IBM®

V5R3 iDoctor for iSeries Documentation

Previous Next Expand Collapse Search

Terms of use Privacy Close [x]

Table of Contents | Previous | Next

About This Document

This document provides information about iDoctor for iSeries running on an iSeries using IBM® i5/OS™ V5R3.

Table of Contents Previous Next

Part I iDoctor for iSeries Overview

This part covers basic information about each iDoctor for iSeries component and installation steps.

Table of Contents | Previous | Next

Chapter 1 General Information

iDoctor for iSeries is a suite of tools consisting of the following components:

- Job Watcher
- PEX Analyzer
- Heap Analysis Tools for Java™ (also known as Heap Analyzer)
- PTDV (Performance Trace Data Visualizer)

All of these tools assist the user with performance analysis or performance investigation on the iSeries. A brief explanation of each tool follows:

iDoctor for iSeries - Job Watcher
Job Watcher provides the user with the ability to collect many different types of information about sets of jobs (or all jobs) on a system. The type of data gathered includes wait times, CPU, IO activity, call stacks, SQL statements, communications, and activation group statistics. Many graphs at various scopes are available to help illustrate what jobs are performing badly on a system at a high level down to the low level interval by interval view of what an individual job or thread is doing.

This component is available for a trial evaluation or purchase from this website.

iDoctor for iSeries - PEX Analyzer
PEX Analyzer assists the user with the analysis of PEX (Performance Explorer) data. PEX is a component of i5/OS. The server-side of PEX Analyzer includes a command STRPACOL which simplifies the process of creating a collection by wrapping the i5/OS commands ADDPEXDFN, STRPEX and ENDPEX into one step.
The client side of PEX Analyzer includes many graphing and query capabilities which allow a user to quickly identify performance bottlenecks.

This component is available for a trial evaluation or purchase via this website.

iDoctor for iSeries - Heap Analysis Tools for Java
Heap Analyzer provides three different modes of operation which are useful for investigating Java memory related performance problems.
The first mode provides a dump of the Java Virtual Machine for the selected job. This dump includes object size, and class information for all objects within the JVM at the time the dump was collected.
The second mode provides a profile of the objects being created within the Java Virtual Machine.

This profile also includes a sample of call stacks.
The third mode of Heap Analyzer is called the object root finder. It produces a object hierarchical tree which can be used to determine where a particular object was rooted (i.e. to identify the class that created the object)

Heap Analyzer is a free component.

iDoctor for iSeries - PTDV (Performance Trace Data Visualizer)
Performance Trace Data Visualizer (PTDV) for iSeries is a tool for processing, analyzing, and viewing PEX data with an emphasis on the PEX events used in Java programs.

PTDV is a free component.

Table of Contents | Previous | Next

1.1 Job Watcher

Job Watcher returns real-time information about all jobs, threads and/or LIC tasks running on a system (or on a selected set of jobs/threads or tasks). The data is collected by a server job, stored in database files, and displayed on a client via the iDoctor GUI presentation facilities. Job Watcher is similar in sampling function to the system commands WRKACTJOB and WRKSYSACT in that each "refresh" computes delta information for the ending snapshot interval. Refreshes can be set to occur automatically, as frequently as every 100 milliseconds. The data harvested from the jobs/threads/tasks being watched is done so in a non-intrusive manner (similar to WRKSYSACT).

This "watch" data is summarized in many different types of reports and graphs by the client which can provide a quick picture of what is happening per thread when many different threads are being watched. As the situation merits, the user can select a job, an interval, and then drill down for the details while the watch is in progress or after a watch has ended.

For V5R3 the "engine" for Job Watcher was completely rewritten to make it faster and easier to setup and collect data system-wide.

The information harvested includes:

- Standard WRKSYSACT type info: CPU, DASD I/O breakdown, DASD space consumption, etc.
- Some data currently only seen in Collection Services: "real" user name, seize time, breakdown of what types of waits (all waits) that occurred.
- Some data not available anywhere else in real time: details on the current wait (duration, object, conflicting job info, specific LIC block point id), 50 deep invocation stack including LIC stack frames.
- SQL statements, host variables, communications data, activation group statistics

Job Watcher is available for trial evaluation or purchase via this website. A license for Job Watcher includes:

- Job Watcher software (licensed by system serial number via an access code)
- Electronic defect support for Job Watcher software for the term of the contract
- No charge updates to Job Watcher software for the term of the contract

The IBM Redbook for Job Watcher provides many examples for the use of Job Watcher. This Redbook is available through the following link: Job Watcher Redbook

Table of Contents | Previous | Next

1.2 PEX Analyzer

The iDoctor for iSeries PEX Analyzer Service includes a software tool specifically geared towards pinpointing issues affecting system and application performance. The detailed analysis it provides picks up where the PM/400 and Performance Tools products leave off and supplies a drill down capability offering a low-level summary of disk operations, CPU utilization, file opens, MI programs, wait states,
DASD space consumption and much more. The client component allows a user to condense and graph iSeries PEX trace, statistical and profile data.

PEX Analyzer is available for trial evaluation or purchase via this website. A license for PEX Analyzer includes:

- PEX Analyzer software (licensed by system serial number via an access code)
- Electronic defect support for PEX Analyzer software for the term of the contract
- Installation assistance for PEX Analyzer software
- No charge updates to PEX Analyzer software for the term of the contract

Special Note for PEX Analyzer: Any PEX data collected prior to IBM OS/400® V5R1M0 cannot be processed using PEX Analyzer.

Additional useful information about PEX is available in the iSeries Performance Explorer Tips and Techniques manual, SG24-4781-00. To view an abstract or to order a copy of the manual, refer to the following link below:
Performance Explorer Tips and Techniques Redbook

Table of Contents | Previous | Next

1.3 Heap Analysis Tools for Java

Heap Analysis Tools for Java (also known as Heap Analyzer) is used to perform Java application heap analysis and object create profiling (size and identification) over time. Heap Analyzer includes information about:

- JVM heap growth/size
- The objects being created (type of object, count and object size, object heap size)
- The application "Heap Footprint" for memory sizing and performance considerations
- Includes a call stack for every snapshot when running in profile mode so objects created can be correlated to functions in the application.
- Includes the ability to search the JVM for a particular object to determine where it was created.

Heap Analyzer requires an iSeries running i5/OS V5R3 or higher with the required PTFs. The PTF information is listed on the website.

Table of Contents | Previous | Next

1.4 PTDV

PTDV is a tool for processing, analyzing, and viewing Performance Explorer Collection data residing in PEX database files. It runs with Java, and uses a different GUI than the other iDoctor components.

PTDV provides table and tree views of Performance Explorer Collection data. It summarizes data on various levels, providing many views and various levels of detail, depending on your path through the data. Tables consist of columns that are sortable and rearranageble; trees provide a "drill down" capability to show parent/child relationship for data where appropriate. Each pane of data can be exported to another format for viewing by other tools, such as spreadsheets. More detail can be displayed as the user determines performance problem areas.

PTDV can process collections containing program and/or Java events. Collections containing program events will be processed so that procedure information can be determined, such as inline and cumulative time and cycles. The call flow is reconstructed and displayed in a tree format. If Java events are also present, information from those events are summarized and displayed, and the method context information is available. For example, if object create or object lock events are present then a table displaying the class name, number of objects, total sizes, wait times, are all displayed.

Disclaimer for PTDV

This material is provided for your consideration; it has not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of this material. IBM provides no program services for this material.

THIS MATERIAL IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT. SOME JURISDICTIONS DO NOT ALLOW THE EXCLUSION OF IMPLIED WARRANTIES, SO THE ABOVE EXCLUSION MAY NOT APPLY TO YOU. IN NO EVENT WILL IBM BE LIABLE TO ANY PARTY FOR ANY DIRECT, INDIRECT, SPECIAL OR OTHER CONSEQUENTIAL DAMAGES FOR ANY USE OF THIS MATERIAL INCLUDING, WITHOUT LIMITATION, ANY LOST PROFITS, BUSINESS INTERRUPTION, LOSS OF PROGRAMS OR OTHER DATA ON YOUR INFORMATION HANDLING SYSTEM OR OTHERWISE, EVEN IF WE ARE EXPRESSLY ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Table of Contents | Previous | Next

1.5 Other Components

There are some components that become visible if a file is installed on the PC provided by IBM support. Contact idoctor@us.ibm.com if you need access to these "as-is" components.
One of the components provides views of Collection Services data.

Table of Contents | Previous | Next

Chapter 2 Installation

This chapter includes information about the following:

- iSeries installation requirements
- PC installation requirements
- PTF prerequisites
- Installing and uninstalling iDoctor for iSeries

Table of Contents | Previous | Next

2.1 Installing Job Watcher

PC Requirements:

- Windows NT 4.0, 2000, XP
- IBM iSeries Access for Windows V5R3 or higher
- 512 MB of RAM or higher recommended
- 800 Mhz processor or higher

Server Requirements:

- IBM i5/OS V5R3 or higher
- The Required PTFs
- The user profile performing the installation must have *SECOFR user class and special authorities *ALLOBJ and *SECADM.
- The following host servers (identified by the SERVER parameter values on the STRHOSTSVR command) need to be running on the server: *DATABASE, *RMTCMD, *SIGNON, *SRVMAP
- System value QALWOBJRST must be *ALL or (*ALWSYSSTT and *ALWPGMADP)
- **Note:** If English is not installed as the primary language, the user profiles used to connect to the server with should set their CCSID parameter value to 37.

After installation you will have the following new libraries on your server: QIDRGUI and QIDRWCH

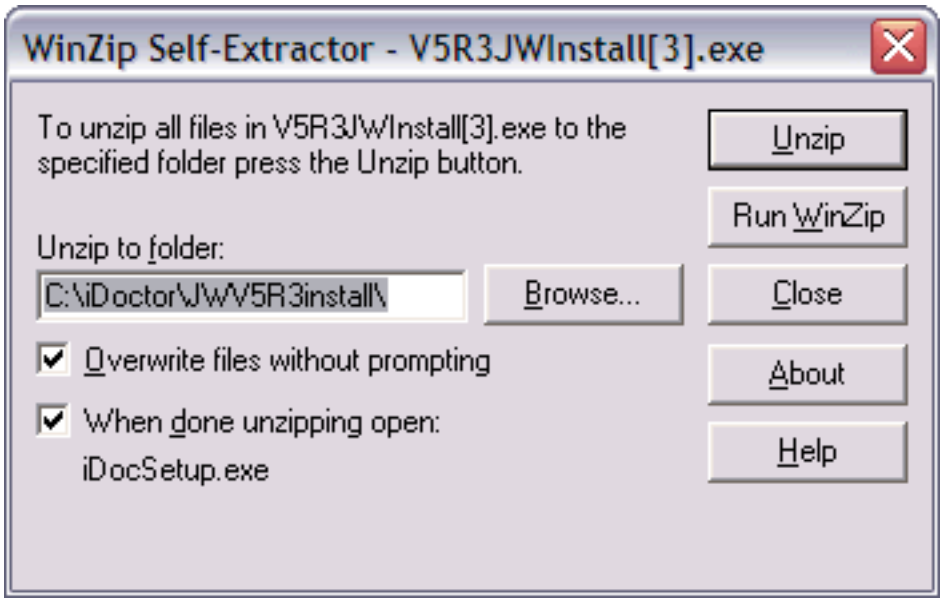
The Install Process:

The GUI install process will install the server objects and programs on your iSeries and the client code on your PC. The PTFs required for Job Watcher must be manually installed via whichever method normally used to get PTFs for your iSeries. The PTFs required are listed on the download pages of this website. If unsure of which method to use for installing the PTFs, use the Fix Central website. Instructions for installing these PTFs can be found in the next section.

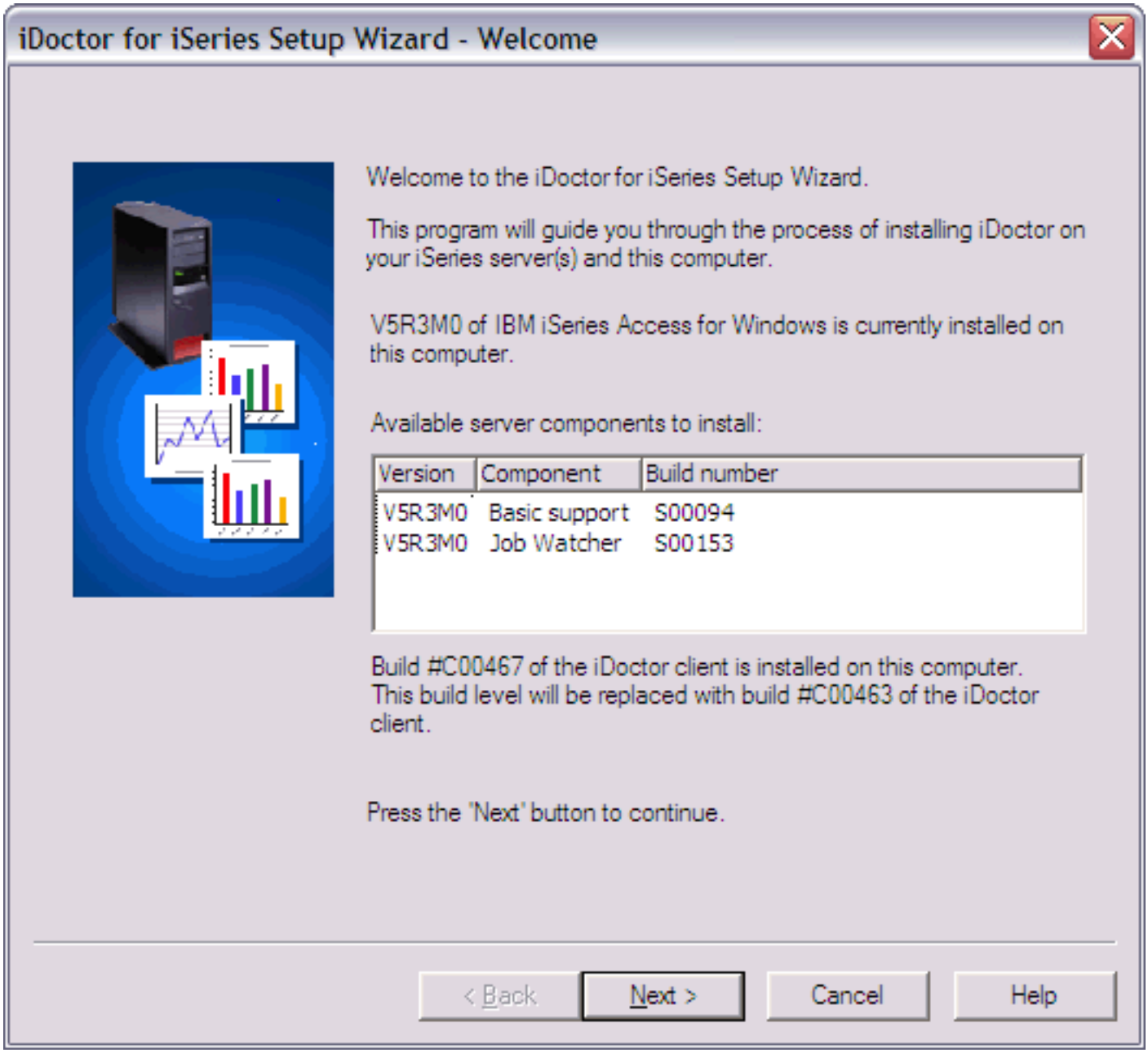
After the PTFs are loaded and applied on your iSeries, perform the following steps:

Step 1 Download the install image for Job Watcher from our website to your PC.

Step 2 Double-click on the install image (it is a self-extracting .exe) from within Windows Explorer. You will see the following:

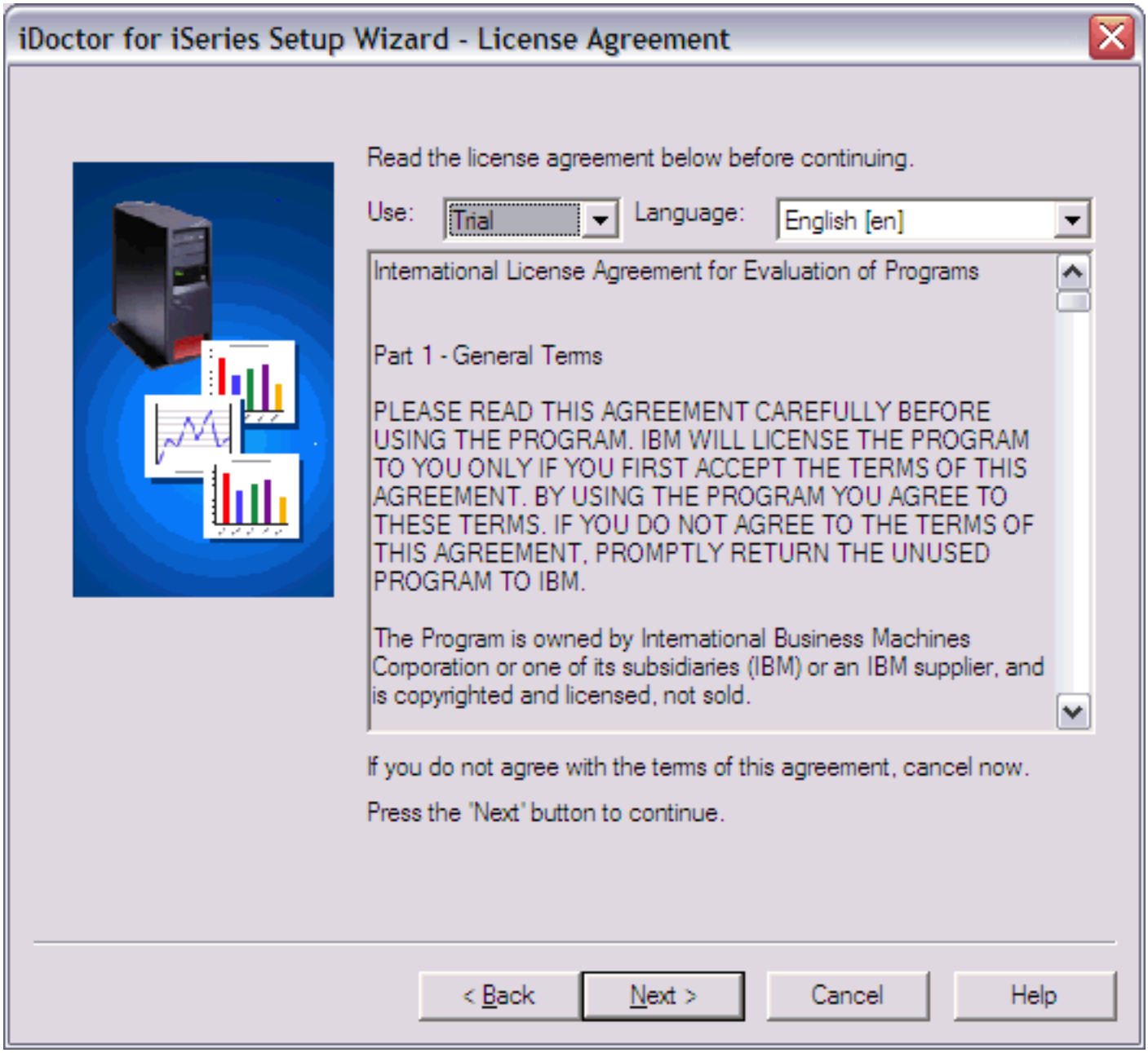


Step 3 Change the path where the install image will be extracted to on the PC if desired and click the 'Unzip' button. Wait a moment for the files to be extracted and the setup program to be launched.

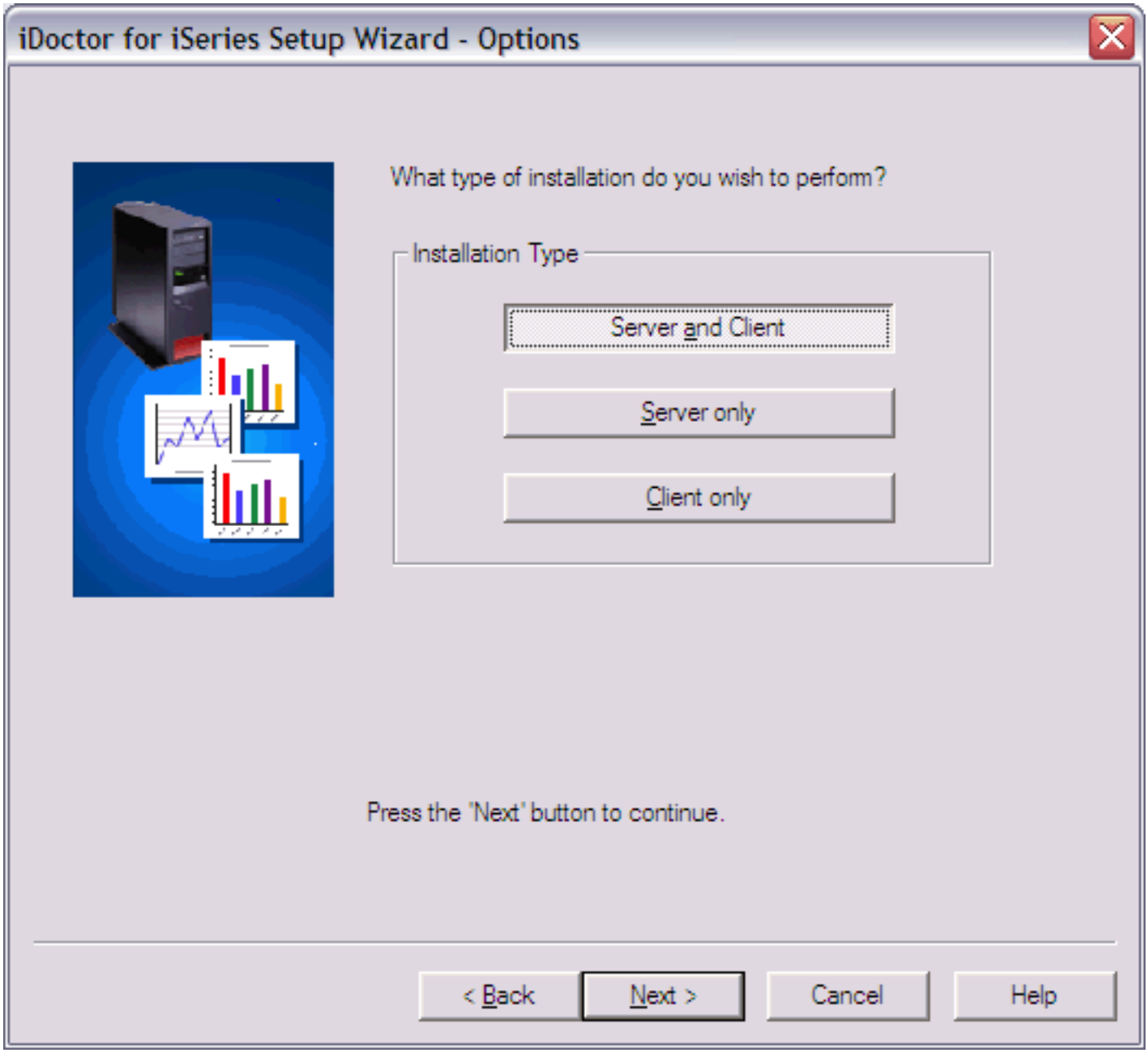


This screen identifies the version of IBM iSeries Access for Windows installed as well as the version of iDoctor for iSeries client installed (if found). Click 'Next'.

Step 4 On the next screen, click the 'Next' button to indicate acceptance of the license agreement. Use the drop down lists at the top of this window to view the license agreement in another language.

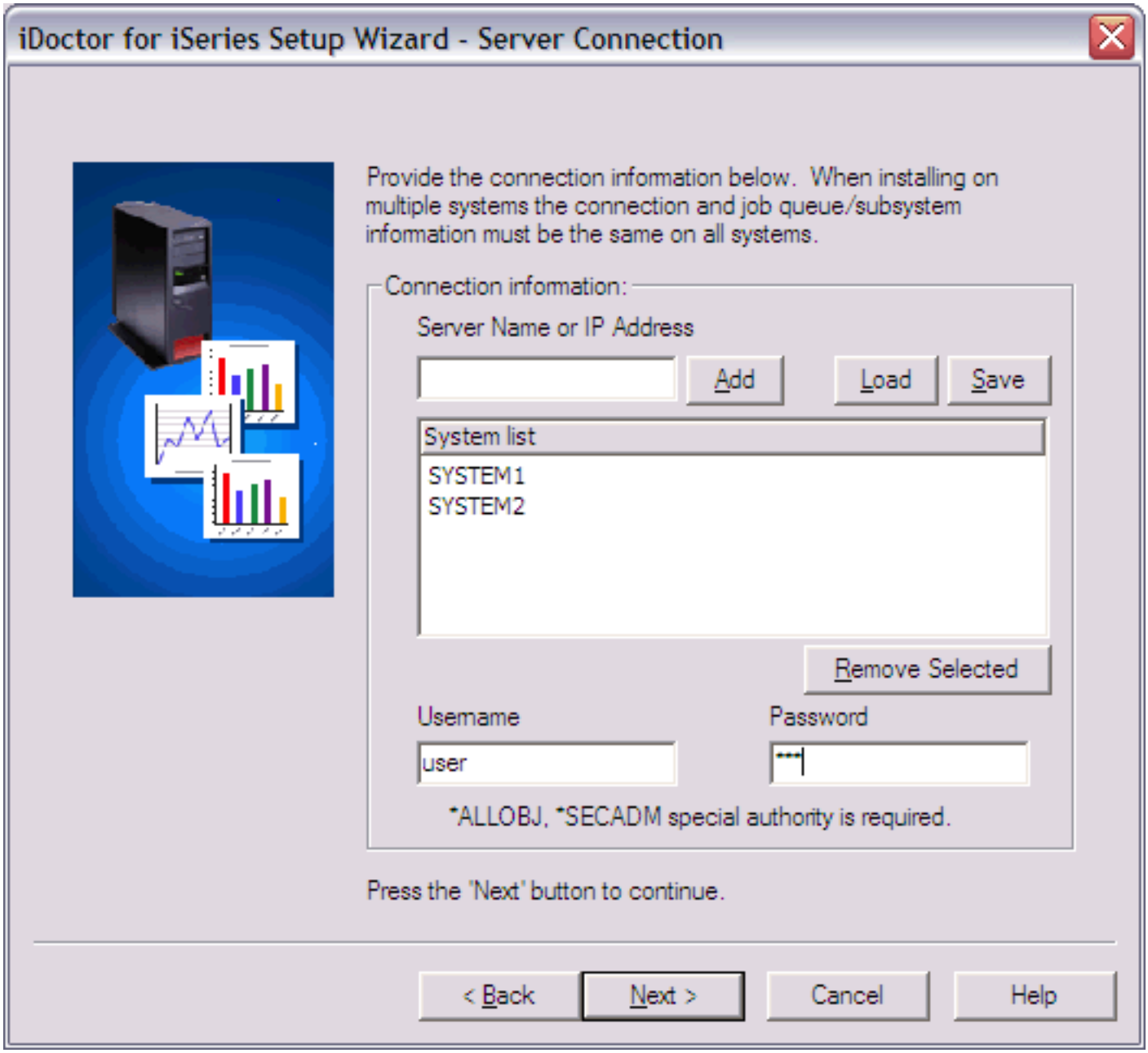


Step 5 Select the type of installation to perform. This screen allows you to choose whether to install the server side of Job Watcher, the client side of iDoctor or both.



Step 6 If you are installing the server portion of Job Watcher you will see a screen asking for the connection information to use to access the server(s). The user profile must have the user class authority of *SECOFR and *ALLOBJ, *SECADM special authorities. If installing on multiple systems the user profile and password must be the same on all systems specified in the system list.

If desired you may use the Save and Load buttons to save a system list to a text file or load one into this interface that was previously saved. This file is a simple list of system names or IP addresses with each entry on a separate line.



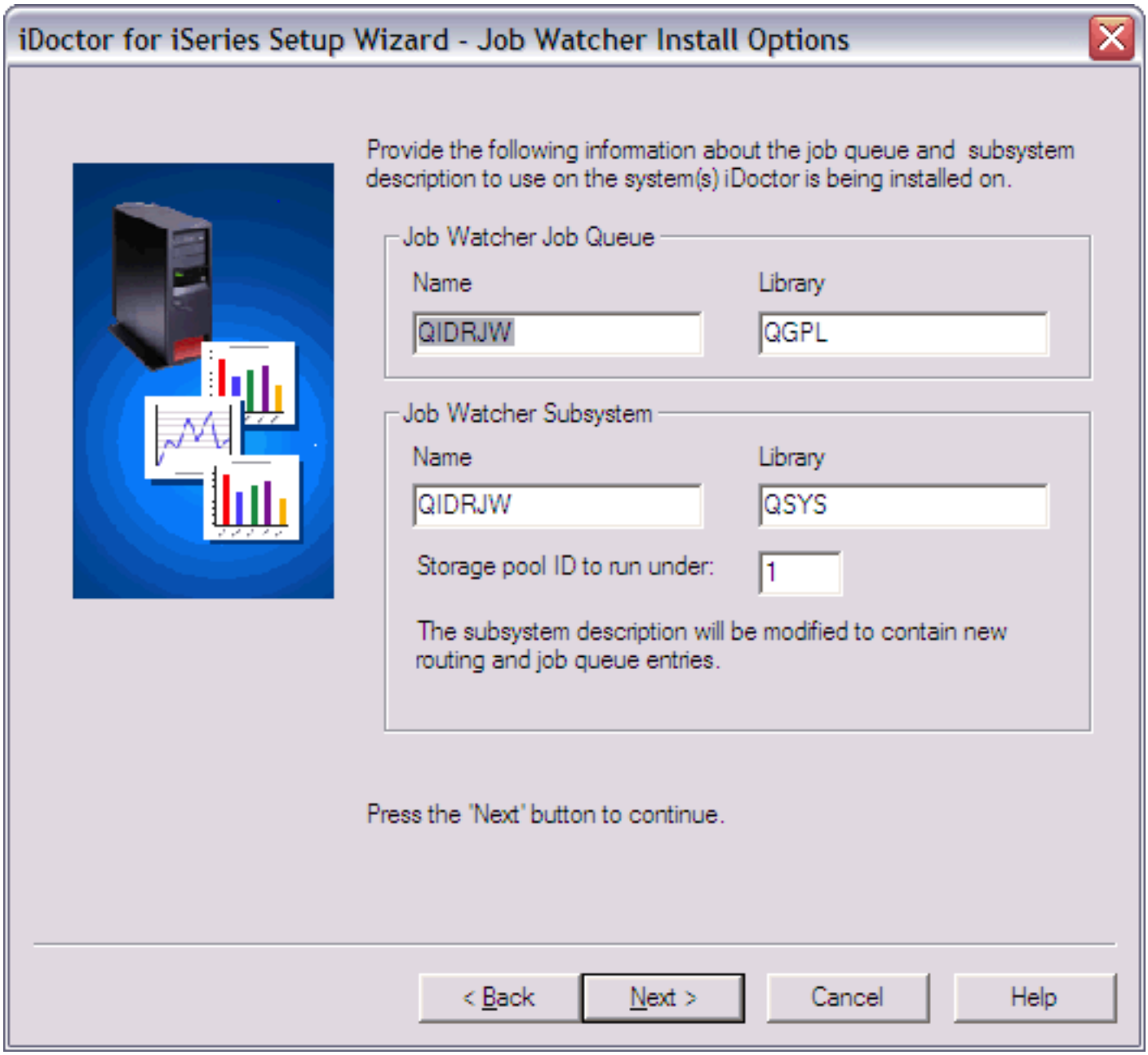
Click the 'Next' button to continue. Click 'Next' again on the Component Selection screen, using the default options.

Step 7 If you are installing the server portion of Job Watcher you will also be asked to specify a job queue name and a subsystem description the server code may use when running Job Watcher. If the subsystem does not already exist, the install program will ask if it should be created. This subsystem and job queue will be used to run the Job Watcher "watch" jobs named QWCHJOB. Job Watcher does not have a limit to the number of "watches" that can be active at one time and the max active parameter value will be set accordingly.

You must also indicate the storage pool ID that the Job Watcher jobs should run under. This parameter is required in order to add routing entries to the subsystem description.

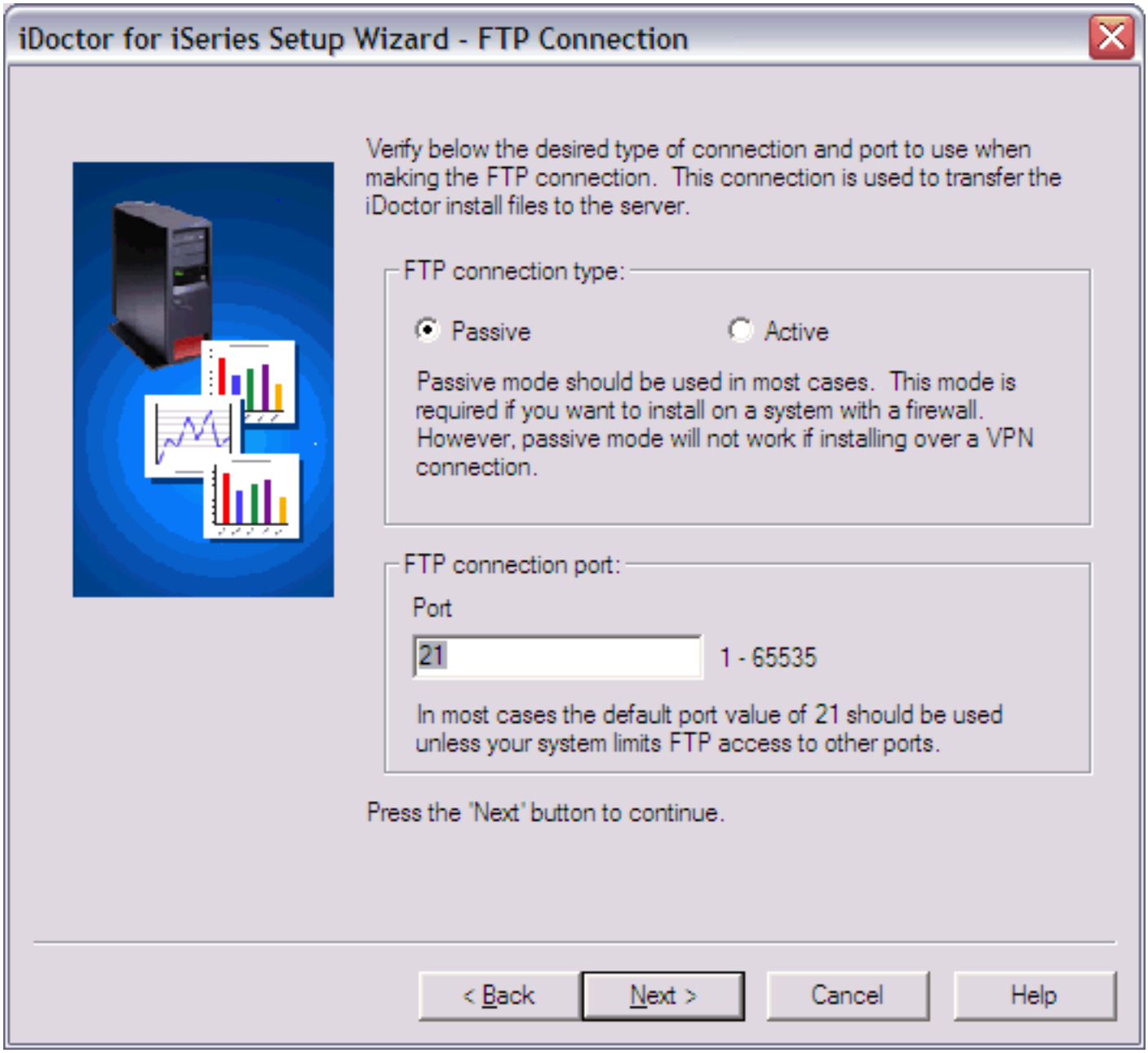
If installing on multiple systems these same values specified will be used for all systems.

Press the 'Next' button to continue.

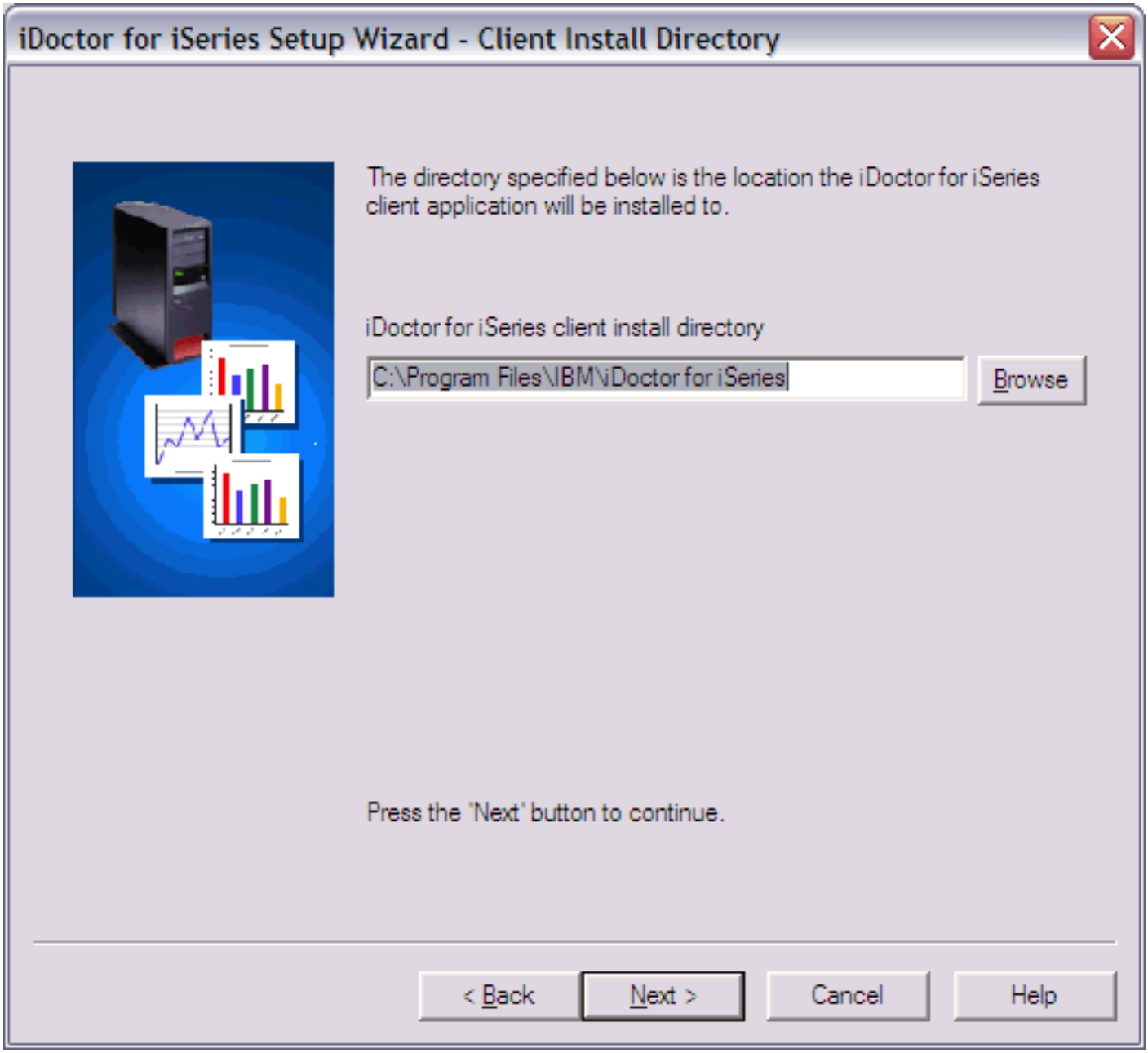


Step 8 The next page gives you the option to specify which type of FTP connection should be used when performing the install. Only in unusual circumstances should anything other than the defaults be used on this page. However, if installing over a VPN connection and "Passive" FTP does not work, try using "Active" FTP instead.

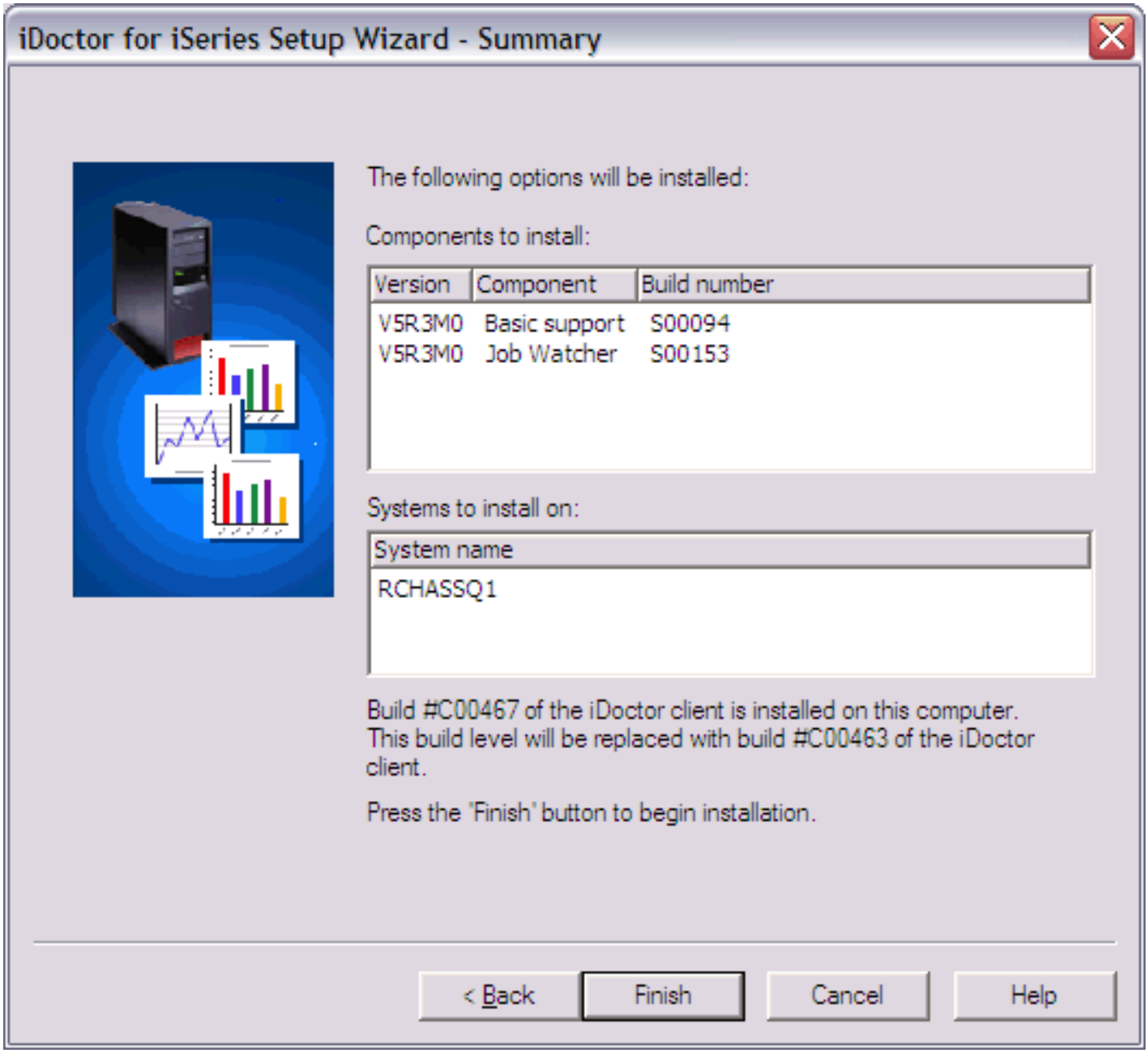
Clicking 'Next' on this screen will verify that the FTP connection is working between the PC and the server specified (or the 1st server specified if installing on multiple systems).



Step 9 This screen allows you to specify the directory the iDoctor client program should be installed to.



Step 10 A summary of your selections appears on the final screen.



Step 11 Clicking the 'Finish' button will copy all of the files and run the commands necessary to install the server and/or client portion of Job Watcher. The server portion of the installation may take a few minutes.

After the install completes the setup logfile will be viewable if needed through the install program. If any errors occur, send the logfile to idoctor@us.ibm.com for assistance.

Table of Contents | Previous | Next

2.1.1 Manual installation steps for remote service

Purpose:
This section describes the steps to install Job Watcher on an iSeries running i5/OS V5R3. Using the GUI install wizard is preferred in most instances but if this is not possible, these steps are the only other way to get Job Watcher on the desired system.

Self-Installation Prep:
This document assumes that the user has downloaded the Job Watcher install image from the V5R3 downloads page.. This self extracting ZIP file is named V5R3JWInstall.exe and contains the following files which will be used in this document:

baseV5R3.savf (save file for QIDRGUI library)
jobV5R3.savf (save file for QIDRWCH library)

These files should be extracted to a directory on the PC so they may be sent to the iSeries via FTP.
The user profile performing the installation must have *SECADM, *ALLOBJ special authorities. Otherwise all objects may not be restored properly
The system value QALWOBJRST must be set to *ALL or certain objects will not be installed properly. Temporarily change this system value if necessary in order to perform this install. Remember to change it back to what it was when you started after the manual install has completed.

Installation Steps

Perform the following steps in order to manually install Job Watcher.

1. From a green screen window, run the following commands:
- DLTLIB IDOCINST
- DLTLIB QIDRGUI
- DLTLIB QIDRWCH
- DLTLIB IDOCSBA

- CRTLIB IDOCINST
- CRTSAVF FILE(IDOCINST/IDOCINST)

2. Open a DOS FTP session to the iSeries.

3. FTP file baseV5R3.savf from the PC to the iSeries save file IDOCINST in library IDOCINST.

4. From a green screen window, run the following commands:
- DSPSAVF FILE(IDOCINST/IDOCINST)
- verify that the save file contains data
- RSTLIB IDOCSBA *SAVF SAVF(IDOCINST/IDOCINST) MBROPT(*ALL) ALWOBJDIF(*ALL)
- IDOCSBA/INSTIDOCBA
- verify that the message "CPF9898 - SUCCESSFUL iDoctor server Base install." is returned by the command.
- CLRSAVF FILE(IDOCINST/IDOCINST)

5. FTP file jobV5R3.savf from the PC to the iSeries save file IDOCINST in library IDOCINST.

6. From a green screen window, run the following commands:
- DLTLIB IDOCSJW
- DSPSAVF FILE(IDOCINST/IDOCINST)
- verify that the save file contains data
- RSTLIB IDOCSJW *SAVF SAVF(IDOCINST/IDOCINST) MBROPT(*ALL) ALWOBJDIF(*ALL)
- IDOCSJW/INSTIDOCJW JOBQ(QGPL/QIDOCJW) CRTJOBQ(*NO) SBSD(QSYS/QIDOCJW) POOLID(1)
- the jobq and sbsd parms are the job queue and subsystem that job watches should run under when submitted from the client
- verify that the message "CPF9898 - SUCCESSFUL iDoctor server Job Watcher install." is returned by the command.

7. From a green screen window, run the following commands:
- DLTLIB IDOCINST
- DLTLIB IDOCSBA
- DLTLIB IDOCSJW

8. The required PTFs listed on the V5R3 downloads page must be loaded and applied on the system before Job Watcher will run correctly.

If you are unable to complete the manual installation successfully contact idoctor@us.ibm.com for assistance.

Table of Contents | Previous | Next

2.2 Installing PEX Analyzer

PC Requirements:

- Windows NT 4.0, 2000, XP
- IBM iSeries Access for Windows V5R3 or higher
- 512 MB of RAM or higher recommended
- 800 Mhz processor or higher

Server Requirements:

- Performance Tools LPP (PT1)
- IBM i5/OS V5R3
- The Required PTFs
- The user profile performing the installation must have *SECOFR user class and special authorities *ALLOBJ and *SECADM.
- The following host servers (identified by the SERVER parameter values on the STRHOSTSVR command) need to be running on the server: *DATABASE, *DTAQ, *RMTCMD, *SIGNON, *SRVMAP
- System value QALWOBJRST must be *ALL or (*ALWSYSSTT and *ALWPGMADP)
- **Note:** If English is not installed as the primary language, the user profiles used to connect to the server with should set their CCSID parameter value to 37.

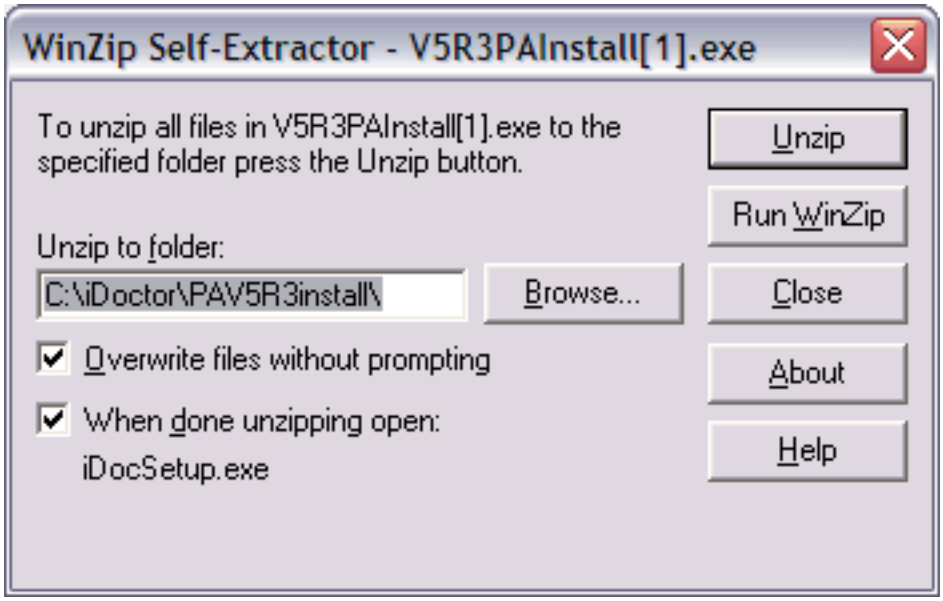
After installation you will have the following new libraries on your server: QIDRGUI and QIDRPA.

The Install Process:

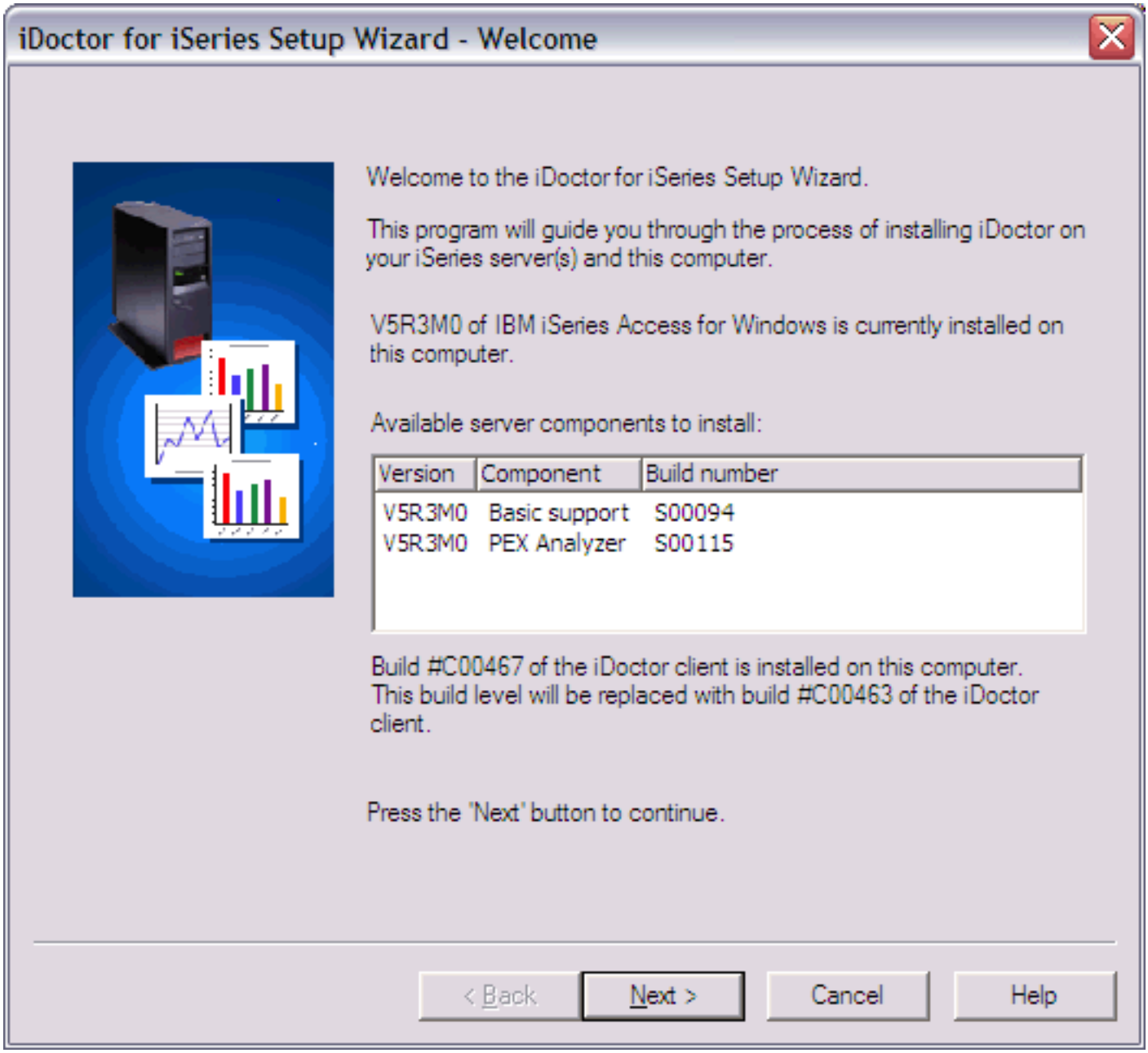
The iDoctor for iSeries install image contains a setup program which fully automates the process of installing PEX Analyzer.

Step 1 Download the install image for PEX Analyzer from our website to your PC.

Step 2 Double-click on the install image (it is a self-extracting .exe.) from within Windows Explorer. You will see the following:

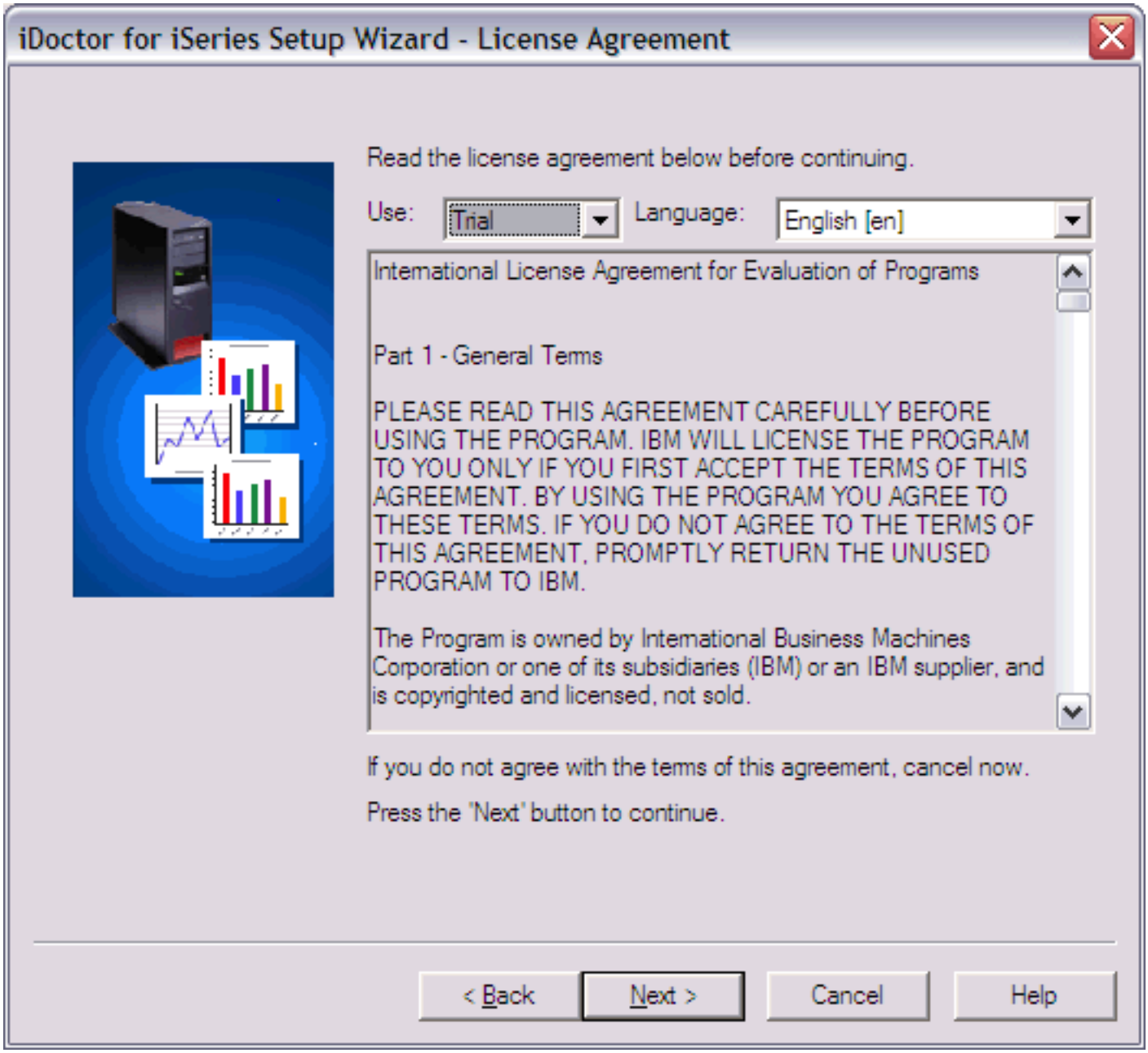


Step 3 Change the path where the install image will be extracted to on the PC if desired and click the 'Unzip' button. Wait a moment for the files to be extracted and the setup program to be launched.

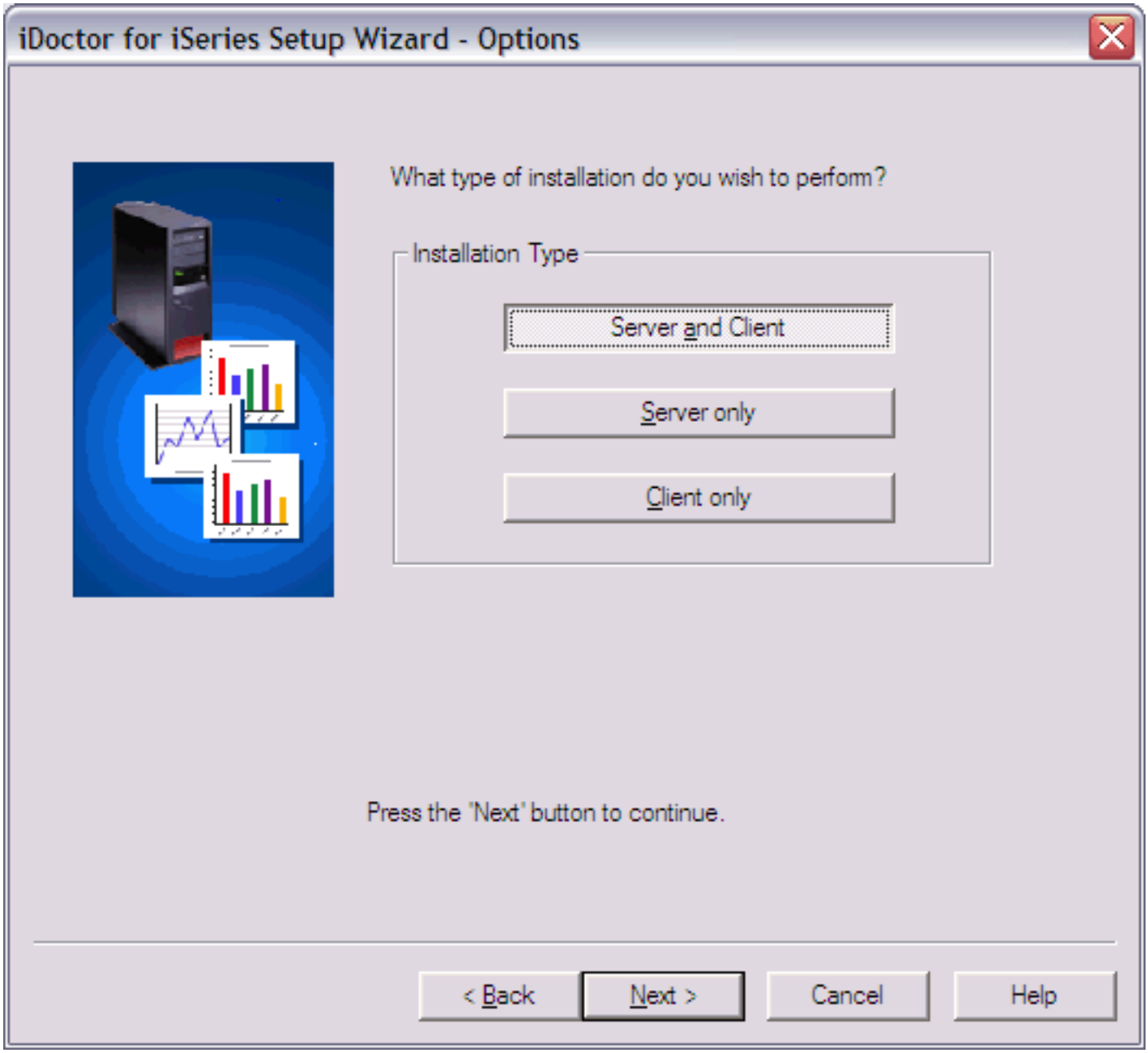


This screen identifies the version of IBM iSeries Access for Windows installed as well as the version of iDoctor for iSeries installed on the client (if any).

Step 4 On the next screen, click the 'Next' button to indicate acceptance of the license agreement. Use the drop down lists at the top of this window to view the license agreement in another language.

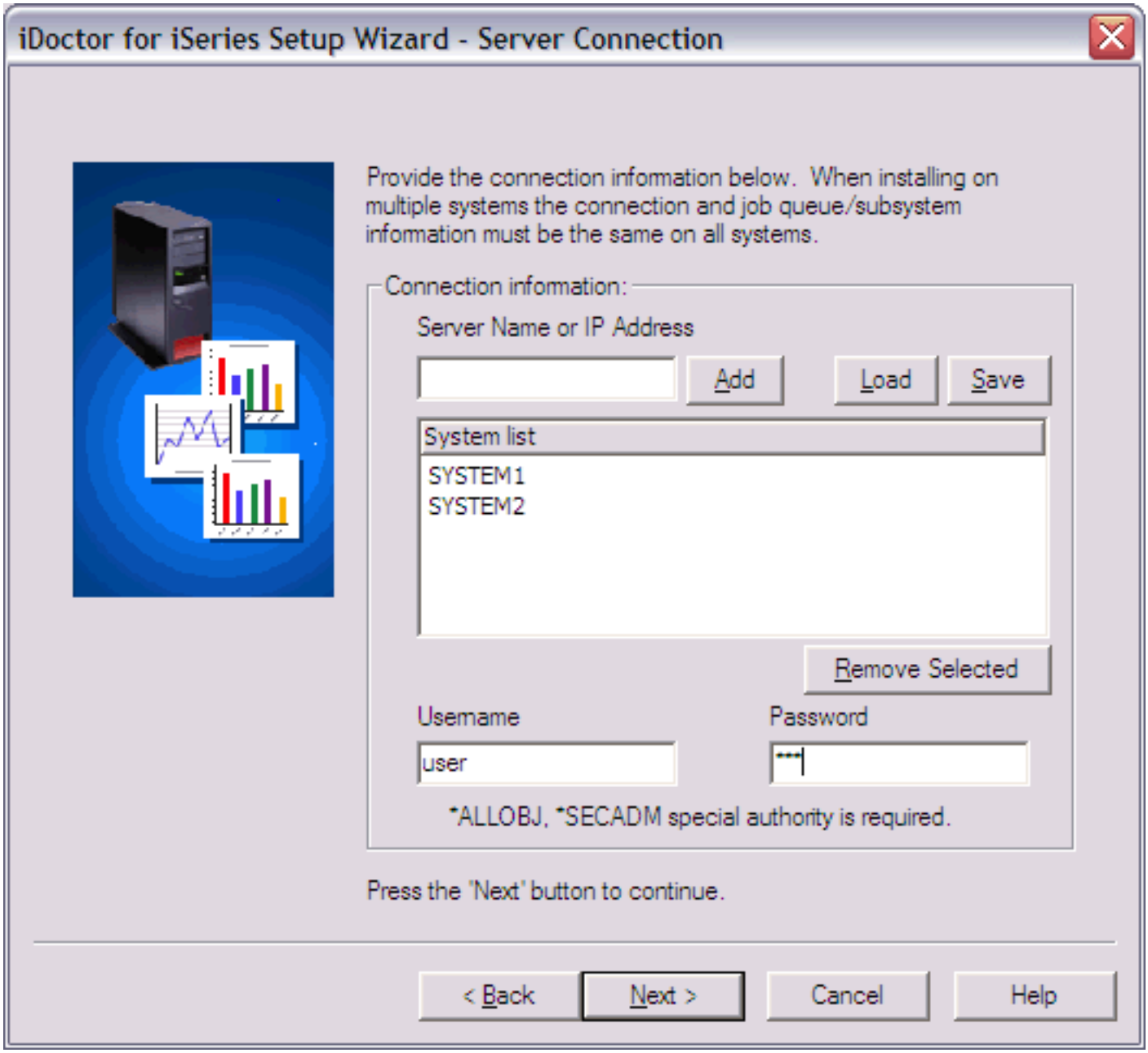


Step 5 Select the type of installation to perform and click 'Next'.



Step 6 If you are installing the server portion of PEX Analyzer you will see a screen asking for the connection information to use to access the server(s). The user profile must have the user class authority of *SECOFR and *ALLOBJ, *SECADM special authorities. If installing on multiple systems the user profile and password must be the same on all systems specified in the system list.

If desired you may use the Save and Load buttons to save a system list to a text file or load one into this interface that was previously saved. This file is a simple list of system names or IP addresses with each entry on a separate line.

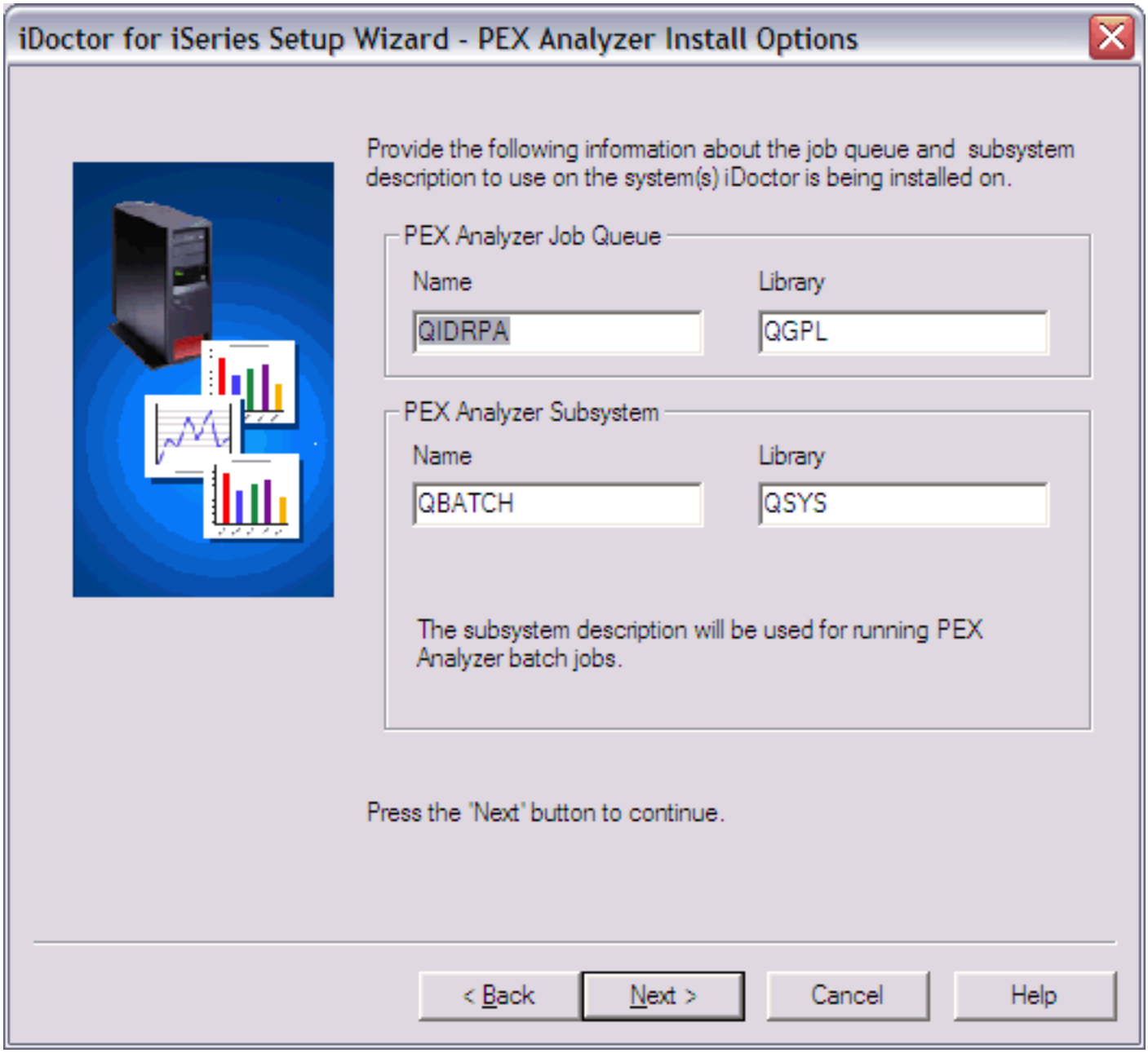


Click the 'Next' button to continue. Click 'Next' again on the Component Selection screen, using the default options.

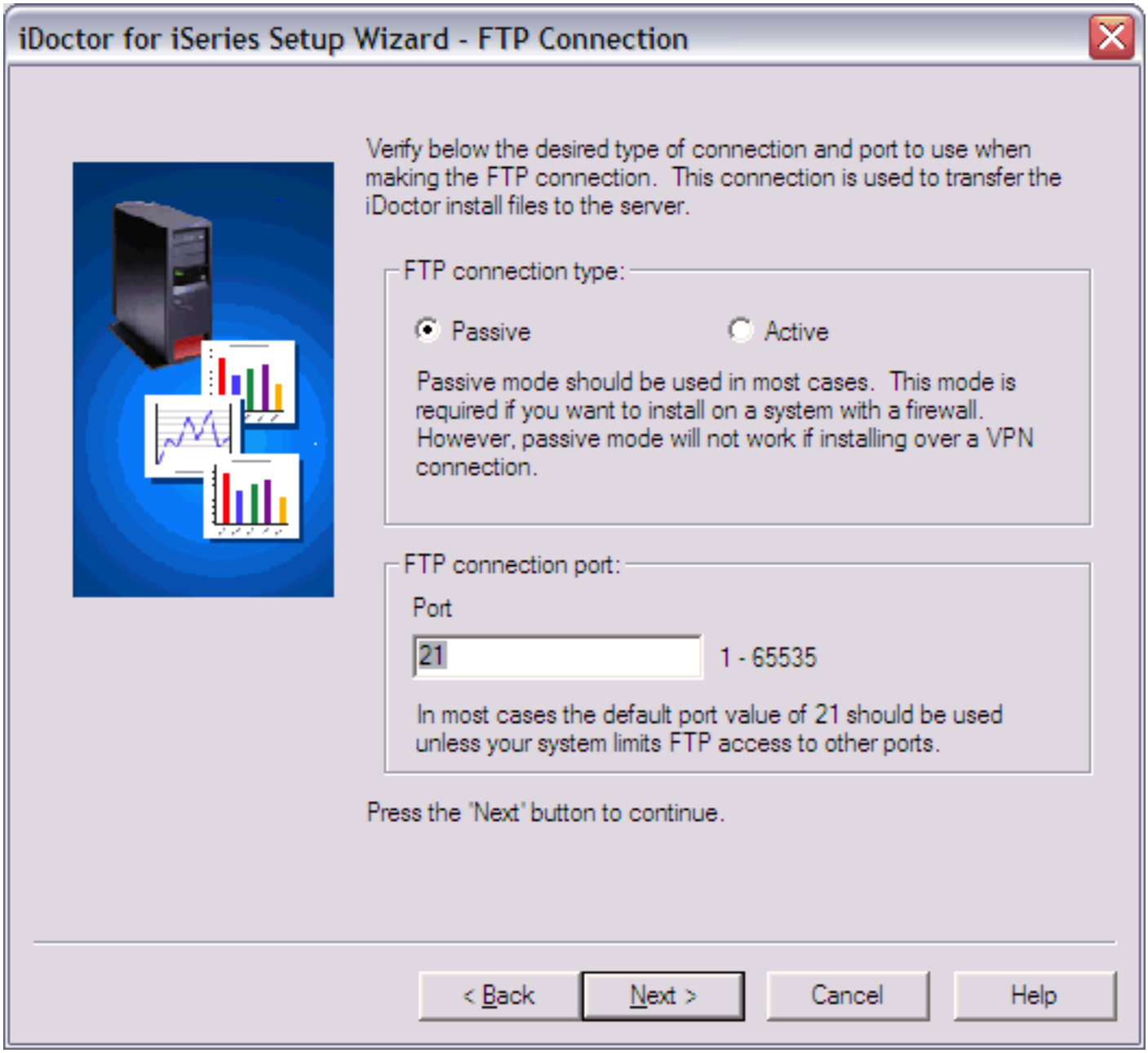
Step 7 If you are installing the server portion of PEX Analyzer you will also be asked to specify a job queue name and a subsystem description the server code may use. This subsystem is used to run the PEX Analyzer analysis jobs. Only one analysis job can be active on the job queue at a time.

If installing on multiple systems these same values specified will be used for all systems.

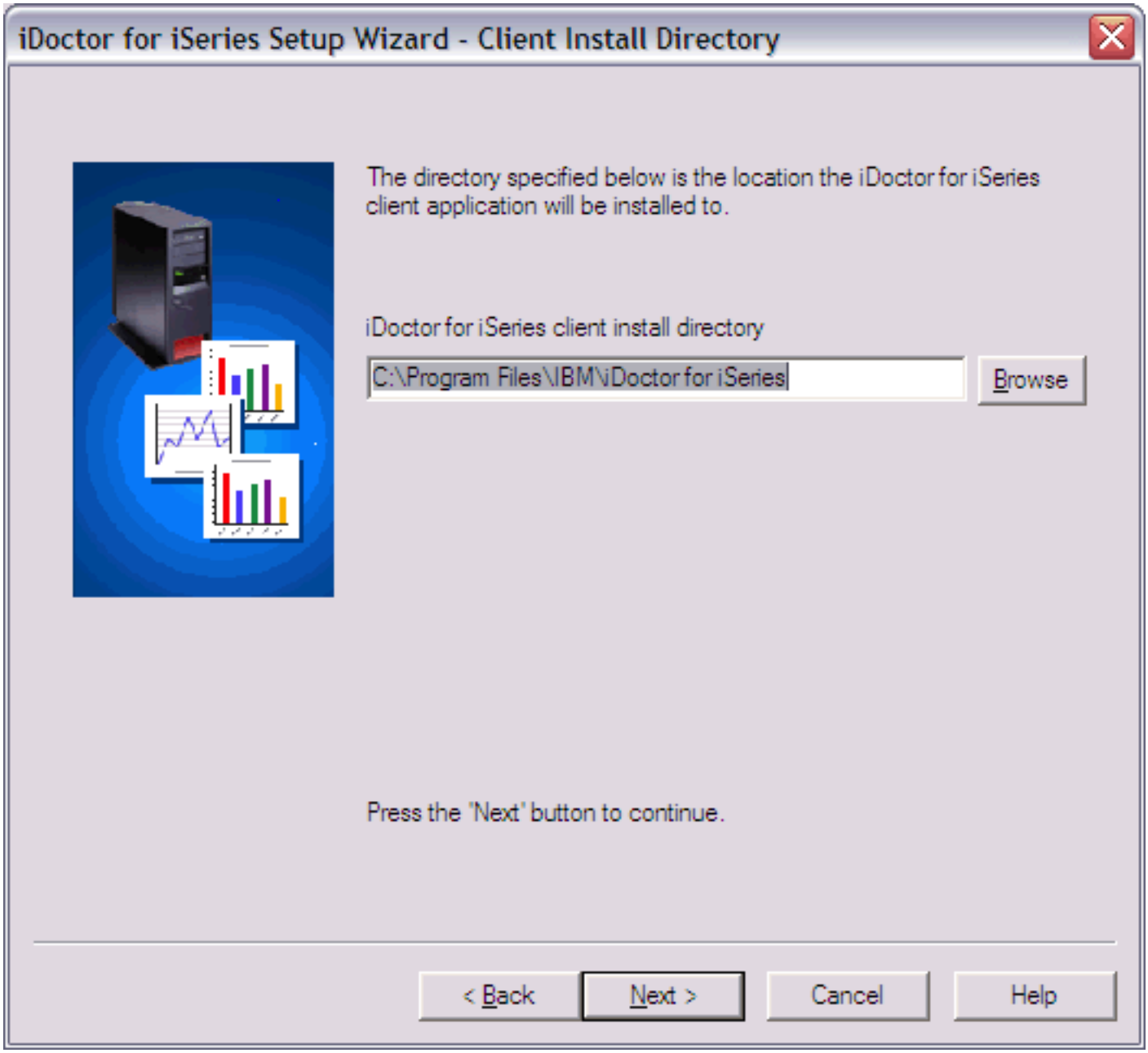
Caution for PEX Analyzer installs: There is nothing that prevents this job queue from being modified to allow more than one active job. However, if that is done, PEX Analyzer will fail to function properly if multiple analyses are ran at the same time and your server may require an IPL. In addition, it may be necessary to delete and restore the libraries containing PEX data.



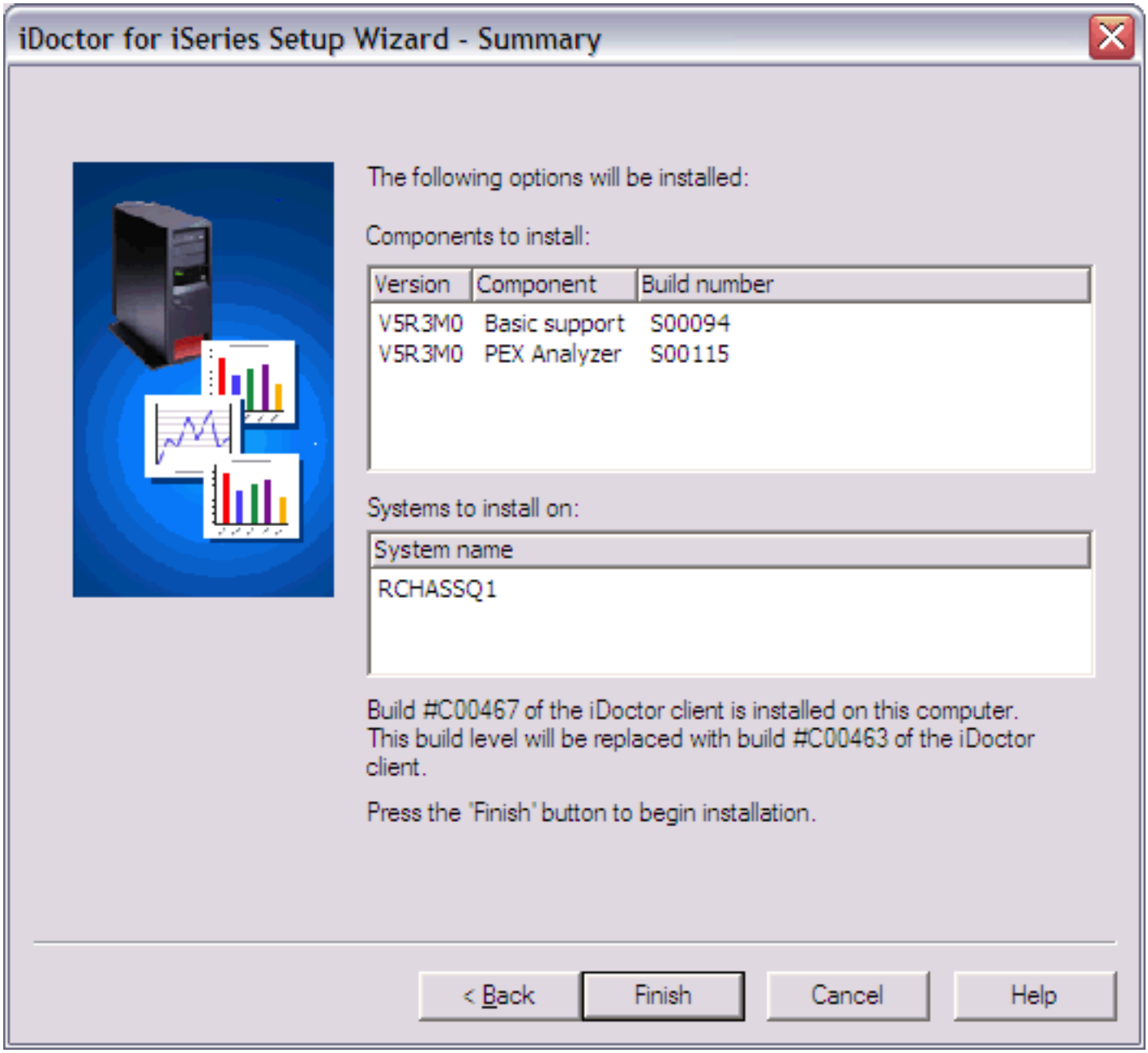
Step 8 The next page gives the user the option to specify which type of FTP connection should be used when performing the install. Only in unusual circumstances should anything other than the defaults be used on this page. However, if installing over a VPN connection and "Passive" FTP does not work, try using "Active" FTP instead.



Step 9 This screen allows you to specify the directory the iDoctor client program should be installed to.



Step 10 A summary of your selections appears on the final screen. This page also contains options for whether or not the temporary libraries used during the installation should be deleted. During the install process these libraries will contain the restored contents of the save files sent to the server. Only if you were having problems with the install would the option to keep these libraries around be useful so they could be used for service.



Step 11 Clicking the 'Finish' button will copy all of the files and run the commands necessary to install the server and/or client portion of PEX Analyzer. The server portion of the installation may take a few minutes.

After the install completes the setup logfile will be viewable if needed through the install program. If any errors occur, send the logfile to idoctor@us.ibm.com for assistance.

Table of Contents | Previous | Next

2.2.1 PTF Prerequisites

Collecting PDC/PEX data on a machine involves running low-level LIC code that runs only when making such a collection. The machine making the collection must be loaded with the latest PTFs for PDC and PEX.

You should periodically use the link below to identify the **Performance Tools for i5/OS Group PTF** for your i5/OS VRM. This Group PTF lists PTFs for performance analysis related tools (including PEX and PDC) in addition to
PTFs for the Performance Tools LPP.

Once you have found the correct Group PTF, download of the Group PTF and subsequent load/apply of all
- PDC PTFs (LIC)
- PEX PTFs (OS/400 or Performance Tools LPP)
within the Group PTF is highly recommended.

All Group PTFs for all releases

Table of Contents | Previous | Next

2.3 Installing Heap Analyzer for Java

PC Requirements:

- Windows NT 4.0, 2000, XP
- IBM iSeries Access for Windows V5R3 or higher
- 512 MB of RAM or higher recommended
- 800 Mhz processor or higher

Server Requirements:

- IBM i5/OS V5R3
- The Required PTFs
- The user profile performing the installation must have *SECOFR user class and special authorities *ALLOBJ and *SECADM.
- The following host servers (identified by the SERVER parameter values on the STRHOSTSVR command) need to be running on the server: *DATABASE, *RMTCMD, *SIGNON, *SRVMAP
- System value QALWOBJRST must be *ALL or (*ALWSYSSTT and *ALWPGMADP)
- The required Java Group PTF must be installed on the iSeries.
- **Note:** If English is not installed as the primary language, the user profiles used to connect to the server with should set their CCSID parameter value to 37.

After installation you will have the following new libraries on your server: QIDRGUI and QIDRHAJ

The Install Process:

The GUI install process will install the server objects and programs on your iSeries and the client code on your PC. The PTFs required for Heap Analyzer must be manually installed via whichever method normally used to get PTFs for your iSeries. The PTFs required are MF33651, SI15117. If unsure of which method to use for installing the PTFs, use the Fix Central website. Instructions for installing PTFs can be found under under the Job Watcher PTF installation steps.

