, the object will appear on the exception report.

Only objects to which you have *USE authority will be included in the list.

OBJ Parameter

Specifies the objects to be included on the report. The indicated library will be searched for objects meeting the name, type, and attribute values you specify. If no library qualifier is specified, *LIBL is assumed and all libraries in the job's library list are searched for the objects.

***ALL:** All objects in the specified library(s) are selected.

object-name: Only objects with this specific name will be included in the list.

***generic*-name:** Specify a partial object name qualified by an asterisk (*) to select several objects meeting the criteria. The following generic forms are allowed: ABC*, *ABC, *ABC*, A*B, abc*, **ALL. Refer to Appendix G, General Functions, in Volume 1 of the CL Reference manual for more information about generic names.

OBJTYPE Parameter

Specifies the object type. You can use this parameter to work with all object types or a subset of objects. An expanded description of the OBJTYPE parameter and a list of valid OS/400 object types can be found in Appendix A, Expanded Parameter Descriptions, of Volume 1 of the CL Reference manual.

***ALL:** All objects regardless of their type are included.

object-type: Specify any valid system object type to display a list of all objects of that particular type.

***DBF:** All data base files will be included. These are *FILE objects with an attribute beginning with PF, LF, or DDMF.

***FLD:** All fields in the ABSTRACT dictionary. The *FLD value will be useful only with *ORPHAN and *LSTLOD reports.

***MBR:** All members in the ABSTRACT dictionary. The *MBR value will be useful only with *ORPHAN and *LSTLOD reports.

***SUB:** All subroutines loaded into the ABSTRACT dictionary. The *SUB value will be useful only with *ORPHAN and *LSTLOD reports.

OBJATR Parameter

Specifies the object attribute. You can use this parameter to work with all object attributes or only specific ones. An expanded description of the OBJATR parameter and a list of valid OS/400 object attributes can be found in Appendix A, Expanded Parameter Descriptions, of Volume 1 of the CL Reference manual.

***ALL:** All objects in the specified library(s) are selected regardless of their attribute definition.

***BLANK:** Only objects with no attribute will be included.

object-attribute: Only objects with this specific attribute will be included in the list. PF-DTA can be used to select only non-source physical files. PF-SRC can be used to select only source physical files.

***generic*-name:** Specify a partial object attribute qualified by an asterisk (*) to select several objects meeting the criteria. The following generic forms are allowed: ABC*, *ABC, *ABC*, A*B, abc*, **ALL. For example, *RPG* will select RPG, RPG36, RPG38, and SQLRPG objects. Refer to Appendix G, General Functions, in Volume 1 of the CL Reference manual for more information about generic names.

OPTION Parameter

The OPTION parameter specifies the type(s) of exception reporting that you want to perform. Specify *ALL to indicate that the report should include all exceptions, or specify up to three specific types:

***ALLOBJ:** list the information for all objects, regardless of their exception status. Any exceptions for the reported objects will be listed as well.

***ALL:** print only exception information, listing *ORPHAN, *NOSRC and *SRCCHG types of exceptions. The report will include both orphaned objects and source conflict information.

***ORPHAN:** objects to which ABSTRACT can find no reference will be included. The ABSTRACT cross-reference dictionary is searched for references to each object selected.

***NOSRC:** lists objects with service information indicating a source file, library, or member that does not exist on the system. ABSTRACT cross reference information is not used in preparing this report.

***SRCCHG:** the object service information will be used to locate the object's source member and then compared to the member change date. If the member change date is subsequent to the date indicated by the service information, the object will be included in the exception report. Cross reference information is not used to prepare this report.

***NONE:** objects without orphan or source conflict exceptions will be printed if they meet criteria specified by the LSTUSE and LSTLOD parameter values.

LSTUSE Parameter

Specifies additional selection criteria for the objects to be included on the report. Current object usage information will be examined to find objects that qualify.

***NOUSE:** objects that have never been used will be included on the report in addition to those already selected due to the OPTION and LSTLOD parameter values.

date-value: objects that have not been used since the specified date will be included on the report in addition to those already selected due to the OPTION and LSTLOD parameter values.

LSTLOD Parameter

This parameter allows you to apply selection criteria based on the date that ABSTRACT cross reference information was created. Objects that have been changed, or have not been loaded into the cross-reference since the date specified on the LSTLOD parameter will be listed on the report in addition to those already selected due to the OPTION and LSTUSE parameter values.

***NOLOD:** objects that have never been loaded into the ABSTRACT cross-reference files will be listed.

date-value: objects that have not been loaded into the ABSTRACT dictionary since the specified date will be included.

DTASET Parameter

This parameter specifies the name of the ABSTRACT data set to use when accessing cross reference information. The list of current data set names can be accessed by prompting the command and then prompting the data set parameter.

<u>***PRV:**</u> Use the value for this parameter from the previous session. If no previous session has occurred, use *FIRST.

***FIRST:** information from the first (default) data set should be accessed.

name: specify the name of the data set to be used when accessing cross reference information. The name of a current data set must be specified or an error will result.

Examples

```
PRTOBJEXCP OBJ(PRODLIB/*ALL) OPTION(*ALLOBJ)
```

This request will print a report showing all objects in the PRODLIB library whether or not any exceptions are detected. All exceptions will be noted on the report - source conflicts, unreferenced objects, unused objects, and objects not loaded into the ABSTRACT cross references.

```
PRTOBJEXCP OBJ(PRODLIB/*ALL) OPTION(*NOSRC)
```

This request will print a report showing only the objects in the PRODLIB library that reference source members that don't exist on the system.

```
PRTOBJEXCP OBJ(PRODLIB/*ALL) OPTION(*NONE) +
     LSTUSE(*NOUSE)
```

Lists only the objects in PRODLIB that have no last used date. Source conflict and unreferenced object exceptions will not be listed on the report. Specifying LSTUSE(010192) would print objects with a last used date prior to January 1, 1992.

The PRTOBJEXCP Report

The PRTOBJEXCP report will include the following information:

Object name, type, and attribute
Source file, library, and member used to create the object
Object creation date
Source change date (if source can be located)
Date the object was last used
Cross-reference load date (if loaded)
Object text
Object size

Items with a source conflict (source not found or changed since object creation) will be noted with an asterisk at the left margin of the report. The source file and member will be underlined if the source could not be found. The object creation date will be underlined if the source change date is more recent than the object creation date.

Orphaned objects (unreferenced within the cross reference dictionary) will be noted with an asterisk next to the cross reference load date.

The example on page 6-9 shows each of the different types of exceptions that can appear on the report. Selected portions of the report are shown below for additional explanation.

```
   Object     Type       Src file   Src lib    Src member Creation Source   Last use XRF load  Object text
   ADDTIME    *PGM RPG   QRPGSRC    PILOT#     ADDTIME    11/07/88 11/07/88
 * AUTOSCH    *PGM RPG   QRPGSRC    PILOT#     AUTOSCH    05/13/91 02/19/92          02/13/92
 * AUTOSCHC   *PGM CLP   QCLSRC     PILOT#     AUTOSCHC   05/13/91 02/19/92          02/24/92  AutoSch command
.
.
.
 * MCR010     *PGM MI    QIRPSRC    PILOT#     MCR010     11/13/89          03/04/92
 * MCR030     *PGM MI    QIRPSRC    PILOT#     MCR030     11/09/88                           *
.
.
.
 * PJOBLOG    *FILE PRTF QDDSSRC    PILOT#     PJOBLOG    03/17/86          06/11/90
```

Source conflicts

An asterisk at the left margin indicates that either the source could not be located, or it has changed since the object was created.

Programs AUTOSCH and AUTOSCHC demonstrate a conflict between the creation date of the program and the last change date of the source member. These conflicts will be highlighted by an underlined creation date.

Programs MCR010 and MCR030 show that the original source code can no longer be found on the system. The source member used to create the PJOBLOG file couldn't be found either, though its source file could. The underlined source file and member names indicate that source could not be located. Either it does not exist, or you are not authorized to it.

```
   Object     Type       Src file   Src lib    Src member Creation Source   Last use XRF load  Object text
   DSPJLG     *PGM CLP   QCLSRC     PILOT#     DSPJLG     10/26/88 10/26/88 03/04/92 02/13/92  Display joblog
   DSPJOBC    *PGM CLP   QCLSRC     PILOT#     DSPJOBC    10/27/88 10/27/88 03/04/92           Display job
   DSPLGC     *PGM CLP   QCLSRC     PILOT#     DSPLGC     10/26/88 10/26/88          02/13/92* Display joblog
   GETJOBAC   *PGM CLP   QCLSRC     PILOT#     GETJOBAC   10/26/88 10/26/88                  * Retrieve job ac
.
.
   DEPTJOIN2  *FILE LF   QDDSSRC    PILOT#     DEPTJOIN2  10/26/88 06/05/86                  *
.
.
   JOBSCHL1   *FILE LF   QDDSSRC    PILOT#     JOBSCHL1   03/10/92 03/31/86                  *
```

Orphaned objects

An asterisk to the right of the load date informs you of an orphaned object exception. There are no references to the object in any of the ABSTRACT cross references. Although this would be expected for the first program in a job stream, most objects that are really used in your application should have at least one reference to them.

The sample report indicates that no ABSTRACT references to programs DSPLGC, GETJOBAC and MCR030 (among others) could be found. Likewise, no references to the DEPTJOIN2 or JOBSCHL1 files could be found.

```
   Object     Type       Src file   Src lib    Src member Creation Source   Last use XRF load  Object text
   ADDTIME    *PGM RPG   QRPGSRC    PILOT#     ADDTIME    11/07/88 11/07/88
 * AUTOSCH    *PGM RPG   QRPGSRC    PILOT#     AUTOSCH    05/13/91 02/19/92          02/13/92
 * AUTOSCHC   *PGM CLP   QCLSRC     PILOT#     AUTOSCHC   05/13/91 02/19/92          02/24/92  AutoSch command
   CHKJOBC    *PGM CLP   QCLSRC     PILOT#     CHKJOBC    10/26/88 10/26/88 03/04/92           Validate jobq,j
   CHKUSRC    *PGM CLP   QCLSRC     PILOT#     CHKUSRC    10/26/88 10/26/88 03/04/92           Check user prof
```

Unused objects

A missing last use date (ADDTIME, AUTOSCH, AUTOSCHC) indicates that the OS400 operating system has never detected this object being referenced.

Undocumented objects

A missing cross reference load date (CHKJOBC, CHKUSRC) indicates that the object has never been processed by the LOADXREF command.

```
 10/31/2010 17:00:30                                    ABSTRACT                      Page . . . :   1
                                                 Object Exception Report                  System . . : ASC401
Object name . . : *ALL
Object type . . : *ALL       Attribute  : *ALL
Object library  : PILOT
Options . . . . : *ALLOBJ

  Object     Type       Src file   Src lib    Src member Creation Source   Last use XRF load  Object text
  ADDTIME    *PGM RPG   QRPGSRC    PILOT#     ADDTIME    11/07/88 11/07/88
* AUTOSCH    *PGM RPG   QRPGSRC    PILOT#     AUTOSCH    05/13/91 02/19/92          02/13/92
* AUTOSCHC   *PGM CLP   QCLSRC     PILOT#     AUTOSCHC   05/13/91 02/19/92          02/24/92  AutoSch command
  DSPJOBC    *PGM CLP   QCLSRC     PILOT#     DSPJOBC    10/27/88 10/27/88 03/04/92           Display job
  DSPLGC     *PGM CLP   QCLSRC     PILOT#     DSPLGC     10/26/88 10/26/88          02/13/92* Display joblog
  GETJOBAC   *PGM CLP   QCLSRC     PILOT#     GETJOBAC   10/26/88 10/26/88                  * Retrieve job ac
  HLP001     *PGM RPG   QRPGSRC    PILOT#     HLP001     10/27/88 10/27/88          02/17/92  Help text proce
  HOLLST     *PGM RPG   QRPGSRC    PILOT#     HOLLST     11/08/88 11/08/88          02/13/92  Holiday listing
  HOLMNT     *PGM RPG   QRPGSRC    PILOT#     HOLMNT     10/28/88 10/27/88          02/13/92  Holiday definit
* INSTALL2   *PGM MI    QIRPSRC    PILOT#     INSTALL2   11/09/88
  INT001     *PGM CLP   QCLSRC     PILOT#     INT001     10/26/88 10/26/88 03/04/92 02/13/92  Package default
  JOBCHK     *PGM RPG   QRPGSRC    PILOT#     JOBCHK     02/12/90 02/12/90 03/04/92 02/13/92  Scheduling - Ch
  JOBLST     *PGM RPG   QRPGSRC    PILOT#     JOBLST     11/08/88 11/08/88          02/25/92  Job definition
  JOBMNT     *PGM RPG   QRPGSRC    PILOT#     JOBMNT     10/16/90 10/16/90 03/04/92 02/13/92  Job definition
  JOBSBMC    *PGM CLP   QCLSRC     PILOT#     JOBSBMC    04/17/89 04/17/89 03/04/92 02/13/92  Submit job to e
* MCR010     *PGM MI    QIRPSRC    PILOT#     MCR010     11/13/89          03/04/92
* MCR030     *PGM MI    QIRPSRC    PILOT#     MCR030     11/09/88                           *
  MENUC      *PGM CLP   QCLSRC     PILOT#     MENUC      11/15/90 11/15/90 03/04/92 02/18/92  Pilot menu driv
* PILOTDTA   *PGM CLP   QCLSRC     PILOT#     PILOTDTA   02/15/89 08/08/90          02/13/92*
* PIL020     *PGM MI    QIRPSRC    PILOT#     PIL020     11/13/89          03/04/92
  PRG001     *PGM RPG   QRPGSRC    PILOT#     PRG001     10/27/88 10/27/88          02/13/92  Schedule purge
* PRG001C    *PGM CLP   QCLSRC     PILOT#     PRG001C    03/27/89 01/17/92          02/13/92  Submit purge re
* PRG002     *PGM RPG   QRPGSRC    PILOT#     PRG002     10/27/88 02/04/92          02/13/92  Schedule purge
  READLOG    *PGM RPG   QRPGSRC    PILOT#     READLOG    10/27/88 03/17/86          02/13/92  Job log reader
  RPT003     *PGM RPG   QRPGSRC    PILOT#     RPT003     05/14/91 02/22/91          02/25/92  Report distribu
* RTVJOBDA   *PGM MI    QIRPSRC    PILOT#     RTVJOBDA   07/26/88
* SCHMNT     *PGM RPG   QRPGSRC    PILOT#     SCHMNT     11/28/89 12/12/91 03/04/92 02/13/92  Schedule mainte
  SCHTMPJOB  *PGM CLP   QCLSRC     PILOT#     SCHTMPJOB  11/09/88 11/09/88          02/13/92
  SEC901     *PGM CLP   QCLSRC     PILOT#     SEC901     10/26/88 10/26/88                  *
  SNDMSG     *PGM CLP   QCLSRC     PILOT#     SNDMSG     10/26/88 10/26/88 03/04/92         *
* WCBT01     *PGM MI    QIRPSRC    PILOT#     WCBT01     11/07/88          03/04/92           Work control bl
  PILOT      *OUTQ                                       02/06/85          03/04/92         *
  DOCMSG     *MSGF                                       07/31/90                           *
  CALWRK     *FILE PF   QDDSSRC    PILOT#     CALWRK     10/26/88 09/19/85
  HELPTXT    *FILE PF   QDDSSRC    ABSTRACT#  HELPTXT    10/26/88 10/18/88
* HOLIDAYS   *FILE PF   QDDSSRC    PILOT#     HOLIDAYS   10/26/88 02/19/92
  HOLMNT     *FILE DSPF QDDSSRC    PILOT#     HOLMNT     10/28/88 10/28/88 06/11/90           Holiday definit
  INT001     *FILE DSPF QDDSSRC    PILOT#     INT001     10/26/88 10/26/88 03/04/92           Package default
  JOBLOG     *FILE PF                                    10/26/88
  JOBMNT     *FILE DSPF QDDSSRC    PILOT#     JOBMNT     11/08/88 11/08/88 03/04/92           Job definition
  JOBSCH     *FILE PF   QDDSSRC    PILOT#     JOBSCH     03/10/92 01/06/86                    Job Schedule
  JOBSCHL1   *FILE LF   QDDSSRC    PILOT#     JOBSCHL1   03/10/92 03/31/86                  *
  MENUC      *FILE DSPF QDDSSRC    PILOT#     MENUC      10/26/88 10/26/88 03/04/92           Pilot menu
  PERHDR     *FILE PF   QDDSSRC    PILOT#     PERHDR     10/26/88 09/12/85
  PERMNT     *FILE DSPF QDDSSRC    PILOT#     PERMNT     10/28/88 10/28/88                    Period code def
  PILOTPR    *FILE PRTF                                  11/08/88          06/11/90
* PJOBLOG    *FILE PRTF QDDSSRC    PILOT#     PJOBLOG    03/17/86          06/11/90
  PRG001     *FILE DSPF QDDSSRC    PILOT#     PRG001     10/26/88 10/26/88                    Schedule purge
  REPORTS    *FILE DSPF QDDSSRC    PILOT#     REPORTS    11/09/88 11/09/88 06/11/90           Report requests
  RPTDTLL1   *FILE LF   QDDSSRC    PILOT#     RPTDTLL1   10/26/88 03/31/86
  RPTHDR     *FILE PF   QDDSSRC    PILOT#     RPTHDR     10/26/88 11/11/85
  SCHMNT     *FILE DSPF QDDSSRC    PILOT#     SCHMNT     11/09/88 11/09/88 03/04/92           Schedule mainte
  SEC900     *FILE DSPF QDDSSRC    PILOT#     SEC900     10/17/89 10/17/89 03/04/92           Security displa
  WAKEUP     *MSGQ                                       12/19/85          03/04/92         *
  AUTOPILOT  *JOBD                                       05/07/86                           *
  PILOT      *JOBD                                       02/06/85          03/04/92         *
* AUTOSCH    *CMD       QCMDSRC    PILOT#     AUTOSCH    04/14/87                   02/13/92* Add a job to th
* PILOT      *CMD       QCMDSRC    PILOT#     PILOT      01/13/86                   02/13/92
  ASC#PL     *DTAARA                                     03/04/92          03/04/92
  USRPRF     *DTAARA                                     03/17/86          03/04/92

  74 Objects analyzed        9  Objects created from source no longer on system
```

Print Source Member Usage (PRTSRCMBRU)

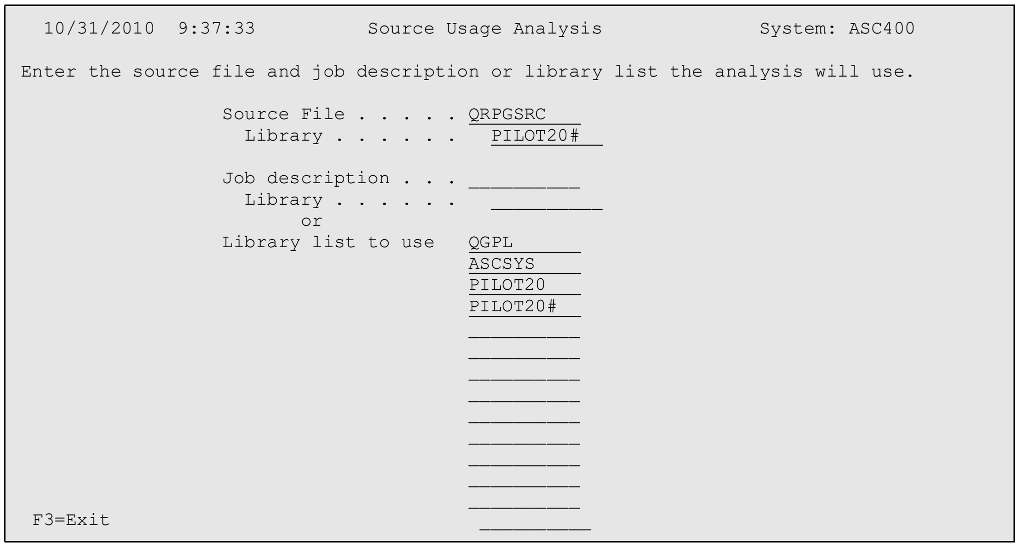
The Print Source Member Usage (PRTSRCMBRU) command will print a report showing all the objects that have been created from the members in a source file. You provide a list of libraries to be searched and ABSTRACT searches them for objects that have been compiled from the source file you name. Source members that do not have any corresponding objects will also be listed on the report.

The PRTSRCMBRU command can be run by selecting option 9 from the APEXCP menu or by executing the command directly from a command entry prompt.

The command has no parameters.

```
>>--PRTSRCMBRU---------------------------------------------------------------------><
```

When it is run, a display will prompt you for the source file you want analyzed and the list of libraries that you want searched. When you complete the display a batch job will be submitted using your default job description.



```
  10/31/2010  9:37:33          Source Usage Analysis             System: ASC400

Enter the source file and job description or library list the analysis will use.

                  Source File . . . . . QRPGSRC
                    Library . . . . . .   PILOT20#

                  Job description . . . __________
                    Library . . . . . .   __________
                         or
                  Library list to use   QGPL
                                        ASCSYS
                                        PILOT20
                                        PILOT20#
                                        __________
                                        __________
                                        __________
                                        __________
                                        __________
                                        __________
                                        __________
                                        __________
                                        __________
 F3=Exit                                 __________
```

This display appears when the PRTSRCMBRU command is run. Type the requested information and press Enter to submit the report to batch, or press F3 to exit the display.

Specify the source file that you want analyzed. Each library specified in the job description or list will be searched for objects that were compiled using the members in the file that you name.

Indicate whether you want ABSTRACT to search libraries on the library list associated with a job description, or to search a specific list of libraries.

Note: *The analysis uses object service data. If the service data does not properly indicate the library, file, and member containing an object's source code, the report cannot reflect the "true" status of your source/object relationships. Review the Change Service Data (CHGSVCDTA) command for instructions to correct this problem.*

The report will look like the one below. The first section of the report lists the objects that have been created from the members in the source file. If the source member has been changed since the object was created, an asterisk (*) will appear next to the object's name. (c.f. PERMNT). The second section of the report lists source members that do not have a corresponding object within the list of libraries you provided.

```
13:37:55                                Member Usage Analysis                             Page     1
 9/21/93

Source File      : QRPGSRC    Library: PILOT20#

Library list used: QGPL PILOT20 PILOT20# ABSTRACT

(Objects with source date > compile date are marked with an *)

   Object     Library     Type       Compile Date           Source Date
   AUTOSCH    PILOT20     *PGM RPG    8/19/93  15:43:12      8/19/93  15:42:16 Auto schedule processor
   DSPTRIG    PILOT20     *PGM RPG    8/27/93  11:58:53      8/27/93  11:57:57 Display pending triggers
   GETSPLFL   PILOT20     *PGM RPG    7/01/93  10:13:48      7/01/93  10:13:13
   JOBCHK     PILOT20     *PGM RPG    8/09/93   8:12:52      8/09/93   8:11:46 Scheduling - Check for jobs
   JOBMNT     PILOT20     *PGM RPG    8/30/93   8:30:33      8/30/93   8:22:10 Job definition maintenance
   LIBLMNT    PILOT20     *PGM RPG    6/14/93  16:29:51      6/14/93  16:29:13
*  PERMNT     PILOT20     *PGM RPG    8/09/93   8:36:12      8/09/93   9:35:01 Period code maintenance
   SCHMNT     PILOT20     *PGM RPG    9/14/93   8:58:19      9/14/93   8:55:54 Schedule maintenance
   SCH001     PILOT20     *PGM RPG    8/27/93  10:26:34      8/27/93  10:25:14 Schedule loader
```

```
13:37:55                                Member Usage Analysis                             Page     2
 9/21/93

Source File      : QRPGSRC    Library: PILOT20#

Library list used: QGPL PILOT20 PILOT20# ABSTRACT

No objects were found for the following members:

   Member     Type       Source Date
   ADDTIME    RPG         6/15/93  13:17:01 Add time offset
   CAL003     RPG         3/25/92  10:32:19 Current schedule report
   HOLLST     RPG         5/07/93  11:59:09 Holiday report
   LBLLIST    RPG        10/08/92  15:46:10 Library list report
   OPSPLA0100 RPG         6/29/93   9:05:27 SPLA0100 - Data Structure for QUSRSPLA API
   PLTDATES   RPG        12/17/91  10:50:46 Build PILOT date database
   PRG003     RPG         3/13/92   9:09:46 Purge schedule based on retention days
   READLOG    RPG         6/15/93  13:21:43 Job log reader
   RMTLIST    RPG         3/03/92  14:11:36 Remote location report
   RPT003     RPG         5/07/93  11:56:32 Report Distribution by Date
```

Programming Tools

In addition to its documentation facilities, ABSTRACT provides a number of programming tools that can assist you in creating and maintaining your applications software. This section describes these additional ABSTRACT facilities.

There are two types of programming tools available. Most of these tools work like other ABSTRACT functions, and are accessible from a menu or through command entry. There are also two program callable tools that permit the imbedding of ABSTRACT functions in user written applications or other vendor products such as a change management system.

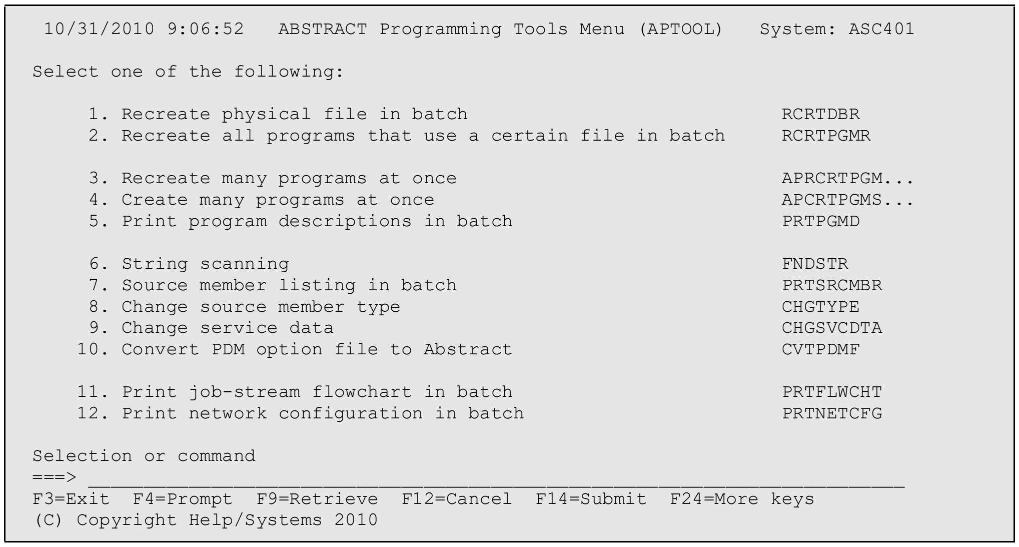
Programming Tools (APTOOL) Menu

The primary programming tools menu (APTOOL) can be accessed from the ABSTRACT main menu through option 6, or by typing:

```
GO ABSTRACT/APTOOL
```

on a command entry line and pressing the Enter key.

As with other ABSTRACT menus, the APTOOL menu provides full access to your authorized command set through the command entry line at the bottom of the display.



```
 10/31/2010 9:06:52   ABSTRACT Programming Tools Menu (APTOOL)   System: ASC401

Select one of the following:

     1. Recreate physical file in batch                           RCRTDBR
     2. Recreate all programs that use a certain file in batch    RCRTPGMR

     3. Recreate many programs at once                            APRCRTPGM...
     4. Create many programs at once                              APCRTPGMS...
     5. Print program descriptions in batch                       PRTPGMD

     6. String scanning                                           FNDSTR
     7. Source member listing in batch                            PRTSRCMBR
     8. Change source member type                                 CHGTYPE
     9. Change service data                                       CHGSVCDTA
    10. Convert PDM option file to Abstract                       CVTPDMF

    11. Print job-stream flowchart in batch                       PRTFLWCHT
    12. Print network configuration in batch                      PRTNETCFG

Selection or command
===> ___________________________________________________________________________
F3=Exit  F4=Prompt  F9=Retrieve  F12=Cancel  F14=Submit  F24=More keys
(C) Copyright Help/Systems 2010
```

Most of the menu options will prompt, and then run, an ABSTRACT command. Depending on the menu item, the command will be submitted to the batch job queue or it will begin running immediately when you press the Enter key. Given that some requests can run for an extended period of time, this may not be an efficient use of your display device or interactive priority.

Your request will always be submitted to the batch subsystem if you type the menu option number and press F14 or F15. Refer to the section describing the ABSTRACT menu driver files in the Introduction for information about changing the batch/interactive environment for each menu option.

Select option 1 to prompt the Recreate Data Base Relations (RCRTDBR) command. It provides a simple, easy to use mechanism for re-creating a physical file (and all the logical files built over it) and retaining the current data. Refer to page 7-6 for complete information about this function.

Use option 2 to prompt and run the Recreate Program References (RCRTPGMR) command. Like the RTCRDBR command, it allows you a simple method of re-creating existing objects. Based on cross reference information, ABSTRACT will automatically recompile the programs that use the file you specify on the RCRTPGMR command. See page 7-14 for details.

Menu options 3 and 4 can be used to access two menus that provide support for additional ABSTRACT compiling options. Option 3 will present the APRCRTPGM menu, option 4 will begin the APCRTPGMS menu.

Use option 5 to print a report showing information about the program objects in one of your libraries. Program name, type, owner, authority adoption, etc. will be presented. See page 7-28 for a complete description of this function.

Menu option 6 can be used to scan source or message files for a given string. Similar to the string search function provided by PDM, it also provides capabilities to search for several string sequences, search specific columns, and search multiple source files with a single request. The string scanning functions are described starting on page 7-31.

Option 7 will run the Print Source Member (PRTSRCMBR) command. It can be used to print two types of source listings: standard or indented. Refer to page 7-35 for information on the source listing reports.

Use menu option 8 to run the Change Source Type (CHGTYPE) command and update the source attribute for one or more members in a given source file.

Menu option 9 can be used to update the service information for objects on your system. Use it to change the link between an object and its source code. Refer to the description of the Change Service Description (CHGSVCDTA) command (page 7-40) for details.

Use option 10 to convert one of your current PDM option files into a format that is usable by ABSTRACT. Refer to the Convert PDM Option File (CVTPDMF) command on page 7-44.

Option 11 will print a graphic representation of a job stream as a flowchart. Transfers of control and file usage will be documented in an easy to read format. See page 7-45 for a description of the Print Flow Chart (PRTFLWCHT) command that this option runs.

Menu option 12 will print a graphic representation of your current network configuration, including the lines, controllers and devices described to the system. Because this can take an extended time to complete, this option will run the Print Network Configuration (PRTNETCFG) command from batch using the default job description. See the description beginning on page 7-53 for details.

Function Keys

Like the other ABSTRACT menus, the programming tool menus (APTOOL, APRCRTPGM, APCRTPGM) provide a consistent set of function keys:

F3	Exit this ABSTRACT function
F4	Prompt the option or command that is on the command line
F9	Retrieve ‘backwards’ through previously executed commands
F12	Exit the current display and return to the previous function

F14 Submit the option or command on the command line to batch using the default job description defined for this user

F15 Prompt the Submit Job (SBMJOB) command before submitting the option or the command on the command line

F16 Work with options in the user option file

F18 Change the default job description and user option file defined for the current user

F19 Retrieve ‘forwards’ through previously executed commands

F24 Display more function keys

Recreate Program (APRCRTPGM) Menu

This menu accesses the ABSTRACT program recompiling functions to recreate existing program objects from the source indicated by their object service data. Four ABSTRACT commands provide the capability to recompile RPG, COBOL, CL, and auto-report programs in your application library.

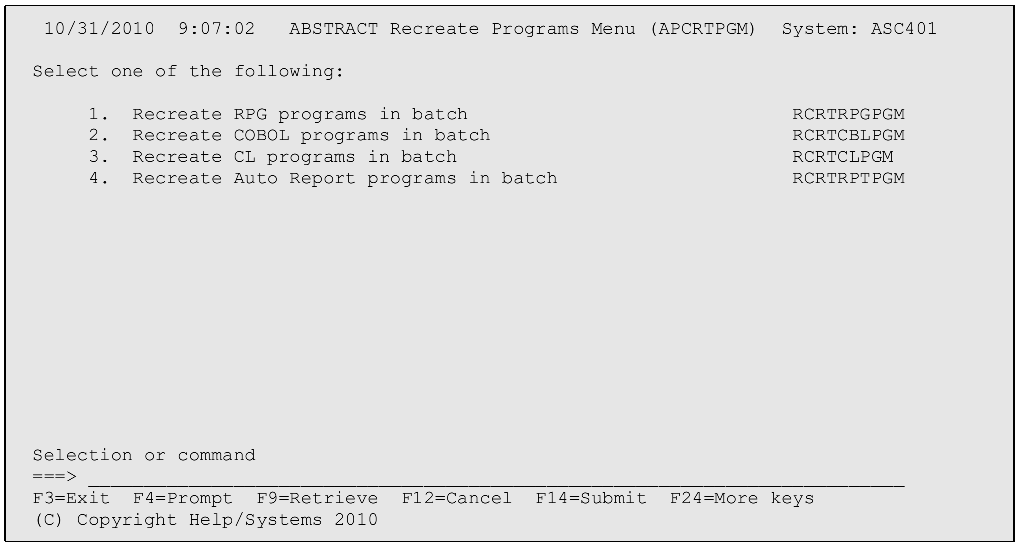
The functions on this menu differ from the functions of the Create Programs (APCRTPGM) menu in that existing program objects, rather than the member names in a source file, determine the source code to be compiled.

Acquire the menu by selecting option 3 from the primary programming tool (APTOOL) menu or by typing

```
GO ABSTRACT/APRCRTPGM
```

on a command entry line and pressing the Enter key.

As with other ABSTRACT menus, the APRCRTPGM menu provides full access to your authorized command set through the command entry line at the bottom of the display.


```
 10/31/2010  9:07:02   ABSTRACT Recreate Programs Menu (APCRTPGM)  System: ASC401

Select one of the following:

     1.  Recreate RPG programs in batch                          RCRTRPGPGM
     2.  Recreate COBOL programs in batch                        RCRTCBLPGM
     3.  Recreate CL programs in batch                           RCRTCLPGM
     4.  Recreate Auto Report programs in batch                  RCRTRPTPGM














Selection or command
===> _________________________________________________________________________
F3=Exit  F4=Prompt  F9=Retrieve  F12=Cancel  F14=Submit  F24=More keys
(C) Copyright Help/Systems 2010
```

Each menu option will prompt (and then run) the corresponding command. Refer to the description for the recompiling commands on page 7-19 for information about the various parameter values you can supply.

Depending on the menu item and the current value of the run in Batch flag, the command will be submitted to the batch job queue or it will begin running immediately when you press the Enter key. Given that some requests can run for an extended period of time, this may not be an efficient use of your display device or interactive priority.

Your request will always be submitted to the batch subsystem if you type the menu option number and press F14 or F15. Refer to the section describing the ABSTRACT menu driver files in the Introduction for information about changing the batch/interactive environment for each menu option.

Refer to page 7-2 for a description of the active function keys and their use.

Create Program (APCRTPGMS) Menu

This menu accesses the ABSTRACT program compiling functions to create all the programs of a given type (RPG, CL, COBOL) within a specified source file. Four ABSTRACT commands provide the capability to compile several programs with a single request.

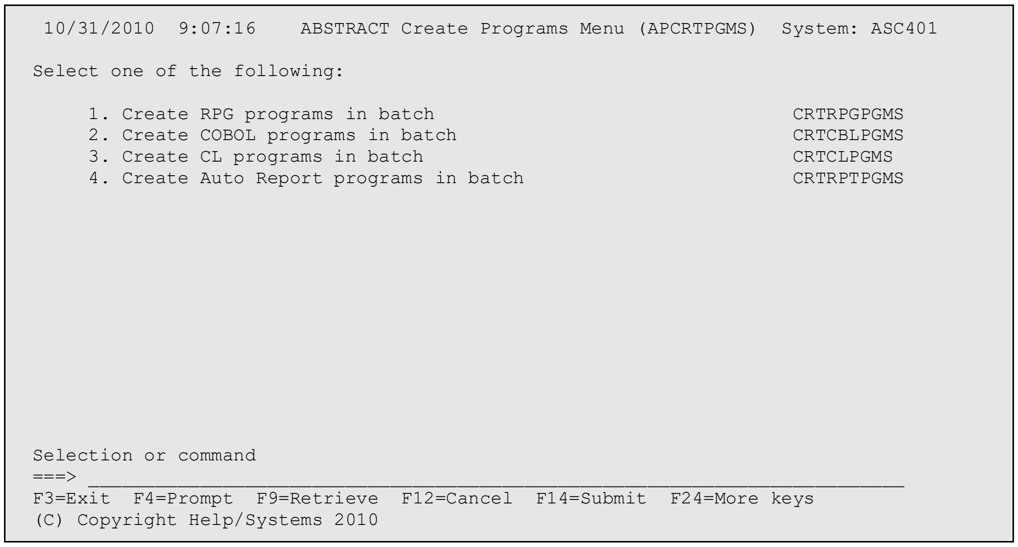
The functions on this menu differ from the functions of the Recreate Programs (APRCRTPGMS) menu in that all source members of the specified type are compiled, rather than recreating the existing program objects within a library.

Acquire the menu by selecting option 4 from the primary programming tool (APTOOL) menu or by typing

```
GO ABSTRACT/APCRTPGMS
```

on a command entry line and pressing the Enter key.

As with other ABSTRACT menus, the APRCRTPGM menu provides full access to your authorized command set through the command entry line at the bottom of the display.



```
 10/31/2010  9:07:16    ABSTRACT Create Programs Menu (APCRTPGMS)  System: ASC401

Select one of the following:

     1. Create RPG programs in batch                                   CRTRPGPGMS
     2. Create COBOL programs in batch                                 CRTCBLPGMS
     3. Create CL programs in batch                                    CRTCLPGMS
     4. Create Auto Report programs in batch                           CRTRPTPGMS











Selection or command
===> __________________________________________________________________________
F3=Exit  F4=Prompt  F9=Retrieve  F12=Cancel  F14=Submit  F24=More keys
 (C) Copyright Help/Systems 2010
```

Each menu option will prompt (and then run) the corresponding command. Refer to the description for the compiling commands on page 7-24 for information about the various parameter values you can supply.

Depending on the menu item and the current value of the run in Batch flag, the command will be submitted to the batch job queue or it will begin running immediately when you press the Enter key. Given that some requests can run for an extended period of time, this may not be an efficient use of your display device or interactive priority.

Your request will always be submitted to the batch subsystem if you type the menu option number and press F14 or F15. Refer to the section describing the ABSTRACT menu driver files in the Introduction for information about changing the batch/interactive environment for each menu option.

Turn to page 7-2 for a description of the active function keys and their use.

Recreate Data Base Relations (RCRTDBR) Command

The Recreate Data Base Relations (RCRTDBR) command will greatly reduce the time and effort required to make changes to a physical file record layout. All current data is preserved, along with file attributes, physical/logical structure, etc. The command does not rely on any cross-reference information, acquiring all information about current data base structure, file attributes, etc. from the files when the command is run.

Access the file recompiler by selecting option 1 of the APTOOL menu or by typing the command on a command entry line. The command can be submitted to a batch subsystem or run from an interactive or batch program.

The command syntax is substantially the same as the syntax for the Create Physical File (CRTPF) command. The parameter descriptions on the following pages are specific to the RCRTDBR command. Refer to the CL Reference Manual - Volume 3 for a complete description of the command parameters and the values they allow.

An ABSTRACT prompt override program will automatically supply values for command parameters if you supply the file name prior to prompting the command, or before pressing function key 9 or 10 from the command prompt display.

```
                    .-*LIBL/---------.       .-generic*_name-.
>>--RCRTDBR----FILE-+----------------+---OBJ-+---------------+-------------------------->
                    +-*library_name/-+       +-*ALL ---------+
                    '-*ALLLOD/-------'       '-*CVTSELECT----'
                                                                                Required
=========================================================================================
                                                                                 Optional

           .-*SAME-------------------.          .-*SAME-------.          .-*MAP---.
>--SRCFILE-+-------------------------+---SRCMBR-+-------------+---RTNDTA-+--------+----->
           +-LIBL/-------------------+          '-member_name-'          +-*CVT---+
           '-*library_name/file_name-'                                   +-*NO----+
                                                                         '-*NOCHK-'

           .-*ALL---.                             .-20-----.             .-*SAME-.
>--NBRRCDS-+--------+---RECLEN-+-number-+---GENLVL-+--------+---FILETYPE-+-------+------>
           '-number-'                             '-number-'             +-*DATA-+
                                                                         '-*SRC--'

        .-*SAME-.           .-*LIBL/-------.   .-*ALL----------.          .-*DBF----.
>--TEXT-+-------+---SUPFILE-+--------------+---+---------------+---RCRTFR-+---------+--->
        '-text--'           +-library_name-+   +-generic*_name-+          +-*ALL----+
                            '-*ALLLOD/-----'   '-*CVTSELECT----'          +-*NONE---+
                                                                          +-*DEVF---+
                                                                          +-*REFFIL-+
                                                                          '-*REFLST-'

            .-*NO----.         .-*FROMLIB-----.          .-*YES-.          .-*PRV---.
>--RCRTPGMR-+--------+---TOLIB-+--------------+---ALWRNM-+------+---DTASET-+--------+--->
            +-*YES---+         +-*CVT---------+          '-*NO--'          +-*FIRST-+
            '*DTAARA-'         '-library_name-'                            '-name---'

          .-*SRCABN---.           .-*SAME--.         .-*SAME--.           .-*SAME---.
>--OPTION-+-----------+---MAXMBRS-+--------+---MAINT-+--------+---RECOVER-+---------+--->
          +-*SOURCE---+           +-*NOMAX-+         +-*IMMED-+           +-*NO-----+
          +-*NOSOURCE-+           '-number-'         +-*DLY---+           +-*IPL----+
          +-*SRC------+                              '-*REBLD-'           '-AFTIPL-'
          '-*NOSRC----'

             .-*SAME-.        .-*SAME------------.            .-*SAME-.
>--FRCACCPTH-+-------+---SIZE-+------------------+---ALLOCATE-+-------+----------------->
             +-*NO---+        +-*NOMAX-----------+            +-*NO---+
             '-*YES--'        '-initial_incr_nbr-'            '-*YES--'

          .-*SAME-.        .-*SAME-.            .-*SAME--.            .-*SAME---.
>--CONTIG-+-------+---UNIT-+-------+---FRCRATIO-+--------+---WAITFILE-+---------+------->
          +-*NO---+'       +-*ANY--+            +-*NONE--+            +-*IMMED--+
          '-*YES--'        '-unit--'            '-number-'            +-*CLS----+
                                                                      '-seconds-'

           .-*SAME---.         .-*SAME-.          .-*SAME--.          .-*SAME-.
>--WAITRCD-+---------+---SHARE-+-------+---DLTPCT-+--------+---ALWUPD-+-------+--------->
           +-*IMMED--+'        +-*YES--+          +-number-+          +-*YES--+
           +-*NOMAX--+         '-*NO---'          '-*NONE--'          '-*NO---'
           '-seconds-'

          .-*SAME-.          .-*SAME-.
>--ALWDLT-+-------+---LVLCHK-+-------+-------------------------------------------------><
          +-*YES--+          +-*YES--+
          '-*NO---'          '-*NO---'
```

FILE Parameter

Specifies the name of the physical file to be recreated. The file must exist in the library indicated and you must have existence, management and operational rights to it. If a single file name is entered, the library name must be specified explicitly or must be the special value *LIBL. Note that the complete list of files that are recompiled by this command depends on the FILE/LIBRARY, SUPFILE and RCRTFR parameters. In particular, the RCRTFR parameter controls the type of related files to include in the compile process.

***ALL -** Recompile all database files in specified library. *ALL can be used with *ALLLOD or with a specific library name to limit the scope of the compile. Files are compiled whether or not the corresponding source member was converted into the conversion library. When the source member does not exist in the conversion library, the file is compiled from the source identified by service data if possible. If service data is incorrect, CU will attempt to find a source member with the same name among the source files in libraries loaded in the cross-reference.

Name - Recompiles the file/library entered. This parameter also supports generic file names. A specific filename can be used with *LIBL or a specific library. A generic file name can only be used with *ALLLOD or with a specific library name.

***CVTSELECT** – not applicable for ABSTRACT

The possible library values are:

***ALLLOD** - Recompiles all physical files included by the name parameter that exist in libraries loaded into the ABSTRACT cross-reference.

***LIBL** - Recompiles all physical files included by the name parameter that exist in libraries on the job's library list. All libraries in the user and system portions of the job's library list are searched for the file or files specified above.

SRCFILE Parameter

This parameter is generally used with the default value of '*SAME, allowing ABSTRACT to automatically locate source code. When the default is overridden, this parameter provides the ability to recreate a single file when the source code cannot be found by normal means. This parameter provides information used by the compile process and is ignored if the request does not involve a file compile. If the FILE parameter includes more than a single file, the SRCFILE parameter provides information only for the first file to be compiled.

***SAME** - ABSTRACT will use service data to obtain the source file name and source library. If source code cannot be located using service data , ABSTRACT will search the libraries loaded in the cross-reference. If source cannot be located by this second method, an error message is recorded in the job log and the summary report will indicate that the file was not compiled.

Name - Enter the name of the source file containing the Data Description Specifications for the file named in the FILE parameter. If a special value *CVTSELECT or *ALL is specified for the file name on the first parameter, the value entered on the source file parameter will only apply to the first file processed.

Library-name -Specifies the name of the library containing the source file.

RTNDTA Parameter

Specifies whether data from the physical file should be retained and if so, how it should be copied into the new file.

***MAP:** Retain the data and use the FMTOPT(*MAP) parameter when copying it into the file to be created. Refer to the Copy File (CPYF) command for a complete description of this parameter.

***NO:** Do not retain the current file data. All data in the physical file will be lost.

***NOCHK:** Retain the data and use the FMTOPT(*NOCHK) parameter when copying it into the file to be created. Refer to the Copy File (CPYF) command for a complete description of this parameter.

***CVT:** Same as *MAP.

SUPFILE Parameter

Specifies a list of additional files that can be recompiled at the same time.

RCRTFR Parameter

Specifies what file relations are to be created. Enter one of the following values:

***DBF**- Recreates the files named in the FILE and SUPFILE parameters along with all associated logical files. Unless this value is specified, the files listed in the FILE and SUPFILE parameters will not be recreated.

***DEVF** - Recreate all device (printer or display) files that depend (via the REF or REFFLD keyword) on any of the files named in the FILE and SUPFILE parameters (along with all associated logical files), and any files included by the *REFFIL/*REFLST values described below.

***REFFIL** - Recreate all files that depend (via the REF or REFFLD keyword) on the files listed in the FILE and SUPFILE parameters and also those files that depend on any of the dependent files identified by this process.

***ALL** - Recreate all physical, logical, and device files impacted by a database change. This option is an easy way to include *DBF, *REFFIL and *DEVF.

Instead of using one or more of the values above, you may choose to specify one of the values listed below: The following values do **not** cause the selected files to be recompiled.

***REFLST** - Build a list of files that includes those named on the FILE and SUPFILE parameters and all of the physical and/or dependent logical files that reference (via the REF or REFFLD keywords) those files. This option is used with RTNDTA(*MAP or *NOCHK) to convert data and/or the RCRTPGMR(*YES) option to recreate an entire group of programs impacted by a database change. This option is useful if database changes were compiled in a separate, previous step.

***NONE -** Do not compile any file or generate any list of file relations. Instead, use the list of files defined by the FILE and SUPFILE parameters while processing the remaining

RCRTPGMR Parameter

This parameter lets you recreate all of the programs that use any of the files in the group specified by the FILE, SUPFILE and RCRTFR parameters. Programs that do not use any file identified by the RCRTDBR process will not generally be compiled. See below and also see the '**TOLIB**' parameter. Programs that neither use files nor have dates passed as parameters cannot be compiled through this process:

***NO** - Do not recreate any dependent programs.

YES - Recreates all dependent programs.

When the program recompilation is completed ABSTRACT will print an audit report showing the attempted compiles and the outcome of each. The audit report will precede the compile listings in the output queue.

The columns on the report identify the program name and the source used to compile it. The result of the compile operation is listed as a Yes ('Y') or No ('N') indicator. The end of the report provides a total of successful and unsuccessful compile counts.

The following example illustrates the appearance of the report.

```
10:22:36                                    RPG RECOMPILER                                PAGE   1
 10/31/2010
Compile from: PILOT          to Object Library: QTEMP

User Profile: *USER

Program    Source     Source Lib Member     Type     Attribute  Compiled?    Object text
ADDTIME    QRPGSRC    PILOT#     ADDTIME    *PGM     RPG        Y
AUTOSCH    QRPGSRC    PILOT#     AUTOSCH    *PGM     RPG        Y
CAL001     QRPGSRC    PILOT#     CAL001     *PGM     RPG        Y            Job schedule by job
CAL002     QRPGSRC    PILOT#     CAL002     *PGM     RPG        Y            Job schedule by date
CAL003     QRPGSRC    PILOT#     CAL003     *PGM     RPG        Y            Current schedule
CHKJOB     QRPGSRC    PILOT#     CHKJOB     *PGM     RPG        Y
CVTHDR     QRPGSRC    PILOT#     CVTHDR     *PGM     RPG        Y
EDTTMPJOB  QRPGSRC    PILOT#     EDTTMPJOB  *PGM     RPG        Y
HLP001     QRPGSRC    PILOT#     HLP001     *PGM     RPG        N            Help text processor
HOLLST     QRPGSRC    PILOT#     HOLLST     *PGM     RPG        Y            Holiday listing
HOLMNT     QRPGSRC    PILOT#     HOLMNT     *PGM     RPG        Y            Holiday definition


  10 COMPILED SUCCESSFULLY
   1 COMPILED UNSUCCESSFULLY
```

TOLIB Parameter

Names the library to receive the recompiled file and program objects.

***FROMLIB**: Recompile the programs back into the library named on the FROMLIB parameter.

***CVT**: Compile the objects into the 'test' library defined within WRKLIBX (usually the library name with 'CU' appended to it).

name: Specify a valid library name.

ALWRNM Parameter

When the TOLIB is not *FROMLIB, this parameter specifies whether or not the base physical file or files can be temporarily renamed. This can be important if compiling from a production library. It is necessary to rename a base physical file if the source code for any of the logical files specifies the library name for the based on physical file. In that case, we rename the original physical, move the new physical file into the base library and compile the new logicals into the target library. After the logicals are compiled, the new physical file is moved to the target library and the old physical file is renamed back to its original name.

<u>*NO</u> - Do not rename the base physical file. ABSTRACT will add the target library to the library list of the compile job before compiling logical files. This option requires that the source code for all logical files reference the related physical file/s without naming the library for the physical file/s.

***YES** - Allow temporary rename of the base physical file. ABSTRACT will rename each physical file while compiling the related logical files. Renaming takes place whether or not the source code for the logical files specifies the library name for the physical.

OPTION Parameter

Controls the level of output from the CRTPF/CRTLF commands issued during the recompile process.

<u>*SRCABN:</u> Retain the source listing only if the CRTxx command fails. If the CRTPF/CRTLF command completes normally, the spool file containing the source listing will be deleted.