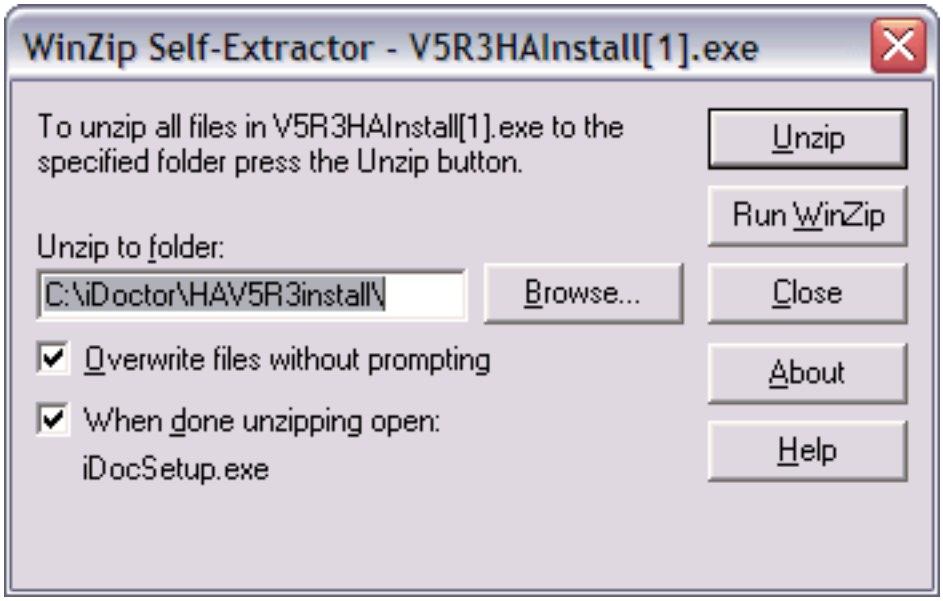
After the PTFs are loaded and applied on your iSeries, perform the following steps:

The iDoctor for iSeries install image contains a setup program which fully automates the process of

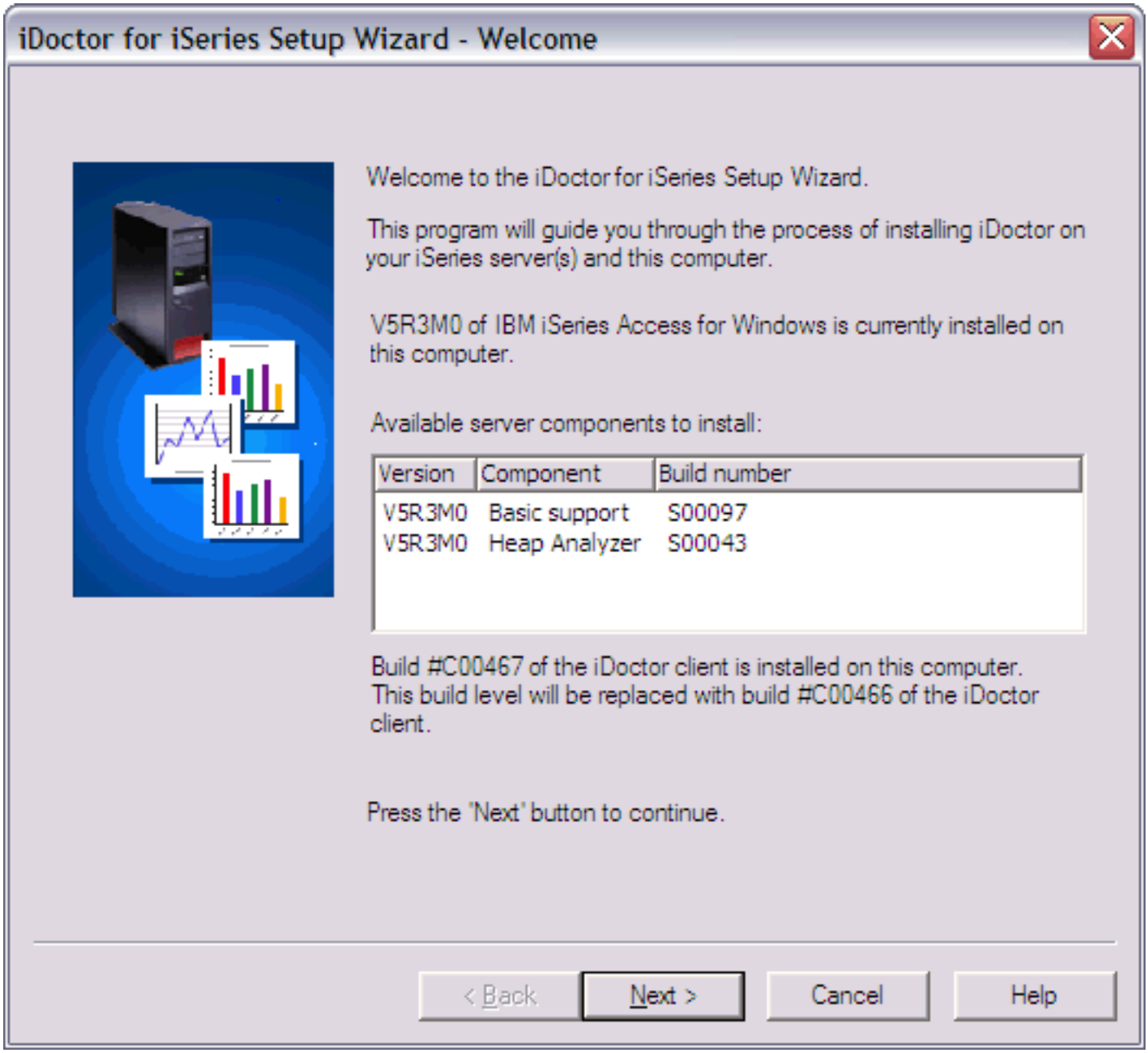
installing Heap Analyzer.

Step 1 Download the install image for Heap Analyzer from our website to your PC.

Step 2 Double-click on the install image (it is a self-extracting .exe.) from within Windows Explorer. You will see the following:



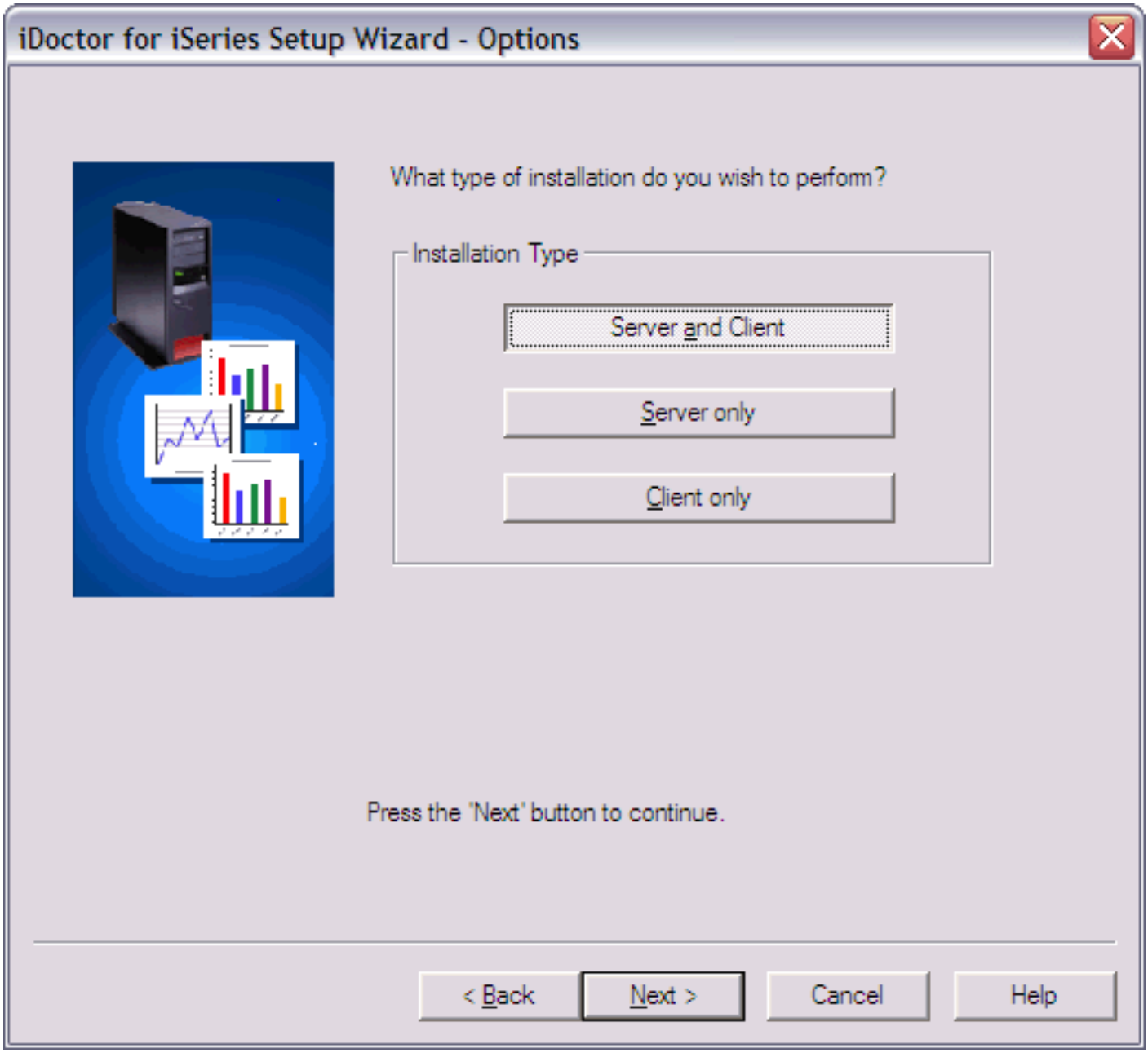
Step 3 Change the path where the install image will be extracted to on the PC if desired and click the 'Unzip' button. Wait a moment for the files to be extracted and the setup program to be launched.



This screen identifies the version of iSeries Access for Windows installed as well as the version of iDoctor for iSeries client installed (if found). Click 'Next'.

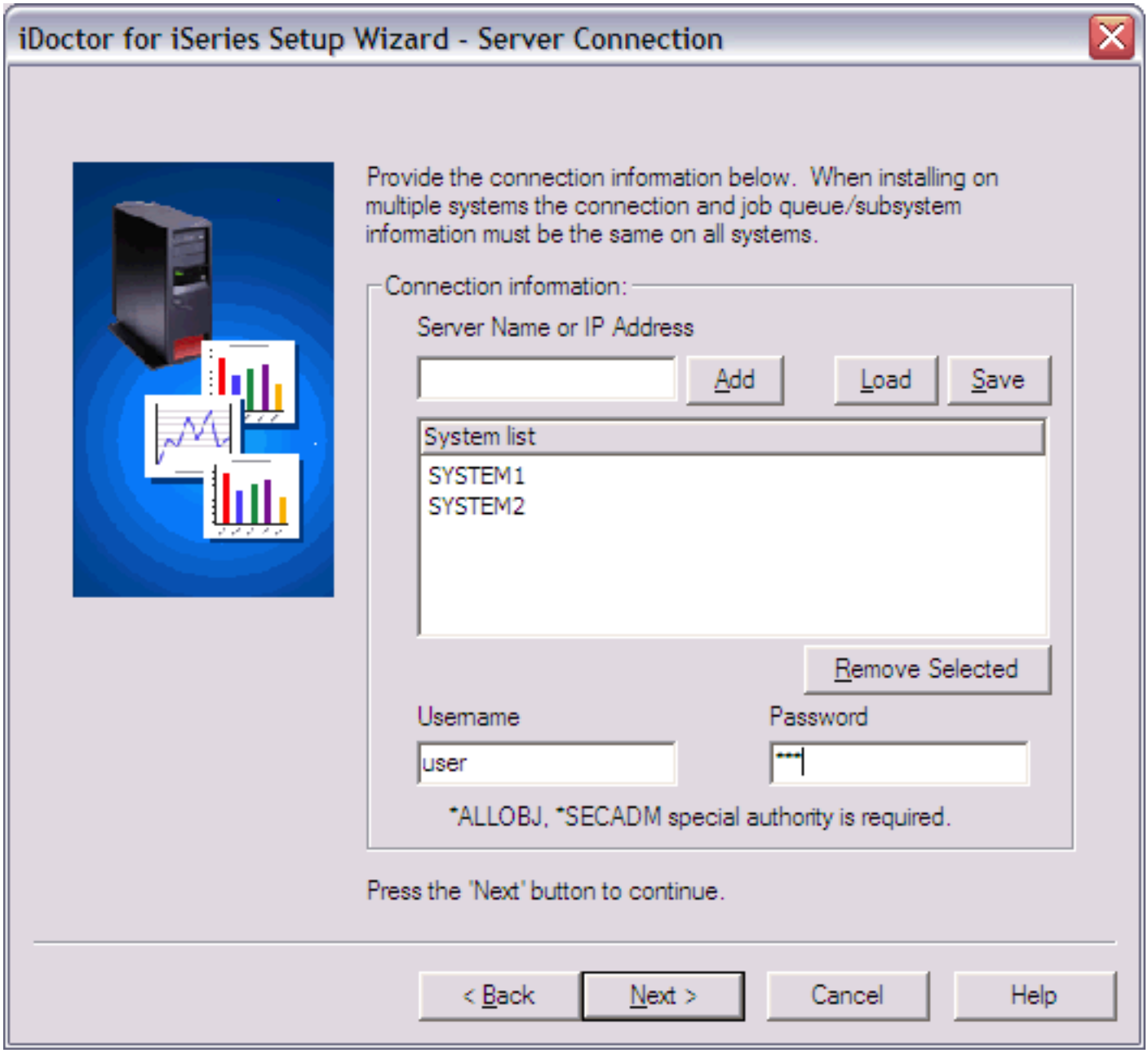
Step 4 On the next screen, click the 'Next' button to indicate acceptance of the license agreement.

Step 5 Select the type of installation to perform. This screen allows you to choose whether to install the server side of Heap Analyzer, the client side of iDoctor or both.



Step 6 If you are installing the server portion of Heap Analyzer you will see a screen asking for the connection information to use to access the server(s). The user profile must have the user class authority of *SECOFR and *ALLOBJ, *SECADM special authorities. If installing on multiple systems the user profile and password must be the same on all systems specified in the system list.

If desired you may use the Save and Load buttons to save a system list to a text file or load one into this interface that was previously saved. This file is a simple list of system names or IP addresses with each entry on a separate line.



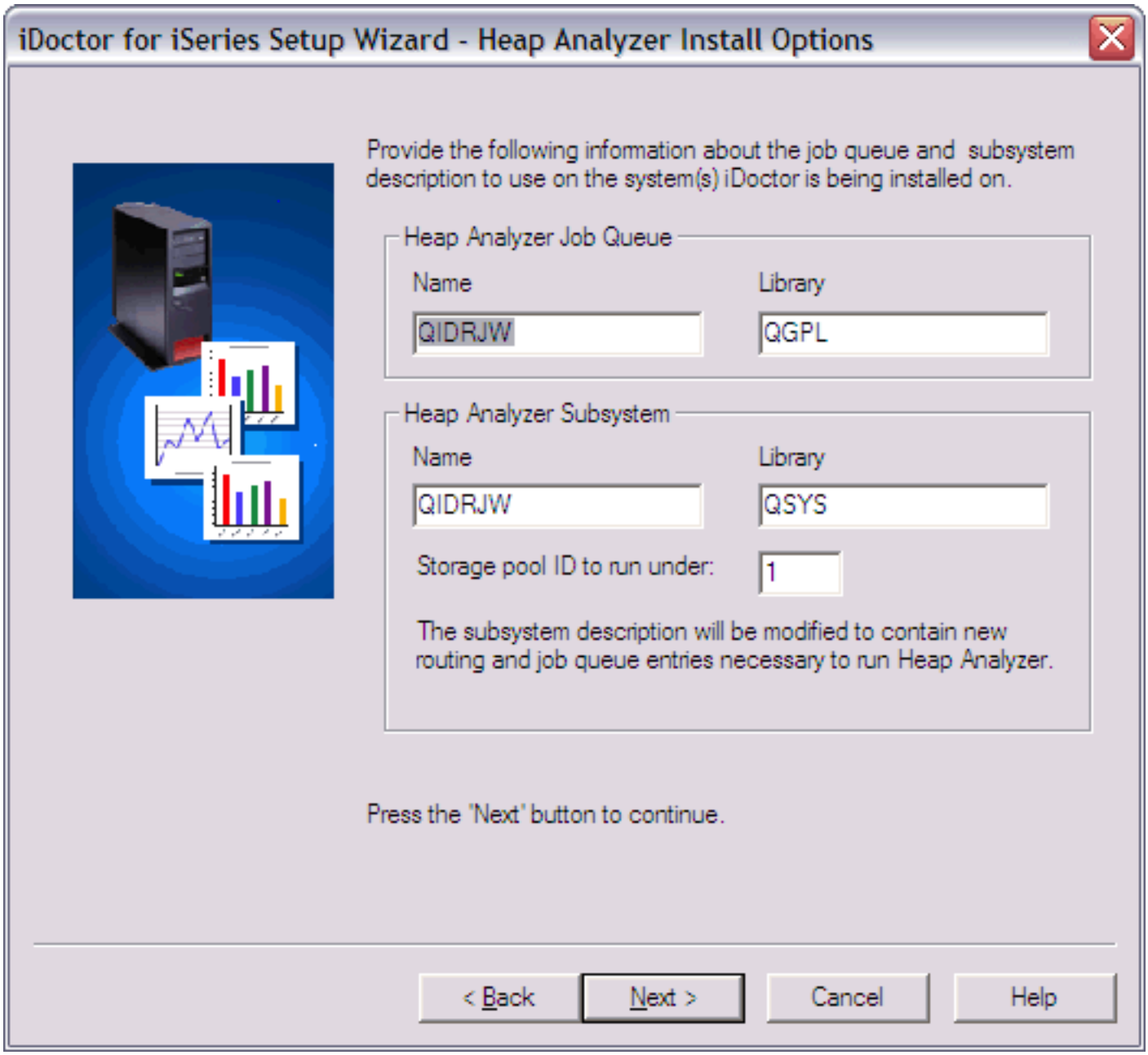
Click the 'Next' button to continue. Click 'Next' again on the Component Selection screen, using the default options.

Step 7 If you are installing the server portion of Heap Analyzer you will also be asked to specify a job queue name and a subsystem description the server code may use when running Heap Analyzer. If the subsystem description specified does not exist it may be optionally created. This subsystem and job queue will be used to run the Heap Analyzer "watch" jobs named WCHJVAOBJ and WCHJVAPROF. Heap Analyzer does not have a limit to the number of "watches" that can be active at one time.

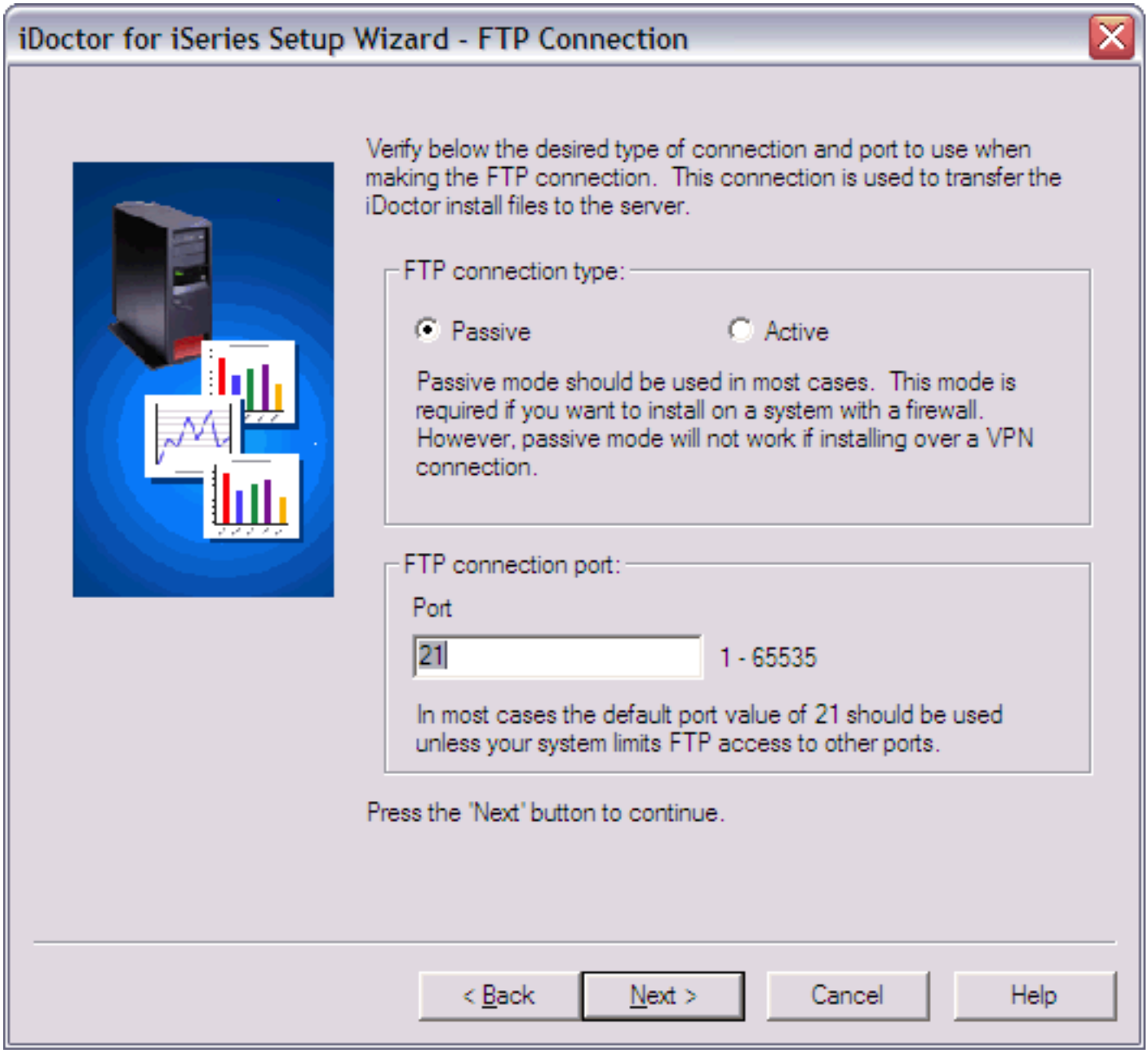
You must also indicate the storage pool ID that the Heap Analyzer jobs should run under. This parameter is required in order to add routing entries to the subsystem description.

If installing on multiple systems these same values specified will be used for all systems.

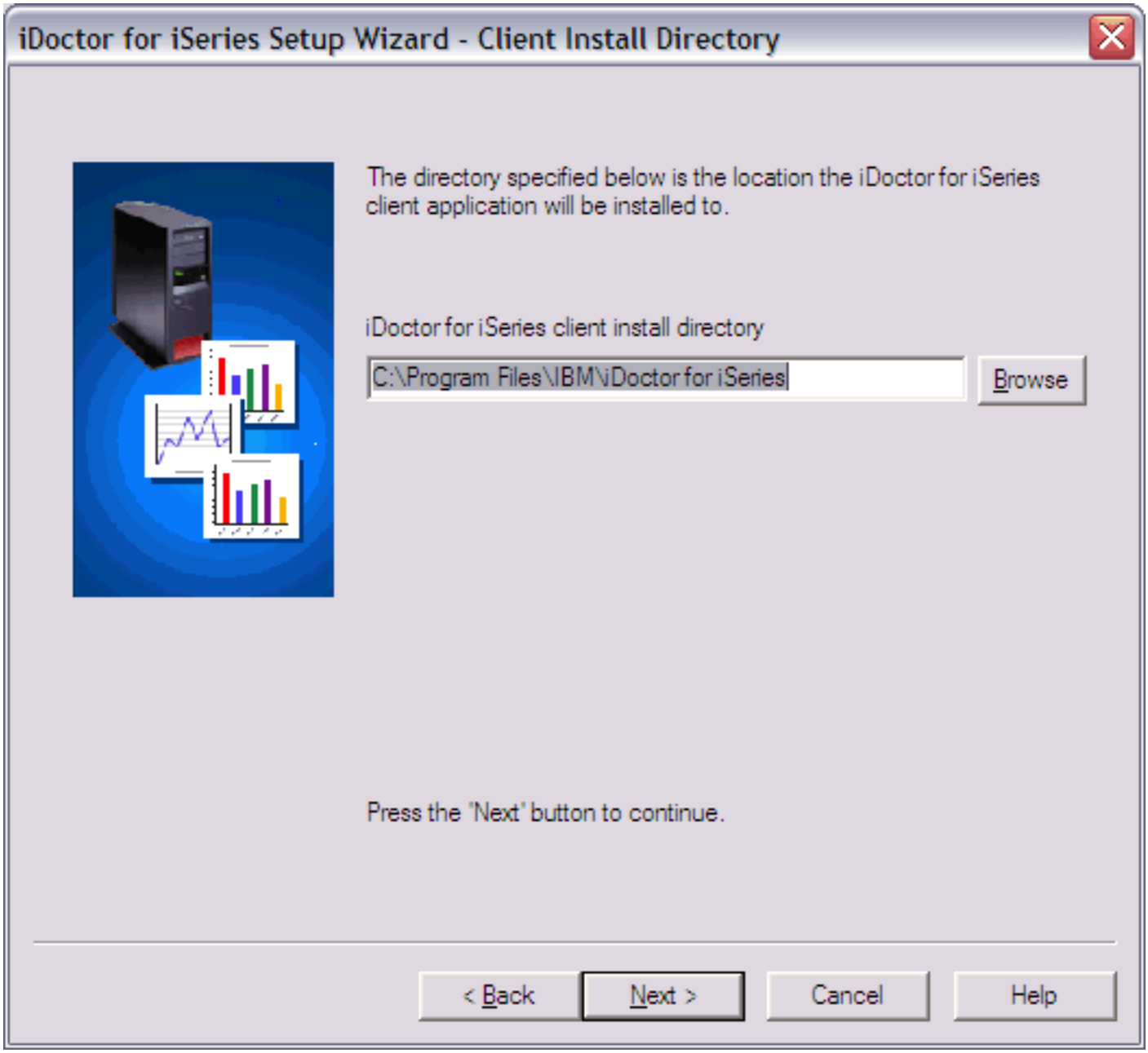
Press the 'Next' button to continue to the summary page.



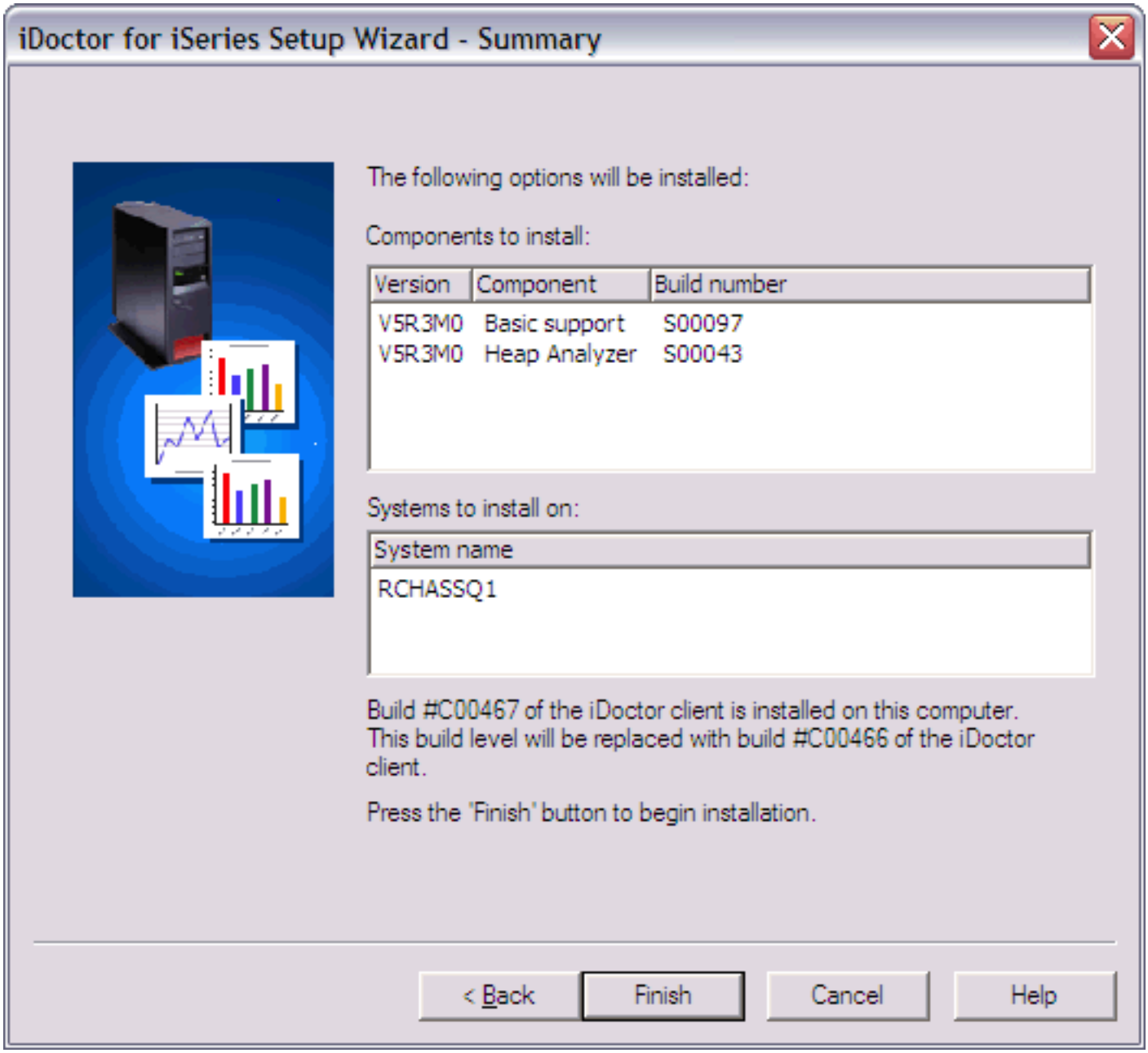
Step 8 The next page gives the user the option to specify which type of FTP connection should be used when performing the install. Only in unusual circumstances should anything other than the defaults be used on this page. However, if installing over a VPN connection and "Passive" FTP does not work, try using "Active" FTP instead.



Step 9 This screen allows you to specify the directory the iDoctor client program should be installed to.



Step 10 A summary of your selections appears on the final screen.



Step 11 Clicking the 'Finish' button will copy all of the files and run the commands necessary to install the server and/or client portion of Heap Analyzer. The server portion of the installation may take a few minutes.

After the install completes the setup log file will be shown. If any errors occur, send this file to idoctor@us.ibm.com for assistance.

Table of Contents | Previous | Next

2.4 Installing PTDV

PTDV has the following requirements:

On the server:

Requirements	Thin Client	Thick Client	Three Tier
OS/400 V4R5 or later	X	X	X
TC1 - TCP/IP Connectivity Utilities	X	X	X
JV1 - Developer Kit for Java		X	X
JC1 - Toolbox for Java	X		X
Host servers option of OS/400		X	X

On the client:
The PTDV client requires the following Java environment to run correctly:

- Java 2 runtime environment (JDK 1.2 or higher) is required. PTDV is not supported for JDK 1.1.x. Java 1.3 or higher is recommended.
- iSeries Toolbox for Java. This must be OS/400 V5R3M0, or a release more recent than that of the server.

Refer to the alphaWorks web site for additional information on downloading and installing PTDV.

Table of Contents Previous Next

2.5 Installing Other Components

The components referred to in section 1.5 are installed automatically when installing either PEX Analyzer, Job Watcher or Heap Analyzer. No other libraries need be installed on the server.

Table of Contents | Previous | Next

2.6 How to authorize use of iDoctor for iSeries

Note: This section only applies to Job Watcher and PEX Analyzer.

There are two ways to authorize use of iDoctor for iSeries on a server.

Option 1 Provide the access code after launching a component from the GUI. You will be prompted for the access code if a valid access code could not be found on your system.

Option 2 By using the green screen command QIDRGUI/ADDPRDACS.

Follow these steps:
a) Open an iSeries interactive session:
b) Type QIDRGUI/ADDPRDACS and press F4
c) Type in the access code that you were given by IBM Support.
d) Press enter.

You are now authorized to use iDoctor for iSeries. If you are using the tool with an evaluation code, your access will expire 45 days from the date it was dispatched.

You may request an access code for an evaluation version by sending email to idoctor@us.ibm.com

Table of Contents | Previous | Next

2.7 Uninstalling iDoctor for iSeries

Note: This section does not apply to the PTDV component.

Server side

To remove an iDoctor for iSeries component from the server, the libraries created during installation must be deleted.

The following table describes the libraries installed on the server, per component.

	Data Explorer	Heap Analyzer	PEX Analyzer	Job Watcher
Libraries	QIDRGUI	QIDRGUI, QIDRHAJ	QIDRGUI, QIDRPA	QIDRGUI, QIDRWCH

Client side

To remove iDoctor for iSeries from your PC select the uninstall program from the Start Menu: Start -> Programs -> iDoctor for iSeries -> Uninstall iDoctor for iSeries.

Table of Contents | Previous | Next

2.8 Ports needed for GUI access

The following table lists the various ports needed for GUI connections on the iSeries. The shaded entries are the ports/functions required by iDoctor.

	PC Function		Server Name	Port Non-SSL	Port SSL
	Server Mapper		as-svrmap	449	449
	License Management		as-central	8470	9470
	Database Access		as-database	8471	9471
	Data Queues		as-dtaq	8472	9472
	Network Drives		as-file	8473	9473
	Network Printers		as-netprt	8474	9474
	Remote Command		as-rmtcmd	8475	9475
	Signon Verification		as-signon	8476	9476

Telnet (PC5250 Emulation)	telnet	23	992
HTTP Administration	as-admi >	2001	2010
POP3 (MAPI)	pop3	5010	---
Management Central	as-mgtc >	5555	5566
Ultimedia Services	as-usf	8480	9480
DRDA	DRDA	446	---
DDM	DDM	447	448
IBM® AnyNet®	APPCoverTCPIP	397 (TCP and UDP)	---
NetServer	netbios >	137	---
NetServer	CIFS	455	
NetServer	netbios >	139	---
RUNRMTCMD	REXEC	512	---

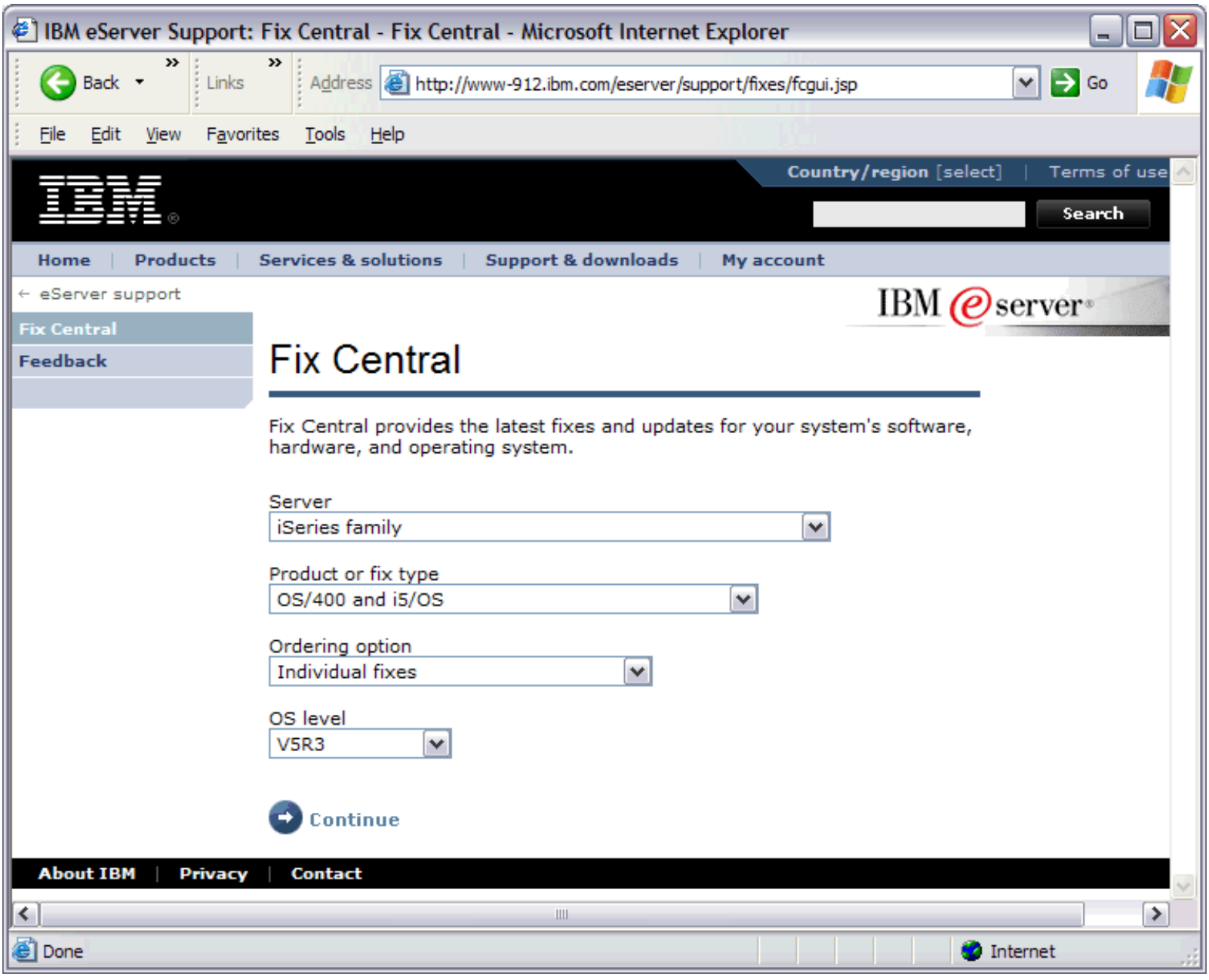
Table of Contents | Previous | Next

2.9 PTF installation

The GUI install process will install the server objects and programs on your iSeries and the client code on your PC. However, this does not include the process of installing the required PTFs for the iDoctor component you are using. The PTFs required must be manually installed via whichever method normally used to get PTFs for your iSeries.

The following information outlines the steps needed to download and install the PTFs onto your iSeries using the Fix Central website:

Step 1 Visit the Fix Central website and select the choices shown in the screenshot below. Click the 'Continue' button.

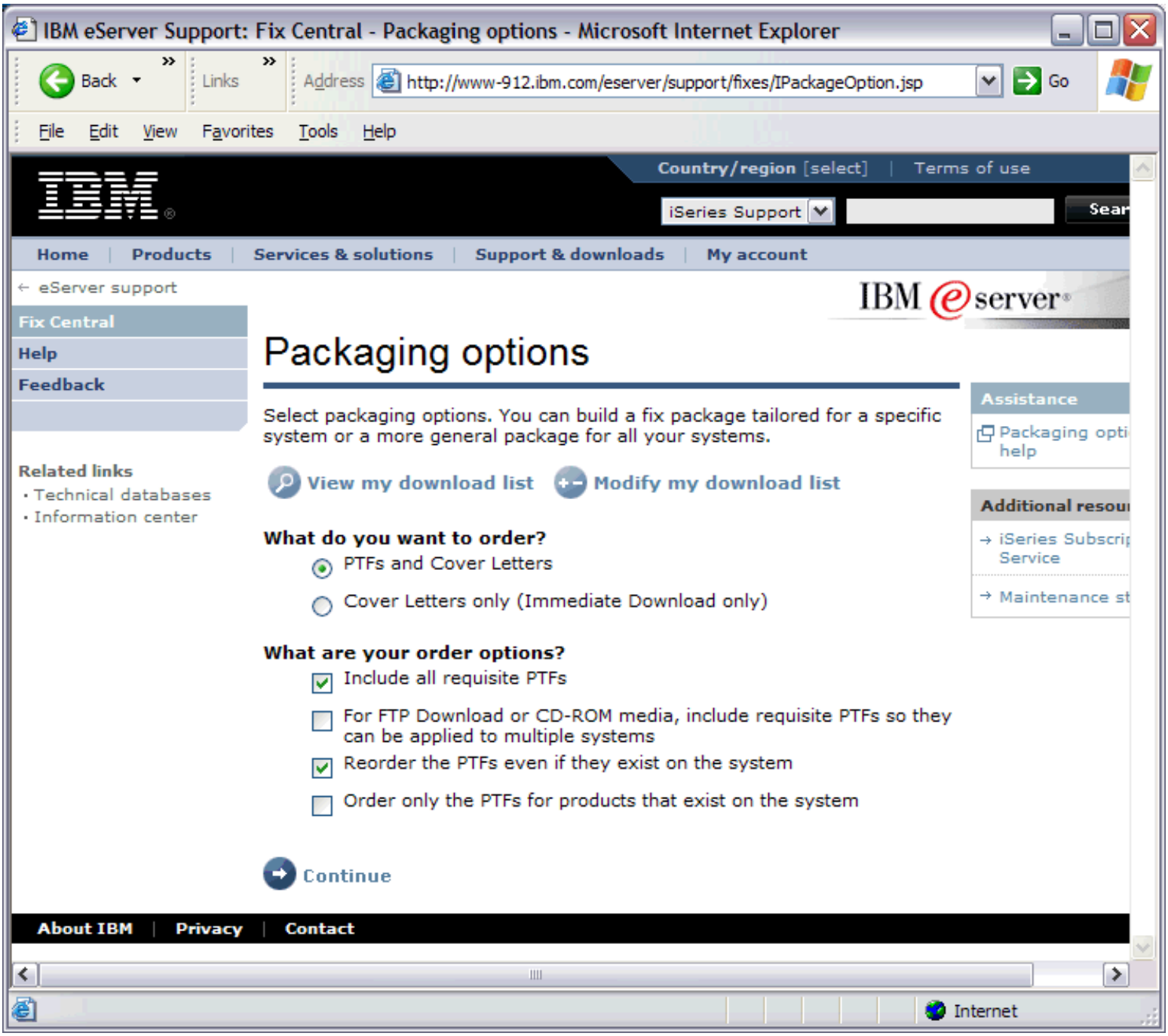


Step 2 Signon using your IBM registration user id and password. If you don't have a user id, click the "Need a user ID?" link in the right navigation bar.

Step 3 On the next screen enter the names of the PTFs needed. The PTF names are listed on the following page: https://www-912.ibm.com/i_dir/idoctor.nsf/downloadsV5R3.html

Step 4 Click the 'Continue' button.

Step 5 On the Packaging options page take the defaults and click the 'Continue' button.



Step 6 On the Download options page scroll to the last question on the page and answer the question about the location of your iSeries and click the 'Continue' button.

Step 7 On the next page you will be prompted for the system name and user id and password to use to connect to your iSeries so the PTFs can be uploaded to your system. Scroll down and click the 'Next' button on this page after providing this information (if it is not visible).

Step 8 On the next screen, fill in the customer information and click the 'Next' button.

Step 9 The PTFs will be downloaded to yours iSeries.

Step 10 At this point the PTFs have been sent to your iSeries but are not yet useable. You can check the status of the PTFs that were just sent to your iSeries using the WRKPTF command.
Use option 11 (load/apply) to load and apply the PTF on your system.

Table of Contents | Previous | Next

Part II Server-side components

This part covers the server-side (libraries, database files, commands) of each iDoctor for iSeries component.

Table of Contents Previous Next

Chapter 1 Job Watcher

This chapter describes the Job Watcher component on the server side. The following topics will be covered:

- Libraries QIDRGUI, QIDRWCH
- Database files
- Command descriptions

Table of Contents | Previous | Next

1.1 Libraries

When Job Watcher is installed, two libraries are created on the iSeries. These libraries are named QIDRGUI and QIDRWCH.

QIDRGUI contains programs and commands needed by the iDoctor for iSeries client. It also contains objects that are used by both Job Watcher and PEX Analyzer.

Library QIDRWCH is the Job Watcher library for release V5R3. This library contains programs and commands needed to create and work with Job Watcher data. Most of the output files produced by Job Watcher are located in library QSYS (the files are named QAPYJW*). Some of the output files are additional files (named QAIDR*) used by the collection command WCHJOB and can be found in library QIDRWCH. Both sets of files are described in the next sections.

Table of Contents | Previous | Next

1.2 Job Watcher collection files (from QSYS)

This section describes the database files that are produced by Job Watcher. These files are created in user libraries when a job watch is created using the WCHJOB command in library QIDRWCH. A member will be created in each file matching the name of the Job Watch.

Files with optional data are not created unless specified to be included by the collector command WCHJOB.

Table of Contents | Previous | Next

1.2.1 QAPYJWAIGP - Activation groups

Description: This file contains activation group information

Optional: Yes using WCHJOB fixed data type parameter DATATYPEU option *ACTGRPDETAIL

Record: One record is created per process per activation group per interval

Field name	Field Description	Format	Comments
INTERVAL	Interval number	Binary(4)	
TASKCOUNT	Initial thread task count	Binary(8)	
ACTGRPKEY	Activation group key	Binary(4)	
AGRESERVE	Reserved	Binary(4)	
ACTGRP	Activation group name	Char(30)	
AGROOTNAME	Activation group root program name	Char(30)	
AGROOTTYPE	Activation group root program type	Hex(2)	
AGROOTLIB	Activation group root program library name	Char(10)	
AGTYPE	Activation group type	Hex(2)	
AGSTATE	Activation group state	Binary(2)	
AGSTGMODL	Activation group storage model	Binary(2)	
AGSHARED	Activation group shared flag	Binary(2)	
AGMARK	Activation group mark	Binary(8)	
AGDFTHSIZ	Activation group default heap size	Binary(8)	
AGDFTHBLKS	Activation group default heap blocks	Binary(4)	
AGOTHERHS	Activation group number of other heaps	Binary(4)	

Table of Contents | Previous | Next

1.2.2 QAPYJWAIHP - Activation group - heap statistics

Description: This file contains heap information for an activation group

Optional: Yes using WCHJOB fixed data type parameter DATATYPEU option *ACTGRPDETAIL

Record: One record is created per heap per activation group per process per interval

Field name	Field Description	Format	Comments
INTERVAL	Interval number	Binary(4)	
TASKCOUNT	Initial thread task count	Binary(8)	
ACTGRPKEY	Activation group key	Binary(4)	
AGOTSIZE	Activation group other heaps heap size	Binary(8)	
AGOTID	Activation group other heaps heap ID	Binary(8)	
AGOTBLKS	Activation group other heaps heap block count	Binary(4)	
AGHRESERVE	Reserved	Char(4)	

Table of Contents | Previous | Next

1.2.3 QAPYJWAIPA - Activation group - program activations

Description: This file contains program activation information

Optional: Yes using WCHJOB fixed data type parameter DATATYPEU option *ACTGRPDETAIL

Record: One record is created per program activation per activation group per process per interval

Field name	Field Description	Format	Comments
INTERVAL	Interval number	Binary(4)	
TASKCOUNT	Initial thread task count	Binary(8)	
ACTGRPKEY	Activation group key	Binary(4)	
PACTNAME	Program activation program name	Char(30)	
PACTPGMTYP	Program activation program type	Hex(2)	
PACTLIB	Program activation program library name	Char(10)	
PACTRSVD	Reserved	Char(20)	
PACTLICTYP	Program activation LIC activation type	Hex(2)	
PACTRES	Reserved	Hex(2)	
PACTFRAMES	Program activation static frame count	Binary(4)	
PACTFRAMSZ	Program activation total static frame size	Binary(8)	

Table of Contents Previous Next

1.2.4 QAPYJWBKT - Job wait state accounting bucket mapping

Description: This file contains the mapping information for job wait state accounting

Optional: No

Record: One record is created per enum - the full set of enums will be written at the beginning of the collection

Field name	Field description	Format	Comments
INTERVAL	Interval number	Binary(4)	
BUCKETNUM	Bucket number	Binary(4)	
BUCKETDESC	Bucket description	Char(50)	
ENUM	Specific block ID number	Binary(4)	
EYE	Eye catcher	Char(3)	

Table of Contents | Previous | Next

1.2.5 QAPYJWINTI - Collector interval totals

Description: This file contains information pertaining to the interval

Optional: No

Record: One record is created per interval

Field name	Field description	Format	Comments
INTERVAL	Interval number	Binary(4)	
ISTARTTOD	Time of day at start of interval	Timestamp(26)	
IENDTOD	Time of day at end of interval	Timestamp(26)	
SYSTDECOUNT	System TDE count Reflects the total number of TDEs that were running on the system at the time the sample was taken.	Binary(4)	
SELTDECNT	Selected TDE count The number of TDEs that matched the thread selection criteria. These TDEs are reported in the QAPYJWSTS file.	Binary(4)	
ASELTDECNT	Active selected TDE count The number of TDEs that matched the thread selection criteria and consumed CPU during the interval. These TDEs will be reported in the QAPYJWTDE file.		

EXMTDECNT	Examined TDE count Currently unused.	Binary(4)	
ICRITSTAT	Conditional criteria status 1 = condition met 0 = condition not met	Char(1)	

Table of Contents | Previous | Next

1.2.6 QAPYJWJVM - Java virtual machine

Description: This file contains java JVM scoped data

Optional: No

Record: One record is created per JVM per interval. Note: GC in field descriptions indicates “Garbage Collector”

Field name	Field description	Format	Comments
INTERVAL	Interval number	Binary(4)	
TASKCOUNT	Initial thread task count	Binary(8)	
GCHEAPSZ	GC heap size (in bytes)	Binary(8)	
GCINITSIZE	GC initial size	Binary(4)	
GCMAXSIZE	GC maximum size	Binary(4)	
GCSWEEPS	GC sweep count	Binary(4)	
GCSTAGE	GC current stage ID	Binary(4)	
COL_TYPE	GC collection type ID	Binary(8)	
CYCLEDURAT	GC last cycle duration (in millisecs)	Binary(8)	
SECGCTHRDS	Secondary GC threads	Binary(8)	
ALCHPFND	GC allocated heap space found	Binary(8)	
ALCHPPRV	GC allocated heap space previous	Binary(8)	
ALCHPTCYCL	GC allocated heap this cycle	Binary(8)	
ALCHPPCYCL	GC allocated heap previous cycle	Binary(8)	
GC_LIVE	GC active object count	Binary(8)	
GCDEADINT	GC interned strings collected	Binary(8)	
GC_DEAD	GC collected object count	Binary(8)	
GCCURTHRH	GC current threshold	Binary(8)	
COLSTRTIME	GC collection start time (microseconds since IPL)	Binary(8)	

HPSIZESTRT	GC heap size start (in bytes)	Binary(8)	
HPSIZEEND	GC heap size end (in bytes)	Binary(8)	
HPFREESTRT	GC heap free memory start (in bytes)	Binary(8)	
HPFREEEND	GC heap free memory end (in bytes)	Binary(8)	
CTECOLSTR	GC collection cycle table entry start time (microseconds since IPL)	Binary(8)	
GCCYCLENBR	GC cycle number	Binary(4)	
CYCLRUNTIM	GC cycle run time (in millisecs)	Binary(4)	
OLDMARKSET	GC empty old mark set (in millisecs)	Binary(4)	
SYNC1DUR	GC handshake duration (in millisecs)	Binary(4)	
SYNC12DUR	GC transition 1 -> 2 duration (in millisecs)	Binary(4)	
SYNC2DUR	GC handshake duration (in millisecs)	Binary(4)	
SYNC2ASDUR	GC transition 2 -> async duration (in millisecs)	Binary(4)	
ASYNCDUR	GC async handshake duration (in millisecs)	Binary(4)	
ASYNFINDUR	GC async finish TRC duration (in millisecs)	Binary(4)	
GLOBALDUR	GC process object in reg (in millisecs)	Binary(4)	
FINTRCDUR	GC finish trace duration (in millisecs)	Binary(4)	
REFOBJDUR	GC reference object duration (in millisecs)	Binary(4)	
SWEEPDUR	GC sweep duration (in millisecs)	Binary(4)	
SOFTREFS	GC soft refs processed	Binary(4)	
CLRDREFS	GC cleared refs	Binary(4)	
WEAKREFS	GC weak refs processed	Binary(4)	
FINALREFS	GC final refs processed	Binary(4)	
PHNTOMREFS	GC phantom refs processed	Binary(4)	

GLOBALWEAK	GC global weak refs processed	Binary(4)	
CLRDSFTREF	GC soft refs cleared	Binary(4)	
CLRDWKREF	GC weak refs cleared	Binary(4)	
CLRDFNLREF	GC final refs cleared	Binary(4)	
CLRDPHREF	GC phantom refs cleared	Binary(4)	
CLRDGBLREF	GC global weak refs cleared	Binary(4)	
JVMRESERVE	Reserved	Char(8)	
JVMHANDLE	JVM handle	Hex(8)	
GCSTAGEC	GC current stage	Char(30)	
COL_TYPEC	GC collection type	Char(20)	
COLTIMESTC	GC collection start time	Timestamp(26)	
CTECOLSTRC	GC collection cycle table entry start time	Timestamp(26)	

Table of Contents | Previous | Next

1.2.7 QAPYJWJVTH - Java wait object information

Description: This file contains java TDE scoped data

Optional: No

Record: One record is created per java thread in java wait per interval

Field name	Field description	Format	Comments
INTERVAL	Interval number	Bin(4)	
TASKCOUNT	Task count	Bin(8)	
JVAOBJN	Last waited on java object name	VARCHAR(UTF8)	
THDNAME	Java thread name	VARCHAR(Unicode)	

Table of Contents | Previous | Next

1.2.8 QAPYJWPRC - Process scoped data

Description: This is the main file for Job Watcher process scoped data.

Optional: No

Record: One record is created per process per interval

Field name	Field description	Format	Delta?	Comments
INTERVAL	Interval number	Binary(4)	N	
TASKCOUNT	Initial thread task count	Binary(8)	N	
JOBSBS	Job subsystem	Char(10)	N	
JOBTYPE	Job type	Char(1)	N	
JOBFNCTN	Job function	Char(14)	N	
JOBSTATUS	Job status	Char(4)	N	
DELTAPRCPU	Job CPU in microseconds	Binary(8)	Y	
ACTTHREADD	Threads active	Binary(4)	Y	
ACTTHREADC	Total threads active since job start	Binary(4)	N	
CRTTHREADD	Threads created	Binary(4)	Y	
CRTTHREADC	Total threads created since job start	Binary(4)	N	
LDIOWRT	LDIO writes	Binary(4)	Y	
LDIORD	LDIO reads	Binary(4)	Y	
LDIOOTHR	LDIO other (non reads/writes)	Binary(4)	Y	
LDIOUPD	LDIO updates	Binary(4)	Y	
LDIODEL	LDIO deletes	Binary(4)	Y	
LDIOFEOD	LDIO FEODs	Binary(4)	Y	

LDIOCOMIT	LDIO commits	Binary(4)	Y	
LDIOROLLB	LDIO rollbacks	Binary(4)	Y	
LDIOOPEN	LDIO opens	Binary(4)	Y	
LDIOCLOSE	LDIO closes	Binary(4)	Y	
LDIOIXBLD	LDIO index builds	Binary(4)	Y	
LDIOSORT	LDIO sorts	Binary(4)	Y	
CMNWRT	Communication file writes	Binary(4)	Y	
CMNRD	Communication file reads	Binary(4)	Y	
LDTAQSND	Data queue sends	Binary(4)	Y	
LDTAQRCV	Data queue receives	Binary(4)	Y	
LDTAAOP	Data area operations	Binary(4)	Y	
LUSRSPCIOP	User space/index operations	Binary(4)	Y	
TXAPPIQT	Application input queuing time (in microsecs)	Binary(8)	Y	
TXINQTRAN	Application input queuing transactions	Binary(4)	Y	
TXRSCUT	Resource usage time (in microsecs)	Binary(8)	Y	
TXRSCUTRAN	Resource usage transactions	Binary(4)	Y	
TXDSPLRT	Display I/O response time (in microsecs)	Binary(8)	Y	
TXDSPLTRAN	Display I/O transactions	Binary(4)	Y	
IFSSYMLRD	IFS symbolic link reads	Binary(4)	Y	
IFSDIRRD	IFS directory reads	Binary(4)	Y	
IFSLUCHIT	IFS lookup cache hits	Binary(4)	Y	
IFSLUCMIS	IFS lookup cache misses	Binary(4)	Y	
IFSOPENS	IFS opens	Binary(4)	Y	
IFSDIRCRT	IFS directory creates	Binary(4)	Y	
IFSNDIRCRT	IFS non-directory creates	Binary(4)	Y	
IFSDIRDLT	IFS directory deletes	Binary(4)	Y	
IFSNDIRDLT	IFS non-directory deletes	Binary(4)	Y	
SOCKRD	Socket reads	Binary(4)	Y	
SOCKWRT	Socket writes	Binary(4)	Y	

SOCKBRD	Socket bytes read	Binary(8)	Y	
SOCKBWRT	Socket bytes written	Binary(8)	Y	
OPENCURSOR	Fully opened SQL cursors	Binary(4)	N	
PSUCLOCURS	Pseudo closed SQL cursors	Binary(4)	N	
CURNUMACTG	Current number of activation groups **Note:** This value is always zero unless WCHJOB fixed data type parameter DATATYPEU option *ACTGRPSUM is specified	Binary(4)	N	
CURNUMACT	Current number of activations **Note:** This value is always zero unless WCHJOB fixed data type parameter DATATYPEU option *ACTGRPSUM is specified	Binary(4)	N	

Table of Contents | Previous | Next

1.2.9 QAPYPROC - Call stack procedure information

Description: This is the file contains procedure information for call stacks.

Optional: Yes using WCHJOB fixed data type parameter DATATYPEU option *CALLSTACK or dynamic data type parameter DATATYPEC options *CONFLICTCALLSTACK or *BADBLOCKCALLSTACK

Record: One record is created per Trace Back Table (TBT) address.

Field name	Field Description	Format	Comments
TBTADDR	TBT address	Hex(8)	
PGMLIB	Program library	Char(10)	
PGMNAME	Program name	Char(10)	
MODNAME	Module name	Char(10)	
PROCTYPE	Procedure type 0 – SLIC 1 – NMI 2 - OMI	Binary(2)	
PROCSTRADR	Procedure start address	Hex(8)	
PROCENDADR	Procedure end address	Hex(8)	
PROCNAME	Procedure name	VARCHAR	

Table of Contents | Previous | Next

1.2.10 QAPYJWRUNI - Collector status

Description: This file contains basic information about the collection and includes some system information – the file member will have a single record that will be updated each interval

Optional: No

Record: One record is created per collection

Field name	Field Description	Format	Comments
INTERVAL	Interval number	Binary(4)	
FILELEVEL	File level indicator	Binary(2)	
STARTTOD	Start time-of-day	Timestamp(26)	
ENDTOD	End time-of-day	Timestamp(26)	
COLLSIZE	MB of data collected	Binary(4)	
TDERCDCNT	Count of TDEs in last interval	Binary(4)	
CYCUSEC	Cycles per microsecond	Binary(4)	
COLLSTAT	Collector status S - Starting R – Running E - Ended	Char(1)	
CRITSTAT	Conditional criteria status	Char(1)	
SYSTNAME	System name	Char(8)	
SYSTSERIAL	System serial number	Char(8)	
SYSTTYPE	System type	Char(4)	
SYSTMODEL	System model	Char(4)	
NUMPROC	Number of processors	Binary(4)	

OSVRM	Operating system VRM	Char(6)	
CALLJOB	Fully qualified job name of caller	Char(28)	

Table of Contents | Previous | Next

1.2.11 QAPYJWSKJB - Socket communications - Job names

Description: This file lists jobs which are using each socket

Optional: Yes using WCHJOB fixed data type parameter DATATYPEU option *SOCKETJOBS

Record: One record is created per using job per socket per process per interval

Field name	Field description	Format	Comments
INTERVAL	Interval number	Bin(4)	
SKJBKEY	Key from QAPYJWSKTC	Bin(4)	
SKJOBNAME	Job or task using socket	Char(28)	

Table of Contents | Previous | Next

1.2.12 QAPYJWSKTC - Socket communications

Description: This file contains socket and TCP data

Optional: Yes using WCHJOB fixed data type parameter DATATYPEU option *SOCKETTCP

Record: One record is created per file descriptor per process per interval

Field name	Field description	Format	Comments
INTERVAL	Interval number	Binary(4)	
TASKCOUNT	Initial thread task count	Binary(8)	
SKJBKEY	Key into QAPYJWSKJB file	Binary(4)	
SOCRESRVAD	Reserved	Char(8)	
SOCRESRVDR	Reserved	Binary(2)	
SOCKOBJT	Socket object type	Binary(2)	
SOCRESRV1	Reserved	Binary(4)	
SOCRESRVVL	Reserved	Binary(2)	
SOCRESRVJC	Job count	Binary(2)	
SOCKFAM	Socket family	Binary(4)	
SOCKTYPE	Socket type (inet, unix, netbios, inet6)	Binary(4)	
SOCKSTATE	Socket state	Binary(4)	
SOCKTPIST	Socket TPI state	Binary(4)	
SOCKERR	Socket error	Binary(4)	
SOCKRBUF	Socket receive buffer	Binary(4)	
SOCKRLWAT	Socket receive lowat	Binary(4)	
SOCKSBUF	Socket send buffer	Binary(4)	

SOCKRCBQ	Socket receive bytes queued	Binary(4)	
SOCKLING	Socket linger	Binary(4)	
SOCKLINGO	Socket linger on/off	Char(1)	
SOCKRFC	Socket receive flow controlled	Char(1)	
SOCKKEOF	Socket end of file	Char(1)	
SOCKSFC	Socket send flow controlled	Char(1)	
SOCKSECUR	Socket secure	Char(1)	
SOCKNONBLK	Socket non-blocking	Char(1)	
SOCKKA	Socket keep alive	Char(1)	
SOCKDBG	Socket debug	Char(1)	
SOCKURCV	Socket receive timeout in microseconds	Binary(8)	
SOCKUSND	Socket send timeout in microseconds	Binary(8)	
TCPSTATE	TCP state	Binary(2)	
TCPTPIST	TCP TPI state	Binary(2)	
TCPUPROF	TCP user profile	Char(10)	
TCPRESRV1	Reserved	Char(2)	
TCPSRCADR	TCP source address	Binary(4)	
TCPDSTADR	TCP destination address	Binary(4)	
TCPSRBUF	TCP socket receive buffer	Binary(4)	
TCPRBUF	TCP receive buffer	Binary(4)	
TCPSSBUF	TCP socket send buffer	Binary(4)	
TCPSBUF	TCP send buffer	Binary(4)	

TCPFLAGS	TCP flags TCP flags 1 through 8 TCP bind ID TCP linger sync TCP FN2 sync TCP rxmt sync TCP probe sync TCP dally sync TCP keep-alive sync TCP DUP sync	Char(16)	
TCPCWND	TCP congestion window	Binary(4)	
TCPSQLEN	TCP send queue length	Binary(4)	
TCPSUNA	TCP suna	Binary(4)	
TCPSNEXT	TCP bytes sent	Binary(4)	
TCPUSNEXT	Application bytes sent to TCP	Binary(4)	
TCPSENDWIN	TCP send window	Binary(4)	
TCPSENDMAX	TCP send maximum	Binary(4)	
TCPRQLEN	TCP receive queue length	Binary(4)	
TCPRECVWIN	TCP receive window	Binary(4)	
TCPCURRXMT	TCP current re-transmit	Binary(4)	
TCPRXMTCNT	TCP re-transmit count	Binary(2)	
TCPRXMTTOT	TCP re-transmit total	Binary(2)	
TCPRXMTFST	TCP fast re-transmit	Binary(2)	
TCPRESRV2	Reserved	Char(2)	
TCPMAXBLOG	TCP maximum backlog	Binary(4)	
TCPCURBLOG	TCP current backlog	Binary(4)	

TCPLASTACK	TCP last ACK	Binary(4)	
TCPSRCPT	TCP source port	Binary(2)	
TCPDSTPT	TCP destination port	Binary(2)	
TCPSEQNUM	TCP sequence number	Binary(4)	
TCPACKNUM	TCP ACK number	Binary(4)	
TCP6SRCADR	TCP IP6 source address	Char(16)	
TCP6DSTADR	TCP IP6 destination address	Char(16)	
SKTCUSECS	Elapsed interval time in microseconds	Binary(8)	
SOCKFAMC	Socket family	Char(13)	
SOCKTYPEC	Socket type	Char(9)	
SOCKSTATEC	Socket state	Char(13)	
SOCKTPISTC	Socket TPI state	Char(24)	
SOCKERRC	Socket error	Char(15)	
SOCKDESCR	Socket descriptor	Binary(4)	
SOCKHAND	Socket handle	Hex(8)	
SOCKLCLPOR	Socket local port	Binary(4)	
SOCKRMTPOR	Socket remote port	Binsry(4)	
SOCKLCLADR	Socket local address	VARLEN	
SOCKRMTADR	Socket remote address	VARLEN	
TCPSTATEC	TCP state	Char(11)	
TCPTPISTC	TCP TPI state	Char(25)	
TCPSRCADRC	TCP source address	VARLEN	
TCPDSTADRC	TCP destination address	VARLEN	
TCPAPPSBYD	Delta application bytes sent to TCP	Binary(4)	
TCPDBS	Delta TCP bytes sent	Binary(4)	
TCPDBR	Delta TCP bytes received	Binary(4)	
TCPDBA	Delta TCP bytes ACKed	Binary(4)	
TCPDSQL	Delta TCP send queue length	Binary(4)	
TCPRQLEND	Delta TCP receive queue length	Binary(4)	
TCPCURBLOD	Delta TCP current backlog	Binary(4)	
TCPDBSPS	TCP bytes sent per second	Binary(4)	

TCPDBRPS	TCP bytes received per second	Binary(4)	
TCPCWNDCSQ	TCP consecutive congestion decreases	Binary(4)	
TCPCWNDA	TCP congestion alert	Char(1)	
TCPDSQLA	TCP delta send queue length alert	Char(1)	
TCPAPPSBYA	Application send bytes to TCP alert	Char(1)	
TCPRMTRWIA	TCP remote receive window alert	Char(1)	
TCPLCLRWID	TCP local receive window alert	Char(1)	

Table of Contents | Previous | Next

1.2.13 QAPYJWSQL - SQL statements

Description: This file contains SQL statement information

Optional: Yes

Record: One record is created per statement level per thread per interval. There can be up to 2 statements reported for a thread in any given interval.

Field name	Field description	Format	Comments
INTERVAL	Interval number	Binary(4)	
TASKCOUNT	Task count	Binary(8)	
SQLHVARKEY	Key for QPYJWSQLH file	Binary(8)	
SRCLIB	SQL package source library	Char(10)	
SRCFILE	SQL package source file	Char(10)	
SRCMBR	SQL package source member	Char(10)	
SRCDATE	SQL package source date	Char(13)	
PKGLIB	SQL package library/container	Char(18)	
PKGNAME	SQL package name	Char(18)	
RDBSNAME	Remote DBS name	Char(18)	
HOSTREAL	Actual number of host variables	Binary(2)	
HOSTLOGGED	Number of host variables in QPYJWSQLH	Binary(2)	
MORE	Another statement also associated with this statement (1 = Yes)	Char(1)	
STMTCSID	SQL statement CCSID	Binary(2)	
SQLSTMTLEN	SQL statement full length	Binary(4)	
SQLSTMT	SQL statement	VARCHAR	

Table of Contents | Previous | Next

1.2.14 QAPYJWSQLH - SQL host variables

Description: This file contains SQL host variable information

Optional: Yes

Record: One record is created per host variable per thread per interval

Field name	Field description	Format	Comments
INTERVAL	Interval number	Binary(4)	
TASKCOUNT	Task count	Binary(8)	
HVARNUM	Number of SQL host variables returned	Binary(4)	
HDATANUM	Host variable number	Binary(2)	
HVARTYPE	Host variable type	Binary(2)	
HDATALEN	Host variable full length	Binary(2)	
HDECIMAL	Number of decimals	Binary(2)	
HDATA	Host variable data	VARCHAR	

Table of Contents | Previous | Next

1.2.15 QAPYJWSQLO - SQL open cursors

Description: This file contains Open Cursor List (OCL) information

Optional: Yes

Record: One record is created per OCL per process per interval

Field name	Field description	Format	Comments
INTERVAL	Interval number	Binary(4)	
TASKCOUNT	Initial thread task count	Binary(8)	
OSQCCSR	Cursor name	Char(18)	
OSQCSTMT	Prepared statement name	Char(18)	
OSQCFILE	File name	Char(10)	
OSQCOAID	AuthId or library name	Char(10)	
OSQCINDX	Statement array index	Binary(2)	
OSQCCUSR	Current user at open time	Char(10)	
OSQCCACT	Activation group for cursor	Binary(4)	
OSQCCMTLVL	Cursor commit level	Char(1)	
OSQCCCTXID	Context ID at open time	Char(8)	
OSQCPCLS	Pseudo close flag	Char(1)	
OSQCXTD	Extended dynamic flag	Char(1)	
OSQCTEMP	Space-constrained destroy at close flag	Char(1)	
OSQCHOLD	Cursor hold attribute flag	Char(1)	
OSQCRR	Repeatable read cursor flag	Char(1)	
OSQCCPRCED	Opened by QSQPRCED API flag	Char(1)	
OSQCLOBLOC	LOBs associated flag	Char(1)	

OSQCCMTLVE	Commit level escalated flag	Char(1)	
OSQCUDF	Uses UDFs flag	Char(1)	
OSQCMGLCKS	Managing locks flag	Char(1)	
WSQCHDRCLS	Always hard close flag	Char(1)	
WSQCSEQ	SQE processing flag	Char(1)	
WSQCCNTS	NTS use flag	Char(1)	
OCLSTCSID	SQL statement CCSID	Binary(2)	
OCLSTMTLEN	SQL statement full length	Binary(4)	
OCLSTMTTXT	SQL statement text	VARCHAR	

Table of Contents | Previous | Next

1.2.16 QAPYJWSQLP - SQL prepared statements

Description: This file contains SQL Prepared Statement Area (PSA) information

Optional: Yes

Record: One record is created per PSA per thread per interval

Field name	Field description	Format	Comments
INTERVAL	Interval number	Binary(4)	
TASKCOUNT	Task count	Binary(8)	
PSQPSALC	Number of allocated entries	Binary(4)	
PSQPSNUM	Number of in-use entries	Binary(4)	
PSQPPSASZ	Size of prepared statement area	Binary(4)	
PSQPSUBP	*ENDSQL option flag	Char(1)	
PSQPENDJ	*ENDJOB option flag	Char(1)	
PSQPNOCC	No CCSID for any host variables flag	Char(1)	
PSQPENDACT	*ENDACTGRP specified flag	Char(1)	
PSQPCMPTHR	Compression threshold	Binary(4)	
PSQPDUMCNT	Dummy statement count	Binary(4)	
PSQPRTNTYP	Routine type	Binary(2)	
PSQPLSTIDX	Last index in area	Binary(4)	
PSQPCMPCNT	Number of compresses	Binary(2)	
PSQPNUMV	Host variable count	Binary(2)	
PSQPSQTL	QDT and access plan length	Binary(4)	
PSQPSQL2	Second QDT and access plan length	Binary(4)	
PSQPSTML	Statement name length	Binary(2)	
PSQPSNAM	Statement name	Char(18)	

PSQPSUSES	Usage or open count	Binary(4)	
PSQP_SWC	In system-wide cache flag	Char(1)	
PSQPCMTUSE	Compresses since last used	Binary(2)	
PSASTCSID	SQL statement CCSID	Binary(2)	
PSASTMTLEN	SQL statement full length	Binary(4)	
STMTTXT	SQL statement text	VARCHAR	

Table of Contents | Previous | Next

1.2.17 QAPYJWSTK - Call stack

Description: This file contains job call stack data

Optional: Yes using WCHJOB fixed data type parameter DATATYPEU option *CALLSTACK or dynamic data type parameter DATATYPEC options *CONFLICTCALLSTACK or *BADBLOCKCALLSTACK

Record: One record is created per thread/task per interval

Field name	Field description	Format	Comments
INTERVAL	Interval number	Binary(4)	
TASKCOUNT	Task count	Binary(8)	
NUMFRAMES	Number of stack frames	Binary(4)	
REASON	Stack collection reason S = Collecting all stacks B = Bad wait C = Conflict	Char(1)	
STKERROR	Error indicator. This indicator is set if there is an error during the collection of stack data	Char(1)	

STACK	Call stack Format of STACK field: hex (8) Instruction address bin (4) Offset char(2) Flags char(2) Reserved (alignment) hex (8) Stack frame address hex (8) TBT address Note: This format repeats within the STACK field for the number of times indicated in the field NUMFRAMES	VARCHAR	

Table of Contents | Previous | Next

1.2.18 QAPYJWSTS - TDE status

Description: This file contains TDE status information

Optional: No

Record: One record is created per selected TDE per interval

Field name	Field Description	Format	Comments
INTERVAL	Interval number	Binary(4)	
TASKCOUNT	Task count	Binary(8)	
TDESTATUS	Status of TDE in collection A = active I = idle (zero cpu) F = forced T = terminated	Char(1)	
CURWAIT	Current wait	Binary(2)	
CURWAITD	Current wait duration	Binary(8)	

Table of Contents | Previous | Next

1.2.19 QAPYJWTDE - TDE statistics

Description: This is the main file for Job Watcher job specific data. This file will not contain records for threads which have not consumed CPU during the interval. Records for zero CPU threads will be contained in the QAPYJWSTS file with a TDESTATUS value of “I”.

Optional: No

Record: One record is created per active selected TDE per interval.

Field Name	Field Description	Format	Delta?	Comments
INTERVAL	Interval number	Binary(4)	N	
TASKCOUNT	Task count	Binary(8)	N	
TDEUSECS	Elapsed interval time in microsecs. The time from the start of one “snap” to the start of the next “snap”.	Binary(8)	Y	
STARTOD	Time of day at snapshot start	Timestamp(26)	N	
STARTUSECS	Microseconds since IPL at start of snapshot	Binary(8)	N	
ENDUSECS	Microseconds since IPL at end of snapshot	Binary(8)	N	
THREADID	Thread ID	Hex(8)	N	
TDEJOBNAME	Job/task name	Char(32)	N	
THRDSTATUS	Thread status	Char(4)	N	
CURRUP	Current user profile	Char(10)	N	
BIRTHDAY	Job/task birth TOD	Timestamp(26)	N	
EXTENDER	Job name extender	Char(2)	N	
TDETYPE	Job or task type flag indicates LIC task or process	Char(1)	N	
DELTACPU	CPU Time (in microsecs) actual CPU used	Binary(8)	Y	

ORIGPRI	Original priority	Binary(2)	N	
PRIORITY	Current LIC priority	Binary(2)	N	
THREADPRI	Current XPF priority	Binary(2)	N	
PRICHG	Priority change indicator provided for file query purposes	Char(1)	N	
POOL	Pool ID	Binary(2)	N	
POOLCHG	Pool change indicator provided for file query purposes	Char(1)	N	
TOTWRT	Total DASD writes	Bin(4)	Y	
SYNDBRD	Synchronous DB reads	Bin(4)	Y	
SYNNDBRD	Synchronous non-DB reads	Bin(4)	Y	
SYNDBWRT	Synchronous DB writes	Bin(4)	Y	
SYNNDBWRT	Synchronous non-DB writes	Bin(4)	Y	
ASYDBRD	Asynchronous DB reads	Bin(4)	Y	
ASYNDBRD	Asynchronous non-DB reads	Bin(4)	Y	
ASYDBWRT	Asynchronous DB writes	Bin(4)	Y	
ASYNDBWRT	Asynchronous non-DB writes	Bin(4)	Y	
IOPENDING	I/O pending page faults	Bin(4)	Y	
SMSYNCIO	Waits for asynchronous writes	Bin(4)	Y	
FLTS	Page faults resulting in DASD reads	Bin(4)	Y	
ALLOCATED	Allocated DASD pages	Bin(4)	Y	
DEALLOCATED	Deallocated DASD pages	Bin(4)	Y	
ALLOCATEDT	Total DASD pages allocated since thread/task start	Bin(8)	N	
DEALLOCATEDT	Total DASD pages deallocated since thread/task start	Bin(8)	N	
SEIZE	Seize time (in microsecs)	Bin(4)	Y	
BINOVER	Binary overflows	Bin(4)	Y	
DECOVER	Decimal overflows	Bin(4)	Y	
FLOATOVER	Float overflows	Bin(4)	Y	
STMFRD	Stream file reads	Bin(4)	Y	
STMFWRT	Stream file writes	Bin(4)	Y	

MUTEX	Mutex wait time (in microsecs)	Bin(8)	Y	
ACTWAIT	Active to wait transitions	Bin(4)	Y	
WAITINEL	Wait to ineligible transitions	Bin(4)	Y	
ACTINEL	Active to ineligible transitions	Bin(4)	Y	
QCOUNT01	Queuing bucket 01 count - total CPU	Bin(4)	Y	
QCOUNT02	Queuing bucket 02 count - total waits	Bin(4)	Y	
QCOUNT03	Queuing bucket 03 count - total CPU queuing	Bin(4)	Y	
QCOUNT04	Queuing bucket 04 count	Bin(4)	Y	
QCOUNT05	Queuing bucket 05 count	Bin(4)	Y	
QCOUNT06	Queuing bucket 06 count	Bin(4)	Y	
QCOUNT07	Queuing bucket 07 count	Bin(4)	Y	
QCOUNT08	Queuing bucket 08 count	Bin(4)	Y	
QCOUNT09	Queuing bucket 09 count	Bin(4)	Y	
QCOUNT10	Queuing bucket 10 count	Bin(4)	Y	
QCOUNT11	Queuing bucket 11 count	Bin(4)	Y	
QCOUNT12	Queuing bucket 12 count	Bin(4)	Y	
QCOUNT13	Queuing bucket 13 count	Bin(4)	Y	
QCOUNT14	Queuing bucket 14 count	Bin(4)	Y	
QCOUNT15	Queuing bucket 15 count	Bin(4)	Y	
QCOUNT16	Queuing bucket 16 count	Bin(4)	Y	
QCOUNT17	Queuing bucket 17 count	Bin(4)	Y	
QCOUNT18	Queuing bucket 18 count	Bin(4)	Y	
QCOUNT19	Queuing bucket 19 count	Bin(4)	Y	
QCOUNT20	Queuing bucket 20 count	Bin(4)	Y	
QCOUNT21	Queuing bucket 21 count	Bin(4)	Y	
QCOUNT22	Queuing bucket 22 count	Bin(4)	Y	
QCOUNT23	Queuing bucket 23 count	Bin(4)	Y	
QCOUNT24	Queuing bucket 24 count	Bin(4)	Y	
QCOUNT25	Queuing bucket 25 count	Bin(4)	Y	

QCOUNT26	Queuing bucket 26 count	Bin(4)	Y	
QCOUNT27	Queuing bucket 27 count	Bin(4)	Y	
QCOUNT28	Queuing bucket 28 count	Bin(4)	Y	
QCOUNT29	Queuing bucket 29 count	Bin(4)	Y	
QCOUNT30	Queuing bucket 30 count	Bin(4)	Y	
QCOUNT31	Queuing bucket 31 count	Bin(4)	Y	
QCOUNT32	Queuing bucket 32 count	Bin(4)	Y	
	Note: all bucket times are in microsecs			
QTIME01	Queuing bucket 01 time - total CPU	Bin(8)	Y	
QTIME02	Queuing bucket 02 time - total wait time	Bin(8)	Y	
QTIME03	Queuing bucket 03 time - total CPU queuing	Bin(8)	Y	
QTIME04	Queuing bucket 04 time	Bin(8)	Y	
QTIME05	Queuing bucket 05 time	Bin(8)	Y	
QTIME06	Queuing bucket 06 time	Bin(8)	Y	
QTIME07	Queuing bucket 07 time	Bin(8)	Y	
QTIME08	Queuing bucket 08 time	Bin(8)	Y	
QTIME09	Queuing bucket 09 time	Bin(8)	Y	
QTIME10	Queuing bucket 10 time	Bin(8)	Y	
QTIME11	Queuing bucket 11 time	Bin(8)	Y	
QTIME12	Queuing bucket 12 time	Bin(8)	Y	
QTIME13	Queuing bucket 13 time	Bin(8)	Y	
QTIME14	Queuing bucket 14 time	Bin(8)	Y	
QTIME15	Queuing bucket 15 time	Bin(8)	Y	
QTIME16	Queuing bucket 16 time	Bin(8)	Y	
QTIME17	Queuing bucket 17 time	Bin(8)	Y	
QTIME18	Queuing bucket 18 time	Bin(8)	Y	
QTIME19	Queuing bucket 19 time	Bin(8)	Y	
QTIME20	Queuing bucket 20 time	Bin(8)	Y	
QTIME21	Queuing bucket 21 time	Bin(8)	Y	

QTIME22	Queuing bucket 22 time	Bin(8)	Y	
QTIME23	Queuing bucket 23 time	Bin(8)	Y	
QTIME24	Queuing bucket 24 time	Bin(8)	Y	
QTIME25	Queuing bucket 25 time	Bin(8)	Y	
QTIME26	Queuing bucket 26 time	Bin(8)	Y	
QTIME27	Queuing bucket 27 time	Bin(8)	Y	
QTIME28	Queuing bucket 28 time	Bin(8)	Y	
QTIME29	Queuing bucket 29 time	Bin(8)	Y	
QTIME30	Queuing bucket 30 time	Bin(8)	Y	
QTIME31	Queuing bucket 31 time	Bin(8)	Y	
QTIME32	Queuing bucket 32 time	Bin(8)	Y	
CURRSTATE	Current or last state (run, CPQ, wait)	Char(4)	N	
BLOCKBCKT	Current or last blocking bucket	Bin(2)	N	
LICWO	Current or last LIC wait object	Char(4)	N	
LICWOHNDL	Current or last LIC wait object handle	Hex(8)	N	
WOSEGTYP	Wait object segment type derived from wait obj addr	Hex(4)	N	
WOBASSEG	Wait object base segment address derived from wait obj addr	Hex(8)	N	
WOOBJTYP	Wait object object type derived from wait obj addr	Hex(4)	N	
WOOBJNAM	Wait object name derived from wait obj addr	Char(30)	N	
WOOBJTYPD	Wait object object type description derived from wait obj addr	Char(35)	N	
WOSEGTYPD	Wait object segment type description derived from wait obj addr	Char(35)	N	
HTASKCNT	Holding thread/task task count	Bin(8)	N	
HTYPE	Holding thread/task type (MI process or LIC task)	Char(1)	N	

HTASKNAME	Holding thread/task name	Char(28)	N	
BLOCKENUM	Current or last blocking enum	Bin(2)	N	
CURRWTDUR	Total time in current wait (in microsecs)	Bin(8)	N	
RECCNFLCT	Ordinal record number if DB record lock conflict	Bin(8)	N	
DFTSOCKD	Default socket descriptor	Bin(4)	N	
DFTSOCKTOD	Default socket TOD	Timestamp(26)	N	
DFTSOCKH	Default socket handle	Hex(8)	N	
DFTSOCKCHLV	Default socket cache level	Bin(2)	N	
LISSOCKD	Listen socket descriptor	Bin(4)	N	
LISSOCKTOD	Listen socket TOD	Timestamp(26)	N	
LISSOCKH	Listen socket handle	Hex(8)	N	
LISSOCKCHLV	Listen socket cache level	Bin(2)	N	
FRMESTOL	New mainstore frames stolen	Binary(4)	Y	
SREMOVE	Successful removes	Binary(4)	Y	
PSAINUSE	PSA entries in use	Binary(4)	Y	
SQLINTHRD	SQL statement in progress	Char(1)	N	
TRESERVE1	Reserved	Binary(4)	N	
TRESERVE2	Reserved	Binary(4)	N	
TRESERVE3	Reserved	Char(8)	N	

Table of Contents | Previous | Next

1.3 Job Watcher files (from QIDRWCH)

This section describes the database files that are used by Job Watcher residing in library QIDRWCH.

Table of Contents Previous Next

1.3.1 QAIDRGPH08 - Graph definitions

Description: This file is no longer used. Graph definitions are now stored in the database iDocQueries.mdb within the client install directory.

Table of Contents | Previous | Next

1.3.2 QAIDRSQL04 - Query definitions

Description: This file is no longer used. Query definitions are now stored in the database iDocQueries.mdb within the client install directory.

Table of Contents | Previous | Next

1.3.3 QAIDRSQC04 - Query definition column descriptions

Description: This file is no longer used. Column descriptions are now stored in the database iDocQueries.mdb within the client install directory.

Table of Contents | Previous | Next

1.3.4 QAIDRJWCPU - CPU Utilization statistics

Description: This file contains CPU utilization statistics to complement the data produced by the Job Watcher engine.

Collection: Yes

Record: Each record represents an interval the CPU utilization was gathered. These statistics may not be gathered every interval depending on the value specified by parameter MINCPUDTAT on the WCHJOB command.

Field name	Field Description	Format	Comments
INTERVAL	Interval number	Bin(8)	
TOD	Date/time the CPU statistics were gathered	Timestamp	
RSVDBIN1	Reserved field	Bin(4)	
DUSECS	Interval delta time (usecs)	Bin(18)	
DCPU01	Interval delta CPU used (usecs)	Bin(18)	
DCPU02	Interval delta configured CPU available	Bin(18)	
DCPU03	Interval delta uncapped CPU available	Bin(18)	
DCPU04	Interval delta secondary workload CPU used	Bin(18)	
DCPU05	Interval delta DB2 CPU used	Bin(18)	
DCPU06	Interval delta interactive CPU used	Bin(18)	
DCPU07	Interval delta interactive CPU available	Bin(18)	
DBTHRSHLD	DB2 CPU threshold	Bin(4)	
DBLIMIT	DB2 CPU limit	Bin(4)	

INTTHRSHLD	Interactive CPU threshold	Bin(4)	
INTLIMIT	Interactive CPU limit	Bin(4)	
CPC	Current processing capacity	Bin(9)	
PRCCOUNT	Current number of processors	Bin(4)	
RSVDBIN2	Reserved field	Bin(4)	
RSVDBIN3	Reserved field	Bin(18)	
RSVDBIN4	Reserved field	Bin(9)	

Table of Contents | Previous | Next

1.3.5 QAIDRJWDFN - Job Watcher definitions

Description: This file contains Job Watcher definitions that have been created in a user's library. This file is created everytime a Job Watch is submitted from the client in the library the Job Watch is being created into. The file stores the command string used to create the job watch. This information is used to construct the "creation settings" property page shown in the GUI.

Collection: No

Record: Each record contains the WCHJOB command string used to create a job watch in the library the file is located.

Field name	Field Description	Format	Comments
NAME	This is the member name/job watch name this record applies to	Char(10)	
OSVRM	i5/OS VRM 3 digit VRM level this graph defintion applies to V5R3 = 530	Char(3)	
COMMAND	WCHJOB command string	Char(6500)	

Table of Contents | Previous | Next

1.3.6 QAIDRJWENM - Wait point descriptions

Description: This file contains the specific wait point descriptions that are used by the GUI to dispay information about each type of wait that occurs on the system and shown in Job Watcher.

Collection: No

Record: Each record contains a wait point identifier (enum) and its descriptions. The ENUM field can be used to join against file QAPYJWBKT to determine the bucket and bucket description the enum is a part of.

Field name	Field Description	Format	Comments
ENUM	ENUM (wait point) identifier	Bin(4)	
ENUMDESC	ENUM description	Char(76)	

Table of Contents | Previous | Next

1.3.7 QAIDRJWRI - GUI collector status file

Description: This file is a nearly identical clone of file QAPYJWRUNI with some additioanl fields added at the end so the client can retrieve information about a list of job watches in a library more quickly. Unlike file QAPYJWRUNI which creates a separate member for every job watch, this file is a single member file.

Collection: Yes

Record: One record is created per collection

Field name	Field Description	Format	Comments
INTERVAL	Interval number	Binary(4)	
FILELEVEL	File level indicator	Binary(2)	
STARTTOD	Start time-of-day	Timestamp(26)	
ENDTOD	End time-of-day	Timestamp(26)	
COLLSIZE	MB of data collected	Binary(4)	
TDERCDCNT	Count of TDEs in last interval	Binary(4)	
CYCUSEC	Cycles per microsecond	Binary(4)	
COLLSTAT	Collector status S - Starting R – Running E - Ended	Char(1)	
CRITSTAT	Conditional criteria status	Char(1)	
SYSTNAME	System name	Char(8)	
SYSTSERIAL	System serial number	Char(8)	
SYSTTYPE	System type	Char(4)	

SYSTMODEL	System model	Char(4)	
NUMPROC	Number of processors	Binary(4)	
OSVRM	Operating system VRM	Char(6)	
CALLJOB	Fully qualified job name of caller	Char(28)	
MBR	Member/Job Watch name	Char(10)	
ENDSTS	WCHJOB ending status	Bin(4)	
APIRETCODE	API return code	Bin(4)	
TEXTDESC	Job Watch description	Char(50)	

Table of Contents | Previous | Next

1.3.8 QAIDRJWDBG - Summary debug file

Description: This file is for service use only and is only created using the CRTWCHSUM command with the DEBUG parameter set to Y.

Collection: No

Field name	Field Description	Format	Comments
NOTES01	Notes 01	Char(16)	
NOTES02	Notes 02	Char(16)	
NOTES03	Notes 03	Char(32)	
NOTES04	Notes 04	Char(32)	
NOTES05	Notes 05	Char(32)	

Table of Contents | Previous | Next

1.3.9 QAIDRJWIG1 - Summary file detail records #1

Description: This file is no longer used.

Table of Contents | Previous | Next

1.3.10 QAIDRJWIG2 - Summary file detail records #2

Description: This file contains the detail records produced by the default Job Watcher summary. It's used primarily for filling in the idle waits for jobs when displaying the run/wait time signature graph for a single job.

Collection: No but a member matching collection member is created when default summarization is ran from GUI.

Record: One record is created per TDE detected (active or idle) in the collection per interval.

Field Name	Field Description	Format	Comments
INTERVAL	Interval number	Binary(4)	
TASKCOUNT	Task count	Binary(8)	
THREADID	Thread ID	Hex(8)	
TDEJOBNAME	Job/task name	Char(32)	
TDETYPE	Job or task type flag indicates LIC task or process	Char(1)	
PRIORITY	Current LIC priority	Binary(2)	
	Note: all bucket times are in microsecs		
QTIME01	Wait bucket 01 time - total CPU	Bin(8)	
QTIME02	Wait bucket 02 time - total wait time	Bin(8)	
QTIME03	Wait bucket 03 time - total CPU queuing	Bin(8)	
QTIME04	Wait bucket 04 time	Bin(8)	
QTIME05	Wait bucket 05 time	Bin(8)	
QTIME06	Wait bucket 06 time	Bin(8)	
QTIME07	Wait bucket 07 time	Bin(8)	

QTIME08	Wait bucket 08 time	Bin(8)	
QTIME09	Wait bucket 09 time	Bin(8)	
QTIME10	Wait bucket 10 time	Bin(8)	
QTIME11	Wait bucket 11 time	Bin(8)	
QTIME12	Wait bucket 12 time	Bin(8)	
QTIME13	Wait bucket 13 time	Bin(8)	
QTIME14	Wait bucket 14 time	Bin(8)	
QTIME15	Wait bucket 15 time	Bin(8)	
QTIME16	Wait bucket 16 time	Bin(8)	
QTIME17	Wait bucket 17 time	Bin(8)	
QTIME18	Wait bucket 18 time	Bin(8)	
QTIME19	Wait bucket 19 time	Bin(8)	
QTIME20	Wait bucket 20 time	Bin(8)	
QTIME21	Wait bucket 21 time	Bin(8)	
QTIME22	Wait bucket 22 time	Bin(8)	
QTIME23	Wait bucket 23 time	Bin(8)	
QTIME24	Wait bucket 24 time	Bin(8)	
QTIME25	Wait bucket 25 time	Bin(8)	
QTIME26	Wait bucket 26 time	Bin(8)	
QTIME27	Wait bucket 27 time	Bin(8)	
QTIME28	Wait bucket 28 time	Bin(8)	
QTIME29	Wait bucket 29 time	Bin(8)	
QTIME30	Wait bucket 30 time	Bin(8)	
QTIME31	Wait bucket 31 time	Bin(8)	
QTIME32	Wait bucket 32 time	Bin(8)	
CURRSTATE	Current or last state (run, CPQ, wait)	Char(4)	
BLOCKBCKT	Current or last blocking bucket	Bin(2)	
LICWO	Current or last LIC wait object	Char(4)	
BLOCKENUM	Current or last blocking enum	Bin(2)	
CURRWTDUR	Total time in current wait (in microsecs)	Bin(8)	

LICWOHNDL	Current or last LIC wait object handle	Hex(8)	
WOSEGTYP	Wait object segment type derived from wait obj addr	Hex(4)	
WOOBJTYP	Wait object object type derived from wait obj addr	Hex(4)	
WOOBJNAM	Wait object name derived from wait obj addr	Char(30)	
SQLINTHRD	SQL statement in progress	Char(1)	
TNXVLGEND	Very long transaction (> 10 secs) ended in interval	Char(1)	
STSCODE	STS file record code	Char(1)	
HTASKCNT	Holding thread/task task count	Bin(8)	
GPHTDEUS	Job elapsed interval time (usecs)	Bin(8)	
GPHTIME	Job interval end timestamp	Timestamp(26)	
GPHOFFSET	Job interval offset adjustment (usecs)	Bin(8)	In relation to GPHTIME when this TDE was sampled
TIME01A	Active CPU time (usecs)	Bin(8)	Dispatched to CPU and burning CPU
TIME01D	Dispatched CPU but waiting time (usecs)	Bin(8)	
TIME01T	Transferred CPU estimate (usecs)	Bin(8)	This is usually the CPU time reported from tasks for this interval. This number is not included in the CPU wait bucket QTIME01.
POOL	Pool ID	Bin(2)	

Table of Contents | Previous | Next

1.3.11 QAIDRJWIG3 - Summary file detail records #3

Description: This file contains the records needed to produce the graph in the GUI 'Run/wait time signature with sliced intervals'.

Collection: No but a member matching collection member is created when default summarization is ran from GUI.

Record: One record is created for each slice size (in microseconds) specified for a specific TDE.

Field Name	Field Description	Format	Comments
INTERVAL	Interval number	Binary(4)	
INTSLICE	Interval slice	Binary(4)	
TASKCOUNT	Task count	Binary(8)	
THREADID	Thread ID	Hex(8)	
TDEJOBNAME	Job/task name	Char(32)	
TDETYPE	Job or task type flag indicates LIC task or process	Char(1)	
PRIORITY	Current LIC priority	Binary(2)	
SLICETYPE	Slice type code	Char(1)	
SLICETIME	Elapsed microseconds for this slice	Binary(8)	
SLICETMSP	Slice end timestamp	Timestamp(26)	
SLICEPARM	Slice size parameter	Packed(5)	
BKT01A	Actual % CPU	Packed(4)	
BKT01B	Dispatched but waiting % CPU	Packed(4)	
BKT01C	Transferred % CPU	Packed(4)	
BKT01	Dispatched % CPU	Packed(4)	
BKT02	CPU queueing %	Packed(4)	

BKT03	Bucket 03 %	Packed(4)	
BKT04	Bucket 04 %	Packed(4)	
BKT05	Bucket 05 %	Packed(4)	
BKT06	Bucket 06 %	Packed(4)	
BKT07	Bucket 07 %	Packed(4)	
BKT08	Bucket 08 %	Packed(4)	
BKT09	Bucket 09 %	Packed(4)	
BKT10	Bucket 10 %	Packed(4)	
BKT11	Bucket 11 %	Packed(4)	
BKT12	Bucket 12 %	Packed(4)	
BKT13	Bucket 13 %	Packed(4)	
BKT14	Bucket 14 %	Packed(4)	
BKT15	Bucket 15 %	Packed(4)	
BKT16	Bucket 16 %	Packed(4)	
BKT17	Bucket 17 %	Packed(4)	
BKT18	Bucket 18 %	Packed(4)	
BKT19	Bucket 19 %	Packed(4)	
BKT20	Bucket 20 %	Packed(4)	
BKT21	Bucket 21 %	Packed(4)	
BKT22	Bucket 22 %	Packed(4)	
BKT23	Bucket 23 %	Packed(4)	
BKT24	Bucket 24 %	Packed(4)	
BKT25	Bucket 25 %	Packed(4)	
BKT26	Bucket 26 %	Packed(4)	
BKT27	Bucket 27 %	Packed(4)	
BKT28	Bucket 28 %	Packed(4)	
BKT29	Bucket 29 %	Packed(4)	
BKT30	Bucket 30 %	Packed(4)	
BKT31	Bucket 31 %	Packed(4)	
BKT32	Bucket 32 %	Packed(4)	
CURRSTATE	Current or last state (run, CPQ, wait)	Char(4)	
BLOCKBCKT	Current or last blocking bucket	Bin(2)	

LICWO	Current or last LIC wait object	Char(4)	
BLOCKENUM	Current or last blocking enum	Bin(2)	
CURRWTDUR	Total time in current wait (in microsecs)	Bin(8)	
LICWOHNDL	Current or last LIC wait object handle	Hex(8)	
WOSEGTYP	Wait object segment type derived from wait obj addr	Hex(4)	
WOOBJTYP	Wait object object type derived from wait obj addr	Hex(4)	
WOOBJNAM	Wait object name derived from wait obj addr	Char(30)	
SQLINTHRD	SQL statement in progress	Char(1)	
TNXVLGEND	Very long transaction (> 10 secs) ended in interval	Char(1)	
STSCODE	STS file record code	Char(1)	
HTASKCNT	Holding thread/task task count	Bin(8)	
GPHTDEUS	Job elapsed interval time (usecs)	Bin(8)	
GPHTIME	Job interval end timestamp	Timestamp(26)	
GPHOFFSET	Job interval offset adjustment (usecs)	Bin(8)	In relation to GPHTIME when this TDE was sampled

Table of Contents | Previous | Next

1.3.12 QAIDRJWIS0 - Summary control file

Description: This single member file contains information about each summary created in a library.

Collection: No

Record: One record is created for each summary created in the library.

Field Name	Field Description	Format	Comments
SRCCOL	Source collection name	Char(10)	
SRCLIB	Source collection library	Char(10)	
SRCVRM	Source VRM	Char(4)	
ANLSTS	Analysis status	Char(7)	
ANLNAME	Analysis name	Char(10)	
ANLLIB	Analysis library name	Char(10)	
ANLTYPE	Analysis type	Char(2)	
ANLDESC	Analysis description	Char(50)	
ANLTMSP	Analysis create timestamp	Timestamp(26)	
ANLCBYU	Analysis created by user profile	Char(10)	
ANLCBYJ	Analysis created by job name	Char(10)	
ANLCBYN	Aanlysis created by user name	Char(10)	
ANLCBY#	Analysis created by job number	Char(6)	

Table of Contents | Previous | Next

1.3.13 QAIDRJWIS2 - Interval summary file

Description: This file contains wait bucket summaries for each interval of the collections.

Collection: No but a member matching collection member is created when default summarization is ran from GUI.

Record: Two records produced each interval: one for the active TDEs and one for the idle TDEs.

Field Name	Field Description	Format	Comments
INTRCCD	Record code A = active TDE summary I = idle TDE summary	Char(1)	
INTBHHMM	Interval from HH.MM	Char(5)	
INTEHHMM	Interval to HH.MM	Char(5)	
INTERVALB	Interval from number	Binary(4)	
INTERVAL	Interval to number	Binary(4)	
INTRVLS	Intervals summed	Binary(4)	
INTBEGTMS	Interval snapshot start timestamp	Timestamp(26)	
INTENDTMS	Interval snapshot end timestamp	Timestamp(26)	
INTDELTA	Interval elapsed secs	Packed(8)	
INTPCTCPU	Interval active % CPU used	Packed(3)	
INTPCTSYS	Interval system % CPU used	Packed(3)	
INTAVGSNAP	Average JW snapshot time (secs)	Packed(8)	
INTAVGJNAP	Average JOBs snapshot time (usecs)	Packed(8)	

INTTOTACT	Number active tasks/threads	Packed(7)	
INTTOTIDL	Number idle tasks/threads	Packed(7)	
INTTOTBLKS	Number blocked at snapshot	Packed(7)	
INTTOTBLKT	Number blocked thru entire interval	Packed(7)	
INTTOTCPU	Interval CPU usecs	Packed(7)	
INTTOTCPU1	Interval CPU01 usecs	Packed(7)	
INTTOTIO	Interval Total IO requests	Packed(7)	
INTTOTSYN	Interval total sync IO requests	Packed(7)	
INTTOTASY	Interval total async IO requests	Packed(7)	
INTTOTRD	Interval total read requests	Packed(7)	
INTTOTWRT	Interval total write requests	Packed(7)	
INTTOTSDR	Interval total sync DB read requests	Packed(7)	
INTTOTSNR	Interval total sync NDB read requests	Packed(7)	
INTTOTSDW	Interval total sync DB write requests	Packed(7)	
INTTOTSNW	Interval total sync NDB write requests	Packed(7)	
INTTOTADR	Interval total async DB read requests	Packed(7)	
INTTOTANR	Interval total async NDB read requests	Packed(7)	
INTTOTADW	Interval total async DB write requests	Packed(7)	
INTTOTANW	Interval total async NDB write requests	Packed(7)	
INTTOTIOP	Interval total IOP requests	Packed(7)	
INTTOTWIO	Interval total WIO requests	Packed(7)	
INTTOTFLT	Interval total fault requests	Packed(7)	
INTTOTADP	Total allocated DASD pages	Packed(7)	
INTTOTDDP	Total deallocated DASD pages	Packed(7)	
INTTOTSZ	Interval total seize time (usecs)	Packed(7)	
INTTOTBINO	Interval total binary overflows	Packed(7)	
INTTOTDECO	Interval total decimal overflows	Packed(7)	

INTTOTFLOO	Interval total float overflows	Packed(7)	
INTTOTSTMR	Interval total stream file reads	Packed(7)	
INTTOTSTMW	Interval total stream file writes	Packed(7)	
INTTOTAW	Interval total active to wait transitions	Packed(7)	
INTTOTWI	Interval total wait to inactive transitions	Packed(7)	
INTTOTAI	Interval total active to ineligible transitions	Packed(7)	
INTJWINIT	Interval JW initiations	Packed(3)	
INTJBINIT	Interval job initiations	Packed(3)	
INTJBTERM	Interval job terminations	Packed(3)	
INTSTKCNT	Interval job stack counts	Packed(3)	
INTSQLCNT	Interval job SQL counts	Packed(3)	
INTDUMPREQ	DUMP requests member name	Char(3)	
INTDUMPR#	Number of DUMP requests	Packed(3)	
INTDUMPSEC	Estimated recorder coverage (secs)	Packed(3)	
INTRULEH	Interval satisfied rule hits	Packed(3)	
USRSC	User selected current wait	Char(4)	
INTUSCWH	Interval USCW hits	Packed(3)	
INTUSCWU	Interval USCW usecs	Packed(8)	
SNPUSECS	Usecs since IPL this snapshot was started	Packed(11)	
INTCPC	Current processor capacity	Packed(3)	
INTNUMPROC	Number of processors	Packed(3)	
INTPROCSHR	Processor sharing attribute	Char(1)	
INTSSTSHMS	QWCRSSTS elapsed HHMMSS	Char(6)	
INTSNAPPCU	Snapshot percent CPU used	Packed(3)	
INTJBIANT	Interval job initiations and terminations	Packed(3)	
INTTM01	Wait bucket total time 01 (usecs)	Packed(11)	
INTTM02	Wait bucket total time 02 (usecs)	Packed(11)	
INTTM03	Wait bucket total time 03 (usecs)	Packed(11)	

INTTM04	Wait bucket total time 04 (usecs)	Packed(11)	
INTTM05	Wait bucket total time 05 (usecs)	Packed(11)	
INTTM06	Wait bucket total time 06 (usecs)	Packed(11)	
INTTM07	Wait bucket total time 07 (usecs)	Packed(11)	
INTTM08	Wait bucket total time 08 (usecs)	Packed(11)	
INTTM09	Wait bucket total time 09 (usecs)	Packed(11)	
INTTM10	Wait bucket total time 10 (usecs)	Packed(11)	
INTTM11	Wait bucket total time 11 (usecs)	Packed(11)	
INTTM12	Wait bucket total time 12 (usecs)	Packed(11)	
INTTM13	Wait bucket total time 13 (usecs)	Packed(11)	
INTTM14	Wait bucket total time 14 (usecs)	Packed(11)	
INTTM15	Wait bucket total time 15 (usecs)	Packed(11)	
INTTM16	Wait bucket total time 16 (usecs)	Packed(11)	
INTTM17	Wait bucket total time 17 (usecs)	Packed(11)	
INTTM18	Wait bucket total time 18 (usecs)	Packed(11)	
INTTM19	Wait bucket total time 19 (usecs)	Packed(11)	
INTTM20	Wait bucket total time 20 (usecs)	Packed(11)	
INTTM21	Wait bucket total time 21 (usecs)	Packed(11)	
INTTM22	Wait bucket total time 22 (usecs)	Packed(11)	
INTTM23	Wait bucket total time 23 (usecs)	Packed(11)	
INTTM24	Wait bucket total time 24 (usecs)	Packed(11)	
INTTM25	Wait bucket total time 25 (usecs)	Packed(11)	
INTTM26	Wait bucket total time 26 (usecs)	Packed(11)	
INTTM27	Wait bucket total time 27 (usecs)	Packed(11)	
INTTM28	Wait bucket total time 28 (usecs)	Packed(11)	
INTTM29	Wait bucket total time 29 (usecs)	Packed(11)	
INTTM30	Wait bucket total time 30 (usecs)	Packed(11)	
INTTM31	Wait bucket total time 31 (usecs)	Packed(11)	
INTTM32	Wait bucket total time 32 (usecs)	Packed(11)	
INTCNT01	Wait bucket count 01	Binary(4)	
INTCNT02	Wait bucket count 02	Binary(4)	
INTCNT03	Wait bucket count 03	Binary(4)	

INTCNT04	Wait bucket count 04	Binary(4)	
INTCNT05	Wait bucket count 05	Binary(4)	
INTCNT06	Wait bucket count 06	Binary(4)	
INTCNT07	Wait bucket count 07	Binary(4)	
INTCNT08	Wait bucket count 08	Binary(4)	
INTCNT09	Wait bucket count 09	Binary(4)	
INTCNT10	Wait bucket count 10	Binary(4)	
INTCNT11	Wait bucket count 11	Binary(4)	
INTCNT12	Wait bucket count 12	Binary(4)	
INTCNT13	Wait bucket count 13	Binary(4)	
INTCNT14	Wait bucket count 14	Binary(4)	
INTCNT15	Wait bucket count 15	Binary(4)	
INTCNT16	Wait bucket count 16	Binary(4)	
INTCNT17	Wait bucket count 17	Binary(4)	
INTCNT18	Wait bucket count 18	Binary(4)	
INTCNT19	Wait bucket count 19	Binary(4)	
INTCNT20	Wait bucket count 20	Binary(4)	
INTCNT21	Wait bucket count 21	Binary(4)	
INTCNT22	Wait bucket count 22	Binary(4)	
INTCNT23	Wait bucket count 23	Binary(4)	
INTCNT24	Wait bucket count 24	Binary(4)	
INTCNT25	Wait bucket count 25	Binary(4)	
INTCNT26	Wait bucket count 26	Binary(4)	
INTCNT27	Wait bucket count 27	Binary(4)	
INTCNT28	Wait bucket count 28	Binary(4)	
INTCNT29	Wait bucket count 29	Binary(4)	
INTCNT30	Wait bucket count 30	Binary(4)	
INTCNT31	Wait bucket count 31	Binary(4)	
INTCNT32	Wait bucket count 32	Binary(4)	
INTJBS01	Wait bucket jobs 01	Binary(4)	
INTJBS02	Wait bucket jobs 02	Binary(4)	
INTJBS03	Wait bucket jobs 03	Binary(4)	

INTJBS04	Wait bucket jobs 04	Binary(4)	
INTJBS05	Wait bucket jobs 05	Binary(4)	
INTJBS06	Wait bucket jobs 06	Binary(4)	
INTJBS07	Wait bucket jobs 07	Binary(4)	
INTJBS08	Wait bucket jobs 08	Binary(4)	
INTJBS09	Wait bucket jobs 09	Binary(4)	
INTJBS10	Wait bucket jobs 10	Binary(4)	
INTJBS11	Wait bucket jobs 11	Binary(4)	
INTJBS12	Wait bucket jobs 12	Binary(4)	
INTJBS13	Wait bucket jobs 13	Binary(4)	
INTJBS14	Wait bucket jobs 14	Binary(4)	
INTJBS15	Wait bucket jobs 15	Binary(4)	
INTJBS16	Wait bucket jobs 16	Binary(4)	
INTJBS17	Wait bucket jobs 17	Binary(4)	
INTJBS18	Wait bucket jobs 18	Binary(4)	
INTJBS19	Wait bucket jobs 19	Binary(4)	
INTJBS20	Wait bucket jobs 20	Binary(4)	
INTJBS21	Wait bucket jobs 21	Binary(4)	
INTJBS22	Wait bucket jobs 22	Binary(4)	
INTJBS23	Wait bucket jobs 23	Binary(4)	
INTJBS24	Wait bucket jobs 24	Binary(4)	
INTJBS25	Wait bucket jobs 25	Binary(4)	
INTJBS26	Wait bucket jobs 26	Binary(4)	
INTJBS27	Wait bucket jobs 27	Binary(4)	
INTJBS28	Wait bucket jobs 28	Binary(4)	
INTJBS29	Wait bucket jobs 29	Binary(4)	
INTJBS30	Wait bucket jobs 30	Binary(4)	
INTJBS31	Wait bucket jobs 31	Binary(4)	
INTJBS32	Wait bucket jobs 32	Binary(4)	
INTTXTM	Display transaction time (usecs)	Packed(11)	
INTTXCNT	Display transaction count	Binary(4)	
INTTXJBS	Display transaction job count	Binary(4)	

INTVLTXJBS	Very long transaction ended (> 10 secs) job count	Binary(4)	
INTSQLACNT	Active SQL statements job count	Binary(4)	
INTBEGEST	Estimated start of interval timestamp	Timestamp(26)	
INTSIPLUS	Estimated usecs since IPL for interval start	Binary(8)	

Table of Contents | Previous | Next

1.3.14 QAIDRJWIX1 - Summary transactions file

Description: This file contains information about long SQL and display transactions that occurred within the collection.

Collection: No but a member matching collection member is created when default summarization is ran from GUI.

Record: One record is produced for each long transaction (> 1 sec) detected by Job Watcher

Field Name	Field Description	Format	Comments
TXCODE	Transaction record code	Char(6)	SQLtxn or DSPtxn
TXPFLAG	Transaction partial flag	Char(6)	
TXBEGINT	Transaction begin interval number	Binary(4)	
TXENDINT	Transaction end interval number	Binary(4)	
TXTIME	Transaction response time (usecs)	Binary(8)	
TXCOUNT	Transaction count	Binary(4)	
TASKCOUNT	Task count	Binary(8)	
THREADID	Thread ID	Hex(8)	
TDEJOBNAME	Job/task name	Char(32)	
CURRUP	Current user profile	Char(10)	
PRIORITY	Current LIC priority	Binary(2)	
THREADPRI	Current XPF priority	Binary(2)	
PRICHG	Priority changed flag	Char(1)	Y or N
POOL	Pool ID	Binary(2)	
POOLCHG	Pool changed flag	Char(1)	Y or N

JOBSBS	Job subsystem	Char(10)	
JOBTYPE	Job type	Char(1)	
JOBFNCTN	Job function	Char(14)	
JOBSTATUS	Job status	Char(4)	
STARTOD	Time of day at snapshot start	Timestamp(26)	
STARTUSECS	Microseconds since IPL at snapshot start	Binary(8)	

Table of Contents | Previous | Next

1.3.15 QAIDRJWM1 - Monitor control file

Description: This file contains information about the monitors created on the system. This information about the monitors is stored in this file in library QGPL.

Collection: No

Record: One record is produced for each monitor created on a system.

Field Name	Field Description	Format	Comments
MONITOR	Monitor name	Char(8)	8 char prefix of collection names produced by monitor
LIB	Monitor library	Char(10)	
DFNNAME	Definition name	Char(10)	
COLNS	Number of collections	Binary(4)	
STRGAP	Collection start gap (in minutes)	Binary(4)	
STATUS	Status	Char(1)	
STARTTIME	Start time	Timestamp(26)	
COLLJOB	Monitor job	Char(26)	
DESC	Description	Char(50)	
ACTIVEIDX	Active collection index	Char(2)	Identifies the current or last collection processed
STROVRLAP	Collection overlap	Binary(4)	Number of seconds to wait before ending a collection after starting a new one

Table of Contents | Previous | Next

1.3.16 QAIDRJWIX1 - Unique taskcount list

Description: This file contains a list of taskcounts in the collection and the job/thread ID or task name for it.

Collection: No but created with the same member name as the collection name during analysis by the GUI or when running the CRTWCHSUM command.

Record: One record is produced for each taskcount found in the collection.

Field Name	Field Description	Format	Comments
TASKCOUNT	Task count	Binary(8)	
TDEJOBNAME	Job/task name	Char(32)	
THREADID	Thread ID	Hex(8)	

Table of Contents | Previous | Next

1.4 Commands

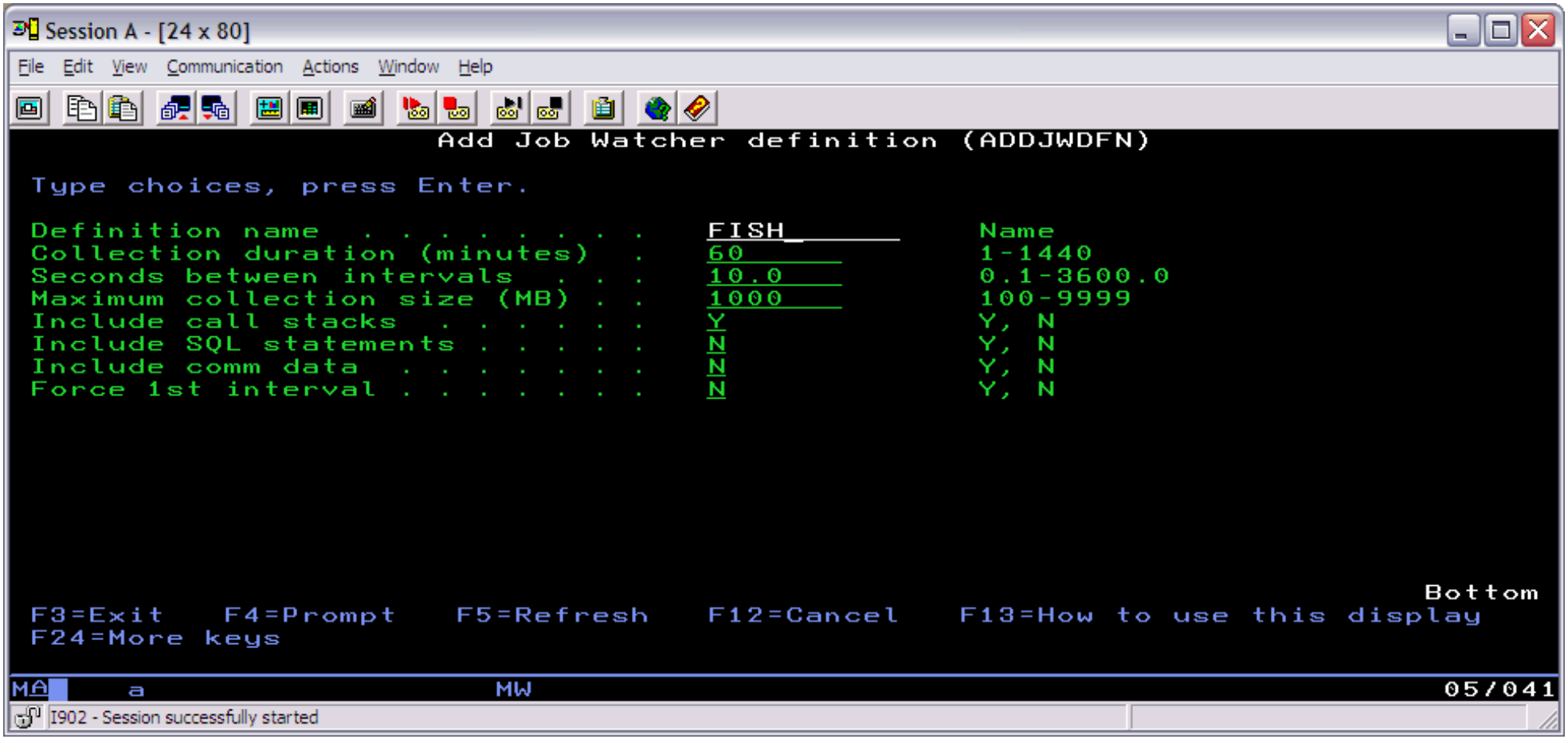
This section discusses the green screen commands that are shipped with Job Watcher in library QIDRWCH.

Table of Contents | Previous | Next

1.4.1 ADDJWDFN

This command is used to create a Job Watcher definition. A Job Watcher definition is the WCHJOB command string used to create a Job Watch or a Job Watcher Monitor.

ADDJWDFN provides the most commonly used options on the WCHJOB command but not all of those options. Use the GUI Start Job Watch Wizard to create a Job Watcher definition that handles parameters not available with this command. Definitions are stored in library QGPL in file QAIDRJWDFN



ADDJWDFN Parameters

DFNNAME - Definition name
The name of the definition to create.

COLDUR - Collection duration (minutes)
This parameter indicates how long the collection should run (in minutes).

INTDUR - Seconds between intervals
The interval duration of the collection in seconds.

MAXCOLMB - Maximum collection size (MB)
The maximum size the collection can reach. If the total size of all collection files exceeds this value, the collection will end.

STACKS - Include call stacks
Y or N if call stacks should be included in the collection.

SQL - Include SQL statements
Y or N if SQL statements and their host variables should be included in the collection.

COMM - Include communications data
Y or N if comm data (sockets) should be included in the collection.

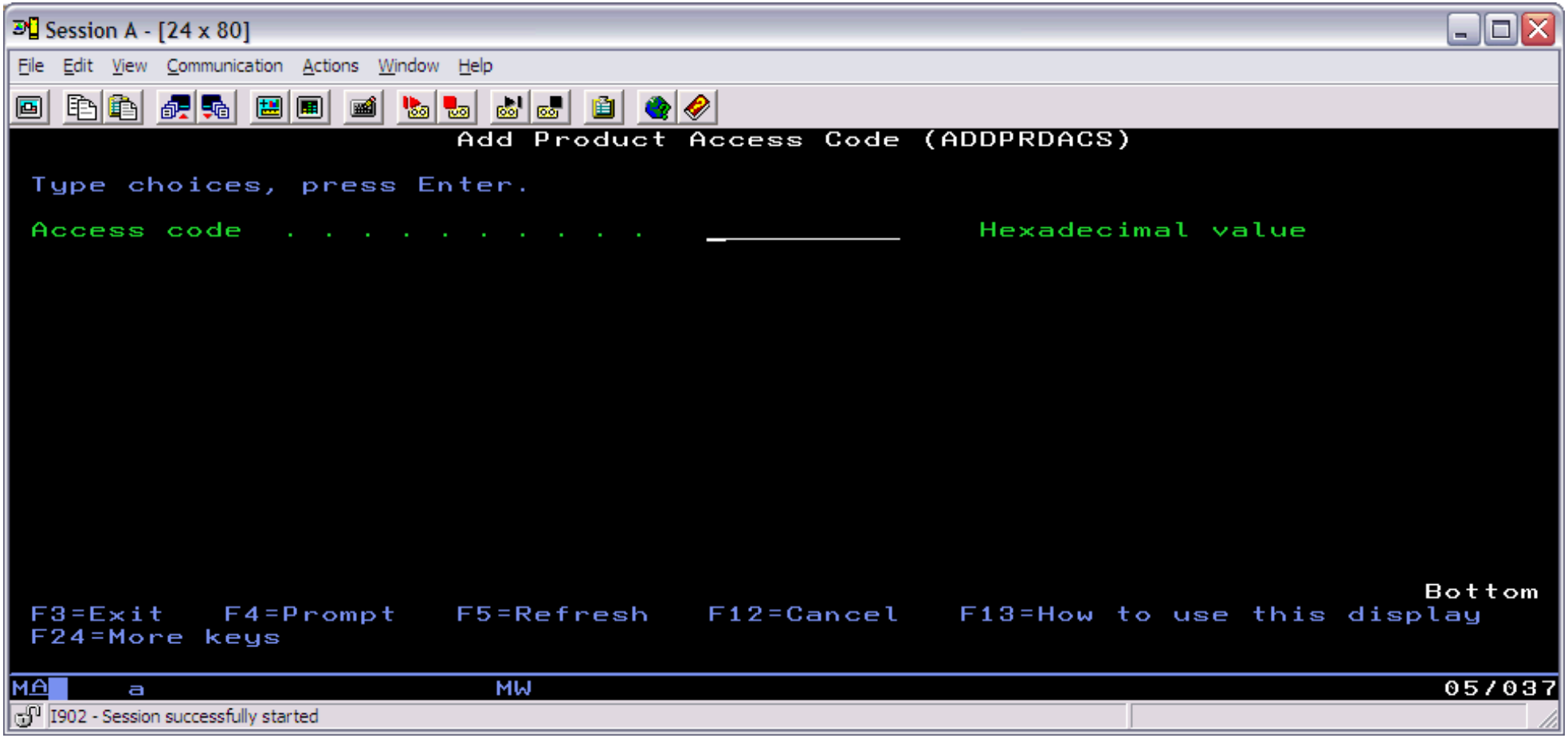
FORCE1ST - Force 1st interval
Y or N if the collection should collect information about ALL jobs and tasks on the system on the 1st interval. If your system has many thousands of jobs this option may greatly slow down the initialization of the job watch especially if other options such as to include SQL statements and communications data are selected.

Table of Contents | Previous | Next

1.4.2 ADDPRDACS

This command is used to authorize users of the system to an iDoctor component.

Access codes are granted through the website which this product was downloaded from: http://www-912.ibm.com/i_dir/idoctor.nsf



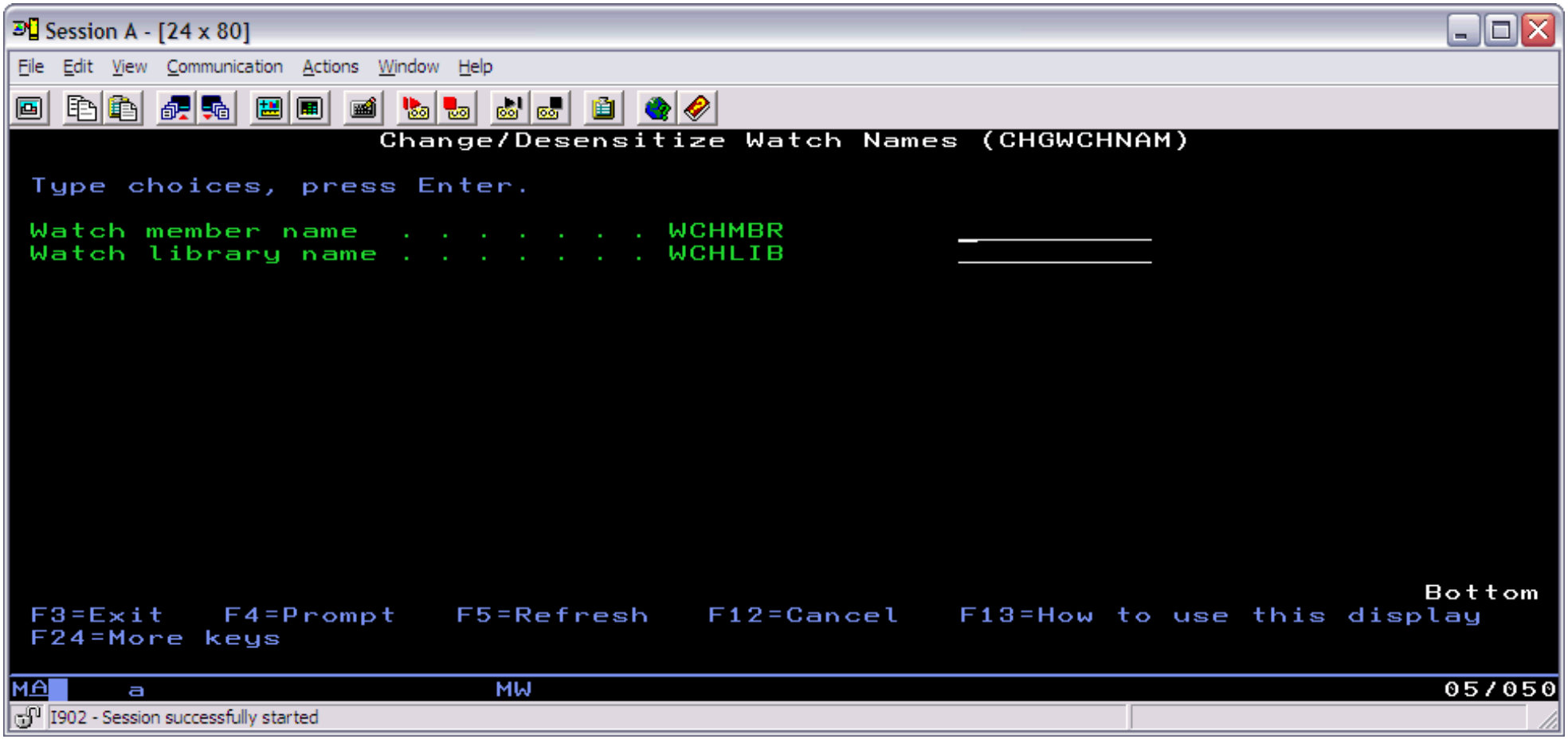
ADDPRDACS Parameters

ACSCDE - Access code
The access code provided by IBM to grant use of an iDoctor component on the current system.

Table of Contents | Previous | Next

1.4.3 CHGWCHNAM

This command is used mainly by service to remove sensitive names like program names, system names and user profile names from collection data before using it in presentations.
It can also be used by customers to achieve the same purpose. This file changes sensitive names to numbers and stores a mapping of its changes in a new file QAIDRJWNM1 with a member name matching the collection name.



CHGWCHNAM Parameters

WCHMBR - Watch member name
The name of the collection (member name) to have any customer sensitive names found within it changed.

WCHLIB - Watch library name

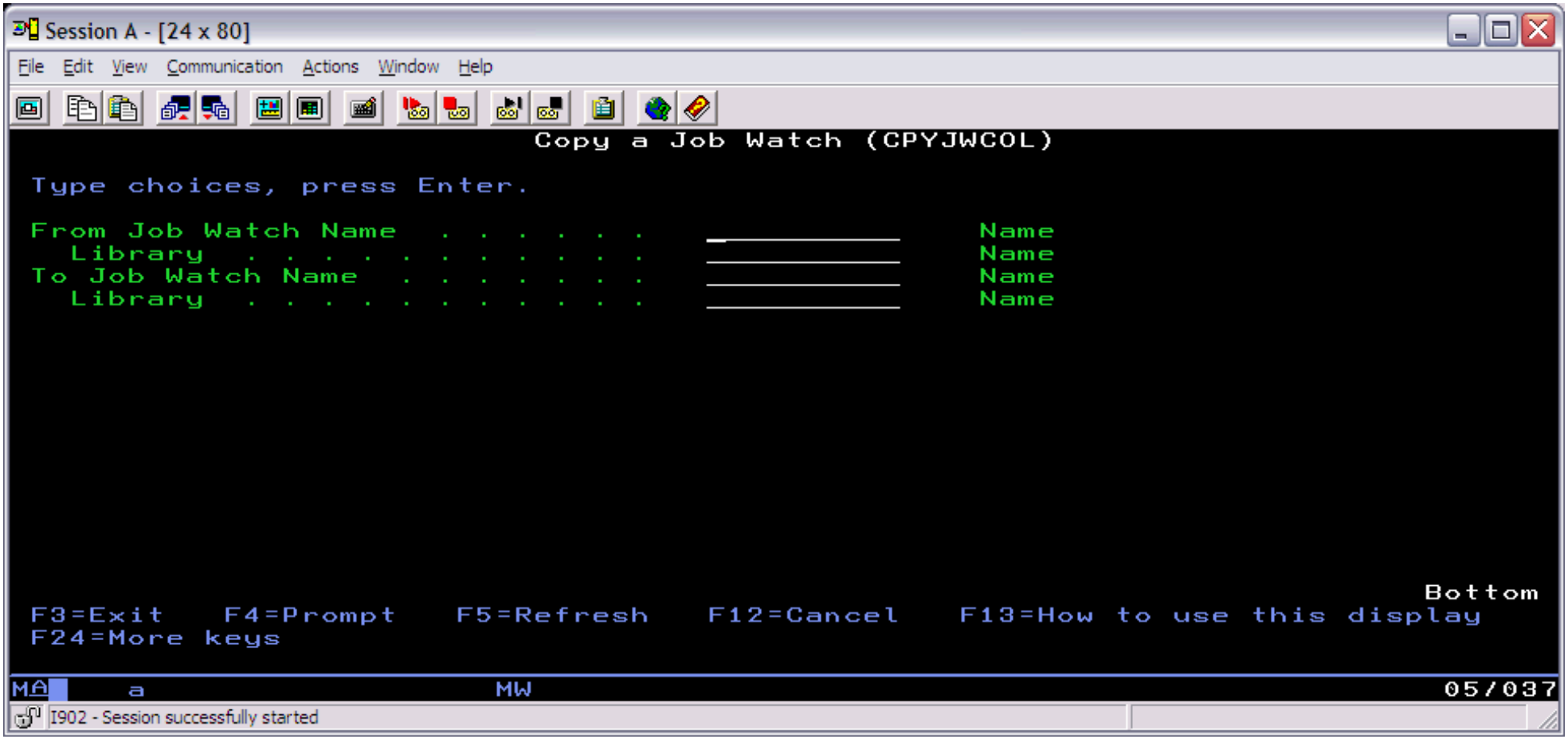
The name of the library of the collection to change.

Table of Contents | Previous | Next

1.4.4 CPYJWCOL

This command is used to copy a job watch. This action will copy every member matching the job watch name from the "from" library to the "to" library. If desired the job watch can be renamed by setting the TOCOL parameter with a new name.

An example of this command is:



CPYJWCOL Parameters

FROMCOL - From Job Watch Name
The name of the job watch to copy.

FROMLIB - From Job Watch Library Name
The library name of the job watch to copy.

TOCOL - To Job Watch Name
The name to give the job watch when copied to its new library.

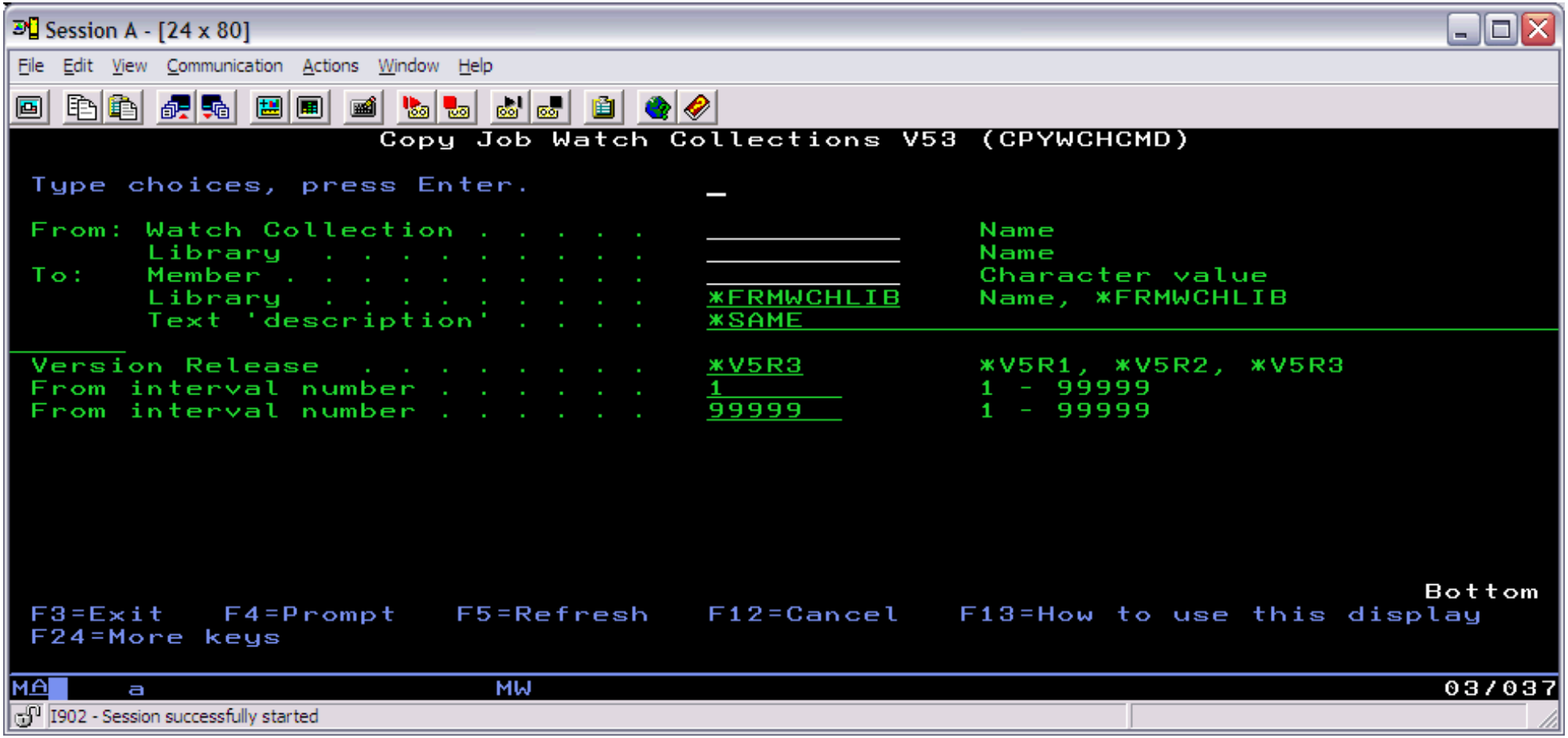
TOLIB - To Job Watch Library Name
The name of the library to copy the job watch to. The library may be the same as the FROMLIB parameter as long as the TOCOL parameter differs from the FROMCOL parameter.

Table of Contents | Previous | Next

1.4.5 CPYWCHCMD

The Copy Watch command copies a range of intervals from one collection into another new, user-defined collection. This command will copy the selected range of intervals for all related files. If the new collection already exists, the data will be replaced with the current copy. Copying a portion of a large collection to another named collection can reduce interactive query and analysis time as well as transport time if required. The selected intervals could be used to focus on the events leading up to a problem.

An example of this command is:



CPYWCHCMD Parameters

FRMWCHMBR - From watch collection

The name of the Job Watch collection used as input. This will also be the member name of the data base files in the Job Watch library.

FRMWCHLIB - From library
The name of the input library for the Job Watch collection data.

TOWCHMBR - To watch collection
The member name for the new collection. If the member already exists, the existing data will be replaced.

TOWCHLIB - To watch library
The name of the collection library. The library must already exist.

TOWCHDESC - To watch description
The description of the collection.

VRM - Version Release
The version and release of the collection.

FRMINTRVL - From interval number
Copy intervals starting with this interval.

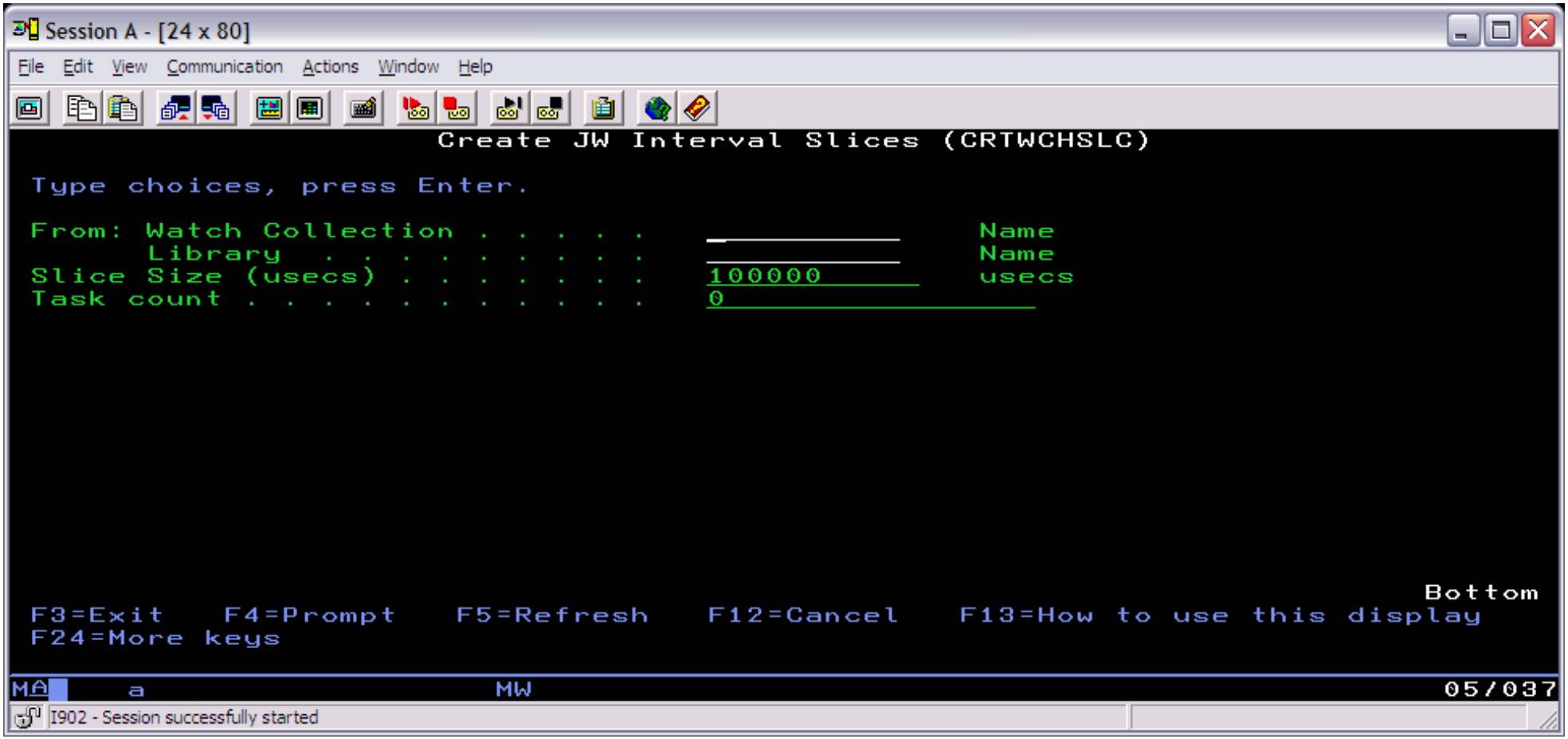
TOINTRVL - To interval number
Copy intervals up to and including this interval.

Table of Contents | Previous | Next

1.4.6 CRTWCHSLC

This command is used to create a file to produce a graph showing the percentage of wait times for a job or task seperated by small time slices over time. The time slice is usually much less than a second, the default is 1/10th of a second. The file created by this command is QAIDRJWIG3.

This option is primarly intended for service use.



CRTWCHSLC Parameter

FRMWCHMBR - From collection name
The name of the collection (member name) to create a time slice report over.

FRMWCHLIB - From collection library
The name of the library of the collection to create a time slice report over.

SLICESIZ - Slice size (usecs)
The amount of time that each record of the report should represent. This value must be in microseconds.

TASKCOUNT - Task count
The taskcount of the job or task to create the time slice report over.

Table of Contents | Previous | Next

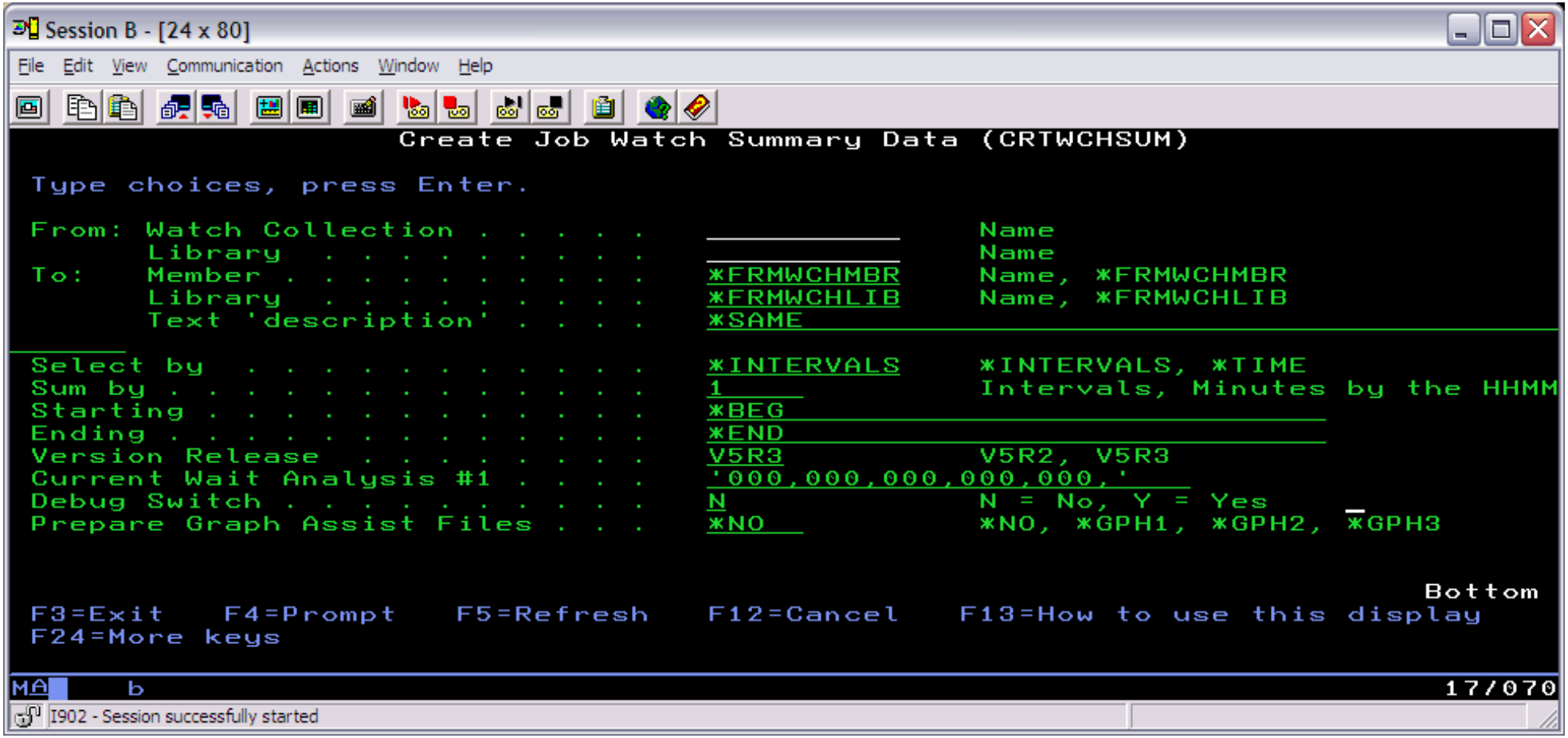
1.4.7 CRTWCHSUM

The Create Watch Summary command summarizes Job Watcher collection data in order to speed up the execution of many of the reports in the GUI and to fill in wait times contributing from idle jobs and tasks.

The default summarization of a collection in the GUI will summarize a collection for every interval but this command also supports the ability to summarize over a range of intervals so that 1 record is produced for every X intervals of data in the original collection.

This command creates several files including:

QAIDRJWIG2 - similiar to the QAPYJWTDE file, provides TDE level detail at the interval level.
QAIDRJWIS0 - control file that identifies the summarizations that have been run in library
QAIDRJWIS2 - interval summary file. Adds up numbers for wait buckets across all jobs in the collection or specified interval range.
QAIDRJWIX1 - long transactions file. Provides information about transactions that took longer than 1 second to complete.



CRTWCHSUM Parameters

FRMWCHMBR - From collection name
The name of the collection (member name) to summarize.

FRMWCHLIB - From library name
The name of the library of the collection to summarize.

TOWCHMBR - To collection name
The member name to use when creating the QAIDRJW* output files. If the member already exists, it will be replaced.

TOWCHLIB - To library name
The name of the output library. The library must already exist.

TOWCHDESC - To text description
Description of the collection.

SELECTBY - Select by

This parameter indicates how the collection will be summarized. It must be set to *INTERVALS. *TIME is not supported.

SUMBY - Sum by
Indicates how many intervals should be summarized per output record.

START - Starting
Indicates the interval number where the summarization should begin. *BEG indicates that the summariation should start at the beginning of the collection.

END - Ending
Indicates the interval number where the summarization should end. *END indicates that the summariation should end at the end of the collection.

VRM - Version release
Indicates the version and release of the collection to be summarized. By default this will match the version of Job Watcher installed on the system.

CURWT1 - Current wait analysis #1
This parameter is for service use only.

DEBUG - Debug switch
This parameter is for service use only.

GPHPREP - Prepare graph assist files
This parameter identifies if file QAIDRJWIG2 should be created.

*NO indicates that the QAIDRJWIG2 file should not be created.

*GPH1 not supported

*GPH2 indicates that the QAIDRJWIG2 file should be created with only data for jobs in waits that are considered 'bad'.

*GPH3 indicates that the QAIDRJWIG2 file should be created with all job data included for active and idle intervals.

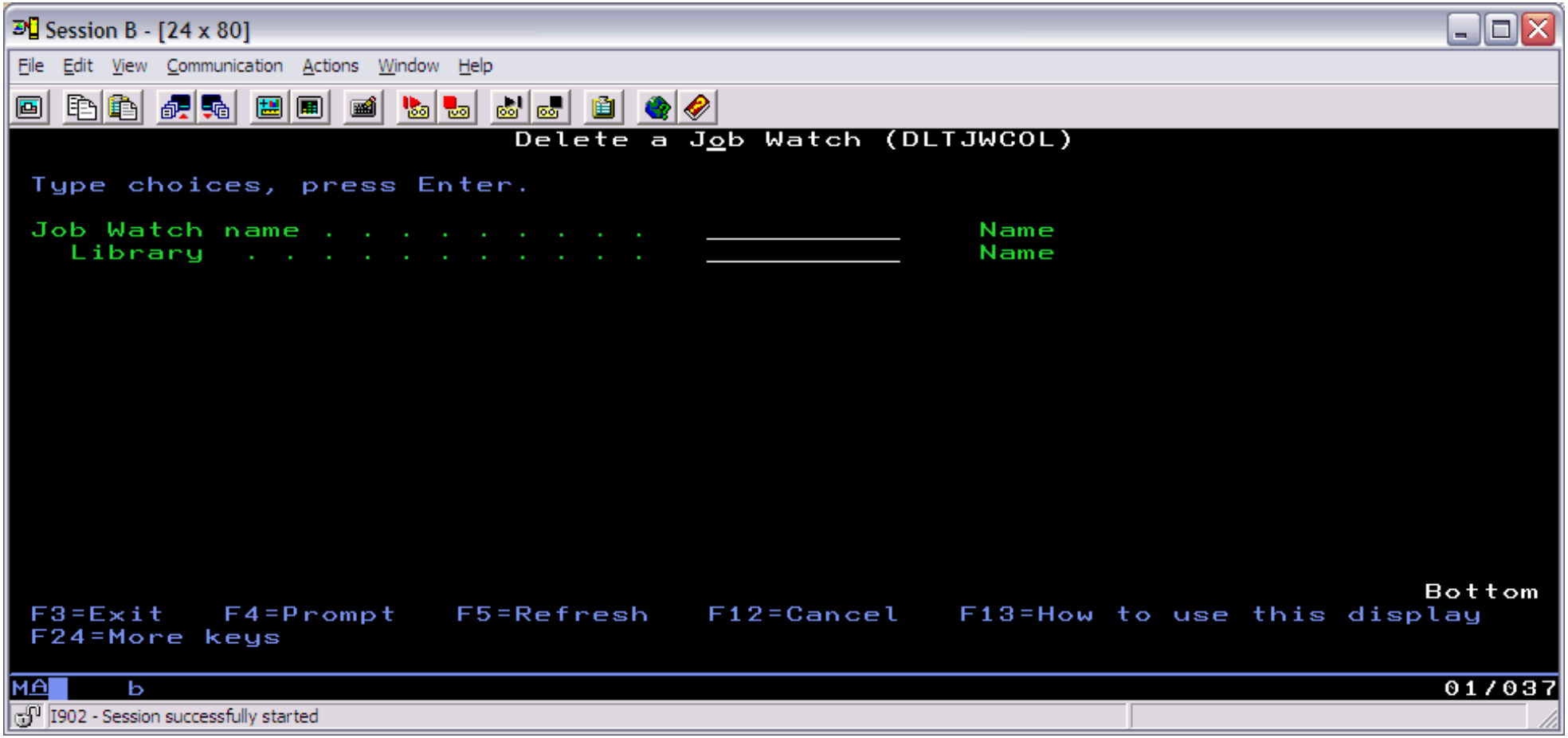
Table of Contents | Previous | Next

1.4.8 DLTJWCOL

This command is used to delete a job watch from a user's library on a system. This action will remove all members matching the job watch name from every QAPYJW* files found in the library specified.

The command also will remove the record in file QAIDRJWRI where the MBR field value equals the job watch name. This single member file is used by the GUI to keep track of the job watches that have been created in the library.

An example of this command is:



DLTJWCOL Parameters

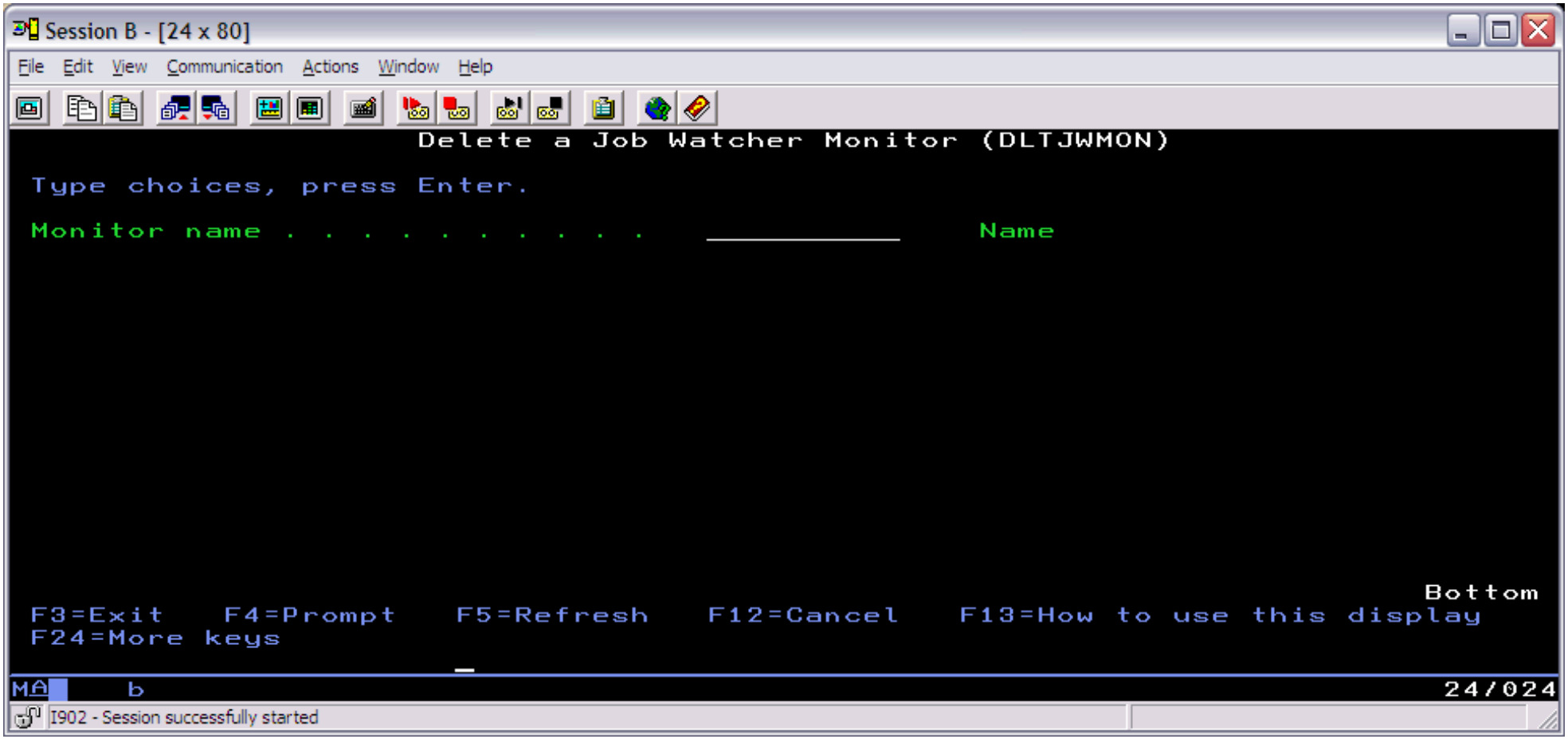
WCHNAME - Job Watch name
The name of the Job Watch to delete. This value equals the member name used in the Job Watcher output files.

LIB - Job Watch library name
The library name where the job watch will be deleted from.

Table of Contents | Previous | Next

1.4.9 DLTJWMON

This command is used to delete a job watcher monitor and all collections it contains on the current system. If the monitor is still running, the monitor will first be ended using the ENDJWMON command.
The record in file QGPL/QAIDRJWM1 that identifies the existance of this monitor will also be removed by this command.



DLTJWMON Parameters

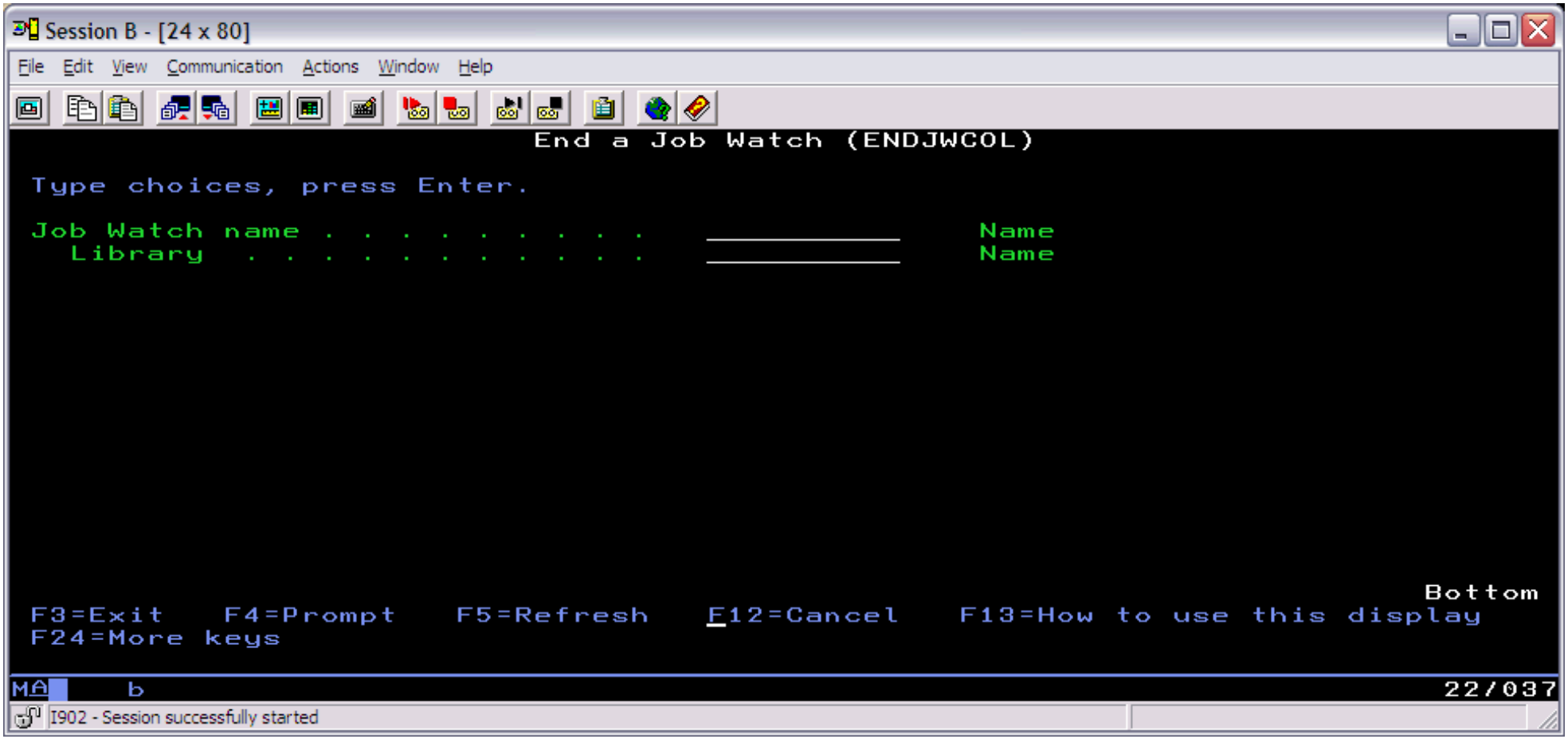
MONITOR - Monitor name
The name of the monitor to delete.

Table of Contents | Previous | Next

1.4.10 ENDJWCOL

This command is used to immediately end the job running a job watch (collection) on the current system.

An example of this command is:



ENDJWCOL Parameters

WCHNAME - Job watch name
The name of the Job Watch to end. This value is the member name used in the Job Watcher output files.

LIB - Library name

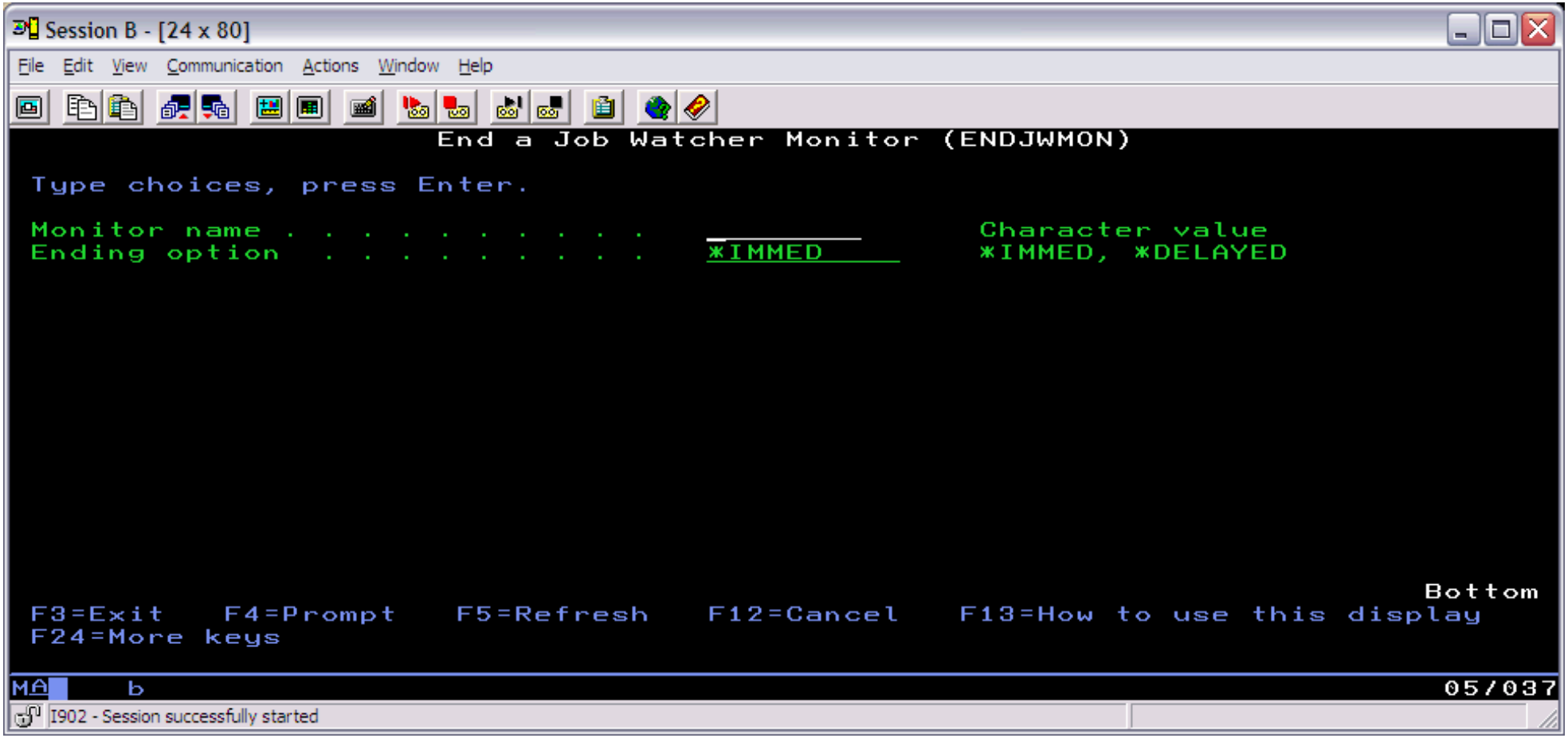
1.4.10 ENDJWCOL

The library name of the Job Watch to end.

Table of Contents | Previous | Next

1.4.11 ENDJWMON

This command is used to end the job running a Job Watcher monitor and any jobs currently running collections within the monitor.



ENDJWMON Parameters

MONITOR - Monitor name
The name of the monitor to end.

OPTION - Ending option
This parameter determines if the monitor and any active collection jobs should end immediately or after the currently running collection completes.

Use the *DELAYED option to indicate the monitor should remain active and then end once the current collection completes.

Table of Contents | Previous | Next

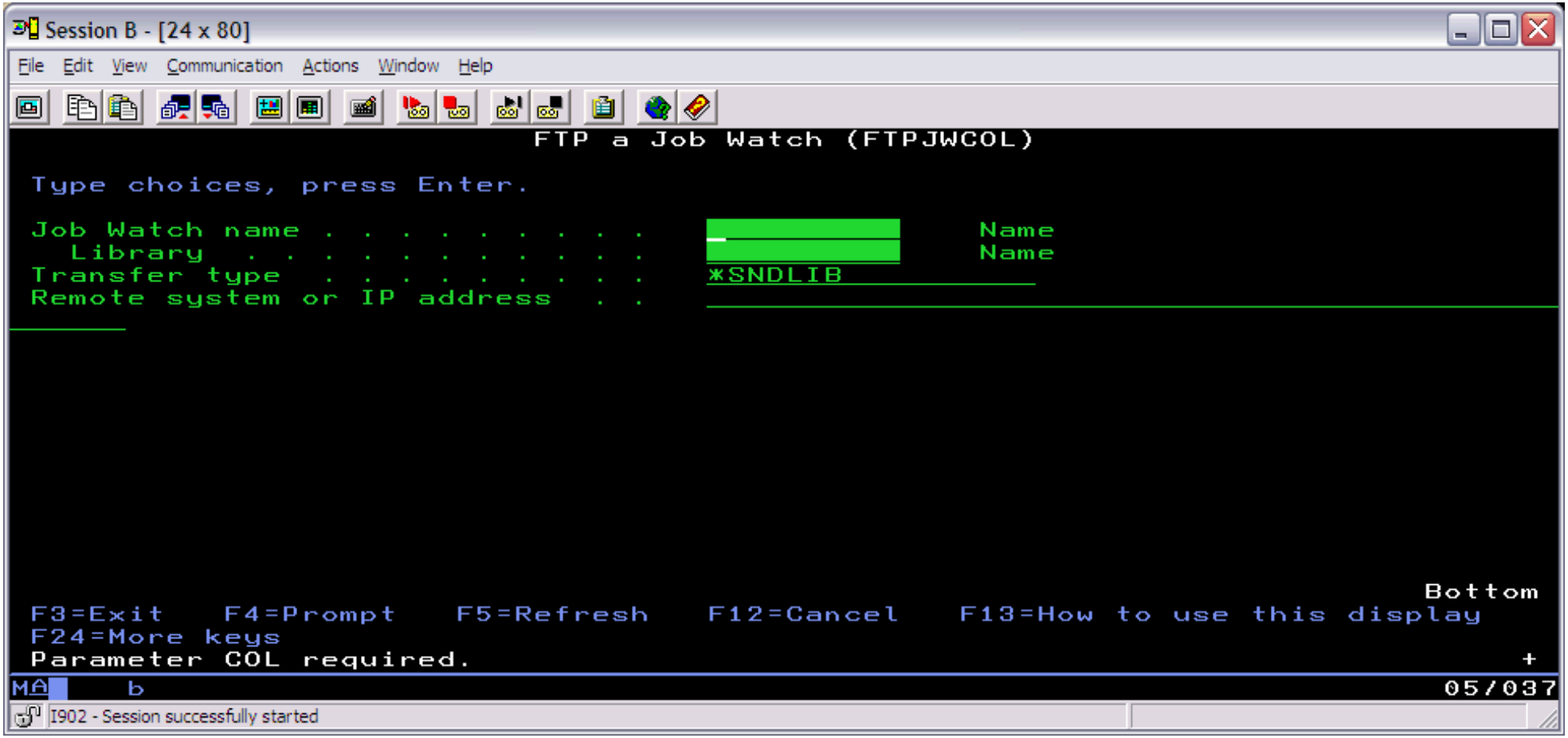
1.4.12 FTPJWCOL

This command is used to send a job watch to another system. To perform this action, the name of the job watch to be transferred must be provided along with information about the remote system including the system name, library and file name to send the saved job watch to.

This command will first save the collection using command SAVJWCOL into a save file and then utilize command FTPFILE in library QIDRGUI to perform the actual FTP. The collection is not restored by this command. This must be done on the remote system by using command RSTJWCOL.

If a problem occurs while attempting to FTP the file to the remote system a log file showing the content of the FTP session is saved to the file FTPLOG in library QIDRGUI.

An example of this command is:



FTPJWCOL Parameters

COL - Job Watch name
The name of the job watch to send to the remote system.

LIB - Job Watch library name
The library name of the job watch to send to the remote system.

ACTION - Transfer type
Indicates the type of remote system: whether the data is being sent to an iSeries library or an FTP directory.

*SNDLIB indicates that the collection is being sent to an iSeries library.
*SNDFTPDIR indicates that the collection is being sent to an FTPdirectory.

RMTSYSTEM - Remote system
The name of the remote system to transfer the collection too.

RMTLIB - Remote library
The name of the library on the remote system to transfer the save file containing the collection to.

RMTDIR - Remote directory
The full path of the FTP directory the collection should be sent to. The directory must already exist.

RMTFILE - Remote file
The name of the save file to create on the remote system containing the collection.

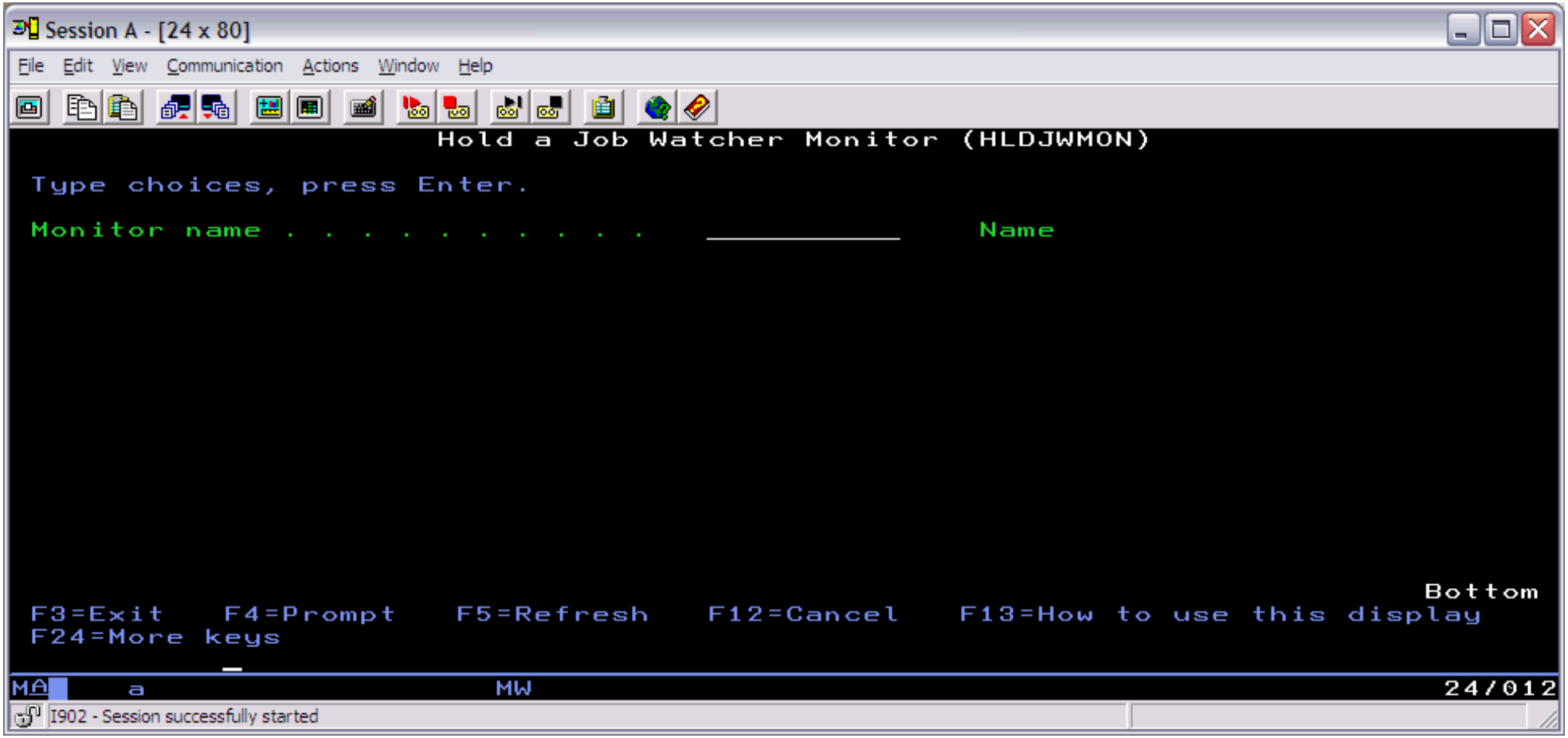
USR - User name for remote system
The name of the user profile to use when making the connection to the remote system.