***SOURCE:** Always retain the source listings from the CRTxx command.

***NOSOURCE:** Do not generate source listings when the CRTxx commands are executed.

Using the RCRTDBR command

The file recompiler performs the following steps automatically:

Rename the current physical file to preserve its data
Create the new physical file in using the current file attributes
Create all logical files built on the old physical into QTEMP
Add members to the newly created physical
Add members to the newly created logicals, preserving original DTAMBRS structure
Copy data from old physical file to new file
Delete logical files from old physical file
Delete old physical
Move the new logicals from QTEMP to their original library(s)

When device files are recompiled, the following attributes are specifically retained:

`PAGESIZE, LPI, CPI, OVRFLW, LVLCHK, PRTTXT, SHARE, DUPLEX, OUTQ, FORMTYPE, HOLD, SAVE, DEV, PRTQLTY and MAXRCDS`

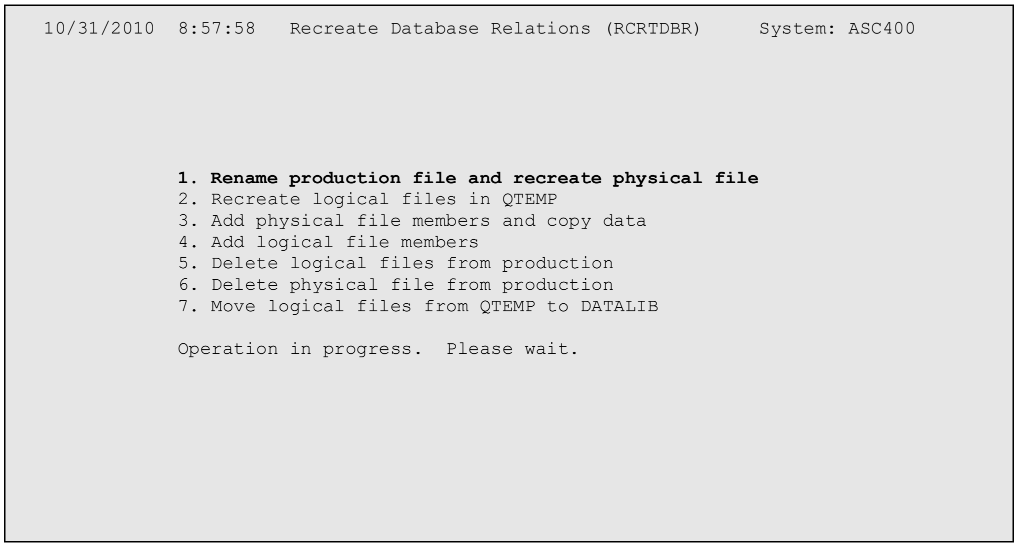
The process will provide a report showing which steps were taken and their outcomes.

Built-in error recovery automatically restores the database to its original status if an error is encountered while compiling the new objects.

The RCRTDBR command can be run interactively or submitted to a batch subsystem. Running the command interactively may require an extended period of time and may not be an efficient use of your device or interactive priority level. You can submit the command to the batch job queue from the APTOOLS menu by selecting option 1 and pressing function key 14 instead of the Enter key. Once you have completed the prompt, the request will be submitted using the job description you have identified through function key 18.

Monitoring the process

If the RCRTDBR command is run interactively, the display below will appear and monitor the process as it progresses. The current step will be highlighted on the display as it occurs.



```
  10/31/2010  8:57:58   Recreate Database Relations (RCRTDBR)     System: ASC400




              1. Rename production file and recreate physical file
              2. Recreate logical files in QTEMP
              3. Add physical file members and copy data
              4. Add logical file members
              5. Delete logical files from production
              6. Delete physical file from production
              7. Move logical files from QTEMP to DATALIB

              Operation in progress.  Please wait.
```

Audit report

When the process completes, a report will be generated showing all of the activity that has taken place. The example below shows each step in the recreation of a physical file named JOBSCH in the PILOT library.

```
15:06:15   File Re-creation Utility                       Page     1
 10/31/2010

JOBSCH     OF PILOT      RENAMED TO FILE150615

COMPILATION OF JOBSCH     OF PILOT      ENDED NORMALLY

COMPILATION OF JOBSCHL1   OF PILOT      ENDED NORMALLY
COMPILATION OF JOBSCHL3   OF PILOT      ENDED NORMALLY
COMPILATION OF JOBSCHL4   OF PILOT      ENDED NORMALLY
COMPILATION OF JOBSCHL2   OF PILOT      ENDED NORMALLY
COMPILATION OF DATEJOIN   OF PILOT      ENDED NORMALLY

PHYSICAL MEMBER JOBSCH     ADDED TO JOBSCH     OF PILOT
DATA COPIED TO JOBSCH     OF JOBSCH     OF PILOT

LOGICAL MEMBER JOBSCHL1   ADDED TO JOBSCHL1   OF PILOT
LOGICAL MEMBER JOBSCHL3   ADDED TO JOBSCHL3   OF PILOT
LOGICAL MEMBER JOBSCHL4   ADDED TO JOBSCHL4   OF PILOT
LOGICAL MEMBER JOBSCHL2   ADDED TO JOBSCHL2   OF PILOT
LOGICAL MEMBER DATEJOIN   ADDED TO DATEJOIN   OF PILOT
```

Recreate Program Relations (RCRTPGMR) Command

ABSTRACT provides a facility that allows you to recompile programs that use a given file. It can greatly reduce your effort when making changes to the application data base by automatically locating and then recompiling the programs using a newly created file definition. You specify the file that you have changed, ABSTRACT does the rest.

The Recreate Program Relations (RCRTPGMR) command uses cross reference information built during the initialization process (refer to the LOADXREF command) in determining the programs that need to be recompiled. The file(s) you name can be either a data base (physical or logical) or a device file.

If you specify a physical file name, ABSTRACT will locate it (using your library list), determine all of the logical files that are dependent on it and recompile any programs that use the physical or one of these logical files. If a logical file name is specified on the RCRTPGMS command, just programs that use it will be recompiled.

Access the RCRTPGMR command by selecting option 2 from the APTOOL menu, or by entering the command on a command entry line.

The command syntax allows you to specify values for these parameters:

```
                      .-50 maximum-.
                      v            |            .-*ALL---------.
>>--RCRTPGMR----FILE---+-file_name-+---FROMFLIB-+--------------+------------------------->
                                                '-library_name-'

                                    .-*ALL-.
>--TOLIB-+-library_name-+---PGMTYPE-+------+--------------------------------------------->
                                    +-*CBL-+
                                    +-*CLP-+
                                    +-*RPG-+
                                    '-*RPT-'

                                                                                 Required
==========================================================================================
                                                                                 Optional

             .-*NO--.          .-9------.             .-*YES-.
>--IGNDECERR-+------+---GENLVL-+--------+---ALWRTVSRC-+------+--------------------------->
             +-*YES-+          '-number-'             '-*NO--'

          .-*SRCABN---.          .-*SAME--.          .-*PRV---.
>--OPTION-+-----------+---USRPRF-+--------+---DTASET-+--------+-------------------------><
          +-*SOURCE---+          +-*USER--+          +-*FIRST-+
          '-*NOSOURCE-'          '-*OWNER-'          '-name---'
```

FILE Parameter

Specifies the name of up to fifty files that must be used in order for a given program to be recompiled. The files can be either database (physical/logical) or a device files. The ABSTRACT program-file cross reference will be searched for programs in the FROMLIB that use any of these files. Any program referencing any of the files you indicate will be recompiled, subject to the value of the PGMTYPE parameter.

If you have named a physical file, ABSTRACT will search your current library list for it and determine all the logical files built over it. Once they have been identified, the ABSTRACT program-file cross reference will be searched and recompile programs in the FROMLIB that use any of these files.

FROMLIB Parameter

Specifies the library to be searched for programs that use the specified file. The current cross reference files will be searched to find programs in the indicated library that use any file indicated by the FILE parameter, or a member in its group. Once the programs have been identified, they will be recompiled and placed into the TOLIB library.

***ALL:** Search all libraries loaded in the cross reference for programs that use any of the files listed in the FILE parameter.

TOLIB Parameter

Names the library to receive the recompiled program objects.

PGMTYPE Parameter

Specifies which of the recompiling commands should be run. Indicate the types of programs to be recompiled once they are identified as using the indicated file and the corresponding Recreate Program Object (RCRTxxxPGM) command will be used to recreate them.

***ALL:** Recompilation will be attempted for all programs that use the indicated file (as shown by the ABSTRACT cross reference dictionary)

***CLP:** Only recompile the CL programs that use the indicated file.

***CBL:** Only recompile the COBOL programs that use the indicated file.

***RPG:** Only recompile the RPG programs that use the indicated file.

***RPT:** Only recompile the RPG auto-report programs that use the indicated file.

IGNDECERR Parameter

If the program type (PGMTYPE) value is *ALL, *CBL, *RPG, or *RPT, this parameter value specifies whether the compiled program should ignore decimal data errors.

***NO:** Decimal data errors are not ignored.

***YES:** Decimal data errors are ignored by the program. The result of the operation being performed when the error occurs is unknown. The compiler prints an error message on the compile listing to notify the user that this option was specified.

GENLVL Parameter

Specifies whether a program is created, depending on the severity of messages created as a result of compile-time errors. If errors occur in a program with a severity level greater than or equal to the value specified in this parameter, the compile operation will end.

9: The default value for the severity level. This is most likely an appropriate value for an RPG program compilation, but may not be appropriate for a COBOL compile.

severity-level: Specify a number ranging from 0 through 99 for the severity level value.

ALWRTVSRC Parameter

If the program type (PGMTYPE) value is *ALL, or *CLP, this parameter value specifies whether the source for a CL program is saved with the program. This source can be retrieved later by using the Retrieve CL Source (RTVCLSRC) command.

***YES:** The source for the CL program is saved with the program object.

***NO:** The source for the CL program is not saved with the program object.

OPTION Parameter

Controls the level of output from the compile commands issued during the recompile process.

***SRCABN:** Retain the source listing only if the compile command fails. If the compile completes normally, the spool file containing the source listing will be deleted.

***SOURCE:** Always retain the source listings from the CRTxxxPGM command.

***NOSOURCE:** Do not generate source listings when the CRTxxxPGM commands are executed.

USRPRF Parameter

Specifies whether the authority checking done while this program is running should include only the user who is running the program (*USER) or both the user and the program's owner (*OWNER). The profiles of either the program user or both the program user and the program owner are used to control which objects can be used by the program, including the authority the program has for each object.

***SAME:** The program's current user profile value is retained for the new program object.

***USER:** The program user's user profile is used when the program is processed.

***OWNER:** The user profiles of both the program owner and the program's user are used when the program is processed. The collective sets of object authority in both user profiles are used to find and access objects during program processing. Authority from the owning user profile's group profile is not included in the authority for the running program.

DTASET Parameter

Specify the cross reference data set to be used in acquiring the program reference information.

***PRV:** Use the value for this parameter from the previous session. If no previous session has occurred, use *FIRST.

***FIRST:** information from the first (default) data set should be accessed.

Name: Specify any existing data set. Use the command prompter to find the names available to you.

Examples

```
RCRTPGMR FILE(CUSTMAST) FROMLIB(PRODLIB) TOLIB(NEWPGMS)
```

All programs in the PRODLIB library that use the CUSTMAST file will be recompiled to the NEW-PGMS library. If CUSTMAST is a physical data base file, ABSTRACT will identify the logical files built over it and recompile them to the NEWPGMS library as well.

```
RCRTPGMR FILE(ORDENTFM) FROMLIB(OELIB) TOLIB(OELIB) +
      USRPRF(*OWNER)
```

All programs in the OELIB library that use the ORDENTFM file will be recompiled with the USR-PRF(*OWNER) attribute. The new programs will replace the old ones in the library. If ORDENTFM is a physical data base file, ABSTRACT will identify the logical files built over it and recompile them too.

Audit Trail Report

When the recompilation work is completed, the Recreate Program Object (RCRTxxxPGM) commands that were used will print an audit report showing the attempted compiles and the outcome of each. The audit report will precede the compile listings in the output queue. A separate report will be printed for each different type of program (CLP, CBL, etc) that is processed by the RCRTPGMR command.

The example on the next page shows output from the Recreate CL Program Object (RCRTCLPGM) command. The columns on the report identify the program name and the source used to compile it. The result of the compile operation is listed as a Yes (‘Y’) or No (‘N’) indicator. The file that caused the program to be recompiled is also listed along with the program’s text description. The end of the report provides a total of successful and unsuccessful compile counts.

The sample below is just one of the audit reports produced by the following command:

```
RCRTPGMR FILE(OBJREF) FROMLIB(ABSTRACT) TOLIB(QTEMP) TYPE(*ALL)
```

Each program shown on the listing is an object in the ABSTRACT library that uses either the OBJREF file itself or one of the logical files built on it. The file in this file group that is referenced by the program is shown to the left of the program's text.

```
 9:48:35                           CL PROGRAM RECOMPILER                               PAGE   1
 10/31/2010
Compile from: ABSTRACT          to Object Library: QTEMP

User Profile: *USER


Program    Source    Source Lib Member     Type    Attribute  Compiled?   Object text
ADDDTASET  QCLSRC    APLUS#     ADDDTASET  *PGM    CLP        Y OBJREF
CLRDTASET  QCLSRC    APLUS#     CLRDTASET  *PGM    CLP        Y OBJREF    Clear data set
LOADCMDC   QCLSRC    APLUS#     LOADCMDC   *PGM    CLP        Y RLTLTN
LOADMENU   QCLSRC    APLUS#     LOADMENU   *PGM    CLP        Y CMD
LOADQRYC   QCLSRC    APLUS#     LOADQRYC   *PGM    CLP        Y RLTLTN
LOADXRFC   QCLSRC    APLUS#     LOADXRFC   *PGM    CLP        Y OBJREF
LOAD300C2  QCLSRC    APLUS#     LOAD300C2  *PGM    CLP        Y OBJREF    Member added by copy file
LSTSETCL   QCLSRC    APLUS#     LSTSETCL   *PGM    CLP        Y OBJREF    List data sets available to user
PRTFLWCHT  QCLSRC    APLUS#     PRTFLWCHT  *PGM    CLP        Y PRGPRGL3  Print Printer File Layout
PRTOBJEXCP QCLSRC    APLUS#     PRTOBJEXCP *PGM    CLP        Y RLTLTN    Print object exceptions
RGZFILESC  QCLSRC    APLUS#     RGZFILESC  *PGM    CLP        Y OBJREF
RMVDTASET  QCLSRC    APLUS#     RMVDTASET  *PGM    CLP        Y OBJREF
WRKFGUCL   QCLSRC    APLUS#     WRKFGUCL   *PGM    CLP        Y PGMLIB    CL front end for WRKFGU command
WRKFLDGUCL QCLSRC    APLUS#     WRKFLDGUCL *PGM    CLP        Y PGMLIB    CL front end for WRKFLDGU cmd
WRKLIBX    QCLSRC    APLUS#     WRKLIBX    *PGM    CLP        Y ALLFL     Work with libraries in X-Ref
WRKOBJRCL  QCLSRC    APLUS#     WRKOBJRCL  *PGM    CLP        Y PGMLIB    CL front end for WRKOBJR command
WRKOBJRSCL QCLSRC    APLUS#     WRKOBJRSCL *PGM    CLP        Y PGMLIB    CL front end for WRKOBJRS cmd
WRKOBJRXCL QCLSRC    APLUS#     WRKOBJRXCL *PGM    CLP        Y PGMLIB    CL front end for WRKOBJRX cmd
WRKOBJUCL  QCLSRC    APLUS#     WRKOBJUCL  *PGM    CLP        Y PGMLIB    CL front end for WRKOBJU command
WRKOBJUXCL QCLSRC    APLUS#     WRKOBJUXCL *PGM    CLP        Y PGMLIB    CL front end for WRKOBJUX cmd

   20 COMPILED SUCCESSFULLY
    0 COMPILED UNSUCCESSFULLY
```

Recreate Program Object (RCRTxxxPGM) Commands

The ABSTRACT RCRTxxxPGM commands allow you to recompile **existing** program objects of the specified type with a single request.

Each of the Recreate Program commands is accessible from the Recreate Programs (APRCRTPGM) menu. See page 7-2 for a description of the menu and its options. The commands can also be run immediately or submitted to batch by using a command entry line.

Given that the recompilation step may involve many programs of varying complexity, you should normally choose to submit these commands to a batch subsystem for execution.

Each of the four Recreate Program commands has similar syntax and command parameters. The Recreate RPG Program (RCRTRPGPGM) and Recreate AutoReport Program (RCRTRPTPGM) commands are identical.

```
                      .-*CURLIB------.   .-*ALL---------.
>>--RCRTRPGPGM----PGM-+--------------+---+--------------+------------------------------>
                      '-library_name-'   '-generic_name-'

                                                                                Required
==========================================================================================
                                                                                Optional

         .-*FROMLIB-----.          .-9------.          .-*SRCABN---.
>--TOLIB-+--------------+---GENLVL-+--------+---OPTION-+-----------+-------------------->
         '-library_name-'          '-number-'          +-*SOURCE---+
                                                       '-*NOSOURCE-'

             .-*NO--.          .-*SAME--.
>--IGNDECERR-+------+---USRPRF-+--------+-------------------------------------------><
             '-*YES-'          +-*USER--+
                               '-*OWNER-'
```

```
                      .-*CURLIB------.   .-*ALL---------.
>>--RCRTCBLPGM----PGM-+--------------+---+--------------+------------------------------>
                      '-library_name-'   '-generic_name-'

                                                                                Required
==========================================================================================
                                                                                Optional

         .-*FROMLIB-----.          .-29-----.          .-*SRCABN---.
>--TOLIB-+--------------+---GENLVL-+--------+---OPTION-+-----------+-------------------->
         '-library_name-'          '-number-'          +-*SOURCE---+
                                                       '-*NOSOURCE-'

          .-*SAME--.
>--USRPRF-+--------+------------------------------------------------------------------><
          +-*USER--+
          '-*OWNER-'
```

```
                     .-*CURLIB------.   .-*ALL---------.
>>--RCRTCLPGM----PGM-+--------------+---+--------------+---------------------------------->
                     '-library_name-'   '-generic_name-'

                                                                                  Required
==========================================================================================
                                                                                  Optional

         .-*FROMLIB-----.          .-*SRCABN---.
>--TOLIB-+--------------+---OPTION-+-----------+------------------------------------------>
         '-library_name-'          +-*SOURCE---+
                                   '-*NOSOURCE-'

             .-*YES-.          .-*SAME--.
>--ALWRTVSRC-+------+---USRPRF-+--------+----------------------------------------------><
             '-*NO--'          +-*USER--+
                               '-*OWNER-'
```

Each command will search the indicated library for program objects of the given type (RPG, COBOL, CLP, RPT) and attempt to recompile them to the target library you have specified. Object service data for the current program objects will be referenced to locate the associated source code.

PGM Parameter

Specify the qualified name of the program(s) to be recompiled. A generic name (leading characters followed by an asterisk) can be specified to select objects with similar names. Specify a library name to be searched for the program(s) indicated or use the *CURLIB special value.

***CURLIB:** recompile all of the programs in your job’s current library that meet the name criteria you specify.

***ALL:** recompile all of the programs in the specified library having the type indicated by the RCRTxxxPGM command (RPG, CBL, CLP, RPT).

TOLIB Parameter

Specify the name of the library to receive the compiled program objects. Each object that is created will be placed into the library specified by this parameter.

***FROMLIB:** place the compiled programs into the same library specified by the PGM parameter. The new program objects will replace the existing programs.

name: recompile the programs to the specified library. The existing programs will be retained in the library specified by the PGM parameter and new ones will be created and placed into the named library.

IGNDECERR Parameter

If you have selected the RCRTRPGPGM or the RCRTRPTPGM commands to recompile RPG or RPG auto-report programs, this parameter value specifies whether the compiled program should ignore decimal data errors.

***NO:** Decimal data errors are not ignored.

***YES:** Decimal data errors are ignored by the program. The result of the operation being performed when the error occurs is unknown. The compiler prints an error message on the compile listing to notify the user that this option was specified.

GENLVL Parameter

Specifies whether a program is created, depending on the severity of messages created as a result of compile-time errors. If errors occur in a program with a severity level greater than or equal to the value specified in this parameter, the compile operation will end.

The default value for the severity level depends on the compiler you have selected. The default for COBOL program compilation is 29. The default for RPG and RPG auto-report compiles is 9.

severity-level: Specify a number ranging from 0 through 99 for the severity level value.

ALWRTVSRC Parameter

If you have selected the RCRTCLPGM command to recompile CL programs, this parameter value specifies whether the source for a CL program is saved with the program. This source can be retrieved later by using the Retrieve CL Source (RTVCLSRC) command.

<u>***YES:**</u> The source for the CL program is saved with the program object.

***NO:** The source for the CL program is not saved with the program object.

OPTION Parameter

Controls the level of output from the compilers during the recompile process.

<u>***SRCABN:**</u> Retain the source listing only if the compile fails. If the compile completes normally, the spool file containing the source listing will be deleted.

***SOURCE:** Always retain the source listings from the CRTxxxPGM command.

***NOSOURCE:** Do not generate source listings when the CRTxxxPGM commands are executed.

USRPRF Parameter

Specifies whether the authority checking done while this program is running should include only the user who is running the program (*USER) or both the user and the program's owner (*OWNER). The profiles of either the program user or both the program user and the program owner are used to control which objects can be used by the program, including the authority the program has for each object.

<u>***SAME:**</u> The program's current user profile attribute is retained for the new program object.

***USER:** The program user's user profile is used when the program is processed.

***OWNER:** The user profiles of both the program owner and the program's user are used when the program is processed. The collective sets of object authority in both user profiles are used to find and access objects during program processing. Authority from the owning user profile's group profile is not included in the authority for the running program.

Examples

All RPG programs in the SEQUELEX library can be recreated from their source code (regardless of the library it is located in) by using the command:

```
RCRTRPGPGM PGM(SEQUELEX/*ALL)
```

CL programs in the library TESTLIB starting with INST will be recompiled to the DISTLIB library if the following command is run. The RTVCLSRC command will not work against the new program objects.

```
RCRTCLPGM PGM(TESTLIB/INST*) TOLIB(DISTLIB) ALWRTVSRC(*NO)
```

Audit Trail Report

When the recompilation work is completed, each Recreate Program Object (RCRTxxxPGM) command will print an audit report showing the attempted compiles and the outcome of each. The audit report will precede the compile listings in the output queue.

The example below shows output from the Recreate RPG Program Object (RCRTRPGPGM) command. The columns on the report identify the program name and the source used to compile it. The result of the compile operation is listed as a Yes (‘Y’) or No (‘N’) indicator. The end of the report provides a total of successful and unsuccessful compile counts.

The sample below is an audit report produced by the following command:

```
RCRTRPGPGM PGM(ABSTRACT/*ALL)  TOLIB(QTEMP)  OPTION(*SRCABN)
```

Each program shown on the listing is an RPG program object in the ABSTRACT library. Using OPTION(*SRCABN) causes source listings to be removed from the output queue if the program compiles successfully. Based on the audit report below, there should be only four compile listings left in the output queue when the command completes.

```
10:22:36                                    RPG RECOMPILER                                    PAGE   1
 10/31/2010
Compile from: PILOT          to Object Library: QTEMP

User Profile: *USER



Program    Source     Source Lib Member     Type     Attribute  Compiled?    Object text
ADDTIME    QRPGSRC    PILOT#     ADDTIME    *PGM     RPG        Y
AUTOSCH    QRPGSRC    PILOT#     AUTOSCH    *PGM     RPG        Y
CAL001     QRPGSRC    PILOT#     CAL001     *PGM     RPG        Y            Job schedule by job
CAL002     QRPGSRC    PILOT#     CAL002     *PGM     RPG        Y            Job schedule by date
CAL003     QRPGSRC    PILOT#     CAL003     *PGM     RPG        Y            Current schedule
CHKJOB     QRPGSRC    PILOT#     CHKJOB     *PGM     RPG        Y
CVTHDR     QRPGSRC    PILOT#     CVTHDR     *PGM     RPG        Y
EDTTMPJOB  QRPGSRC    PILOT#     EDTTMPJOB  *PGM     RPG        Y
HLP001     QRPGSRC    PILOT#     HLP001     *PGM     RPG        N            Help text processor
HOLLST     QRPGSRC    PILOT#     HOLLST     *PGM     RPG        Y            Holiday listing
HOLMNT     QRPGSRC    PILOT#     HOLMNT     *PGM     RPG        Y            Holiday definition
JOBCHK     QRPGSRC    PILOT#     JOBCHK     *PGM     RPG        Y            Scheduling - Check for jobs
JOBEXE     QRPGSRC    PILOT#     JOBEXE     *PGM     RPG        N            Job Execution
JOBLST     QRPGSRC    PILOT#     JOBLST     *PGM     RPG        Y            Job definition listing
JOBMNT     QRPGSRC    PILOT#     JOBMNT     *PGM     RPG        Y            Job definition prompting
JOBSBMT2   QRPGSRC    PILOT#     JOBSBMT2   *PGM     RPG        Y
PERLST     QRPGSRC    PILOT#     PERLST     *PGM     RPG        Y            Period code listing
PERMNT     QRPGSRC    PILOT#     PERMNT     *PGM     RPG        Y            Period code definition
PRG001     QRPGSRC    PILOT#     PRG001     *PGM     RPG        Y            Schedule purge prompt
PRG002     QRPGSRC    PILOT#     PRG002     *PGM     RPG        Y            Schedule purge processor
READLOG    QRPGSRC    PILOT#     READLOG    *PGM     RPG        Y            Job log reader
REPORTS    QRPGSRC    PILOT#     REPORTS    *PGM     RPG        Y            Report request
RNMJOB     QRPGSRC    PILOT#     RNMJOB     *PGM     RPG        Y
RPT001     QRPGSRC    PILOT#     RPT001     *PGM     RPG        Y            Pilot distribution by job
RPT002     QRPGSRC    PILOT#     RPT002     *PGM     RPG        N            Report distribution by dept
RPT003     QRPGSRC    PILOT#     RPT003     *PGM     RPG        Y            Report distribution by date
SCHMNT     QRPGSRC    PILOT#     SCHMNT     *PGM     RPG        Y            Schedule maintenance
SCH001     QRPGSRC    PILOT#     SCH001     *PGM     RPG        Y            Schedule loader
SEC900     QRPGSRC    PILOT#     SEC900     *PGM     RPG        Y
SEC910     QRPGSRC    PILOT#     SEC910     *PGM     RPG        N


  26 COMPILED SUCCESSFULLY
   4 COMPILED UNSUCCESSFULLY
```

Create Programs From Source (CRTxxxPGMS)

The ABSTRACT CRTxxxPGMS commands allow you to create programs from several members in a source file with a single request.

Each of the Create Program commands is accessible from the Create Programs (APCRTPGMS) menu. See page 7-2 for a description of the menu and its options. The commands can also be run immediately or submitted to batch by using a command entry line.

Given that the compilation step may involve many programs of varying complexity, you should normally choose to submit these commands to a batch subsystem for execution.