PWD - Password for remote system
The password to use along with the user profile when making the connection to the remote system.

Table of Contents | Previous | Next

1.4.13 HLDJWMON

This command is used to hold a job watcher monitor. A held monitor will not delete any old collections or create any new collections until it is released using the RLSJWMON command.



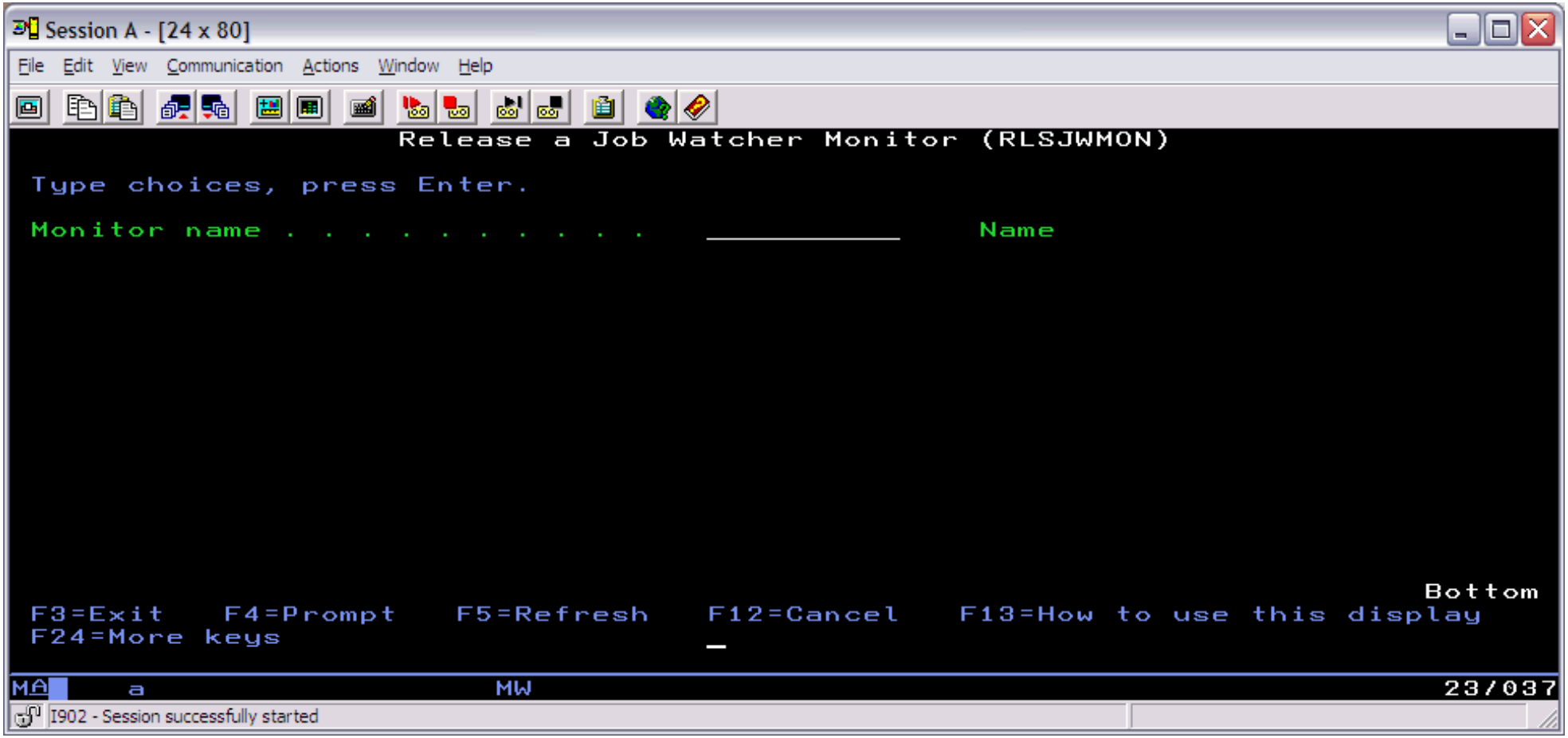
HLDJWMON Parameters

MONITOR - Monitor name
The name of the monitor to hold.

Table of Contents | Previous | Next

1.4.14 RLSJWMON

This command is used to release a monitor that is currently in a held state. Once released the monitor will continue to create new collections and delete old collections normally.



RLSJWMON Parameters

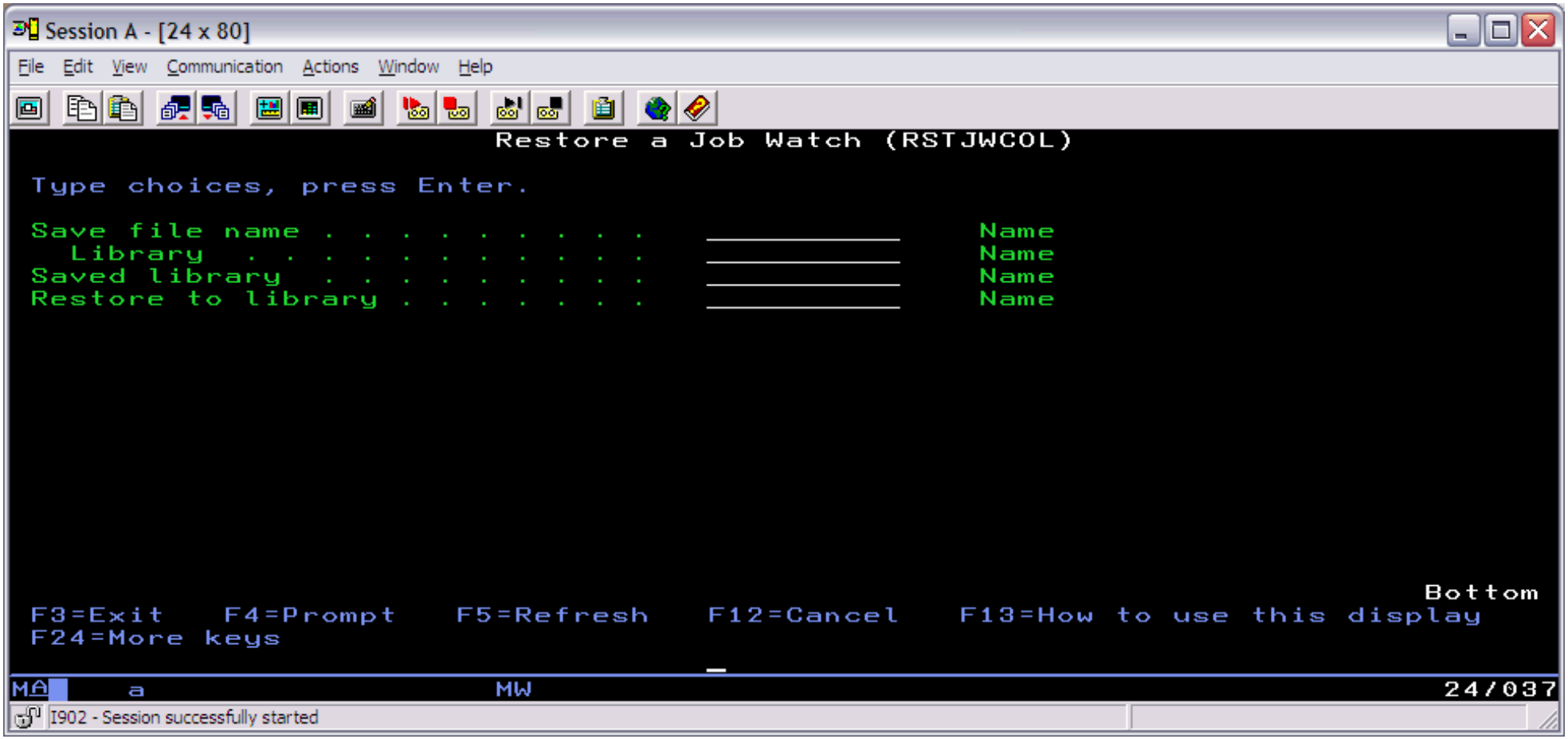
MONITOR - Monitor name
The name of the monitor to release.

Table of Contents | Previous | Next

1.4.15 RSTJWCOL

This command is used to restore a job watch from a save file created by the command SAVJWCOL or indirectly using the command FTPJWCOL. The save file being restored must have been saved over a collection created on a system running i5/OS V5R3.

An example of this command is:



RSTJWCOL Parameters

SAVFLIB - Save file library
The library name of the save file to restore.

SAVFNAME - Save file name
The name of the save file to restore.

SAVEDLIB - Saved library
The name of the library saved within the save file being restored. Use the DSPSAVF command on the save file being restored to determine this library name.

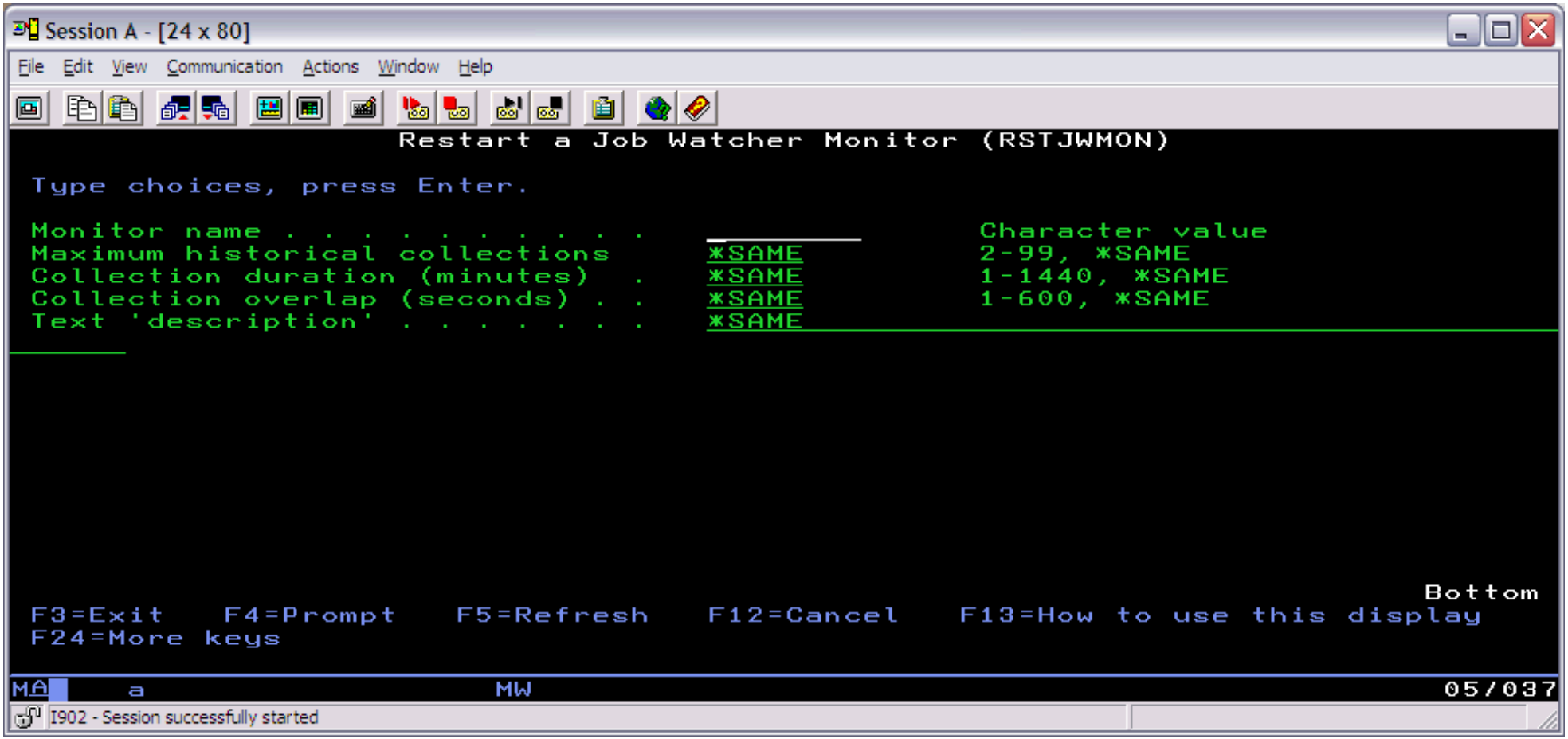
TOLIB - Restore to library
The name of the library to restore the job watch to.

Table of Contents | Previous | Next

1.4.16 RSTJWMON

This command is used to restart a Job Watcher monitor. The monitor must already exist and the definition used to create the monitor must still exist in the definition file QAIDRJWDFN in library QGPL.

See the STRJWMON command for more information on Job Watcher monitors.



RSTJWMON Parameters

MONITOR - Monitor name
The name of the monitor to restart. This can be up to 8 characters long. The collection names are created using the monitor name with a number between 01 and 99 appended to the end of the monitor name.

COLNS - Maximum historical collections
The maximum number of collections to keep in the monitor library at one time. If the number of collections is exceeded upon creation of a new collection the oldest collection in the monitor will be deleted. This value must be between 2 and 99.

STRGAP - Collection duration (minutes)
This value indicates how much time should elapse between each collection being submitted by the monitor. This value is in minutes. Because of the large amounts of data collected by Job Watcher it is often better to collect smaller amounts of data per collection. This helps the queries running over the collection return their results faster.

Keeping this number less than or equal to 60 minutes is recommended.

OVRLAP - Collection overlap (seconds)
This value indicates how much time should elapse between starting the the next collection and ending the job that was running the previous collection. On busy systems it may take a minute or two for a Job Watch to initialize and begin collecting data. This parameter allows for continuous collection to occur during this initialization time.

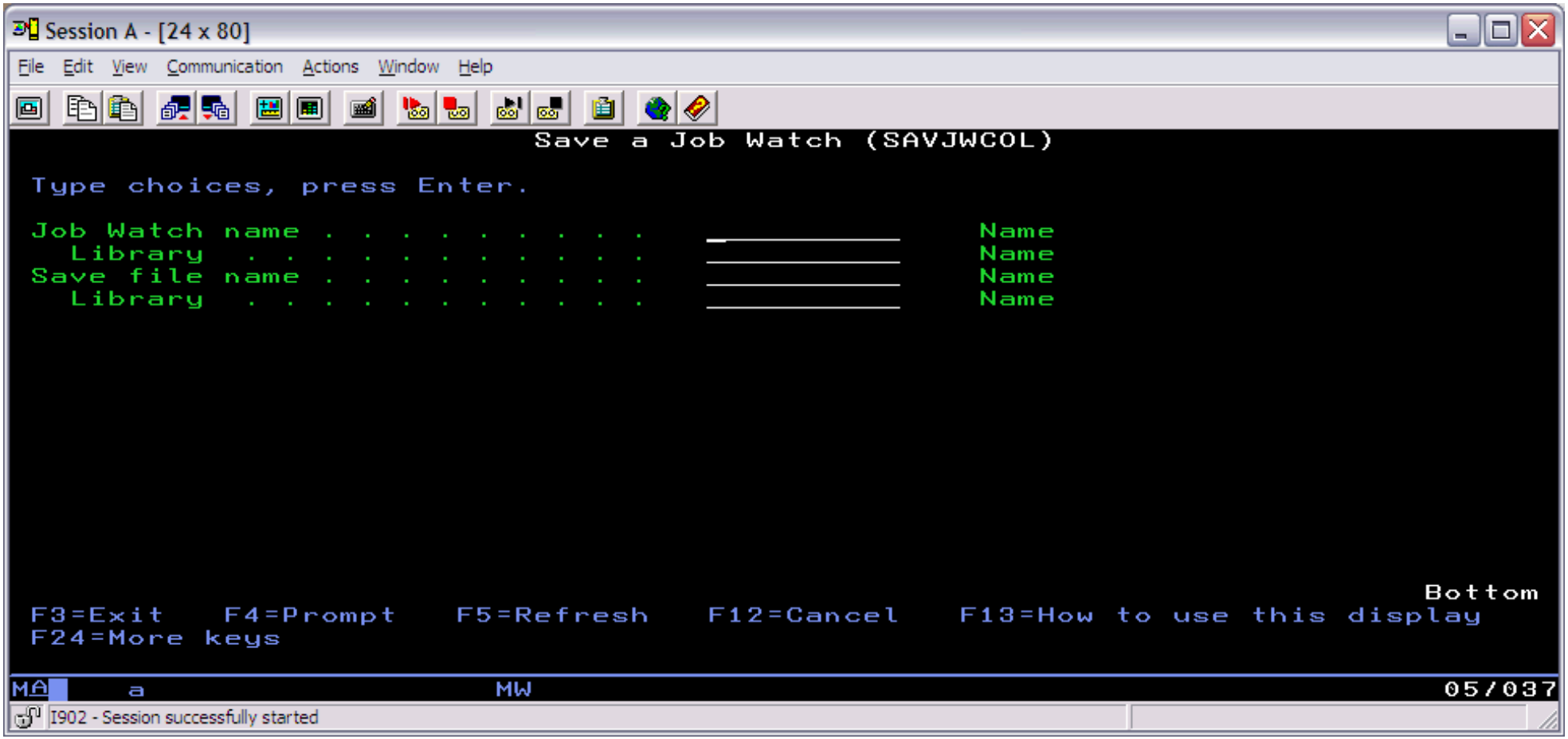
TEXT - Text description
This is an optional text description to give the monitor. The value must be no greater than 50 characters long.

Table of Contents | Previous | Next

1.4.17 SAVJWCOL

This command is used to save a job watch that was created on a system running i5/OS V5R3. The job watch is saved to a save file specified by the user. This save file must already exist, and if it contains data this data will be destroyed.

This command should be used along with the command RSTJWCOL in order to restore the job watch.



SAVJWCOL Parameters

COL - Job Watch name
The name of the job watch to save.

LIB - Job Watch library name
The library name of the job watch to save.

SAVFNAME - Save file name
The name of the save file to save the job watch into.

SAVFLIB - Save file library name
The library name of the save file to save the job watch into.

Table of Contents Previous Next

1.4.18 STRJWMON

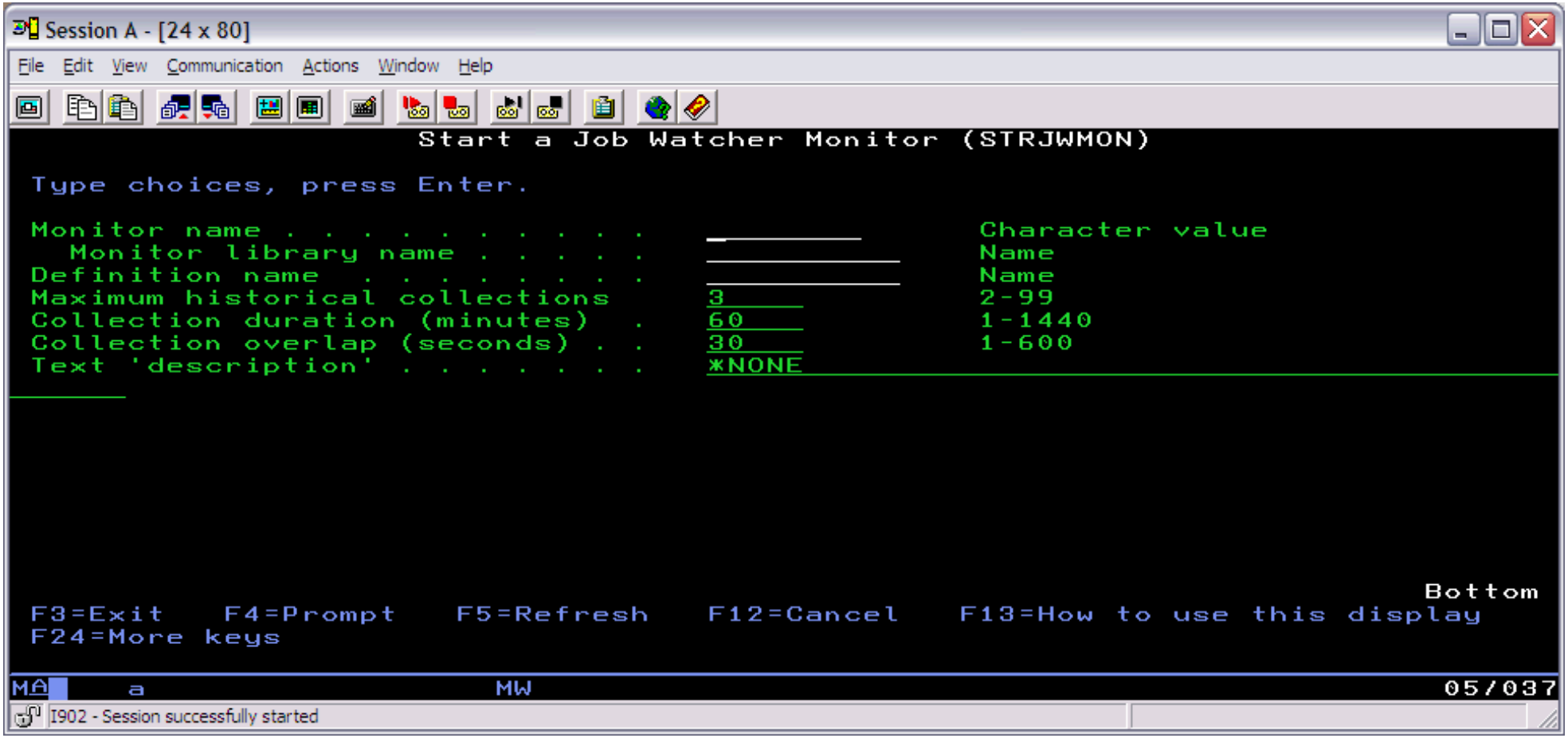
This command is used to start a Job Watcher monitor. The monitor must not already exist. Use the DLTJWMON to delete an existing monitor.

A monitor is a set of collections that continuously collect data over a system overwriting the oldest collection when it creates a new collection. Monitors are built from a Job Watcher definition which are stored in file QGPL/QAIDRJWDFN. A definition is a WCHJOB command string that indicates the parameters the monitor should use in its collections. Definitions can be created using the iDoctor client or by using the ADDJWDFN command.
The maximum historical collections parameter (COLNS) determines how many collections should be saved at one time.

A record in file QGPL/QAIDRJWM1 that identifies the existance and status of the monitor is created and updated by this command.

Note: This command should be submitted to batch using the SBMJOB command and not be ran interactively.

An example of this command is:



STRJWMON Parameters

MONITOR - Monitor name
The name of the monitor to create. This can be up to 8 characters long. The collection names are created using the monitor name with a number between 01 and 99 appended to the end of the monitor name.

Monitor names must be unique. You cannot create two monitors with the same name even into different libraries.

COLLIB - Monitor library name
The name of the library to create collections in for this monitor.

DFNNAME - Definition name
The name of the definition to use when creating collections for this monitor. A definition is a WCHJOB command string stored in file QGPL/QAIDRJWDFN.

COLNS - Maximum historical collections
The maximum number of collections to keep in the monitor library at one time. If the number of collections is exceeded upon creation of a new collection the oldest collection in the monitor will be deleted. This value must be between 2 and 99.

STRGAP - Collection duration (minutes)
This value indicates how much time should elapse between each collection being submitted by the monitor. This value is in minutes. Because of the large amounts of data collected by Job Watcher it is often better to collect smaller amounts of data per collection. This helps the queries running over the collection return their results faster.

Keeping this number less than or equal to 60 minutes is recommended.

OVRLAP - Collection overlap (second)
This value indicates how much time should elapse between starting the the next collection and ending the job that was running the previous collection. On busy systems it may take a minute or two for a Job Watch to initialize and begin collecting data. This parameter allows for continuous collection to occur during this initialization time.

TEXT - Text description
This is an optional text description to give the monitor. The value must be no greater than 50 characters long.

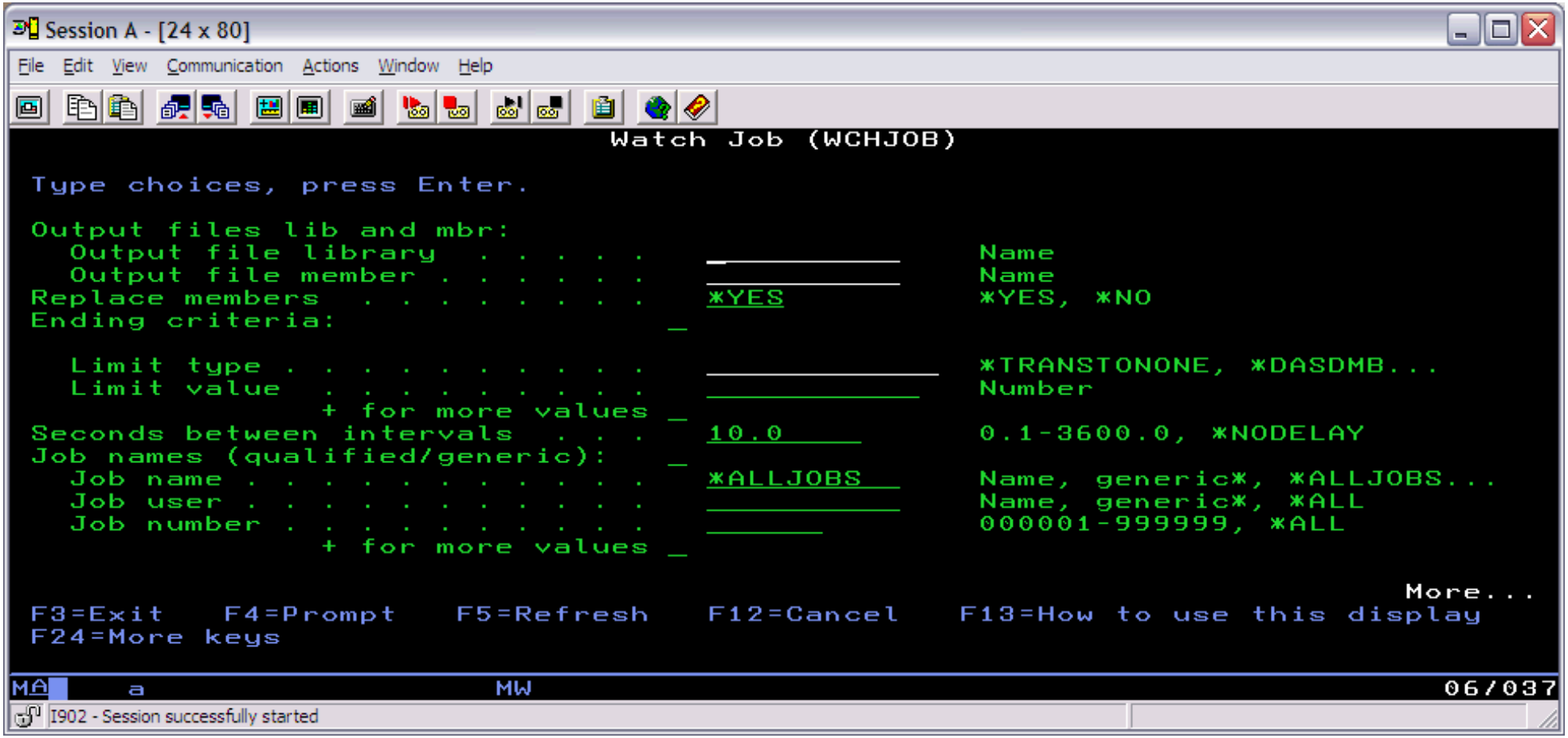
Table of Contents | Previous | Next

1.4.19 WCHJOB

The Job Watcher GUI provides real time views of job/tasks and/or threads on a system. These views are based on data files created by the WCHJOB command in library QIDRWCH (the Job Watcher library).

Job Watcher is a non-intrusive real time sampling of jobs/threads/tasks collecting performance information. All data collected is stored in database files named QAPYJW* in the collection library.
NOTE: The command runs interactively so it is recommended that the job be submitted to batch.

A screenshot of the WCHJOB command is shown below:



WCHJOB Parameters

OUTLIBMBR

Output files library
The name of the Job Watch collection library. The library must already exist.

Output files member
Specifies the name of the collection member. Creating a job watch will create several files in the output files library each having this member name.

REPLACE - Replace members

***YES**
A member with the same name in the collection library will be replaced with the new collection member.

***NO**
A member of the same name is not replaced and the collection will fail if the member name specified already exists.

UNTIL - Ending criteria

Limit type
The possible values for the limit type are:

***TRANSTONONE**
Job Watcher collection will end once all of the watching jobs/tasks/threads end. *TRANSTONONE is a single value and does not require the Limit value parameter.

***DASDMB**
Job Watcher collection will end once the total disk space consumed by the job watch exceeds the value specified (in megabytes).

***NBRSEC**
Job Watcher collection will end once the specified number of seconds has elapsed since the collection was started.

***NBRITV**
Job Watcher collection will end once the specified number of intervals have been collected.

Limit value
Enter a number for each. **Limit type***DASDMB = megabytes, *NBRSEC = seconds and *NBRITV = intervals.

INTDELAY - Seconds between intervals
Time between collection intervals in seconds.

***NODELAY**
Use this option to indicate that the Job Watcher engine should begin taking another snapshot immediately after the previous **snapshot completes.**

JOBS - Job names (qualified/generic)
Identify the job names to include in the collection. Each entry shall contain a specific or generic job name. If the special value *ALLJOBS is used then information about every job/thread on the system is collected.

Job name
This value is the specific job name, a generic job name, *ALL or *ALLJOBS.

Job user
This value may be the specific user name, a generic user name or *ALL.

Job number
This value is the 6 digit job number or *ALL.

TASKS - Task names
Identify the system tasks to include in the collection. Tasks may be specified using exact names or generic task names. Up to 20 entries are allowed for this parameter.

CURRUSER - Current user profile name
This parameter is used to select jobs by their current user profile. Up to 20 profiles may be specified.

TASKCOUNT - Thread task task count
List the task count(s) of the jobs/tasks to be included in the collection. This value must be entered in decimal format (not hex). Up to 20 are allowed.

CURRSBS - Current job subsystem name
Watch jobs running in the specified subsystem. Only one subsystem entry is allowed.

CURRPOOL - Current thread/task pool
Watch jobs running in the specified pool. Only one pool entry is allowed.

TSS - Thread selection string
The thread selection string is an alternate means to specify which jobs or tasks should be included in the collection.

The following parameter values must be used when defining the thread selection string: JOB, TASKNAME, CURRUSER, TASKCOUNT, CURRSBS, CURRPOOL Each parameter value must be enclosed in single quotes for all of the parameter types listed above. The Boolean values of .AND. or .OR. should be used if multiple entries are desired in the thread selection string.
Examples:

JOB = 'MYJOB/MYUSER/123111' .OR. CURRSBS = 'IDOCTOR'
CURRPOOL = '2'

TASKCOUNT = '131331112' .OR. TASKCOUNT = '13122223'

JOB = 'QZDA*/*ALL/*ALL' .OR. TASKNAME = 'RTZ*'

DATATYPEU - Fixed data

Data types
The following data types are available for collection:

***ACTGRPDETAIL**
Includes the activation group and program information along with heap size and blocks. Also contains *ACTGRPSUM

***ACTGRPSUM**
Collects activation group totals. Data is found in file QAPYJWPRC.

***SQLSTMT**
Collects last executed SQL statements and host variables.

***SQLCURSTMT**
Collects active SQL statements and host variables.

***SQLDETAIL**
Collects last executed SQL statements, host variables, prepared statement arrays and open cursors.

***CALLSTACK**
Collects call stacks. Up to 1000 levels deep may be collected for each stack.

***SOCKETTCP**
Lists details about all sockets open to an TCP/IP connection at the job/task/thread level.

***SOCKETJOBS**
Lists all jobs/threads/tasks using the same socket. Should be used in combination with *SOCKETTCP

Interval frequency
All fixed data types may be collected in one of the following ways:

***ALWAYS**
Data is collected every interval.

***NEVER**
Data is never collected.

Numeric value
Data is collected every Nth interval.

DATATYPEC - Dynamic data types

Data types
The following dynamic data types are available for collection.

***CONFLICTCALLSTACK**
Call stacks are captured when there is a holder/waiter conflict (i.e. seize)

***NONCONFLICTCALLSTACK**
Call stacks are captured for other conflict condition when there is not necessarily a conflicting holder.

Minimum duration (usecs)
Minimum wait in order to produce a call stack for either *CONFLICTCALLSTACK or *NONCONFLICTCALLSTACK. Time is measured in microseconds (usecs).

DATATYPES - Data selection string
Like the thread selection string parameter, this parameter provides an alternate means to specify the Fixed data types and Dynamic Data types parameters.

TRIGF - Conditional collection control
This option is used to conditionally gather data when certain performance thresholds have been met. A file must be provided to the engine that contains the conditional criteria selection string.

This conditional criteria selection string may contain direct field comparisons as well as functional comparisons. When making comparisons the following operators must be used: .GT., .LT., .EQ., .NE., .GE., .LE.

Direct field comparison:
format: fieldname .op. value
example: QTIME01 .GT. 200 .OR. QTIME05 .GT. 1000
valid fields: Any fields in file QAPYJWTDE or QAPYJWPRC.

"Functional" comparisons:
These comparisons involve a function applied to a field such as RATE or PERCENT.

Percent function
format: PERCENT(fieldname) .op. value
example: PERCENT(QTIME09) .GT. 50
valid fields: Any wait bucket TIME values. Field names QTIME01-QTIME32.

Average function
format: AVERAGE(fieldname1)(fieldname2) .op. value
example: AVERAGE(QTIME01)(QCOUNT01) .LT. 35
valid fields: This function requires a time/count pair, both fields must be specified. The only pairs like this available at this time are the wait bucket fields QTIME01-QTIME32 and QCOUNT01-QCOUNT32

Rate function
format: RATE(fieldname).comparand.value
example: RATE(QCOUNT08).EQ.25
valid fields: Fields QCOUNT01-CCOUNT32

File name
Name of the source file containing the conditional criteria selection string.

Library name
The library name where the conditional criteria file resides.

TRIGFMBR - Condition collection file member
Name of the conditional criteria file member.

ENDASPTHR - DB2 file ASP threshold
End the collection based on the ASP threshold or override with a percentage.

***ASPTHR**
Collection will end if the ASP threshold defined in the DST/SST Work with ASP threshold option is exceeded.

Percentage
Override the ASP threshold with another percentage. This change will not affect the current ASP setting and is only valid for this collection.

SETBLKMAP - Use WCHJOB block mappings
Job Watcher default block bucket mappings. For IBM representatives only.

TEXT - Text 'description'
Description of the collection.

DTAAVAIL - DB2 data availability
This option controls how often records are written out to the database files during collection. The following choices are available:

***ITVEND**
The database files will be updated immediately after each interval is gathered.

***END**
The database files will not be updated after each interval. Data will be blocked in large chunks and written out to the database files once the block of data is full or the job watch ends. This option can be used to greatly increase the speed that the Job Watcher engine can sample data.

MINCPUDTAT - Minimum CPU statistics gathering time
A separate file is created by the WCHJOB command called QAIDRJWCPU which collects many different types of CPU statistics. This parameter determines how often this information should be captured and written to this file. This value determines the number of seconds to wait before gathering the CPU information again.

FRC1STINT - Force all threads 1st interval

***NO**
Idle jobs and tasks will not be collected as if they were active on the 1st interval of collection.

***YES**

On the 1st interval of collection all threads will be sampled even idle threads. This will provide the job name or task name for idle jobs/tasks.

NOTE: Using this option may take several minutes (possibly hours) to sample the 1st interval of data if the options for SQL statements, activation groups are turned on and there are several hundreds of thousands of jobs on the system.

Table of Contents Previous Next

Chapter 2 Heap Analyzer

This chapter describes the Heap Analyzer component on the server side. Heap Analyzer provides tools for doing Java heap analysis, advanced debug, and application review.

The following topics will be covered:

- The types of collections available
- A description of the files created by Heap Analyzer
- Green-screen commands

Table of Contents | Previous | Next

2.1 Libraries

When Heap Analyzer is installed, two libraries are created on the iSeries. These libraries are named QIDRGUI and QIDRHAJ.

QIDRGUI contains programs and commands needed by the iDoctor for iSeries client. It also contains objects that are used by all other iDoctor for iSeries components (except PTDV).

Library QIDRHAJ is the Heap Analyzer library for release V5R3. This library contains programs and commands needed to create and work with the data Heap Analyzer produces. The output files produced by Heap Analyzer are located in library QIDRHAJ. These files are named QPYRTJVM*.

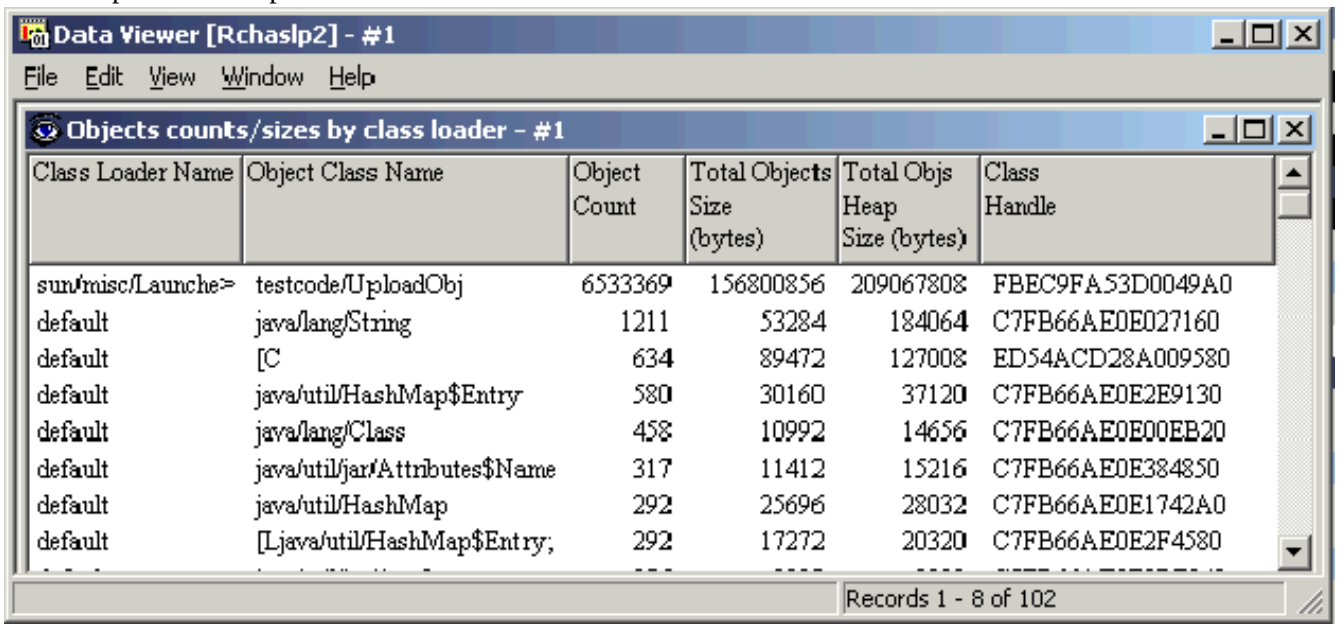
Table of Contents | Previous | Next

2.2 Types of Collections

Heap Analyzer provides two different modes for collecting information about a heap: Object Table Snapshot and the Object Create Profiler. Collections are created using the WCHJVA command or by using the client interface over a single job's JVM.

Object Table Snapshot - Similar to DMPJVM, however it is a non-intrusive snapshot which provides detailed information on the object counts, object size, and actual heap size used. DMPJVM requires the job to be held in order to collect heap object table statistics and it will not work on large Websphere heaps. Heap Analyzer runs in the background and will run on enormous heaps without any affect on the JVM operation while it's running.

An example of the output:



Class Loader Name	Object Class Name	Object Count	Total Objects Size (bytes)	Total Objs Heap Size (bytes)	Class Handle
sun/misc/Launche>	testcode/UploadObj	6533369	156800856	209067808	FBEC9FA53D0049A0
default	java/lang/String	1211	53284	184064	C7FB66AE0E027160
default	[C	634	89472	127008	ED54ACD28A009580
default	java/util/HashMap$Entry	580	30160	37120	C7FB66AE0E2E9130
default	java/lang/Class	458	10992	14656	C7FB66AE0E00EB20
default	java/util/jar/Attributes$Name	317	11412	15216	C7FB66AE0E384850
default	java/util/HashMap	292	25696	28032	C7FB66AE0E1742A0
default	[Ljava/util/HashMap$Entry;	292	17272	20320	C7FB66AE0E2F4580

Object Create Profiler - Used to "profile" object creates, **in real time.** To really understand how this works you have to understand a little about the architecture of how the JVM handles object creates. Every object has two sizes associated with it, the actual object size and the heap size that object required. For instance in OS/400 V5R2 there are 23 bucket sizes, every object will fall into one of these buckets.

When you turn on the profiler for a JVM it causes every object created for each bucket and thread to be logged. The information logged includes:

1. A counter for the bucket in this thread
2. The object class name
3. The block size

4. The object size
5. The object, class, and loader address
6. Stack information to identify where the object was created from.

The code that does the harvesting of this data can not run as fast as 100's of threads can create 1000's of objects, so we get the information as fast as we can. What we do preserve is the count for each bucket. So at each interval that we harvest the data we will know how many objects of this bucket size were created and we will have the details for this specific one. With the information we've reviewed so far this has proven to be a very reliable way of profiling an application for who is creating objects, what objects they are and what size they are, even with a limited sample.

An example of the output:

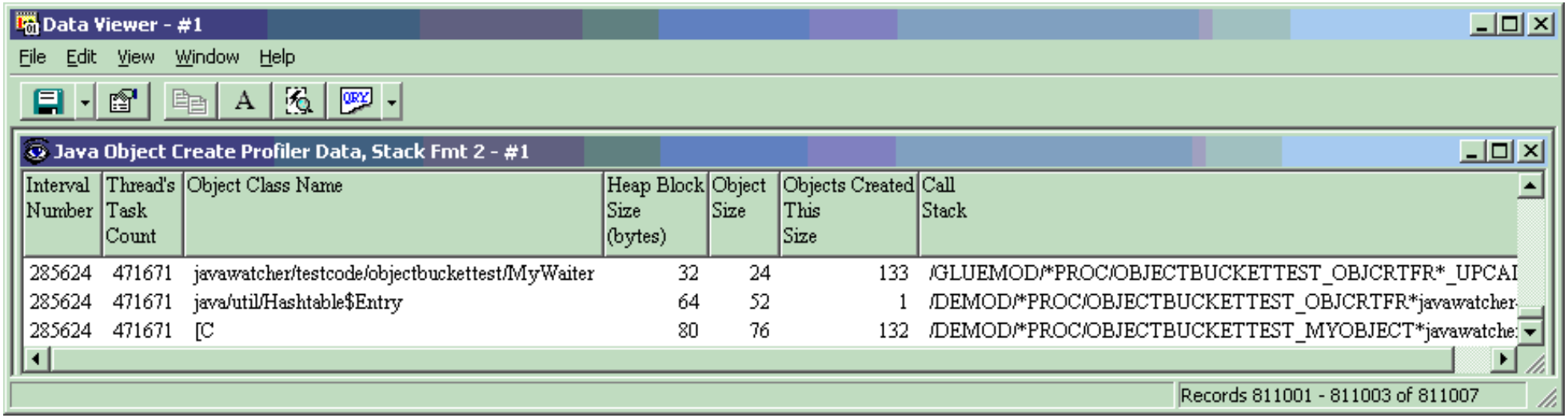


Table of Contents | Previous | Next

2.3 Heap Analyzer - Object table dump output files

This section describes the output files for the object table dump collection. Collections are created via the WCHJVA command in library QIDRHAJ.

The files exist in the QIDRHAJ library, and get duplicated when creating a collection in a library for the first time. Each of these files will contain a member name matching each collection created in the given library unless otherwise noted.

Table of Contents | Previous | Next

2.3.1 QPYRTJVMHD - Object table dump definitions

Description: This file stores the definition (command string) used to create the object table dump collection.

Record: The single record in this file contains the command string used to create the collection.

Field name	Field Description	Format	Comments
NAME	Definition name	Char(10)	
OSVRM	OS VRM	Char(3)	
COMMAND	Command string	Char(1000)	

Table of Contents | Previous | Next

2.3.2 QPYRTJVMH0 - Object table dump control file

Description: This file contains information about the object table dump collections that have been created and still currently exist in a library.

Record: One record is created in this file per object table dump that exists in the collection library.

Field name	Field Description	Format	Comments
NAME	Job name	Char(32)	Job that was analyzed (is running the JVM)
TASKCOUNT	Initial thread task count	Packed(11)	
JVMHANDLE	JVM handle	Hex(16)	Uniquely identifies the Java Virtual Machine
GCCYCLE	GC cycle	Binary(4)	
MBRNAME	Member name	Char(10)	
STARTTOD	Start sample time	Timestamp	
ENDTOD	End sample time	Timestamp	
STATUS	Collection status	Char(1)	
COLLJOB	Collecting job name	Char(32)	Job creating the collection
LIMITTYPE	Collection limit type	Char(1)	
LIMITVALUE	Limit value	Packed(6)	
DESC	Collection description	Char(50)	
SYSOSVRM	Collecting system OS VRM	Char(3)	

IDOCVRM	Heap Analyzer VRM	Char(3)	This field no longer used.

Table of Contents | Previous | Next

2.3.3 QPYRTJVMH1 - Object table dump class loaders

Description: This file stores the names of class loaders for objects found within the object table dump

Record: A record is created in this file per class loader found.

Field name	Field Description	Format	Comments
GCCYCLE	GC cycle	Binary(4)	
LOADERID	Class loader ID	Binary(4)	
TRUCLNAML	True class loader name length	Binary(4)	
CLNAME	Class loader name	Char(5000)	

Table of Contents | Previous | Next

2.3.4 QPYRTJVMH2 - Object table dump object listing

Description: This file contains information about the objects found in the JVM at the moment the collection was taken.

Record: Each record represents a different object (class name) found within the JVM.

Field name	Field Description	Format	Comments
GCCYCLE	GC cycle	Binary(4)	
LOADERID	Class loader ID	Binary(4)	
GLBLOBJECT	Global registry info	Char(1)	
GLBLROOTS	Global registry roots	Char(1)	Y = roots in Global registry
OBJCOUNT	Object count	Packed(11)	
TOTOBJSIZ	Total object size (bytes)	Packed(11)	
TOTHEPASIZ	Total object heap size (bytes)	Packed(11)	
TRUOCNAML	True object class name length	Binary(4)	
OCNAME	Object class name	Char(5000)	
ANOBJHNDL	Object class handle	Char(16)	

Table of Contents | Previous | Next

2.4 Heap Analyzer - object create profile output files

This section describes the output files for the object create profile collection. Collections are created via the WCHJVA command in library QIDRHAJ.

The files exist in the QIDRHAJ library, and get duplicated when creating a collection in a library for the first time. Each of these files will contain a member name matching each collection created in the given library unless otherwise noted.

Table of Contents | Previous | Next

2.4.1 QPYRTJVMFD - Object create profile definitions

Description: This file stores the definition (command string) used to create the object create profile collection.

Record: The single record in this file contains the command string used to create the collection.

Field name	Field Description	Format	Comments
NAME	Definition name	Char(10)	
OSVRM	OS VRM	Char(3)	
COMMAND	Command string	Char(1000)	

Table of Contents | Previous | Next

2.4.2 QPYRTJVMF - Object create profile control file

Description: This file contains information about the object create profile collections that have been created and still currently exist in a library.

Record: One record is created in this file per object create profile that exists in the collection library.

Field name	Field Description	Format	Comments
NAME	Job name	Char(32)	Job that was analyzed (is running the JVM)
MBRNAME	Member name	Char(10)	
STARTTOD	Start sample time	Timestamp	
ENDTOD	End sample time	Timestamp	
SYSOSVRM	Collecting system OS VRM	Char(3)	
JDKLVL	JDK level	Char(16)	
STACKFMT	Stack output format	Char(1)	
STATUS	Collection status	Char(1)	
TOTUSECS	Collection duration (usecs)	Packed(11)	
COLLJOB	Collecting job name	Char(28)	Job creating the collection
LIMITTYPE	Collection limit type	Char(1)	
LIMITVALUE	Limit value	Packed(6)	
STACKSKIP	Additional stack frames to skip	Binary(2)	
DESC	Collection description	Char(50)	
IDOCVRM	Heap Analyzer VRM	Char(3)	This field no longer used.

Table of Contents | Previous | Next

2.4.3 QPYRTJVMF0 - Object create profile interval summary

Description: Provides summary information for the object create profile for each interval.

Record: One record is created per interval collected.

Field name	Field Description	Format	Comments
INTERVAL	Interval number	Binary(4)	
NUMTHDS	Number of threads this interval	Binary(2)	
NUMTHDSR	Number of threads reporting this interval	Binary(2)	
STARTTOD	Interval start time	Timestamp	
ENDTOD	Interval end time	Timestamp	
USECS	Interval delta/elapsed time (usecs)	Packed(11)	
CUMMUSECS	Total time from start of collection (usecs)	Packed(11)	
PGMLOOKUPS	Tool perf statistic	Binary(4)	
PGMHITS	Tool perf statistic	Binary(4)	
JVMHANDLE	JVM handle	Hex(16)	Uniquely identifies the JVM

Table of Contents | Previous | Next

2.4.4 QPYRTJVMF1 - Object create profile threads

Description: This file contains information about the threads identified during collection.

Record: One record is created per thread per interval.

Field name	Field Description	Format	Comments
INTERVAL	Interval number	Binary(4)	
TASKCOUNT	Job's task count	Packed(11)	
THDNAMEL	Java thread name true length	Binary(2)	
THDNAME	Java thread name	Char(128)	Contains the 1st 128 characters only
NUMENTS	Number of entries this interval	Binary(2)	

Table of Contents | Previous | Next

2.4.5 QPYRTJVMF2 - Object create profile details - call stack fmt 1

Description: This file contains information about the objects created during collection including partial call stacks from some of the object creates.

Record: One record is created in this file for each object class name created per interval.

Field name	Field Description	Format	Comments
INTERVAL	Interval number	Binary(4)	
TASKCOUNT	Thread's task count	Packed(11)	
TRUOCNAML	True class name length	Binary(2)	
OCNAME	Object class name	Char(256)	Contains the 1st 256 characters only
BLOCKSIZ	Heap object size (bytes)	Packed(6)	
OBJSIZ	Object size (bytes)	Packed(6)	
OBJADDR	Object address	Hex(8)	
CLASSADDR	Class address	Hex(8)	
LOADRADDR	Loader address	Hex(8)	
OBJCOUNT	Objects created this size	Packed(11)	
INSTADDR1 - INSTADDR5	Stack 1-5 instruction addresses	Hex(8)	
MTHDNMLEN1 - MTHDNMLEN5	Stack 1-5 true method name lengths	Binary(4)	
MTHDNAME1 - MTHDNAME5	Stack 1-5 method names	Char(100)	

PGMNAME1 - PGMNAME5	Stack 1-5 program names	Char(10)	
MODNAME1 - MODNAME5	Stack 1-5 module names	Char(10)	

Table of Contents | Previous | Next

2.4.6 QPYRTJVMF3 - Object create profile details - call stack fmt 2

Description: This file contains information about the objects created during collection including partial call stacks from some of the object creates.

Record: One record is created in this file for each object class name created per interval.

Field name	Field Description	Format	Comments
INTERVAL	Interval number	Binary(4)	
TASKCOUNT	Thread's task count	Packed(11)	
TRUOCNAML	True class name length	Binary(2)	
OCNAME	Object class name	Char(256)	Contains the 1st 256 characters only
BLOCKSIZ	Heap object size (bytes)	Packed(6)	
OBJSIZ	Object size (bytes)	Packed(6)	
OBJADDR	Object address	Hex(8)	
CLASSADDR	Class address	Hex(8)	
LOADRADDR	Loader address	Hex(8)	
OBJCOUNT	Objects created this size	Packed(11)	
INSTADDR1 - INSTADDR5	Stack 1-5 instruction addresses	Hex(8)	
MTHDNMLEN1 - MTHDNMLEN5	Stack 1-5 true method name length	Binary(4)	

STACK	Call stack details	Char(624)	Contains 5 levels of the call stack for one of the object creates of this type

Table of Contents | Previous | Next

2.5 Commands

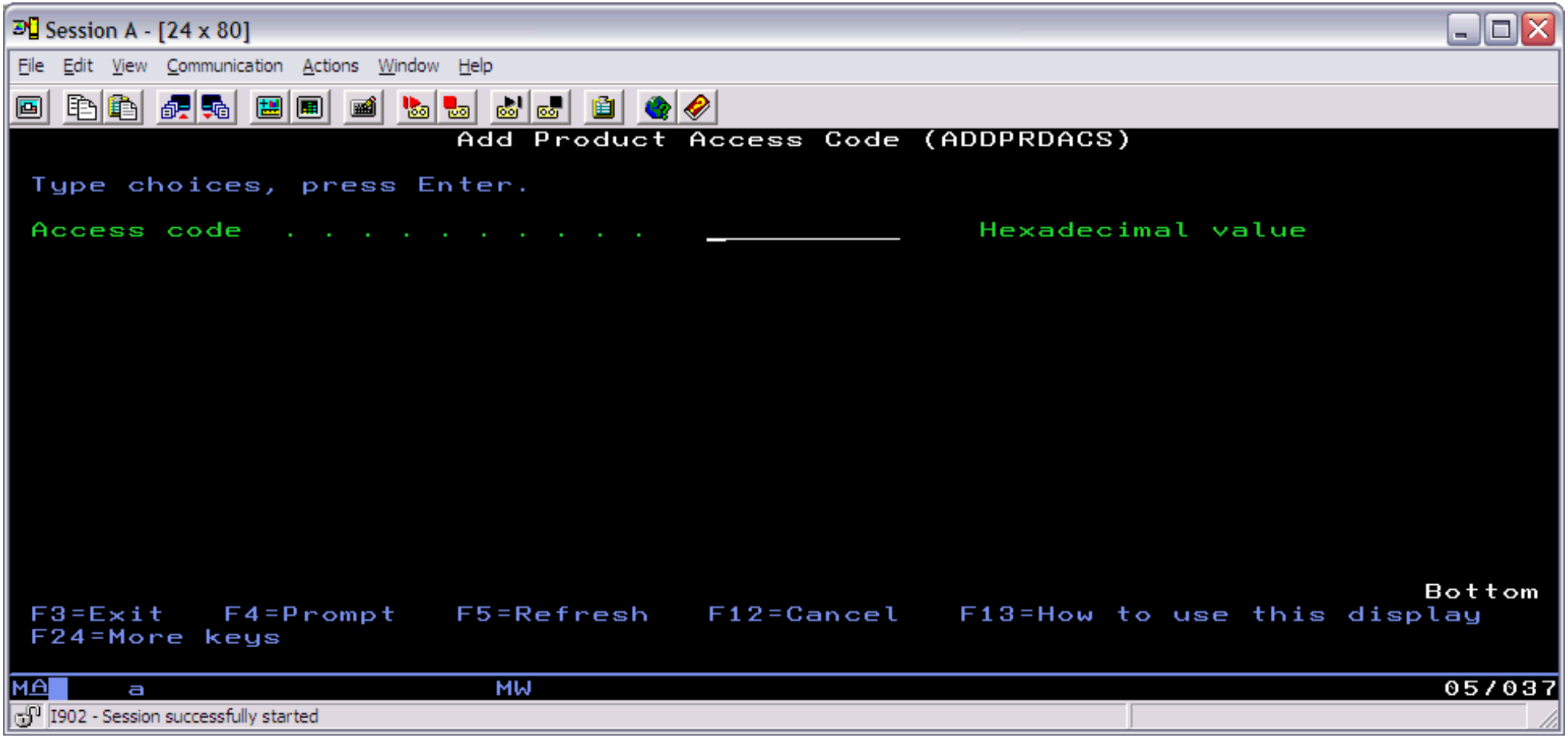
This section discusses the green screen commands that are shipped with Heap Analyzer in library QIDRWCH.

Table of Contents | Previous | Next

2.5.1 ADDPRDACS

This command is used to authorize users of the system to an iDoctor component.

Access codes are granted through the website which this product was downloaded from: http://www-912.ibm.com/i_dir/idoctor.nsf



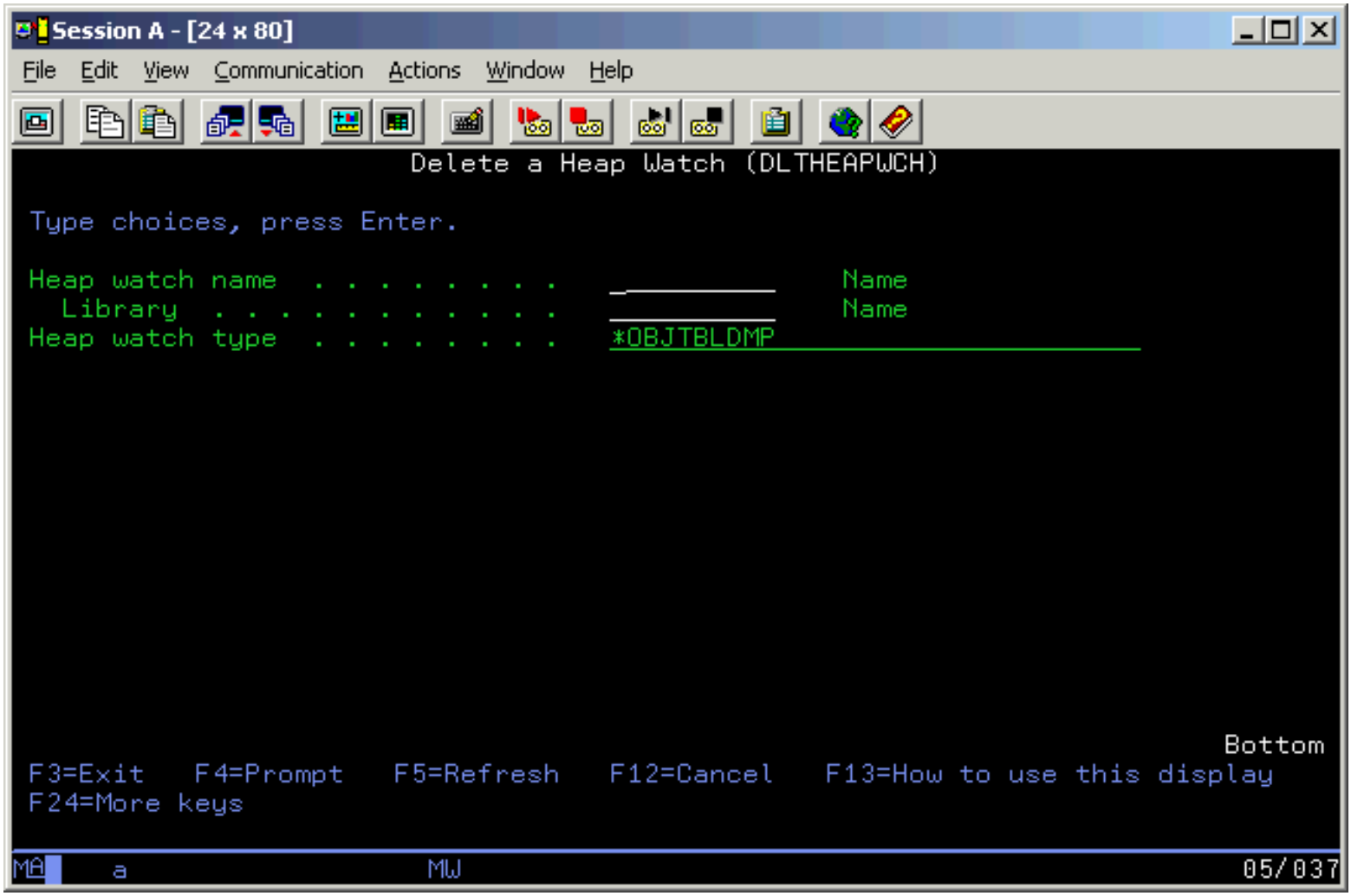
ADDPRDACS Parameters

ACSCDE - Access code
The access code provided by IBM to grant use of an iDoctor component on the current system.

Table of Contents | Previous | Next

2.5.2 DLTHEAPWCH

This command is used to delete a Heap Analyzer collection from a library. It will delete the member name specified for all files of the indicated type.



DLTHEAPWCH Parameters

WCHNAME - Collection name
The name of the collection. This value equals the member name used in the Heap Analyzer output files.

LIB - Library name
The library name where the collection will be deleted from.

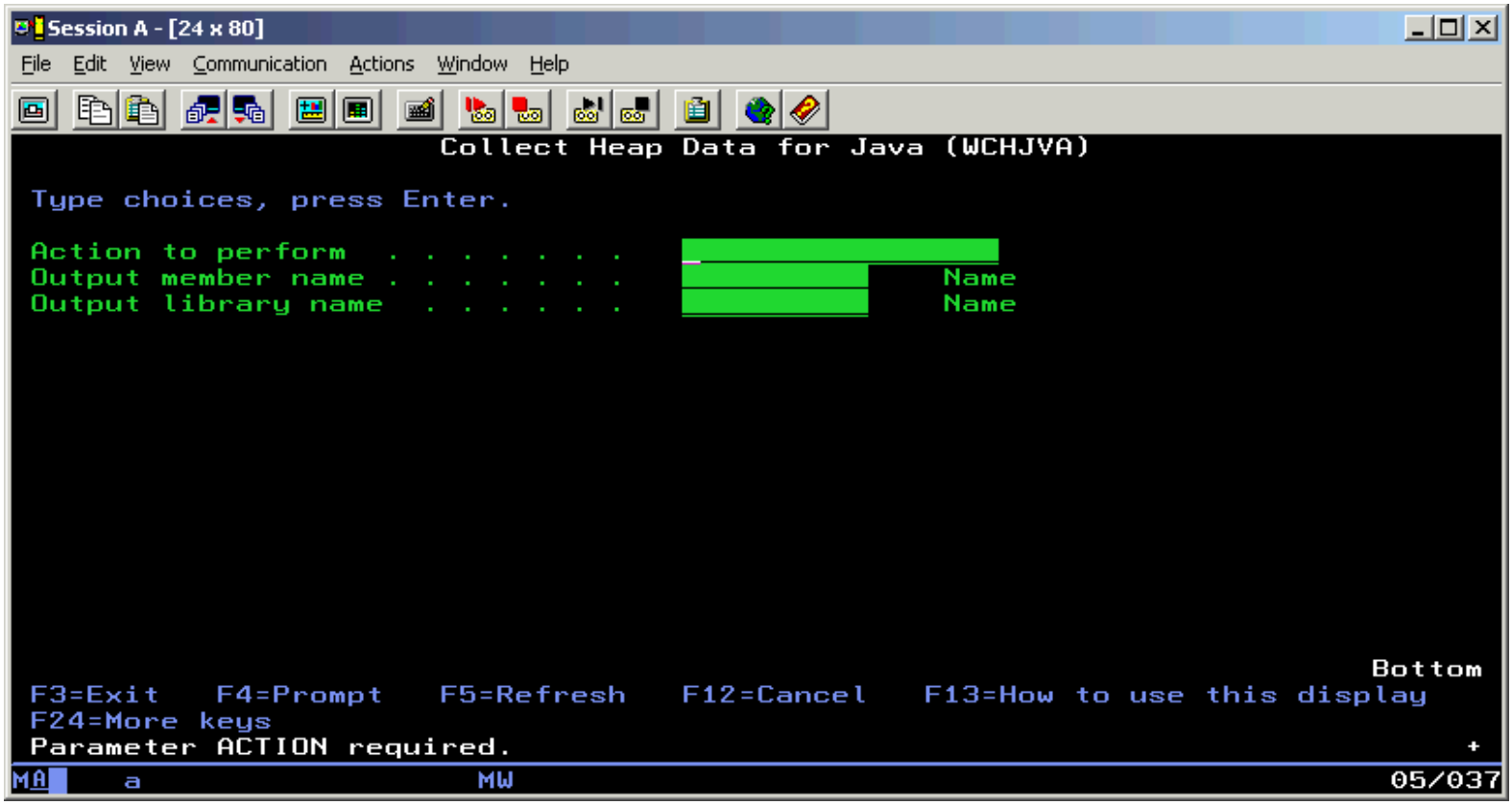
TYPE - Collection type
The Heap Analyzer collection type. This value can either be *OBJTBLDMP for an object table dump collection or *OBJCRTPROFILE for an object create profile collection.

Table of Contents | Previous | Next

2.5.4 WCHJVA command

Heap Analyzer is shipped in library QIDRHAJ. You need to add this library to your library list prior to running from the green screen.

To create a Heap Watch, prompt the WCHJVA command for the following display:

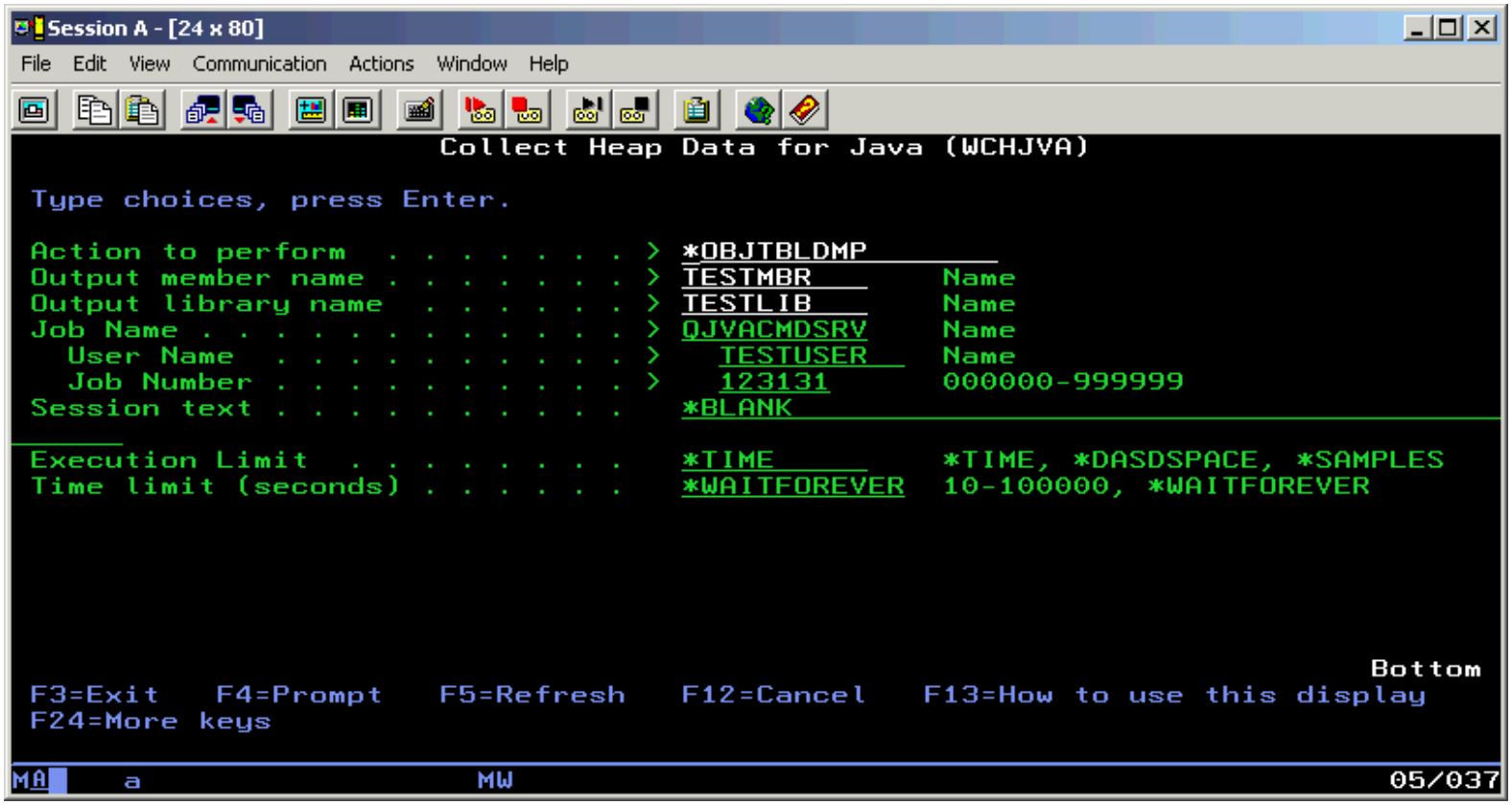


There are three options that you can select for the **Action to perform** parameter.

- *OBJTBLDMP - Used to get a snapshot of the object table in the heap.
- *OBJCRTPROFILE - Used to collect object create statistics.
- *OBJROOTFINDER - Used to investigate an object for roots **(for service use only)**

The **Output member name** and **Output library name** define the name of the library and member name to use when creating the Heap Watch. When running from green screen the library you specify must exist or the command will fail.

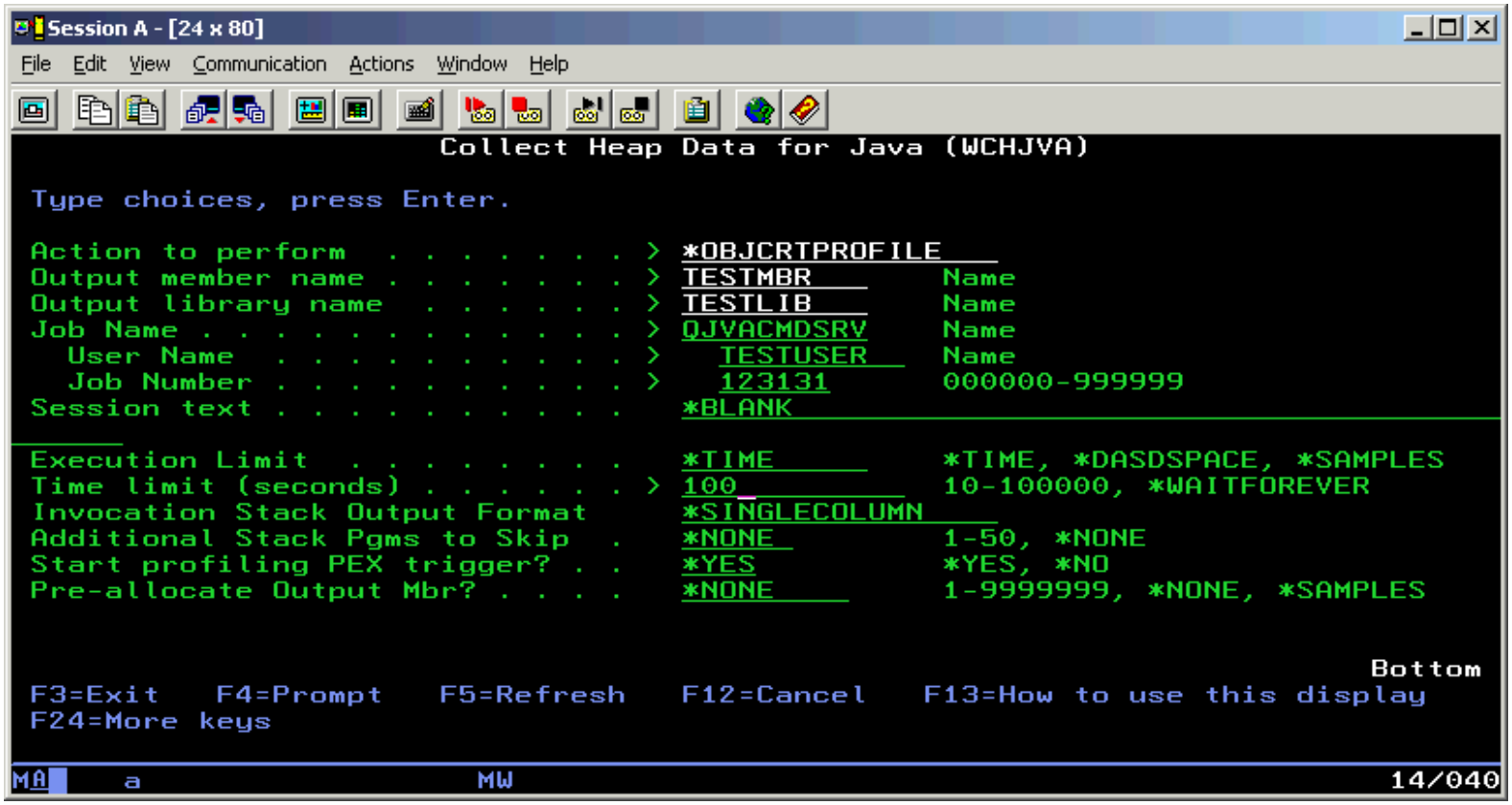
The available options when the **Action to perform** parameter is *OBJTBLDMP are:



The **Job Name** is the job that contains the JVM that you need to collect information about.

The **Execution Limit** should be set to *TIME and then set the Time Limit as appropriate for your situation. When possible it's recommended that you use *waitforever (no timeout). If you specify a time limit and you hit it then you have to be aware that you are working with incomplete data, and use it accordingly. The volume of data collected is relatively low and we are not collecting multiple samples so *DASDSPACE and *SAMPLES cannot be used and in fact are not available selections when using the GUI.

The available options when the **Action to perform** parameter is *OBJCRTPROFILE are:



The **Job Name** is the job that contains the JVM that you need to collect information about.

The **Execution Limit** can be set for TIME, DASDSPACE, or SAMPLES and should be set appropriately based on any resource concerns. DO NOT use *TIME without a limit set as you would eventually run out of space if the JVM remained active.

The **Invocation Stack Output Format** should be set to *SINGLECOLUMN for the format. This will put all the stack information into a single “STACK” field in QPYRTJVMF3 file. The *SEPARATECOLUMNS format is not often used and will likely be removed from future support.

The **Additional Stack Programs to Skip** is a parameter that allows you to skip x number of frames. Our stack support gives us a 5 frame view of the request to create the object that was logged, this skip parameter allows this view to float on the call stack, the default view is to skip the first (oldest) 4 frames and then log 5 through 9. If you supply a skip count that puts us off the end of the stack, rather than fail, you will get the default output. The stack information is presented in a single field (*SINGLECOLUMN) with the most recent, or top of the stack, on the left side of the data.

The **Start profiling PEX trigger** parameter should be set to *yes, with this setting the PEX support required for the logging to occur will automatically be started (and ended) when you submit the command. The only reason to specify *no would be if the PEX definition is already active for some reason. The PEX definition has to have the Java *Service Event enabled for the object creates to be logged.

The **Pre-allocate Output Mbr** parameter can be used in some cases to get more consistent times between intervals or collections. In some cases you can notice delays or gaps between some of the intervals, this is due to the overhead of allocating space for the output member, with this setting you can take care of this work up front and avoid these delays at collection time.

Table of Contents Previous Next

Chapter 3 PEX Analyzer

This chapter describes how to collect Performance Explorer (PEX) data for analysis with iDoctor for iSeries PEX Analyzer. This chapter includes:

- A description of the problems or low-level information that can be determined with a PEX collection
- The method and command syntax needed to collect PEX data
- How to move collected PEX data from one iSeries to another.
- Descriptions of green-screen commands shipped with PEX Analyzer

Table of Contents | Previous | Next

3.1 Libraries

When PEX Analyzer is installed, two libraries are created on the iSeries. These libraries are named QIDRGUI and QIDRPA. QIDRGUI contains programs and commands needed by the iDoctor for iSeries client. It also contains objects that are used by all other iDoctor for iSeries components (except PTDV). Library QIDRPA is the PEX Analyzer library for release V5R3. This library contains programs and commands needed to create and work with PEX data and PEX analyses.

Table of Contents | Previous | Next

3.2 Performance Analysis on the iSeries platform

Performance Explorer (PEX) is a set of performance collection and reporting functions and commands that became available on RISC versions of OS/400, 5716-SS1 and the iSeries Performance Tools/400, licensed program product (LPP) 5716-PT1. PEX is the above-the-Machine-Interface portion and is the term most often used in this and other related documentation. Another term that might be seen is **Performance Data Collector** (PDC). PDC is the below-the-Machine-Interface (Licensed Internal Code) layer that supports PEX functions.

The PEX functions, which are part of a total iSeries performance management methodology, should **not** be the first set of performance tools used when analyzing a performance problem. PEX performance analysis is a detailed, **bottom up** approach. An overall picture of system or application performance should be known (using the GO PERFORM functions of the Performance Tools/400 licensed program) before using PEX for detailed analysis.

The PEX **collection** functions and related commands are included as part of the base OS/400 support.

The base PEX **reporting** function and its associated command are part of the Performance Tools/400 LPP, manager feature 5101.
This separation of PEX definition and collection capabilities (under OS/400) and reporting capabilities (under the Performance Tools/400 licensed program) is similar to using OS/400 Performance Monitor support to collect performance data and the Performance Tools/400 licensed program to generate reports from the data collected using Performance Monitor. The data collected using Performance Monitor and the data collected using PEX are placed in multiple OS/400 database files.

The iDoctor for iSeries component **PEX Analyzer** provides graphical visualization of PEX collection data. Previously, the Print PEX Report (PRTPEXRPT) command (part of the Performance Tools/400 LPP) was the typical method used to view reports over PEX data. PEX Analyzer is a value-add to PRTPEXRPT by providing more flexible interfaces for viewing potentially large and complex data.

PEX supports the following independent methods, or modes, of data collection:

STATS: Gathers CPU and paging information that is summarized on the fly about MI program invocation and complex MI instruction invocation. For those familiar with CISC-based iSeries performance tools, STATS mode is the replacement for TPST (Time and Paging Statistics). PEX

STATS is focused on identifying application and IBM programs or modules consuming excessive CPU utilization or performing a high number of disk I/O operations. STATS is **program-oriented**.

PROFILE: Gathers CPU usage profile information over a select set of programs. For those familiar with CISC-based AS/400 performance tools, PROFILE mode is the replacement for SAM (Sampled Address Monitor). PEX PROFILE support is focused on identifying high-level language (HLL) program hot spots (high CPU consumption) based on source program statement numbers. Typically, PEX STATS is used to identify programs that should be investigated further as potential performance bottlenecks. PEX PROFILE can then be used on the programs identified for further analysis.

TRACE: Gathers detailed, information that is not summarized about a wide variety of **events**.

The iDoctor for iSeries client component PEX Analyzer provides graphical data management and visualization for a subset of PEX **TRACE**, **PROFILE** and **STATS** data.

Types of Data That Can Be Collected by PEX and Analyzed by PEX Analyzer

PEX (and the underlying PDC) provide approximately one hundred different types of TRACE events. While all of these can provide interesting performance-related and nonperformance-related perspectives, only a subset of these TRACE events currently are **available** within PEX Analyzer.

The following PEX TRACE events can be analyzed within PEX Analyzer:

- Activation group events
- Logical DASD requests
- Physical DASD operations
- DASD **space** change requests
- Local (non-DDM) data base file full OPENs and CLOSEs
- Local (non-DDM) data base file logical I/Os
- Local and remote (DDM) change and retrieve data area
- Local and remote (DDM) send and receive data queue
- MI program CALLs/RETURNs
- Task switch events
- Websphere events

Note: The MI CALL/RETURN events are optionally used to assign responsibility of the occurrences of the other event types to specific system or application programs.

Why Use PEX Analyzer to Analyze a System or Application

At a high level, the following may result in the need to analyze a system or application:

- Some other command or tool indicates a possible problem. PEX Analyzer can be used to

provide more information.
- As part of a general system clean-up to investigate possible inefficient operations.

The following reviews the event types and describes briefly how each could be used.

The first, **Logical DASD requests** and **Physical DASD Operations,** work in tandem to provide detailed information about DASD operations. The information gathered in each event includes the following:

- A tag that indicates the job or system task requesting the operation
- The time of day the operation started (with microsecond granularity)
- The virtual or real address involved
- The name of the object (or LIC segment) involved
- Type of operation (sync-read, async-read, sync-write, and so on)
- DASD unit involved
- DASD sector numbers involved
- LIC and/or MI programs requesting the operation
- Number of DASD sectors read or written
- Main store pool involved
- Elapsed time the operation took to complete (microsecond granularity) (Optional)

This information can be gathered about **every** DASD operation that occurs during a PDC/PEX TRACE. It is a very detailed source of DASD operation information. This information could provide additional information on questions or problems. For example:

- Some DASD units are much busier than others (what objects or jobs are paging on them?)
- After adding a new application, all of the DASD units are **too busy**. What exactly are they doing?
- Sometimes the Work with Disk Status (WRKDSKSTS) command and performance monitor data disagree on **percent busy**. What is the real DASD response time? What are the average and maximum times?
- Would a Save/Restore of a file help in its distribution across DASD units?

DASD space change request events gather the following information about **every** DASD space change request during a trace:

A tag that indicates the job or system task requesting the change

Time of day the change was requested (with microsecond granularity)

- The virtual or real address involved
- The name of the object (or LIC segment) involved
- Type of operation (create, extend, truncate or destroy)
- LIC and/or MI programs requesting the change
- Size of the space being consumed or freed on DASD

Like the DASD operations above and all the other types of TRACE events, DASD space change events provide a tremendous amount of detailed information. These events are used primarily to investigate how DASD free space is being consumed. It is important to note that this can determine only **changes** to DASD space allocations. It does **not** statically determine where objects are placed or the size of objects. It does, however, determine how new objects are being created and how old objects are extended.

Local (non-DDM) data base file full OPENs and CLOSEs gathers the following information about every local data base full file open:

- A tag that indicates the job requesting the OPEN or CLOSE
- Time of day (microsecond granularity)
- File, library, and member name involved

Users who work on iSeries application analysis should not need usage examples for this, and should be anxious to use it.

Local (non-DDM) data base file logical I/Os (LDIOs) is probably the most valuable application-analyzing event type. It gathers the following:

A tag that indicates the job requesting the LDIO

- Time of day (microsecond granularity)
- File, library, and member name involved
- Detailed operation type information (get-by-get-nodata, get-by-key-data, put-single, put-multiple, and so on
- Record-not-found and end-of-file indication
- First relative record number involved
- Number of records retrieved

Note: This information is gathered about **every** non-DDM LDIO that occurs during the trace.

Additional Details Available with the Database OPEN/CLOSE and LDIO Trace Events

As mentioned above, PEX Analyzer can use a combination of trace events to derive additional details

about certain events. If a TRACE includes MI CALLs/RETURNs and the two database events, PEX Analyzer reports the following:

- Elapsed time each OPEN, CLOSE, or LDIO took (microsecond granularity)
- CPU time each OPEN, CLOSE, or LDIO took (microsecond)
- Number and type of physical DASD operations each OPEN, CLOSE, and LDIO caused
- Causing program name (with an attempt to determine application versus system program name)

Local and remote (DDM) change and retrieve data area is similar to the database events. It collects information on:

- Tag that indicates the job requesting the data area operation
- Time of day (microsecond granularity)
- Data area and library names
- Detailed operation type information (change local, change DDM, retrieve local, and so on)
- Data area type (char, decimal, logical)
- Up to the first 20 bytes of the changed/retrieved data is also collected (although this is not reported on by PEX Analyzer)

Like the database events, the data area events can be combined with MI CALLs/RETURNs to derive **additional details**; for example:

- Elapsed time each data area operation took (microsecond granularity)
- CPU time each data area operation took (microsecond)
- Number and type of physical DASD operations for each operation
- Causing program name (with an attempt to determine application versus system program name)

Local and remote (DDM) send and receive data queue is also similar to the database events. It collects the following information:

- Tag that indicates the job requesting the data queue operation
- Time of day (microsecond granularity)
- Data queue and library names
- Detailed operation type information (send local, send DDM, receive keyed local, receive keyed DDM, and so on)
- Key length, message length, message, and key data are also collected, but not reported on by PEX Analyzer

Like the database events, the data queue events can be combined with MI CALLs/RETURNs to derive the following **additional details:**

- Elapsed time each data area operation took (microsecond granularity)
- CPU time each data area operation took (microsecond)
- Number and type of physical DASD operations for each operation
- Causing program name (with an attempt to determine application versus system program name)

Note: PEX Analyzer analyzes PEX data that has been previously collected on the current or some other iSeries. The following is an obvious statement, but needs to be made: **PEX Analyzer can analyze only data that has been collected**. For example, if the PEX collection did **not** include database file activity, PEX Analyzer cannot produce reports on database file activity. It is important to understand that it is at the time of PEX data **collection** that the decision is made concerning which types of data will be collected. The decision about what data is collected controls the type of analyses available.

PEX Analyzer analyses can be used to do additional job and program performance analysis. In cases where other performance tools usage cannot identify an application, set of jobs, or programs, PEX Analyzer can be used system-wide to assist in identifying applications or programs that need further analysis.

Table of Contents | Previous | Next

3.3 PEX Background Information

Performance analysis based on PEX Trace is designed to answer questions at a level lower than the iSeries Performance Monitor reports can provide.

Note: The following examples refer to a particular type of PEX TRACE event: DASD operations. This is just one of many PEX TRACE types that can be analyzed with iDoctor for iSeries tools.

For example, the Performance Monitor can report that DASD unit U is performing X operations per second with an average I/O response time of T. PEX trace of DASD operations gathers detailed data about **every** DASD operation. Specifically, it can report the following:

- What objects were being paged into and out of main storage/DASD
- What object **types** were being paged into and out of main storage/DASD
- How many DASD sectors were read/written on each operation
- What job or system task was performing the DASD operations
- How long individual DASD operations took to complete
- What DASD units and main storage pools were involved with the DASD operations

A typical scenario involving the Performance Monitor and PEX trace includes the following:

- Performance Monitor sample data shows that between 7:05 and 7:30 PM (on most days) DASD units 45, 46, and 54 are much busier than all other units
- A PEX trace collection is made between 7:10 and 7:20 PM on 1 or 2 typical days. The PEX definition and resulting collection include the following types of trace events:
  -- Logical DASD/Main storage operation starts
  -- Physical DASD operation starts
  -- Possibly all MI program CALLs/RETURNs
- The PEX collection is analyzed on the collecting machine or Save/Restored to a different AS/400 system at the same version and release.
- The iDoctor for iSeries tool is used to analyze and graph such items as:
  -- DASD operations by disk unit and causing jobs/tasks
  -- DASD operations by paging object names
  -- DASD operations by paging object types
  -- DASD operations by DASD unit

In summary, a graphical presentation is created showing what jobs/tasks and objects are making units 45, 46, and 54 busier than usual.

Table of Contents | Previous | Next

3.4 Overview of Collecting PEX Data

The basic steps for creating a PEX Collection are the following:

1. Creating or choosing an existing **PEX definition**
2. Starting a collection using the definition.
3. Ending the collection and directing the data to be discarded or stored in a library.

If an iSeries has the iDoctor for iSeries PEX Analyzer component installed, the user has the following choices when making a PEX collection:

- The **manual** method of using the individual QSYS commands including the Add PEX Definition (**ADDPEXDFN**), Start PEX, (**STRPEX**), and End PEX (**ENDPEX**) commands.
- The **automatic** method by using the single command named **STRPACOL** in library QIDRPA

Note: The PEX Analyzer client provides interfaces for creating and/or changing PEX definitions. The client also has an interface for creating a collection, viewing it as runs, and then analyzing its results. See the client side documentation for PEX Analyzer for more information.

Table of Contents | Previous | Next

3.5 PEX Definitions

A **PEX definition** is a member in a specific system database, and it controls most of the aspects of making a PEX collection except the following:

- When to begin making the collection
- When to end the collection
- The library where collection data is stored

A PEX definition also controls:

- Which subset of the hundreds of trace event types are to be activated
- Granularity of CPU sampling
- Maximum amount of data to be collected
- Subset of jobs or system tasks (or all jobs and all tasks) that are to be traced

This document does not go into detail of the very complex **ADDPEXDFN** command. However, the types of controls and data that are contained in a PEX definition will be described.

The **ADDPEXDFN** command and resulting PEX definition contain keywords and data that control the following:

- Definition name
- What subset (or all) of jobs and/or system tasks are to participate in the tracing
- Maximum amount of DASD space that should be consumed to hold the trace event data
- A CPU profiling interval (in milliseconds)
- The exact set of specific trace events that are to be active (more detail below)

There are dozens of PDC/PEX TRACE event types. A subset of them are **analyzed** by iDoctor for iSeries PEX Analyzer. The TRACE event types known by iDoctor for iSeries are:

- MI program CALLs and RETURNs
- CPU profiling tics
- Logical DASD I/O start requests

- Physical DASD operation starts and completions
- DASD space change requests
- Database file full OPENs and CLOSEs
- Logical database file I/Os
- Data area change and retrieves
- Data queue sends and receives
- Task switch trace
- IFS events

Note: The PEX Analyzer client provides interfaces for creating and/or changing PEX definitions. The client also has an interface for creating a collection, viewing it as runs, and then analyzing its results. See the client side documentation for PEX Analyzer for more information.

Table of Contents | Previous | Next

3.6 Commands

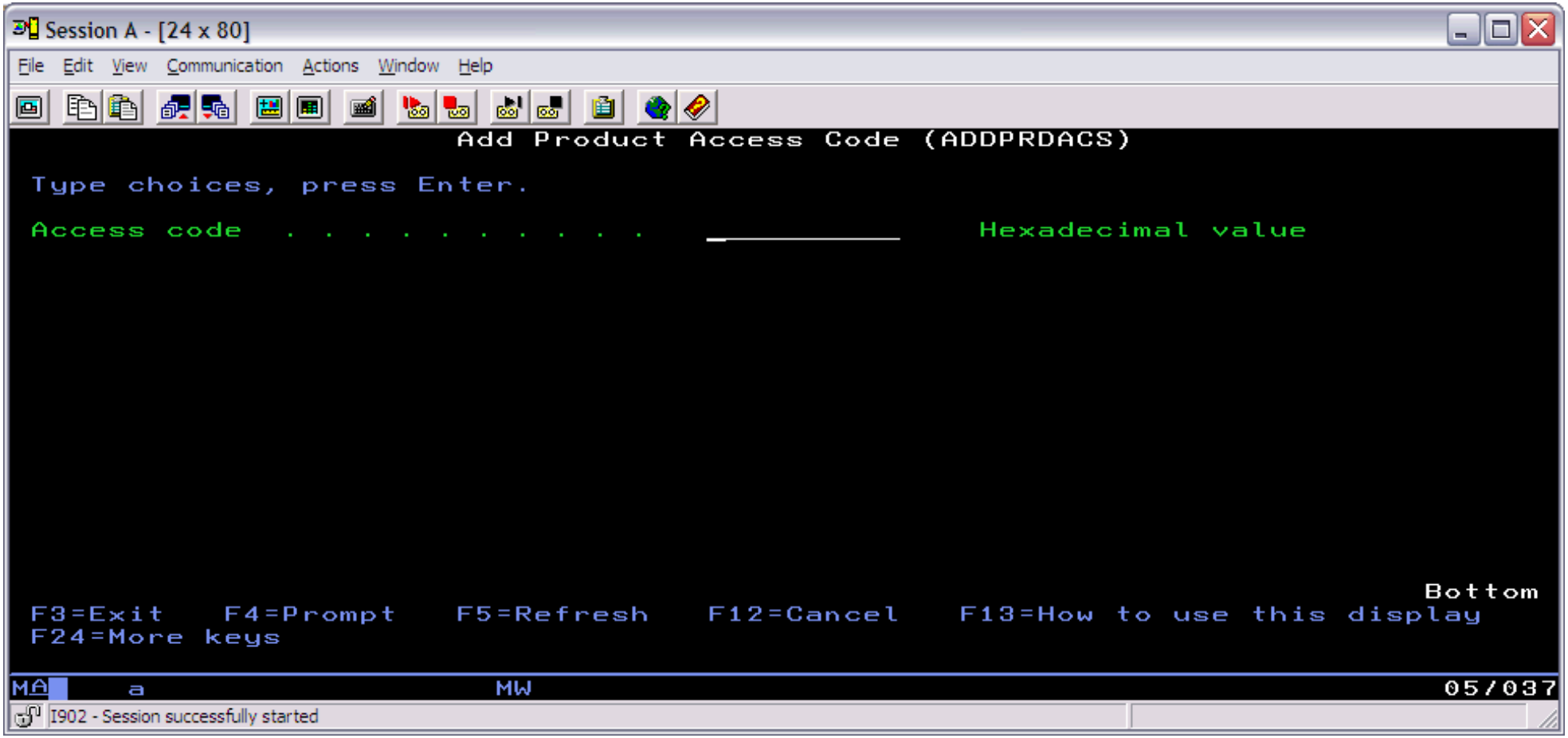
This section discusses the green screen commands that are shipped with PEX Analyzer in library QIDRPA.

Table of Contents | Previous | Next

3.6.1 ADDPRDACS

This command is used to authorize users of the system to an iDoctor component.

Access codes are granted through the website which this product was downloaded from: http://www-912.ibm.com/i_dir/idoctor.nsf



ADDPRDACS Parameters

ACSCDE - Access code
The access code provided by IBM to grant use of an iDoctor component on the current system.

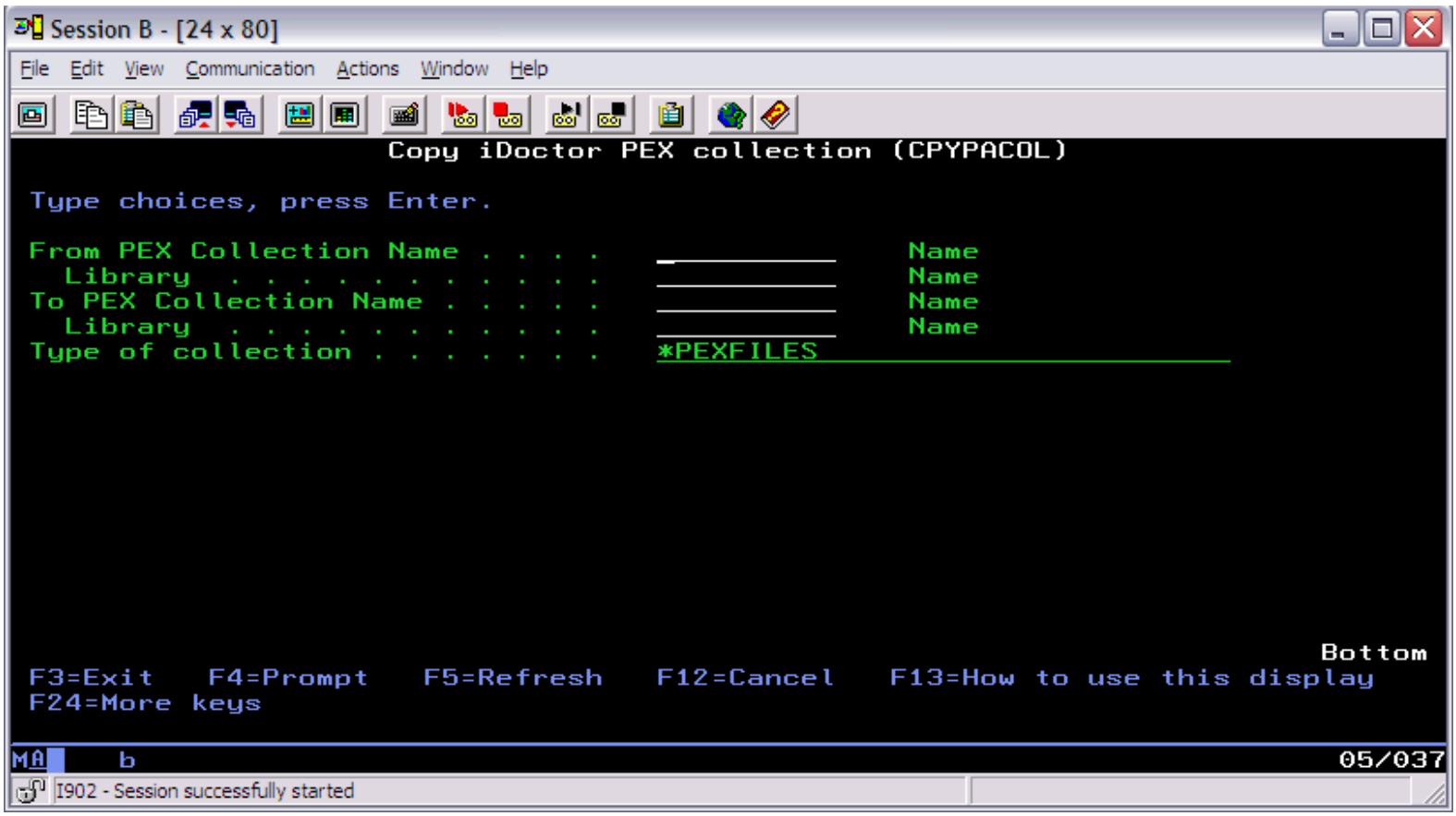
Table of Contents | Previous | Next

3.6.2 CPYPACOL

This command is used to copy a PEX collection that may or may not have been created with the STRPACOL command, from one library to another.

The STRPACOL command will create a PEX collection but may optionally include several SMTR* files in the user's library. If these files exist, they will be copied by the CPYPACOL command along with the rest of the collection data. This command also supports the copying of single object PEX collections which are stored as management collection objects.

This command does NOT copy PEX Analyzer's analysis files.



CPYPACOL Parameters

FROMCOL - From collection name
Specify the name of the collection you wish to copy.

FROMLIB - From collection library name
Specify the name of the library containing the collection you wish to copy.

TOCOL - To collection name
Specify the name to give the new collection that will be copied.

TOLIB - To collection library name
Specify the name of the library to copy the collection to.

TYPE - Type of collection
This value determines the type of PEX collection that will be copied.

*PEXFILES
This is the normal type of PEX collection that exists in the QAYPE* files in the user library.

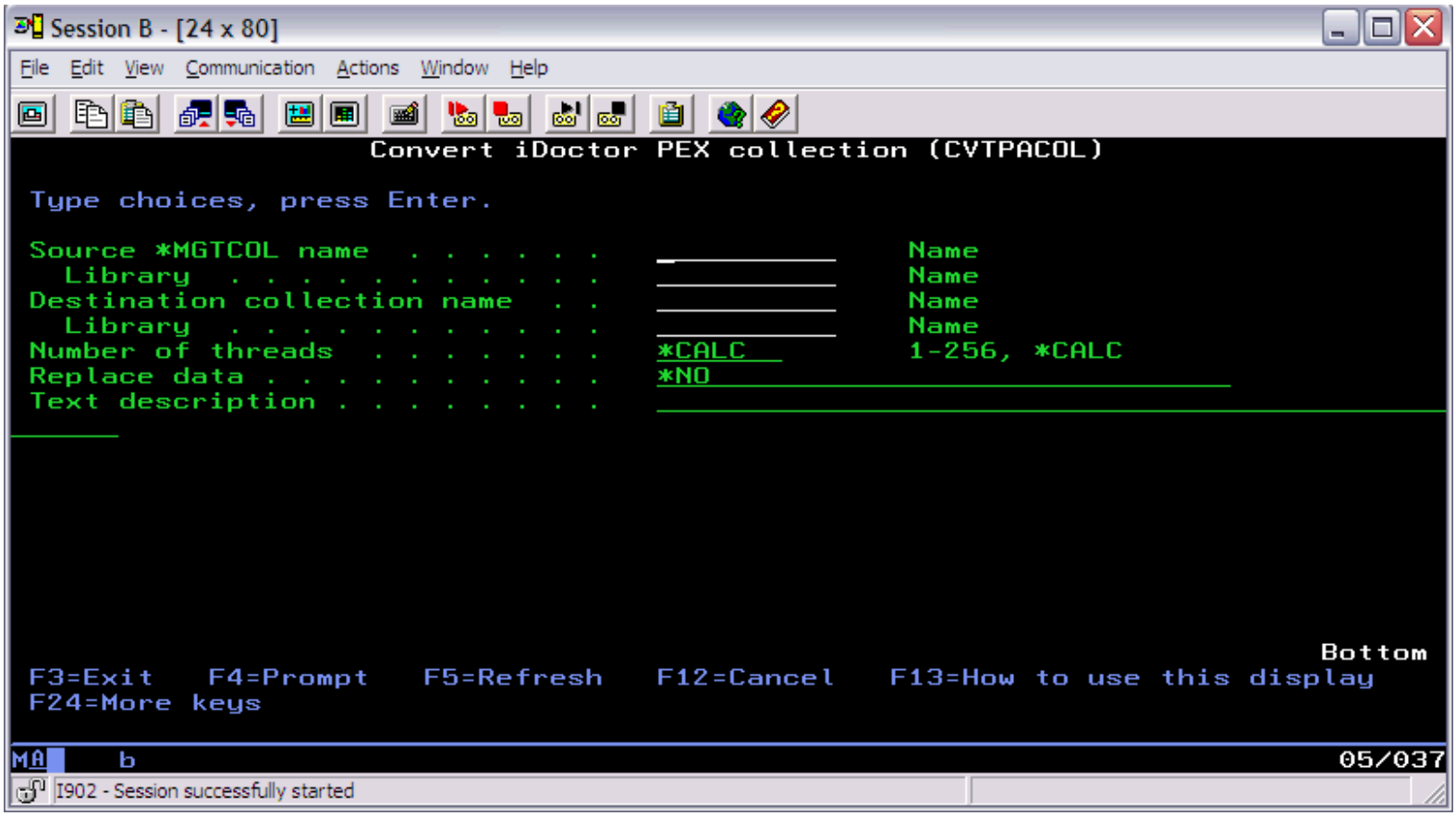
*SINGLEOBJ
A single object collection is a *MGTCOL object that contains all of the data of a PEX collection but is in an unuseable format until it is split out into the QAYPE* files.
Use the command CVTPACOL to split the *MGTCOL object into useable QAYPE* files.

Table of Contents | Previous | Next

3.6.3 CVTPACOL

This command is used to convert a PEX collection from a *MGTCOL object format to the useable QAYPE* collection file format.

During the creation process the ENDPEX command is called in order to produce the PEX output files. The 'number of threads', 'replace data' and 'description' parameters on the CVTPACOL command are passed to the ENDPEX command during the creation process.



CVTPACOL Parameters

FROMCOL - Source *MGTCOL name
Specify the name of the *MGTCOL object that will be converted to a useable PEX collection.

FROMLIB - Source *MGTCOL library
Specify the name of the library containing the collection that will be converted to a useable PEX collection.

TOCOL - Destination collection name
Specify the name to give the new collection that will be created.

TOLIB - Destiniation library name
Specify the name of the library to contain the new collection.

NBRTHD - Number of threads
Specifies the number of concurrent threads that the ENDPEX command uses to process the data in the session being ended. Specifying a number greater than 1 allows the ENDPEX command to take advantage of available CPU cycles, especially on a multi-processor system. While this may speed up the command processing, it may also degrade the performance of other jobs on the system. You can minimize this impact by changing the priority of the job that runs the ENDPEX command to a higher number. You should also verify that the disk subsystem can handle the additional threads.

*CALC
The system calculates a reasonable number of threads to do the command processing which does not use excessive CPU or disk resources.

*MAX
The system calculates a maximum number of threads to do the command processing. An attempt will be made to maximize utilization on all resources to minimize processing time. This may cause severe degradation for all other jobs on the system.

number-of-threads
Specify the number of threads for the ENDPEX command to use to process the collected data.

RPLDTA - Replace data
Specifies whether to replace the data in an existing collection with the new PEX collection data. The possible values are:

*NO
If a collection already exists with the same name, an error message is sent to the user. This prevents the user from inadvertently writing over existing data.

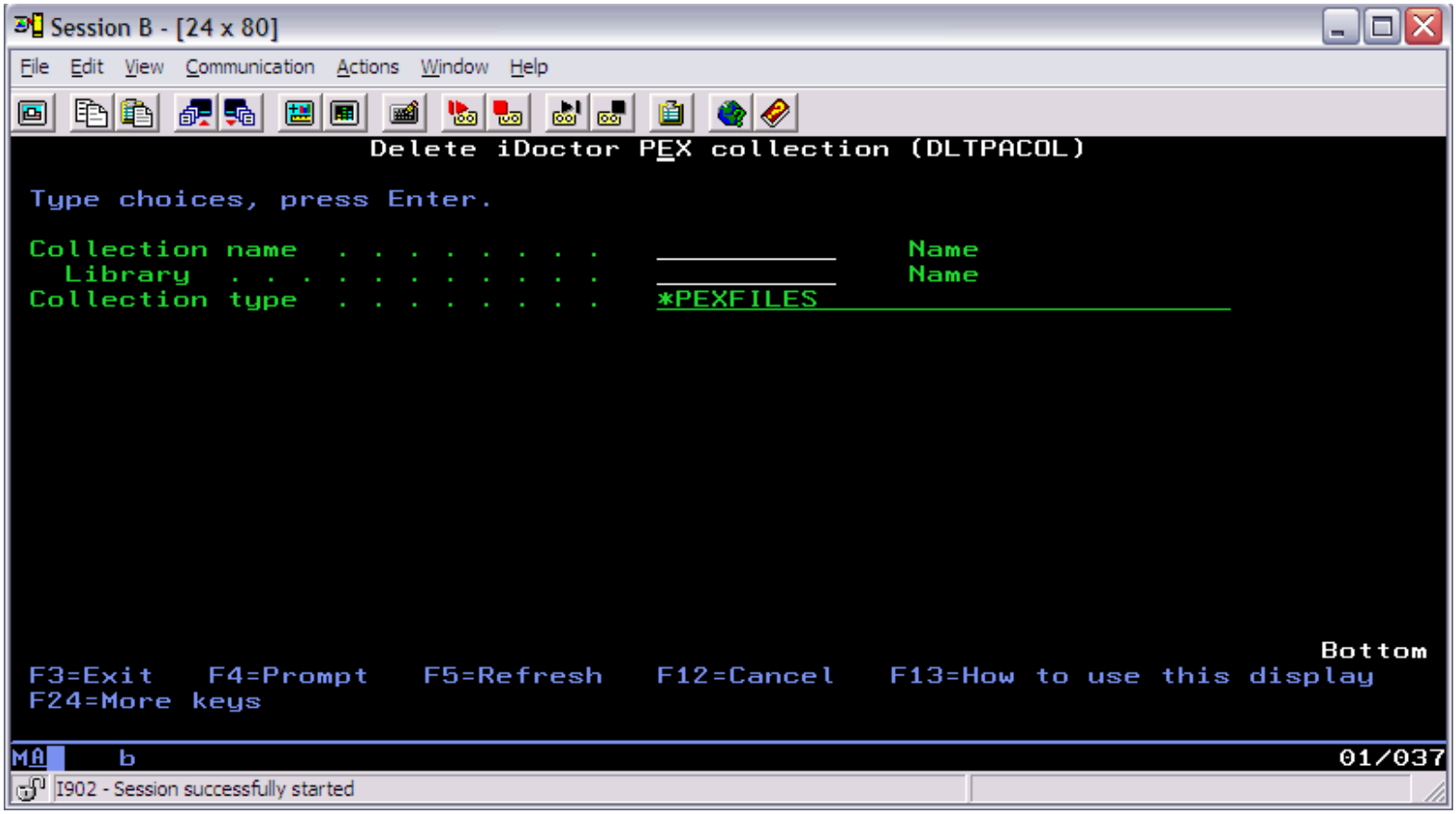
*YES
If a collection already exists with the same name, the old data is lost and is replaced by the new data.

TEXT - Text description
Specifies the text that briefly describes the type of data collected. Specify no more than 50 characters of text, enclosed in apostrophes.

Table of Contents | Previous | Next

3.6.4 DLTPACOL

This command is used to delete a PEX collection from a user's library and all the iDoctor collection and analysis files associated with it.



DLTPACOL Parameters

COLNAME - Collection name
This is the name of the collection that will be deleted from the user library.

LIB - Collection library
Indicates the library name which contains the PEX collection to be deleted.

TYPE - Collection type
This value determines the type of PEX collection that will be deleted.

*PEXFILES
This is the normal type of PEX collection that exists in the QAYPE* files in the user library.

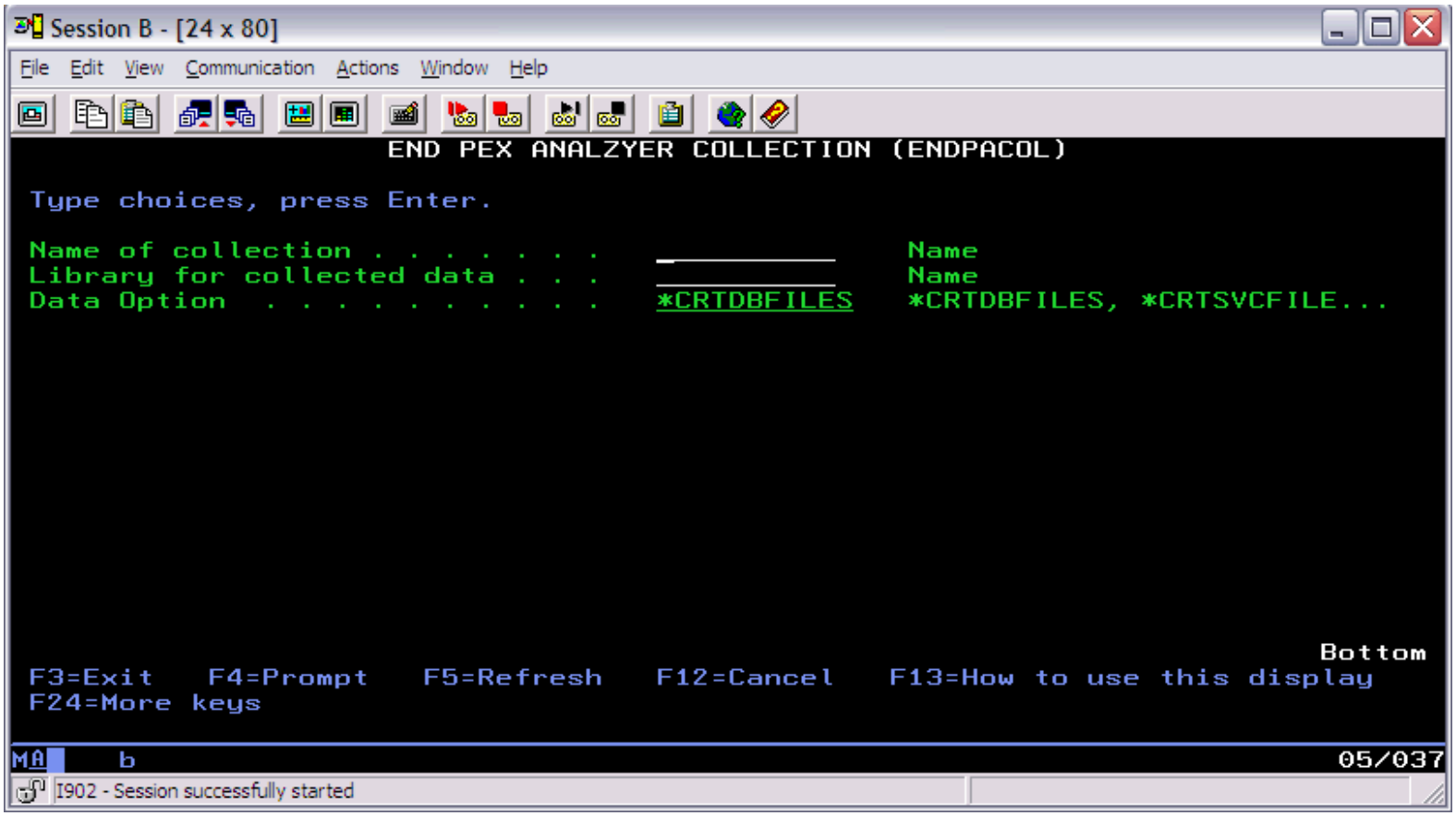
*SINGLEOBJ
A single object collection is a *MGTCOL object that contains all of the data of a PEX collection but is in an unuseable format until it is split out into the QAYPE* files.
Use the command CVTPACOL to split the *MGTCOL object into useable QAYPE* files.

Table of Contents | Previous | Next

3.6.5 ENDPACOL

This command is used to end a PEX collection that was started with the STRPACOL command, prior to the expiration of the provided time value. ENDPACOL has no effect once the time value provided on the STRPACOL has expired.

This command can also be used to override the option on the STRPACOL command for how the data is stored.



ENDPACOL Parameters

COLNAME - Name of collection
Specify the name of the collection you wish to end.

DATALIB - Library for collected data
The library you specify here must be the one which was supplied on the STRPACOL command. Depending on the value supplied to the DATAOPT parameter on this command, any data that is kept will be put into this library.
DATAOPT - Data option
The value entered here determines the actions taken for this collection.

*CRTDBFILES
Data is collected and put into the QAYPE* files, ready for data analysis commands.

*CRTSVCFILE
Data is collected and put into a single file which is easily transported to another system at the same release where data analysis can be performed.

*DELETE
The collection of data is stopped and any data already collected is discarded.

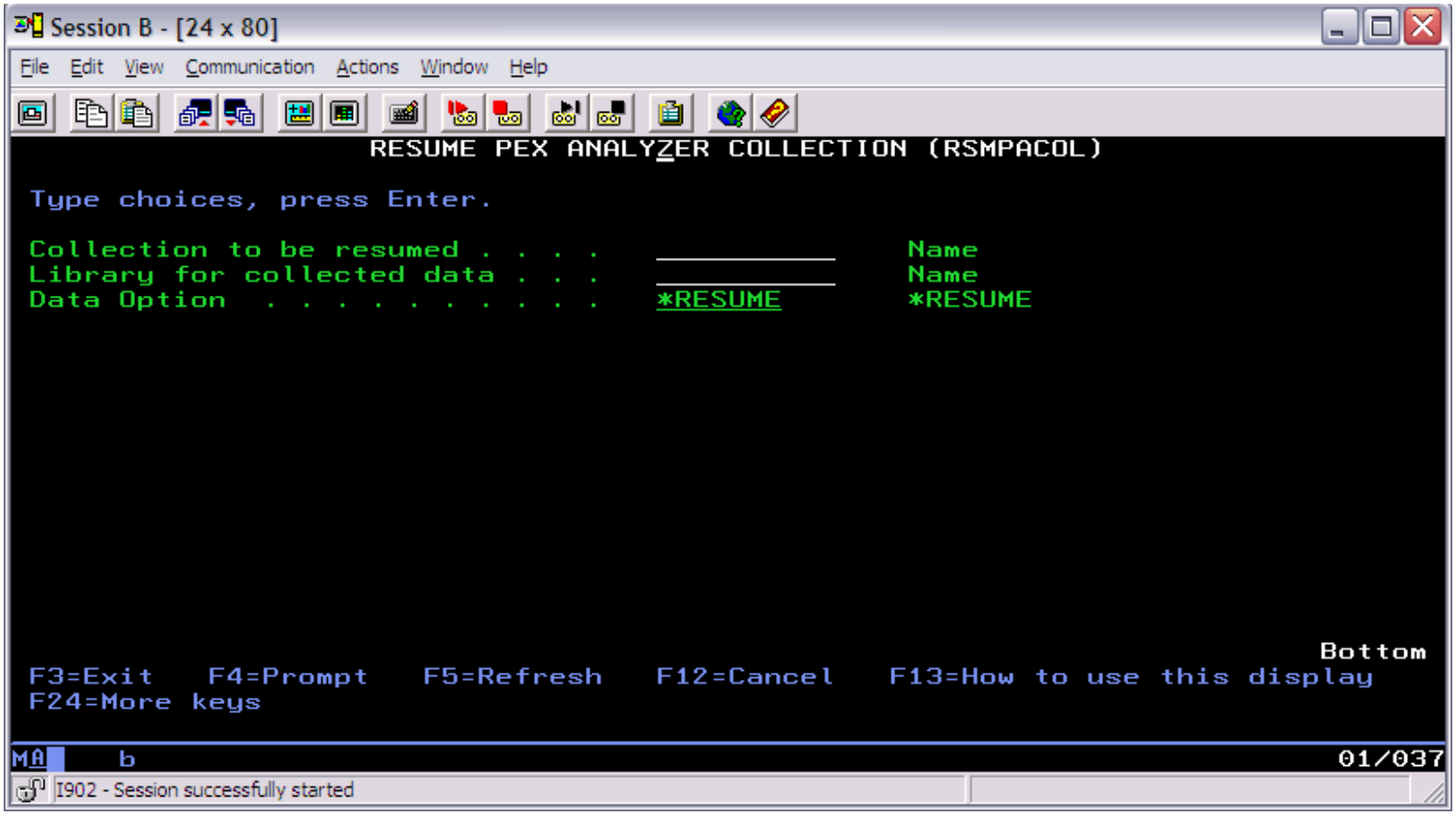
*RESTART
The collection is stopped, any data that was collected is discarded, then the collection is restarted with the same values originally provided on the STRPACOL command.

*STOP
The collection is stopped. It will remain in a stopped status until another action is requested from the ENDPACOL command.

Table of Contents | Previous | Next

3.6.6 RSMPACOL

This command is used to resume an iDoctor collection that was started with the STRPACOL command with the STNDBY parameter set to a value of Y.



COLNAME - Collection to be resumed
Specify the name of the collection you wish to end.

DATALIB - Library for collected data
The library you specify here must be the one which was supplied on the STRPACOL command. Depending on the value supplied to the DATAOPT parameter on this command, any data that is kept will be put into this library.
DATAOPT - Data option
The value entered here determines the actions taken for this collection.

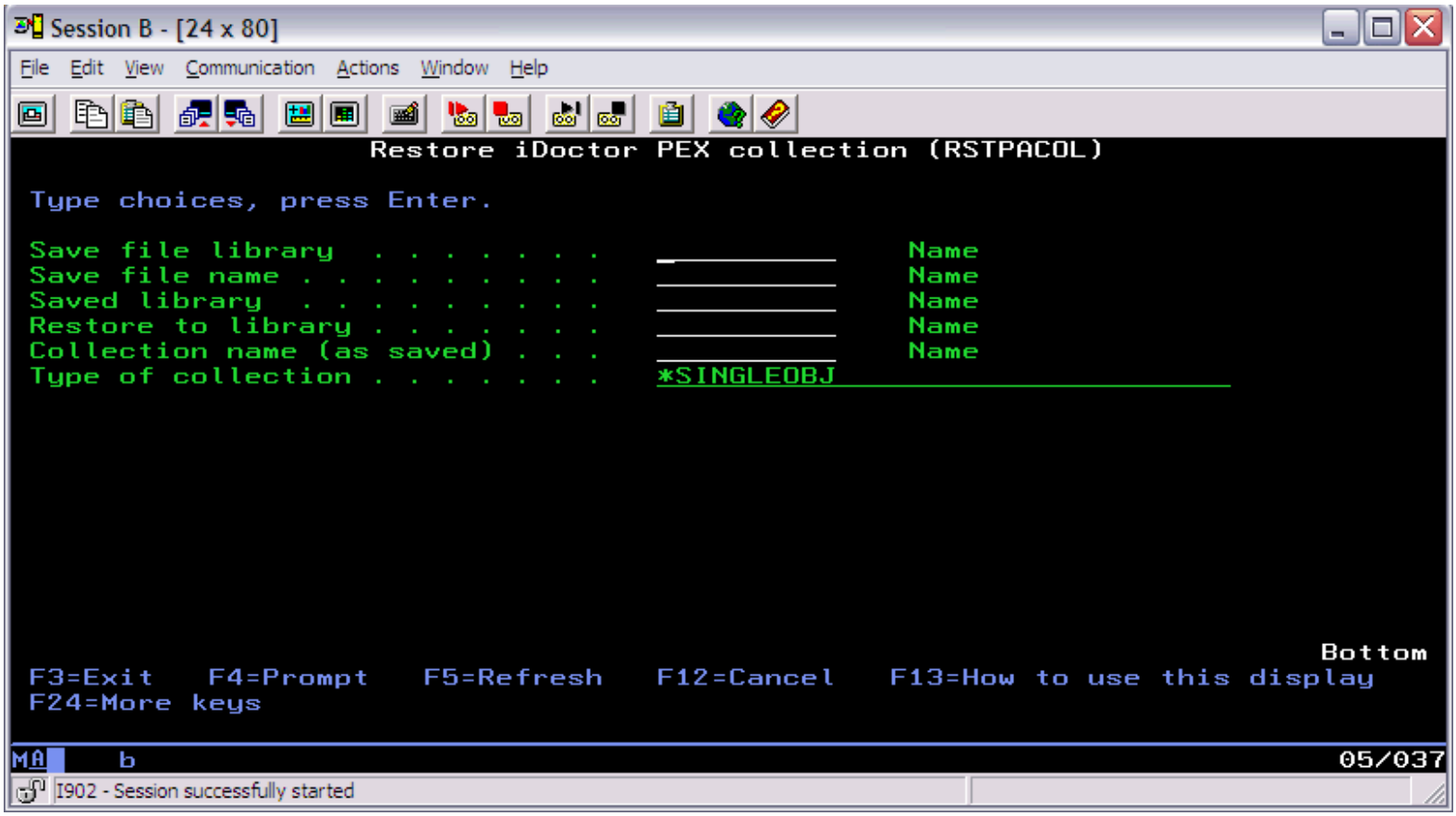
*RESUME
Collection of iDoctor data to begin.

Table of Contents | Previous | Next

3.6.7 RSTPACOL

This command is used to restore a PEX collection that has been saved using the SAVPACOL command or sent from another system using the FTPPACOL command.

This command will restore the collection to a library on the current system. The name of the PEX collection that was saved must be provided to this command.
This command also supports the restoring of single object PEX collections which are stored as *MGTCOL objects.



RSTPACOL Parameters

SAVFLIB - Save file library
The library where the save file exists that contains the PEX collection to be restored. The save file must be created using the SAVPACOL command (or indirectly using the FTPPACOL command).

SAVFNAME - Save file name
The name of the save file containing the PEX collection to be restored. This save file must be created using the SAVPACOL command.

SAVEDLIB - Saved library
Indicates the name of the library where the collection was saved. If this is not known, use the DSPSAVF command to determine this library.

TOLIB - Restore to library
The name of the library to restore the collection to.

TOCOL - Collection name (as saved)
The name of the PEX collection within the save file being restored.

TYPE - Collection type

This value determines the type of PEX collection that will be restored.

*SINGLEOBJ
A single object collection is a *MGTCOL object that contains all of the data of a PEX collection but is in an unuseable format until it is split out into the QAYPE* files.

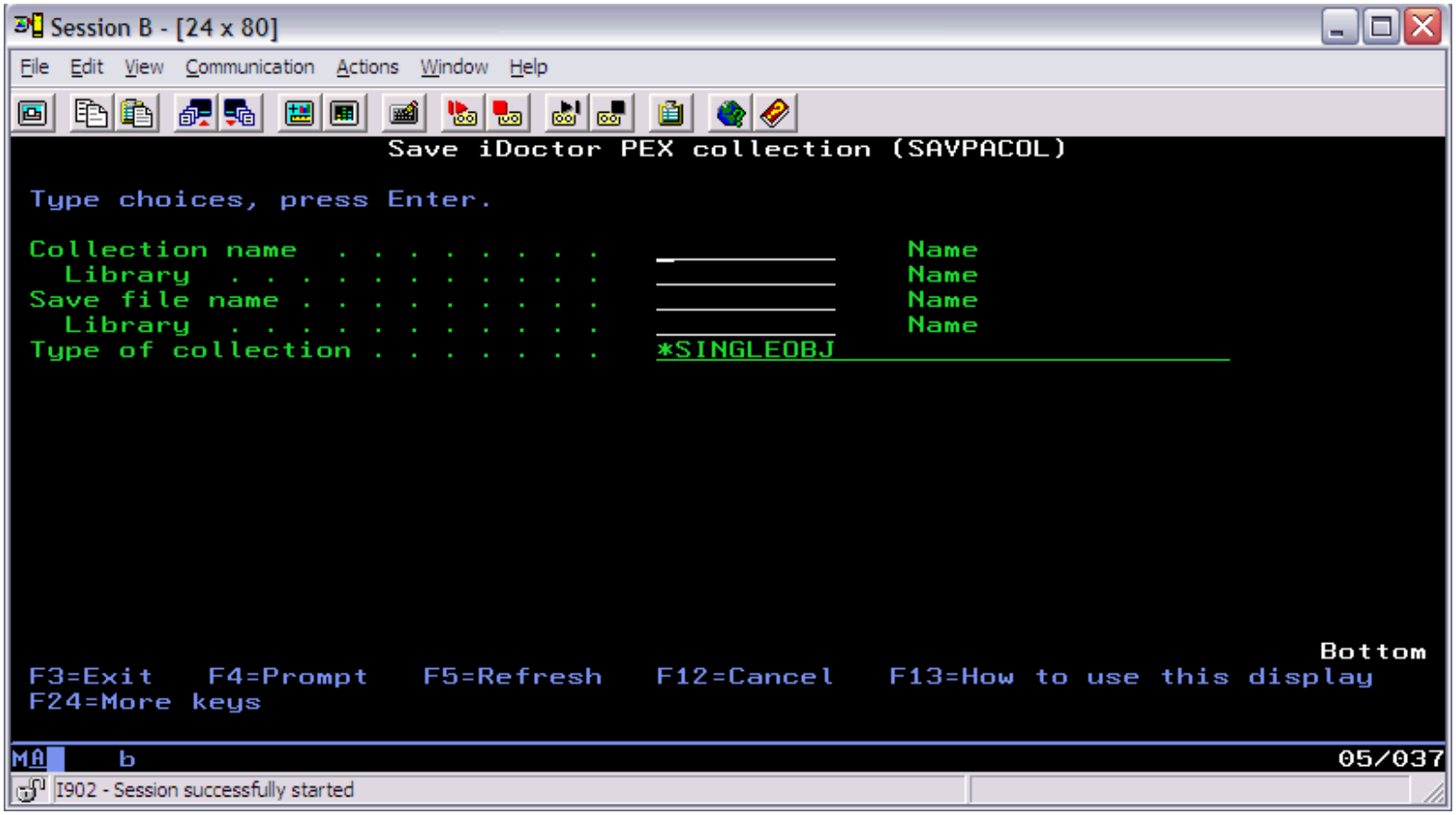
*PEXFILES
This is the normal type of PEX collection that exists in the QAYPE* files in the user library.
Use the command CVTPACOL to split the *MGTCOL object into useable QAYPE* files.

Table of Contents | Previous | Next

3.6.8 SAVPACOL

This command is used to save a PEX collection into a save file.

Use the RSTPACOL command to restore a collection saved by this command.



SAVPACOL Parameters

COL - Collection name
Specify the name of the collection you wish to save.

LIB - Library name
Specify the name of the library containing the collection you wish to save.

SAVFNAME - Save file name
Specify the name of the save file the collection will be saved into.

SAVFLIB - Save file library name
Specify the name of the library where the save file exists.

TYPE - Type of collection
This value determines the type of PEX collection that will be saved.

*PEXFILES
This is the normal type of PEX collection that exists in the QAYPE* files in the user library.

*SINGLEOBJ
A single object collection is a *MGTCOL object that contains all of the data of a PEX collection but is in an unuseable format until it is split

out into the QAYPE* files.
Use the command CVTPACOL to split the *MGTCOL object into useable QAYPE* files.

Table of Contents Previous Next

3.6.9 STRPACOL

Reminder:

- PEX collections can be created on one iSeries and saved or restored to another for analysis by iDoctor for iSeries. However, both systems **must** be at the same version/release (i.e. V5R1, V5R2, or V5R3)

The commands used to make a PEX collection are:

Add PEX Definition (**ADDPEXDFN**): Creates a PEX definition
Start PEX (**STRPEX**): Start a PEX collection
End PEX (**ENDPEX**): End a PEX collection

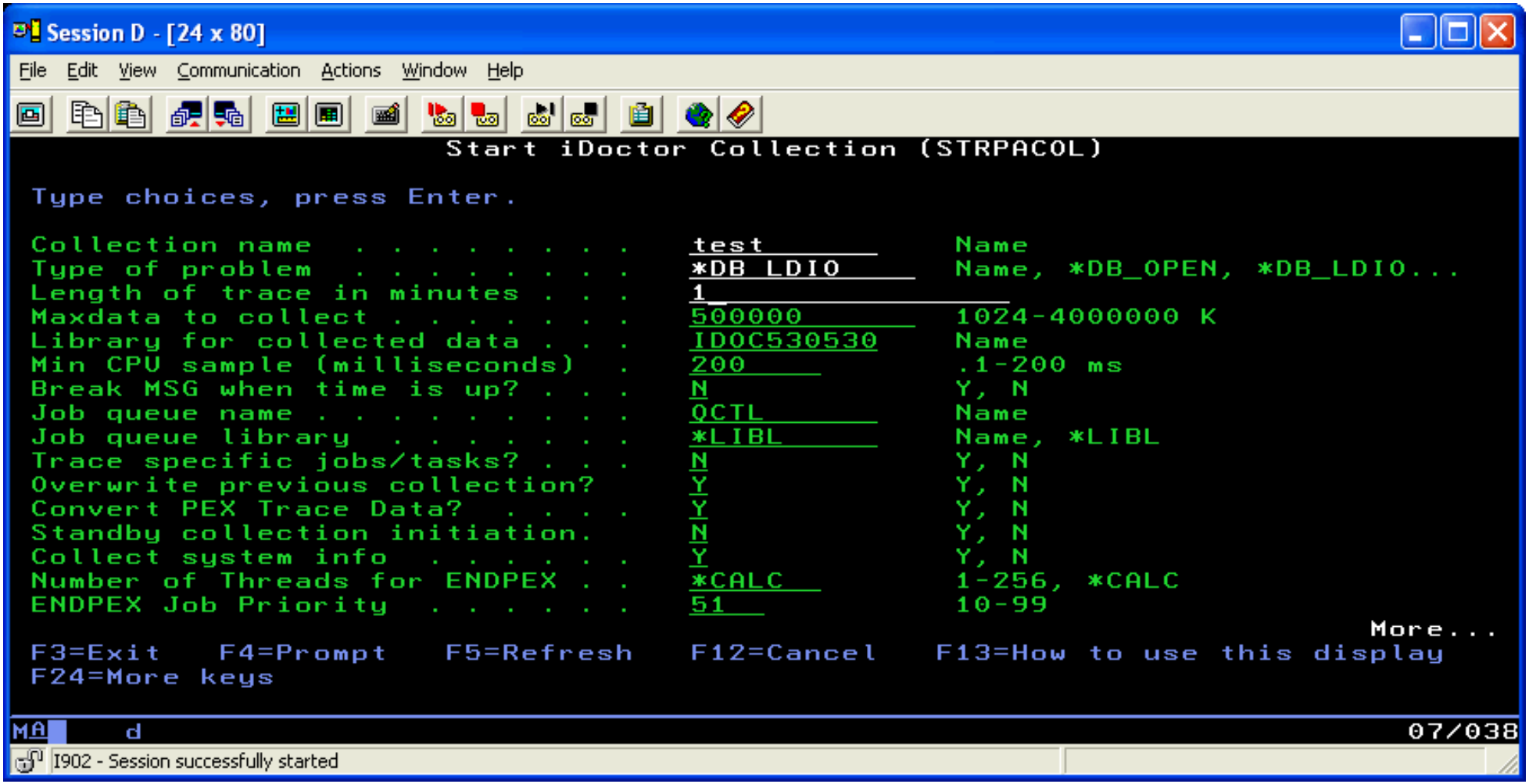
The above commands may be used, however PEX Analyzer includes an ease-of-use feature that incorporates all of the above commands into a single command:

QIDRPA/STRPACOL

This command does the following:

- Creates the necessary PEX definition (deleting and replacing one by the same name, if necessary)
- Starts the PEX collection
- Ends the PEX collection after a predetermined amount of time and directs the data to a predetermined library
- Periodically issues the Work with System Status (**WRKSYSSTS**) and Work with Disk Status (**WRKDSKSTS**) commands and copies output of each to files named SMTRSTS and SMTRDTS in the target library

A prompt of the Start iDoc Collection (**STRPACOL**) command is shown below:



STRPACOL Command

In the following **STRPACOL** section, it is recommended to **use the default values** in most cases.

The value specified for the PROBLEM keyword is used to build a PEX definition that collects the MINIMUM types of events necessary to enable report generation of that type. The less events collected, the longer the collection can be run.

When the PROBLEM parameter chosen results in a TRACE collection (as opposed to a STATS or PROFILE collection), it is possible to add to the events collected by use of the ADDEVTS keyword.

STRPACOL Parameters

COLLECTION - name of the collection
The value provided here will be used to uniquely identify the PEX collection within the specified library.

PROBLEM - type of problem for which data is being collected
The value entered here causes a PEX definition to be created with predetermined values. The possible responses are:

***DB_OPEN**
Used to determine jobs or programs that are performing excessive file opens. This does not include IFS files.

Suggested maximum value for the DURATION parameter: 30 minutes

***DB_LDIO**
Used to determine the rates at which jobs or programs are performing logical data base I/O's. This does not include IFS files.

Suggested maximum value for the DURATION parameter: 10 minutes

***DTAQ_IO**
Allows you to determine the rates of logical I/O's against your Data Queues.

Suggested maximum value for the DURATION parameter: 30 minutes

***DTAARA_IO**
Allows you to determine the rate of logical I/O's against your Data Areas.

Suggested maximum value for the DURATION parameter: 30 minutes

***PDIOCOUNT**
Allows you to determine the rates of physical I/O's jobs for Jobs, Objects, Disk Units.

Suggested maximum value for the DURATION parameter: 15 minutes

***PDIOTIME**
Allows you to determine the rates of physical I/O's for Jobs, Objects, Disk Units, and the time values associated with these I/O's.

Suggested maximum value for the DURATION parameter: 10 minutes

***TASKSWITCH**
Allows you to determine how a job spends it's time within the system during the collection interval.

Suggested maximum value for the DURATION parameter: 5 minutes

***OBJ_NETSIZE**
Allows you to determine where storage is being used within the system during the collection interval.

Suggested maximum value for the DURATION parameter: 4 hours

***PROFILE**
User specifies from 1 to 16 programs and/or service programs for CPU profiling PEX PROFILE data collection. This collection type helps identify CPU 'hot spots' within the selected programs/service programs. The user specified pane size (default of 4) is used to logically divide each program/service program up into 'panes' for granularity of analysis. When CPU is used by a program/service program in the collection data is captured by pane within program/service program.

***STATSFLATY**
This PEX STATISTICAL collection captures CPU and IO statistics by program/procedure/method. Procedure/method statistics are only provided for programs/service programs with an ENDPFRCOL value other than *NONE or *PEP. One collection-wide count of both CPU and IO in-line and cumulative values is stored for every program/procedure/method that was called (program/procedure/method statistics for all jobs/tasks in the collection are merged).

***STATSHIER**
This PEX STATISTICAL collection captures call flow, CPU and IO statistics by program/procedure/method by calling program/procedure/method. Procedure/method statistics are only provided for programs/service programs with an ENDPFRCOL value other than *NONE or *PEP. One count of both CPU and IO in-line and cumulative values is stored by calling program/procedure method for every program/procedure/method that was called by the job. We recommend that you try to keep the number of jobs in a collection of this type to less than 10.

***TPROF**
This PEX TRACE collection type helps identify system-wide CPU 'hot spots' within application programs/service programs, OS/400 programs/procedures/methods and SLIC procedures. Unlike *PROFILE, the user does not need to specify program names. The ENDPFRCOL setting is not used to determine whether a program/procedure/method will be recorded.

***STATSFLATN**
This PEX STATISTICAL collection captures CPU and IO statistics by program/procedure/method. Procedure/method statistics are only provided for programs/service programs with an ENDPFRCOL value other than *NONE or *PEP. One count of both CPU and IO in-line and cumulative values is stored for every program/procedure/method that was called by job (program/procedure/method statistics for jobs/tasks in the collection are stored separately for each individual job-thread or task). We recommend that you try to keep the number of jobs in a collection of this type to less than 10.

***TAPE_IO**
Collects information on commands issued to the tape device for use in diagnosing tape performance issues. This function can be run for an extended period of time. You may require the additional information provided by the *DASDSTR and *DASDEND options to diagnose the problem.

***ILEACTGRP**
Collects information on the creation of activation groups and the destruction of activation groups. This can aid you in determining if there are issues within your jobs normal execution affected by the creation or destruction, such as what is the performance cost associated with their creation/destruction, etc. This can be useful in squeezing out that last bit of application performance. In addition, for those problem types which collect trace type data, you will be presented with an additional parameter ADDEVTS(14), that allows you to extend the data collections capabilities.

***HEAPSTG**
Collects information about the creation and destruction of heap allocations on your system. This option can collect a lot of events in a very short period of time, particularly on systems running newer workloads.

***WEBSPHERE**
Collects information about WebApp/Servlet's, Container Transactions, EJB/EJB methods, Database Connection Pool, EJB Object Pool, and ORB Thread Pool. This option could collect a lot of events in a very short period of time. It is advisable to only run this trace for short periods of time, 5-10 minutes.

***USR_TRNSACT**
Collects information on CPU usage, response time, I/O activity, and Seize/Lock activity for those applications which are using the Transaction API's. The length of time you can collect data for will vary with the numbers of jobs running the application which is using the Transactions API's. Most likely you will be able to collect data for 20-30 minute intervals.

***IFSEVTS**
Collects information whenever an IFS file is accessed, and when an IFS file is opened or closed. The length of time you can collect data for will vary with the numbers of jobs over which data is being collected which are actually using the Itegrated File System.

***DOMINOTRC**
Collects information about Domino server activity. The data which is collected is based on the value supplied to DEBUG_OS400_PEX in the NOTES.INI file. The length of time you can collect data for will vary with the number of Domino server jobs included in the trace collection and the kind of data which is being collected.

These terms were used in the previous information:

LDIO (Logical Data Base I/O): Actual calls to OS/400 database run-time routines made by any MI program (application or system). Example: GET-BY-KEY-DATA, GET-BY-GET-NODATA, PUT-SINGLE, PUT-MULTIPLE, and so on. This term is also expanded within PEX Analyzer to include data area and data queue operations. For example, a program calls to QSNDDTAQ or the issuing of the CHGDTAARA commands are termed data queue LDIOs and data area LDIOs.

PDIO (Physical DASD I/O): Physical (real) DASD reads and writes.

DURATION - Length of trace in minutes
The supplied value, in minutes, is used to control the amount of time spent collecting events. This is the time from STRPEX to ENDPEX. The job this command submits, QIDRPACOL, will run for a much longer time than the value supplied on the DURATION parameter because this job is doing the ENDPEX function, which can be long running.

This is probably the most difficult keyword to explain and is the most difficult to give a recommended value for. The size of a collection and how long it takes to process are directly proportional to the number of events in a collection. In turn, this number is controlled by:

- Type of events that are active
- How long the collection is active
- How **busy** the machine is

General DURATION Recommendations

The following are based on tracing large (8+ processors), busy, machines.

Note: The term **busy** is used to describe how much work a system is doing. Contrast a system with 2000 interactive users at lunch time versus a night time batch run with 4 active batch jobs. It is more likely that the batch run would be busier than the lunch time interactive run.

This information is also based on the assumption that **all** jobs and system tasks are participating in the trace. If a single job or task, or even a small subset, is traced in place of all jobs and tasks, the recommended DURATION can be increased.

Of all the PROBLEM types, *OBJ_NETSIZE can run the longest. This is intentional. It could take a few hours to detect a problem in DASD space usage. **A run of 4 hours for *OBJ_NETSIZE is reasonable.** If the DASD space problem is detected in less time, run the trace for the time required.

On the other extreme, the *TASKSWITCH can produce trace events at a particularly high rate. **A run of 2 minutes for *TASKSWITCH is reasonable.**

For the remainder of the PROBLEM types, a duration of 5 to 10 minutes is reasonable. A longer time may be required for types that collect less data (for example, *PDIO_COUNTS with ADDEVTS(*NONE) . A shorter time may be sufficient for types that collect more data (*DB_LDIO with ADDEVTS(*DASDSTR *DASDEND)).

The addition of ADDEVTS can dramatically increase the number of trace events logged per second.

DURATION Considerations

Similar to **STRPFRMON ... TRACE(*ALL)**, the submitted job running the **STRPACOL** functions uses little resources (CPU, DASD I/O) during the actual trace collection. However, when the DURATION expires, (or the **ENDPACOL** command is used, see below) and the trace data is being surfaced into DB2/400 PEX database files, the job can and will use considerable resources. In addition, the elapsed time the submitted job runs can be determined.

MAXDATA -Maximum amount of data to collect, in DASD kilobytes
The default amount of space in K byes to store PEX TRACE event data is 500000K. This allocates DASD space to record the trace data and should be sufficient for as long as a TRACE collection should be run. Assume roughly 80 bytes per trace event; therefore, 500000K gives approximately 500000 x 1024 / 80 = 6 400 000 events. When MAXDATA is reached, the PDC/PEX trace goes into **suspended** mode, and trace event collection halts. This does not hurt anything but can dilute the content of some of the data. In particular, reporting of object names and types can suffer. How? During trace, collection of events that contain object references only, the address of the object is captured when the trace record is cut. When the trace is stopped and the data is surfaced to PEX, a translation of address to name/lib/type occurs. **Therefore, the longer the time between the trace event is cut and the trace is formally ended, the more likely some objects will be destroyed in the meantime, therefore, eliminating the possibility of reporting about these by name.** When MAXDATA is reached, the trace is suspended rather than ended. This address to name/lib/type translation does not occur until the trace is formally ended. **Therefore, a big suspension time due to MAXDATA being reached may reduce the number of objects that are reported on by name.**

MAXDATA works hand-in-hand with DURATION.

Generally, use the default value of 500000.

More Detail on the MAXDATA Keyword

There is an important aspect to the MAXDATA keyword. This keyword controls the maximum number of DASD bytes that are used to hold trace event data **during trace collection**. Trace events are collected and stored by LIC programs into internal, LIC storage segments. These segments are limited by MAXDATA. However, during the ending of a trace the trace event data exists in the internal LIC storage **and** in PEX DB2/400 database file members. It is not until all the trace data is surfaced into PEX files that LIC frees the internal storage. The database format of the data takes slightly more bytes/event than the LIC format. Therefore, **count on a peak DASD storage demand of approximately double the MAXDATA value (assuming that MAXDATA is reached).**

DATALIB - Library for collected data
If a library name other than IDOCDATA is specified, the library is created if it does not already exist.

CPUSAMPLE - CPU sample time, in milliseconds
The value, in milliseconds, is used to cause a process to wake up at this inteval value, and indicate what program is executing and the percentage of CPU used between this wakeup and the previous wakeup within this job. By setting this to a low value, 1-10ms, you can profile the entire system, if all jobs are within the collection, and determine what programs are using the CPU.

This event data is collected regardless of the ENBPFRCOL setting for a particular pgm.

BREAKMSG - whether to issue a break message when the time is up
This keyword allows a user to decide after the DURATION if the data should be:

- Saved and tracing stopped
- Discarded and tracing stopped
- Discarded and tracing restarted

This keyword is useful when a problem occurs intermittently (busy DASD units, lost DASD space, and so on). If it does not, the choice can be made to discard the trace data and optionally start the trace again.

Normally, N (No) is specified for this keyword, thus permitting the PEX collection to complete without operator intervention. To be prompted when the trace DURATION has been reached, specify Y (Yes). If Y is specified, an inquiry message is sent to QSYSOPR when the trace DURATION is reached. The operator must respond with one of the following options:

F = End the trace and dump the data into a single file (**Note:** This is **not** the recommended format for the data if it is to be analyzed with PEX Analyzer. Rather, use D below.)
E = End the trace and discard the collected data.
R = Discard the data and start the collection again.
D = End the trace and dump the collected data into several database files; ready for transport and/or analysis.

```
Additional Message Information

Message ID . . . . . : CPF9898 Severity . . . . . : 40
Message type . . . . . : Inquiry
Date sent . . . . . : 01/24/01 Time sent . . . . . : 16:54:37

Message . . . . : PEXTRACE: F=dump data and end trace, *CRTSVCFILE option,
E=Discard data and end trace, R=Discard data and restart trace, D=Dump data
and end trace, *LIB option.
Cause . . . . . : This message is used by application programs as a general
escape message.




Bottom
Type reply below, then press Enter.
Reply . . . . D_____________________________________________________

F3=Exit F6=Print F9=Display message details
F10=Display messages in job log F12=Cancel F21=Select assistance level
```

JOBQ and **JOBQLIB** - job queue to be used for the batch job
These keywords control what job queue is used to for running the collection in batch.

ADDJOBS - Trace specific jobs/tasks
The default of **N** (no) ensures that you are tracing all jobs and all system tasks on the system.
Specify **Y** to type an individual job name/user/number to be collected; this causes a lower collection overhead than ADDJOBS(N).

Specifying **Y** causes a selective interactive prompt (of the QIDRPA/ADDPEXDFN command) to be issued, which permits the user to direct the tracing to a subset of jobs and/or system tasks.

OVWPRVCOL - Overwrite previous collection
This option will allow the user to overwrite a previous PEX Analyzer collection with the same name in the target data library. The default of **N** (no) ensures that any existing collection data will not be destroyed unless **Y** (yes) is specified.

CVTPEX - Convert PEX Trace Data
With the default value of 'Y', the data at ENDPEX time is written to the necessary set of QAYPE* files which allows for execution of the PRTPEXRPT command and the ability to generate PEX Analyzer reports. If 'N' is supplied as the value, the data is written to a single file as a management collection object, which will later require that a CVTPACOL command be executed against this object if any data analysis is to be performed.

STNDBY - Standby mode
With the default value of 'N', the start of data collection begins as soon as the start pex message is received. With a value of 'Y' the collection object is built and the collection is in suspended status waiting infinitely on a data queue. The command RSMPACOL is used to cause a collection which has been initiated with this option, to wake up and begin collecting data.

COLSYS - Collect system information
With the default value of 'Y', WRKSYSSTS and WRKDSKSTS information is collected and saved into two files, SMTRSTS and SMTRDTS in the collection library. The member names in each file will be the same as the collection name. This information can give you insight into what the system was doing during the collection of the PEX data. Change this value to 'N' if you do not wish to collect this information. There are some circumstances where the collection start up time will be significantly reduced by changing this to 'N'.

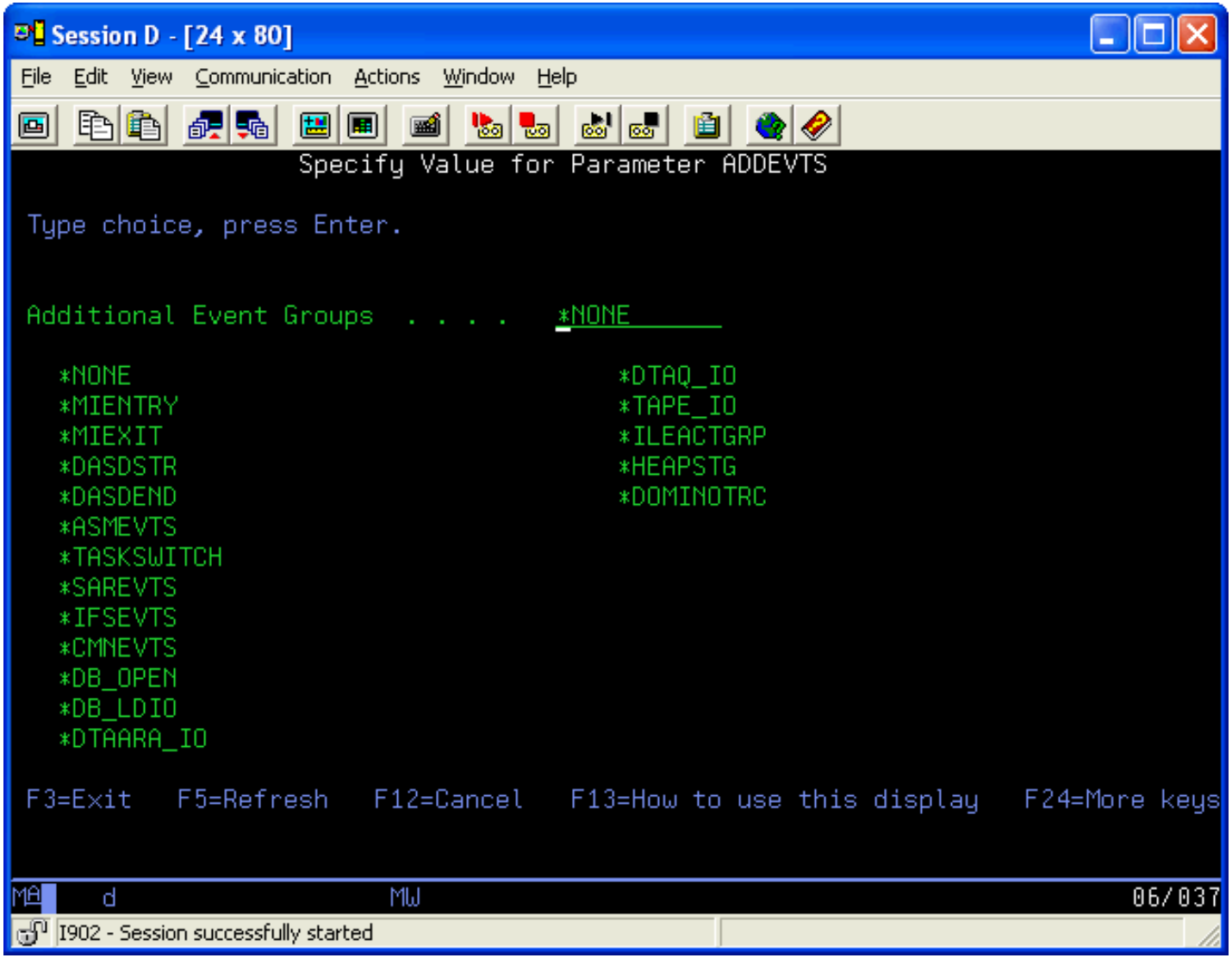
NBRTHD - Number of threads for ENDPEX
With the default value of '*CALC', ENDPEX will determine the an appropriate number of threads to use when the data is being dumped into the collection files. You can supply a value of between 1 and 256 depending on the impact that you want on the system when the data is being dumped.

ENDPEXPTY - ENDPEX job priority
This value controls the run priority of the job/threads created for use by the ENDPEX processing of the collected data. The default value is 51, change as you feel is appropriate.

PEXFTR - PEX Filter
Supply the name of a previously created PEX filter to enable more precise collecting of data.

TEXT - a description for this collection
Use this keyword to provide a more meaningful description of your collection.

ADDEVTS - Add Additional Events
Allow for the addition of extra events to beyond those normally collected for a specific problem type. Any event group or combination of event groups can be added to any problem type that results in a *TRACE type definition. This means that you cannot add these events to problem type that creates a *STATS or *PROFILE definition.



Prematurely Ending an STRPACOL Collection

When the **STRPACOL** command is used to gather a PEX collection, the collection is normally active for the DURATION number of minutes. However, frequently it is necessary to end the collection early. For that purpose, the **ENDPACOL** command is provided (also in library QIDRPA).

This command has two parameters, DATALIB, which must be the same library as the library that you supplied on the **STRPACOL** command, and DATAOPT which controls what actions are to be taken on this collection.

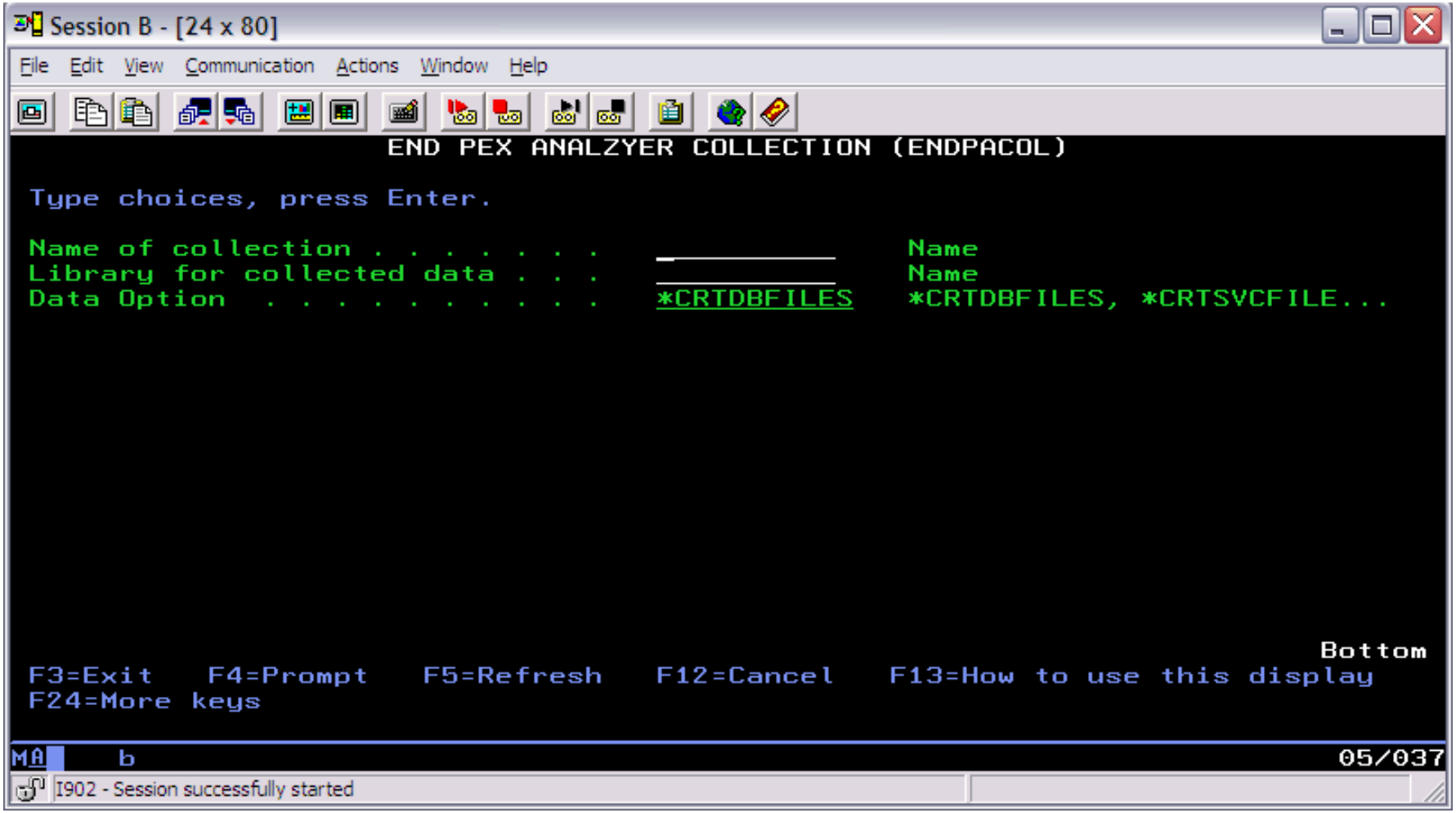
DATAOPT keyword - data option

- ***CRTDBFILES** - Stop the collection immediately and save the PEX data into DB2/400 in a format ready for analysis.

- ***CRTSVCFILE** - Stop the collection immediately and save the PEX data into a management collection object for sending to IBM for analysis
- ***DELETE** - Stop PEX collection immediately without saving the data because the problem you are investigating did not occur.
- ***RESTART** - Stop PEX collection immediately, discard the data and immediately start another collection.

The **ENDPACOL** command works only when the collection job (the submitted batch job named QIDRPACOL or the interactive job running **STRPACOL**) is still collecting data. If ENDPEX has already started the command will not work.

The command is shown below:



Transporting PEX Collection Data

Assuming the above recommendations are followed on the **STRPACOL** command, the collection data will be stored into approximately 52 different DB2/400 database files starting with the prefix QAYPE*. Some of these files hold much more data than others. However, when saving or restoring this data

from machine to machine, it is required that **all** the QAYPE* files be saved or restored. The easiest way to do this is to use the FTPPACOL command to transfer a collection to another system. Another option would is to use the SAVLIB command or the SAVPACOL command to save the collection to a save file and transfer the file to the other system manually. If SAVPACOL is used then RSTPACOL must be used on the target system to properly restore the collection from the save file.

Version/Release Considerations

Unfortunately, PEX data is version and release sensitive. You can only perform an analysis using the PEX Analyzer GUI on collections matching the VRM of the operating system VRM. However, you may view already created analyses for prior level collections using PEX Analyzer (look at already created V4R5 analyses on a V5R1 system for example).

Table of Contents Previous Next

Chapter 4 PTDV

This chapter describes how to collect Performance Explorer (PEX) data for analysis with iDoctor for iSeries PTDV. This chapter includes:

- The method and command syntax needed to collect PEX TRACE data for PTDV

Table of Contents | Previous | Next

4.1 Overview of Collecting PEX Data

The basic steps for creating a PEX Collection are the following:

1. Creating or choosing an existing **PEX definition** (explained in Section 4.2, Creating PEX Definitions).
2. Starting a collection using the definition.
3. Ending the collection and directing the data to be discarded or stored in a library.

Note: The PEX Analyzer client provides interfaces for creating and/or changing PEX definitions. The client also has an interface for creating a collection, viewing it as runs, and then analyzing its results. See the client side documentation for PEX Analyzer for more information.

Table of Contents | Previous | Next

4.2 Creating PEX definitions

A variety of PEX definitions can be used with PTDV. The most commonly used definitions (those listed in Standard Java Definitions or Standard ILE Definitions) can be created automatically once you've installed the tools library PTDV according to section "Installing the Tools Library" from the web page where this tool was installed. To add the standard definitions mentioned in this section, invoke the commmand:

PTDV/ADDPTDVDFN

Alternatively, they can be created manually by using the command given in the description of each definition. You can also create your own Custom Definitions so that your trace contains only the data you need.

Note: You can use copy-paste to easily enter any of the command examples from this section into an iSeries command entry screen.

Table of Contents | Previous | Next

4.2.1 Standard Java definitions

This section contains definitions commonly used with PTDV.

Table of Contents Previous Next

4.2.1.1 Java Method Trace events only

For Java applications where you only want to trace the Java methods, the following collection can be used for all supported releases:

ADDPEXDFN DFN(PTDVJVAMTH) TYPE(*TRACE) JOB(*ALL) TASK(*ALL) MAXSTG(1000000) TRCTYPE(*SLTEVT) SLTEVT(*YES) BASEVT(*PMCO) PGMEVT(*JVAENTRY *JVAEXIT) TEXT('Java method events')

This collection includes:

- Java method entry and exit events

Table of Contents | Previous | Next

4.2.1.2 Java Method Trace with basic set of Java events

For Java applications, the most commonly used PEX definition include Java method entry and exit events, along with Java object create/delete events, Java object lock/unlock events, and WebSphere events. In V5R2, more events are available. The exact definition command will depend on the release, and the type of events you want to include.

For releases prior to V5R2:

ADDPEXDFN DFN(PTDVJAVA) TYPE(*TRACE) JOB(*ALL) TASK(*ALL)
MAXSTG(1000000)
TRCTYPE(*SLTEVT) SLTEVT(*YES) BASEVT(*PMCO) PGMEVT(*JVAENTRY *JVAEXIT)
JVAEVT(*OBJCRT *THDSTTCHG) OSEVT(*MIEV13) TEXT('PTDV basic Java events')

This collection includes:

- Java method entry and exit events
- Object create and delete events
- Object lock and unlock events
- WebSphere events

For V5R2 and later releases:

ADDPEXDFN DFN(PTDVJAVA) TYPE(*TRACE) JOB(*ALL) TASK(*ALL)
MAXSTG(1000000)
TRCTYPE(*SLTEVT) SLTEVT(*YES) BASEVT(*PMCO) PGMEVT(*JVAENTRY *JVAEXIT)
JVAEVT(*OBJCRT *LCKSTR *UNLCK *THDCRT *THDDLT) APPEVT(*WAS)
TEXT('PTDV basic Java events')

This collection includes:

- Java method entry and exit events
- Object create and delete events

- Object lock and unlock events
- Thread create and delete events
- WebSphere events

In V5R2, additional events can also be added:

ADDPEXDFN DFN(PTDVALLJVA) TYPE(*TRACE) JOB(*ALL) TASK(*ALL) MAXSTG(1000000)
TRCTYPE(*SLTEVT) SLTEVT(*YES) BASEVT(*PMCO) PGMEVT(*JVAENTRY *JVAEXIT)
JVAEVT(*OBJCRT *LCKSTR *UNLCK *THDCRT *THDDLT *CLSLOAD *GBGCOLSWEEP)
APPEVT(*WAS) TEXT('PTDV all supported Java events')

This collection includes:

- Java method entry and exit events
- Object create and delete events
- Object lock and unlock events
- Thread create and delete events
- Class load events
- Garbage collector sweep events
- WebSphere events

Table of Contents | Previous | Next

4.2.1.3 Java Method Trace with no Object Creates

In some cases, a trace like PTDVJAVA is needed, but object create events are not needed. Since the number of object create events can be very large, it is sometimes useful to use a definition with the basic set of PTDV events excluding object create events.

For releases prior to V5R2:

ADDPEXDFN DFN(PTDVNOCRT) TYPE(*TRACE) JOB(*ALL) TASK(*ALL)
MAXSTG(1000000)
TRCTYPE(*SLTEVT) SLTEVT(*YES) BASEVT(*PMCO) PGMEVT(*JVAENTRY *JVAEXIT)
JVAEVT(*THDSTTCHG) OSEVT(*MIEV13)
TEXT('PTDV Java method entry/exit, Object Lock, and WAS events')

This definition includes:

- Java method entry and exit events
- Object lock and unlock events
- WebSphere events

For V5R2 and following releases:

ADDPEXDFN DFN(PTDVNOCRT) TYPE(*TRACE) JOB(*ALL) TASK(*ALL)
MAXSTG(1000000)
TRCTYPE(*SLTEVT) SLTEVT(*YES) BASEVT(*PMCO) PGMEVT(*JVAENTRY *JVAEXIT)
JVAEVT(*LCKSTR *UNLCK) APPEVT(*WAS)
TEXT('PTDV Java method entry/exit, Object Lock, WAS events')

Table of Contents | Previous | Next

4.2.1.4 Java Trace with only Object Creates

Note that **no** Java method entry and exit events appear in this definition, so no performance information for the methods in the application will appear when PTDV processes a collection created with this definition.

For all releases:

ADDPEXDFN DFN(PTDVCRT) TYPE(*TRACE) JOB(*ALL) TASK(*ALL) MAXSTG(1000000) TRCTYPE(*SLTEVT) SLTEVT(*YES) BASEVT(*PMCO) JVAEVT(*OBJCRT) TEXT('PTDV Java Object Create events')

This collection includes:

- Object create and delete events

By default, this event contains 5 levels of call stack information, which displays what the call stack was at the point of the object create. In V5R3 or later, the user has the option of saving 16 levels of call stack information for each object create. This is done by setting the following environment variable at the time of using the ADDPEXDFN command:

SETENVVAR ENVVAR(QYPE_JAVA_FORMAT) VALUE('2')

Using the ADDPEXDFN command from above while this environment variable has the value 2, will indicate that 16 levels of call stack information should be saved.

Table of Contents | Previous | Next

4.2.1.5 Java Trace with Object Locks

Note that **no** Java method entry and exit events appear in this definition, so no performance information for the methods in the application will appear when PTDV processes a collection created with this definition.

For releases prior to V5R2:

ADDPEXDFN DFN(PTDVLOCK) TYPE(*TRACE) JOB(*ALL) TASK(*ALL)
MAXSTG(1000000)
TRCTYPE(*SLTEVT) SLTEVT(*YES) BASEVT(*PMCO) JVAEVT(*THDSTTCHG)
TEXT('PTDV Java Object Lock events')

This definition includes:

- Object lock and unlock events

For V5R2 and following releases:

ADDPEXDFN DFN(PTDVLOCK) TYPE(*TRACE) JOB(*ALL) TASK(*ALL)
MAXSTG(1000000)
TRCTYPE(*SLTEVT) SLTEVT(*YES) BASEVT(*PMCO) JVAEVT(*LCKSTR *UNLCK)
TEXT('PTDV Java Object Lock events')

This definition includes:

- Object lock and unlock events

In V5R2, thread notify events can also be added:

ADDPEXDFN DFN(PTDVLCKNFY) TYPE(*TRACE) JOB(*ALL) TASK(*ALL)
MAXSTG(1000000)
TRCTYPE(*SLTEVT) SLTEVT(*YES) BASEVT(*PMCO)
JVAEVT(*LCKSTR *UNLCK *THDNFY *THDNFYALL *THDWAIT)

TEXT('PTDV Java Object Lock and Notify events')

This definition includes:

- Object lock and unlock events
- Thread notify and wait events

By default, this event contains 5 levels of call stack information, which displays what the call stack was at the point of the object create. In V5R3 or later, the user has the option of saving 16 levels of call stack information for each object lock and unlock. This does not apply to the thread notify and wait events. This is done by setting the following environment variable at the time of using the ADDPEXDFN command:

SETENVVAR ENVVAR(QYPE_JAVA_FORMAT) VALUE('2')

Using the ADDPEXDFN command from above while this environment variable has the value 2, will indicate that 16 levels of call stack information should be saved.

Table of Contents | Previous | Next

4.2.1.6 Java Trace with Exception events

In V5R3, java exception events are available:

ADDPEXDFN DFN(PTDVEXCP) TYPE(*TRACE) JOB(*ALL) TASK(*ALL)
MAXSTG(1000000)
TRCTYPE(*SLTEVT) SLTEVT(*YES) JVAEVT(*JVAEXCP)
TEXT('PTDV Java Exception events')

This definition includes:

- Java exception events

A definition containing java exception events and java method entry/exit events:

ADDPEXDFN DFN(PTDVJVAEX) TYPE(*TRACE) JOB(*ALL) TASK(*ALL)
MAXSTG(1000000)
TRCTYPE(*SLTEVT) SLTEVT(*YES) BASEVT(*PMCO) PGMEVT(*JVAENTRY *JVAEXIT)
JVAEVT(*JVAEXCP) TEXT('PTDV Java Exception events')

This definition includes:

- Java exception events
- Java method entry and exit events

Table of Contents | Previous | Next

4.2.1.7 Java Trace with Class load and Transform events

In V5R3, events describing class loads and java transform (JIT compile) events are available:

ADDPEXDFN DFN(PTDVCLSTFM) TYPE(*TRACE) JOB(*ALL) TASK(*ALL) MAXSTG(1000000) TRCTYPE(*SLTEVT) SLTEVT(*YES) JVAEVT(*CLSLOAD *TFMSTR) TEXT('PTDV Java class load and transform events')

Table of Contents | Previous | Next

4.2.2 Standard ILE/OPM Definitions

For ILE or OPM applications, PTDV can provide procedure summary and call trace information. Refer to Generating ILE data for more information on generating procedure and program hooks in non-Java code.

The following definition will generate events for ILE and OPM code which have been compiled to contain standard performance hooks:

ADDPEXDFN DFN(PTDVMI) TYPE(*TRACE) JOB(*ALL) TASK(*ALL) MAXSTG(1000000) TRCTYPE(*SLTEVT) SLTEVT(*YES) BASEVT(*PMCO) PGMEVT(*MIENTRY *MIEXIT) TEXT('PTDV MI Entry/Exit events')

ILE code can also contain "Trace Job" hooks, and a PEX collection can contain program events of this type if the code has been compiled appropriately (see Generating ILE data). Events of this type will be generated with this definition:

ADDPEXDFN DFN(PTDVMI) TYPE(*TRACE) JOB(*ALL) TASK(*ALL) MAXSTG(1000000) TRCTYPE(*SLTEVT) SLTEVT(*YES) BASEVT(*PMCO) PGMEVT(*PRCENTRY *PRCEXIT) TEXT('PTDV MI Trace Job Entry/Exit events')

To include MI complex information in either of these definitions, add *MISTR and *MIEND to the PGMEVT parameter.

Table of Contents | Previous | Next

4.2.3 Custom definitions

You can also create your own custom PEX definitions for use with PTDV. You will use a command such as:

ADDPEXDFN DFN(*MyPTDVDef*) TYPE(*TRACE) JOB(*ALL) TASK(*ALL) MAXSTG(1000000) TRCTYPE(*SLTEVT) SLTEVT(*YES) BASEVT(*PMCO) PGMEVT(*program entry/exit events*) JVAEVT(*Java events*) OSEVT(*OS events*) APPEVT(*APP events*) TEXT('My PTDV PEX Definition')

You can create a definition containing the specific event types that your analysis requires. PTDV can provide procedure/method summary and call trace information if the appropriate program events are in the definition, and those procedures or methods contain performance hooks. The following table describes the set of program events supported by PTDV. For information on how to generate performance hooks in your application code, refer to Generating Java Data or Generating ILE data.