Each of the four Create Program commands has similar syntax and command parameters. The Create RPG Programs (CRTRPGPGMS) and Create AutoReport Programs (CRTRPTPGMS) commands are identical.

```
                           .-*CURLIB------.   .-QRPGSRC---.       .-*ALL---------.
>>--CRTRPGPGMS----SRCFILE-+--------------+---+-----------+---MBR-+--------------+------->
                           '-library_name-'   '-file_name-'       '-member_name--'

                                                                                Required
==========================================================================================
                                                                                Optional

         .-*FROMLIB-----.          .-9------.          .-*SRCABN---.
>--TOLIB-+--------------+---GENLVL-+--------+---OPTION-+-----------+-------------------->
         '-library_name-'          '-number-'          +-*SOURCE---+
                                                       '-*NOSOURCE-'

             .-*NO--.          .-*SAME--.
>--IGNDECERR-+------+---USRPRF-+--------+--------------------------------------------><
             '-*YES-'          +-*USER--+
                               '-*OWNER-'
```

```
                           .-*CURLIB------.   .-QCBLSRC---.       .-*ALL---------.
>>--CRTCBLPGMS----SRCFILE-+--------------+---+-----------+---MBR-+--------------+------->
                           '-library_name-'   '-file_name-'       '-member_name--'

                                                                                Required
==========================================================================================
                                                                                Optional

         .-*FROMLIB-----.          .-29-----.          .-*SRCABN---.
>--TOLIB-+--------------+---GENLVL-+--------+---OPTION-+-----------+-------------------->
         '-library_name-'          '-number-'          +-*SOURCE---+
                                                       '-*NOSOURCE-'

          .-*SAME--.
>--USRPRF-+--------+-----------------------------------------------------------------><
          +-*USER--+
          '-*OWNER-'
```

```
                          .-*CURLIB------.   .-QRPGSRC---.       .-*ALL---------.
>>--CRTCLPGMS----SRCFILE-+--------------+---+-----------+---MBR-+--------------+-------->
                          '-library_name-'   '-file_name-'       '-member_name--'

                                                                                 Required
=========================================================================================
                                                                                 Optional

         .-*FROMLIB-----.          .-*SRCABN---.
>--TOLIB-+--------------+---OPTION-+-----------+--------------------------------------->
         '-library_name-'          +-*SOURCE---+
                                   '-*NOSOURCE-'

             .-*YES-.          .-*SAME--.
>--ALWRTVSRC-+------+---USRPRF-+--------+-------------------------------------------><
             '-*NO--'          +-*USER--+
                               '-*OWNER-'
```

SRCFILE Parameter

Specify the qualified name of the source file that contains the code for the programs to be generated. The default source file name depends on the command used. The library list will be searched for the source file if you do not explicitly qualify the file name.

MBR Parameter

Specify the name of the member(s) to be compiled.

***ALL:** All members in the source file meeting the type specified on the command (RPG, CBL, CLP, or RPT) will be compiled to the target library.

name: specify the individual member to be compiled.

TOLIB Parameter

Specify the name of the library to receive the compiled program objects. Each object that is created will be placed into the library specified by this parameter.

***SRCLIB:** place the compiled programs into the same library containing the source file specified by the SRCFILE parameter.

name: compile the programs to the specified library.

IGNDECERR Parameter

If you have selected the CRTRPGPGMS or the CRTRPTPGMS commands to compile RPG or RPG auto-report programs, this parameter value specifies whether the compiled program should ignore decimal data errors.

***NO:** Decimal data errors are not ignored.

***YES:** Decimal data errors are ignored by the program. The result of the operation being performed when the error occurs is unknown. The compiler prints an error message on the compile listing to notify the user that this option was specified.

GENLVL Parameter

Specifies whether a program is created, depending on the severity of messages created as a result of compile-time errors. If errors occur in a program with a severity level greater than or equal to the value specified in this parameter, the compile operation will end.

The default value for the severity level depends on the compiler you have selected. The default for COBOL program compilation is 29. The default for RPG and RPG auto-report compiles is 9.

severity-level: Specify a number ranging from 0 through 99 for the severity level value.

ALWRTVSRC Parameter

If you have selected the CRTCLPGMS command to compile CL programs, this parameter value specifies whether the source for a CL program is saved with the program. This source can be retrieved later by using the Retrieve CL Source (RTVCLSRC) command.

***YES:** The source for the CL program is saved with the program object.

***NO:** The source for the CL program is not saved with the program object.

OPTION Parameter

Controls the level of output from the compilers during the compile process.

***SRCABN:** Retain the source listing only if the compile fails. If the compile completes normally, the spool file containing the source listing will be deleted.

***SOURCE:** Always retain the source listings from the CRTxxxPGM command.

***NOSOURCE:** Do not generate source listings when the CRTxxxPGM commands are executed.

USRPRF Parameter

Specifies whether the authority checking done while this program is running should include only the user who is running the program (*USER) or both the user and the program's owner (*OWNER). The profiles of either the program user or both the program user and the program owner are used to control which objects can be used by the program, including the authority the program has for each object.

***SAME:** The program's current user profile attribute is retained for the new program object.

***USER:** The program user's user profile is used when the program is processed.

***OWNER:** The user profiles of both the program owner and the program's user are used when the program is processed. The collective sets of object authority in both user profiles are used to find and access objects during program processing. Authority from the owning user profile's group profile is not included in the authority for the running program.

Examples

All RPG members in the QRPGSRC source file in the SEQUELEX library can be compiled by using the command below. The program objects will be placed into the SEQUELEX library.

```
CRTRPGPGMS SRCFILE(SEQUELEX/QRPGSRC)
```

CLP members in the QCLSRC file in library SRCLIB will be compiled to the DISTLIB library if the following command is run. The RTVCLSRC command will not work against the new program objects.

```
CRTCLPGMS SRCFILE(TESTLIB/QCLSRC) TOLIB(DISTLIB) +
     ALWRTVSRC(*NO)
```

Audit Trail Report

When the recompilation work is completed, each Create Program From Source (CRTxxxPGMS) command will print an audit report showing the attempted compiles and the outcome of each. The audit report will precede the compile listings in the output queue.

The example below shows output from the Recreate RPG Program Object (CRTRPGPGMS) command. The columns on the report identify the program name and the source used to compile it. The result of the compile operation is listed as a Yes ('Y') or No ('N') indicator. The end of the report provides a total of successful and unsuccessful compile counts.

The sample below is an audit report produced by the following command:

```
CRTRPGPGMS PGM(PILOT#/QRPGSRC)  TOLIB(QTEMP)  OPTION(*SRCABN)
```

Each item shown on the listing is a member within the QRPGSRC source file in the PILOT# library. Using OPTION(*SRCABN) causes source listings to be removed from the output queue if the program compiles successfully.

```
10:22:36                                   RPG RECOMPILER                                PAGE   1
 10/31/2010
Compile from: PILOT          to Object Library: QTEMP

User Profile: *USER



Program     Source      Source Lib Member      Type     Attribute  Compiled?    Object text
ADDTIME     QRPGSRC     PILOT#     ADDTIME     *PGM     RPG        Y
AUTOSCH     QRPGSRC     PILOT#     AUTOSCH     *PGM     RPG        Y
CAL001      QRPGSRC     PILOT#     CAL001      *PGM     RPG        Y            Job schedule by job
CAL002      QRPGSRC     PILOT#     CAL002      *PGM     RPG        Y            Job schedule by date
CAL003      QRPGSRC     PILOT#     CAL003      *PGM     RPG        Y            Current schedule
CHKJOB      QRPGSRC     PILOT#     CHKJOB      *PGM     RPG        Y
CVTHDR      QRPGSRC     PILOT#     CVTHDR      *PGM     RPG        Y
EDTTMPJOB   QRPGSRC     PILOT#     EDTTMPJOB   *PGM     RPG        Y
HLP001      QRPGSRC     PILOT#     HLP001      *PGM     RPG        N            Help text processor
HOLLST      QRPGSRC     PILOT#     HOLLST      *PGM     RPG        Y            Holiday listing
HOLMNT      QRPGSRC     PILOT#     HOLMNT      *PGM     RPG        Y            Holiday definition
JOBCHK      QRPGSRC     PILOT#     JOBCHK      *PGM     RPG        Y            Scheduling - Check for jobs
JOBEXE      QRPGSRC     PILOT#     JOBEXE      *PGM     RPG        N            Job Execution
JOBLST      QRPGSRC     PILOT#     JOBLST      *PGM     RPG        Y            Job definition listing
JOBMNT      QRPGSRC     PILOT#     JOBMNT      *PGM     RPG        Y            Job definition prompting
JOBSBMT2    QRPGSRC     PILOT#     JOBSBMT2    *PGM     RPG        Y
PERLST      QRPGSRC     PILOT#     PERLST      *PGM     RPG        Y            Period code listing
PERMNT      QRPGSRC     PILOT#     PERMNT      *PGM     RPG        Y            Period code definition
PRG001      QRPGSRC     PILOT#     PRG001      *PGM     RPG        Y            Schedule purge prompt
PRG002      QRPGSRC     PILOT#     PRG002      *PGM     RPG        Y            Schedule purge processor
READLOG     QRPGSRC     PILOT#     READLOG     *PGM     RPG        Y            Job log reader
REPORTS     QRPGSRC     PILOT#     REPORTS     *PGM     RPG        Y            Report request
RNMJOB      QRPGSRC     PILOT#     RNMJOB      *PGM     RPG        Y
RPT001      QRPGSRC     PILOT#     RPT001      *PGM     RPG        Y            Pilot distribution by job
RPT002      QRPGSRC     PILOT#     RPT002      *PGM     RPG        N            Report distribution by dept
RPT003      QRPGSRC     PILOT#     RPT003      *PGM     RPG        Y            Report distribution by date
SCHMNT      QRPGSRC     PILOT#     SCHMNT      *PGM     RPG        Y            Schedule maintenance
SCH001      QRPGSRC     PILOT#     SCH001      *PGM     RPG        Y            Schedule loader
SEC900      QRPGSRC     PILOT#     SEC900      *PGM     RPG        Y
SEC910      QRPGSRC     PILOT#     SEC910      *PGM     RPG        N


  26 COMPILED SUCCESSFULLY
   4 COMPILED UNSUCCESSFULLY
```

Print Program Description (PRTPGMD) Command

ABSTRACT can print a formatted report listing pertinent information about the program objects in your application library. This report is not dependent on any cross reference information.

The report lists the following for each program object in the library:

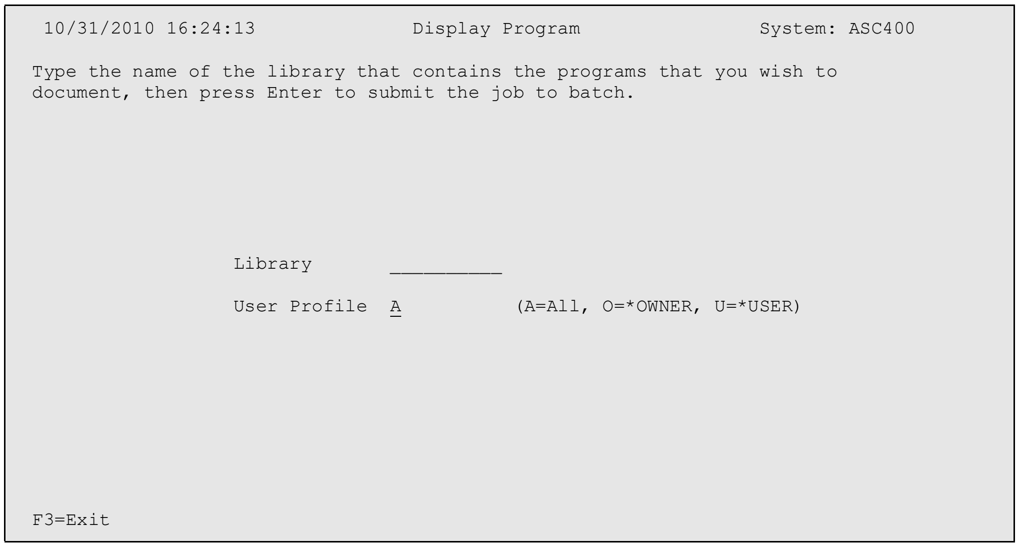
Program name
Program type (RPG, CBL, CL, QRY, etc.)
User profile (*OWNER or *USER)
Owning user profile
Program size
Source file and member used to create the program
Number of parameters accepted by the program
Total size of executable storage (PSSA+PASA)
Number of MI instructions in the program template
Number of ODT references associated with the program
Compiler used to create the program

The Print Program Description (PRTPGMD) command invokes an interactive program that prompts you for the name of the library you want to document and the types of programs that should be included on the report. Access the PRTPGMD command by selecting option 5 of the APTOOL menu or from a command entry line. The PRTPGMD command has no parameters.

```
>>--PRTPGMD---------------------------------------------------------------------------><
```

Once the command is invoked, a prompt display will appear requesting you to provide the name of the library you want the report to document and whether all programs or only programs with USR-PRF(*OWNER) or USRPRF(*USER) should be listed.

The prompt display will be similar to the one shown on the next page.



```
 10/31/2010 16:24:13             Display Program               System: ASC400

Type the name of the library that contains the programs that you wish to
document, then press Enter to submit the job to batch.






                  Library       __________

                  User Profile  A          (A=All, O=*OWNER, U=*USER)







F3=Exit
```

Type the name of a library to which you have *USE authority and select the types of programs to be included. If you choose to include all programs, no USRPRF selection will be performed. Otherwise choose the type of programs that should be included by typing an ‘O’ or a ‘U’ into the prompt.

The USRPRF user profile attribute of each program object specifies whether the authority checking done while the program is running should include only the user who is running the program (*USER) or both the user and the program’s owner (*OWNER). The profiles of either the program user or both the program user and the program owner are used to control which objects can be used by the program, including the authority the program has for each object.

Press Enter to submit the request to the batch job queue. The generated report will look like the one below.

The Print Program Description Report

```
15:57:49                         Program Descriptions                                          Page 1
 10/31/2010

 Library: ABSTRACT

Program    Type    User Prof. Owner    Size Source File            Parms    Stg Inst Refs  Compiler
ABPCLR     CLP     *OWNER     ANDREW   7168 QCLSRC  ANDREW  ABPCLR           464   25   71  5714SS1 R03M00
ABPSEND    CLP     *OWNER     ANDREW   8192 QCLSRC  ANDREW  ABPSEND      3   784   38   87  5714SS1 R03M00
ABSTRACT   CLP     *USER      ANDREW  62976 QCLSRC  APLUS   ABSTRACT     1  4560  964  616  5714SS1 R03M00
ADDLFMR    RPG     *USER      ANDREW  58368 QRPGSRC TOOLS#  ADDLFMR        12896 2505       5728RG1 R01M00
ADDPFMC    CLP     *USER      ANDREW   6656 QCLSRC  TOOLS#  ADDPFMC          736  132       5714SS1 R01M00
ALLOC      CLP     *USER      ANDREW   3584 QCLSRC  TOOLS#  ALLOC            480   42       5714SS1 R01M00
ANZFILE    CLP     *USER      ANDREW  15360 QCLSRC  ANDREW  WRKFD        1  1760  129  144  5714SS1 R03M00
APLUS      RPG     *USER      ANDREW  90624 QRPGSRC APLUS   APLUS        3 11232 1977 2125  5728RG1 R03M00
CACHECK    CLP     *USER      ANDREW  13312 QCLSRC  ANDREW  CACHECK      2  3584   25  110  5714SS1 R03M00
CASSIST    MI      *USER      QSECOFR 18432 QIRPSRC SEQUEL# CASSIST      3  1520  121  366  ASC_ASM R01M00
CHGSVCD    MI      *USER      ANDREW  10240 QIRPSRC TOOLS#  CHGSVCD      5   704   95  116  ASC_ASM R01M00
CHGTYPE    RPG     *USER      QSYS     5632 QRPGSRC TOOLS#  CHGTYPE        12528  249       5728RG1 R01M00
CHKOBJ     CLP     *USER      ANDREW   8704 QCLSRC  ANDREW  CHKOBJ       5   624   59  108  5714SS1 R03M00
CHKSRC     CLP     *USER      ANDREW   6144 QCLSRC  ABSTRAC CHKSRC           624  162       5714SS1 R01M00
```

Find String (FNDSTR) Command

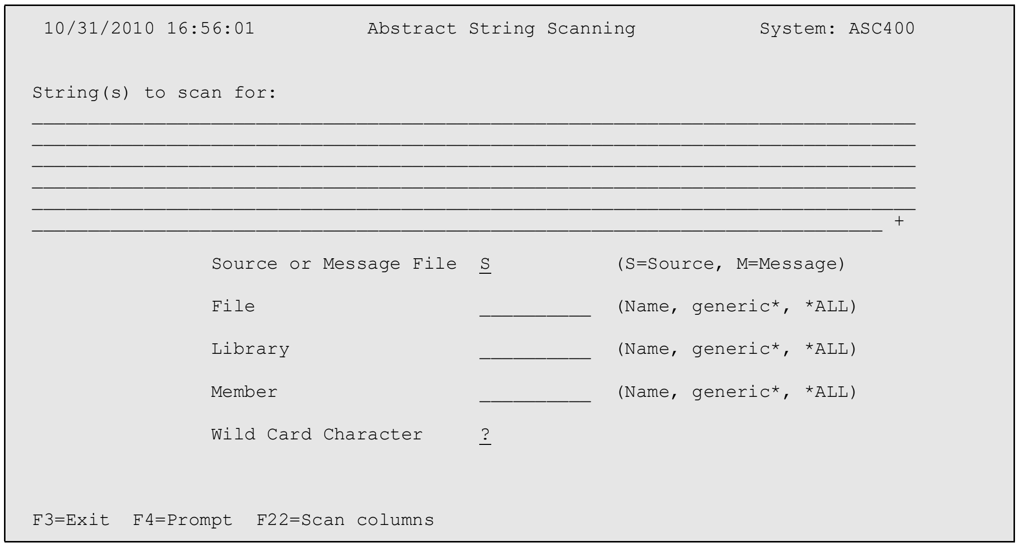
The APTOOL menu includes a facility to search source and message files for occurrences of one of more strings that you provide. Using the ABSTRACT Find String (FNDSTR) command, you specify the strings you want to locate and the files you want searched. ABSTRACT will create a report showing each position of the string(s). The FNDSTR command does not rely upon cross reference information to perform the search.

Access the FNDSTR command by selecting option 6 from the APTOOL Programming Tools menu or through the command entry line. The command invokes an interactive program that will prompt for the search request. It will be allow you to run the search interactively or to submit it to the batch subsystem.

There are no parameters for the FNDSTR command.

```
>>--FNDSTR---------------------------------------------------------------------------><
```

The display below will appear whenever the command is run.



```
 10/31/2010 16:56:01         Abstract String Scanning          System: ASC400

String(s) to scan for:
________________________________________________________________________________
________________________________________________________________________________
________________________________________________________________________________
________________________________________________________________________________
________________________________________________________________________________
_____________________________________________________________________________ +

                Source or Message File  S           (S=Source, M=Message)

                File                    __________  (Name, generic*, *ALL)

                Library                 __________  (Name, generic*, *ALL)

                Member                  __________  (Name, generic*, *ALL)

                Wild Card Character     ?


F3=Exit  F4=Prompt  F22=Scan columns
```

Enter the strings you want ABSTRACT to search for by keying into the subfile at the top of the display. Each search string may be up to 79 characters in length. You can specify multiple search strings by using multiple entry lines on the display. Use the roll keys to enter more information if the subfile is full. ABSTRACT will search each member or file for each of the strings you specify. Text in the source or message files will be converted to uppercase so that case sensitivity will not be a problem.

Your search strings may include a *wild-card* character in the body of the string(s). The default wild-card character is a question mark '?', but it can be changed by typing any valid character into the last prompt on the display. If a match is found for all characters in the string except the wild card character(s), the source will be included on the report.

For example, a search string of OVR???F and a CL source file name will cause ABSTRACT to find all occurrences of OVRDSPF, OVRPRTF, OVRMXDF, etc. without returning occurrences of OVRDBF.

Specify whether you want ABSTRACT to search source files or message files for the strings you have indicated. Type an 'S' to specify a source file search, an 'M' to specify a message file search.

Enter the name of the source or message file to be searched in the middle of the display. Any source or message file may be indicated including QSYS/QCPFMSG. You can specify generic names (leading characters followed by an asterisk) or *ALL for each of these prompts to indicate that more than one file, library, or member should be searched.

Function Keys

Press function key 3 to exit the display without performing any search.

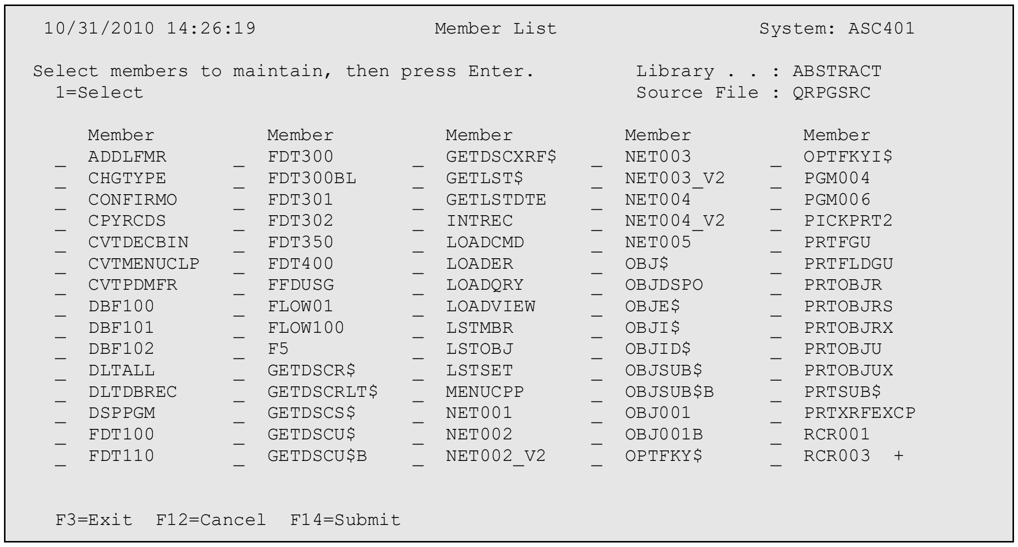
Several source members may be selected by specifying a source file and library name and then pressing function key 4. A prompt similar to the one below will appear. You can choose the members to be searched by placing a ‘1’ next to them prior to making your request.

Use function key 14 to submit your request to the batch job queue. Press the Enter key to begin the search immediately from your interactive session. Running the command interactively may require an extended period of time and may not be an efficient use of your device or interactive priority level.

Function key 22 will drop the search string subfile and allow you to choose specific locations to search for each string in your request.

Selecting Multiple Members

The Multiple Member display can be accessed by specifying a source file and library name and pressing function key 4. All the members included in the source file will be placed into a subfile so that you can select the members you want to search.



```
 10/31/2010 14:26:19               Member List                System: ASC401

Select members to maintain, then press Enter.        Library . . : ABSTRACT
  1=Select                                           Source File : QRPGSRC

     Member          Member          Member          Member          Member
  _  ADDLFMR      _  FDT300       _  GETDSCXRF$   _  NET003       _  OPTFKYI$
  _  CHGTYPE      _  FDT300BL     _  GETLST$      _  NET003_V2    _  PGM004
  _  CONFIRMO     _  FDT301       _  GETLSTDTE    _  NET004       _  PGM006
  _  CPYRCDS      _  FDT302       _  INTREC       _  NET004_V2    _  PICKPRT2
  _  CVTDECBIN    _  FDT350       _  LOADCMD      _  NET005       _  PRTFGU
  _  CVTMENUCLP   _  FDT400       _  LOADER       _  OBJ$         _  PRTFLDGU
  _  CVTPDMFR     _  FFDUSG       _  LOADQRY      _  OBJDSPO      _  PRTOBJR
  _  DBF100       _  FLOW01       _  LOADVIEW     _  OBJE$        _  PRTOBJRS
  _  DBF101       _  FLOW100      _  LSTMBR       _  OBJI$        _  PRTOBJRX
  _  DBF102       _  F5           _  LSTOBJ       _  OBJID$       _  PRTOBJU
  _  DLTALL       _  GETDSCR$     _  LSTSET       _  OBJSUB$      _  PRTOBJUX
  _  DLTDBREC     _  GETDSCRLT$   _  MENUCPP      _  OBJSUB$B     _  PRTSUB$
  _  DSPPGM       _  GETDSCS$     _  NET001       _  OBJ001       _  PRTXRFEXCP
  _  FDT100       _  GETDSCU$     _  NET002       _  OBJ001B      _  RCR001
  _  FDT110       _  GETDSCU$B    _  NET002_V2    _  OPTFKY$      _  RCR003  +


  F3=Exit  F12=Cancel  F14=Submit
```

Place a ‘1’ next to each member you want searched. Carry out the search request by pressing Enter or function key 14. If you press the Enter key, the search will begin immediately. If you press function key 14, your request will be placed on the batch job queue.

Press function key 12 from the multiple member display to cancel the search request and return to the primary display. Use function key 3 to exit the search function completely.

The FNDSTR Report

The report will examine the indicated file(s) and list the results in a fashion similar to that on the following page.

The first example shows the result of a message file search. The FNDSTR request searched the QCPFMSG message file in the QSYS library. It indicated that the entire message (no position specification via function key 22) should be searched for the word Domain.

The report shows the five messages that include the word Domain and their message identifiers. It concludes by indicating the number of messages searched, and the number of occurrences located.

```
13:51:33                    ABSTRACT                                      Page   1
 10/31/2010               String Scanning Utility

String(s)  : From   1 to  80  DOMAIN

Message File: QSYS/QCPFMSG          Wild card character: ?

Msgid   Message
CPD0578 Object domain error for program &1 in library &2.
CPF1989 Domain violation occurred for variable &3.
CPI2247 Domain failure by program &28/&27 for object &22/&21 type &23.
CPPAA2A Time domain reflectometry limit reached
MCH6801 Object Domain error for object &1.

  17316 messages searched
      5 occurrences of requested string found
```

No wild card characters were indicated, although DO?AIN and ??MAIN searches would have located the same messages (and possibly others as well).

The next example shows the output from a source scan. The entire source file (ABSTRACT#/QDDSSRC) was searched for instances of PNLGRP. No position restrictions or wild card characters were used. As before, the end of the report indicates the results of the search.

```
15:36:07                                    ABSTRACT                                    Page   1
 10/31/2010                          String Scanning Utility

String     : From   1 to 256  PNLGRP
Source File: APLUS#/QDDSSRC        Member: *ALL       Wild card character: ?

Member     Sequence Source statement
APCRTPGMS      .08       A                                  HLPPNLGRP('APCRTPGMS' APUSR6)
              5.21       A          H                       HLPPNLGRP('CRTXXXPGMS' APUSR6)
              5.23       A          H                       HLPPNLGRP('MENU/CMDLINE' APUSRINT)
              5.25       A          H                       HLPPNLGRP('MENU/MENUFKY' APUSRINT)
APEXCP         .06       A                                  HLPPNLGRP('APEXCP' APUSR5)
              5.21       A          H                       HLPPNLGRP('APEXCP/ONE' APUSR5)
              5.23       A          H                       HLPPNLGRP('APEXCP/TWO' APUSR5)
              5.25       A          H                       HLPPNLGRP('APEXCP/THREE' APUSR5)
              5.37       A          H                       HLPPNLGRP('MENU/CMDLINE' APUSRINT)
              5.39       A          H                       HLPPNLGRP('MENU/MENUFKY' APUSRINT)
APFILE         .05       A                                  HLPPNLGRP('APFILE' APUSR4)
              5.21       A          H                       HLPPNLGRP('APFILE/ONE' APUSR4)
              5.23       A          H                       HLPPNLGRP('APFILE/TWO' APUSR4)
              5.25       A          H                       HLPPNLGRP('APFILE/THREE' APUSR4)
              5.27       A          H                       HLPPNLGRP('APFILE/FOUR' APUSR4)
              5.29       A          H                       HLPPNLGRP('APFILE/FIVE' APUSR4)
              5.33       A          H                       HLPPNLGRP('MENU/CMDLINE' APUSRINT)
              5.35       A          H                       HLPPNLGRP('MENU/MENUFKY' APUSRINT)
APINZ          .90       A                                  HLPPNLGRP('APINZ' APUSR1)
              5.60       A          H                       HLPPNLGRP('APINZ/ONE' APUSR1)
              5.90       A          H                       HLPPNLGRP('APINZ/TWO' APUSR1)
              6.20       A          H                       HLPPNLGRP('APINZ/THREE' APUSR1)
              6.50       A          H                       HLPPNLGRP('APINZ/FOUR' APUSR1)
              6.80       A          H                       HLPPNLGRP('APINZ/FIVE' APUSR1)
              8.83       A          H                       HLPPNLGRP('MENU/CMDLINE' APUSRINT)
              8.90       A          H                       HLPPNLGRP('APINZ/MENUFKY' APUSR1)
APLUS          .90       A                                  HLPPNLGRP('MENU' APUSRINT)
              5.70       A          H                       HLPPNLGRP('APINZ' APUSR1)
              6.00       A          H                       HLPPNLGRP('APOBJR' APUSR2)
              6.30       A          H                       HLPPNLGRP('APOBJU' APUSR3)
              6.60       A          H                       HLPPNLGRP('APFILE' APUSR4)
              6.90       A          H                       HLPPNLGRP('APEXCP' APUSR5)
              7.20       A          H                       HLPPNLGRP('APTOOL' APUSR6)
              7.80       A          H                       HLPPNLGRP('MENU/CMDLINE' APUSRINT)
              8.10       A          H                       HLPPNLGRP('MENU/MENUFKY' APUSRINT)
APOBJR         .07       A                                  HLPPNLGRP('APOBJR' APUSR2)
              5.22       A          H                       HLPPNLGRP('WRKOBJR' APUSR2)
              5.25       A          H                       HLPPNLGRP('WRKOBJRS' APUSR3)
              5.28       A          H                       HLPPNLGRP('WRKOBJR/PGM' APUSR2)
       .
       .
       .
    135 members searched
  22280 source lines scanned
    522 occurrences of requested string found
```

Print Source Member (PRTSRCMBR) Command

ABSTRACT can provide two types of source listings for the objects in your applications. A standard listing prints the source as it exists in the source member, along with the sequence numbers and change date. An indented listing allows you to see the effects of structured programming groups within an RPG program. Both of these types of listings are produced by the Print Source Member (PRTSRCMBR) command.

You can invoke the PRTSRCMBR command from any command entry line, or by using option 7 of the APTOOL menu.

The PRTSRCMBR command can be run interactively or submitted to a batch subsystem. Running the command interactively may require an extended period of time and may not be an efficient use of your device or interactive priority level. You can submit the command to the batch job queue from the APTOOLS menu by selecting option 7 and pressing function key 14 instead of the Enter key. Once you have completed the prompt, the request will be submitted using the job description you have identified through function key 18.

The syntax for the command is shown below:

```
                          .-*LIBL------.                         .-*ALL-----------.
>>--PRTSRCMBR----SRCFILE-+--------------+---+-file_name-+---MBR-+----------------+------>
                          '-library_name-'                       +-member_name----+
                                                                 '-*generic*_name-'

                                                                                Required
=========================================================================================
                                                                                Optional

          .-*NO--.
>--INDENT-+----------------------------------------------------------------------------><
          '-*YES-'
```

SRCFILE Parameter

Specify the qualified name of the source file that contains the code you want printed. The library list will be searched for the source file if you do not explicitly qualify the file name.

MBR Parameter

Specify the name of the member to be printed.

***ALL:** All members in the source file will be printed. If you have specified INDENT(*YES), only members with an RPG or RPT type will be printed.

name: specify the individual member to be printed.

***generic*-name:** Specify a partial member name qualified by an asterisk (*) to select several members meeting the criteria. The following generic forms are allowed: ABC*, *ABC, *ABC*, A*B. Refer to Appendix G, General Functions, in Volume 1 of the CL Reference manual for more information about generic names.

INDENT Parameter

Specify whether you want an indented RPG listing that shows the structured programming groups within the program.

<u>*NO:</u> print a standard listing without indenting

***YES:** print an indented listing of RPG program members that shows the structured programming constructs. The printout indents the beginning of each structured block three spaces. When the end of the block is reached, the indenting is undone.

The PRTSRCMBR Listing

The examples below show the output created by the PRTSRCMBR command. The first example shows a standard source listing - change date, sequence number and source line are listed for each record in the member.

```
                                        A/P+ Source Listing                                    Page   1

Source File:  ABSTRACT#/QRPGSRC      Member:  SRC002

  Date     Seqnbr  *... ... 1 ... ... 2 ... ... 3 ... ... 4 ... ... 5... ... 6 ... ... 7 ... .. ... 1

05/06/86     1.00       FSOURCE  IF  F      92  6AI     1 DISK
05/06/86     2.00       FQSYSPRT O   F     132     OF     PRINTER
05/06/86     3.00       E                    LINE       95  1
06/11/86     4.00       ISOURCE  NS  10  18 CC  19NC*
05/07/86     5.00       I       OR   11
05/06/86     6.00       I                                        1   62SRCSEQ
   .
   .
05/06/86    22.00       C                     Z-ADD0         NSTLVL  30
05/06/86    23.00       C                     SETON                     OF
            24.00       C           $READ     TAG
05/06/86    25.00       C                     SETOF                     1011
05/06/86    26.00       C                     READ SOURCE                   LR
05/07/86    27.00       C         LR          RETRN
08/11/86    28.00       C         10OP2       IFNE ‘CAS’
08/11/86    29.00       C                     MOVEL’N’       CASE
08/11/86    30.00       C                     END
06/11/86    31.00       C         10OP        IFEQ ‘DO’
06/11/86    32.00       C           OP        OREQ ‘IF’
07/25/86    33.00       C           OP2       OREQ ‘CAS’
08/11/86    34.00       C           CASE      ANDNE’Y’
   .
   .
05/06/86    52.00       C         OF          EXCPT#HDING
06/11/86    63.00       C                     END
            64.00       C                     GOTO $READ
06/11/86    65.00       OQSYSPRT E  102           #HDING
06/11/86    66.00       O                         UDATE Y    8
09/27/88    67.00       O                                   62 ‘ABSTRACT Indentati’
05/06/86    68.00       O                                      ‘on listing for ‘
05/06/86    69.00       O                         MEMBER
   .
   .
05/06/86    84.00       O                         LINE
08/13/86    85.00       O                    10N01       +  4 ‘Hi’
05/07/86    86.00       O                    10N02           ‘Lo’
05/07/86    87.00       O                    10N03           ‘Eq’

                                          * * * * * E n d   O f   S o u r c e * * * * *
```

The next example shows an indented RPG listing. Each record in the source member is printed, just as in a standard source listing. Next to the date and sequence is the nesting level of the current statement. This corresponds to the nesting level on a compiler listing.

Each time a structured block is encountered, the block is indented from the rest of the program. When the block ends, indenting is discontinued. A vertical bar is printed to extend a line from the beginning to the end of the block. If the block is an IF block containing an ELSE clause, each statement of the ELSE can be distinguished by a ‘.’ preceding the statement. Refer to statements 399-411 for an example of this.

Indicator usage is noted to the right of each statement with values Hi, Lo and Eq to the right of the statement to indicate the position used by the indicator.

```
 10/31/2010                        A/P+ Indentation listing for   CONFIRMO                    Page   1
 9:46:50

  Date    Seqnbr Lvl                                                                               Indicators
92/01/31    1.00           FOBJCNFF CF  E                    WORKSTN
92/01/31    1.01           F                                              KINFDS DSPINF
92/01/02    4.02           FCNFRMOBJCF  E                    WORKSTN
92/01/02    4.03           F                                        CRRN  KSFILE CNFRMSFL
   .
   .
91/10/01  214.00           C                     WRITECNFRMCTL
92/01/02  214.01           C                     SETOF                     44                      Hi
91/09/30  216.00           C                     MOVE *LOVAL    FSTTOP
91/09/30  217.00           C                     MOVE *HIVAL    FSTNAM
91/09/30  218.00            /SPACE
91/09/30  219.00           C                     SETON                     25                      Hi
91/10/02  219.01   1       | C           2         DO   OPMX      I
91/10/02  219.02   1       | C                     MOVE *HIVAL    WNBG,I
91/10/02  219.03   1       | C                     MOVE *LOVAL    WNEN,I
91/10/02  219.04   1       | C                     END                             2 DO OPMX I
91/10/02  219.05           C                     Z-ADD1         WNBG,1
91/10/02  219.06           C                     MOVE *HIVAL    WNEN,1
91/10/02  219.07   1       | C                     DO   OPMX      C
91/05/31  229.00   2       | | C           *IN03     DOUNE*ZERO
91/10/14  230.00   2       | | C           *IN98     OREQ *ZERO
91/10/02  230.01   3       | | | C           WNBG,C    IFNE *HIVAL
91/09/30  232.00   3       | | | C                     Z-ADD1         RDBEG
91/09/30  233.00   3       | | | C           SFLCNT    ADD  1         RDEND
91/09/30  234.00   3       | | | C                     Z-ADD1         RDINC
91/06/04  235.00   3       | | | C                     EXSR FILSFL
91/10/02  236.00   3       | | | C                     END                             WNBG,C IFNE *HIVAL
91/10/02  236.01   2       | | C                     END                             *IN03 DOUNE *ZERO
91/10/02  236.02   1       | C                     END                             DO OPMX C
91/05/31  237.00            /SPACE
91/11/20  238.00            * If the user has made any changes to the temporary option file (PRBOBJ),
91/06/06  239.00            * reflect those changes in the real option file, but  ONLY if F3 has not
91/06/06  240.00            * been pressed.
92/03/25  240.01           C           EXIT      TAG
92/02/06  240.02           C        N03          BITOF’7’       OBJER@
   .
   .
91/10/02  392.00   3       | | | C           OKEY,I    ADD  1         CRRN
92/03/26  393.00   3       | | | C           CRRN      CHAINCNFRMSFL             90                Hi
92/03/26  394.00   4       | | | | C           *IN90     IFNE *ZERO
91/10/02  395.00   4       | | | | C           CRRN      ORGT WNEN,C
91/06/04  396.00   4       | | | | C                     MOVELBOTMSG    MSGFLD
91/06/04  397.00   4       | | | | C                     SETOF                         25              Eq
91/06/04  398.00   4       | | | | C                     END                             *IN90 IFNE *ZERO
91/06/03  399.00   3       | | |.C                     ELSE                            *IN25 IFNE *ZERO
91/06/03  400.00   4       | | |.| C           *IN26     IFNE *ZERO
91/06/03  401.00   4       | | |.| C                     Z-ADDSFLCNT    RDBEG
91/06/03  402.00   4       | | |.| C                     Z-ADD0         RDEND
91/06/10  403.00   4       | | |.| C                     Z-SUB1         RDINC
91/10/02  404.00   4       | | |.| C           OKEY,1    SUB  1         CRRN
91/10/02  405.00   4       | | |.| C           CRRN      CHAINCNFRMSFL             90              Hi
91/06/04  406.00   5       | | |.| | C           *IN90     IFNE *ZERO
91/06/04  407.00   5       | | |.| | C                     MOVELTOPMSG    MSGFLD
91/06/04  408.00   5       | | |.| | C                     SETOF                         26            Eq
91/06/04  409.00   5       | | |.| | C                     END                             *IN90 IFNE *ZERO
91/06/03  410.00   4       | | |.| C                     END                             *IN26 IFNE *ZERO
91/06/03  411.00   3       | | |.C                     END                             *IN25 IFNE *ZERO
91/06/17  412.00   2       | | C                     END                             N03N05N12 LSTBOT IFGT
91/10/01  413.00   1       |  *
91/10/01  414.00   1       |  * F5 - Refresh changed values to their previous contents.
91/06/10  415.00   2       | | C           *IN05     IFNE *ZERO
91/10/02  416.00   2       | | C                     Z-ADD1         CRRN
91/10/02  417.00   2       | | C           CRRN      CHAINCNFRMSFL             89                  Hi
91/06/10  418.00   3       | | | C           *IN90     DOWEQ*ZERO
91/10/02  419.00   3       | | | C                     ADD  1         CRRN
91/10/02  420.00   3       | | | C           CRRN      CHAINCNFRMSFL             89                Hi
91/06/10  421.00   3       | | | C                     END                             *IN90 DOWEQ *ZERO
91/06/10  422.00   2       | | C                     END                             *IN05 IFNE *ZERO
91/06/04  423.00   1       | C                     END                             *IN25 DOUEQ *ZERO
```

Change SEU Source Type (CHGTYPE) Command

ABSTRACT depends on the accuracy of a member's source type to direct its source analysis function. If the source type does not correspond to the type of source in the member, ABSTRACT will use the wrong analysis routines, or may not analyze the member at all.

Occasionally, you may find that the source type for the members in a source file does not accurately reflect the type of source for the file. This can occur if the source file sustained some types of object damage, or if the data was placed into the member with a copy file operation.

The Change SEU Type (CHGTYPE) command can be used to correctly set the source type for one or more members in a source file. Access the CHGTYPE command from a command entry line or by selecting option 8 of the APTOOL menu.

The CHGTYPE command can be run interactively or submitted to a batch subsystem. Although the command processes members very quickly, running the command interactively may require an extended period of time if you are changing each member in an especially large file. This may not be an efficient use of your device or interactive priority level. You can submit the command to the batch job queue from the APTOOLS menu by selecting option 8 and pressing function key 14 instead of the Enter key. Once you have completed the prompt, the request will be submitted using the job description you have identified through function key 18.

The syntax for the CHGTYPE command is shown below:

```
                       .-*LIBL------.                         .-*ALL-----------.
>>--CHGTYPE----SRCFILE-+--------------+---+-file_name-+---MBR-+----------------+-------->
                       '-library_name-'                       +-member_name----+
                                                              '-*generic*_name-'

>--TYPE-+-source_type-+------------------------------------------------------------------><
```

SRCFILE Parameter

Specify the qualified name of the source file that contains the member(s) you want changed. The library list will be searched for the source file if you do not explicitly qualify the file name.

MBR Parameter

Specify the name of the member(s) to be changed.

***ALL:** All members in the source file will be changed so that their source type has the value indicated by the TYPE parameter.

name: specify the individual member to be changed.

***generic*-name:** Specify a partial member name qualified by an asterisk (*) to select several members meeting the criteria. The following generic forms are allowed: ABC*, *ABC, *ABC*, A*B. Refer to Appendix G, General Functions, in Volume 1 of the CL Reference manual for more information about generic names.

TYPE Parameter

Specify the source type you want the member(s) to have. Refer to the CL Programmer's Guide for a listing of valid source types.

Examples

```
CHGTYPE FILE(PRODLIB/QCLSRC) MBR(*ALL) TYPE(CLP)
```

The command above changes all the members in the QCLSRC file in the PRODLIB library so that they have an SEU source type of CLP.

```
CHGTYPE FILE(PRODLIB/QRPGSRC) MBR(TBLPRT) TYPE(RPG)
```

This command changes only one member, TBLPRT in PRODLIB/QRPGSRC, and sets the source type value to RPG.

Change Service Data (CHGSVCDTA) Command

Many ABSTRACT functions access the object service data to determine the source used to create an object. The service data is attached to an object when it is created by the object creation function. You can view it by using the Display Object Description (DSPOBJD) command and selecting DETAIL(*SERVICE).

If the source code is moved once the object is created, the service description does not change and will point to source code that does not exist any longer. This will cause a problem when the ABSTRACT functions attempt to locate the object's source code using the service description for the object.

Depending on the procedures you use to create the production objects in your applications, the object service data may not indicate the current location of the source code. ABSTRACT supplies a command that can update the object service description for your program and file objects so that it will accurately reflect the current location of the source.

The Change Service Data (CHGSVCDTA) command can be accessed from command entry or through option 9 of the APTOOL Programming Tools menu. It allows you to quickly and easily change the service description for many objects at once.

You should always run the CHGSVCDTA command to update an object's service description after you move the source for an object.

The syntax for the Change Service Data (CHGSVCDTA) command is shown below:

```
                     .-*LIBL--------.   .-*ALL-----------.
>>--CHGSVCDTA----OBJ-+--------------+---+----------------+----------------------------->
                     '-library_name-+   +-object_name----+
                                        '-*generic*_name-'

           .-*PGM--------.          .-*ALL-------------.
>--OBJTYPE-+-------------+---OBJATR-+------------------+------------------------------->
           '-object_type-'          +-object_attribute-+
                                    '-generic*_name----'

            .-10 maximum--------------.
            v                         |          .-*OBJ--------.
>--SRCFILE---+-library_name/file_name-+---SRCMBR-+-------------+----------------------->
                                                 '-member_name-'

          .-*NOCHG-.
>--MBROPT-+--------+------------------------------------------------------------------->
          '-*CHG---'

                                                                               Required
========================================================================================
                                                                               Optional

          .-*PRINT-.
>--OUTPUT-+--------+------------------------------------------------------------------><
          '-*NONE--'
```

The CHGSVCDTA command can be run interactively or submitted to a batch subsystem. Although the command processes objects quickly, running the command interactively may require an extended period of time if you are changing each program in an especially large library. This may not be an efficient use of your device or interactive priority level. You can submit the command to the batch job queue from the APTOOLS menu by selecting option 9 and pressing function key 14 instead of the Enter key. Once you have completed the prompt, the request will be submitted using the job description you have identified through function key 18.

OBJ Parameter

Specifies the object name. You can use this parameter to change the service data for all the objects or a subset of objects in the indicated library.

<u>*ALL:</u> All objects in the specified library(s) are selected.

object-name: Only objects with this specific name will be changed.

generic*-name: Specify a partial object name qualified by an asterisk (*) to select several objects meeting the criteria.

OBJTYPE Parameter

Specifies the object type. You can use this parameter to work with any subset of objects. An expanded description of the OBJTYPE parameter and a list of valid OS/400 object types can be found in Appendix A, Expanded Parameter Descriptions, of Volume 1 of the *CL Reference* manual.

<u>*PGM:</u> Program objects in the specified library(s) are selected.

object-type: Specify any valid system object type to select all objects of that particular type.

OBJATR Parameter

Specifies the object attribute. You can use this parameter to change objects with any type of attribute or only specific ones. An expanded description of the OBJATR parameter and a list of valid OS/400 object attributes can be found in Appendix A, Expanded Parameter Descriptions, of Volume 1 of the *CL Reference* manual.

<u>*ALL:</u> All objects in the specified library(s) are selected regardless of their attribute definition.

***BLANK:** Only objects with no attribute will be included.

object-attribute: Only objects with this specific attribute will be included in the list. PF-DTA can be used to select only non-source physical files. PF-SRC can be used to select only source physical files.

generic*-name: Specify a partial object attribute ending with an asterisk (*) to select several objects meeting the criteria. For example, RPG* will select RPG, RPG36, and RPG38 objects. Refer to Appendix G, General Functions, in Volume 1 of the *CL Reference* manual for more information about generic names.

SRCFILE Parameter

Specify the qualified names of up to ten files containing the source code for the objects you are changing. If an object's service data is to be reset, the files will be searched (in order) for a member matching the value of the SRCMBR parameter. If found, the service data will be reset to indicate the library/file and member found by the search.

SRCMBR Parameter

Specify the name of the member to be indicated by the object service data.

<u>*OBJ:</u> the name of the member will be the same as the name of the object specified above.

name: specify the individual member to be referenced by the service data.

MBROPT Parameter

Specify whether objects with valid service data should be checked against the file list, or only those with currently invalid service data.

<u>*NOCHG:</u> Do not change an object's service data if it is currently valid. If the source member, file, and library indicated by the service data exists, do not change the object's service data.

***CHG:** Always search the file list for a member indicated by the SRCMBR parameter value, regardless of the validity of the object's service data.

OUTPUT Parameter

Specify whether a record of service data changes should be printed. The output also includes a record of any errors encountered that prevented service data from being changed.

<u>*PRINT:</u> Print the service data log.

***NONE:** Do not print the log.

Examples

If you have moved your RPG source, you might type the following to update the service information for the programs in your application.

```
CHGSVCDTA OBJ(ProgramLibrary/*ALL) OBJTYPE(*PGM) OBJATR(RPG)
      SRCFILE(SourceLibrary/Sourcefile) MBR(*OBJ)
```

where *ProgramLibrary* is the library that the programs were compiled into, and *SourceLibrary/SourceFile* is the location of the corresponding source code. For instance,

```
CHGSVCDTA OBJ(PRODLIB/*ALL) OBJTYPE(*PGM)
      SRCFILE((SOURCE/QRPGSRC) (SOURCE/QCLSRC)) MBR(*OBJ)
```

will attempt to modify the service data for all the programs in the PRODLIB library without valid service data. If a member having the same name as the program can be found in either the QRPGSRC or the QCLSRC file in the SOURCE library, the service data for the program will be reset to that member.

```
CHGSVCDTA OBJ(DATALIB/*ALL) OBJTYPE(*FILE)
      SRCFILE(SOURCE/QDDSSRC) MBR(*OBJ) MBROPT(*CHG)
```

This command will modify the service data for all the files in the DATALIB library (whether or not they have valid service data) so that the file QDDSSRC in library SOURCE is indicated for each corresponding member in the QDDSSRC file. If a member matching the file's name cannot be found in the QDDSSRC file, the service data will not be changed.

```
CHGSVCDTA OBJ(PRODLIB/CLPGMA) OBJTYPE(*PGM)
      SRCFILE(SOURCE/QCLSRC) MBR(CLPGMB)
```

The service data for program CLPGMA in the PRODLIB library will be changed to indicate that the program was compiled from member CLPGMB in the SOURCE/QCLSRC file.

Convert PDM Option File (CVTPDMF) Command

If you are experienced with IBM's Programming Development Manager (PDM) product, you may have created your own user option file that you find useful during your application development work.

Although ABSTRACT option files have more capabilities than the standard PDM option file, you can use your existing PDM option file as a basis for working with ABSTRACT functions.

Before ABSTRACT can use a PDM option file, you must run a conversion routine that is invoked using the Convert PDM Option File (CVTPDMF) command. You should run this command on a copy of the option file. Once it completes, you will NOT be able to use the converted file with PDM.. Refer to Chapter 8 for a complete description covering the ABSTRACT option file capabilities.

ABSTRACT cannot use PDM option files until the CVTPDMF command is run.

Access the CVTPDMF command from a command entry line or by selecting option 10 from the APTOOL programming tools menu. The command can be run in a batch subsystem or invoked from an interactive or batch program as well. The syntax for the CVTPDMF command is shown below:

```
              .-*LIBL-.
>>--CVTPDMF----+-------+---+-library_name-+---+-file_name-+---------------------------><
              '-FILE--'
```

FILE Parameter

Specifies the PDM option file name. Provide a qualified file name or omit the library name to search your current library list for the option file.

<u>***LIBL:**</u> The current job library list will be searched for the file indicated by the parameter.

Examples

```
CVTPDMF FILE(QGPL/QAUOOPT)
```

This example will convert the QAUOOPT file in the QGPL library so that it can be used by ABSTRACT. The conversion creates a new file with the same name as the original and the original file is replaced. After conversion PDM will not be able to access and run the options in the file.

Print Job Stream Flowchart (PRTFLWCHT) Command

The flowcharting function of ABSTRACT will analyze the cross reference information that tracks the transfers of program control within an application and the file usage that occurs among the application programs.

Two types of flowcharts can be produced:

Graphic flowcharts show a graphic representation of your application job streams. Program transfers and file usage by the programs are noted. Up to six levels of nesting will be indented, others will appear under the sixth column. Up to fourteen files will appear in each ‘file box’ at the right of the flowchart, as many boxes as required will be shown. Overrides will be displayed side by side, such that the from file will be shown on the left hand side, the TOFILE on the right.

Source flowcharts provide an interleaved listing of program source code. Source code from a program is printed until a program transfer is reached, then source from the new program is printed until it transfers or ends. When the program ends, the original program source will resume. This continues until the entire job stream has been processed.

Access the flowcharting functions by typing the Print Flowchart (PRTFLWCHT) command on a command entry line or by selecting option 11 from the APTOOL programming tools menu. Option PFC in the ABSTRACT/OPTFILE option file will also run the PRTFLWCHT command. The command syntax is shown below.

```
                         .-*LIBL--------.   .-pgm_name-.        .-*GPH-.
>>--PRTFLWCHT----PGM-+------------------+---+----------+---TYPE-+------+---------------->
                         '-library_name-'   '-*ALL-----'        '-*SRC-'

                                                                               Required
=========================================================================================
                                                                               Optional

          .-*PRV---.
>--DTASET-+--------+-------------------------------------------------------------------><
          +-*FIRST-+
          '-name---'
```

The PRTFLWCHT command can be run interactively or in a batch subsystem. It can also be invoked from an interactive or batch program. Given that most job streams involve many programs and files, you will probably want to submit this command to the batch subsystem in most circumstances.