All entries in this table will result in entries in the Procedure Summary Tables, Procedure Call Tables, and the Call Trace Tree.

Event Type	Event Subtype	Performance hooks needed	Generates entry and exit events for
Program Event	*MIENTRY and *MIEXIT	yes	ILE procedures and ILE or OPM programs
Program Event	*MISTR and *MIEND	no	MI complex instructions
Program Event	*JVAENTRY and *JVAEXIT	yes	Java methods (JIT and DE)
Program Event	*PRCENTRY and *PRCEXIT	yes	ILE programs and procedures
OS Event	*DBSVRREQ	no	DB server requests

Most Java events are supported by PTDV. A table containing the supported Java events are listed below. When these events are included in a definition containing Java method entry and exit events, event data is analyzed as well as method context information. For example, in a trace containing Java

method entry and exit events and object create events, the methods which create objects is known as well as the class name and size of the object. In a trace containing object lock events along with Java method entry and exit events, the locking/unlocking methods can be determined as well as the class name for the object being locked.

Be aware, however, that Java method entry and exit events, as well as some of the events listed below, have the potential for producing an extremely large collection in a short period of time. Refer to Limiting the size of your collection for more information on keeping the collection to a reasonable size.

In some cases, it may be preferable to create a definition that doesn't contain Java method entry and exit events.

For more details on how each individual event is displayed within PTDV, see the following section.

Event Type	Event Subtype	Release	Description
Java Event	*OBJCRT	all	Java Object Create and Delete events
Java Event	*THDSTTCHG	V4R5, V5R1	Java Object Lock and Unlock events
Java Event	*LCKSTR and *UNLCK	V5R2+	Java Object Lock and Unlock events
Java Event	*THDCRT and *THDDLT	V5R2+	Java Thread Create and Delete events
Java Event	*CLSLOAD	V5R2+	Java Class Load event
Java Event	*THDNFY, *THDNFYALL, and *THDWAIT	V5R2+	Java Thread Notify and Wait events
Java Event	*GBGCOLSWEEP	V5R1+	Java Garbage Collection Sweep events
Java Event	*JVAEXCP	V5R3+	Java Exception event
Java Event	*TFMSTR	V5R3+	Java Transform event
OS Event	*MIEV7	V4R5	Transaction events (for WebSphere Application Server)

OS Event	*USRTNS	V5R1+	Transaction events (for WebSphere Application Server)
OS Event	*MIEV13	V4R5, V5R1	WebSphere events
APP Event	*WAS	V5R2	WebSphere events

Table of Contents | Previous | Next

4.2.4 Performance data provided by Java events

The following list includes the supported Java events, and the data that is viewed about that event through PTDV

- ***OBJCRT**

  The object create event generates an event at each object create during the run of your application, and at each object delete during garbage collection. Information about the object's class name, allocated size, and array information is provided in PTDV. This data is summarized in the Object Summary Table on the Main Frame, Job Frame, Thread Frame, providing totals for the corresponding collection level. This information is also summarized for each Procedure that the object creates occur in, and can be viewed from a Procedure Summary Table and a Procedure Call Table. In addition, for each row in the Call Trace Pane, a count appears showing the number of objects that have been created by the selected procedure call.

  If Full Trace Processing is not selected, then there is a "Quick View"available for this collection. The "Quick View"provides a summary of the object create information in the collection. For collections created on V5R2 or later, the call stack from the point of the object create event is available to be displayed here. The user has the option of collecting 5 levels of call stack information or 16 levels, based on the collection definition used. The "Quick View"provides class name, create and delete counts and sizes, along with the call stack information.

- ***THDSTTCHG (prior to V5R2) or *LCKSTR and *UNLCK (V5R2 or after)**

  The lock events generate a record whenever a thread tries to acquire an object that is held by another thread, and generates another record when releasing the object. As in the object create event, this information is summarized on each of the Object Summary Tables, and if Java method events are present, statistics will be collected for object events that occur within each method, and that data is summarized on the Call Trace Pane, the Procedure Call Table, and the Procedure Summary Table. Note that for this latest version of PTDV, there is a count for both locks and lock events. The lock count is the number of combined lock events that have occurred on a single object at one point in time, i.e., if a thread acquires an object and then later releases it, that is a single lock. The lock event count is the total number of individual lock events that have

occurred for a given object, and therefore would include the event signalling the lock was acquired, and the event indicating the lock was released.

In V5R2 or later, the user has the option of selecting a "Quick View"for these events, instead of doing Full Trace Processing. There can be either 5 levels of call stack information or 16 levels, which the user can optionally select. The number of locks and unlocks is provided, grouped by class name, with a summarized view of the call stack when the locks occurred.

- ***THDNFY, *THDFYALL, *THDWAIT (V5R2 or after)**
  These events correspond to the use of the notify(), notifyAll(), and wait() methods in java.lang.Object. These events are summarized as in the lock case. On the Object Info Frame, there is a Thread Notify Pane containing information about the set of these events that have occurred on a single object. An entry for these events also appears in the Call Trace Pane at the point where the event occurs.

- ***CLSLOAD (V5R2 or after)**
  A class load event is generated whenever the loader loads a class in the application. Each event contains information about the name of the class loader and the DE optimization level if it has been compiled for Direct Execution. This information will be displayed on the Class Summary Table on the Main Frame.

- ***GBGCOLSWEEP (V5R1 or after)**
  This event provides collection information about each GC sweep cycle that occurred during the collection, such as live objects, size collected, size of the heap, etc. It is summarized on the Garbage Collection Sweep Summary Table on the Job Frame.

- ***THDCRT and *THDDLT (V5R2 or after)**
  These events provide timestamp information for the thread creates and deletes that have occurred during the collection. It will also provide the task count identifier for the creator thread. This information is provided on the Thread Pane of the Job Frame.
- ***JVAEXCP (V5R3 or later)**
  These events provide information about java exception throws and catches. The type of exception object, the method where the throw or catch occurred, and the timestamp of the exception are provided. When the view for exception events is selected, then then a Java Exception Pane is generated as on the Main Trace Frame. When Full Trace Processing is selected, then these events will be summarized on the Event Summary Paneof theMain Trace Frame.
- ***TFMSTR (V5R3 or later)**
  These events provide information JIT compile time. The name of the method, and the time it took for the JIT to compile the method are displayed for these events. When these events are in the collection, then a Java Transform/JIT Compile Summary Pane appears on the Main Trace Frame.

4.2.4 Performance data provided by Java events

Table of Contents | Previous | Next

4.2.5 OS/400 Release Differences

The current version of PTDV includes support for PEX collections generated on V5R2, but no longer supports collections created on V4R4.

- **Object Lock events**
  In V4R5 and V5R1, the *THDSTTCHG event is used to generate java lock events. In V5R2, the *THDSTTCHG event is no longer used. The *LCKSTR (for lock) and *UNLCK (for unlock) events must be used to obtain the same information that is available in previous releases with the *THDSTTCHG event.

- **Object Create events**
  In V4R5 and V5R1, the primitive array names are not available on object delete events. As a result, the delete counts for primitive arrays are not correct when processing collections generated on these releases. In V5R2, these names are available and delete counts are accurate.

- **Thread Notify, Thread Notify All, and Thread Wait**
  For collections created for V5R2 or later, PTDV provides support for the Thread Notify, Thread Notify All, and Thread Wait events. These events appear in the Call Trace Frame for the thread where the events occur; and a table summarizing these events is generated on the Object Group Frame and Object Info Frame for those classes that generate this type of event. Each Job Frame and Thread Frame will have an Event Summary Table containing these events if they occurred in the corresponding job or thread.

- **Thread Create and Thread Delete**
  These events are processed in collectios created for V5R2 or later. When these events are present, timestamp information for thread creates and deletes are added to the Thread table, and the task count identifier for the creating thread is available if Thread Create events are included in the collection.

- **Class Load**
  PTDV supports Class Load events for collections created on V5R2 or later. When these events are present, the class name, class loader name, and DE optimization level information is available and is shown on the Object Summary Table for the initial frame.

- **Primitive array names**

  On releases prior to V5R2, if the object associated with any type of Java event was a primitive array (i.e., int[]), the name was not available to PTDV so appeared in the object tables as "unknown". In V5R2, these names are available and show up correctly in the tables.

- **Instruction counters**

  On releases prior to V5R2, when method/procedure entry and exit events are present in the collection, the instruction counts for each method/procedure are available in the PEX data and are displayed on all panes containing method information. This information is no longer available in V5R2, and if displayed on a pane containing method or procedure information, any columns referring to instruction counts will be 0.
- **Call stack information**

  In V5R3, the object create, object lock, and unlock events can be generated to contain 16 levels of call stack information. In prior releases these events contained 5 levels of call stack.
- **Java exception events**

  In V5R3 and later releases, java exception events are available. These events are generated whenever a java exception throw or catch occurs during the Java application. The data from these events is summarized and displayed on a pane in the Main Trace Frame.
- **Java transform (JIT compile) events**

  In V5R3 and later releases, java transform events are available. These events are generated whenever the JIT starts and ends a compilation for a method.

Table of Contents | Previous | Next

4.3 Generating a PEX collection

The next sections cover useful information on generating PEX collections for use by PTDV.

Table of Contents | Previous | Next

4.3.1 Generating PEX Definitions

PTDV processes and displays performance information found in PEX collection data. To collect this type of data for your application, you will need to create a PEX definition containing events that are supported and can be viewed by PTDV. If you have installed the PTDV library as described in the Installation documentation, you can create a standard set of PEX definitions for use with PTDV by calling the program:

CALL PTDV/ADDPTDVDFN

For more information about PEX definitions and PEX events (including information on creating custom definitions) for use with PTDV, please see the section on Creating PEX definitions. The above program will create the definitions described in that section.

Table of Contents | Previous | Next

4.3.2 Analyzing Java applications

PTDV can see Java method information for methods running with the interpreter, the Just In Time (JIT) compiler, or Direct Execution (DE). See the iSeries Java documentation for information on selecting which mode to use when running a Java program.

In order to see Java method information in the Performance Trace Data Visualizer, entry/exit hooks must be enabled in the Java classes. There are two basic methods of doing this:

Run your program with the JIT (or interpreter), and enabled method entry/exit hooks. One way to force your program to run with the JIT is to set the property java.compiler=jitc. To enable performance hooks, you can set os400.enbpfrcol=1. For more information on how to set these properties, see the FAQ.

-or-

Enable method entry/exit hooks in transformed (aka DE'd) Java code. In order to do this, you will need to use CRTJVAPGM with the ENBPFRCOL(*ENTRYEXIT) option to create Java program objects with hooks. You will need to do this for each jar, zip, or class file you want hooks for. The Performance Data Visualizer can view method information for DEd code at any optimization level, but level 40 is preferred, since this is what your application should normally be running at. Using CRTJVAPGM on large files or on slow machines may result in a long wait. If you are running on a machine with a greater batch than interactive capacity, be sure to submit the CRTJVMPGM to batch.

If you are doing performance data collection using Direct Execution java code, then the classes in the JDK will not have hooks enabled and will therefore not show up in your trace.

Note that only classes loaded via the system or bootstrap class loaders can run under Direct Execution mode. Classes loaded with a user class loader (such as servlets running with WebSphere) will always run with the JIT compiler (unless you explicitly place these classes in the system classpath).

If you are interested in seeing what object types are being created and/or which object types are being locked, you can use a PEX definition containing the *OBJCRT and *THDSTTCHG Java events. Object information can be collected with or without enabling entry/exit hooks. For more information on setting up PEX definitions, see the Creating PEX definitions section.

Warning: Do not attempt to run the CRTJVAPGM command against the java.zip, sun.zip, classes.zip, or rt.jar files supplied by IBM on your iSeries. If you do, they will become corrupted and you will need to re-install the JV1 product to recover. You **can** run CRTJVAPGM against the classes used by the native JDBC driver (part of JV1), the Java Toolbox classes installed with JC1, and the open-source version of the Java Toolbox (JTOpen). Each release of IBM's WebSphere Application Server contains several jar files that need special options if they are to be re-created with hooks using CRTJVAPGM. Therefore, you should not use CRTJVAPGM on jar files shipped with WebSphere. In all of these cases, you can force use of the JIT compiler and enable hooks using the first method listed above to see performance information on these classes.

Table of Contents Previous Next

4.3.3 Mixed mode interpreter considerations

Starting in V5R1, Mixed Mode Interpretation is used for compiling Java methods. This is new a feature that is available when using the JIT, which Java methods are first interpreted, then JITed after their execution count has reached a specified threshold. On the iSeries, this value is 2000. Use of mixed mode interpretation improves performance by reducing start up times for most applications and workloads.

This can impact the information associated with a PEX collection. Many of the Java events contain call stack information. This allows the allow the user to view some context information about where an event occurred without having to include entry/exit hooks or events in their collection. This can save time and overhead. However, if the method is an interpreted method, then the call stack information will not be of any value, since the SLIC methods on the stack will reflect the interpreter or JIT compiler rather than the Java code being executed.

If you are using the call stack information found in the Java events, you can avoid or minimize the impact of MMI on performance data collection by setting the property **os400.jit.mmi.threshold=0** . This will cause all Java methods to be JITed the first time they are executed. There is a performance penalty in some cases for setting this property, so it should only be done when doing performance analysis.

Table of Contents | Previous | Next

4.3.4 Analyzing ILE/OPM applications

If you want to trace code written in ILE or OPM languages, including programs called from Java using JNI (Java Native Interface), you must use a PEX defintion which includes the appropriate program events, and your application code must have been compiled to contain performance hooks. See Standard ILE/OPM Definitions for information on creating the correct definition.

In OPM code, standard performance hooks are included by default in every program entry point.

In ILE code, standard performance hooks can be included in your ILE modules and programs in one of two ways:

- On initial create of a module, set the parameter ENBPFRCOL to *ENTRYEXIT.
- For a program that already exists, use the CHGPGM or CHGSRVPGM command, specifying *ENTRYEXIT for the ENBPFRCOL parameter.

In ILE code, there is another type of hook, the trace job hook, which is used by the iSeries TRCJOB command and can also be generated in a PEX collection. These hooks can be generated by:

- Compiling your ILE code at optimization level 30 or below.
- Using the LICOPT CallTracingAtHightOpt when compiling your code at optimization level 40.

MI complex instructions can also be included in your PEX collection, and no hooking is necessary by the user to enable these.

Table of Contents | Previous | Next

4.3.5 Starting and Ending the Trace

Data collection begins once the STRPEX command is invoked. The type of data collected depends on the PEX definition specified on this command; the definition must already exist.

An example command:

STRPEX SSNID(***COLL_NAME***) OPTION(*NEW) DFN(PTDVJAVA)

In V5R2, setting the DFN parameter to the value '*SELECT' will bring up a list of the available PEX definitions for you to choose from.

Data collection stops once the ENDPEX command is invoked. For example:

ENDPEX SSNID(***COLL_NAME***)

Starting in release V5R1, specifying '*SELECT' on the SSNID parameter will allow you to view the set of collections that are currently active, along with the current event counts. Hitting F5 will refresh the data, allowing you to watch how your collection is growing. You can then end your active collection at an appropriate point.

Table of Contents | Previous | Next

4.3.6 Limiting the Size of the Trace

It is very easy to generate a huge number of PEX trace events in a very short period of time, especially when collecting data for a multithreaded program. In Thin Client mode, PTDV can process collections containing 5,000,000 events or more; however processing time can take quite a while in Thin Client mode. For Thick Client mode, a collection of less than 300,000 events is recommended.

There are several methods of limiting the amount of data in a trace. These include:

- Use a PEX definition that includes only the events you need. For example, traces which include method entry/exit events can be very large, so don't include these events in your PEX definition if you are only interested in object creates/locks.
- Likewise, if you are analyzing a Java locking problem, you may need method entry/exit events as well as Java lock events, but you can turn off the object create events. For more information on creating a PEX definition with only the necessary events, see the section on Creating PEX definitions.
- If multiple jobs are running on the system, change the PEX definition to collect events only for the job you are interested in. To do this, you can use the CHGPEXDFN command, and specify the job name in the JOB parameter.
- If you do need method entry/exit events in your trace, try to add entry/exit hooks only to the classes that you need to see the events for. For example, if you are using the CRTJVAPGM ENBPFRCOL(*ENTRYEXIT) method of generating hooks, only generate them for the jar files or classes that you are interested in. If the part of your code you are interested in normally runs in the JIT, then you can enable hooks for the JIT, but don't force all of the other code to run in the JIT. (In other words, specify the os400.enbpfrcol=1 property, but not java.compiler=jitc.) This can be particularly helpful when analyzing code which runs under WebSphere, because you can enable hooks for your application code (which normally runs with the JIT) but not for the WebSphere code (which normally runs DE).
- Run only a small number of transactions or invocations during the trace. For example, if you are analyzing a Java servlet, try just loading the servlet a couple times manually from your web browser, rather than tracing under a full load.
- Note: This method will not be effective for analyzing Java lock problems, since the locking behavior of your application may change significantly under heavier loads. But it can work very well for pathlength or object creation analysis.
- If none of the other methods will work, sometimes the best thing to do is just to start and end PEX as quickly as possible. One method of doing this is to start your program and let it ramp

up to steady state. Then type the STRPEX and ENDPEX statements into two separate windows so that you can do the STRPEX, quickly switch windows, and just press enter to do the ENDPEX.

- Starting in V5R3, you can use PEX filters to include only the data that you want. Refer to that section for details on what filters are available and how to do it.

Table of Contents | Previous | Next

4.3.7 Using PEX filters to limit collection size

Starting in V5R3, PEX filters are available to help in reducing the size of a trace collection. PEX filters provide the ability to filter out performance data based on various conditions, thereby focusing in on the data that is most useful for performance analysis. When collecting PEX data, it is common to generate a very large amount of data in a short period of time; the use of PEX filters allows you to reduce the size of the collected data. For Java applications, there are three filters available:

Java method filters

These types of filters are used to filter out events that contain method information. These apply to java entry and exit events, and java transform events. The following information can be identified on the ADDPEXFTR command when creating a filter:

- relational operator (*EQ or *NE)
- package name (this can be set to *NONE)
- class name (this can be set to *ALL)
- method name (this can be set to *ALL)

For example, if you wanted to generate a method trace **excluding** the methods in java/lang and java/util, you could create a filter like this:

ADDPEXFTR FTR(NOLANGUTIL) JVAFTR(*NE (('java/lang') ('java/util')))

Or, if you only wanted to include methods in java/lang/String, you could use this filter:

ADDPEXFTR FTR(STRING) JVAFTR(*EQ (('java/lang' String)))

Any kind of filter can be generated, depending on the methods you want to focus on.

Java method triggers

This type of filter is used to indicate that all event collection should begin when entering a method, and end when exiting the method. The information included in the filter:

- relational operator (*EQ or *NE)
- package name (this can be set to *NONE)
- class name (this can be set to *ALL)
- method name (this can be set to *ALL)

For example, if you wanted to only trace the events that occurred between entry and exit of the main method in Hello.

ADDPEXFTR FTR(HELLOMAIN) JVATRG(*NONE Hello main)

This filter will cause PEX to start collecting events once the main method in the Hello class has been entered, and to stop collecting events once the main method has exited.

Or, if you want to collect data at after certain point in the application, once a specific method has been entered, you can use this filter:

ADDPEXFTR FTR(DOGETTRG) JVATRG('javax/servlet/http' HttpServlet doGet *ENTRY)

Java class filters

Java class filters are used with events that identify java objects, such as object create, object lock, object unlock, java exception, thread notify, thread notify all, and thread wait. The class filter identifies the class name to filter on. It can be used to include only a small set of objects, or to exclude a set of objects. This is very useful after another tool has been used to identify the class which has had many objects created, or where most of the lock contention is occurring.

The fields on the ADDPEXFTR command for this type of filter include:

- relational operator (*EQ or *NE)
- package name (this can be set to *NONE)
- class name (this can be set to *ALL)
- method name (this can be set to *ALL)
- array indicator (used to indicate array/no array and primitive arrays, such as integer or char arrays)

For example, if you want to exclude all java/lang/ClassNotFoundException objects from your collection:

ADDPEXFTR FTR(CLASSNOTFD) JVACLSFTR(*NE (('java/lang' ClassNotFoundException)))

Or, if you want to exclude objects from java/lang and java/util packages:

ADDPEXFTR FTR(NOJLCLASS) JVACLSFTR(*NE (('java/lang') ('java/util'))

Table of Contents | Previous | Next

Part III Client-side components

This part covers the client-side (graphical screen interfaces) for each iDoctor for iSeries component.

Table of Contents Previous Next

Chapter 1 Common Graphical Interfaces

This chapter provides documentation for the iDoctor for iSeries interfaces that are found in all components (except PTDV).

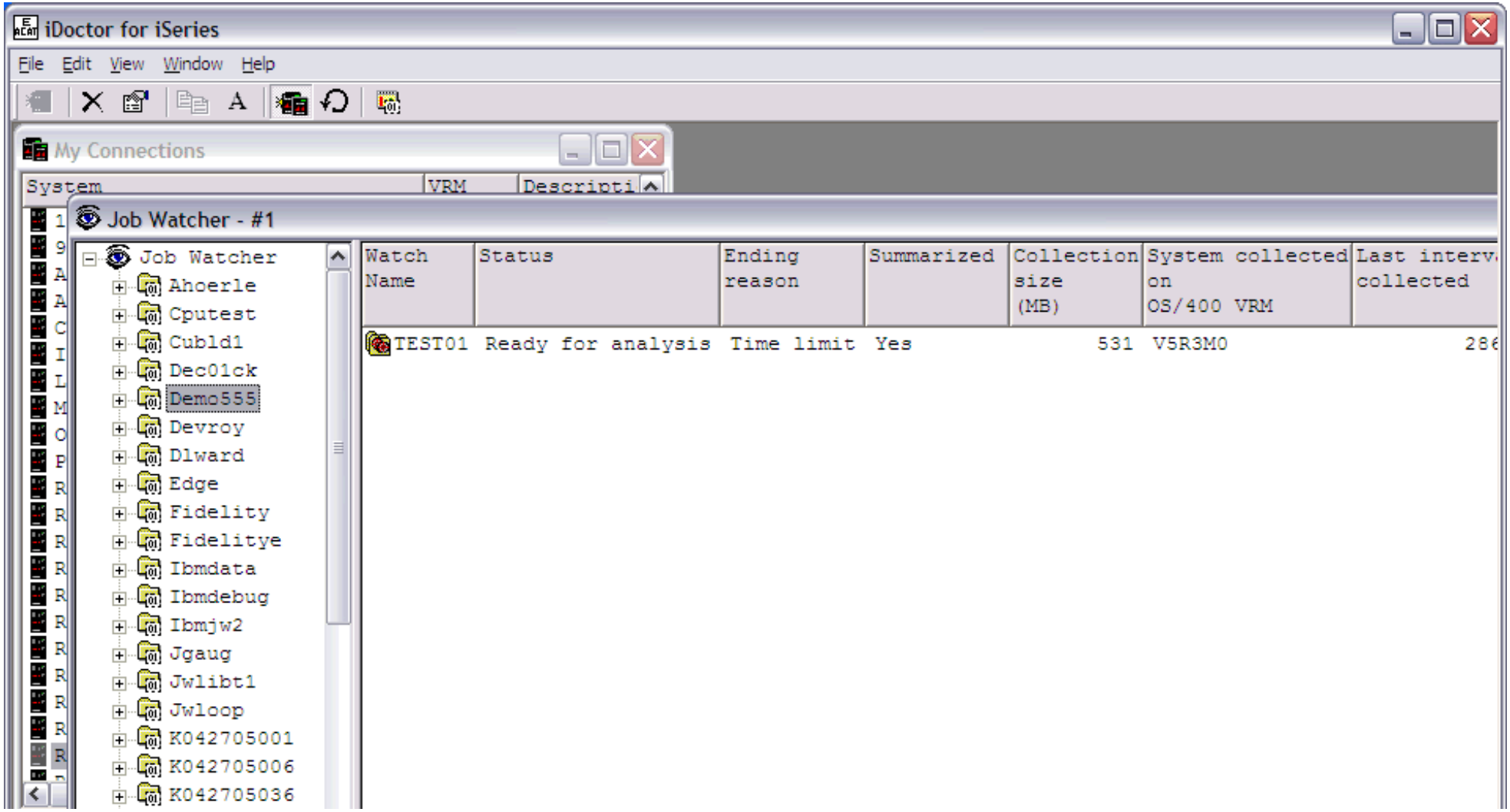
Table of Contents | Previous | Next

1.0 Overview

iDoctor for iSeries may be launched via the desktop icon or using the Start Menu. The shortcut folder on the Start Menu is Start -> Programs -> iDoctor for iSeries.

The Main Window

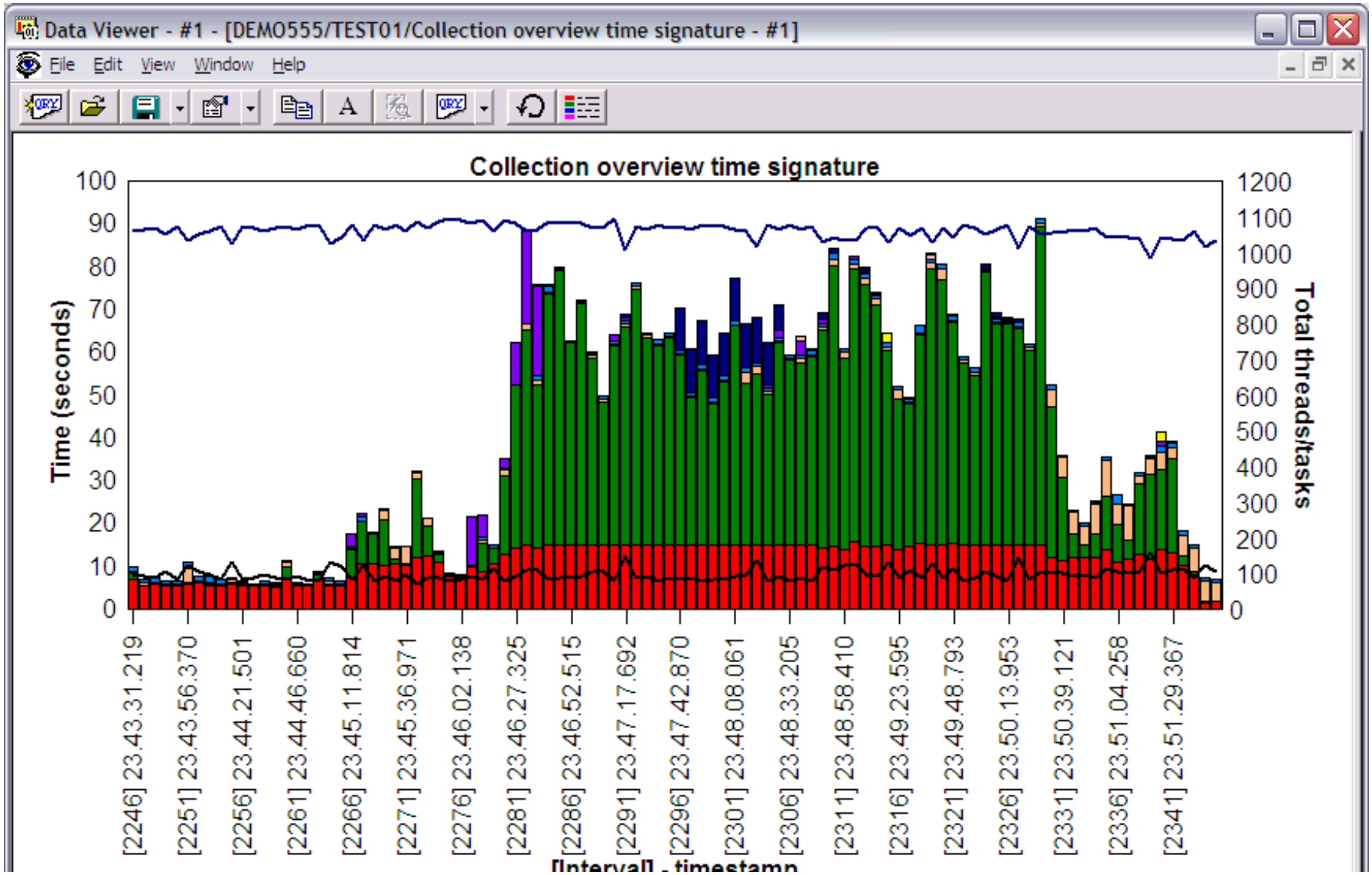
The heart of the iDoctor for iSeries application is the interface shown below. All components are displayed within this GUI each within a seperate "tree/list" view. Each view can be resized, tiled or manipulated at runtime. Multiple component views (a.k.a tree/list) views may be in use within the Main Window as needed.



[The Main Window]

Whenever you wish to look at a table or graph, another window called the Data Viewer will be opened for that purpose. You can have several Data Viewers open at one time over a single main window. Only one Main Window can be open per running instance of the iDoctor application.

An example of a Data Viewer window is shown below.



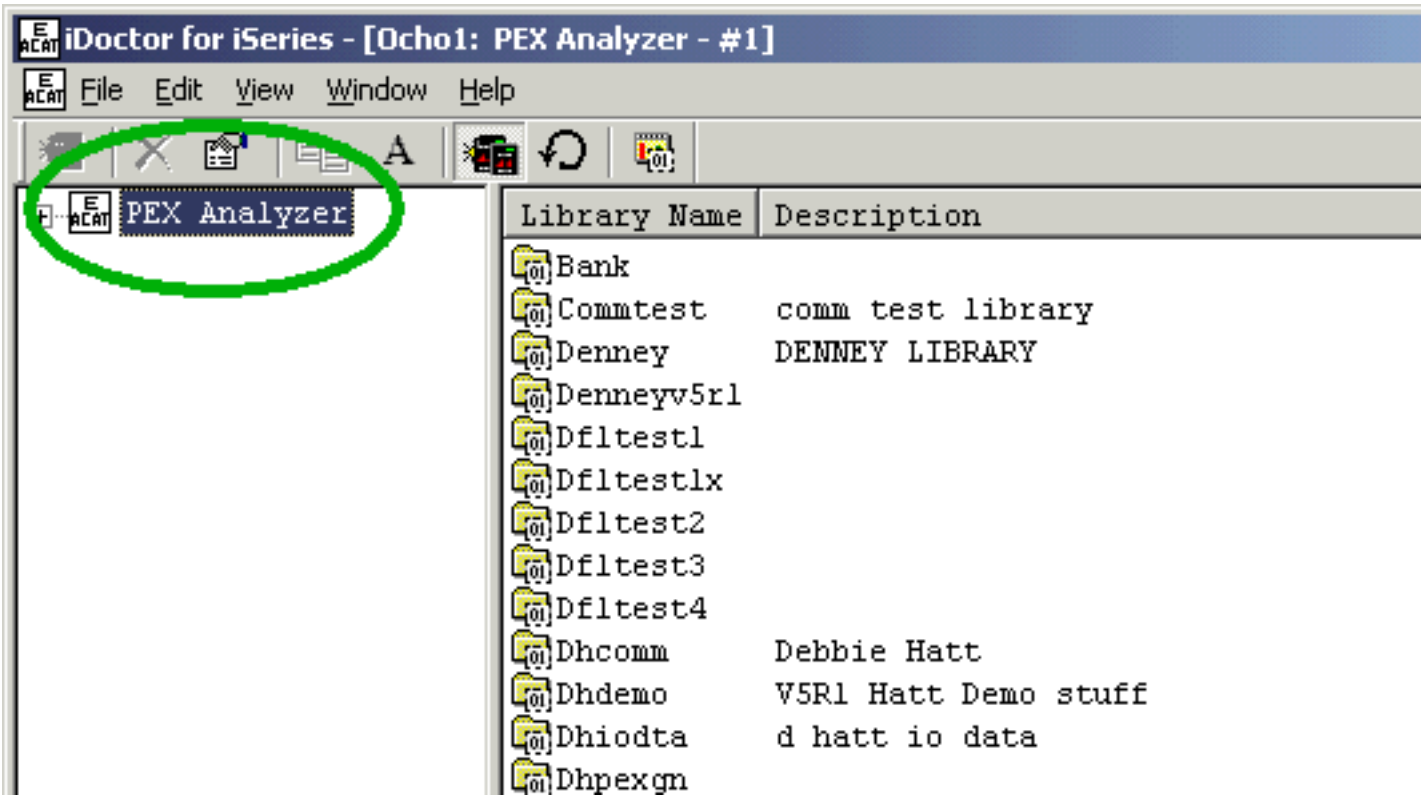
[The Data Viewer]

The Main Window and the Data Viewer as well as other interfaces for defining queries and graphs are common across all components. These interfaces will be defined in greater detail in the next sections.

Table of Contents | Previous | Next

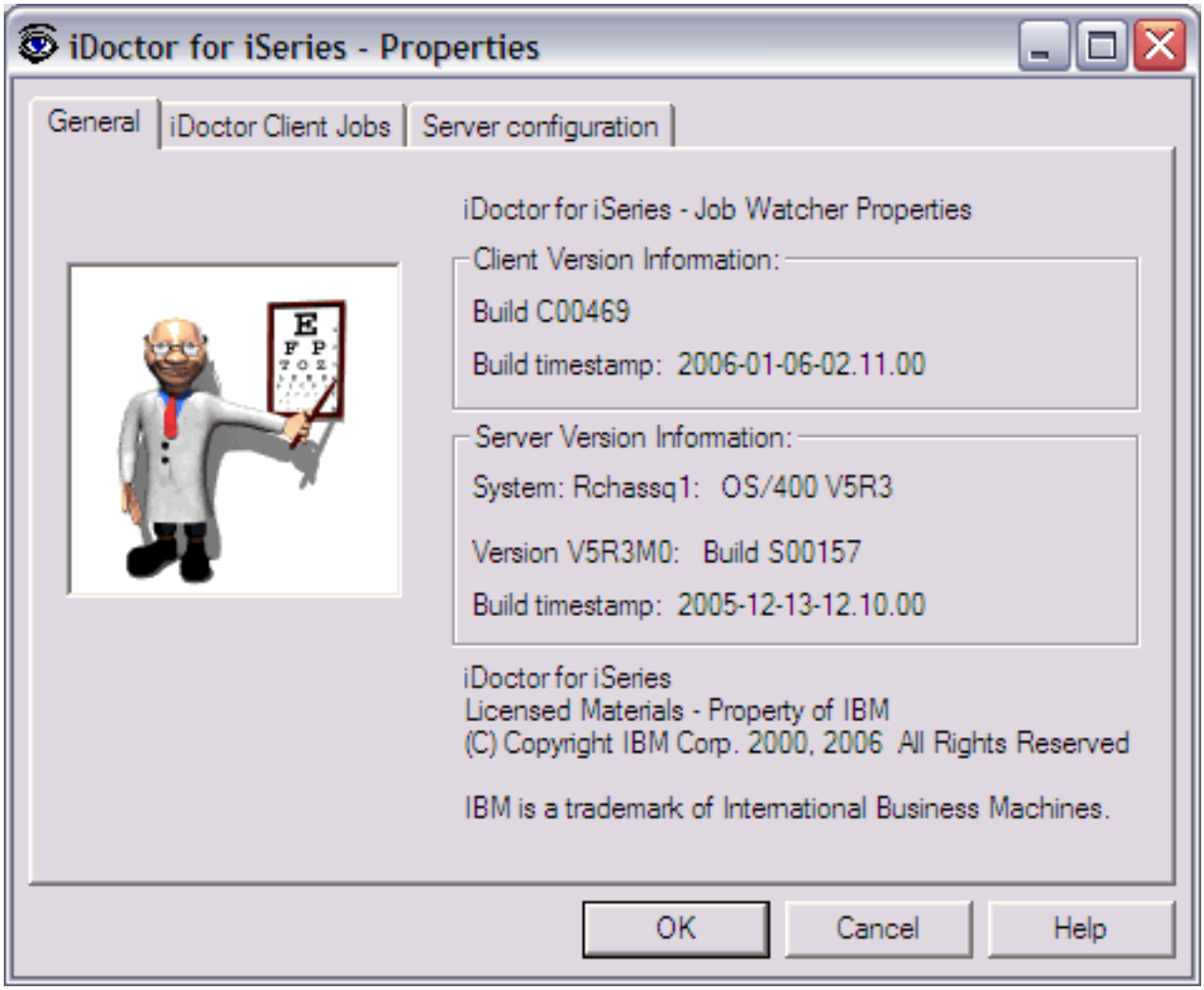
1.0.1 Component Property Pages

Each component view has a property page available by right-clicking on the component icon and choosing the Properties... menu. The component icon is either the 'PEX Analyzer', 'Job Watcher', 'Heap Analyzer' or 'Object Explorer' icon depending on the particular component view you are working with.



[PEX Analyzer component icon]

An example of a component property page for Job Watcher is shown below:



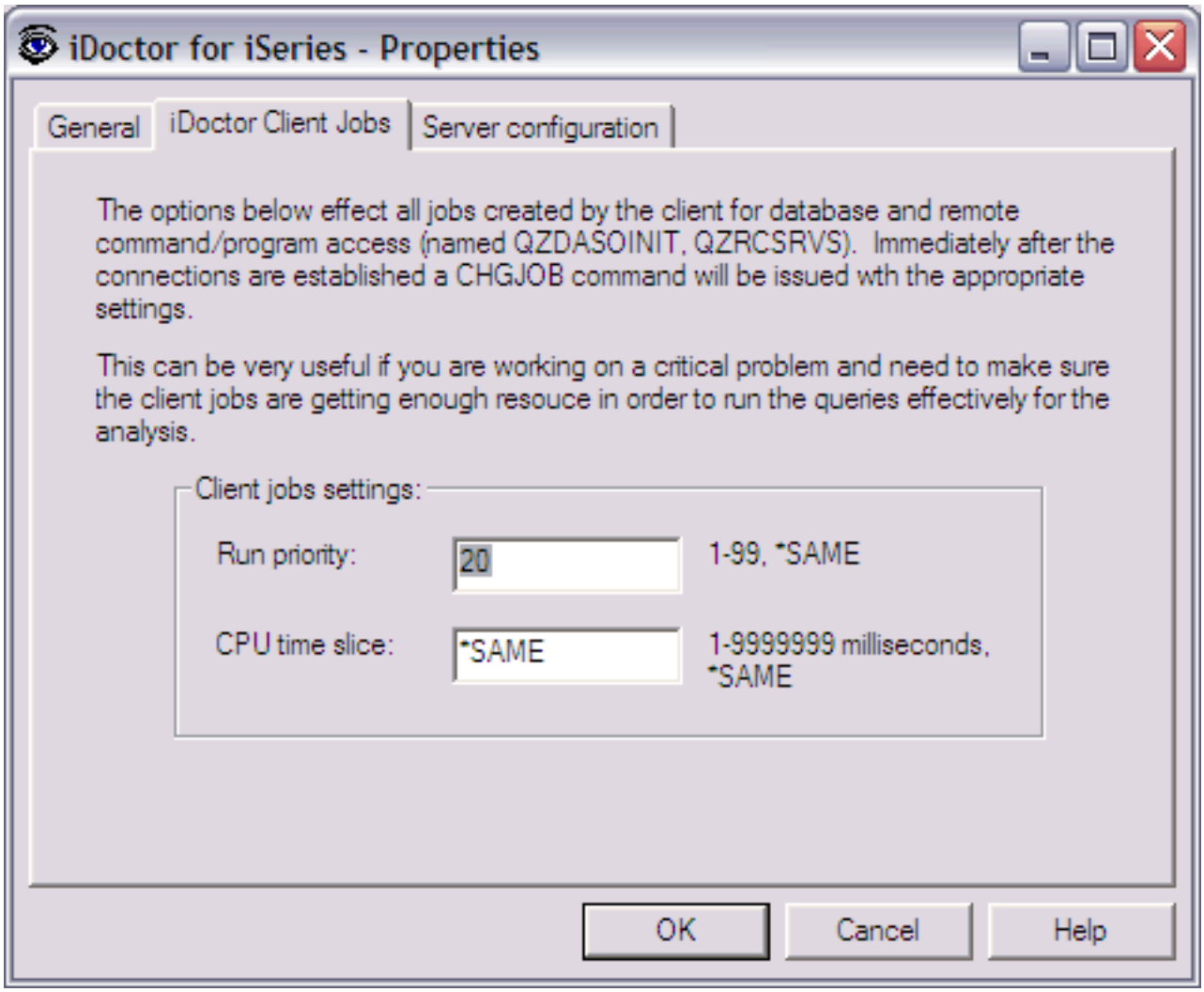
[Job Watcher component properties displaying client and server version information.]

The following information is supplied within the General page of this window:

Client Version Information	**Description**
Version	The iDoctor for iSeries component version/release/mod level of the client. **Note:** The version and build number of the client is always the same regardless of the component you are currently working with.
Build	The build number of the client your are currently running. This number goes up everytime a new build is run for internal (IBM only) testing or for customer use. There will be "internal only" builds resulting in perceived gaps in the build number sequence for customers, but please disregard this.
Build timestamp	The date/time the client build was produced. This value is shown in yyyy-mm-dd-hh.mm.ss format.

Server Version Information	Description
System name	The system that the current component view is connected to.
OS/400	The version and release of OS/400 on the active system.
Version	The iDoctor for iSeries component version/release/mod level installed on the server. The version and build number are different for each component. Only the version for the component you launched this window from will be listed.
Build	Build number of the component installed on the server side
Build timestamp	The date/time the server build was produced. This value is shown in yyyy-mm-dd-hh.mm.ss format.

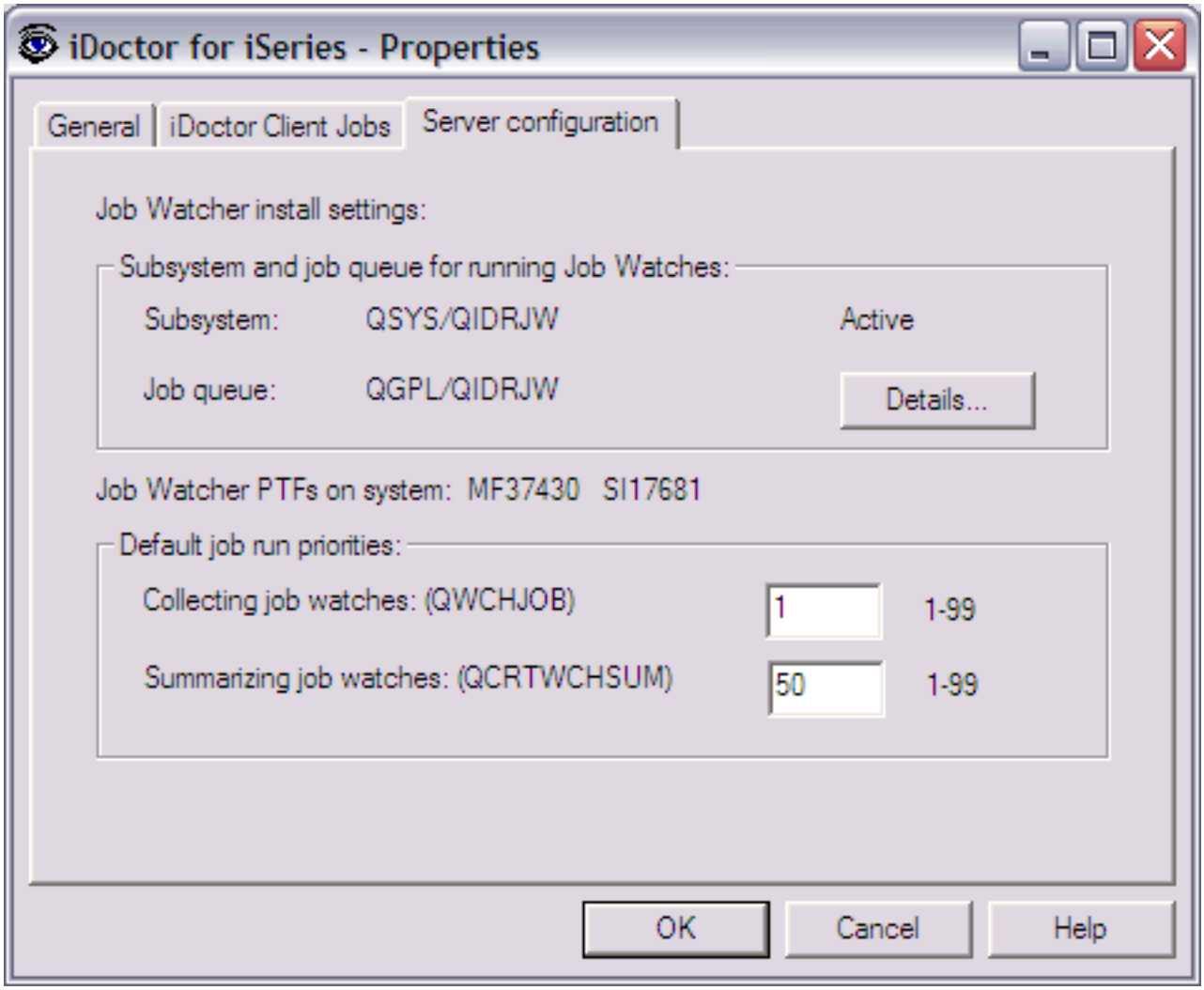
The following is an example of the iDoctor Client Jobs page:



This page lets you set the run priority and CPU time slice of all iDoctor client jobs. This should only

be set by advanced users. You must shut down the client and restart in order for these changes to take effect.

The following is an example of the Server configuration page.



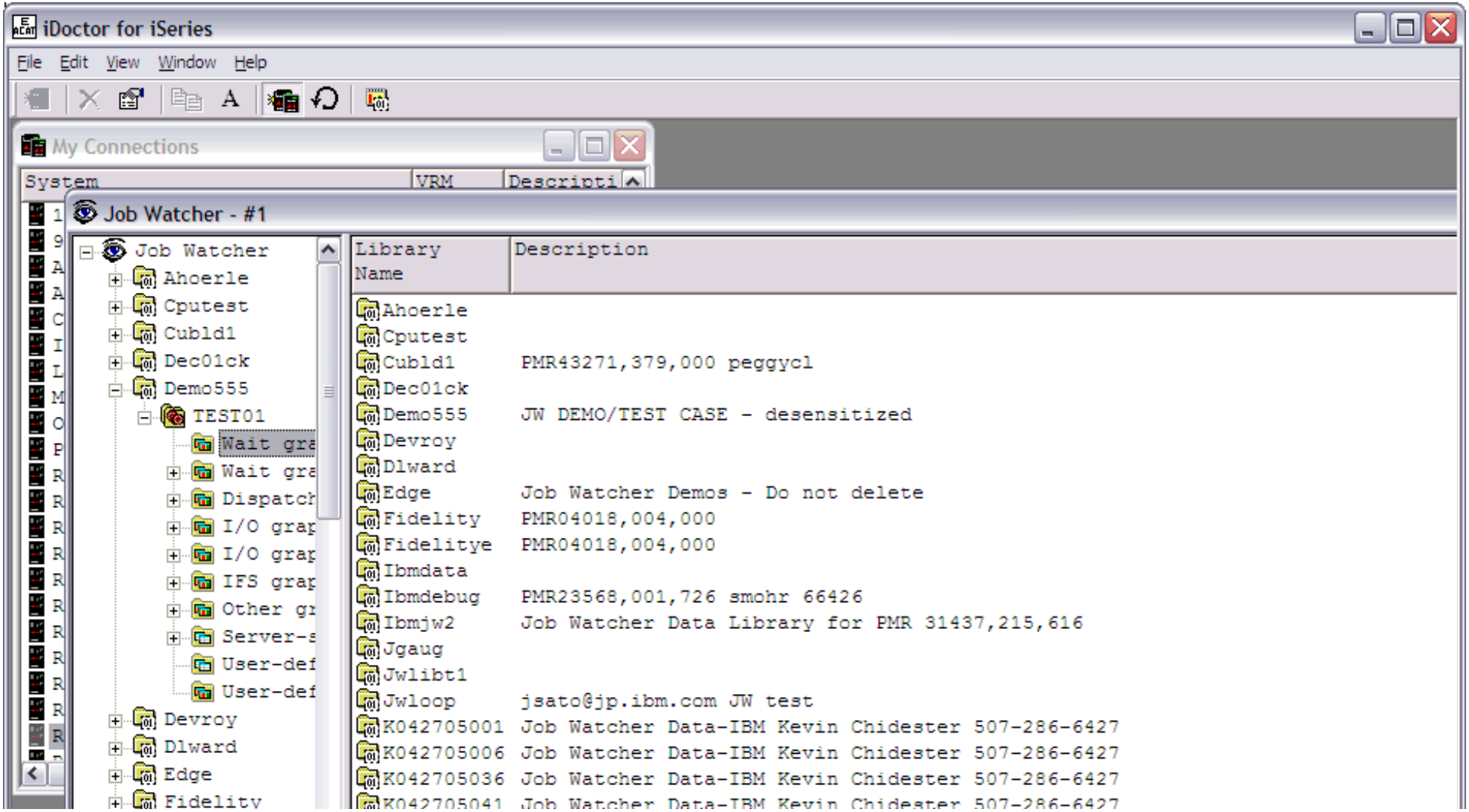
The subsystem and job queue used for running Job Watches is shown on this page. If you were running PEX Analyzer you would see the subsystem and job queue used for running PEX Analysis jobs.

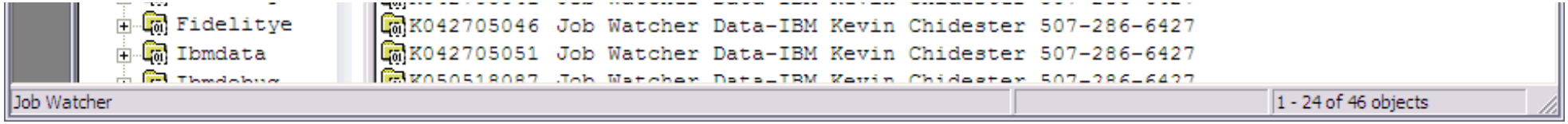
Table of Contents | Previous | Next

1.1 The Main Window

The Main Window displays the various component views as well as some additional views such as the "My Connections View". Each of these views within the Main Window will be discussed in greater detail in the next sections. This section will discuss the general use of the Main Window within iDoctor for iSeries.

The heart of the iDoctor for iSeries application is the interface shown below. All components are displayed within this GUI each within a seperate "tree/list" view. Each view can be resized, tiled or manipulated at runtime. You can have as many component (a.k.a "tree/list") views up at one time as you wish.





[The Main Window]

Main Window Menu Options

The table below outlines the different types of operations that may be performed within the Main Window of iDoctor for iSeries.

File Menu	Description
Connect	Connects to the selected system and displays the list of available components on that system. This menu is only enabled when the My Connections View is open and a system name is selected.
Open New Data Viewer	Opens an empty Data Viewer for the active component view. This menu is only enabled when a component view is open and has focus.
Close	This will close the active view within the Main Window.
Add Connection	Use this menu to add a connection to the My Connections View. This menu is only available when the My Connections View is active.
Delete	This will delete a conneciton from the My Connections View. This menu is only available when the My Connections View is active.
Exit	Exits the iDoctor for iSeries application. All open windows including Data Viewers will be closed down.

Edit Menu	Description
Copy	Copies the current selection from the active view to the clipboard. This is only enabled when the actve view is a list view (i.e. You can't copy text from the tree portion of the tree/list view).
Set Font	Displays a window allowing you to change the font used for the list views in the iDoctor for iSeries application.
Preferences	Displays the preferences window letting you work with iDoctor for iSeries user settings.

View Menu	Description
My Connections	This menu will either show or hide the My Connections view. If the view is already open there will be a checkmark next to the menu.
PEX Definitions	Brings up a window that lets you work with PEX definitions on your system. This is only available when a PEX Analyzer component view is open and has focus.
Remote Command Status	This menu will either show or hide the Remote Command Status view. If the view is already open there will be a checkmark next to the menu.
Toolbar	This menu will either show or hide the toolbar. If the toolbar is already visible there will be a checkmark next to the menu.
Status Bar	This menu will either show or hide the status bar. If the status bar is already visible there will be a checkmark next to the menu.
Refresh Selected	This menu will refresh the currently selected portion of a tree/list view. If a tree item is selected and this menu is clicked, everything underneath the tree item, including the tree item will have its data refreshed. If the list has focus and this menu is clicked, the entire list will be refreshed.

Window Menu	Description
Cascade	Use this menu to rearrange all views in the Main Window in an overlapping sequence starting in the upper left corner of the window.
Tile Horizontally	Use this menu to rearrange all views in the Main Window such that each view will have an equal distribution of the available height in the Main Window. The views will not overlap each other.
Tile Vertically	Use this menu to rearrange all views in the Main Window such that each view will have an equal distribution of the available width in the Main Window. The views will not overlap each other.

Help Menu	Description
Contents	This menu will launch your web browser and takes you to the online documentation start page.
iDoctor for iSeries downloads	Launches your web browser and takes you to the iDoctor download page.
About iDoctor for iSeries	This displays version information for the iDoctor for iSeries client.

Table of Contents | Previous | Next

1.1.1 Component Views

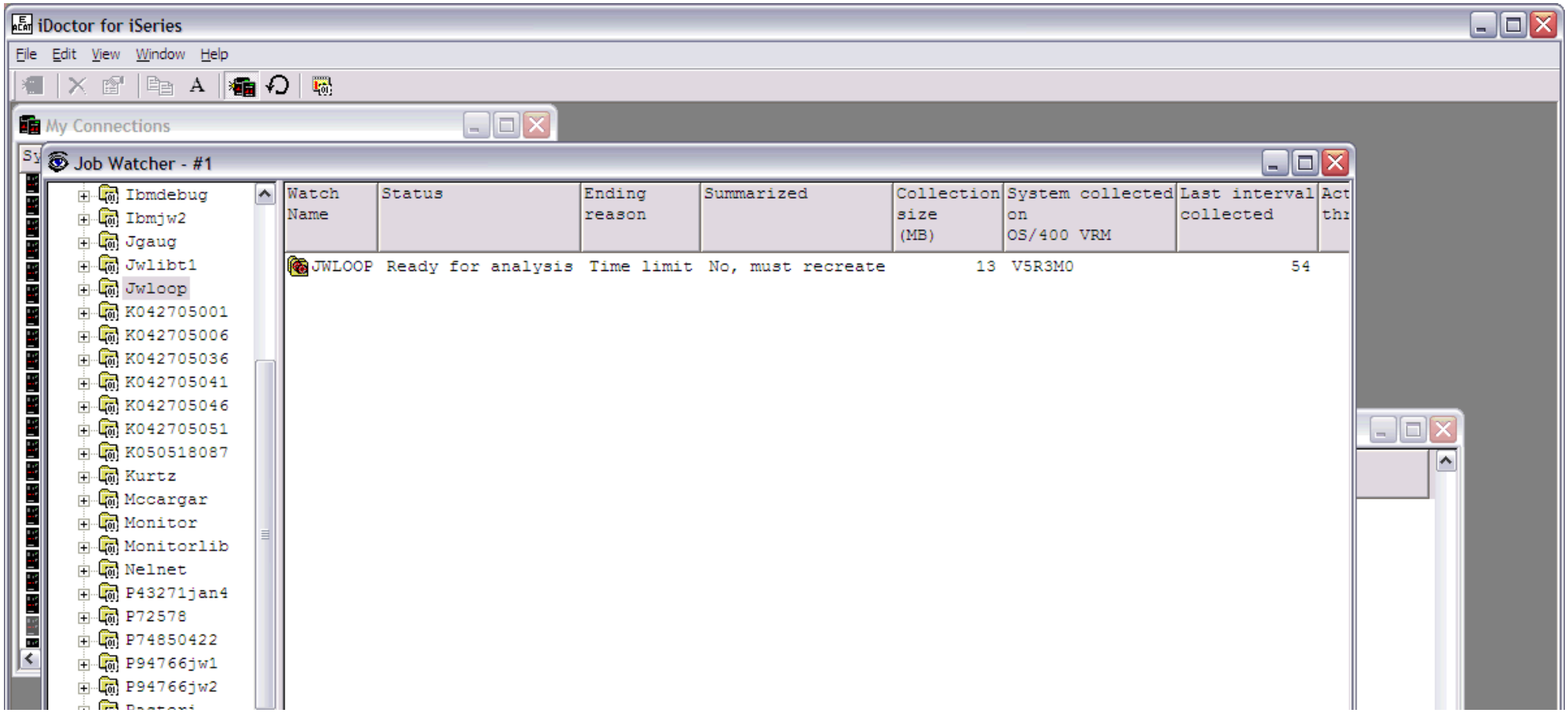
Component views are also known as the "tree/list" views and are the primary means of working with any of the iDoctor for iSeries components. You can have as many component views open within a Main Window as desired.

Component views look and feel consistently across the various components. The tree represents the hierarchy of libraries, collections, or other objects that you are dealing with on the system you are connected to. Moving down the tree lets you drill down into a specific library, collection, etc. Your current selection in the tree is always displayed in the list portion of the tree/list.

The menus shown when right clicking a folder within a component view are also consistent. There will usually be an Explore menu letting you open the next branch of the tree or other menus that you'd expect like delete, copy, or properties.

Whenver a particular component view is active, the Data Viewer icon on the toolbar (far right) will be enabled. This lets you easily open files and run queries over data on the system you are connected to. Visit the Data Viewer documentation for more information on using the Data Viewer.

Because of the tendency to deal with large amounts of data and a desire to have the client perform optimally (reduce comm dependency, etc), refresh has been implemented in a way unlike most other applications. The refresh toolbar icon or menu will refresh only the contents of the selected tree branch. For example if a library is selected in the tree, only the contents of the library will be refreshed, not the list of libraries in the tree. Refreshing the list of libraries would require selecting the folder above the list of libraries (the component icon) before using the refresh option on the toolbar.



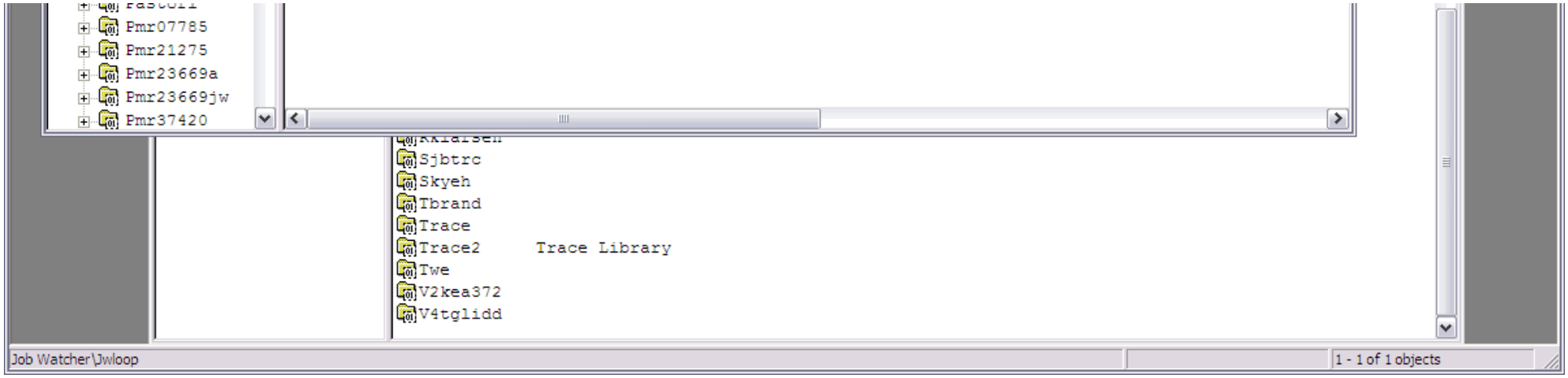
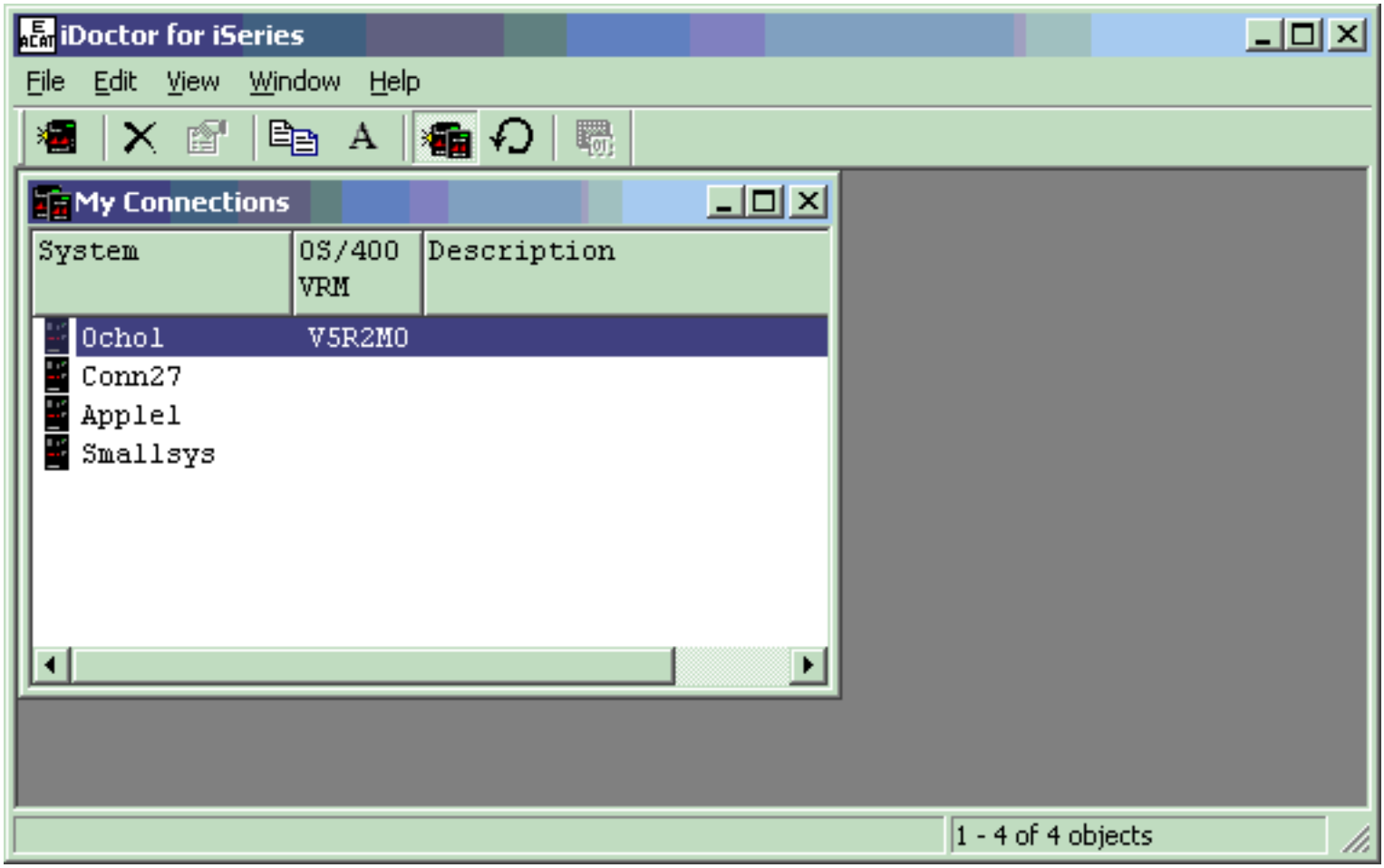

Pmr07785
Pmr21275
Pmr23669a
Pmr23669jw
Pmr37420
Sjbtrc
Skyeh
Tbrand
Trace
Trace2 Trace Library
Twe
V2kea372
V4tglidd
Job Watcher\Jwloop
1 - 1 of 1 objects

Table of Contents | Previous | Next

1.1.2 My Connections View

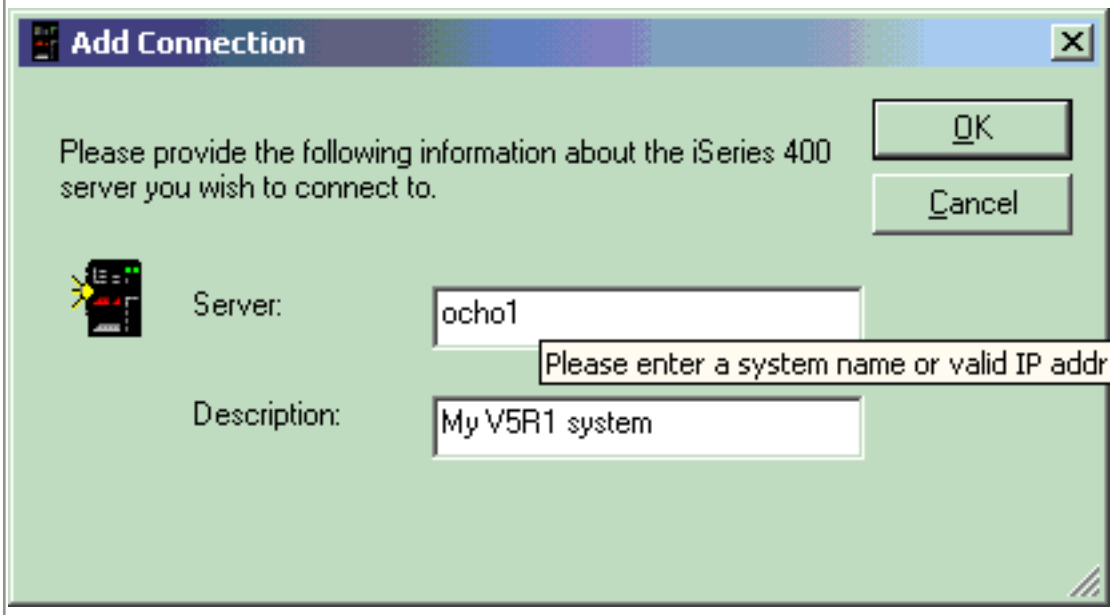
The My Connections view allows you to work with connections to any system you have already defined on your PC through iSeries Navigator or via iDoctor for iSeries. You can easily add or remove connections to other systems through this view. The primary purpose of this view is to provide a quick and easy way to launch the iDoctor for iSeries components for any system desired.

The list of connections shown is for the currently active environment. You can change the currently active environment by right-clicking on the list and choosing the Change Environment... menu.



[The Main Window displaying the My Connections View]

The following actions may be taken in the My Connections View by right-clicking on one of the systems in the list:

Popup Menu	Description
Connect	Connects to the selected system and displays the iDoctor components window with the status of each component installed on the system.
Add Connection	Use this menu to add a connection to the My Connections View. You will see the following dialog:  Please provide the system name or IP address and an optional description and click OK to register the system on your PC and add it to the connections list. Any systems you add will also be shown in iSeries Navigator.
Delete	This will delete a connection from the My Connections View. Any system connections you delete from this list will also be deleted in iSeries Navigator.
Change Environment...	This menu lets you change the currently active environment as desired. Each environment represents a list of connections. The environments may be created/imported through iSeries Navigator. Any connections you add or remove from the environment within iDoctor for iSeries take effect within iSeries Navigator.

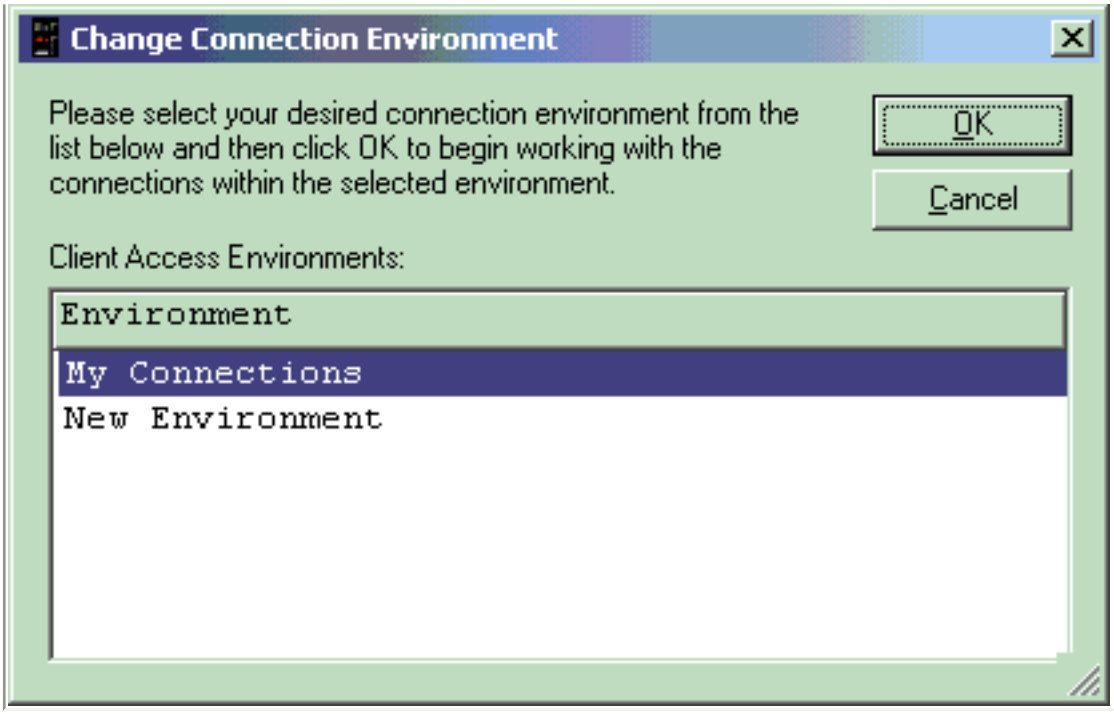

Change Connection Environment
Please select your desired connection environment from the list below and then click OK to begin working with the connections within the selected environment.
OK
Cancel
Client Access Environments:
Environment
My Connections
New Environment

Table of Contents | Previous | Next

1.1.3 iDoctor Components Window

The components window provides the status of the iDoctor components installed on the system selected from the Connections View.
This window allows a user to launch a component, change the user signed on to the system or apply an access code. After applying an access code the component list will refresh to indicate any changes in status (Not Authorized -> Available)

PTF checking for each component does not take place through this interface. PTFs are checked when creating a collection.

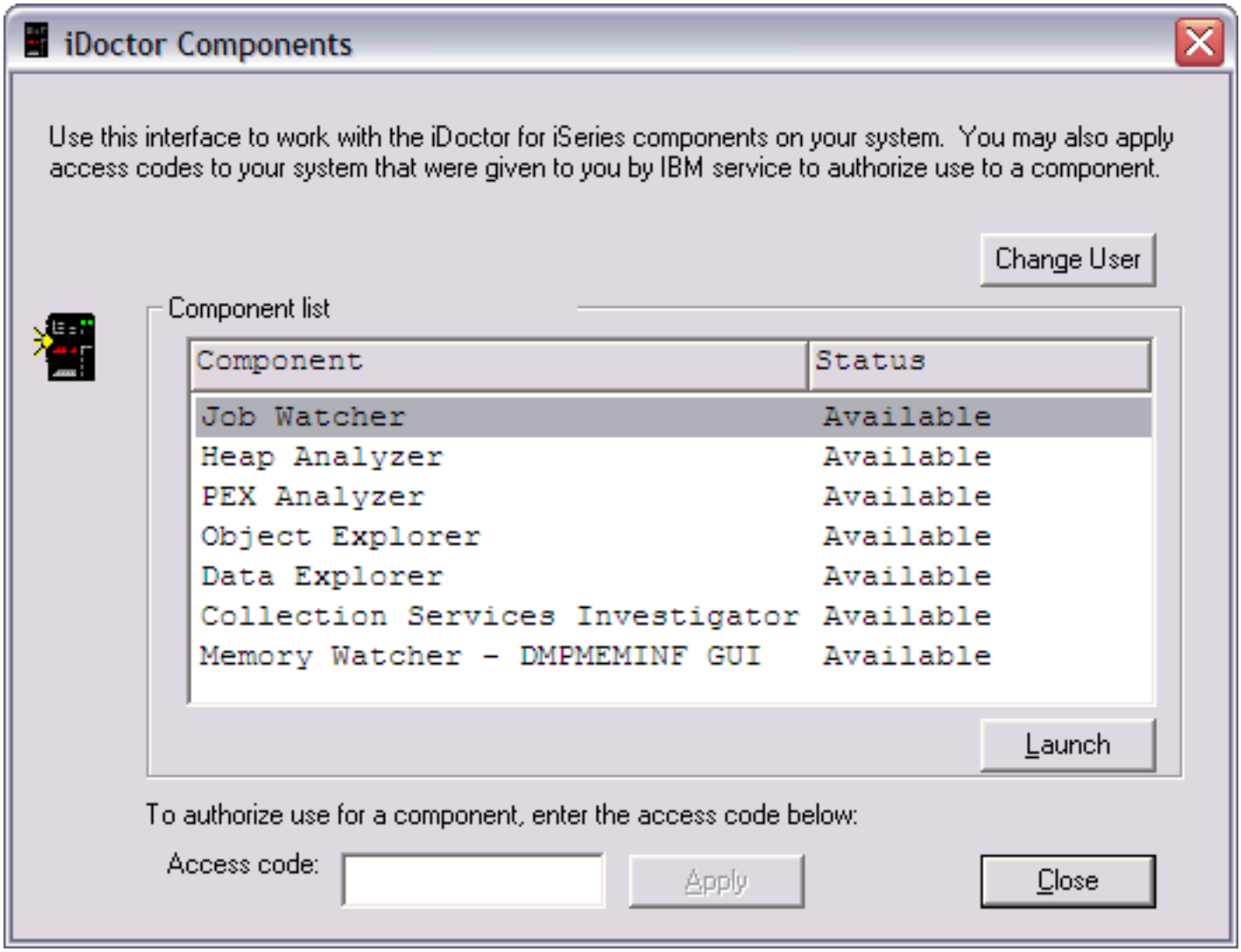


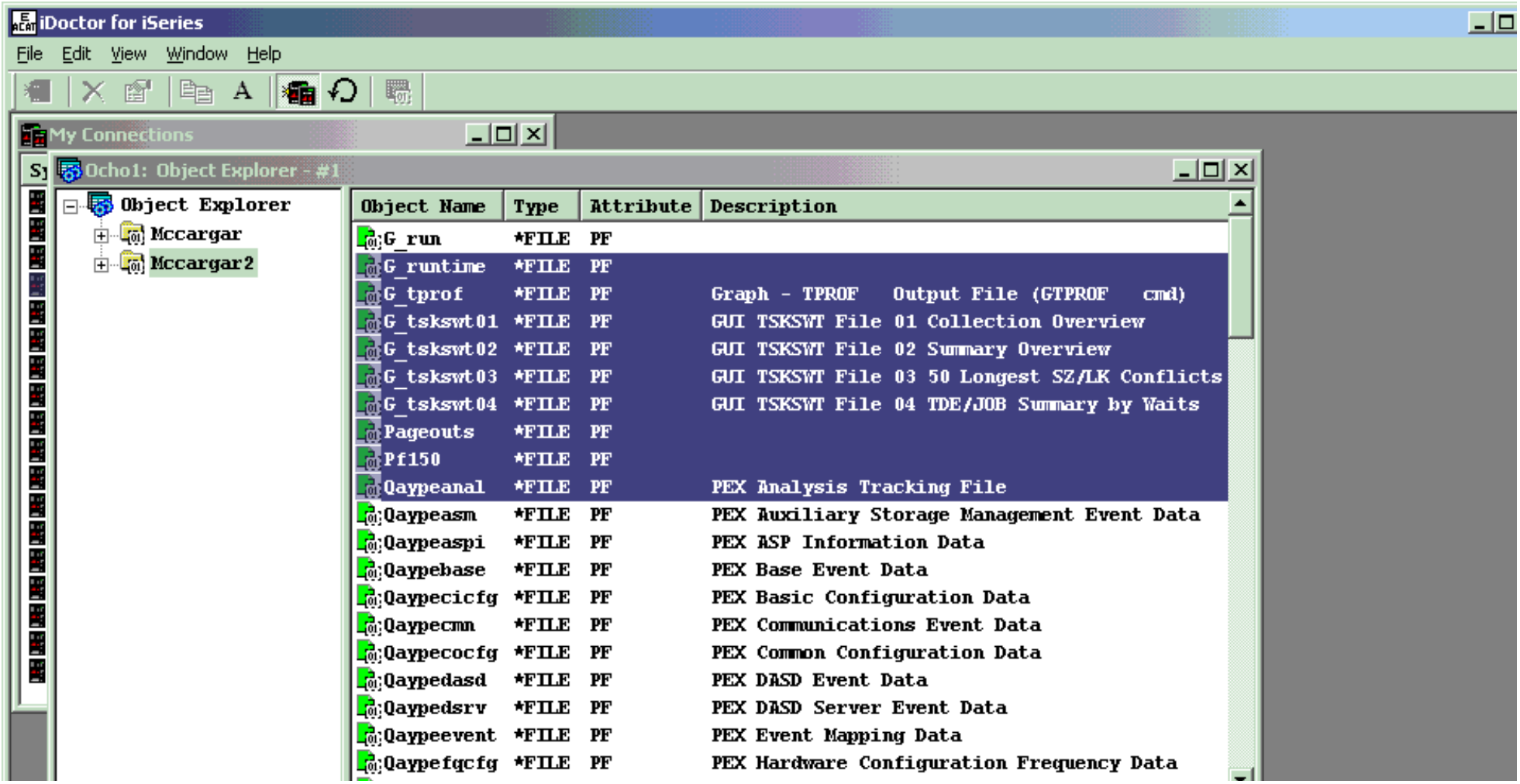
Table of Contents | Previous | Next

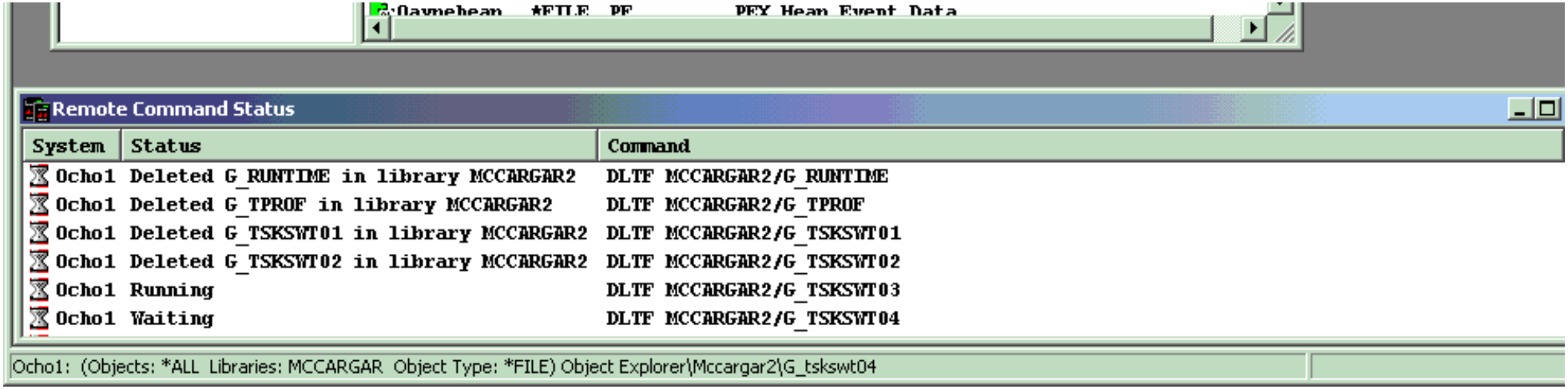
1.1.4 Remote Command Status View

The Remote Command Status view shows you the status of certain remote commands being executed on a system. This allows you to perform lengthy operations like copy objects, or delete files without tying up the GUI (i.e. you can do other things like view a report while the remote commands are executing).

Depending on the function/feature being used (like copying objects in Object Explorer) you will see one or more commands in the remote command status view. As each command completes you will immediately see its result or error message in the view.

You can also close this window and reopen it later while commands are being executed to periodically check the status of the commands issued.





[The Main Window displaying an Object Explorer View and the Remote Command Status View]

The following actions may be taken in the Remote Command Status View by right-clicking on one (or more) entries in the view:

Popup Menu	Description
Remove Completed	Use this menu to remove all remote command entries that have completed.
Remove Selected	Use this menu to remove all selected remote command entries from the view.
Remove All	Use this menu to remove all remote command entries from the view.

Table of Contents | Previous | Next

1.2 Field Selection Window

The Field Selection Window is a generic way to work with the fields shown in the list portion of a tree/list view. This window is available via the 'Select fields...' menu from any objects that has field selection enabled. Not all folders in the tree have field selection enabled, only those that have a large number of available fields to display.

The field selection window lets you drag and drop fields into the desired positions. The first field in the list portion of a tree/list view is fixed, but the rest of the fields can be reordered or hidden as desired. An example of a Field Selection Window is shown below.

Any changes you make are saved to your system registry and reused the next time you open the view you are working with. To restore to the iDoctor-ship default ordering click the "Default" button. The "Toggle Selected" button is a fast way to toggle the show checkbox for several selected fields in the list at once. To select multiples hold down the ctrl or shift key while clicking your mouse on entries in the list.

All components make use of this interface to help make using iDoctor more flexible.