PGM Parameter

Specify the name of the entry point program and library. ABSTRACT will begin with the program you name and show all the control transfers and file usage for each subsequent program in the job stream.

LIBL: The library list will be searched for the program you specify.

ALL: Flowcharts for all programs in the library you specify will be printed. This will generate useful results only if your library contains programs that are the starting point of your job streams rather than all the programs in your application.

TYPE Parameter

Specify the type of flowcharts to be produced.

<u>*GPH</u>: show a graphic representation of the job stream.

***SRC:** create an interleaved listing of program source code.

DTASET Parameter

Specify the cross reference data set to be used in acquiring the information.

<u>*PRV:</u> Use the value for this parameter from the previous session. If no previous session has occurred, use *FIRST.

***FIRST:** information from the first (default) data set should be accessed.

Name: Specify any existing data set. Use the command prompter to find the names available to you.

Examples

The following pages show examples of flowcharts produced by ABSTRACT. The graphic flowchart on the next page depicts programs within the job stream as rectangular boxes at the left and center of the report. The transfer command (CALL, TFRCTL, etc.) appears above the program name. Files used by the program are shown at the right side of the report. File references will also be listed (I = input, O = output, U = update, A = add, ? = unspecified). Where possible, unspecified type references will be resolved into the actual CL command used.

Program transfer can occur across library boundaries. All libraries which are loaded into the cross reference will be traversed as if they were on the library list of the flowcharter.

The example on the next page shows part of the job stream initiated by program MENUC in the PILOT library. The flowchart shows that the CONTROL program (a CL program) called at statement 72.00 in MENUC calls three programs: WCBT01, HLP001 and PILOT. The PILOT program calls two others: SCH001 and JOBEXEC.

Notice that DUPLICATE appears in the HLP001 box. This indicates that the program has been processed earlier in the job stream and any programs that it calls are excluded from this reference.

File overrides are presented as shown in the example for programs JOBEXEC and JOBEXE. The CL program that performs the override shows the from-file references. Subsequent references to the from-file will generate an OVERRIDEN TO reference like those shown for program JOBEXE.

```
10/31/2010 13:50:07                                    ABSTRACT                               Page . . . :   1
                                          Program References by Sequence Number
Object name . . : MENUC                                                                            System . . : ASC400
Object type . . : *PGM                                                                             Data Set . : *FIRST
Object library  : PILOT
  Level 1   Level 2   Level 3   Level 4   Level 5   Level 6   Level 7   Level 8   Level 9   Level 10+
        |
        |               Monitor control information
   _____|______         ___________________________________________________________________________
  | CALL       |       /EXTEND     *DTAARA     Unknown                                            /
  | CONTROL    |      / JOBQ       *DTAARA     Unknown                                           /
  |            |   __/  LASTTIME   *DTAARA     Unknown                                          /
  | *PGM CLP   |     \  LASTWAKE   *DTAARA     Unknown                                          \
  |   72.00    |      \ CONTROL    *FILE DSPF  InpOut                                            \
  |____________|       \__________________________________________________________________________\
                  |
                  |               Work control block access
             _____|______
            | CALL       |
            | WCBT01     |
            |            |
            | *PGM MI    |
            |    24.00   |
            |____________|
                  |
                  |               Help text processor
             _____|______         ___________________________________________________________________________
            | CALL       |       /HELPTXT    *FILE PF    Inp                                                /
            | HLP001     |      / HLP001     *FILE DSPF  InpOut                                            /
            | DUPLICATE  |   __/                                                                          /
            | *PGM RPG   |     \                                                                          \
            |   74.00    |      \                                                                          \
            |____________|       \__________________________________________________________________________\
                  |
                  |
                  |               Pilot command processor
             _____|______         ___________________________________________________________________________
            | CALL       |       /LASTTIME   *DTAARA     Unknown                                            /
            | PILOT      |      / LASTWAKE   *DTAARA     Unknown                                           /
            |            |   __/  LSTLOA     *DTAARA     Unknown                                          /
            | *PGM CLP   |     \  PILOTNBR   *DTAARA     Unknown                                          \
            |   96.00    |      \ WAKEUP     *DTAARA     Unknown                                           \
            |____________|       \__________________________________________________________________________\
                            |
                            |
                            |               Schedule loader
                       _____|______         __________________________________________________________________________
                      | CALL       |       /EXTEND     *DTAARA     Unknown                                           /
                      | SCH001     |      / LSTLOA     *DTAARA     Unknown                                          /
                      |            |   __/  WRKDYS     *DTAARA     Unknown                                         /
                      | *PGM RPG   |     \  HOLIDAYS   *FILE PF    Inp                                             \
                      |   51.00    |      \ JOBHDR     *FILE PF    Inp        Job Master Header Information         \
                      |____________|       \_________________________________________________________________________\
                                      |
                                      |
                                 _____|______
                                | CALL       |
                                | PIL020     |
                                | DUPLICATE  |
                                | *PGM MI    |
                                |  132.00    |
                                |____________|
                                     |
                                     |
                            |               Job execution pre-program
                       _____|______         __________________________________________________________________________
                      | CALL       |       /JOBDTL     *FILE PF    Ovrdbf                                            /
                      | JOBEXEC    |      / JOBHDR     *FILE PF    Ovrdbf                                           /
                      |            |   __/  JOBSCHL3   *FILE PF    Ovrdbf                                          /
                      | *PGM CLP   |     \                                                                         \
                      |  131.00    |      \                                                                         \
                      |____________|       \_________________________________________________________________________\
                                     |
                                     |               Job Execution
                                _____|______         _________________________________________________________________
                               | CALL       |       /JOBDTL     *FILE PF    Upd        OVRTO(PILOT/JOBDTL)           /
                               | JOBEXE     |      / JOBHDR     *FILE PF    Upd        OVRTO(PILOT/JOBDTL)          /
                               |            |   __/  JOBSCHL3   *FILE PF    UpdAdd     OVRTO(PILOT/JOBDTL)         /
                               | *PGM RPG   |     \                                                                \
                               |   17.00    |      \                                                                \
                               |____________|       \________________________________________________________________\
```

Following the graphic flowchart, a summary page like the one below lists all the programs called within the job stream and the files they use. Files are listed alphabetically along with the page number(s) that contain their references.

```
Programs called within job stream of MENUC

PIL020    HLP001    CONTROL    WCBT01    HLP001     PILOT      SCH001     PIL020     JOBCHK     JOBSBMC
JOBEXEC   JOBEXE    QCMDEXC    QCMDEXC   QCMDEXC    QCLSCAN    CM2,1      QCMDEXC    QCAEXEC    PIL020
JOBMNT    HLP001    SCHMNT     HLP001    PRG001     HLP001     PRG001C    PRG002     PIL020     WCBT01
HLP001    PRG001    DSPJOBC    JOBMNT    DSPJLG     JOBMNT     HLP001     PIL020     QCMDCHK    QCACHECK
HLP001    RMVMSGC   QCMDCHK    QCACHECK  QCMDCHK    QCACHECK   HLP001     SCHMNT     CHKJOBC    RTVJOBDA
PIL020    CAL002C   CAL002     PIL020    CAL001     PIL020     RPT001     PIL020     RPT002     PIL020
RPT003    PIL020    PIL020     REPORTS   REPORTS    INT001     HLP001     HLP001     QCMDEXC

File Overrides          Page Program    Usage    Page Program    Usage    Page Program    Usage
           CALWRK         17 CAL002C    OVRFRM
           JOBDTL          4 JOBEXEC    OVRFRM
           JOBHDR          4 JOBEXEC    OVRFRM
           JOBLDA          4 JOBEXEC    OVRFRM
           JOBSCHL3        4 JOBEXEC    OVRFRM
           JOBTRG          4 JOBEXEC    OVRFRM

File Usage              Page Program    Usage    Page Program    Usage    Page Program    Usage
ASC#PL                     1 MENUC      DTAARA
CALWRK                    17 CAL002     (I,O)
CONTROL                    4 CONTROL    (I,O)
DATEJOIN                  18 RPT003     (I)
DEPTJOIN                  18 RPT002     (I)
EXTEND                     4 CONTROL    DTAARA      4 SCH001     DTAARA     11 SCH001     DTAARA
HELPTXT                    4 HLP001     (I)         4 HLP001     (I)         5 HLP001     (I)
                          20 HLP001     (I)
HLP001                     4 HLP001     (I,O)       4 HLP001     (I,O)       5 HLP001     (I,O)
                           6 HLP001     (I,O)       7 HLP001     (I,O)       8 HLP001     (I,O)
                          10 HLP001     (I,O)      11 HLP001     (I,O)      12 HLP001     (I,O)
                          20 HLP001     (I,O)
HOLIDAYS                   4 SCH001     (I)         5 JOBMNT     (I)         6 SCHMNT     (I)
                          13 PERMNT     (I)        14 HOLMNT     (U,A)      14 HOLLST     (I)
                          17 CAL002     (I)        17 CAL001     (I)
  .
  .
JOBTRGL1                   5 JOBMNT     (U)         8 JOBMNT     (U)         8 JOBMNT     (U)
LASTTIME                   4 CONTROL    DTAARA      4 PILOT      DTAARA
LASTWAKE                   4 CONTROL    DTAARA      4 PILOT      DTAARA
LOGFILE                    8 DSPJLG     (I)
LOGFILEL                   7 PRG002     (U)
LSTLOA                     4 CONTROL    DTAARA      4 PILOT      DTAARA      4 SCH001     DTAARA
                           6 SCHMNT     DTAARA      8 JOBMNT     DTAARA      8 JOBMNT     DTAARA
                          11 SCH001     DTAARA     13 SCHMNT     DTAARA     14 REPORTS    DTAARA
                          19 REPORTS    DTAARA
MENUC                      1 MENUC      (I,O)
MSGQ                       1 MENUC      DTAARA     19 INT001     DTAARA
OUTQ                       1 MENUC      DTAARA      6 PRG001C    DTAARA     15 REPORTSC   DTAARA
  .
  .
PILOT/JOBLDA               4 JOBEXEC    OVRTO
PILOT/JOBSCHL3             4 JOBEXEC    OVRTO
PILOT/JOBTRG               4 JOBEXEC    OVRTO
PILOTNBR                   4 CONTROL    DTAARA      4 PILOT      DTAARA
PILOTPR                   14 HOLLST     (O)        15 JOBLST     (O)        16 PERLST     (O)
                          16 CAL003     (O)        17 CAL002     (O)        17 CAL001     (O)
                          18 RPT002     (O)        18 RPT003     (O)
PJOBLOG                    8 DSPJLG     (O)
PRG001                     6 PRG001     (I,O)       7 PRG001     (I,O)
QTEMP/CALWRK              17 CAL002C    OVRTO      17 CAL002C    CLRPFM
```

The next report shows an example of a source flowchart. The invocation level of the job stream is displayed at the left margin in an indented fashion. Source records from individual source members are printed with the sequence numbers at the left. Each time a program transfer is made a double line of asterisks will appear, indicating the program operation (CALL, TFRCTL, QRYDTA, RETURN, etc.), the program name, type, text and source information.

File overrides that apply to the invocation will be listed before the source for the program begins printing. (c.f. the CALL from JOBEXEC to JOBEXE)

All source statements for the CL programs in the job stream will be printed on the report. To keep the report brief, only the file information, entry parameters, and transfers of program control are printed for RPG and COBOL programs.

```
 9:09:58                                          ABSTRACT                                         Page     1
 10/31/2010                                  Procedure Explosion

*************************************************************************************************************
START: MENUC       PILOT      CLP PILOT#/QCLSRC MENUC              Pilot menu driver
*************************************************************************************************************
.1        1.00  PGM
.1        2.00  DCL &MSGDTA *CHAR 100
     .
     .

*************************************************************************************************************
CALL    JOBCHK      PILOT#     RPG PILOT#/QRPGSRC JOBCHK           Scheduling - Check for jobs to execute
*************************************************************************************************************
....4       1.00       FJOBSCHL4UF  E           K        DISK
....4       2.00       FJOBHDR  UF  E           K        DISK
....4      91.00       C                     CALL 'JOBSBMC '

*************************************************************************************************************
CALL    JOBSBMC     *LIBL      CLP PILOT#/QCLSRC JOBSBMC           Submit job to execute
*************************************************************************************************************
.....5       1.00  PGM (&JOB &JOBD  &JOBDLI &OUTQ  &OUTQLI &JOBQ  &JOBQLI &JOBSWS +
.....5       2.00  &ACGCDE &SEQ &JDATE &JTIME &JOBNBR &JOBUSR &JOBUS2)
     .
     .
.....5      31.00  CHGVAR &CMD ('CALL PILOT/JOBEXEC (''' || &JOB || '''  +
.....5      32.00  ''' || &SEQ || '''  ' || &JDATE  || ' ' || &JTIME  || ' )')

*************************************************************************************************************
CALL    JOBEXEC     PILOT      CLP PILOT#/QCLSRC JOBEXEC           Job execution pre-program
*************************************************************************************************************
......6       1.00  PGM (&JOB &SEQ &DATE &TIME)
......6       2.00  DCL &JOB *CHAR 10
......6      15.00  OVRDBF JOBTRG PILOT/JOBTRG
......6      17.00  CALL PILOT/JOBEXE (&JOB &SEQ &CODE)

The following overrides will be applied:
From: JOBDTL     To: JOBDTL     of: PILOT      Member:
      JOBHDR         JOBHDR         PILOT
      JOBTRG         JOBTRG         PILOT

*************************************************************************************************************
CALL    JOBEXE      PILOT      RPG PILOT#/QRPGSRC JOBEXE           Job Execution
*************************************************************************************************************
.......7       2.00       FJOBSCHL3UF  E           K        DISK                      A
.......7       3.00       FJOBTRG  IF  E           K        DISK
.......7      56.00       C                     PARM           @JOB   10
.......7      57.00       C                     PARM           JOBSQN
.......7      58.00       C                     PARM           SCHCDE

*************************************************************************************************************
RETURN JOBEXEC     PILOT      CLP PILOT#/QCLSRC JOBEXEC           Job execution pre-program
*************************************************************************************************************
......6      18.00  IF (&CODE=A) THEN(DO)
......6      19.00              RTVJOBA USER(&USER) NBR(&NUMBER)
......6      20.00              CHGVAR &MSGDT2 (&JOB *CAT &USER *CAT &NUMBER *CAT 'PILOT')
     .
     .
......6      27.00  ENDPGM

*************************************************************************************************************
RETURN JOBSBMC     PILOT      CLP PILOT#/QCLSRC JOBSBMC           Submit job to execute
*************************************************************************************************************
.....5      33.00             IF (&JOBQ='*JOBD') THEN(DO)
.....5      34.00  IF (&OUTQ='*JOBD') THEN(SBMJOB JOB(&JOB) JOBQ(*JOBD) OUTQ(*JOBD) +
     .
     .
*************************************************************************************************************
RETURN MENUC       PILOT      CLP PILOT#/QCLSRC MENUC              Pilot menu driver
*************************************************************************************************************
.1     132.00  END:
.1     137.00       ENDPGM
```

Print Object Flow (PRTOBJFLW) Command

The flowcharting function of ABSTRACT will analyze the cross reference information that tracks the transfers of program control within an application and the file usage that occurs among the application programs.

This function is similar to the PRTFLWCHT command above. It differs in that it offers greater selectivity over objects included on the report and control over the number of levels of nesting the process should include. PRTOBJFLW will only produce the graphic style representation of your application job streams.

Access the flowcharting functions by typing the Print Flowchart (PRTOBJFLW) command on a command entry line. The command syntax is shown below.

```
                     .-*CURLIB------.
>>--PRTOBJFLW----LIB-+--------------+----------------------------------------------------->
                     +-library_name-+'
                     +-*PRV---------+
                     '-*ALL---------'

                                                                                Required
==========================================================================================
                                                                                Optional

       .-*ALL------.           .-*ALL--------.           .-*ALL------.
>--OBJ-+-----------+---OBJTYPE-+-------------+---OBJATR-+-----------+-------------------->
       +-name------+           '-object_type-'           +-attribute-+
       '-*generic*-'                                     '-*generic--'

          .-*PRV---.          .-0------.          .-*PRV---.
>--DSPLVL-+--------+---EXPLVL-+--------+---DTASET-+--------+----------------------------><
          '-number-'          '-number-'          +-*FIRST-+
                                                  '-name---'
```

The object type and attribute parameters permit the user to produce flowcharts for all object types or for specific object types. This control applies to only to the top level objects for which the flow charts are produced. For example, a flow chart for command type objects would show the commands as the top level objects along with programs and files used in processing the commands.

LIB Parameter

Specifies the libraries that are searched for the objects indicated by the other parameters on the command. The possible library values are:

***PRV:** The library used during the previous session, regardless of which of the ABSTRACT WRKxxx commands was used.

***CURLIB:** The current library is searched. If no library is specified as the current library for the job, the QGPL library is used.

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***USRLIBL:** Only the libraries in the user portion of the library list are searched.

***ALLUSR:** All non-system libraries, including all user-defined libraries and the QGPL library, are searched. Libraries with names starting with the letter Q, other than QGPL, are not included.

***ALL**: All libraries in the system, including QSYS, are searched.

library-name: Only the specific library is searched for the objects.

OBJ Parameter

Specifies the object name. You can use this parameter to work with all the objects or a subset of objects in the indicated library.

<u>***ALL:**</u> All objects in the specified library(s) are selected

object-name: Only objects with this specific name will be included in the list.

***generic*-name:** Specify a partial object name qualified by an asterisk (*) to select several objects meeting the criteria. The following generic forms are allowed: ABC*, *ABC, *ABC*, A*B, "abc*", **ALL. Refer to Appendix G General Functions, in Volume 1 of the CL Reference manual for more information about generic names.

OBJTYPE Parameter

Specifies the object type. You can use this parameter to work with all object types or a subset of objects. An expanded description of the OBJTYPE parameter and a list of valid OS/400 object types can be found in Appendix A, Expanded Parameter Descriptions, of Volume 1 of the CL Reference manual.

<u>***ALL:**</u> All objects regardless of their type are included.

object-type: Specify any valid system object type to display a list of all objects of that particular type.

***DBF:** All data base files will be included. These are *FILE objects with an attribute beginning with PF, LF, or DDMF.

***SRCF:** All source physical files will be included. These are *FILE objects with an attribute of PF-SRC or PF38-SRC.

OBJATR Parameter

Specifies the object attribute. You can use this parameter to work with all object attributes or only specific ones. An expanded description of the OBJATR parameter and a list of valid OS/400 object attributes can be found in Appendix A, Expanded Parameter Descriptions, of Volume 1 of the CL Reference manual.

<u>***ALL:**</u> All objects in the specified library(s) are selected regardless of their attribute definition.

***BLANK:** Only objects with no attribute will be included.

object-attribute: Only objects with this specific attribute will be included in the list. PF-DTA can be used to select only non-source physical files. PF-SRC can be used to select only source physical files.

***generic*-name:** Specify a partial object attribute qualified by an asterisk (*) to select several objects meeting the criteria. The following generic forms are allowed: ABC*, *ABC, *ABC*, A*B, "abc*", **ALL. For example, *RPG* will select RPG, RPG36, RPG38, and SQLRPG objects. Refer to Appendix G, General Functions, in Volume 1 of the CL Reference manual for more information about generic names.

DSPLVL Parameter

Specifies the maximum depth of the relationship display. This value controls the number of indented levels that appear under the leftmost (parent) objects.

<u>*PRV:</u> The level used for this parameter during the previous session.

numeric: Specify a value between zero (0) and ten (10) that indicates the maximum level of indentation to appear on the object list display.

EXPLVL Parameter

Specifies the number of recursive iterations for the list explosion. This value determines the depth of the explosion tree before additional relationships are ignored.

<u>0:</u> No explosion recursion occurs for objects on the display. All references for the "parent" object are given, but references for the underlying objects are not.

numeric: Specify a value between zero (0) and ninety-nine (99) that indicates the maximum number of reference generations that should be shown for each "parent" object on the display.

DTASET Parameter

Specify the cross reference data set to be used in acquiring the information.

<u>*PRV:</u> Use the value for this parameter from the previous session. If no previous session has occurred, use *FIRST.

***FIRST:** information from the first (default) data set should be accessed.

Name: Specify any existing data set. Use the command prompter to find the names available to you.

Print Network Configuration (PRTNETCFG) Command

The Print Network Configuration (PRTNETCFG) command will analyze the configuration objects on your system and print a pictorial diagram showing the relationships among them. No cross references are used in generating the report. All information is obtained when the command runs.

Request the PRTNETCFG report by selecting option 12 from the APTOOL programming tools menu, or by specifying the command on a command entry line. The command can be run interactively or in a batch subsystem. It can also be invoked from an interactive or batch program. Because it can take several minutes to generate, the PRTNETCFG report is always submitted to the batch subsystem if selected from the APTOOL menu.

```
                         .-*LIBL-------.
>>--PRTNETCFG----SYSTEM-+-------------+---------------------------------------------><
                         '-system_name-'
```

SYSTEM Parameter

You can print a network configuration for the local system, or for a remote system. If the remote system does not have ABSTRACT installed, you must first restore a few ABSTRACT objects onto the remote system. Use SAVOBJ (NET* DEV*) on the system with ABSTRACT to save the required objects. On the remote system, create an ABSTRACT library and restore the saved objects.

***LOC:** Print network configuration for the local system.

Name :Print the network configuration for the named remote system.
The configuration information will be acquired and the report will be placed in your job's output queue when the command completes. Depending on your configuration, there may be several parts to the report.

Line descriptions and the attached controllers and devices are presented first on the report. Line rate, protocol, etc. are provided for the lines that are shown. Similar details are provided for the attached controllers and devices.

The example below depicts a line, its controller and three devices attached to it. Additional controllers would be listed beneath the first one if they existed.

```
 10/31/2010                            Network Configuration                                    Page   1
15:41:38

 ********LINE********
 *                  *
 * DIRECT_401       *
 * *SDLC  *NONSWTPP *
 * 9600 Baud        *
 *                  *
 ********************
         |
         |
         |
 ****CONTROL UNIT****          ***************             ***************             ***************
 *                  *         **             **           **             **           **             **
 *   DIRECT_401     *        ** ASC38         **         ** ASC402        **         ** DIRECT_401    **
 *   *APPC          *-----> **  *APPC         ** -----> **  *APPC         ** -----> **  *APPC         **
 *                  *        ** LOCADR-00      *         ** LOCADR-00      *         ** LOCADR-00      *
 *                  *         **             **           **             **           **             **
 ********************          ***************             ***************             ***************
```

Following the line descriptions, controllers that are not associated with a line description are printed with their attached devices. Controllers appear at the left and devices are listed to the right. Display devices are pictured with a rounded form, printer devices are shown with an alternative form (c.f. PRT01 below).

```
 10/31/2010                            Network Configuration                                    Page   2
15:41:38

 ********************          ***************             ***************             ***************
 *                  *         **             **           **             **           **             **
 *   ASC_DIAL2      *        ** D3180         **         ** D3196         **         ** D5292         **
 *   5394-2         *-----> **  3180-2        ** -----> **  3196-A1       ** -----> **  5292-2        **
 *                  *        ** LOCADR-02      *         ** LOCADR-00      *         ** LOCADR-03      *
 *                  *         **             **           **             **           **             **
 ********************          ***************             ***************             ***************
```

Device names, types, switch settings, and keyboard type are listed for each device attached to the workstation controller.

```
 10/31/2010                                Network Configuration                                    Page   3
15:41:38

 ********WSC*********
 *                  *
 *   CTL01          *
 *   6040           *
 *                  *
 *                  *
 ********************
         |
         |
         |
         |
         |
         |
 *******PORT*********          ***************             ***************
 *                  *         **             **           **             **
 *                  *        ** DSP01         **         ** DSP10         **
 *   PORT 0         *-----> **  3196-A1       ** -----> **  3196-A1       **
 *                  *        ** SWTSET-0  USB  *         ** SWTSET-2  USB  *
 *                  *         **             **           **             **
 ********************          ***************             ***************
         |
         |
         |
 *******PORT*********       ********************           ***************             ***************
 *                  *       *                 **          **             **           **             **
 *                  *       *   PRT01         **         ** DSP03         **         ** DSP04         **
 *   PORT 1         *-----> *   5225-1      ***  -----> **  3180-2        ** -----> **  5292-2        **
 *                  *       *   SWTSET-0  ***            ** SWTSET-1  USB  *         ** SWTSET-2  USB  *
 *                  *        *          ***               **             **           **             **
 ********************        ***********                   ***************             ***************
         |                     ***************             ***************
         |                    **             **           **             **
         |                   ** DSP07         **         ** DSP08         **
         |           -----> **  3476-EA       ** -----> **  5150-1        **
         |                   ** SWTSET-5  USB  *         ** SWTSET-6  USB  *
         |                    **             **           **             **
         |                     ***************             ***************
         |
         |
         |
 *******PORT*********          ***************             ***************             ***************
 *                  *         **             **           **             **           **             **
 *                  *        ** DSP11         **         ** DSP12         **         ** DSP13         **
 *   PORT 2         *-----> **  3180-2        ** -----> **  3180-2        ** -----> **  5292-2        **
 *                  *        ** SWTSET-0  USB  *         ** SWTSET-1  USB  *         ** SWTSET-2  USB  *
 *                  *         **             **           **             **           **             **
 ********************          ***************             ***************             ***************
                               ***************             ***************             ***************
                              **             **           **             **           **             **
                             ** DSP05         **         ** DSP06         **         ** DSP15         **
                     -----> **  3197-D1       ** -----> **  3197-D1       ** -----> **  3476-EA       **
                             ** SWTSET-4  USB  *         ** SWTSET-5  USB  *         ** SWTSET-6  USB  *
                              **             **           **             **           **             **
                               ***************             ***************             ***************
```

Virtual controllers and the virtual devices attached to them follow the workstation controller(s). Devices are pictured as before, workstations in an oval format; printer devices as a page format. Device information is presented as before, except that virtual devices do not have switch settings.

```
 10/31/2010                                Network Configuration                                    Page  17
13:13:17

 ********************          ***************             ***************             ***************
 *                  *         **             **           **             **           **             **
 *   QVIRCD0001     *        ** ANDREWS1      **         ** ANDREWS2      **         ** ANDREWS3      **
 *   *VWSC          *-----> **  3197-C1       ** -----> **  3197-C1       ** -----> **  5292-2        **
 *                  *        **    USB        **         **    USB        **         **    USB        **
 *                  *         **             **           **             **           **             **
 ********************          ***************             ***************             ***************
         |                     ***************             ***************
         |                    **             **           **             **
         |                   ** KENS1         **         ** KIRKS1        **
         |           -----> **  3180-2        ** -----> **  3180-2        **
         |                   **    USB        **         **    USB        **
         |                    **             **           **             **
         |                     ***************             ***************
 ********************          ***************             ***************             ***************
 *                  *         **             **           **             **           **             **
 *   VWSC           *        ** VWS01         **         ** VWS02         **         ** VWS03         **
 *   *VWSC          *-----> **  5251-11       ** -----> **  5251-11       ** -----> **  5251-11       **
 *                  *        **    USB        **         **    USB        **         **    USB        **
 *                  *         **             **           **             **           **             **
 ********************          ***************             ***************             ***************
         |                     ***************             ***************             ***************
         |                    **             **           **             **           **             **
         |                   ** VWS05         **         ** VWS06         **         ** VWS07         **
         |           -----> **  3180-2        ** -----> **  3197-D1       ** -----> **  3180-2        **
         |                   **    USB        **         **    USB        **         **    USB        **
         |                    **             **           **             **           **             **
         |                     ***************             ***************             ***************
         |                     ***************             ***************             ***************
         |                    **             **           **             **           **             **
         |                   ** VWS09         **         ** VWS10         **         ** VWS11         **
         |           -----> **  3477-FC       ** -----> **  3180-2        ** -----> **  5292-2        **
         |                   **    USB        **         **    USB        **         **    USB        **
         |                    **             **           **             **           **             **
         |                     ***************             ***************             ***************
```

System devices are presented at the end of the report. All directly attached devices (tapes, diskettes, printers, etc.) are shown in a pictorial format.

```
 10/31/2010                                 Network Configuration                                    Page  19
15:41:38

 ******SYSTEM********          ***************         ********************
 *                  *         **             **        *                  *
 *     SYSTEM       *        ** DKT01         **       *   TAP01          *
 *     DEVICES      *-----> **  9331-0001     ** -----> *   9347-0001      *
 *                  *        **               **       *                  *
 *                  *         **             **        *                  *
 ********************          ***************         ********************
```

Following the pictorial network configuration, a text description of the configuration objects on your system will be printed. Information for each object will be printed along with pertinent details of the description.

```
 10/31/2010                        Network Configuration                              Page  20
13:13:17

LINE/CTLU/DEV

DIRECT_401     Direct connect to AS400 C10         Type: *SDLC  Speed: 9600   Connection: *NONSWTPP

  DIRECT_401   Direct connection to AS400 C10      Type: *APPC
    ASC38      SDLC Device - AS400 C10 - ASC401          *APPC        Local address     :     00
    ASC402     AUTOMATICALLY CREATED BY QLUS             *APPC                                00
    DIRECT_401 SDLC Device - AS400 C10                   *APPC                                00
    KEN        AUTOMATICALLY CREATED BY QLUS             *APPC                                00
  .
  .

ASC_DIAL2      Dialup 5394 controller              Type: 5394

      D3180      Remote dialup 3180 display              3180 -2                               02
      D3196      Remote dialup 3196 display              3196 -A1                              00
      D5292      Remote dialup 5292-2 color graphics     5292 -2                               03

CTL01          Controller description by IPL.      Type: 6040

Port: 0
      DSP01      CREATED BY AUTO-CONFIGURATION           3196 -A1    Port,switch,locadr: 0  0  Keyboard: USB
      DSP10      CREATED BY AUTO-CONFIGURATION           3196 -A1                        0  2            USB

Port: 1
      PRT01      CREATED BY AUTO-CONFIGURATION           5225 -1                         1  0
      DSP02      CREATED BY AUTO-CONFIGURATION           3180 -2                         1  4            USB
      DSP04      CREATED BY AUTO-CONFIGURATION           5292 -2                         1  2            USB
      DSP07      CREATED BY AUTO-CONFIGURATION           3476 -EA                        1  5            USB
      DSP08      CREATED BY AUTO-CONFIGURATION           5150 -1                         1  6            USB

Port: 2
      DSP11      CREATED BY AUTO-CONFIGURATION           3180 -2                         2  0            USB
      DSP13      CREATED BY AUTO-CONFIGURATION           5292 -2                         2  2            USB
      DSP14      CREATED BY AUTO-CONFIGURATION           3476 -EA                        2  3            USB
      DSP06      CREATED BY AUTO-CONFIGURATION           3197 -D1                        2  5            USB
      DSP15      CREATED BY AUTO-CONFIGURATION           3476 -EA                        2  6            USB

QESCTL         SERVICELINK controller description   Type: *HOST

      QESPAP     SERVICELINK device description          *APPC                                 02

QSUPCTL                                            Type: 5294

      QSUPDSP                                            5291 -1                               00
      QSUPDSP2                                           3180 -2                               02
      QSUPPRT                                            5256 -3                               01

QTICTL         TIE, TIA, QA control unit desc      Type: *HOST

      QIADSP     TIA display device description                                                02
      QIAPRT     TIA printer device description                                                03
      QQAHOST    QA APPLICATION DEVICE DESCRIPTION       *APPC                                 06
      QTIDA      TIE device description                  *APPC                                 05
      QTIDA2     TIE device description 2nd session      *APPC                                 07

QVIRCD0001     Controller created for KENSPC.      Type: *VWSC

      ANDREWS1   Device created for ANDREW.              3197 -C1
      KENSPCS1   Device created for KENSPC.              3197 -C1
      KIRKS1     Device created for KIRK.                3180 -2

VWSC           Virtual Workstation Controller      Type: *VWSC

      VWS01      Virtual 5251-11                         5251 -11
      VWS05      Virtual 3180-2                          3180 -2
      VWS06      Virtual 3197-D1                         3197 -D1
      VWS08      Virtual 3477-FC                         3477 -FC
      VWS11      Virtual 5292-2                          5292 -2

The following are system devices:

      DKT01      CREATED BY AUTO-CONFIGURATION      Type: 9331 -0001
      TAP01                                              9347 -0001
```

The last part of the network configuration report is an inventory listing each type of object and the number of occurrences that were located. The device inventory looks like the sample below.

```
 10/31/2010                               Network Configuration                                   Page  21
13:13:17

Device     Type    Count
*APPC                23
*HOST                 2
*VWSC                 2
3180 2     Display   11
3196 A1    Display    4
3197 C1    Display    3
3197 D1    Display    3
3476 EA    Display    3
3477 FC    Display    2
5150 1     Display    1
5225 1     Printer    1
5225 4     Printer    1
5251 11    Display    4
5256 3     Printer    1
5291 1     Display    1
5292 2     Display    5
5294 1     Display    1
5394 2     WS Ctrl    1 Controller
6040 1     WS Ctrl    2 Controller
9331 0001  Dkt Rdr    1
9347 0001  Tape       1
```

Program Callable Functions

Two API’s are available for accessing ABSTRACT functions. These API’s offer programmable access to practically all of the ABSTRACT functions.

ABSTRACT Action Bar (APBAR)

This program can be called from anywhere and displays an action bar with a standard set of pull down windows. All of the ABSTRACT functions which are included on other menus are available through these pull downs.

The program is called without parameters and allows the user to select one option. The program performs (or submits) the requested function and returns to the calling program.

The following example shows the action bar overlayed on another screen.

```
_Initialization   Reference   Usage   File   Exception   Tool
------------------------------------------------------------------------------
Data Set . . *FIRST                           Position to name . . . ABPCLR____
                                              Position to type . . . *PGM______
                                              Position to library. . SEQUEL
Type option, then press Enter.
  2=Edit  5=Display  6=Print  8=Display description  9=Where used...
                                                 Library or
Opt Object/Uses          Type        Usage      Qualifier  Text
___ ABPCLR               *PGM CLP               SEQUEL     Clear messages from
___ ABPCLRSF             *PGM CLP               SEQUEL     Clear messages from
___ ABPSEND              *PGM CLP               SEQUEL     Send message from A
___ ANZAUDDTAC           *PGM CLP               SEQUEL     Extract journal ent
___  CPIALL              *FILE PF    Inp        SEQUEL     SEQUEL Outfile
___  CPI432A             *FILE PF    Unknown    SEQUEL     AUDIT:Query optimiz
___  R1                  *FILE                  *LIBL
___  R2                  *FILE                  *LIBL
___ ANZAUDDTAR           *PGM RPG               SEQUEL     ANZAUDDTA CPP
___  CPIALL              *FILE PF    Upd        SEQUEL     SEQUEL Outfile

Parameters or command
===> call_apbar______________________________________________________________
F3=Exit  F4=Prompt  F9=Retrieve  F10=Actions  F23=More Options  F24=More Keys
```

Work With X-ref Object Usage (WRKOBJUX) API

This application programming interface is available to make object usage information, including file and field usage, directly available to a user written program.

The API is called with 3 parameters:

1. Receiver size	Input	Pkd(4,0)
2. Request	Input	Char(71)
3. Receiver	Output	Char(152)

The receiver size parameter tells the API how many objects can be retrieved at one time. This size is determined by the definition of the receiver parameter in the calling program. If you are working with large numbers of objects in a request set (more than fit in your receiver data structure), you can call the API repeatedly. On each call, it continues through the objects in the request set beginning where it left off on the last call. If you want to skip the remaining objects at any time, simply issue a ‘FREE’ operation in your program. The next call to the API will initiate a new request.

Both request and the receiver parameters are defined through externally described data structures (APIINP and APIOUT) in the ABSTRACT library. The request parameter is a structure containing fields that correspond to the parameters on the WRKOBJUX command. Refer to the section on that command for a description of the parameters.

The receiver parameter is a multiple occurrence data structure containing all of the fields returned by the API for each object included in the requested set.

Request Structure

Field	Descriptions	Type	Length	Start	Stop
LIBPRM	Object library	Char	10	1	10
NAMPRM	Object name	Char	10	11	20
TYPPRM	OS/400 objects,*ALL,*FLD,*MBR,*SUB	Char	10	21	30
ATRPRM	Object attribute (not used in this API)	Char	10	31	40
LVLPRM	Object indentation level	Pkd	2, 0	41	42
APIOTH	Show other object types?	Char	1	43	43
APIFIL	Show files?	Char	1	44	44
APIFMT	Show formats?	Char	1	45	45
APIFLD	Show fields?	Char	1	46	46
APIPGM	Show programs?	Char	1	47	47
APIMBR	Show members?	Char	1	48	48
APISUB	Show subroutines?	Char	1	49	49
DSPUSG	Object usage	Char	10	50	59
RCMAX	Explosion level	Pkd	2, 0	60	61
SETPRM	Data set	Char	10	62	71

Receiver Structure

Field	Descriptions	Type	Length	Start	Stop
APIIND	Object indentation level	Pkd	2, 0	1	2
APIOBJ	Object	Char	10	3	12
APILIB	Object library	Char	10	13	22
APITYP	Object type	Char	10	23	32
APIATR	Object attribute	Char	10	33	42
APIUSG	Object usage	Char	10	43	52
APITXT	Object text	Char	50	53	102
APIEXT	Extended description	Char	50	103	152

API call example:

The following code sample is available in the ABSTRACT library in the QRPGSRC file and illustrates how the API can be called.

```
                                       SEU SOURCE LISTING                                        PAGE    1
  SOURCE FILE . . . . . . .  ABSTRACT/QRPGSRC
  MEMBER  . . . . . . . . .  APIXMP
  SEQNBR*...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8 ...+... 9 ...+...
0
    100      IAPIINP    E DSAPIINP
    200      I                                       43  49 APIREL
    300      IAPIOUT    E DSAPIOUT                   20
    400       *
    500       * Constants
    600      I              20                    C         #APIMX
    700       *
    800       * Set up API inputs
    900      C                     MOVEL'*ALL'    LIBPRM           Search all x-ref libs
   1000       ***************************************************************************
   1100       * Your file name would go on the next line                            *
   1200       ***************************************************************************
   1300      C                     MOVEL'OBJREF'  NAMPRM           File name
   1400      C                     MOVEL'*FILE'   TYPPRM           Object type
   1500      C                     MOVEL'*ALL'    ATRPRM           Object type
   1600      C                     Z-ADD10        LVLPRM           Display level
   1700      C                     MOVEL'YYYYYYY' APIREL           Show all object types
   1800      C                     MOVEL'*ALL'    DSPUSG           Show all object usage
   1900      C                     MOVE *ZERO     RCMAX            Do not explode
   2000      C                     MOVEL'*FIRST'  SETPRM           Use *FIRST data set
   2100       *
   2200       * Repeat loop until count is less than maximum
   2300      C                     Z-ADD#APIMX    APICNT
   2400      C           APICNT    DOULT#APIMX
   2500      C                     CALL 'APIOBJUX'APILST
   2600      C           APILST    PLIST
   2700      C                     PARM           APICNT  40
   2800      C                     PARM           APIINP
   2900      C                     PARM           APIOUT
   3000       * Loop through from 1 to APICNT occurrences of information
   3100      C                     DO   APICNT    I       30
   3200      C           I         OCUR APIOUT
   3300       ***************************************************************************
   3400       * Code to process information would go here                               *
   3500       ***************************************************************************
   3600      C                     ENDDO                           DO APICNT I
   3700      C                     ENDDO                           APICNT DOULT #APIMX
   3800      C                     SETON                     LR
   3900      C                     RETRN
                                  * * * *  E N D  O F  S O U R C E  * * * *
```

Using Option Codes

The ABSTRACT option file gives the object list displays much of their extraordinary power and flexibility. With it, you can define a wide range of function that you use during your interactive session.

The option file capabilities are similar to those provided by IBM's Programming Development Manager. In fact, the option file you may have created for use with PDM can be quickly and easily converted for use with ABSTRACT. Refer to the Convert PDM Option File (CVTPDMF) command in part 7 for the procedure to do this.

The ABSTRACT option file offers these significant advantages over the PDM capabilities:

- Three letter option codes
- Object type/subtype specific command sets
- Multiple commands for each option definition
- Options that attach to function keys
- Extended substitution parameters
- Accessibility of the option identifier and descriptive text via function key 23 from the object list display

ABSTRACT is shipped with an extensive option list in a file named in the ABSTRACT library. This is the default option file that is used when you begin working with ABSTRACT. You can work with the options in this file or create your own option file.

The standard option file has five members:

ABSTRACT	The standard options
SEQUEL	Standard options plus additional options that are useful if your company is a licensed user of Help/Systems' SEQUEL data retrieval and management product.
CMS	Standard options plus additional options that are useful if your company is a user of the Aldon/CMS change management system.
MKS	Same as above, except for MKS' Implementor package.
Turnover	Same as above, except for Softlandling's Turnover package.

If you are going to make changes to the option file, it is better to duplicate the ABSTRACT option file, place it into your own library, and make changes to the copy. Subsequent installation of a new version of ABSTRACT will not overwrite your changes.

Each time you use ABSTRACT the data area associated with your user profile is referenced to determine the option file that you used during your most recent ABSTRACT session. This option file is loaded and used for your current session. If you make a change to the option file name via function key 18 or the Option menu from the action bar, your choice will be saved and used the next time you access the product.

You can create a copy of the ABSTRACT option file by using the command:

```
CRTDUPOBJ OPTFILE ABSTRACT *FILE +
     TOLIB(Library) NEWOBJ(Name) DATA(*YES)
```

where *Library* and *Name* indicate the name of the object you want to create.

Type specific options

ABSTRACT options are both object type and attribute specific. Several definitions for a given option code can exist, each acting on a different type (or attribute) of object, each with a different set of commands! ABSTRACT will search for the option definition that most nearly matches the selected object and run the associated command(s).

For instance, if you have selected a data area object with an attribute of ‘SQLVIEW’ and placed an option code next to it, ABSTRACT will try to find an option code matching *DTAARA and SQLVIEW. If one does not exist it will attempt to find a *DTAARA and *ALL definition. If that fails, another attempt will be made to find a definition for *ALL objects and *ALL attributes. If none of these are found, the object is skipped over - no commands are executed.

This facility makes it very easy to have a single option code that performs the same general type of function - executing a set of commands that is determined by the type of object you are working with.

Multiple commands and error processing

ABSTRACT options can run several commands through a single option code. This is accomplished by allowing you to specify a command terminator (;) after each command you specify. By following each command with a semicolon and a space, you can string several commands together, up to a total of 250 characters per option definition.

ABSTRACT will automatically stop processing the list of commands as soon as the first error (escape message) occurs. You can prevent this from happening, and indicate that you want ABSTRACT to continue processing the command list regardless of the errors that may occur, by using the special MONMSG instruction each time you want uninterrupted processing.

The MONMSG instruction must be entered in uppercase letters. You may optionally follow the MONMSG instruction with another command. This command will be executed only if the preceding command ended in error.

For example, the following command string will delete a command definition object and then re-create it. Refer to page 8-9 for a complete definition of the substitution parameters that can be specified in ABSTRACT option codes.

```
DLTCMD &L/&N ; MONMSG ; CRTCMD &L/&N &SRCLIB/&SRCFIL &SRCMBR
      ; MONMSG WRKSPLF
```

The first command in the definition causes the selected command object to be deleted. If an error occurs (perhaps the command is not found) processing will continue and CRTCMD will attempt to create the command using the original source code identified by the command’s service data. If the MONMSG statement had been omitted, processing would stop if the DLTCMD resulted in an error. The CRTCMD command would not run if an error had been detected. Note that MONMSG handles all errors - the CRTCMD above will run regardless of the type of escape error generated by DLTCMD.

If the create command ends abnormally, the MONMSG instruction following the command will be executed. The Work With Spooled Files (WRKSPLF) command will display the spool file list so that the errors can be detected.

When the command string is processed, ABSTRACT will write the entire set of commands into your joblog so that they can be retrieved (and reprocessed) via a function key 9 request. Each command within the set is also issued individually to the joblog as it is executed.

Function Key options

Function keys 2, 6, 15, and 22 are not used by the ABSTRACT object list displays. You can create option definitions that will be available from these function keys by using option codes F02, F06 and F22. The command(s) that you define in the option definition will be run when the corresponding function key is pressed. If the cursor is positioned on an item in the object list display when the key is pressed, the object will be used for substitution information referenced by the command(s) in the definition.

Viewing the options

You can view the available options (and their associated descriptions) on the ABSTRACT object list displays provided you are not using them in the full screen mode. You can change the display mode to show options and command keys by pressing function key 18 or by using the View menu from the action bar accessed via function key 10.

Cycle through the list of options by pressing function key 23. Each time you press it, a different line of options will be presented near the top of the display. When you reach the end, the list will wrap around to the beginning.

The option file cannot be used to regulate five standard options that are always available on the object list displays. Entries that attempt to define these options will be ignored:

1=Select

This option will choose the corresponding object for a function to be performed through the action bar, the command line, or the Select feature of the option display. Multiple objects can be selected with a single request.

3=Copy, 4=Delete, 7=Rename, 11=Move

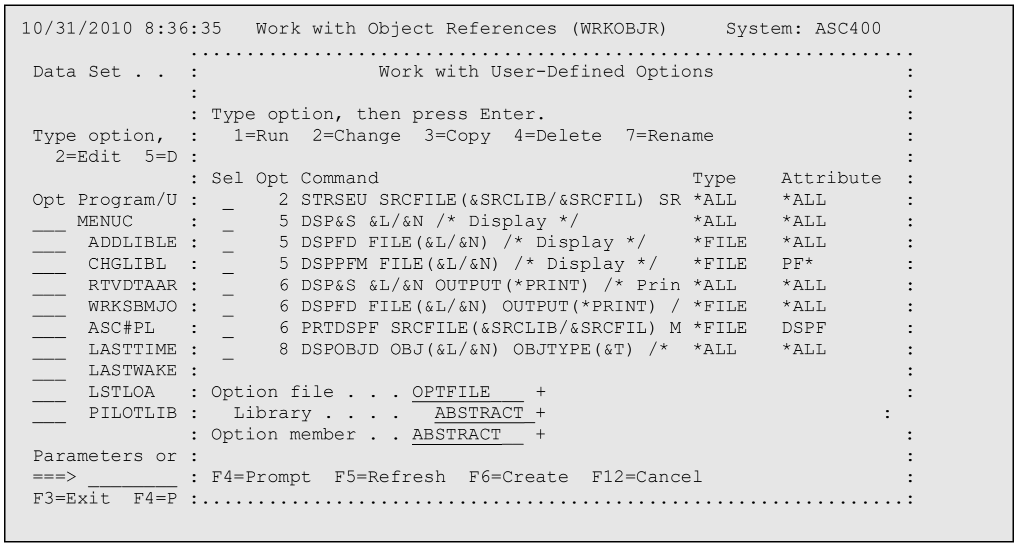
These options will perform the indicated action on the selected object(s). A confirmation panel (delete) or a dialog panel (copy, rename, move) will appear so that you can complete the request.

Working with option definitions

You can display, add, change, or delete entries from the option file you are using through a series of interactive displays.

If one or more objects have been selected (1=Select) before working with the option file, they will be referenced if you choose to run any of the options while working with them. This can be a very easy and effective method of executing several options against a list of objects one after the other.

Begin working with the items in the option file by pressing F16 from an ABSTRACT menu or one of the object list displays, or by selecting the Options part of the action bar. A window, similar to the one overlaying the display below will appear.


```
10/31/2010 8:36:35   Work with Object References (WRKOBJR)     System: ASC400
               ................................................................
 Data Set . .  :                Work with User-Defined Options                 :
               :                                                               :
               : Type option, then press Enter.                                :
 Type option,  :   1=Run  2=Change  3=Copy  4=Delete  7=Rename                 :
   2=Edit  5=D :                                                               :
               : Sel Opt Command                            Type    Attribute  :
 Opt Program/U :  _    2 STRSEU SRCFILE(&SRCLIB/&SRCFIL) SR *ALL    *ALL       :
 ___ MENUC     :  _    5 DSP&S &L/&N /* Display */          *ALL    *ALL       :
 ___  ADDLIBLE :  _    5 DSPFD FILE(&L/&N) /* Display */    *FILE   *ALL       :
 ___  CHGLIBL  :  _    5 DSPPFM FILE(&L/&N) /* Display */   *FILE   PF*        :
 ___  RTVDTAAR :  _    6 DSP&S &L/&N OUTPUT(*PRINT) /* Prin *ALL    *ALL       :
 ___  WRKSBMJO :  _    6 DSPFD FILE(&L/&N) OUTPUT(*PRINT) / *FILE   *ALL       :
 ___  ASC#PL   :  _    6 PRTDSPF SRCFILE(&SRCLIB/&SRCFIL) M *FILE   DSPF       :
 ___  LASTTIME :  _    8 DSPOBJD OBJ(&L/&N) OBJTYPE(&T) /*  *ALL    *ALL       :
 ___  LASTWAKE :                                                               :
 ___  LSTLOA   : Option file . . . OPTFILE___ +                                :
 ___  PILOTLIB :   Library . . . .   ABSTRACT +                              :
               : Option member . . ABSTRACT__ +                                :
 Parameters or :                                                               :
 ===> ________ : F4=Prompt  F5=Refresh  F6=Create  F12=Cancel                  :
 F3=Exit  F4=P :...............................................................:
```

The window lists the currently defined options and shows the name of the option file you are using. The option definition includes the option code and up to 45 characters of the option's command and description. Use the roll keys to see the complete list of option definitions.

Select a different option file by typing over the file, library, and/or member name shown on the display and pressing the Enter key. A new list will appear showing you the option definitions for the file you have selected.

Press one of the function keys listed at the bottom of the display to carry out the associated action.

You may perform additional functions on the option(s) by typing a selection value in the 'Sel' column of the display and pressing the Enter key. Use the following values to request the corresponding function:

1=Run Run option(s) against object(s) you have selected from the object list display. If no objects have been selected, run the option(s) with blank substitution data. Press function key 4 (instead of Enter) to prompt the command(s) prior to running them.

2=Change Change the option description, command list, and/or object type and subtype criteria. A window similar to the one on page 8-6 will appear.

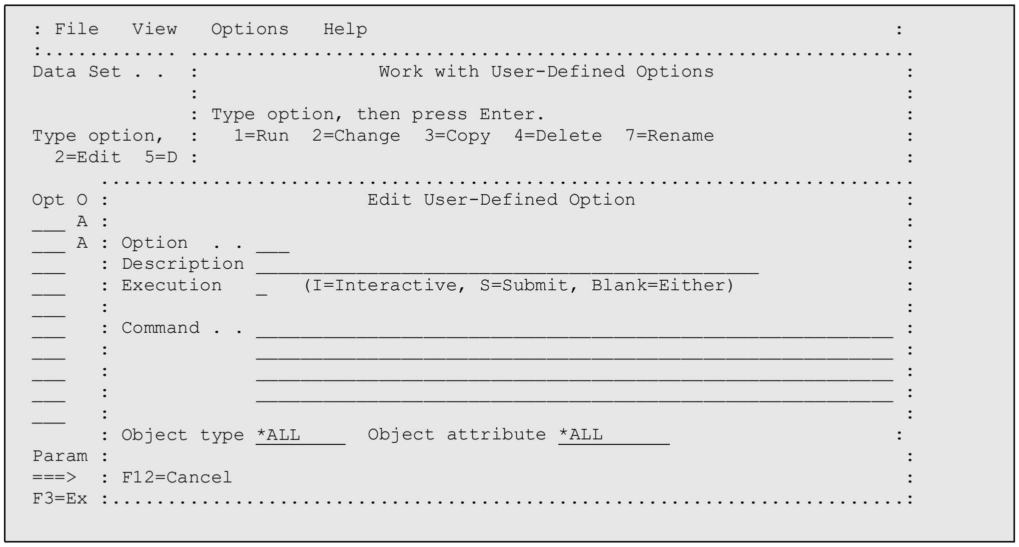
3=Copy Copy this option and create a new option. Refer to page 8-7.

4=Delete Delete this option item. Refer to page 8-8.

7=Rename Rename the option code for this item. Refer to ppage 8-9.