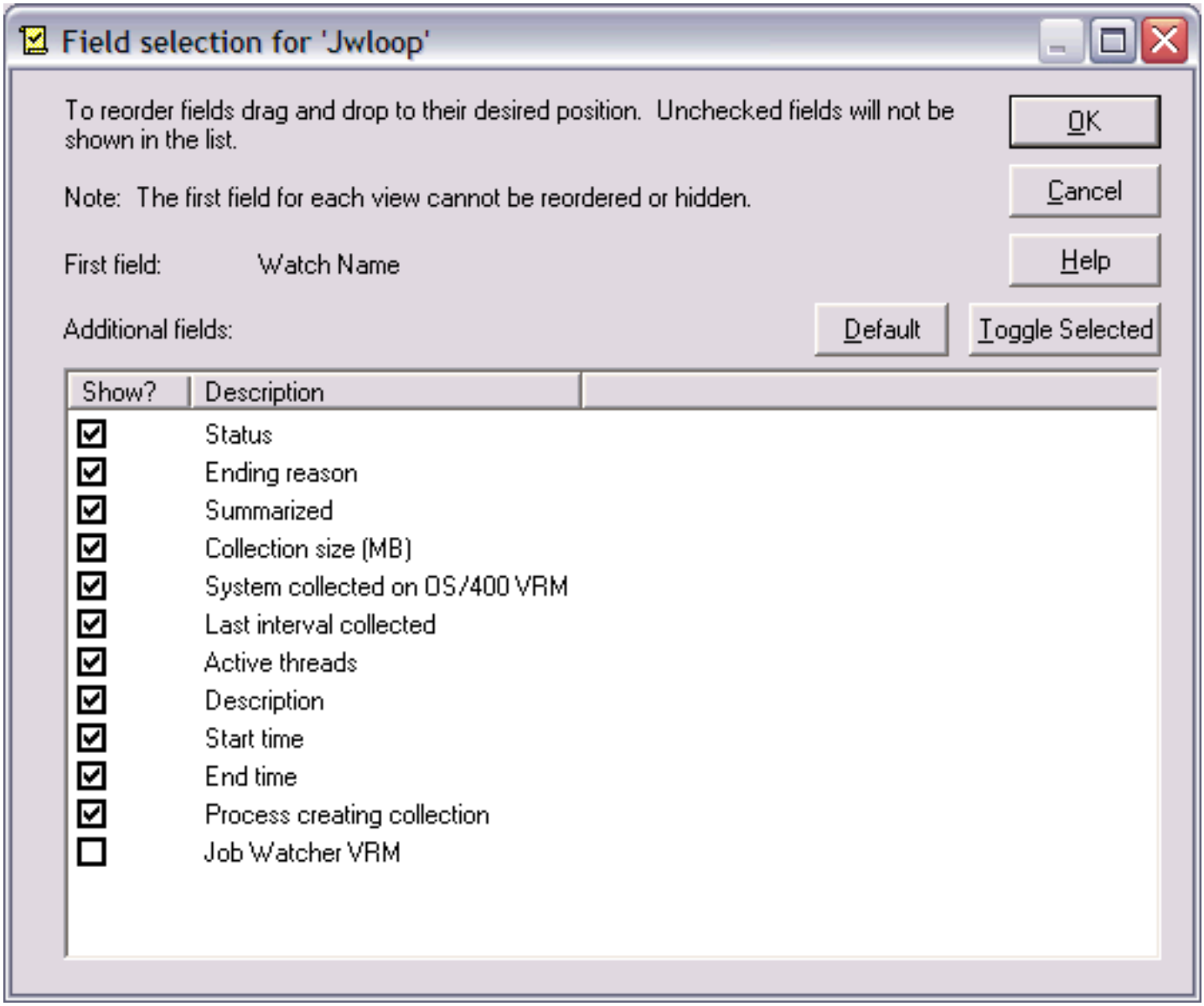
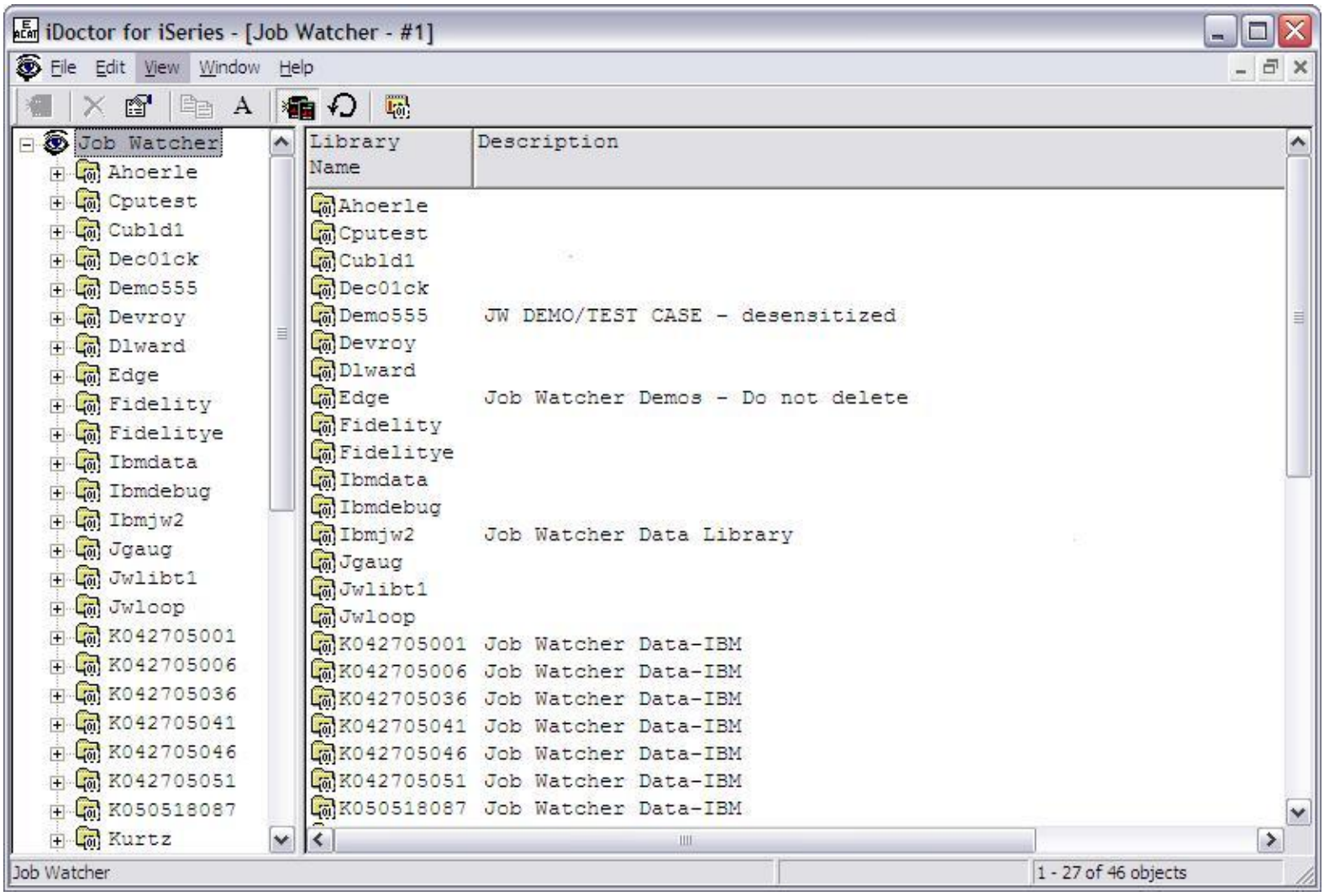

Field selection for 'Jwloop'
To reorder fields drag and drop to their desired position. Unchecked fields will not be shown in the list.
Note: The first field for each view cannot be reordered or hidden.
First field: Watch Name
Additional fields:
OK
Cancel
Help
Default
Toggle Selected
Show?
Description
Status
Ending reason
Summarized
Collection size (MB)
System collected on OS/400 VRM
Last interval collected
Active threads
Description
Start time
End time
Process creating collection
Job Watcher VRM

Table of Contents | Previous | Next

1.3 Libraries

A common characteristic of all component views in iDoctor for iSeries is that they all initially display a list of libraries on the system. The libraries shown depend on the particular component. PEX Analyzer shows libraries with PEX data, Job Watcher shows libraries with Job Watcher data, and Object Explorer shows all libraries on the system matching the current object filter criteria.



All libraries have detailed properties and a set of menu options available. This chapter will discuss each of the library property pages in iDoctor for iSeries as well as all the menu options for a library.

Table of Contents | Previous | Next

1.3.1 Menu Options

A library folder in iDoctor for iSeries has the following menu options available by right-clicking on the library icon:

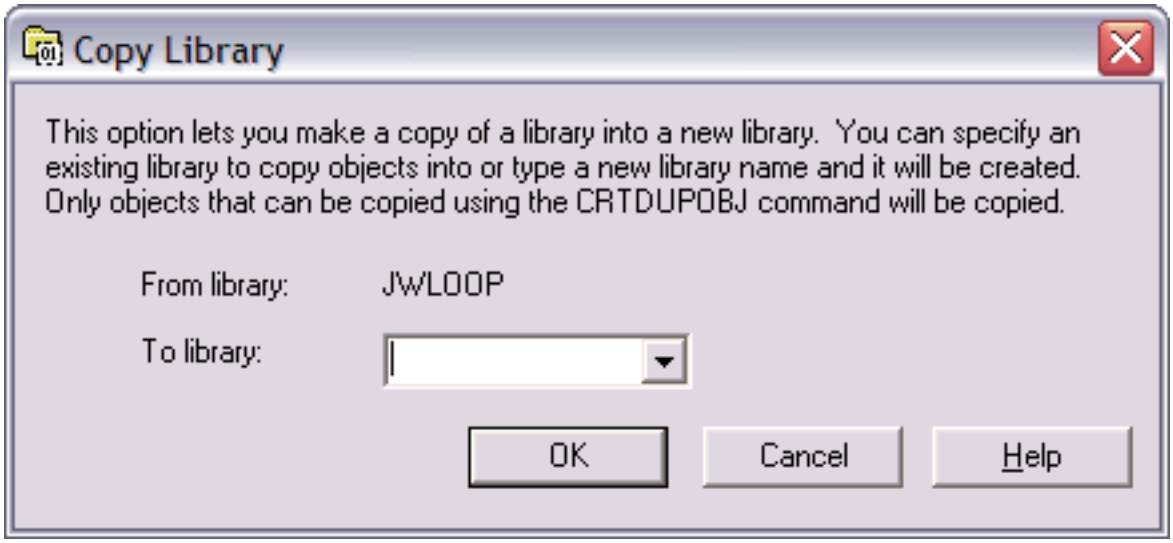
Menu	Description
Explore	Show the contents of the library (the contents depends on the component).
Select fields...	Displays the Field Selection Window which allows you to reorder the fields that are displayed as the contents of the library. For example in Object Explorer, libraries contain objects and this menu lets you configure which fields for the objects are visible within the library.
Copy...	Allows you to copy the library's contents into a new library or into an existing one.
Save...	This option lets you save the library's contents into a save file.
Transfer to...	Allows a user to create a save file of a library and transfer it to another system.
Clear	This option clears a library (deletes all objects in the library).
Delete	Deletes the library.
Rename	Renames the library.
Properties	Displays the property pages for the library.

Depending on the component, a library folder will have another menu option available to create a collection.

Table of Contents | Previous | Next

1.3.2 Copying

A library may have its contents copied into a new library or into an existing library by using the Copy... menu available by right-clicking on a library within iDoctor for iSeries. This option is an interface over the CPYLIB command. The progress of the library being copied may be viewed using the Remote Command Status View

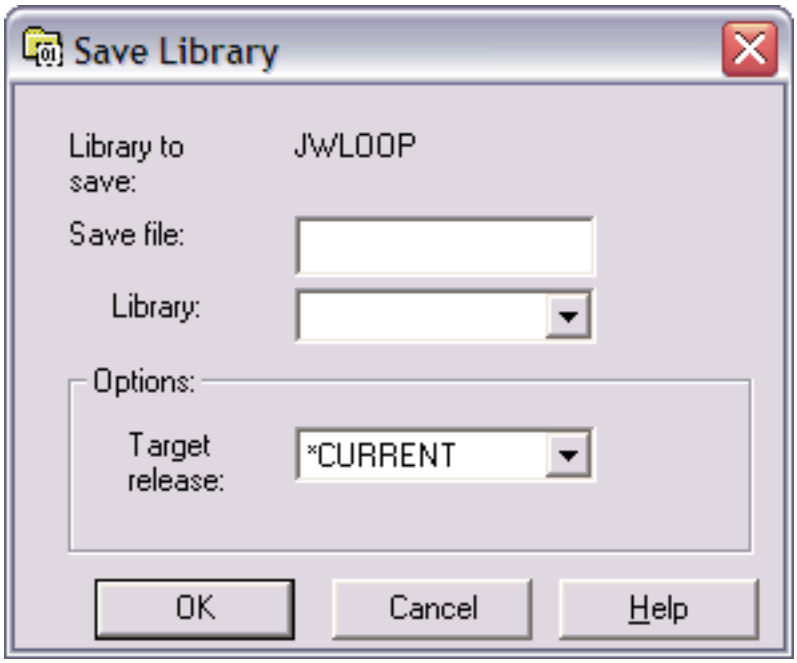


Field	Description
From library	Displays the name of the library to be copied.
To library	The name of the library that will receive the contents of the from library. By clicking the down arrow you can choose from a list of all libraries on the system.

Table of Contents | Previous | Next

1.3.3 Saving

A library's contents can be saved using the Save... menu available by right-clicking on a library within iDoctor for iSeries. This option is a interface over the SAVLIB command. This interface is restricted to saving the library to a save file and is missing some of the advanced options on the command. The progress of the library being saved may be viewed using the Remote Command Status View



Field	Description
Library to save	The name of the library to be saved.
Save file/library	The name of the save file and library to save the contents of the library into. If the save file doesn't exist it is created. If the save file does exist, you will be asked for confirmation before continuing.
Target release	Specifies the release of the operating system on which you intend to restore and use the object.

Table of Contents | Previous | Next

1.3.4 Clearing

A library's contents may be cleared using the Clear... menu available by right-clicking on a library within iDoctor for iSeries. This option will initiate (after confirmation) a CLRLIB command over the specified library. The progress of the library being cleared may be viewed using the Remote Command Status View

Table of Contents | Previous | Next

1.3.5 Deleting

A library's may be deleted using the Delete... menu available by right-clicking on a library within iDoctor for iSeries. This option is a interface over the DLTLIB command. The progress of the library being deleted may be viewed using the Remote Command Status View

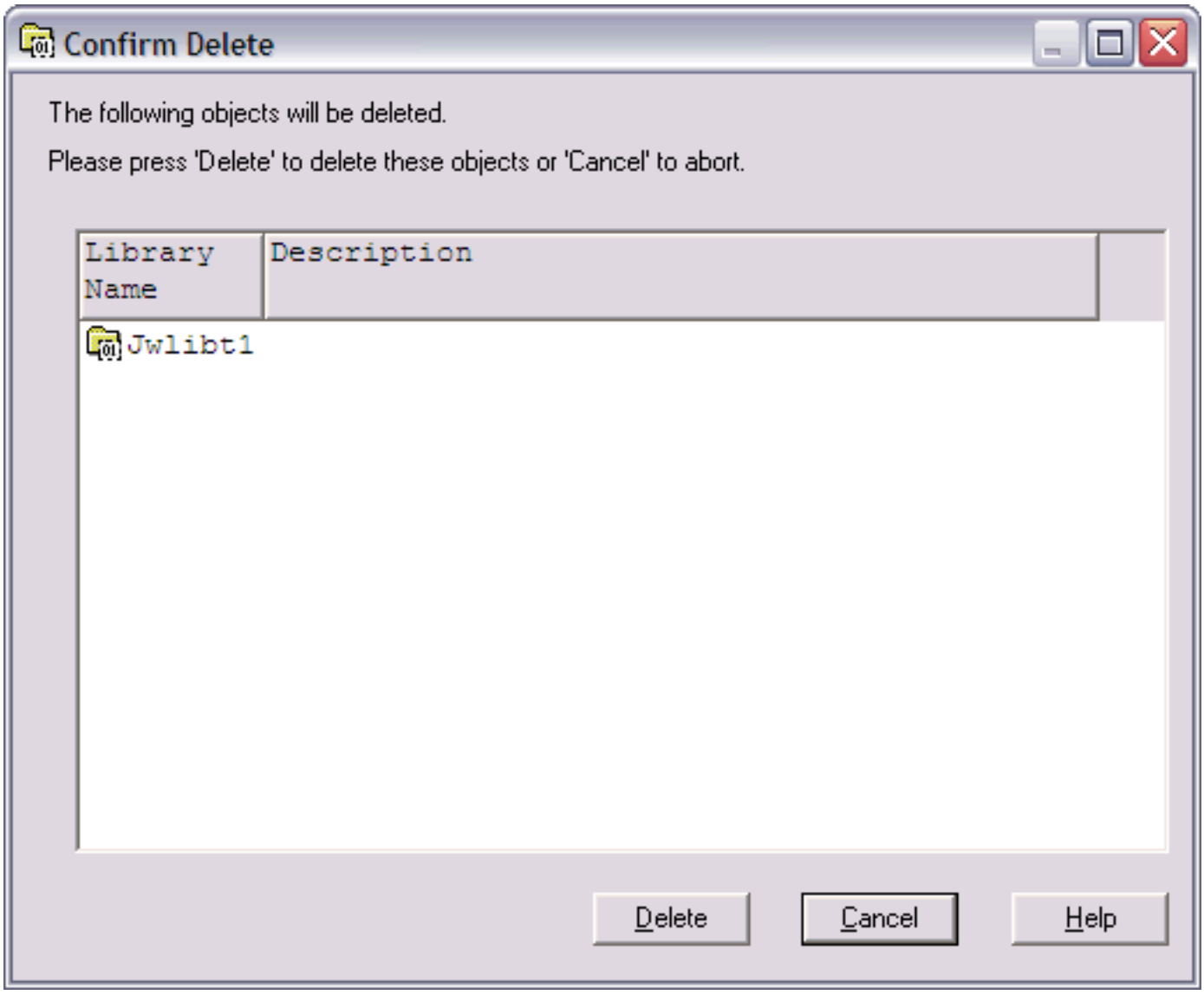
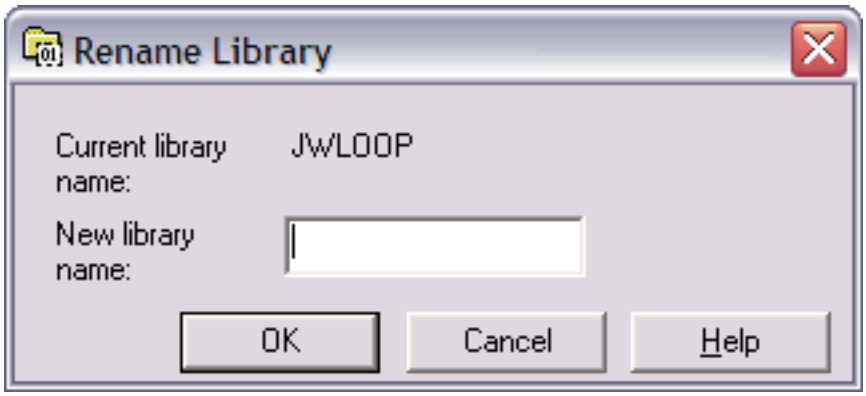


Table of Contents | Previous | Next

1.3.6 Renaming

A library may be renamed using the Rename... menu available by right-clicking on a library within iDoctor for iSeries. This option is a interface over the RNMOBJ command.



Field	Description
Current library name	The name of the library to be renamed.
New library name	The name to replace the current library name.

Table of Contents | Previous | Next

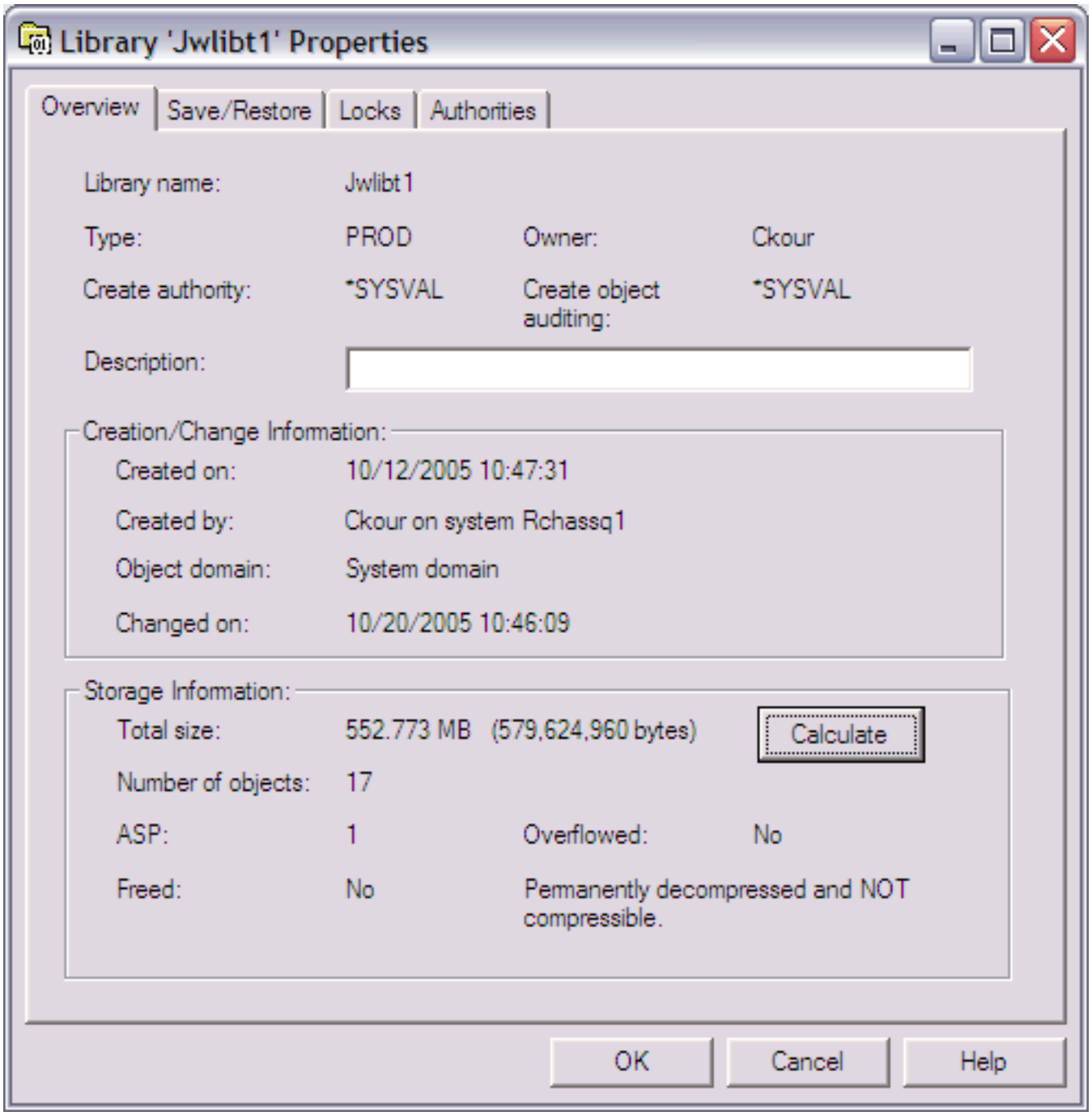
1.3.7 Properties

The library property pages are accessible by right-clicking on a library and choosing the Properties menu. The next sections discusses all of the library properties pages.

Table of Contents | Previous | Next

1.3.7.1 Overview

The overview property page for libraries displays basic information about the library, including the total size of all objects in the library.



The following information is listed on this page:

Field	Field Description
Library Name	Name of the library.
Type	Indicates the libraries type. **PROD** The library is a production library. Database files in production libraries cannot be opened for updating if a user is in debug mode and requested that production libraries be protected. **TEST** The library is a test library. All objects in a test library can be updated during test. See the STRDBG command for more details.
Owner	The name of the user profile which owns the library.
Create Authority	The default public authority used when an object is created into a library. This authority is given to the following users: - Users who do not have specific authority to the object. - Users who are not on the authorization list. - Users whose user group has no specific authority to the object. The valid values are: ***ALL** The user can perform all authorized operations on an object created in this library. ***CHANGE** The user can read the object description and has read, add, update, and delete authority to an object created in this library. ***EXCLUDE** The user is prevented from accessing an object created in this library. ***SYSVAL** The default authority for an object created in this library is determined by the value specified by the QCRTAUT system value. ***USE** The user can read the object and its description but cannot change them for an object created in this library.

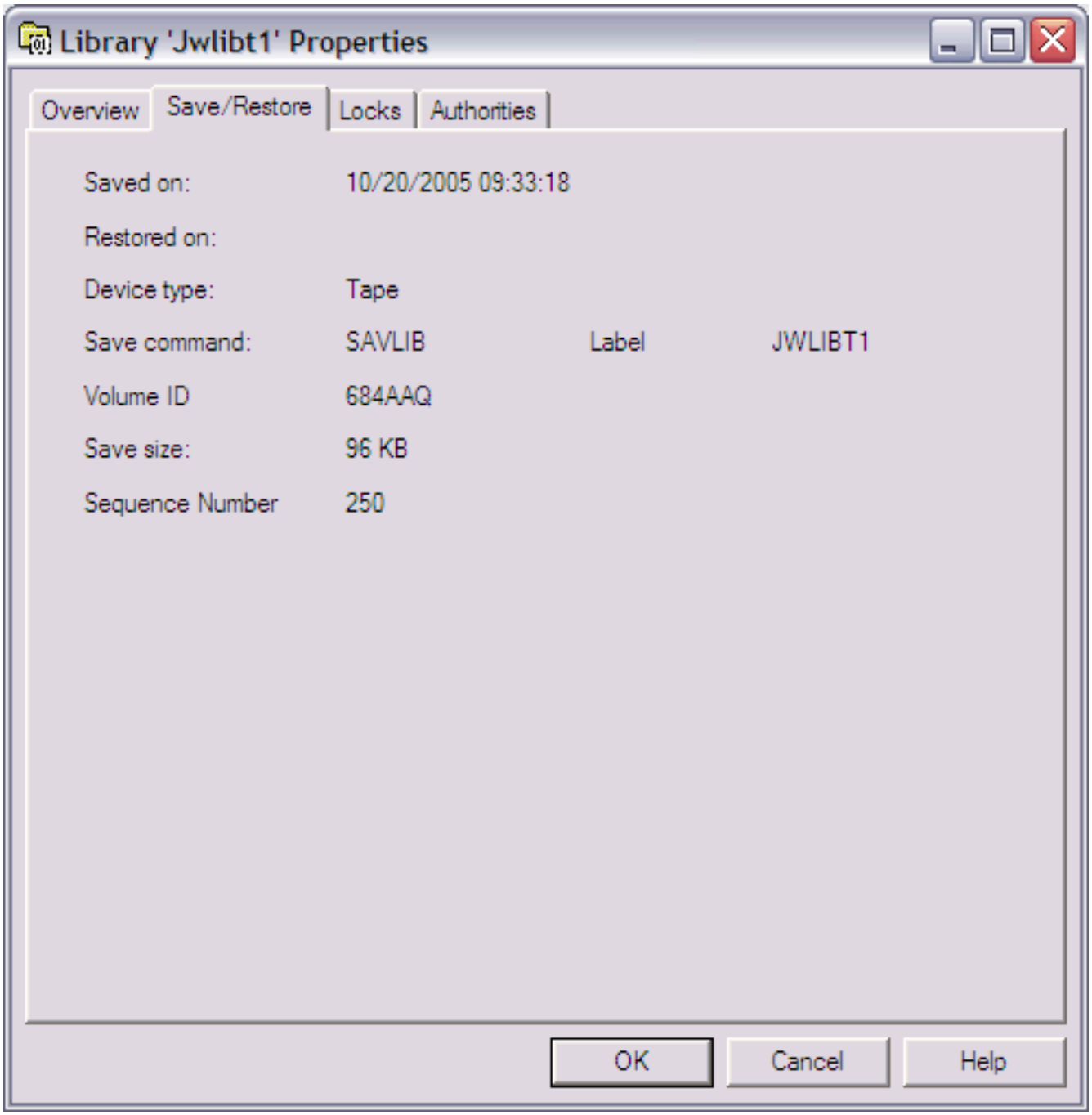
	Authorization list name The name of the authorization list that secures an object created in this library. The default public authority is taken from the authorization list, and the public authority for the object is specified as *AUTL.
Create Object Auditing	The auditing value for objects created in this library. The valid values are: ***ALL** All change or read access to the object is logged. ***CHANGE** All change access to the object by all users is logged. ***NONE** Use or change access to the object is not logged (no audit entry is sent to the security journal). ***SYSVAL** The value specified in the system value QCRTOBJAUD is used. ***USRPRF** The user profile of the user who accesses the object is used to determine if an audit record is sent for this access. The OBJAUD parameter of the Change User Auditing (CHGUSRAUD) command is used to turn auditing on for a specific user.
Description	Library description. You can change this value if you wish.
Created On	The date and time the library was created.
Created By	The name of the user who created the library and the system it was created on.
Object Domain	The domain that contains the object. The possible values are user domain or system domain.
Changed On	The date and time the library was changed.
Total Size	Total size of all objects in the library including the library itself. Click the calculate button to compute this value. Note: This calculation can take a long time (minutes) depending on the number of objects and members in the library.
Object Count	Total number of objects in the library.
ASP	Auxillary Storage Pool: A number indicating the identifier of the auxiliary storage pool from which storage space for the library was allocated.
Overflowed	Indicates if the object has overflowed the auxiliary storage pool it resides in.

Freed	Indicates the storage status of the object (Yes/No). If the storage status is freed, then the object is suspended, otherwise the object is not suspended.
Object Compression	Indicates the compression status of the object.

Table of Contents | Previous | Next

1.3.7.2 Save/Restore

The save/restore property page displays information about how and when the library was last saved or restored.



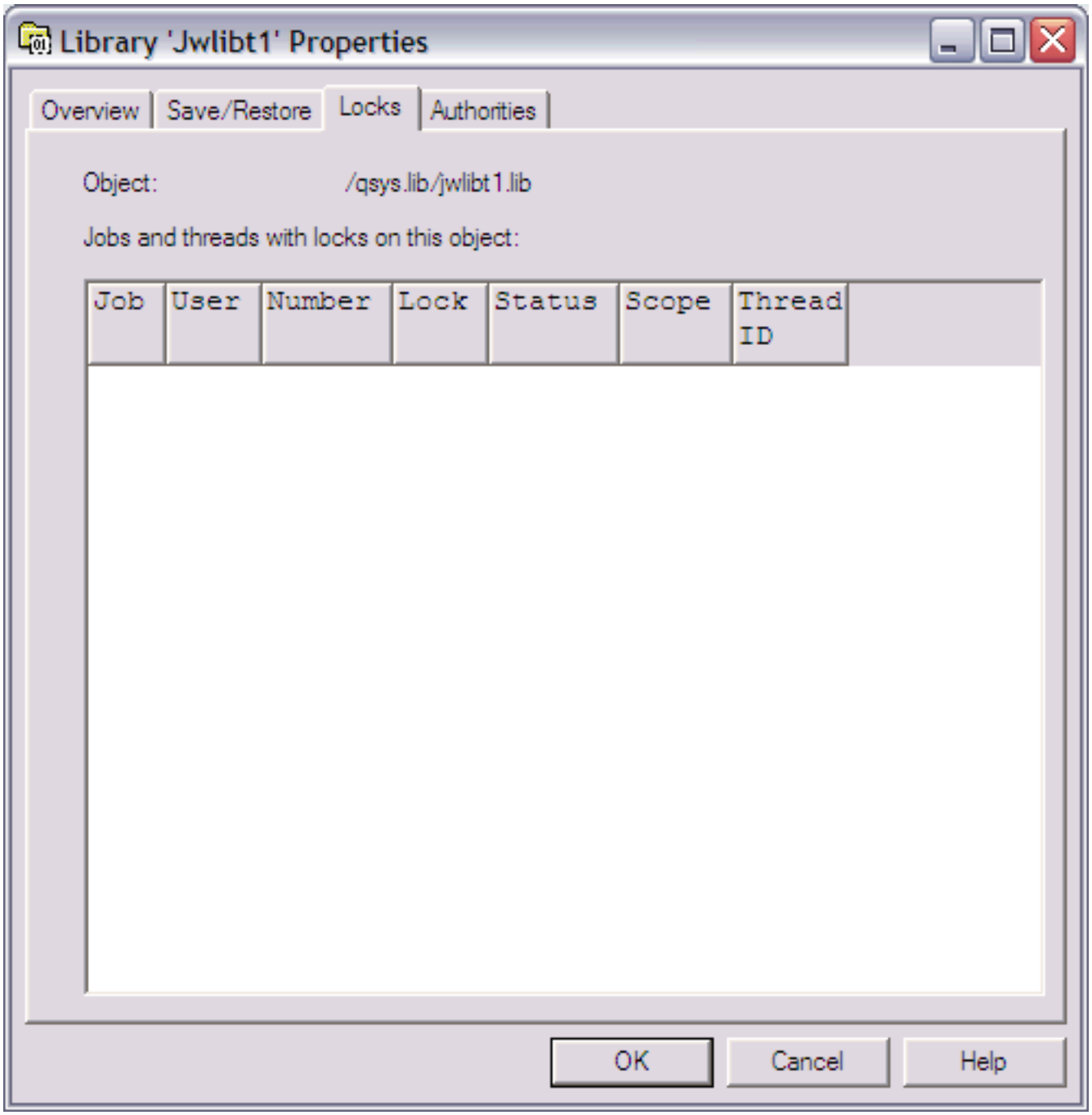
The following information is listed on this page:

Field	Field Description
Saved On	The date and time the library was last saved.
Restored On	The date and time the library was last restored.
Device Type	The type of the device to which the library was last saved. Valid values are: **Blank** The library was not saved. **Diskette** The library was saved to diskette. **Optical** The library was saved to optical. **Save file** The library was saved to a save file. **Tape** The library was saved to tape.
Save Command	The command used to save the object.
Label	The file label used when the object was saved. This value is not shown if the library was not saved to tape, diskette, or optical. The value of this field corresponds to the value specified for the LABEL or OPTFILE parameter on the command used to save the object.
Save File	Displays the library and name of the save file.
Volume ID	The tape, diskette, or optical volumes that are used for saving the library.
Save Size	Displays the size of the save file.
Save Sequence Number	The tape sequence number assigned when the library was saved on tape. If the library was not saved to tape, this value is not displayed.

Table of Contents | Previous | Next

1.3.7.3 Locks

The locks property page for libraries provides an interface similar to the Work Object Lock (WRKOBJLCK) command. This page will tell you which jobs (if any) have a lock on the library.



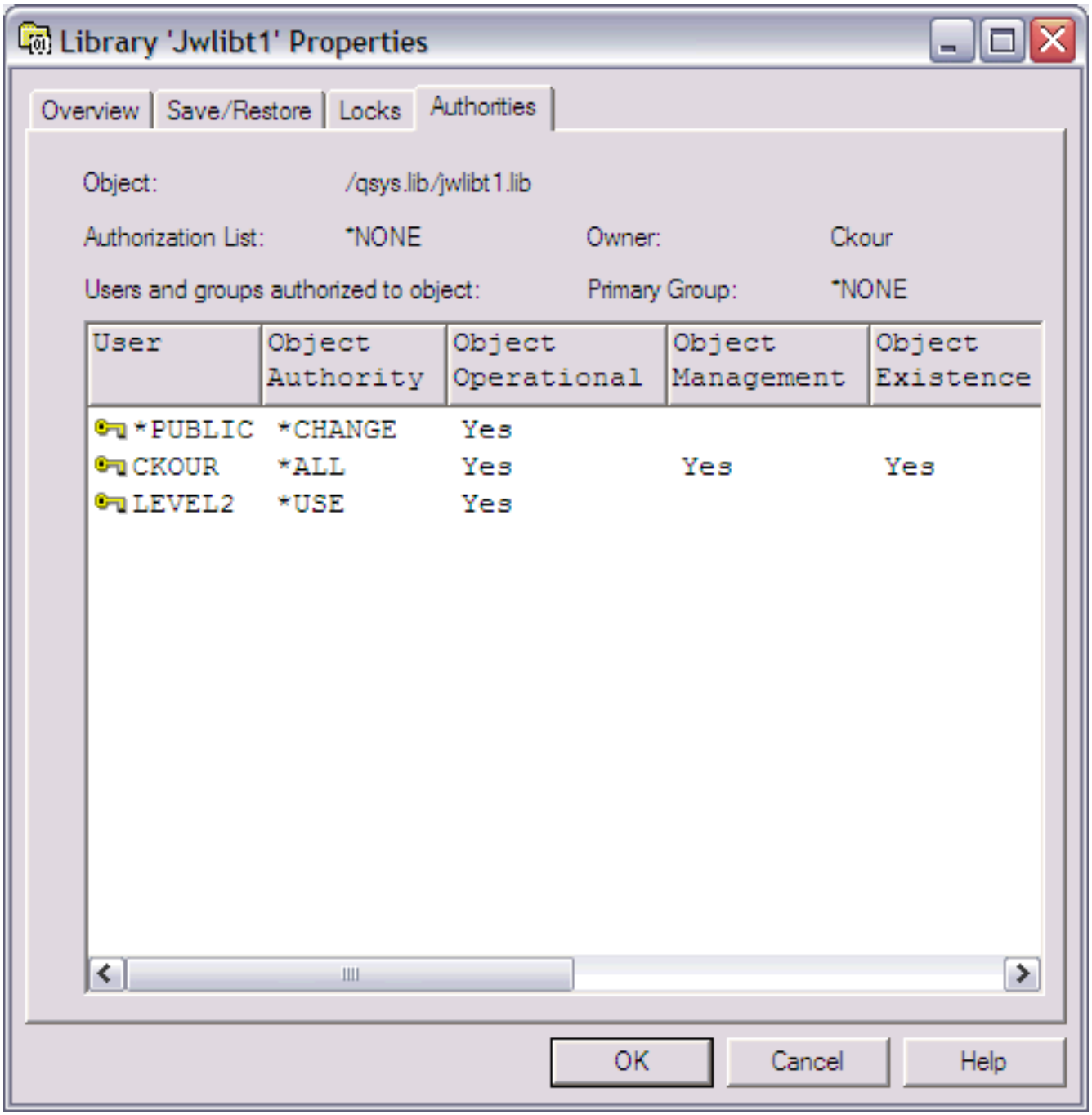
The following information is shown for each job in the list.

Field	Field Description
Job	The simple job name of the job that issued the lock request.
User	The user name of the job that issued the lock request.
Number	The number of the job that issued the lock request.
Lock	The lock condition for the request. The possible values are: ***SHRRD** Lock shared for read. ***SHRUPD** Lock shared for update. ***SHRNUP** Lock shared no update. ***EXCLRD** Lock exclusive allow read. ***EXCL** Lock exclusive no read. ***NONE** Lock entry has a null value and is used to select display of lower-level locks.
Status	The status of the lock. The possible values are: **HELD** The lock is currently held by the job. **WAIT** The job is waiting for the lock. **REQ** The job has a lock request outstanding for the object.
Scope	Specifies whether the lock is scoped to the job or scoped to the thread.
Thread ID	Specifies the thread that is associated with the lock. If a held lock is job scoped, the field is blank. If a held lock is thread scoped, the field contains the identifier for the thread holding the lock. If the lock is requested, but not yet available, this field contains the identifier of the thread requesting the lock.

Table of Contents | Previous | Next

1.3.7.4 Authorities

The Authorities property page shows a list of users that have authority to the library and the users' authorities. This interface is similar to the DSPOBJAUT command.



The following information is shown on this page:

Field	Field Description
Object	The name of the object for which information is being displayed.
Authorization List	The name of the authorization list that is used to secure the named object. The value, *NONE, indicates that no authorization list is used in determining authority to the object.
Owner	The name of the user profile which owns the library.
Primary Group	The name of the user profile that is the primary group for the library. The primary group can be changed using the Change Object Primary Group (CHGOBJPGP) command.
User List	Displays each user authorized to the library and their detailed authorities to it.

The following information is shown for each user in the list.

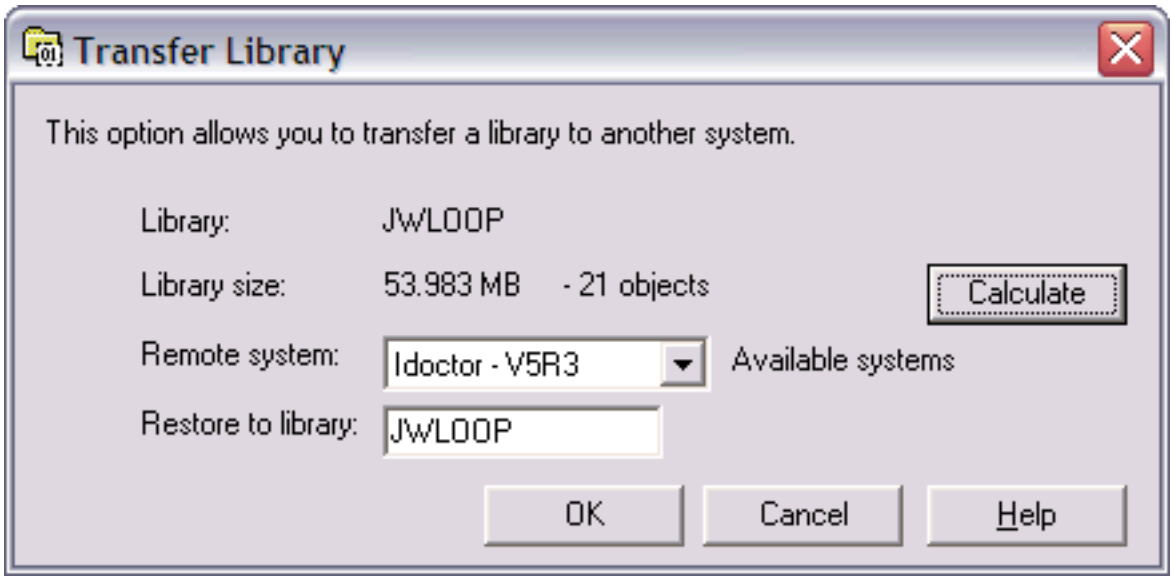
Field	Field Description
User	The names of users who are authorized to use the object. The value *PUBLIC is used to indicate the authorities of users who are not specifically named and are not in the object's authorization list.
Group	A group from which the user receives authority.
Obj Authority	The user's authority to the object. This field contains one of the following values: ***ALL** The user has all object (operational, management, existence, alter, and reference) and data (read, add, update, delete, and execute) authorities to the object. ***CHANGE** The user has object operational and all data authorities to the object. ***USE** The user has object operational and data read and execute authorities to the object. ***EXCLUDE** The user has none of the object or data authorities to the object, or authorization list management authority to the authorization list. ***AUTL** The public authority for the object comes from the public authority on the authorization list securing the object. This value can only be returned if there is an authorization list securing the object and the authorized user is *PUBLIC.

	USER DEF The user has some combination of object and data authorities that do not relate to a special value. The individual authorities for the user should be checked to determine what authority the user has to the object.
Obj Opr	Object operational authority provides authority to look at the object's attributes and to use the object as specified by the data authorities that the user has to the object.
Obj Mgmt	Object management authority provides authority to specify security, to move or rename the object, and to add members if the object is a database file.
Obj Exist	Object existence authority provides authority to control the object's existence and ownership.
Obj Alter	Object alter authority provides authority to change the attributes of an object, such as adding or removing triggers for a database file.
Obj Ref	Object reference authority provides authority to specify the object as the first level in a referential constraint.
Data Read	Read authority provides authority to access the contents of the object.
Data Add	Add authority provides authority to add entries to the object.
Data Update	Update authority provides authority to change the content of existing entries in the object.
Data Delete	Delete authority provides authority to remove entries from the object.
Data Execute	Execute authority provides authority to run a program or search a library or directory.

Table of Contents | Previous | Next

1.3.8 Transfer to

This option allows a user to create a save file of the current library and transfer it to another system. After sending the data to the remote system the library is restored to the library name specified.



Field	Description
Library	The name of the library to transfer
Library size	Displays the library size and number of objects in the library if the calculate button is processed. Depending on the number of objects and physical file members this option could take several minutes to complete.
Remote system	The name of the system to send the libary to.
Restore to library	This is the name of the library the saved library should be restored to on the remote system.

1.5 The Data Viewer

The Data Viewer is a window within iDoctor for iSeries used for displaying tables and graphs over data on an iSeries. You can have as many Data Viewer windows open at one time as you want. The data behind the views within a Data Viewer may come from any number of systems desired.

These views are maniuplated in essentially the same way as within the Main Window (they can be tiled, cascaded, etc). Tables and graphs are typically (but not always) opened from the component views in the Main Window. There are also interfaces to open tables and graphs directly from the Data Viewer window instead of going back to the Main Window.

The data behind a table or graph in iDoctor for iSeries is produced by running an SQL query over one or more database files. This query is supplied by iDoctor for iSeries whenever working with analyses within Job Watcher or PEX Analyzer. The query itself can be viewed and manipulated in the query definition interface. Queries can be saved and restored for future use.

In addition to database files the Data Viewer also has support for viewing spool files containing information such as job logs.

An example of a graph from Job Watcher is shown below.

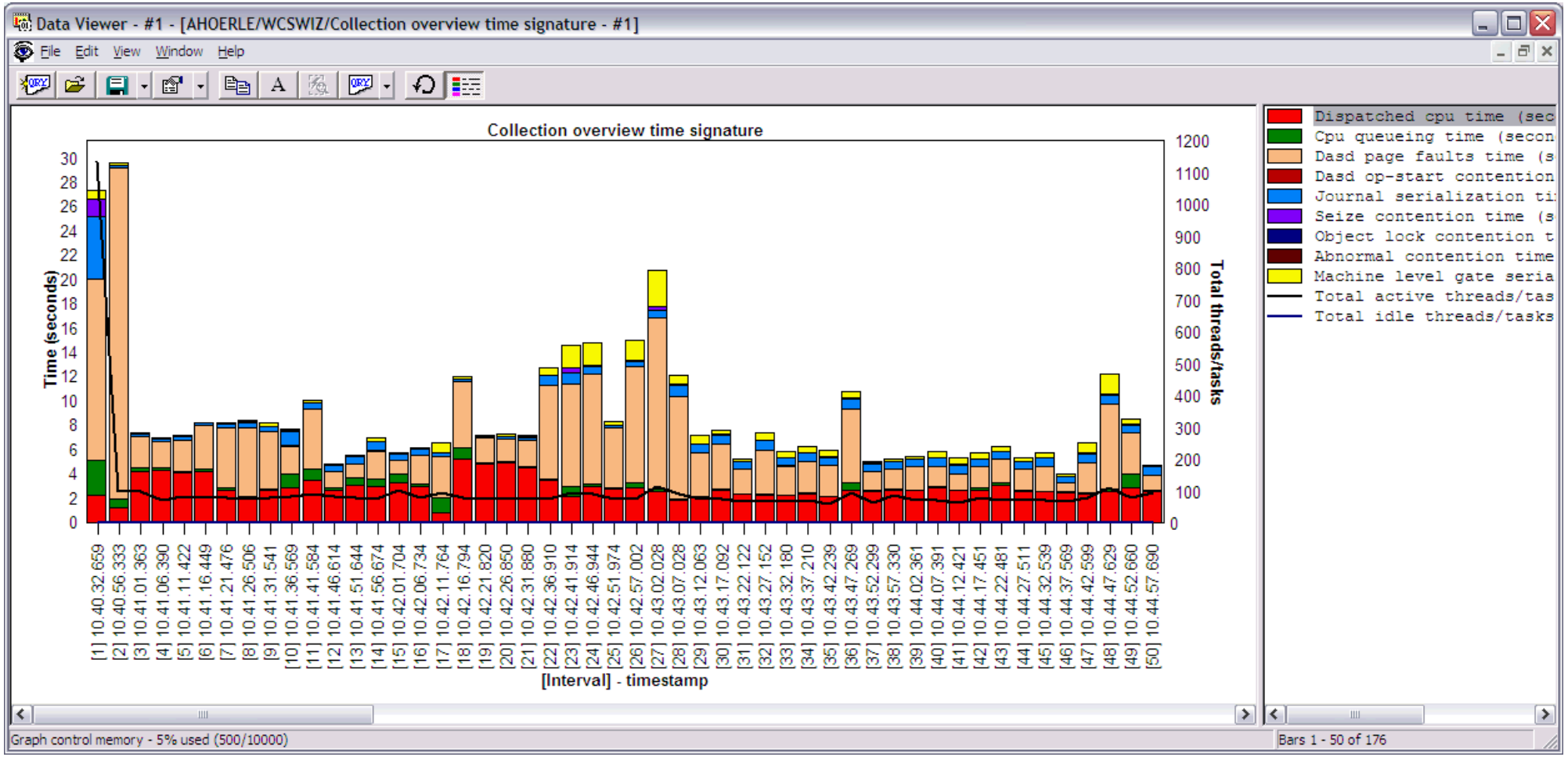
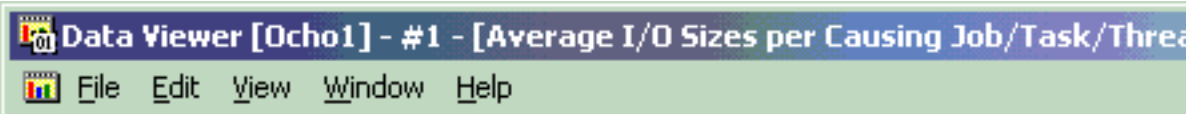


Table of Contents | Previous | Next

1.5.1 Menu Options

This section discusses the menu options available within the iDoctor for iSeries Data Viewer. This only covers the menus available at the top of the Data Viewer window and does not cover the popup menus available within views displayed inside the Data Viewer.



[The Data Viewer Menus]

The table below outlines the different types of operations that may be performed within the Data Viewer.

File Menu	Description
New SQL Query	Opens a new instance of an SQL Query View. The SQL Query View is used to create a query using Structured Query Language (SQL). The top portion of the view is an area where you can enter an SQL statement and the bottom portion is the result or output from the statement above. You can either edit the statement directly or use the query definition interface to change it. Any changes you make via query definition will be immediately visible in the top portion of this view.
Open File/Member	This option allows you to open any library/file/member on the system using the Open File Window. This window lets you browse for the physical or logical file you wish to open. If you do not specify the member, you will be prompted to select the member from a list if the file is a multiple member file.
Save | View As...	This option allows you to save the contents of a table or graph view to a file. When using this option the entire contents of the table or graph view are saved. When saving a table, you can choose between rich text, comma separated and tab separated text formats. When saving a graph view, the file created will be a jpeg image of the current visible page of the graph.

Save | Selection As...	The option allows you to save the **selected** contents of a table to a file. When using this option only the selected records or block of cells are written to the file. When using this option you can choose between rich text, comma separated and tab separated text formats. This option is not available for graph views.
Save | Query Definition...	This option allows you to save the current table's query (SQL) to a file on the server. This feature allows you to save iDoctor for iSeries queries so they can be used later A window will be displayed asking for a description of the query and in which component it should be saved.
Save | Graph Definition...	Allows the graph definition behind the current graph view selected to be saved. Graph definitions are saved into the user-defined graphs folder under collections and can be reused. A window will be displayed asking for a description of the query and in which component it should be saved.
Close	This menu will close the active view in the Data Viewer.
Print	This menu allows you to print the active graph view. This option is only available for graph views.
Close Data Viewer	Use the menu to immediately close the current data viewer and all open views and windows that are open within it.

Edit Menu	**Description**
Copy	Copies the current selection from the active table or graph view to the clipboard. If a table view is active, this is only enabled when something in the table has been selected. For graph views this will copy an image of the current graph to the clipboard.
Find...	This menu allows you to reposition the current record position in a table view, based on some input you supply. The Find Dialog will be displayed and you can use it search through a column for a specific value.
Set Font...	This menu displays a window allowing you to set the font used for the table views in the iDoctor for iSeries application. This option does not apply to the graph views. The font sizes used in the graph views are controlled in the Preferences window.
Preferences...	This menu displays a window allowing you to set user preferences for the iDoctor for iSeries application.

View Menu	**Description**

Record Quick View	This menu will display horizontally the currently selected record in the active table view if one is available. This can be very useful to see all the details for a specific record of data without having to scroll as much.
Toolbar	This menu will either show or hide the toolbar. If the toolbar is already visible then there will be a checkmark next to the menu.
Status Bar	This menu will either show or hide the status bar. If the status bar is already visible then there will be a checkmark next to the menu.
Field Descriptions	Use this menu to toggle the display of fields in table views from short names to long descriptions and vice versa. If a checkbox is next to this menu then long field descriptions are displayed, otherwise short field names are displayed. Changing this setting will effect all future table views opened with iDoctor for iSeries.
Auto-refresh real-time data	Use this menu to toggle the setting for auto-refreshing real-time views (like those in Job Watcher). A checkmark next to this menu means that this feature is turned on.
Always show new real time data	Use this menu to toggle the setting for always showing real-time data in real-time views (like those in Job Watcher). A checkmark next to this menu means that this feature is turned on. If this option is turned on the newest data will always be scrolled into view whenever possible. If the sort order changes from that of the original, this feature won't work properly.
Refresh	Refresh the currently active table or graph view.

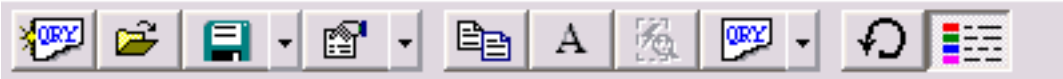
Window Menu	**Description**
Cascade	Use this menu to rearrange all views in the Data Viewer in an overlapping sequence starting in the upper left corner of the window.
Tile Horizontally	Use this menu to rearrange all views in the Data Viewer such that each view will have an equal distribution of the available height in the Data Viewer. The views will not overlap each other.
Tile Vertically	Use this menu to rearrange all views in the Data Viewer such that each view will have an equal distribution of the available width in the Data Viewer. The views will not overlap each other.
Close All	Closes all open views within the Data Viewer. The Data Viewer will remain open.

Help Menu	**Description**
Contents	This menu will launch your web browser and takes you to the online documentation.

iDoctor for iSeries Download Page	Launches your web browser and takes you to the iDoctor download page.
About	This will display version information for the iDoctor for iSeries client.

Table of Contents | Previous | Next

1.5.2 Toolbar



[The Data Viewer Toolbar]

The following table outlines the different toolbar icons available in the Data Viewer and their purpose.

Icon	Description
	Opens a new instance of an SQL Query View. The SQL Query View is used to create a query using Structured Query Language (SQL). The top portion of the view is an area where you can enter an SQL statement and the bottom portion is the result or output from the statement above. You can either edit the statement directly or use the query definition interface to change it. Any changes you make via query definition will of course be immediately visible in the top portion of this view.
	This option allows you to open any library/file/member on the system using the Open File Window. This window lets you browse for the physical or logical file you wish to open. If you do not specify the member, you will be prompted to select the member from a list.
	This option allows you to save the contents of a table or graph view to a file. When using this option the entire contents of the table or graph view are saved. When saving a table, you can choose between rich text, comma separated and tab separated text formats. When saving a graph view, the file created will be a jpeg image of the current visible page of the graph.
	Use this option to view the Query (SQL statement) behind the active graph or table view. From these properties you can also see what file/library/member you are working with. You can copy the SQL to the clipboard and paste it into an SQL Query View to be modified directly if you wish.
	Copies the current selection from the active table or graph view to the clipboard. If a table view is active, this is only enabled when something in the table has been selected. For graph views this will copy an image of the current graph to the clipboard.

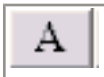
	This menu displays a window allowing you to set the font used for the table views in the iDoctor for iSeries application.
	This menu allows you to reposition the current record position in a table view, based on some input you supply. The Find Dialog will be displayed and you can use it search through a column for a specific value.
	The query definition menu has a large number of submenus each letting you work with the query behind the active table view or in some cases graph views.
	Refresh the currently active table or graph view.
	Hides or shows the graph's legend. If the preference "Always show the legend" is set on the preferences window, this option has no effect.

Table of Contents | Previous | Next

1.5.3 SQL Query View

The SQL Query View lets you dynamically execute and display the results of a query using Structured Query Language (SQL). The top portion of the view is an area where you can enter an SQL statement and the bottom portion is the result or output from the statement above. The bottom portion is a table view and allows the typical copy to clipboard, query definition, export, find functions that are standard in a table view.

The queries you create with this view may be saved and restored for later use and their definitions can be viewed and manipulated using the query definition interface. A benefit of using query definition in concert with the SQL Query View is any changes to the query definition (which changes the SQL statement) will be immediately visible in the top portion of this window.

In order to execute your quey after typing it in, right-click on the top portion of the view and choose the Execute menu or press the F4 key. You can reexecute your query at any time.

An example of this interface is shown below:

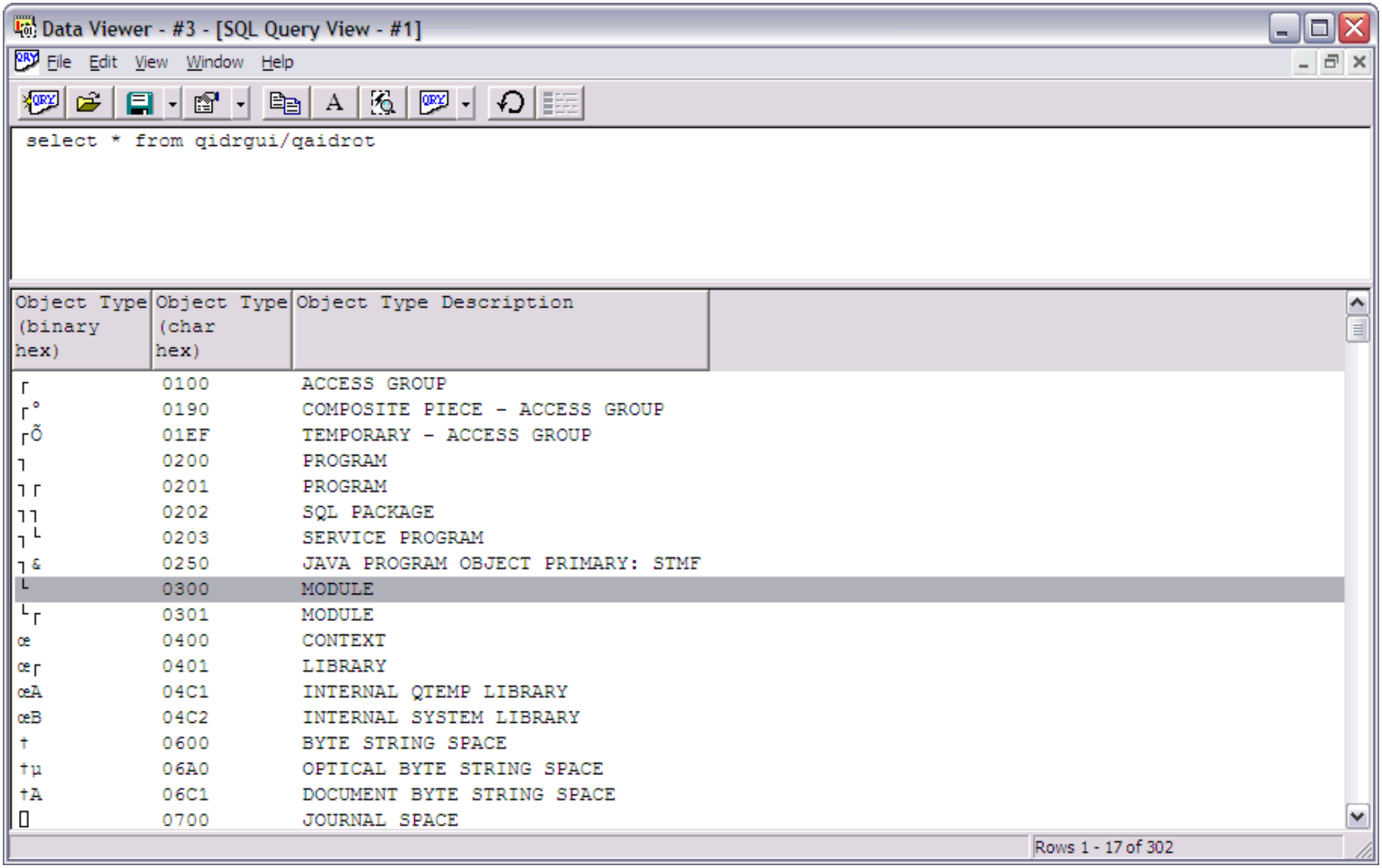
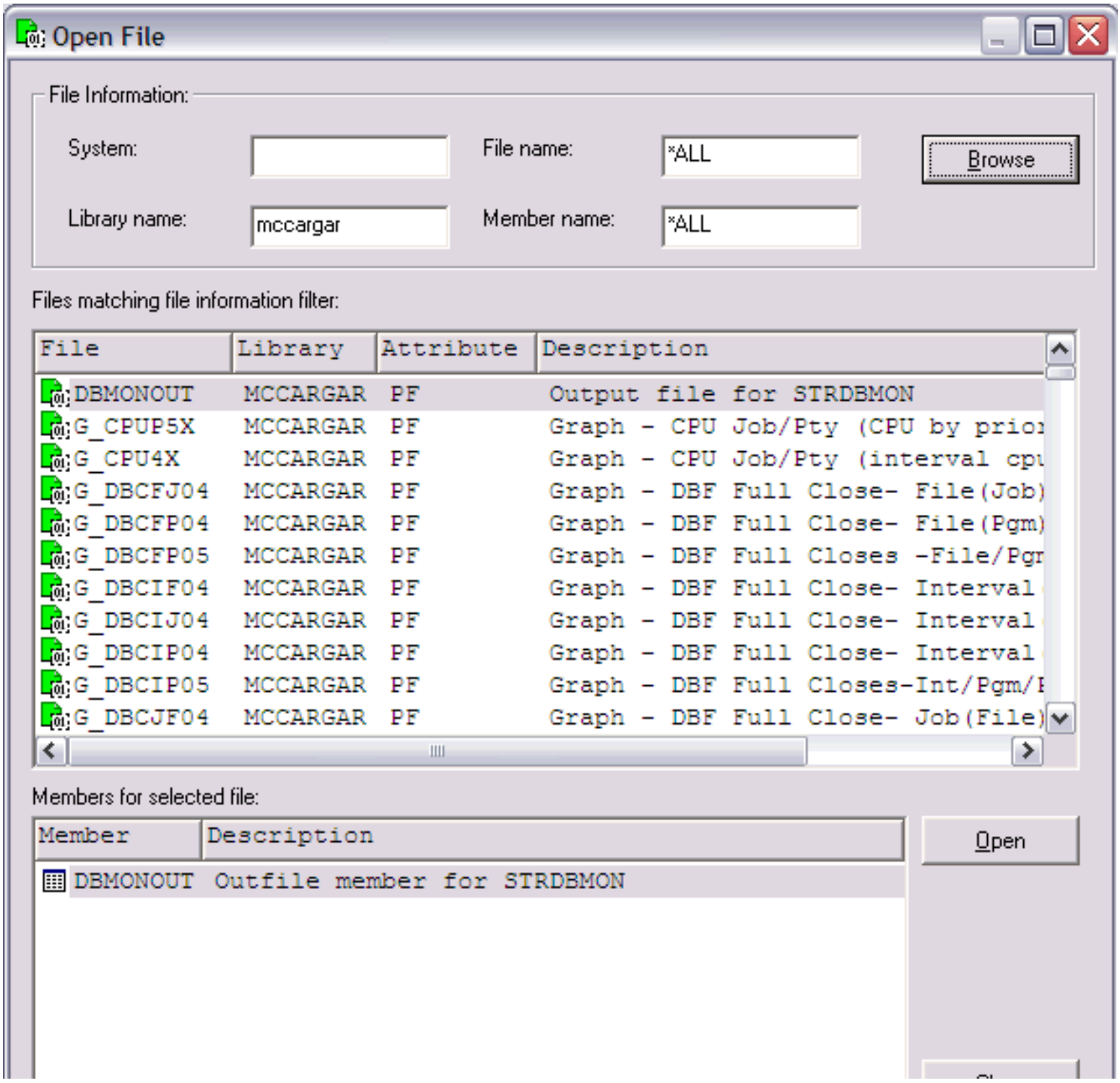

Data Viewer - #3 - [SQL Query View - #1]
File Edit View Window Help
select * from qidrgui/qaidrot
Object Type (binary hex)
Object Type (char hex)
Object Type Description
0100 ACCESS GROUP
0190 COMPOSITE PIECE - ACCESS GROUP
01EF TEMPORARY - ACCESS GROUP
0200 PROGRAM
0201 PROGRAM
0202 SQL PACKAGE
0203 SERVICE PROGRAM
0250 JAVA PROGRAM OBJECT PRIMARY: STMF
0300 MODULE
0301 MODULE
0400 CONTEXT
0401 LIBRARY
04C1 INTERNAL QTEMP LIBRARY
04C2 INTERNAL SYSTEM LIBRARY
0600 BYTE STRING SPACE
06A0 OPTICAL BYTE STRING SPACE
06C1 DOCUMENT BYTE STRING SPACE
0700 JOURNAL SPACE
Rows 1 - 17 of 302

Table of Contents | Previous | Next

1.5.4 Open File Window

This option allows the user to open any library/file/member on the system. A window is displayed where the user can browse for the physical or logical file to open. When the file selection changes, the list of members shown is also updated based on the selection.



The following table describes the interface elements within this window.

Inteface Element	Description
Library name	Type the name of the library to look for files in. Generic library names are not supported for this field. *ALL may also be used for the library name parameter. However if *ALL is used a generic file name should be used otherwise it could take several minutes in order for the list of files to be built.
File name	Type the specific name or generic name of the file to open.
Member name	Type the specific name or generic name of the member you are looking for.
Browse button	This button updates the lists based on the library/file/member information specified.
Files matching file information list	The list of files matching the file library/file specified.
Members for selected file list	The list of members for the selected file in the file list and that match the member name filter.
Open button	Opens the selected library/file/member.

Table of Contents | Previous | Next

1.5.5 Table Views

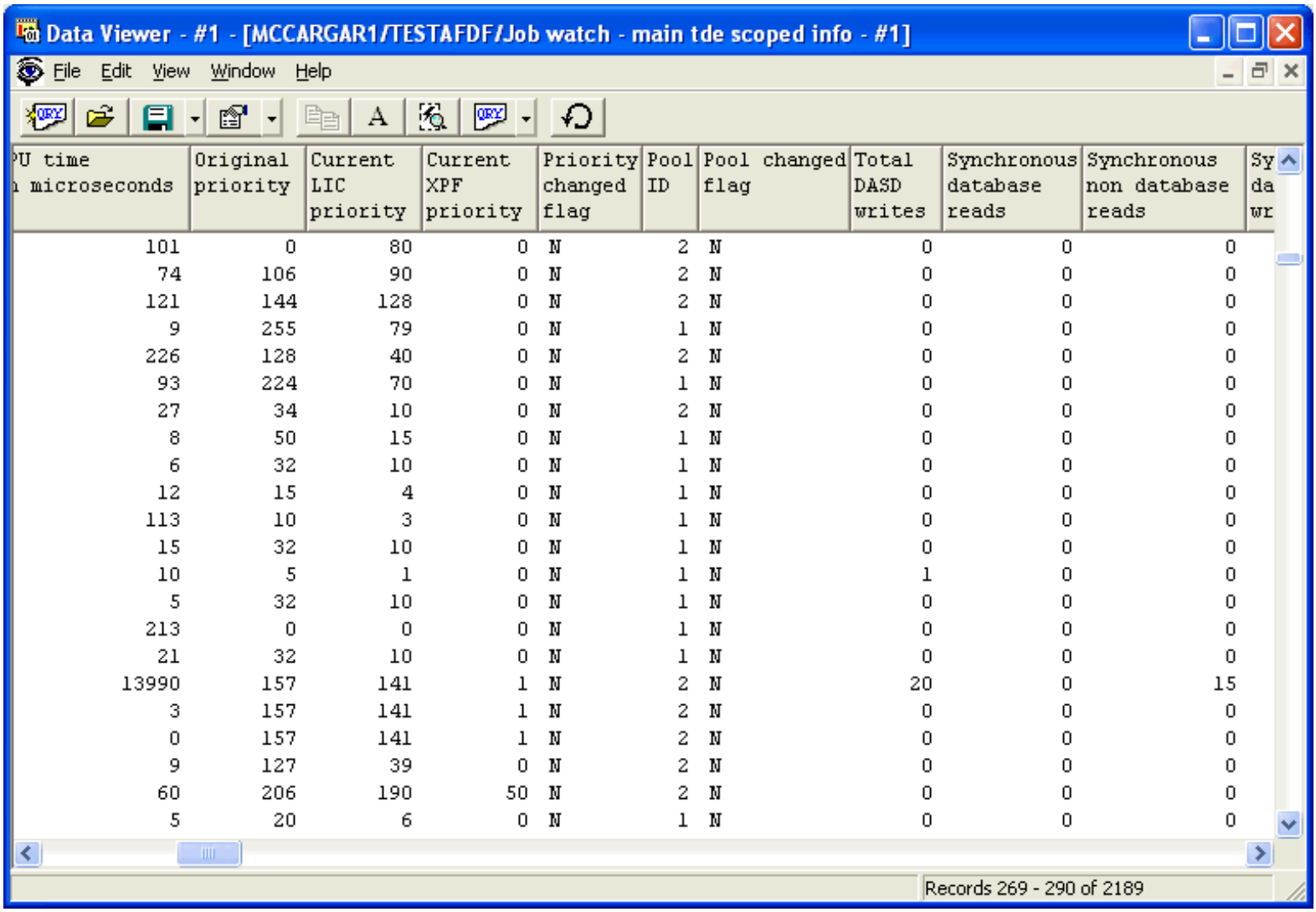
A table view shows data from database files on the iSeries. The user can display millions of records in a table view and use the scroll bar to quickly move to the records desired. For performance reasons, only records actually needed to be displayed are loaded into the client via blocking methods allowing quick relative scrolling.

The SQL behind the table view can be modified at any time using the Query Definition interface. The data may also be sorted by clicking the desired column to sort by. Clicking again on the same column heading will resort the data in the reverse order.

Data in a table view may be selected for copy and paste to a file or to the clipboard. A set of records -or- a block of cells may be selected at any one time. Click the left mouse button and drag across the cells desired in order to make a block selection. Once a selection is made, use the Edit | Copy to copy the current selection to the clipboard. Use the File | Save Selection As... menu to write the selection to a file.

The record indicator in the status bar will show which records are currently being viewed out of the total possible in the active view.

An example of a table view is the following:



PU time microseconds	Original priority	Current LIC priority	Current XPF priority	Priority changed flag	Pool ID	Pool changed flag	Total DASD writes	Synchronous database reads	Synchronous non database reads	Sy da wr
101	0	80	0	N	2	N	0	0	0	
74	106	90	0	N	2	N	0	0	0	
121	144	128	0	N	2	N	0	0	0	
9	255	79	0	N	1	N	0	0	0	
226	128	40	0	N	2	N	0	0	0	
93	224	70	0	N	1	N	0	0	0	
27	34	10	0	N	2	N	0	0	0	
8	50	15	0	N	1	N	0	0	0	
6	32	10	0	N	1	N	0	0	0	
12	15	4	0	N	1	N	0	0	0	
113	10	3	0	N	1	N	0	0	0	
15	32	10	0	N	1	N	0	0	0	
10	5	1	0	N	1	N	1	0	0	
5	32	10	0	N	1	N	0	0	0	
213	0	0	0	N	1	N	0	0	0	
21	32	10	0	N	1	N	0	0	0	
13990	157	141	1	N	2	N	20	0	15	
3	157	141	1	N	2	N	0	0	0	
0	157	141	1	N	2	N	0	0	0	
9	127	39	0	N	2	N	0	0	0	
60	206	190	50	N	2	N	0	0	0	
5	20	6	0	N	1	N	0	0	0	

A popup menu is available by right-clicking anywhere on the table. The following options are available via the popup menu:

Popup Menu	Description
Record Quick View	This option will display in the property pages a vertical view of the current record(s) selected. If multiples are selected this option can be used to show a comparison between two records in a side-by-side view.
Copy	Copies the current text selection to the clipboard.
Find...	This menu allows the user to reposition the current record in a table view, based on input supplied if matching information is found.
Save	Allows the table view, a selection of it or the query definition behind it to be saved.
Set Font...	This menu displays a window allowing customization of the font used for all table views.
Preferences...	This menu displays a window allowing the user to set customized settings for the iDoctor for iSeries application.
Graph Definition	This menu contains an option to create a new user-defined graph from the current report.
Query Definition	This menu has a large number of submenus each letting the user work with the query behind the active table view.
Properties	Displays the properties for the current report. The information shown in the property pages varies based on the type of report being viewed.

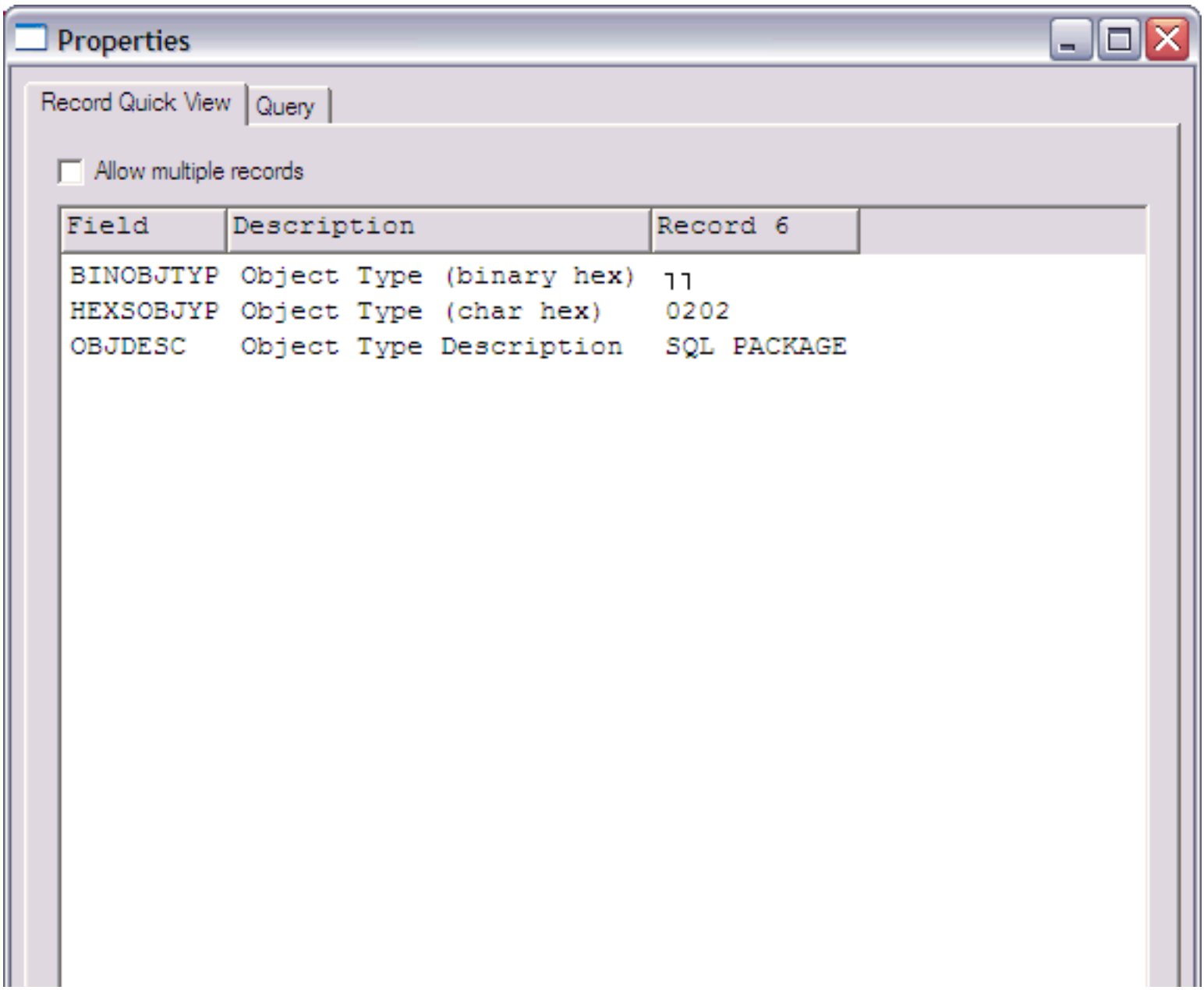
Depending on the type of report shown in a table view, other menu applicable to that report type will be shown. These are mentioned in the Job Watcher and PEX Analyzer documentation for the applicable report types.

Table of Contents | Previous | Next

1.5.5.1 Report Properties - Record Quick View

This window is part of the property pages for a table view. The Record Quick View page shows all of the data for the selected record from the table in a vertical list. This can make it easier to see all the data for a single record if many fields exist in the table. Access this window by double-clicking on any record in a table view or by using the Properties menu.

An example of this window for a PEX Analyzer table view is the following:



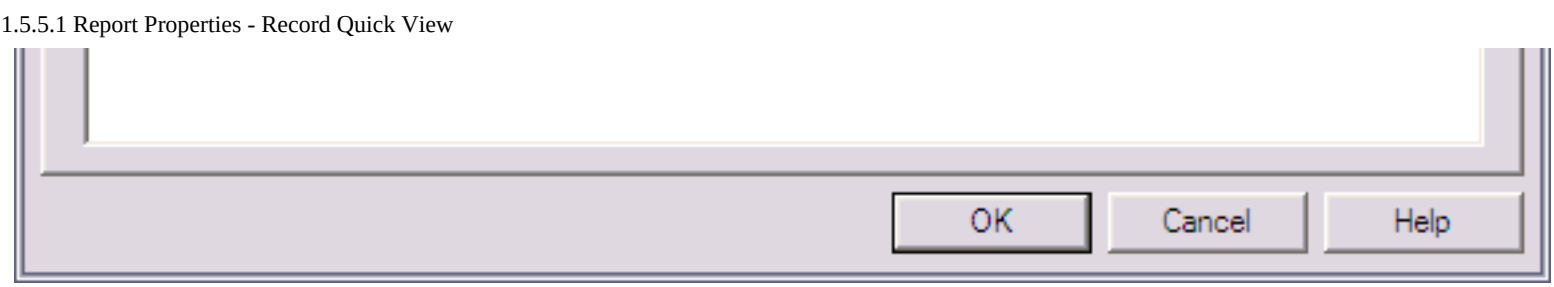
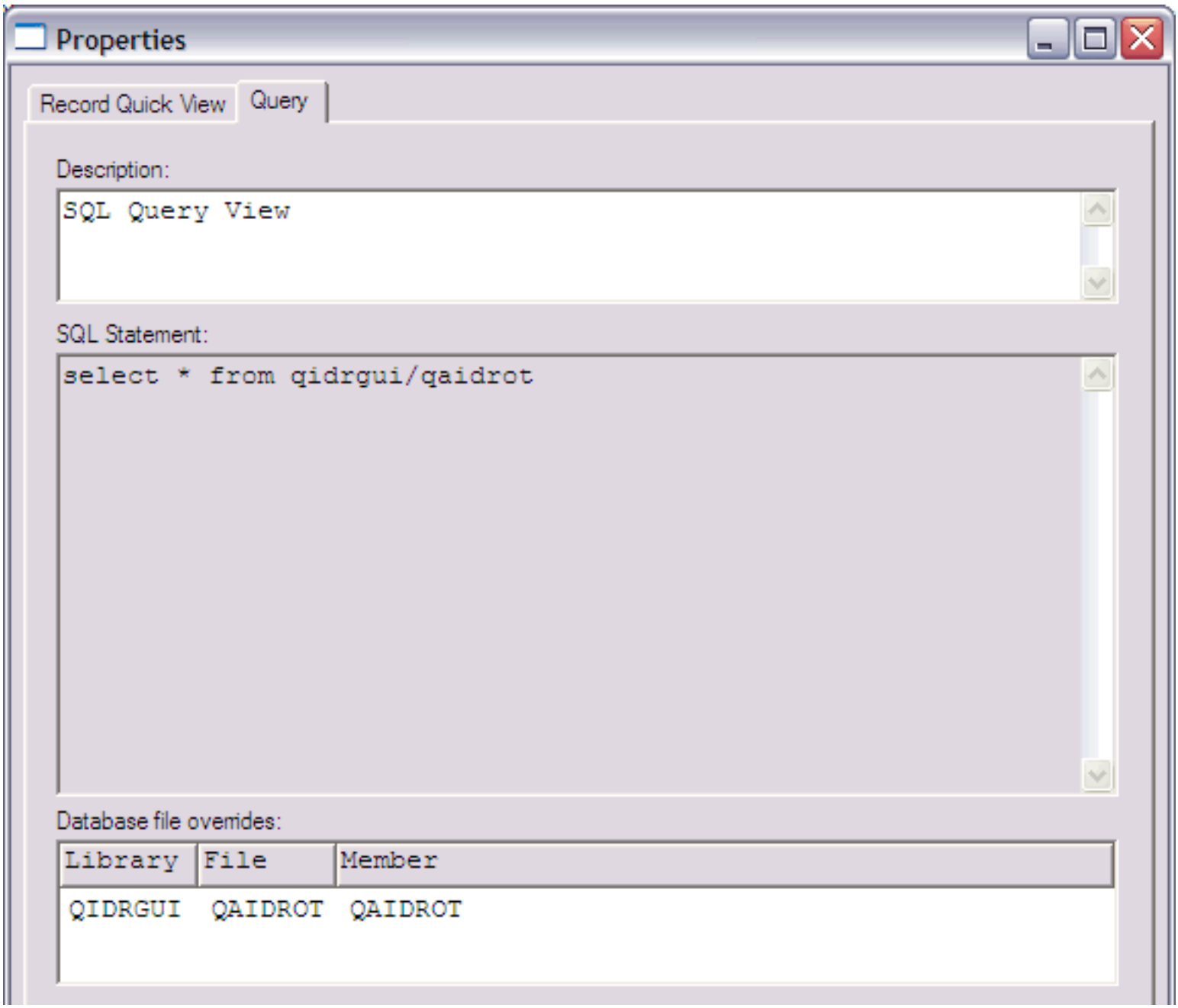

OK
Cancel
Help

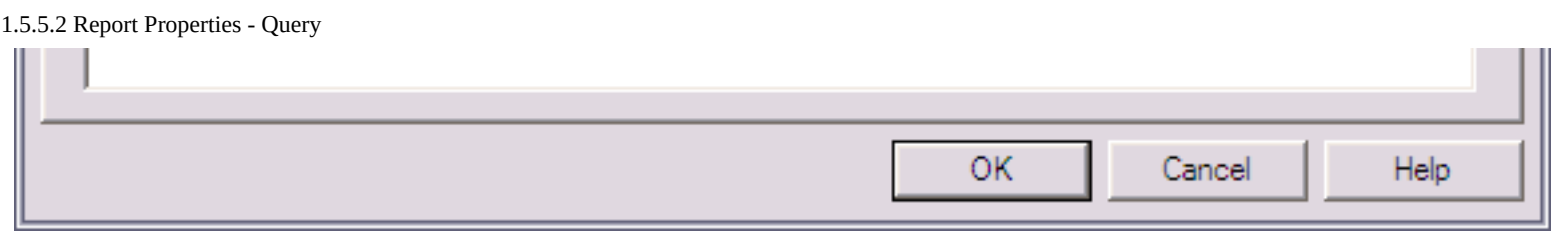
Table of Contents | Previous | Next

1.5.5.2 Report Properties - Query

The Query page of the Report Properties window displays the SQL statement used to produce the current table view. This window also displays the title of the table view and the overrides used to produce the current table view. Because SQL does not support multiple member tables, overrides (see the OVRDBF command) are issued before the query is executed to select which file(s)/member(s) should be used when running the SQL statements.

An example of this page is the following:





The interface elements within this window are described in more detail in the table below:

Interface Element	Description
Description	The text description identifying the report.
SQL statement	The complete SQL statement for the query definition. Any changes made to the query using the Query Definition interfaces (field selection, record selection, sort by, etc) will be reflected in this SQL statement.
Database file overrides	This list identifies all of the members used for each file in the query. An override is used to point to a specific member when executing the query since this cannot be indicated by a SQL statement.

Table of Contents Previous Next

1.5.6 Graph Views

The graph views show visual representations of any data queryable on the system. The only types of graphs supported currently are bar graphs. There are 4 different types of bar graphs supported: stacked horizontal, horizontal bar, stacked vertical bar, vertical bar. The vertical bar graph type and horizontal bar graph type shows bars side-by-side instead of stacked for the same data point. Each color in the graph represents a different value or field in the data. Most graphs will have a legend identifying the information shown at the bottom of the window.

Use the scroll bars to navigate through the information in the graphs. Due to the potential to view vast amounts of data at one time, the graph data is shown a page at a time. The number of bars shown per page is configurable through the Preferences interface. When scrolling through the data the scale of the axes can be set to adjust automatically. This is another option on the Preferences interface. If automatic scaling is disabled then the graph scale will be set to the maximum/minimum values of the first page shown in the graph.

Additional information about each piece of data in the graph is avaialble by moving the mouse over the bar of interest. A flyover help window will appear in yellow providing this information. Some of this information is also displayed in the status bar as the mouse moves from bar to bar. The user can also click on any bar to get a complete look at all the information for that particular piece of the graph and any other applicable data that goes with it (interval, job, etc)..

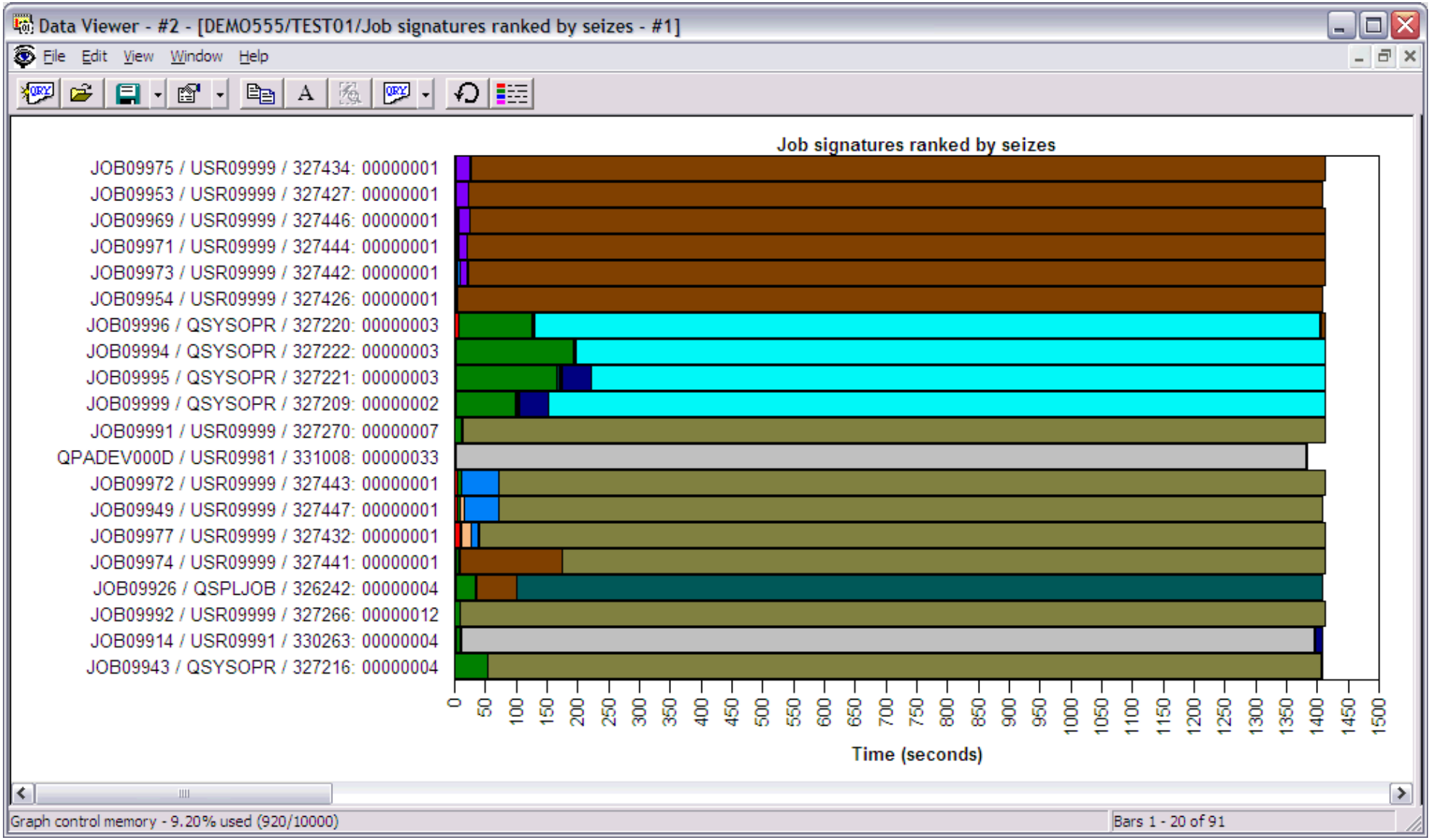
The position indicator in the status bar indicates exactly which bars are being viewed out of the total possible.