Creating a new option

Press function key 6 from the option display to create a new option. A window like the one on the next page will appear. It allows you to enter a new option definition.



```
: File   View   Options   Help                                          :
:............ ..............................................................
Data Set . .  :                Work with User-Defined Options             :
              :                                                           :
              : Type option, then press Enter.                            :
Type option,  :   1=Run  2=Change  3=Copy  4=Delete  7=Rename            :
  2=Edit  5=D :                                                           :
      ....................................................................
Opt O :                        Edit User-Defined Option                   :
___ A :                                                                   :
___ A : Option  . . ___                                                   :
___   : Description ___________________________________________           :
___   : Execution   _   (I=Interactive, S=Submit, Blank=Either)           :
___   :                                                                   :
___   : Command . . _________________________________________________________ :
___   :             _________________________________________________________ :
___   :             _________________________________________________________ :
___   :             _________________________________________________________ :
___   :                                                                   :
      : Object type *ALL____   Object attribute *ALL______               :
Param :                                                                   :
===>  : F12=Cancel                                                        :
F3=Ex :...................................................................:
```

Type the option code and descriptive text that will identify its function. The code and description will appear on the option line of the object list display along with the other available options.

Use the Execution entry to specify whether the option should be restricted to the interactive environment, automatically submitted to batch when selected, or submitted at the user's discretion. If the user's "Run in batch" default value (F18) is set to Y and the Execution option is coded with an S, the option's command(s) will be automatically submitted to the batch subsystem. Otherwise, the command(s) will be submitted to batch only when the user specifically requests batch execution via F14 or F15.

Next, type one or more commands into the command portion of the display. If you are specifying more than one command, the commands in the list must be separated by a semicolon followed by a space (;). Use substitution variables from the list on page 8-9. You can prompt an individual command, but not a command list, by pressing function key 4.

Selective prompting requests that include question marks ('?') are also allowed. Refer to the CL Programmer's Guide for complete information about the selective prompting characters.

Object type/attribute

Finally, identify the type and attribute of the objects that will qualify for this option definition. Any OS/400 object type (*PGM, *FILE, etc.) is allowed, in addition to *DBF (database file), *FLD (field), *MBR (member) and *SUB (subroutine).

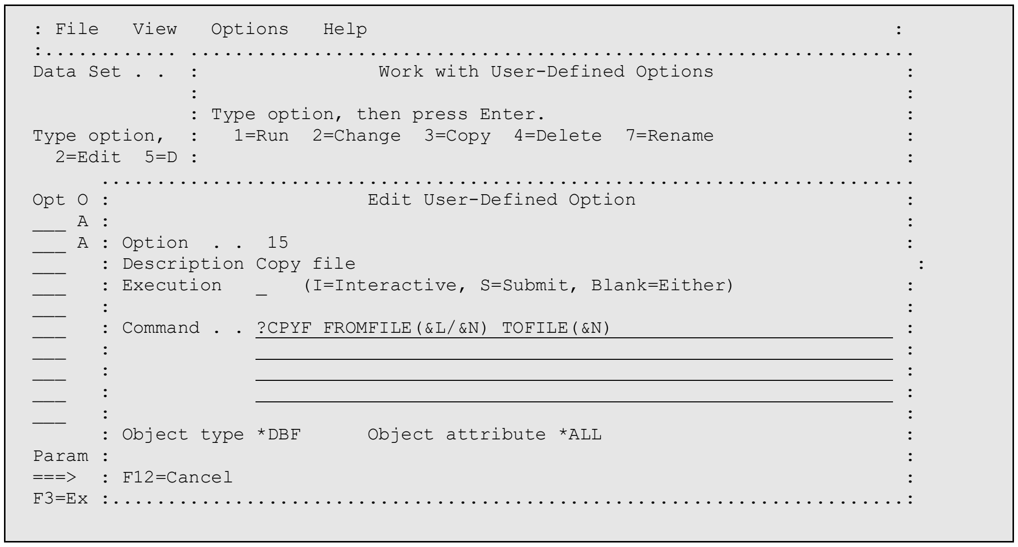
Use a specific attribute or a generic value - a series of characters followed by an asterisk (*). For example use RPG to indicate that only RPG programs will qualify; PF* to select any physical file (PF or PF38).

If the option should apply to all objects, specify *ALL for the object type and *ALL for the object attribute.

Press Enter to add the definition to the list of available options. Press function key 12 to return to the previous display without creating a new definition, or use function key 3 to exit the function completely.

Changing an option

If you select the 2=Change function from the option list display (page 8-3), a window similar to the one below will appear. Use it to change an existing option; modifying the option description, command definitions, or its object specificity.



```
: File   View   Options   Help                                        :
:........... .............................................................
Data Set . .  :                Work with User-Defined Options          :
              :                                                         :
              : Type option, then press Enter.                          :
Type option,  :   1=Run  2=Change  3=Copy  4=Delete  7=Rename          :
  2=Edit  5=D :                                                         :
      ...................................................................
Opt O :                       Edit User-Defined Option                  :
___ A :                                                                 :
___ A : Option  . .  15                                                 :
___   : Description Copy file                                            :
___   : Execution   _   (I=Interactive, S=Submit, Blank=Either)         :
___   :                                                                 :
___   : Command . . ?CPYF FROMFILE(&L/&N) TOFILE(&N)                    :
___   :             ________________________________________________    :
___   :             ________________________________________________    :
___   :             ________________________________________________    :
___   :                                                                 :
      : Object type *DBF      Object attribute *ALL                     :
Param :                                                                 :
===>  : F12=Cancel                                                      :
F3=Ex :.................................................................:
```

Change any portion of the definition simply by typing over it. Identify the type and attribute of the objects that will qualify for the option definition by using any OS/400 object type (*PGM, *FILE, etc.) or one of the ABSTRACT additional types (*DBF, *FLD, *MBR, *SUB). If the option should apply to all objects, specify *ALL for the object type and *ALL for the object attribute.

The example above shows an ABSTRACT definition for option code 15. It applies to all database files. Any objects that are neither physical or logical files will not be processed by this definition. In addition, the option can process all physical or logical files, regardless of their attribute (PF38, LF, etc.). The Copy File (CPYF) command will be prompted and then run by the option. The selected object's name will be used as the target file name.

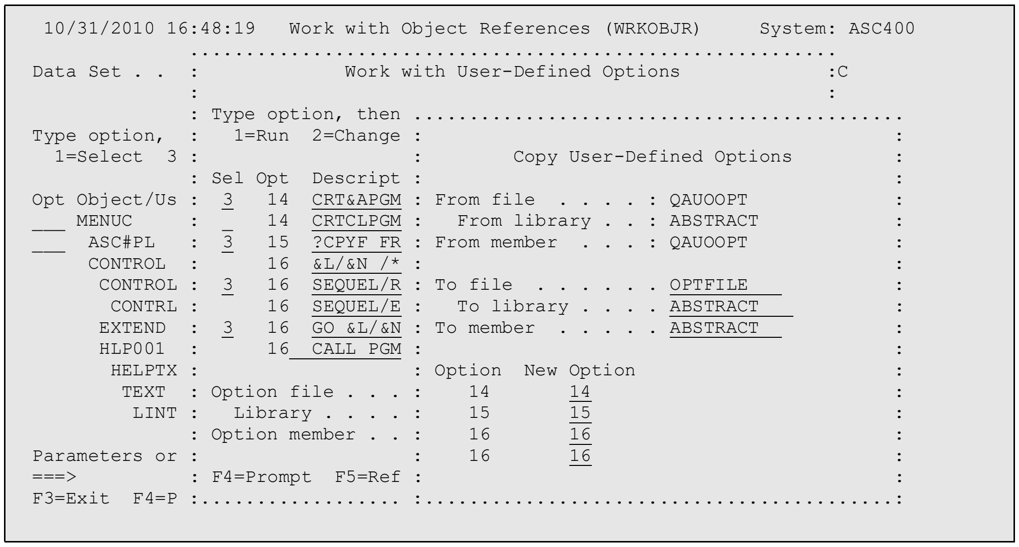
Use function key 4 to prompt the first command in the list. You can use the command prompting display to fill in both values and substitution variables. Press Enter from the command prompting display to return the information you have typed to the option definition, or use function key 12 to return to the option definition without returning any new information.

Press Enter to carry out the modifications you have made. Press function key 12 to return to the previous display without changing the definition, or use function key 3 to exit the function completely.

Copying an option

Select the 3=Copy function from the option list display (page 8-3), to indicate that you want to copy an existing option definition and create a new one. Once you press Enter, a window like the one below will appear. It allows you to copy an option definition to a different option file, to a new option code, or both. Although you can copy several option definitions with a single request, all of them will be copied to the same target file.

The copy window looks like this:

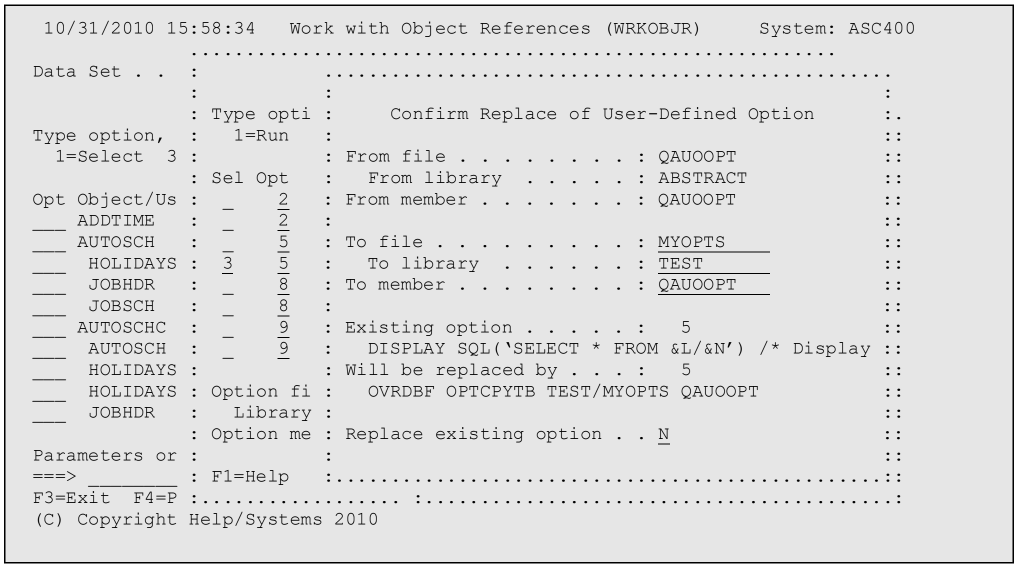


```
 10/31/2010 16:48:19   Work with Object References (WRKOBJR)     System: ASC400
              .......................................................
Data Set . .  :             Work with User-Defined Options            :C
              :                                                       :
              : Type option, then .........................................
Type option,  :   1=Run  2=Change :                                       :
  1=Select  3 :                   :        Copy User-Defined Options      :
              : Sel Opt  Descript :                                       :
Opt Object/Us :  3   14  CRT&APGM : From file  . . . . : QAUOOPT          :
___ MENUC     :  _   14  CRTCLPGM :   From library . . : ABSTRACT         :
___   ASC#PL  :  3   15  ?CPYF FR : From member  . . . : QAUOOPT          :
      CONTROL :      16  &L/&N /* :                                       :
       CONTROL : 3   16  SEQUEL/R : To file  . . . . . . OPTFILE          :
        CONTRL :     16  SEQUEL/E :   To library . . . . ABSTRACT         :
       EXTEND  : 3   16  GO &L/&N : To member  . . . . . ABSTRACT         :
       HLP001  :     16  CALL PGM :                                       :
        HELPTX :                  : Option  New Option                    :
         TEXT  : Option file . . . :    14       14                       :
          LINT :   Library . . . . :    15       15                       :
              : Option member . . :    16       16                        :
Parameters or :                   :    16       16                        :
===>          : F4=Prompt  F5=Ref :                                       :
F3=Exit  F4=P :.................. :.......................................:
```

The copy prompt shows the name of the current option file and the name of the target option file, in addition to the current and target option number. To create a copy of the selected option definition, change the name of the target file and/or specify the option code you want to create and press the Enter key. To create a copy in the current file having the same option code, simply press Enter without making any changes to the display.

Once you have pressed the Enter key, the option definition will be added to the indicated file. If you have not changed the target file (i.e. the target is the same as the current file) and an option definition already exists with this option code and object type/attribute combination, an edit window will appear for the new option and the object type for the new definition will be set to *COPY. You can then change the command sequence, or the object applicability of the new option definition. If you press function key 12 from the edit window, the copy will be removed.

If you are copying the option to another file and a definition with the same code and object specificity already exists, a confirmation window, similar to the one below will appear. It informs you of the conflict and lets you choose whether to replace the existing option, or cancel the copy operation.

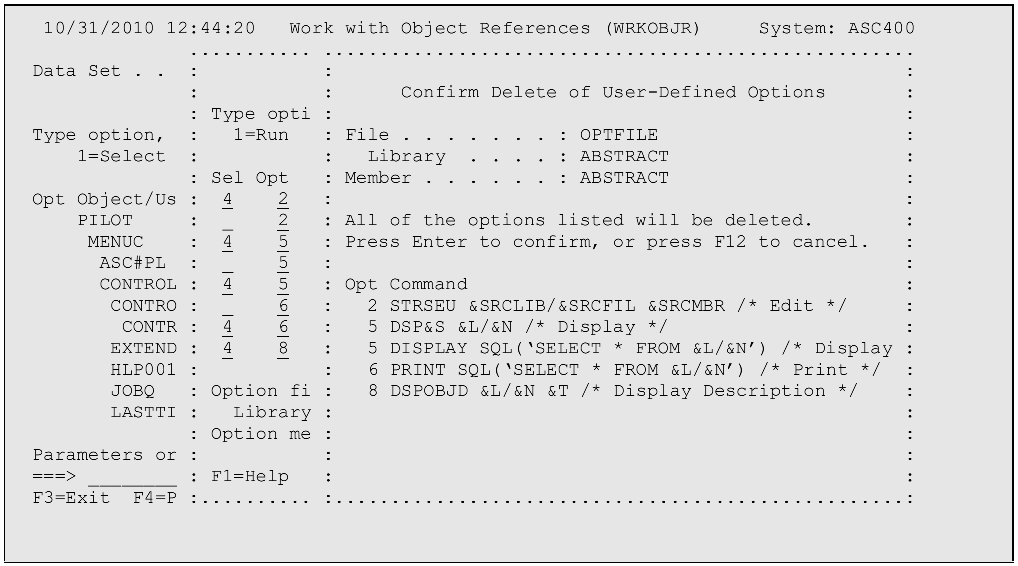


```
 10/31/2010 15:58:34   Work with Object References (WRKOBJR)     System: ASC400
              ..........................................................
Data Set . .  :           ..................................................
              :           :                                                :
              : Type opti :     Confirm Replace of User-Defined Option     :.
Type option,  :   1=Run   :                                                ::
  1=Select  3 :           : From file . . . . . . . . : QAUOOPT            ::
              : Sel Opt   :   From library  . . . . . : ABSTRACT           ::
Opt Object/Us :  _    2   : From member . . . . . . . : QAUOOPT            ::
___ ADDTIME   :  _    2   :                                                ::
___ AUTOSCH   :  _    5   : To file . . . . . . . . . : MYOPTS             ::
___  HOLIDAYS :  3    5   :   To library  . . . . . . : TEST               ::
___  JOBHDR   :  _    8   : To member . . . . . . . . : QAUOOPT            ::
___  JOBSCH   :  _    8   :                                                ::
___ AUTOSCHC  :  _    9   : Existing option . . . . . :   5                ::
___  AUTOSCH  :  _    9   :   DISPLAY SQL('SELECT * FROM &L/&N') /* Display ::
___  HOLIDAYS :           : Will be replaced by . . . :   5                ::
___  HOLIDAYS : Option fi :   OVRDBF OPTCPYTB TEST/MYOPTS QAUOOPT          ::
___  JOBHDR   :   Library :                                                ::
              : Option me : Replace existing option . . N                  ::
Parameters or :           :                                                ::
===> ________ : F1=Help   :................................................::
F3=Exit  F4=P :................ :.........................................:
(C) Copyright Help/Systems 2010
```

Type a ‘Y’ in the entry field at the bottom of the display to replace the existing option with the option you selected from the current file. The target file will be updated when you press the Enter key.

If you leave an ‘N’ in the entry field when Enter is pressed, or if you press function key 12 from this display, the copy operation will be terminated and the option will not be placed into the target file.

Deleting an option



```
 10/31/2010 12:44:20   Work with Object References (WRKOBJR)     System: ASC400
              .......... ..................................................
Data Set . .  :           :                                                  :
              :           :      Confirm Delete of User-Defined Options      :
              : Type opti :                                                  :
Type option,  :   1=Run   : File . . . . . . . : OPTFILE                     :
    1=Select  :           :   Library  . . . . : ABSTRACT                    :
              : Sel Opt   : Member . . . . . . : ABSTRACT                    :
Opt Object/Us :  4    2   :                                                  :
    PILOT     :  _    2   : All of the options listed will be deleted.       :
     MENUC    :  4    5   : Press Enter to confirm, or press F12 to cancel.  :
      ASC#PL  :  _    5   :                                                  :
      CONTROL :  4    5   : Opt Command                                      :
       CONTRO :  _    6   :   2 STRSEU &SRCLIB/&SRCFIL &SRCMBR /* Edit */    :
        CONTR :  4    6   :   5 DSP&S &L/&N /* Display */                    :
       EXTEND :  4    8   :   5 DISPLAY SQL('SELECT * FROM &L/&N') /* Display :
       HLP001 :           :   6 PRINT SQL('SELECT * FROM &L/&N') /* Print */ :
       JOBQ   : Option fi :   8 DSPOBJD &L/&N &T /* Display Description */   :
       LASTTI :   Library :                                                  :
              : Option me :                                                  :
Parameters or :           :                                                  :
===> ________ : F1=Help   :                                                  :
F3=Exit  F4=P :.......... :..................................................:
```

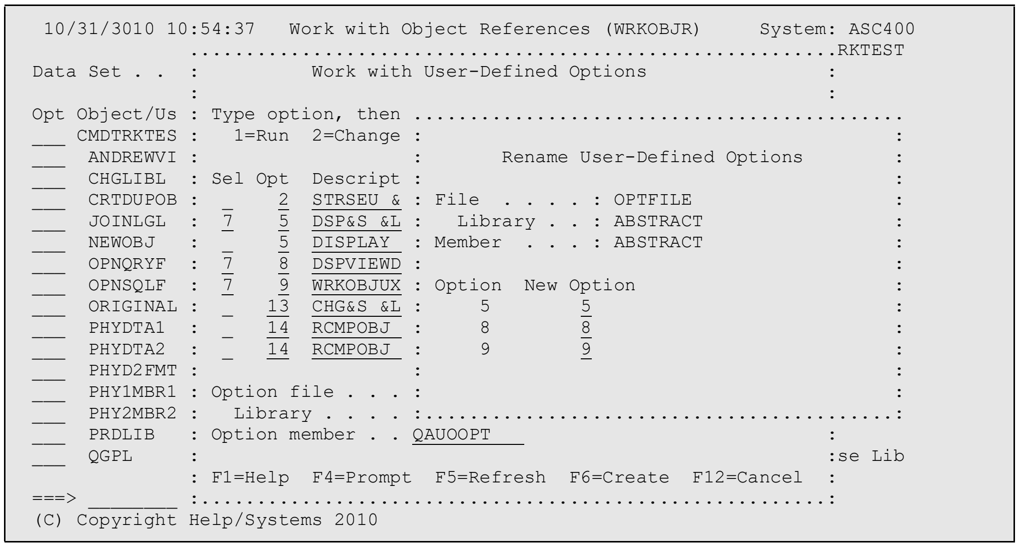
If you choose the 4=Delete function for one or more options on the option list display of page 99-98, a confirmation window like the one above will appear showing the options about to be deleted, and prompt you to confirm your request.

The example shows that five options have been chosen for deletion. Press the Enter key from this display to carry out the request and remove them from the option file. Press function key 12 to cancel the delete request and return to the option list display.

Renaming an option

The 7=Rename function allows you to change the option code for one of the options in the option file. From the option list display (page 8-3), place a '7' in front of all the options you want to rename and press the Enter key. The rename prompt shown below will overlay the option list.

Type the new option code for each option shown on the prompt and press the Enter key. The old option definition will be renamed and the new code assigned. If an option definition having the same code and object type/attribute combination already exists, an error will be signaled and the rename function will not be performed for the option.



```
 10/31/3010 10:54:37   Work with Object References (WRKOBJR)     System: ASC400
              ......................................................RKTEST
Data Set . .  :          Work with User-Defined Options               :
              :                                                       :
Opt Object/Us : Type option, then ........................................
___ CMDTRKTES :   1=Run  2=Change :                                      :
___  ANDREWVI :                   :       Rename User-Defined Options    :
___  CHGLIBL  : Sel Opt  Descript :                                      :
___  CRTDUPOB :  _    2  STRSEU & : File  . . . . : OPTFILE              :
___  JOINLGL  :  7    5  DSP&S &L :   Library . . : ABSTRACT             :
___  NEWOBJ   :  _    5  DISPLAY  : Member  . . . : ABSTRACT             :
___  OPNQRYF  :  7    8  DSPVIEWD :                                      :
___  OPNSQLF  :  7    9  WRKOBJUX : Option  New Option                   :
___  ORIGINAL :  _   13  CHG&S &L :     5       5                        :
___  PHYDTA1  :  _   14  RCMPOBJ  :     8       8                        :
___  PHYDTA2  :  _   14  RCMPOBJ  :     9       9                        :
___  PHYD2FMT :                   :                                      :
___  PHY1MBR1 : Option file . . . :                                      :
___  PHY2MBR2 :   Library . . . . :......................................:
___  PRDLIB   : Option member . . QAUOOPT___                          :
___  QGPL     :                                                       :se Lib
              : F1=Help  F4=Prompt  F5=Refresh  F6=Create  F12=Cancel  :
===> ________ :.......................................................:
(C) Copyright Help/Systems 2010
```

After the rename function completes the option list display will reappear. (See page 8-3)

You can cancel the rename request by pressing function key 12. The option list display will reappear and the rename code will be retained next to the options you selected.

Substitution variables

The commands that are run by the option definitions can be much more versatile if they include references to *substitution variables*. Similar to CL variables, they are substituted by a value when the command is run. By using a variable reference you can request that ABSTRACT acquire information about the object you have selected to be acted upon by the particular option. For example, the statement:

```
DSP&S &L/&N
```

could become any of the following commands depending on the object selected when the option is run:

```
DSPPGM PRODLIB/ORDENTRY
DSPCMD QSYS/WRKJOB
DSPJOBD ABSTRACT/ABSTRACT
```

The flexibility offered by the ABSTRACT substitution set is outstanding! Use the list of variables below in creating (and understanding) the options in the option file you are using.

&A	Object attribute (RPG for RPG programs, CLP for CL programs, PF for physical file (source or data), etc.) as it appears when the DSPOBJD command is run.
&ATR	Same as &A except that any trailing 38 (RPG38, CLP38, PF38, etc.) is removed.
&ATP	Same as &ATR except CLP is changed to CL. An option defined as CRT&ATPPGM becomes CRTCLPGM when used on a CLP type program.
&C	The option code used to invoke this definition.
&E	Run in batch flag. *YES if the 'Run/Compile in batch' parameter of the user's defaults is 'Y'. Otherwise *NO.
&F	The name of the object's source file. Equivalent to &SRCFIL.
&G	The current default job description library.
&H	The current default job description name.
&J	Qualified job description name in library/name format.
&L	The name shown under this object's 'Library' column.
&N	Object name for the select object.
&Q	Qualified job queue name in library/name format.
&S	Object type without a preceding '*'. &S could be replaced by PGM, FILE, CMD, etc.
&T	Object type with a preceding '*'. &T could be replaced by *PGM, *FILE, *CMD, etc.
&U	Name of the current option file in use.
&V	Library containing the current option file in use.
&W	Member of the current option file in use.
&X	Object text of the selected object, in quotes.
&DTASET	Name of the current data set you are using.
&PRNNAM	Object name of the selected object's leftmost parent object.
&PRNLIB	Library name of the selected object's leftmost parent.
&PRNTYP	Object type of the selected object's leftmost parent.

&SEQNBR	First sequence number listed in the SEQ() portion of the extended description for the selected item.
&SRC	Qualified source file used to create the object (from service data).
&SRCFIL	Source file used to create the object (service data).
&SRCLIB	Source library used to create the object (service data).
&SRCMBR	Source member used to create the object (service data).
&SRCTYP	SEU source type of the member indicated by the object's service data.
&SRCDTA	Current real-time source record corresponding to the first sequence number listed in the SEQ() portion of the extended description for the selected item.
&SRCRRN	Current real-time record number corresponding to the first sequence number listed in the SEQ() portion of the extended description for the selected item.

Obviously, you can create very powerful options using these substitution variables. With them it is very easy to simplify a task that involves complex command strings that must be done over and over again like program editing, compiling, and testing.

Learn how you might be able to use substitution variables in your own option definitions by reviewing the ABSTRACT option file that was delivered with the product: ABSTRACT/QAUOOPT

Examples

The ABSTRACT option definition file, QAUOOPT, has several definitions for option code '5':

```
Object
Type  AttributeCommand
*DTAARASQLVIEWSEQUEL/DISPLAY VIEW(&L/&N)
*DBF  *ALLSEQUEL/DISPLAY SQL('select * from &L/&N')
*ALL  *ALLDSP&S &L/&N
```

Depending on the type and attribute of the object selected, the 5=Display option will perform one of three actions. If the object is a SEQUEL*[1] View data area or a data base file, the SEQUEL DISPLAY command will be run to view its contents. Otherwise a command referencing the selected object's type will be used. This might result in the execution of virtually any of the OS/400 display commands (DSPPGM, DSPCMD, DSPDTAARA, etc.)

1. SEQUEL is a licensed program product from Help/Systems

Option definitions are not required to include substitution variables. Obviously, an option definition that makes no substitutions will work the same regardless of the object name that is “selected”. Here are some codes with and without substitution variables that you may want to add to your option file to help you get the most out of ABSTRACT:

```
Code Command
90   SIGNOFF
WAJ  WRKACTJOB/*Work with Active Jobs*/
WJL  WRKJOB OPTION(*JOBLOG)
DBG  ENDDBG; MONMSG; STRDBG &L/&N UPDPROD(*YES); ?CALL &L/&N
```

Index