There are two types of graphs in iDoctor for iSeries: iDoctor-supplied graphs and user-defined graphs.

iDoctor-supplied graphs

iDoctor-supplied graphs are graphs shipped with iDoctor for iSeries. There are several iDoctor-supplied graphs that come with PEX Analyzer and many more in Job Watcher.

An example of an iDoctor-supplied graph from Job Watcher is:



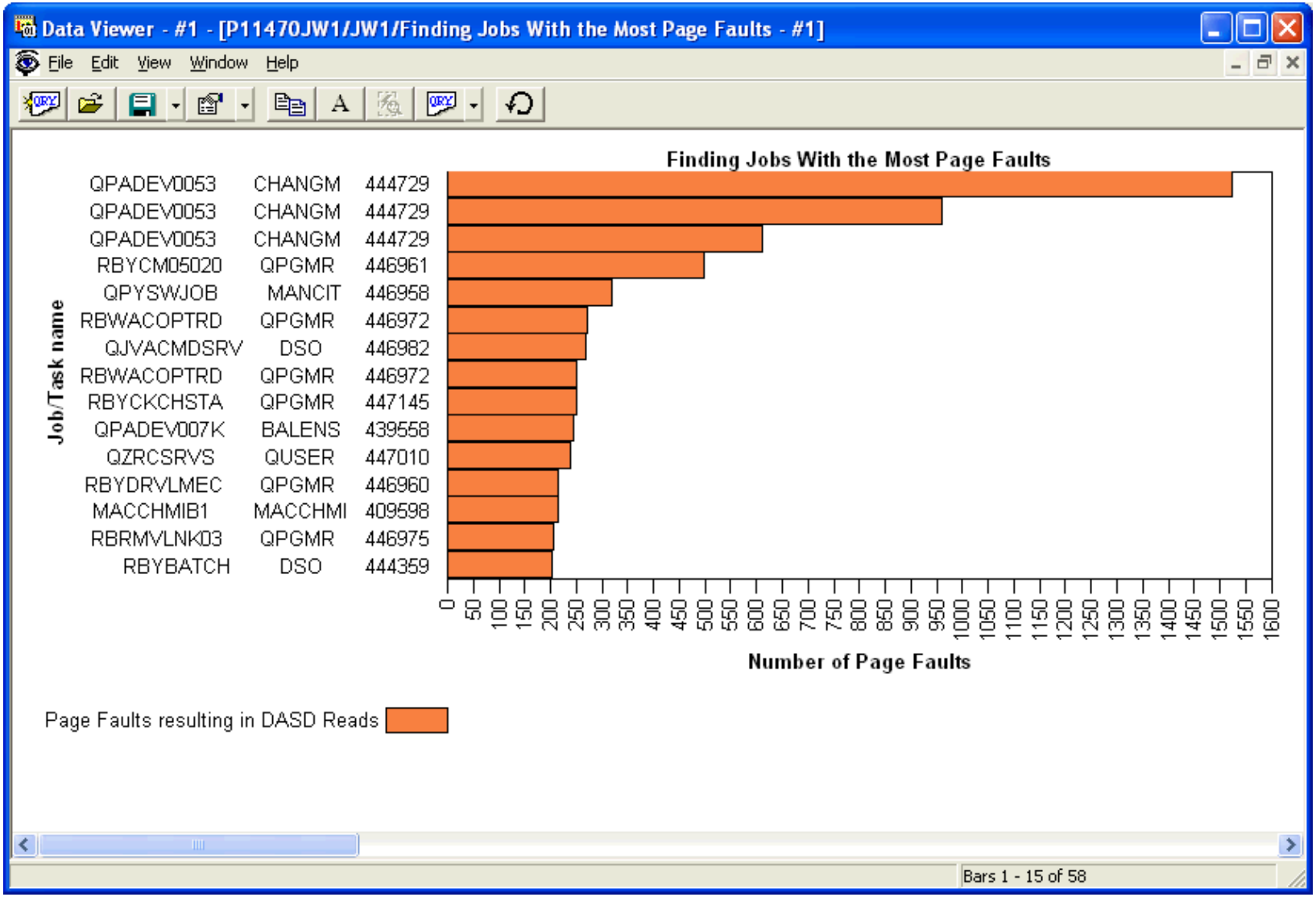
There is a difference between PEX Analyzer graphs and Job Watcher graphs. The Job Watcher graphs allow the user to modify the query definition behind the graph at any time, the PEX Analyzer iDoctor-supplied graphs does not allow this due to the way the PEX Analyzer graphs are constructed.

User-defined graphs

User-defined graphs are created by the user and saved into a definition on the server. The settings for a user-defined graph is called its graph definition.

A user-defined graph is created from a table view. The SQL statement (query) behind the table view is what is used for the query behind the graph. Creating a graph is done using the Graph Definition | Define New... popup menu of a table view.

An example of a user-defined graph over Job Watcher data is shown below:



Popup Menu

Both types of graphs offer the following features via the right-mouse click popup menu:

Popup Menu	Description
Copy	Copy the current graph view as an image (windows bitmap) to the clipboard.
Save As...	Save the current graph view as an image (.jpeg).
Preferences	Displays the preferences window. With this interface the user can change the number of bars shown per page and customize font and other graph settings.
Properties	Displays the properties for the currently selected point on the graph as well as other information such as the SQL statement behind the graph view.

1.5.6 Graph Views

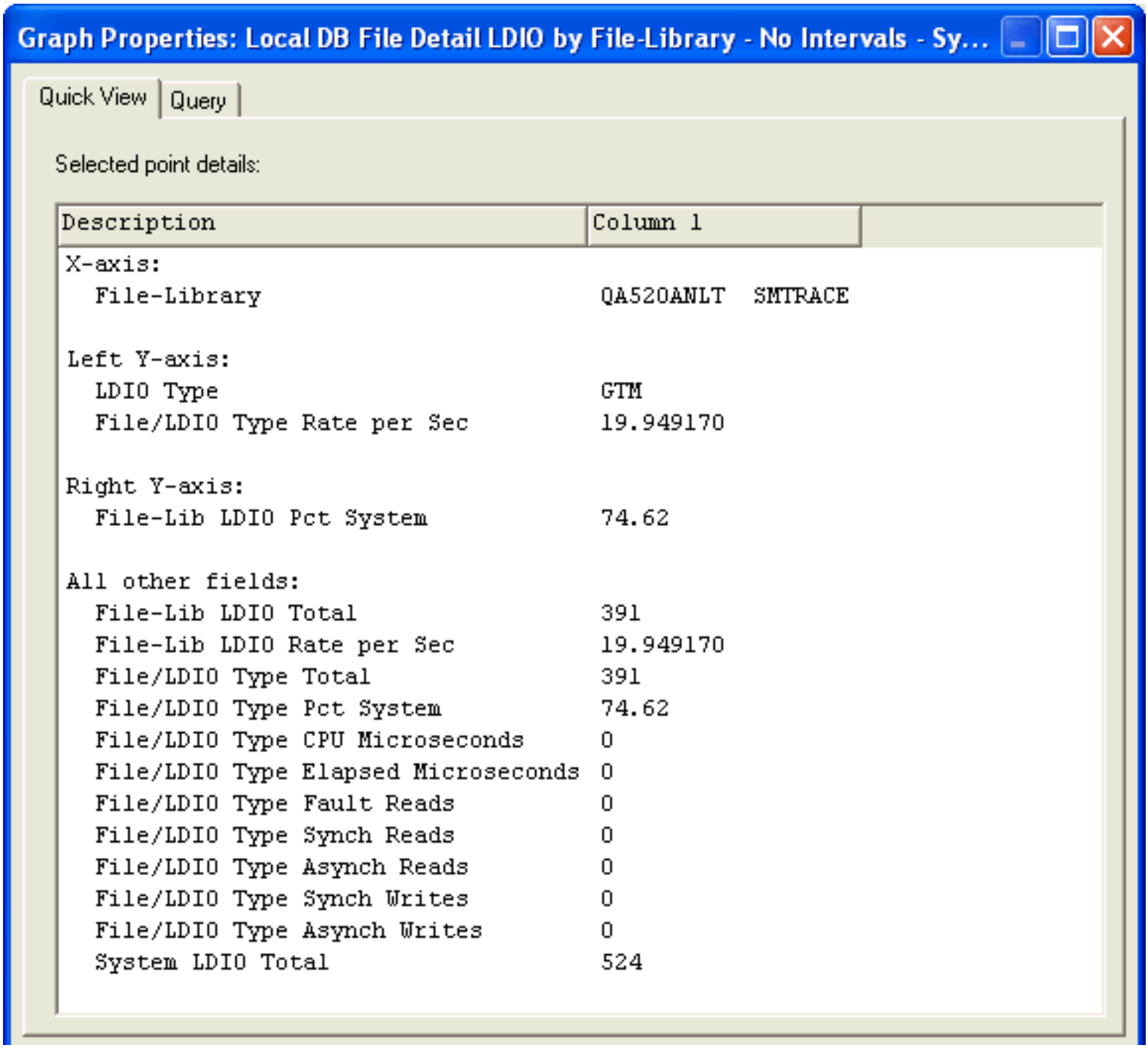
Other popup menu items are shown depending on the type of data/analysis being viewed. These additional options are covered under the documentation for the appropriate analysis type.

Table of Contents | Previous | Next

1.5.6.1 Graph Properties - Quick View

This window is part of the property pages for a graph view. The Quick View page contains all of the information about a particular bar in the graph from the fields available in the query used to created the graph. Access this window by clicking on any bar in a graph view.

An example of this window for a PEX Analyzer graph is the following:



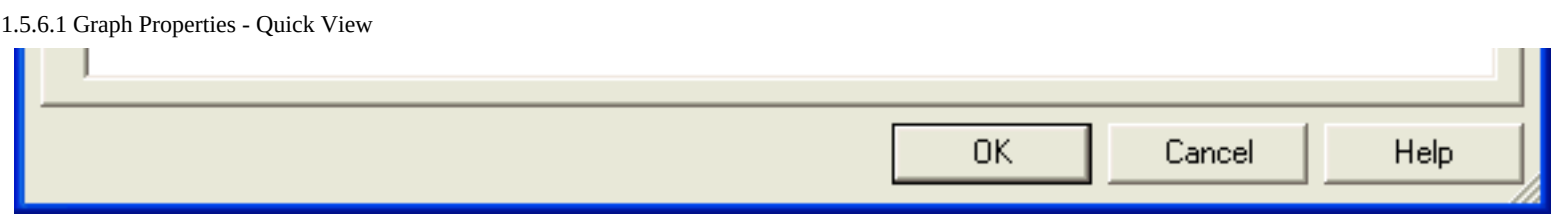
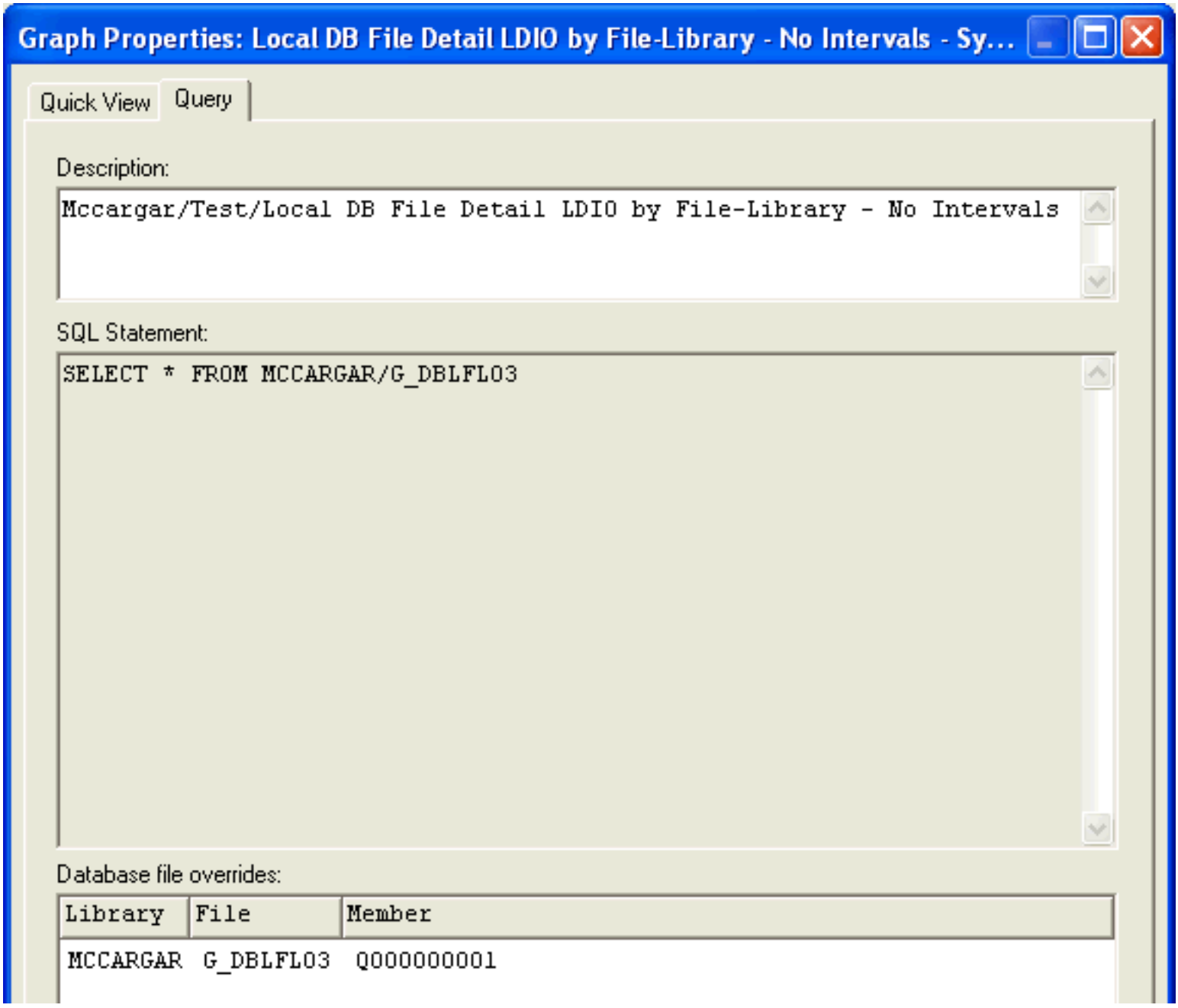

OK
Cancel
Help

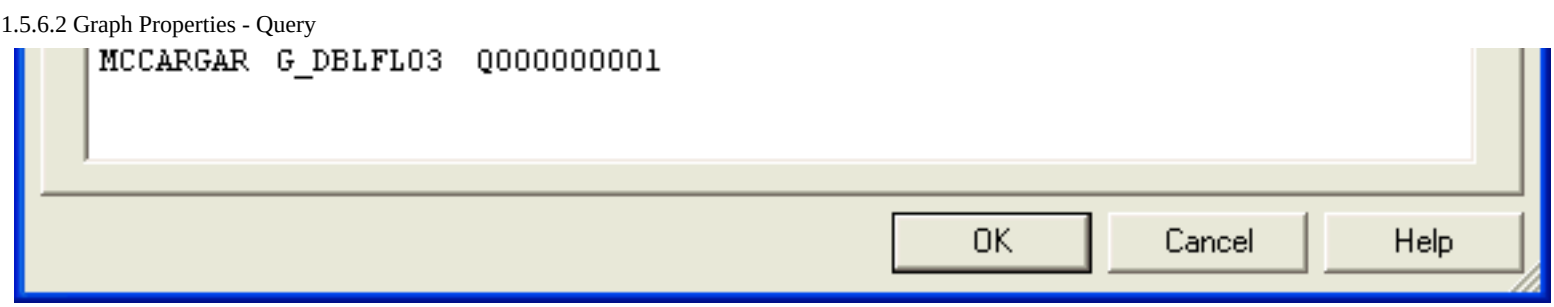
Table of Contents | Previous | Next

1.5.6.2 Graph Properties - Query

The Query page of the Graph Properties window displays the SQL statement used to produce the current graph view. This window also displays the title of the graph view and the overrides used to produce the current view. Because SQL does not support multiple member tables, overrides (see the OVRDBF command) are issued before the query is executed to select which file(s)/member(s) should be used when running the SQL statements.

An example of this page is the following:





The interface elements within this window are described in more detail in the table below:

Interface Element	Description
Description	The text description identifying the report.
SQL statement	The complete SQL statement for the query definition. Any changes made to the query using the Query Definition interfaces (field selection, record selection, sort by, etc) will be reflected in this SQL statement.
Database file overrides	This list identifies all of the members used for each file in the query. An override is used to point to a specific member when executing the query since this cannot be indicated by a SQL statement.

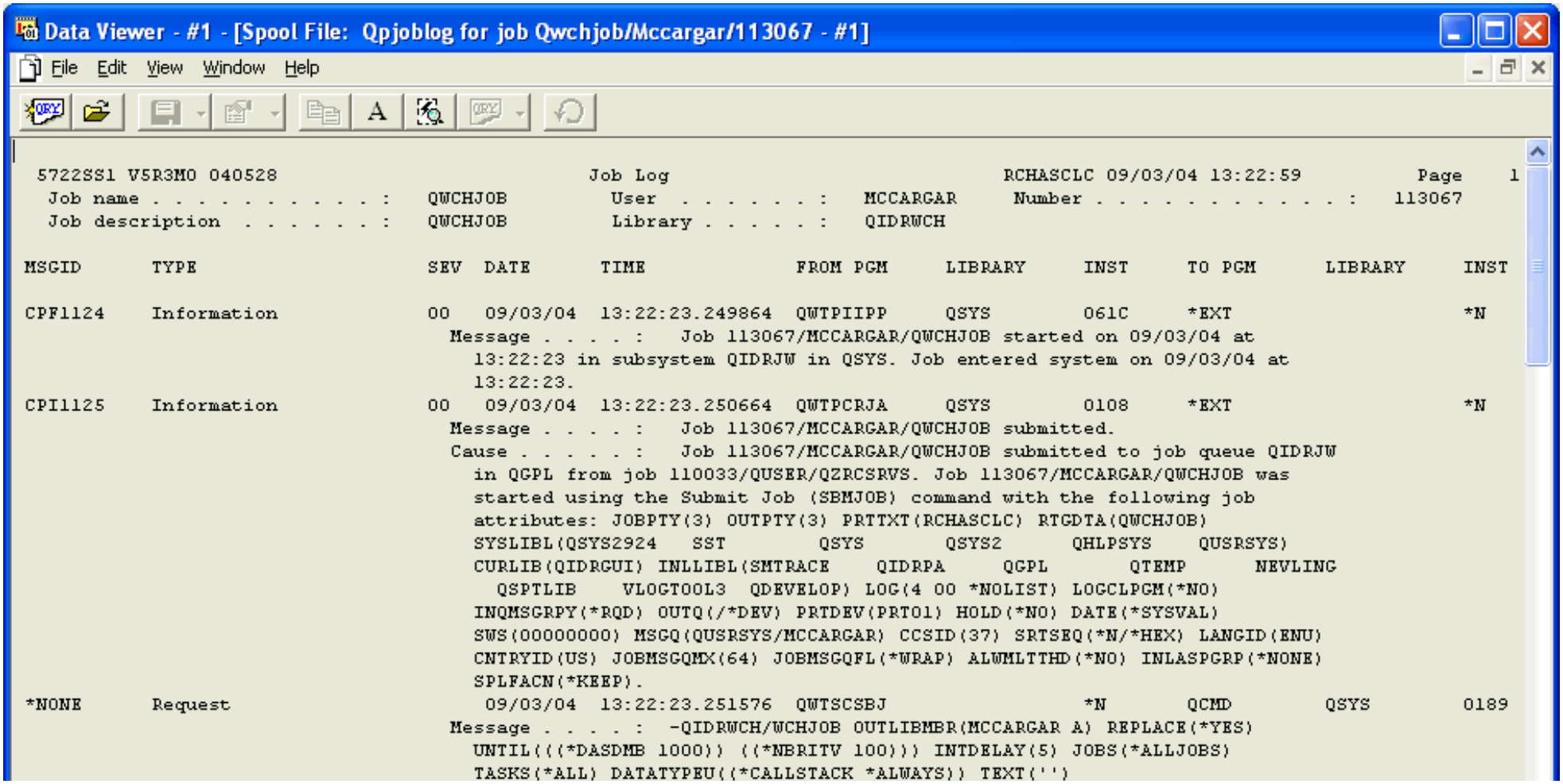
Table of Contents | Previous | Next

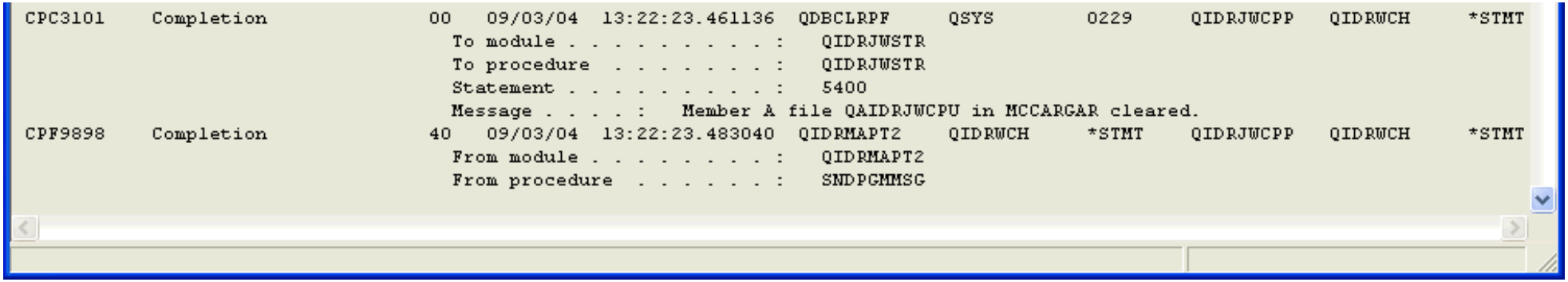
1.5.7 Spool File Views

The Data Viewer can be used to display the contents of spool files on the server. Whenever a job log for a collection is viewed that has already ended the job log is displayed in this viewer.

The spool file viewer will read in the entire contents of the spool file into the viewer. Although this will cause delays when reading large files this allows the user to more quickly perform a text search using the Find feature on the toolbar after the data is loaded into the client.

Other types of spool files besides job logs are displayable. However, they can only be opened using the Object Explorer component. An example of a Spool File View is shown below:





```
CPC3101    Completion              00   09/03/04  13:22:23.461136  QDBCLRPF     QSYS        0229     QIDRJWCPP   QIDRWCH     *STMT
                                     To module . . . . . :   QIDRJWSTR
                                     To procedure  . . . . . :   QIDRJWSTR
                                     Statement . . . . . :   5400
                                     Message . . . . :   Member A file QAIDRJWCPU in MCCARGAR cleared.
CPF9898    Completion              40   09/03/04  13:22:23.483040  QIDRMAPT2    QIDRWCH     *STMT    QIDRJWCPP   QIDRWCH     *STMT
                                     From module . . . . . :   QIDRMAPT2
                                     From procedure  . . . . . :   SNDPGMMSG
```

Table of Contents | Previous | Next

1.6 Query Definitions

Tables and graphs are created via an underlining query definition. The query definition defines exactly how data is to be retrieved and from what file(s). The Query Definition Interface is an interface over an SQL statement. Nearly all table and graph views have a query definition menu available that lets the user work with the query definition for that particular table or graph view. The only exception is the graph views in the PEX Analyzer component. Modifying the query definition for a PEX Analyzer graph is not supported and therefore this interface is not available in that case.

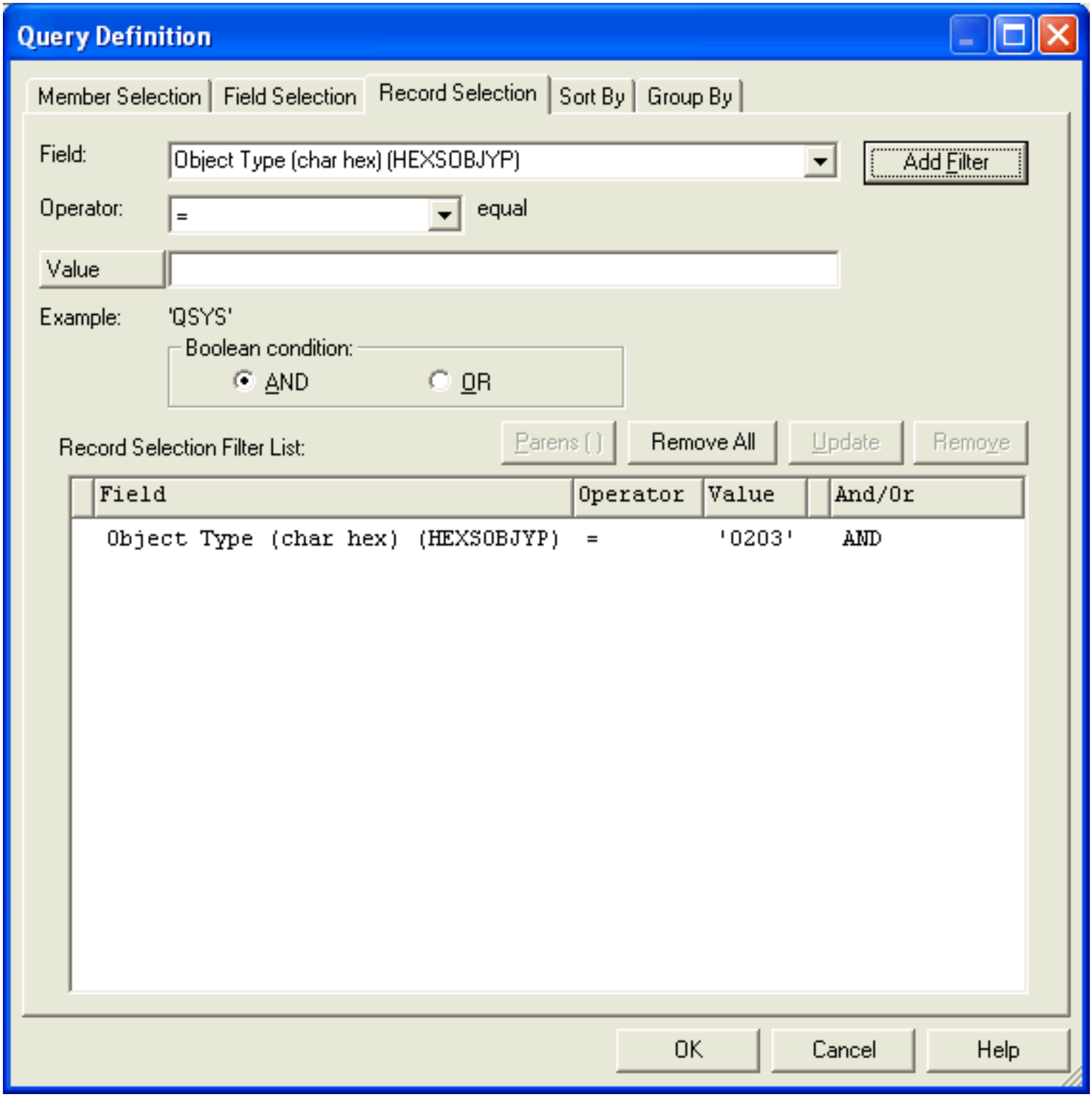
The Query Definition Interface allows a user to customize the query for the active table or graph within the Data Viewer. Right-click on the view and use the Query Definition... popup menu to start the Query Definition Interface. The Query Definition window contains several different tabs each for defining a specific part of the query.

The tabs within the query definition interface are:

Member selection - define which members from the files in the query should be overriden to in order to produce the correct data
Field selection - indicates the order and content of the data in the view
Record selection - used to filter out or include records that meet certain characteristics
Sort by - indicates which field(s) the data should be sorted by
Group by - indicates if the query is a group by query and if it is the group by fields and the having by clause of the SQL statement are definable on this page

In order to use the more advanced features of the Query Definition Interface like the "group by" page, the user needs to have understanding of SQL statement syntax. However, most features like field selection, record selection and sort by have been designed to be understandable by anyone.

An example of the query definition interface is shown below:



Limitations

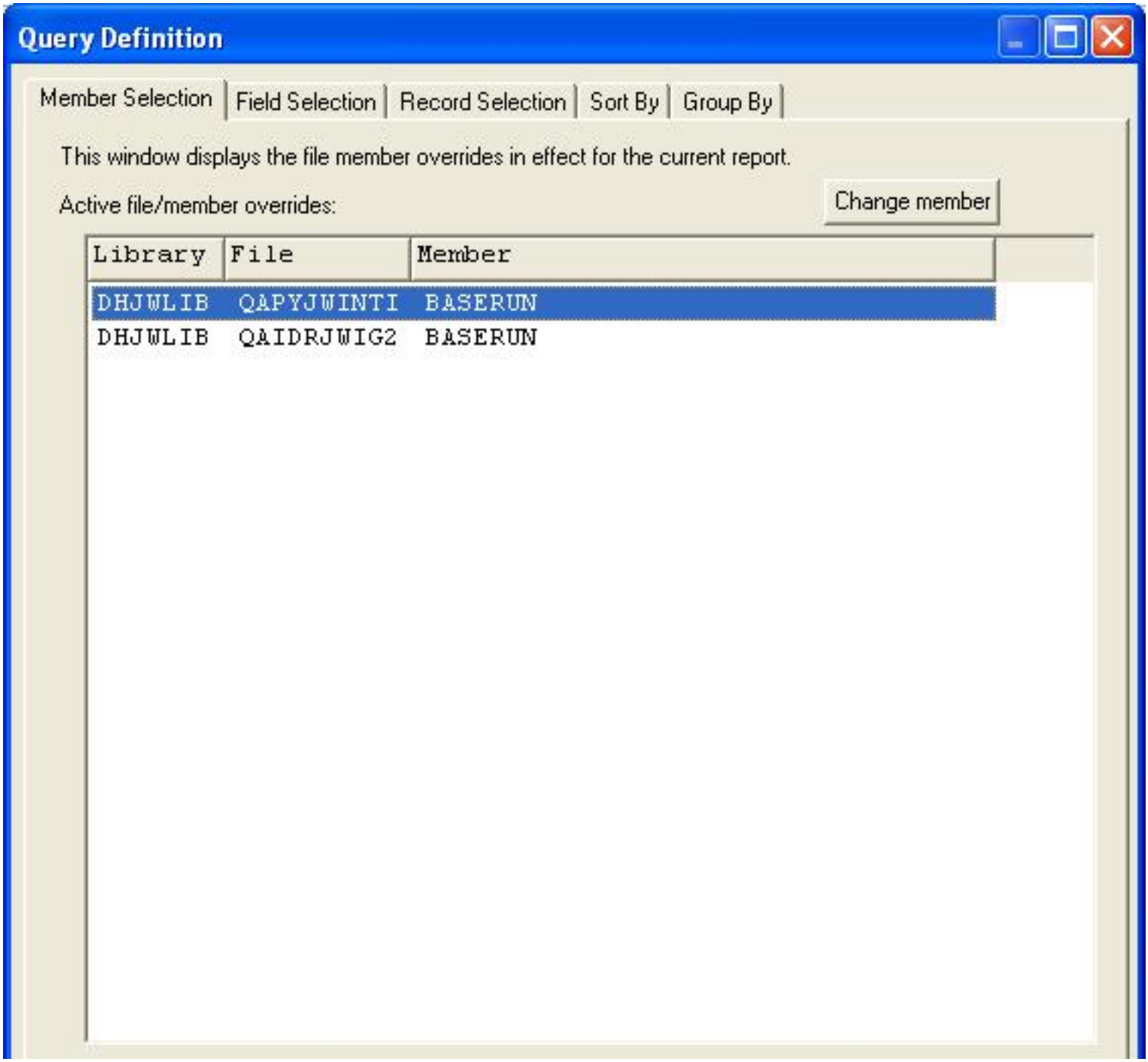
The Query Definition interface is built by parsing the contents of an SQL statement. This parsing works well for many queries but it does not acknowledge all types of SQL syntax. It will parse most SQL select statements containing "joins" but there are some very complex statements that are not parseable (such as UNIONS). Although a query can be parsed that contains joins the types of joins, and the files being joined are not changeable through the interface. In order to do this through the iDoctor for iSeries GUI, the user should use the SQL Query view and type in the SQL statement as needed. The query definition can be used to adjust the where clause, order by and group by clauses of

the outermost part of the SQL statement. Any order by clauses , where clauses, or group by clauses for subqueries within the SQL statement are not configurable through this interface.

Table of Contents | Previous | Next

1.6.1 Member Selection

Use the member selection page to select the members to use when running the query. To change the member used in the query for any file simply click on any member and click the **Change member** button. This will bring up a window allowing the user to change the desired member if the file contains more than 1 member. If there is only 1 member in the selected file, the user will not be prompted.



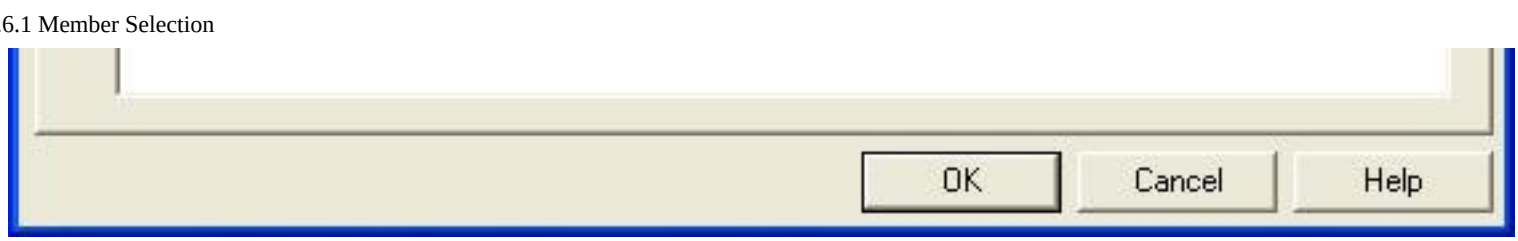

OK
Cancel
Help

Table of Contents | Previous | Next

1.6.2 Field Selection

The field selection panel allows you to hide or reorder the fields in the associated table view. You may also use this panel to create your own fields by typing an SQL expression directly into the SQL expression column in the field list.

The following types of operations can be performed on the field selection page:

- Working with field visibility
- Reordering fields
- Creating new fields

Instructions for performing each of these types of operations follows:

Working with field visibility

Visible fields are indicated by a checkmark in the Show? column within the Field List. If a field is not checked, then it will not be shown.

You may use the Toggle Selected button to check/uncheck the checkbox for the selected fields. This can be very handy when you want to hide or show a large number of fields at once.

Reording fields

The order that the fields are displayed in the Field List, directly effects the order that the fields are displayed in the table view.

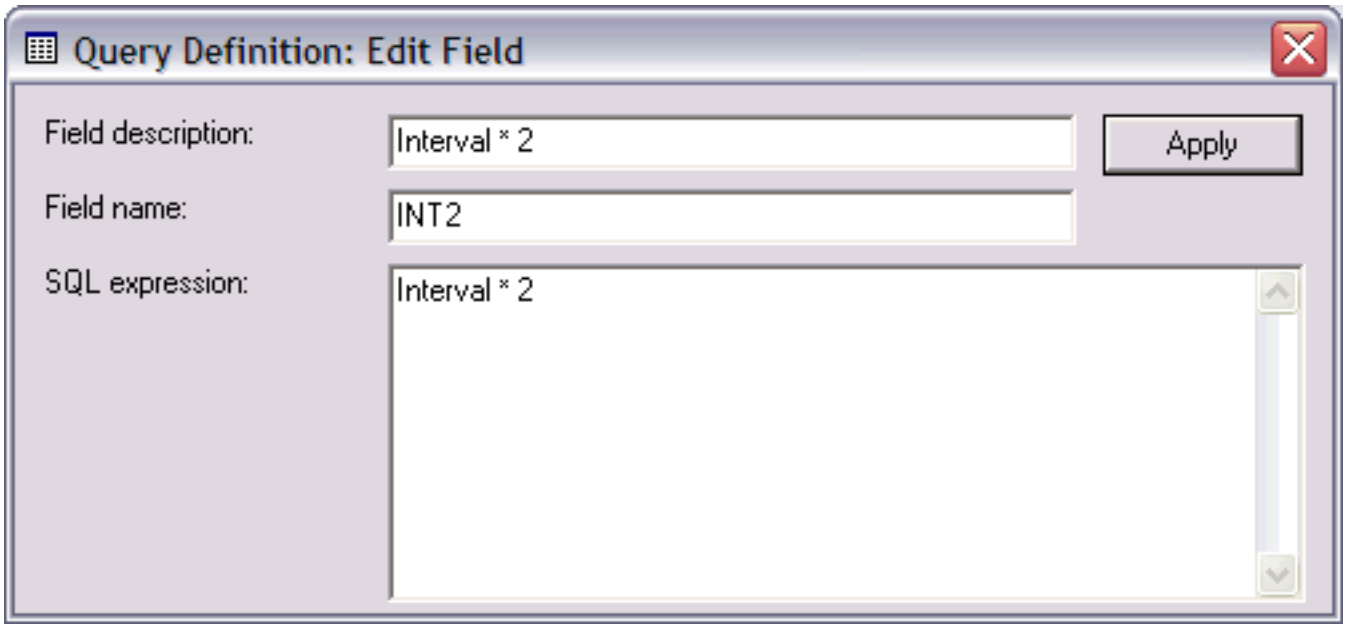
To reorder fields:

1. Select the fields you wish to reorder using the mouse and ctrl/shift keys.
2. Press the left mouse button over one of the selected fields and hold it down.
3. Drag the selection to your desired position in the list. You can scroll through to the bottom of the list if desired.
4. Release the left mouse button.

Creating new fields

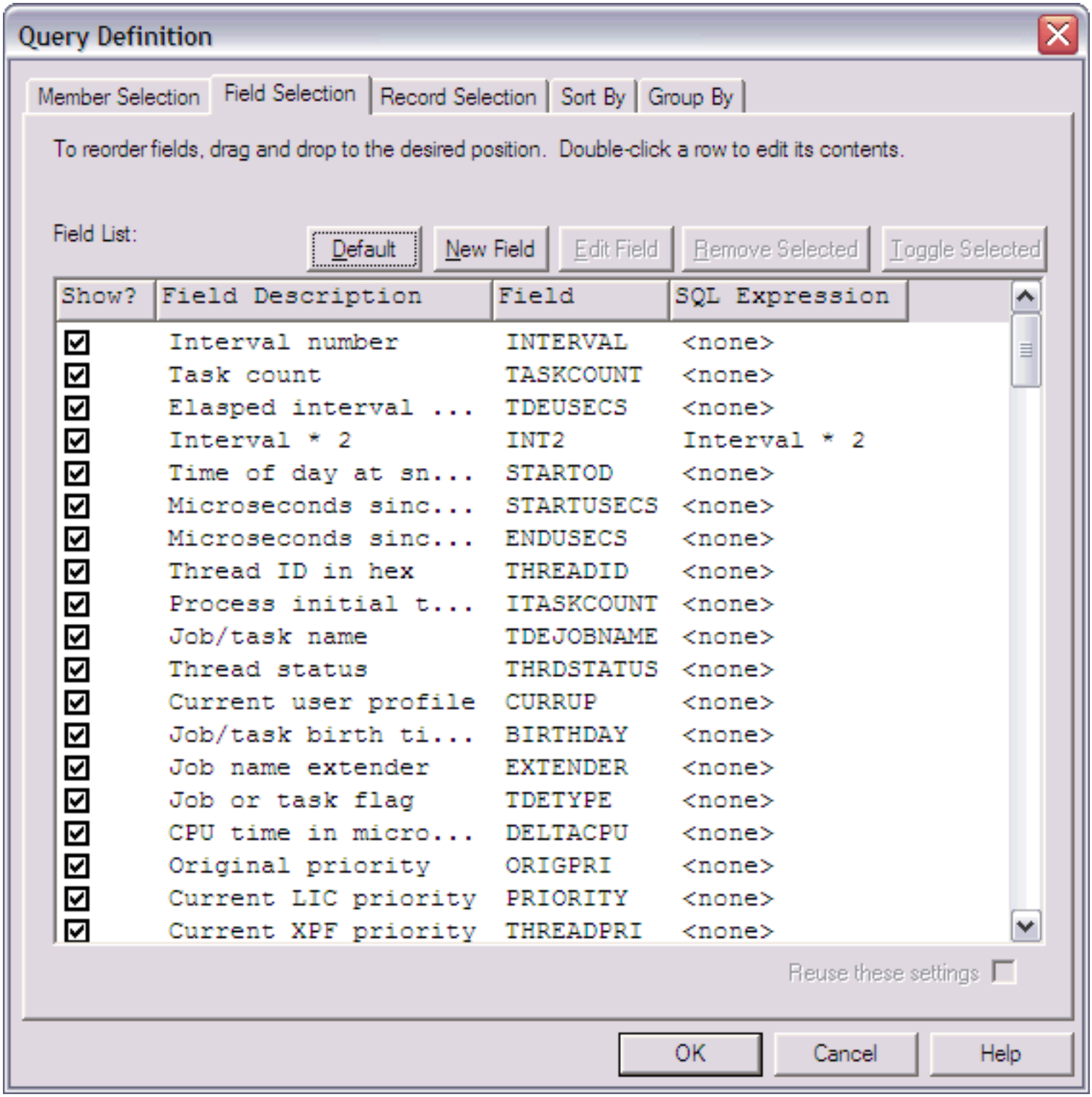
To create a new field:

1. Click the New Field button. After doing this a new row will be added to the list.
2. Double click on the new field added to the list or select it and press the Edit Field button.
3. Modify either the field description, field name or SQL expression through the Edit Field window.



If desired the Edit field window can remain open to change multiple fields at once by clicking other fields from the field list. The values for the selected field will be shown in the Edit Field window.

An example of the field selection page is the following:



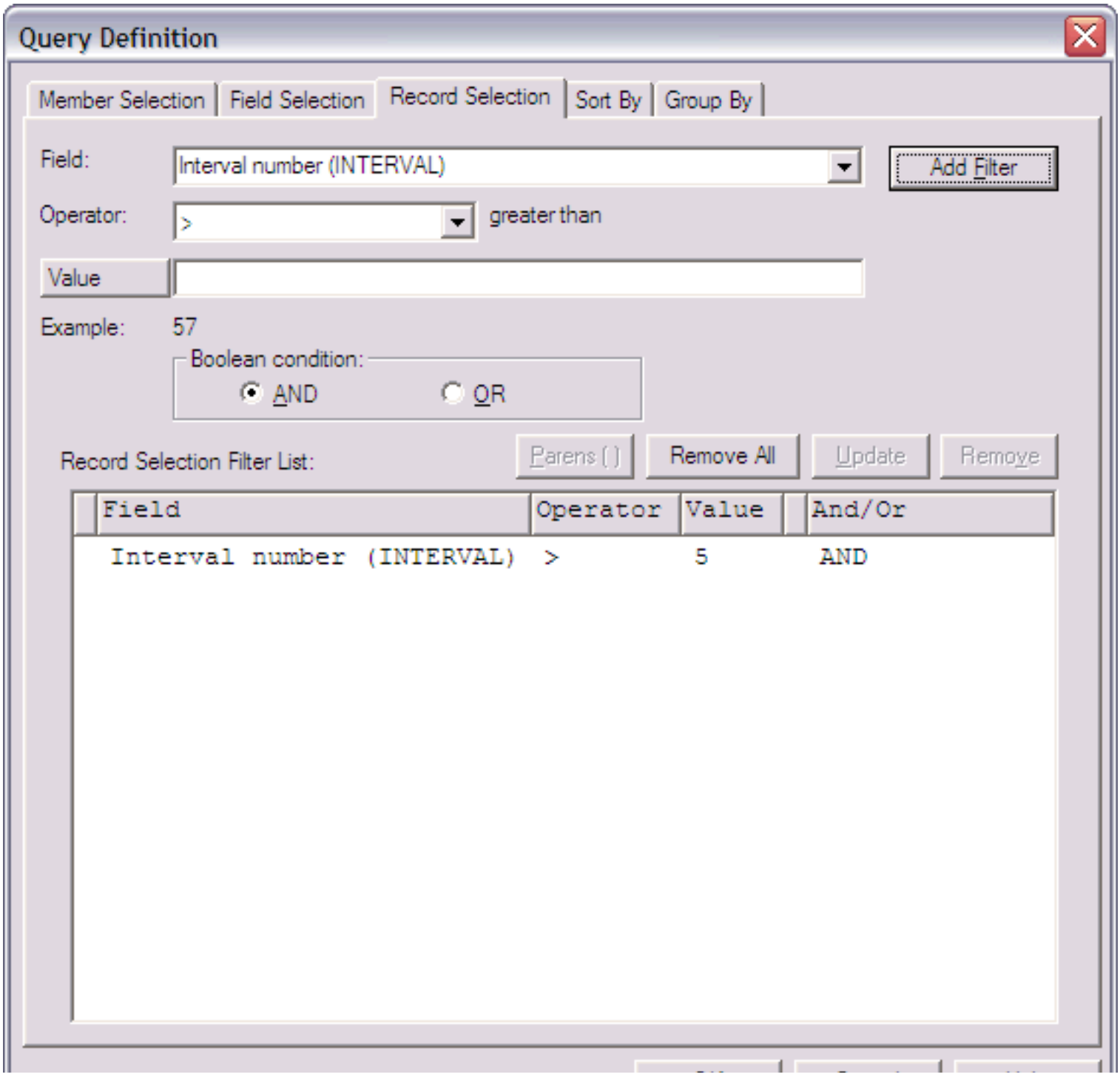
Note: The "reuse these settings" option at the bottom of the page is only enabled for PEX Analyzer report files. This allows the user to reorder the fields and this order saved onto the current PC (in the Windows Registry). The next time the analysis report is opened the same field ordering will be used.

Table of Contents | Previous | Next

1.6.3 Record Selection

The Record Selection Tab allows a user to limit the number of records or data returned in the active table or graph view. Any fields in the report may be filtered on using this interface.

An example of the Record Selection Page is shown below:



To Add a Filter:

1. First select the field or type in its short name into the field drop down list.
2. Depending on the type of the field selected, various operators available to that field will be displayed in the operator list.
3. Select the desired operator from the operator list.
4. Type in the value that the operator should test for. For example, to specify only records where CPUTIME field is greater than 10 the operator selected would be > (greater than) and the value would be 10.
5. Press the Add Filter button to add the filter to the list.
6. Press the OK button to close this interface and run the query using the new filter.

By selecting more than one concurrent records in the list and pressing the 'Parens ( )' button the user can add or remove a set of parentheses. To remove parentheses around multiple filters, select the range of records that contain the starting and ending parentheses and click the 'Parens ( )' button. Parentheses are necessary in order to make complex evaluations in the where clause of an SQL statement such as: CPUTIME >10 OR (IO > 1000 AND CPUTIME >= 1)

As the selection changes in the list, the interface objects above the list will change based on the current selection. This allows the user to quickly change values in the filter list by selecting any item in the list, changing any values from the fields above the list, and clicking the 'Update' button. The 'Update' button will update the selected row in the filter list.

A description of all the GUI elements on this panel follows:

GUI element name	Description
Field drop-down list	This is a list of every field in the current report. Select a field to filter by before clicking the 'Add Filter' or 'Update' buttons. The short name of a field may also be entered.

Operator list	This is a list of every operator available for the currently selected field. A text field has a different set of available operators than does a numeric field. The set of operators is also different for a timestamp field. The operators 'Field contains', 'Field starts with', 'Field ends with', 'Field xxx', etc are not valid for numeric and timestamp fields. The following operators are supported on this page: Equal Less than Less than or equal to Greater than Greater than or equal to Not equal Is null Is Not null Range List Not List Field contains Field starts with Field ends with Field does not contain Field does not start with Field does not end with
Value text box	Use this textbox to enter the value to apply to the current field using the selected operator. The value should match the format presented by the 'Example' label directly beneath the text box. Text fields should have their values enclosed in 'single quotes' and if the operator is 'Range', 'List' or 'Not list' then more than one values each separated by a space is expected. Whenever entering a value, follow the example provided.
Add Filter button	This button creates a new filter and adds the filter to the Record Selection Filter List.
Value/Expression button	This button allows the user to enter a valid SQL expression instead of a single value. This provides greater flexibility but requires that you know SQL syntax. Any errors in the SQL statement will prevent the query from running and will cause an SQL error message.
AND/OR options	Use this to indicate whether two filters should be ANDed together or OR'd together.

Parens ( ) button	The 'Parens ( )' button allows grouping of multiple filters in the Record Selection Filter List into a single logical expression by placing parentheses around the set of filters. If parentheses already exist for the starting and ending record in the selected range, the parenthese will be removed by pressing this button.
Remove All	This button will clear the list of filters.
Update button	This provides the ability to change the selected filter from the Record Selection Filter List.
Remove button	This button allows the user to remove one or more records from the Record Selection Filter List.
Record Selection Filter List	This is a list of all of the active filters to be applied to the report. Use the 'Add Filter' button to add a filter to the list. Press the OK button on the bottom of the Query Definition dialog to close the dialog and display the report using the filters from the list.

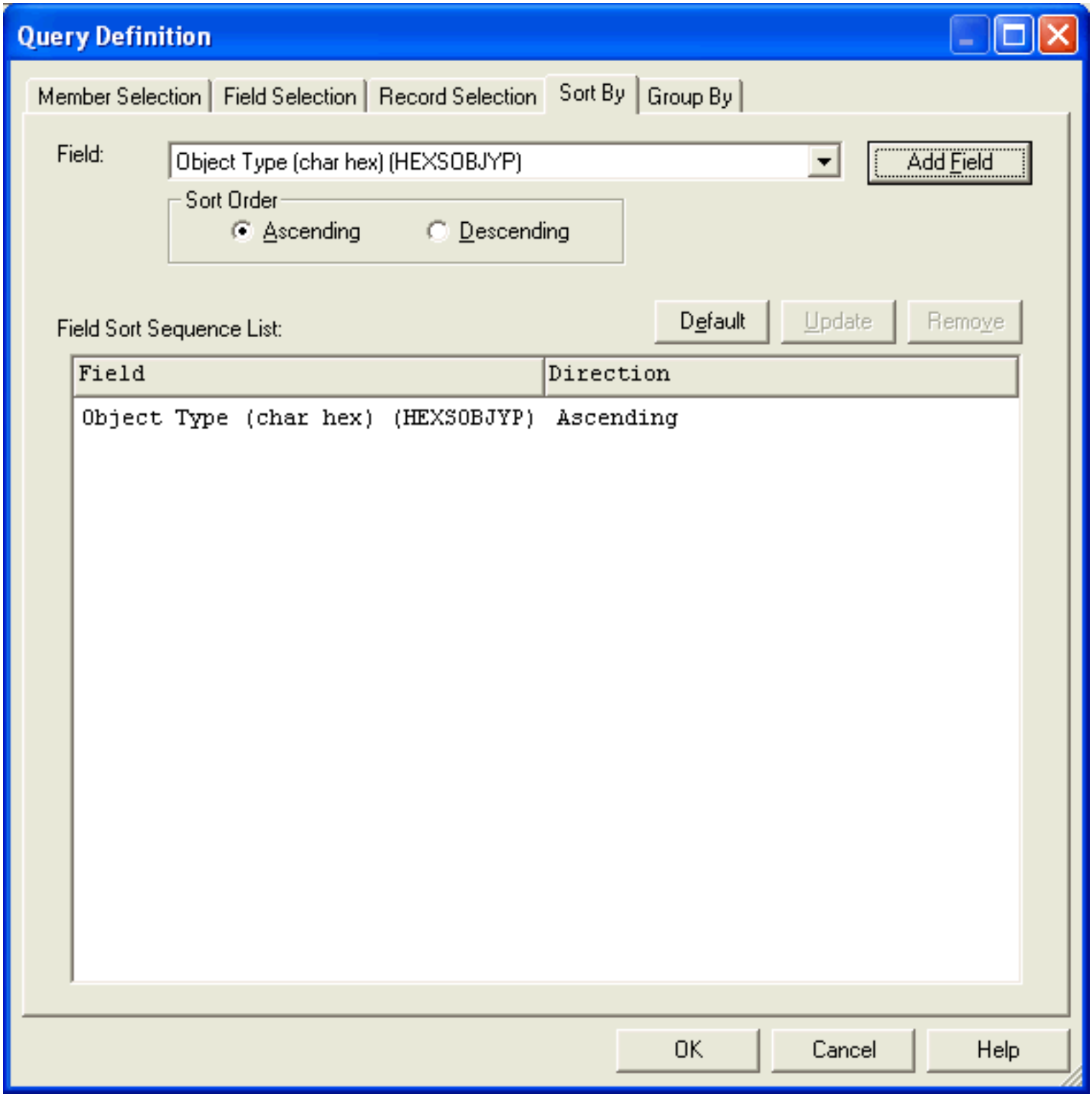
Table of Contents | Previous | Next

1.6.4 Sort By

The Sort By Page allows a user to change the order in which records are sorted in a table or graph view. This screen displays a list of fields to sort by and the sort direction for each field. The field at the top of the list has highest precedence in the sort sequence.

To add a field to the sort sequence list:

1. Select the field add to the list using the Field drop-down list.
2. Select the sort order: ascending or descending.
3. Press the Add Field button. The new field will be added to the Sort Sequence list.



The GUI elements on this page are described in the table below:

GUI element name	Description

Field drop-down list	This is a list of every field and its sort direction to use in the active view. The field at the top of the list has highest precedence in the sort sequence. Choose the field to add to the 'Field Sort Sequence' list before clicking the 'Add Field' or 'Update'
Add Field button	This button adds a field to the sort sequence indicated by the list, using the sort order currently specified.
Sort order options	Each field may be sorted in ascending (A-Z) or descending (Z-A) order. Choose the desired sort order before clicking 'Add Field' or 'Update'.
Default button	The default button changes the active sort order to whatever the default sort sequence is for the active report. In most cases, this will clear the sort sequence to nothing. In this case the sort order will be based on an ascending sort by relative record number of the raw data in the file.
Update button	The update button will change the sort sequence definition for the currently selected item in the list.
Remove button	This button allows the removal of one or more sort definitions from the list.
Field sort sequence list	This list represents the current sort order to apply to the active table or graph. Selecting any item in the list allows the option to change the value for the selected item using the 'Update' button.

Table of Contents | Previous | Next

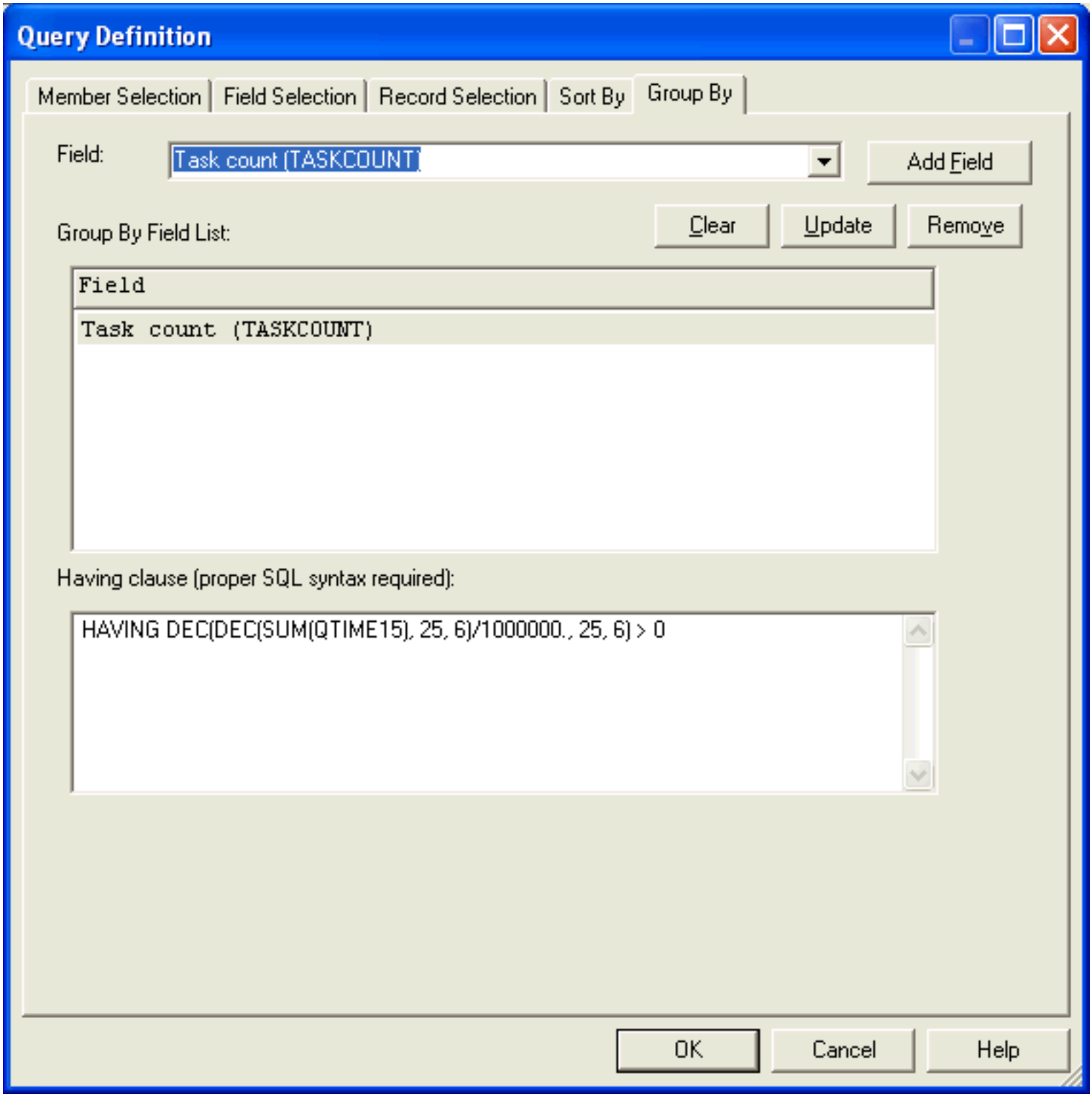
1.6.5 Group By

The Group By Page allows a user to define the fields that should be used as part of a Group by query. The field at the top of the list has highest precedence in the SQL GROUP BY clause.

Group by queries are only valid when the fields on the field selection page comply with the rules SQL has wth running group by SQL statements. Any fields that are not part of the group by clause must be summarized in order to exist in the field selection or the query will not run.

To add a field to the group by list:

1. Select the field to add to the list using the Field drop-down list.
2. Press the Add Field button. The new field will be added to the Group By Field List.



The GUI elements on this page are described in the table below:

GUI element name	Description
Field drop-down list	This is a list of every field in the active view. Choose the field to add to the 'Group By Field' list before clicking the 'Add Field' or 'Update' button

Add Field button	This button adds a field to the group by list.
Clear button	Removes all fields from the list.
Update button	Use the update button to change the selected field in the list to match the selected field in the drop-down list.
Remove button	The remove button will delete all selected fields from the Group By Field List.
Group By Field List	This list represents the GROUP BY clause in the Group By query. Selecting any item in the list allows the option to change the value for the selected item using the 'Update' button.
Having clause	This is the exact syntax to use for the Having clause for the group by query. Speciyfing a Having clause is not required.

Table of Contents | Previous | Next

1.6.6 Resetting

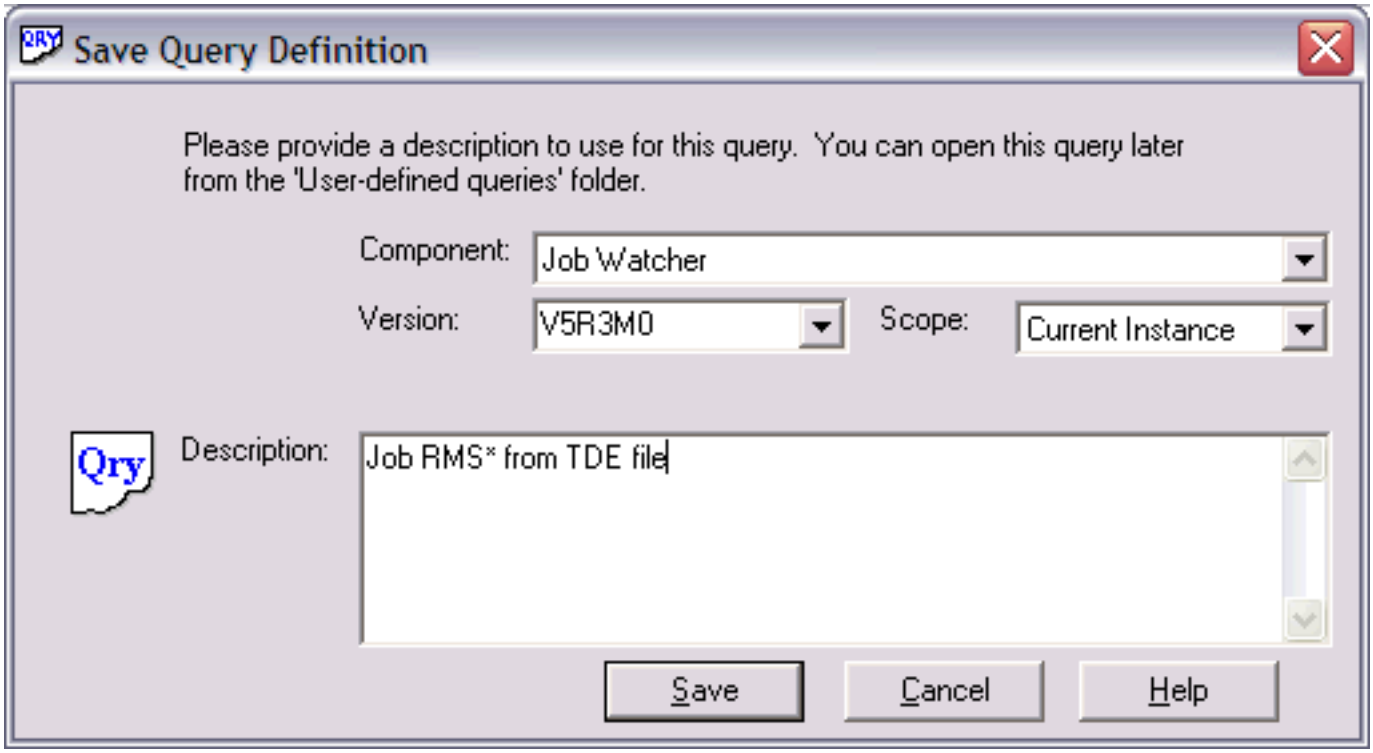
The reset submenu under the query definition popup menu may be used to reset a query back to its original state. Whenever a table or graph view is loaded the initial SQL statement is saved. If at some point it is desired to to discard the changes made to the report, use the Reset menu.

Table of Contents | Previous | Next

1.6.7 Saving

Query Definitions are saved using the Query Definition -> Save As... menu for an active table view. The query definition behind a graph view is saved automatically using the Graph Definition -> Save As... menu for a graph view. All Query Definitions are saved into the file QAIDRSQL04. The library where the query is saved into depends on the scope of the query (explained below).

An example of the Save Query Definition interface is shown below:



The interface objects within this window are described in more detail below:

Filter	Description
Component	The name of the component this query should be visible in.
Version	The version the query definition should be visible for. If this is set as V5R3, then this user defined query will not be visible under a V5R2 collection for example for the component indicated.

Scope	Use this option to set the scope of the query. This determines at which level (system, library, or current instance) the query should be visible. If the scope is library or current instance then the file QAIDRSQL04 in the same library the current collection is in will contain the query definition. System scoped query definitions are stored in library QUSRSYS. System scoped queries can be used against any libraries on the system that have the same file(s) that the current query uses. Query definitions can be opened from the user-defined queries folder underneath a collection.
Replace existing query definition option	Check this box to replace the saved query definition with the one currently being used. This checkbox is only visible if the table view was created from a user-defined query definition.
Description	The user-defined description for the definition. This description can be up to 250 characters long.

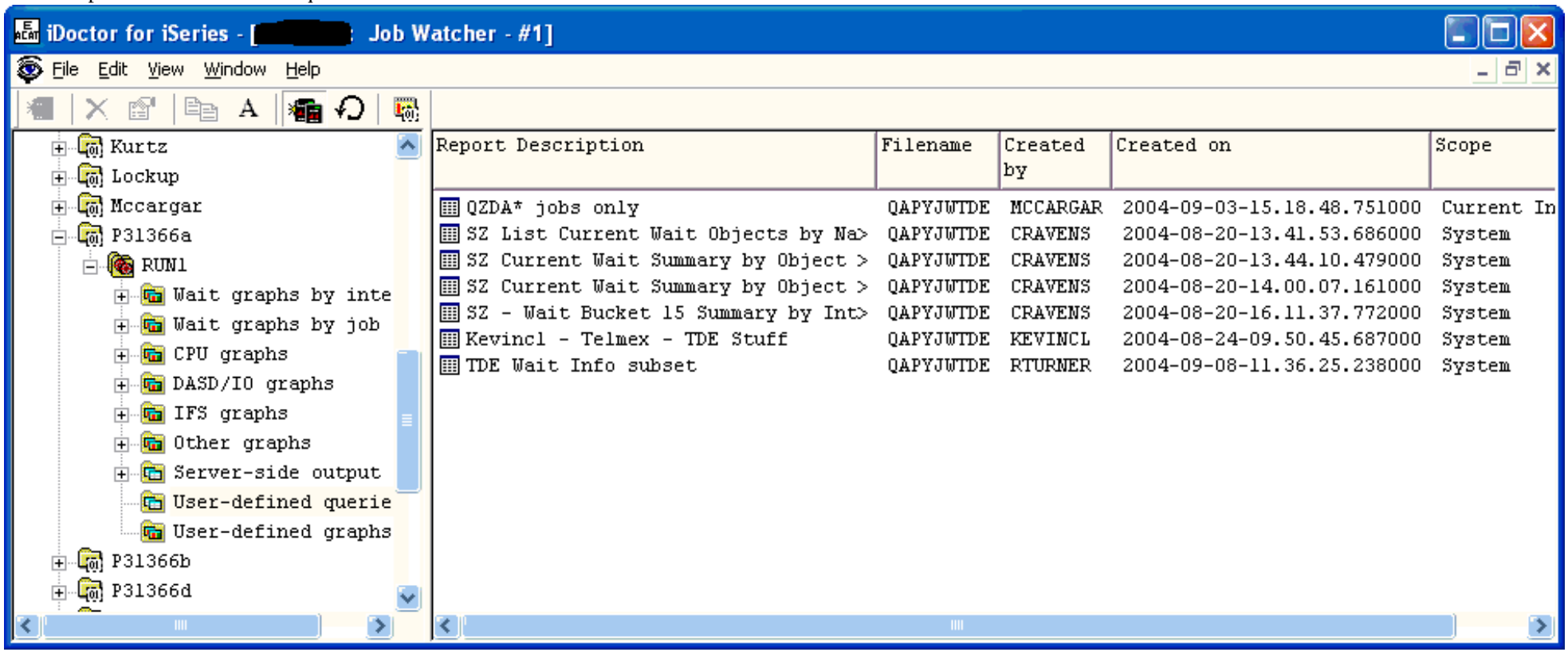
Table of Contents | Previous | Next

1.6.8 Working with Query Definitions

After a query definition has been created it will be displayed in the user-defined queries folder under a collection in any of the iDoctor components.

This folder lists the query definitions available for the current collection. Only queries that were saved over the same type of files as the current selection will be visible. For example, PEX Analyzer queries are not visible when using Job Watcher within the user-defined queries folder.

An example of the user-defined queries folder is:



The fields shown in the user-defined queries folder are:

Report description - name of the query when it was last saved.
Filename - the primary file (or first file) in the SQL statement
Created by - user profile that last saved this query definition
Created on - the date and time when the query definition was last changed or created
Scope - describes the visibility of the query definition

Current instance - the query definition is only visible for the library and collection or analysis for which it was created. The query definition is only visible on the current system.
Library - the query definition is visible for all collections/analyses in the same library of the same type for which the query was created. The query definition is only visible on the current system.
System xyz - the query definition is visible for all collections/analyses in all libraries of the same type for which the query was created. The query definition is only visible on the current system.
All systems - the query definition is visible for all collections/analyses on all systems.

The menu options available for a query definition are:

Menu	**Description**
Open Table	Opens the selected query definition as a table view in a new or existing data viewer.
Delete...	Removes the selected query definition(s) from the system.
Set Scope	Modifies the scope of the selected query definition to the specified type.
Properties	Displays the SQL statement behind the selected query definition.

Table of Contents | Previous | Next

1.7 Graph Definitions

In iDoctor for iSeries, users can define graphs over data within any table view or graph view desired. This is done by creating a graph definition which is the information that builds a user-defined or iDoctor-supplied graph. Like query definitions, graph definitions are stored on the server in a database file.

A graph definition defines everything needed to display the graph including a reference to the query definition needed to produce the graph's data. Whenever a graph view is saved, the current query definition as well as the current graph definition is saved.

The interface is accessible by either defining a new graph definition using the Graph Definition | Define New... popup menu within a table view or by using the Graph Definition popup menu in a user-defined graph view.

There are a sets of pages which let the user define the graph definition. These pages are discussed in greater detail in the next sections. A summary of the pages that make up a graph definition is shown below:

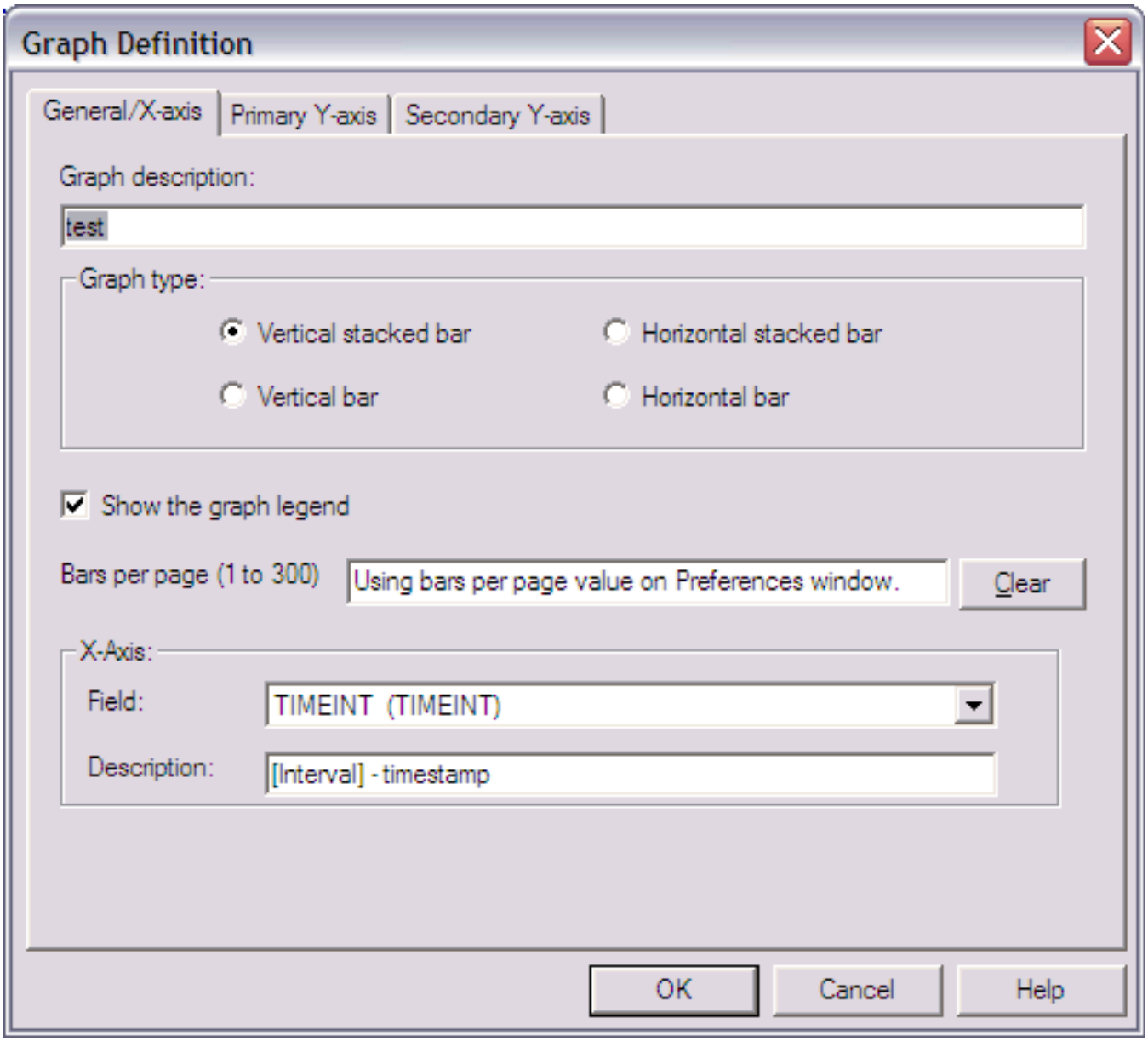
Page Name	**Description**
General/X-Axis	Defines the general features of the graph, like the type of graph, as well as information about the X-Axis.
Primary Y-Axis	This page defines the fields, colors and desciptions to use for the bars in the graph. Up to 32 different fields/colors may be defined in the graph definition.
Secondary Y-Axis	This page identifies the secondary Y-axis. This axis consists of one or two lines of user-defined color. This axis is optional.

Table of Contents | Previous | Next

1.7.1 General Page

The general page lets the user define the graph title, type, bars per page as well as the field used for the X-axis.

A example of the General Page is shown below:



The GUI elements on the General Page are described in detail within the table below:

Element name	Description
Graph Description	A description of the graph. (50 characters max)
Graph type	Indicates the layout of the graph. Vertical bar graph will produce a graph with side-by-side vertical bars. Horizontal bar graph will produce a graph with side-by-side horizontal bars.
Show the graph legend	This checkbox indicates whether or not the graph legend should be displayed. On some graphs with many colors/graphs this option is turned off by default in order to allocate more screen space to the data itself.
Bars per page	This value can be used to optionally specify the number of bars to show per page on this graph. If a value is not specified on this page then the applicable bars per page value on the Preferences window will be used instead.
X-Axis Field	The field information to use when for the X-Axis on the graph. Typically this is something like Interval number or Job name. This can be a numeric field or a character field and is a required field.
X-Axis Description	The description to display under the X-Axis on the graph. This value defaults to the description of the field, but it can be changed to something else if desired.

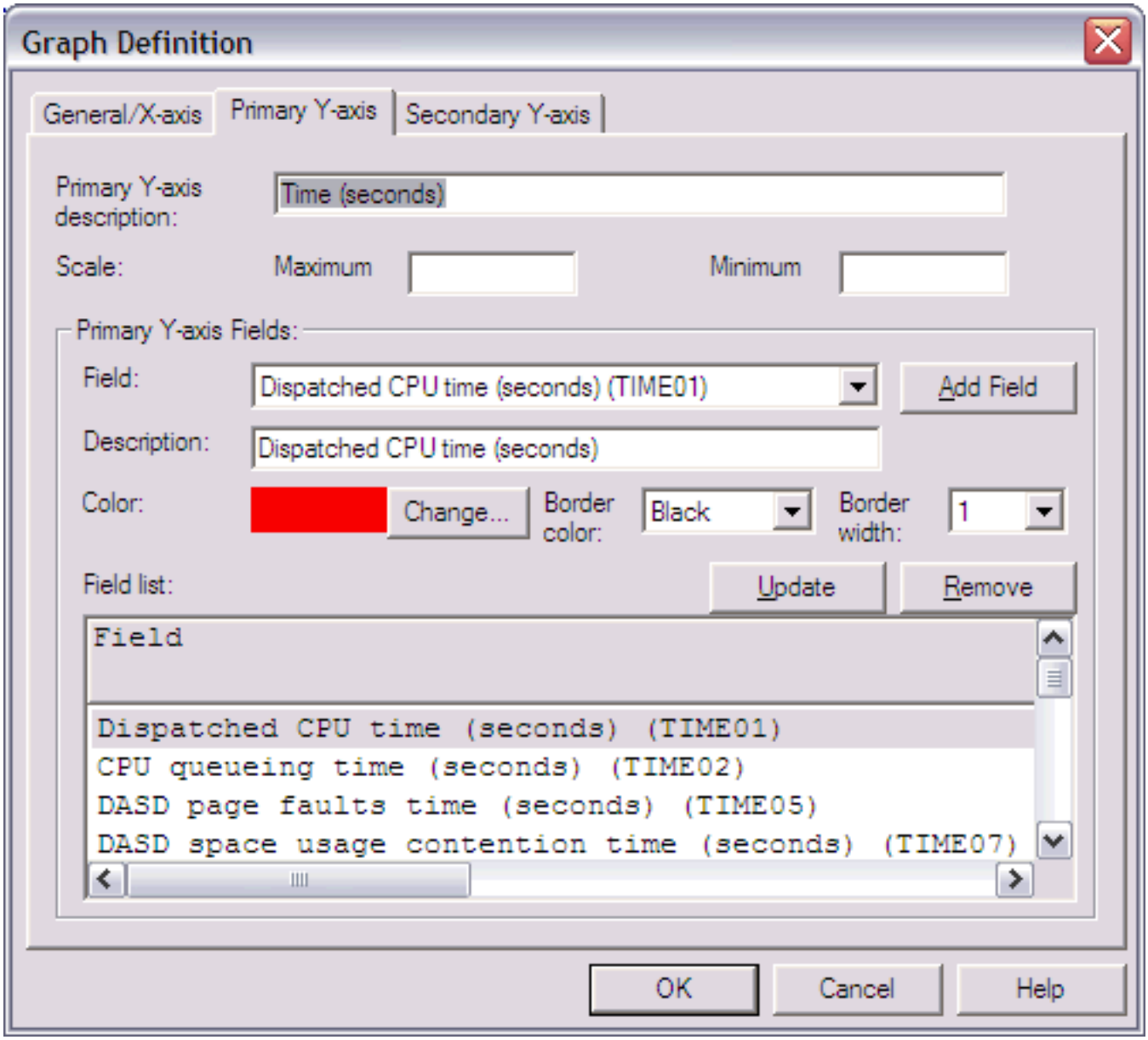
Table of Contents | Previous | Next

1.7.2 Primary Y-axis

Use the Primary Y-axis page to define the fields that should be displayed on the graph. Each field represents a bar on the graph and can have a different color and customized description.

To Add a Field.

1. Select the field you wish to use for the new field from the Field drop-down list.
2. If desired, modify the description of the field from the default field description.
3. If desired, define a color for this Y-Axis field. If this is not done, a color will be automatically assigned.
4. Click the Add Field button to the add the field to the list of fields.



The elements on this page are described in detail within the table below:

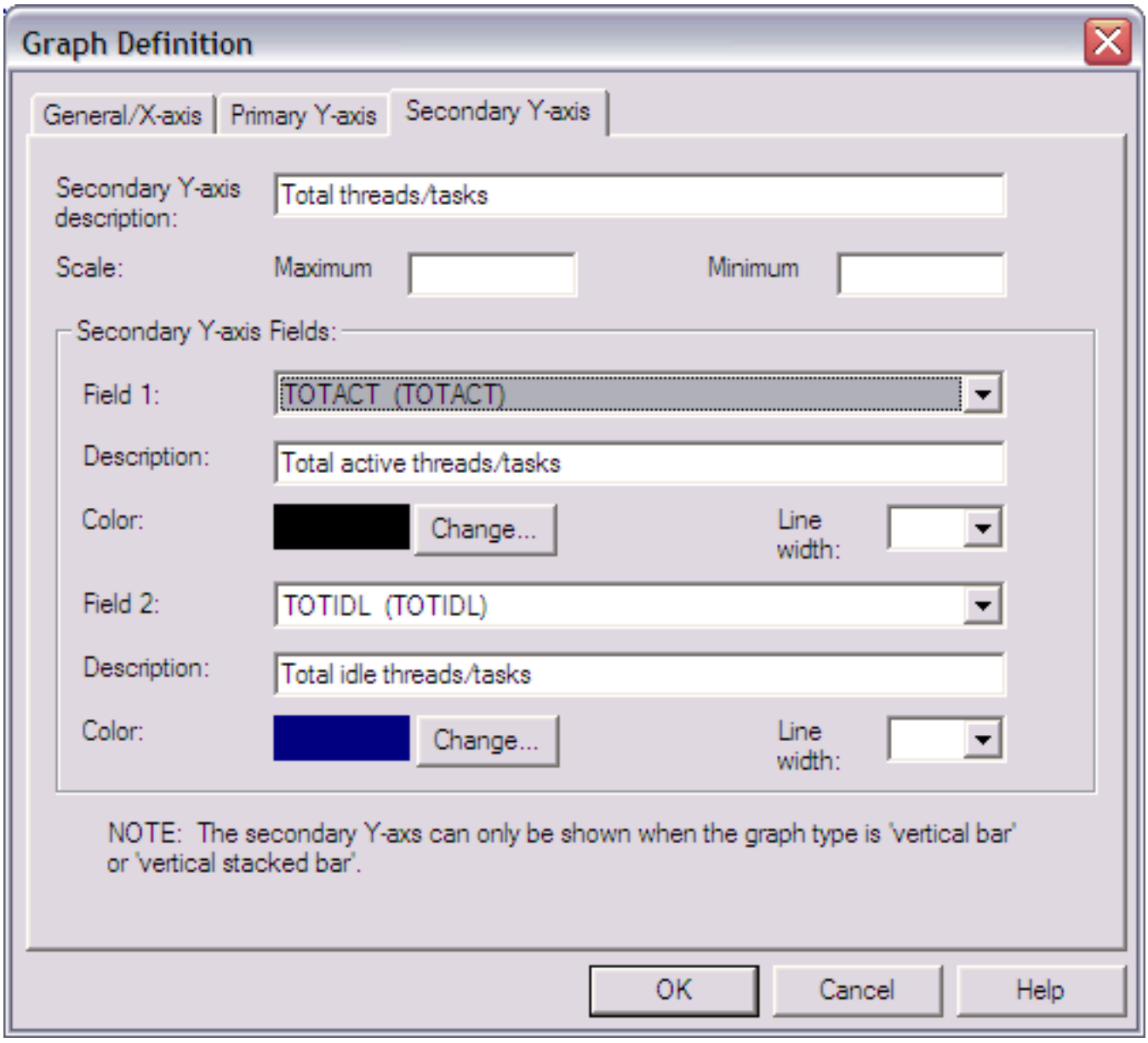
Element name	Description
Y-axis Description	A description of the primary Y-axis. (50 characters max)
Scale	Allows the primary Y-axis scale to be set to a specific maximum and/or minimum value. If any numbers are greater or less than the values specified the bar will be truncated. Using this feature is optional.
Field	Allows selection of a field to add to the field list. Changing the field will update the description to match the field description for the selected field.

Description	The 50 character description that identifies the data in the graph. This description will be displayed in the graph's legend.
Color	The color to use for the field selected. If no color is selected a unique color will be provided automatically.
Border color	The color to give the border. By the default this color is black but it could set to another color to make the bar stand out more on the graph than others.
Border width	The number of pixels the border should contain.
Update	The update button is used to modify the selected field in the field list. For example this option could be used to change the color of an existing field in the field list.
Remove	This option will remove the selected fiels from the field list.
Field list	Displays the field names, descriptions and colors to use for the bars in the bar graph.

Table of Contents | Previous | Next

1.7.3 Secondary Y-axis

This page allows the user to define a secondary Y-axis on a bar graph. This axis can contain 2 fields which are represented as solid lines of a user-defined color. This axis is only visible for horizontal bar graphs.



The GUI elements on this page are described in detail within the table below:

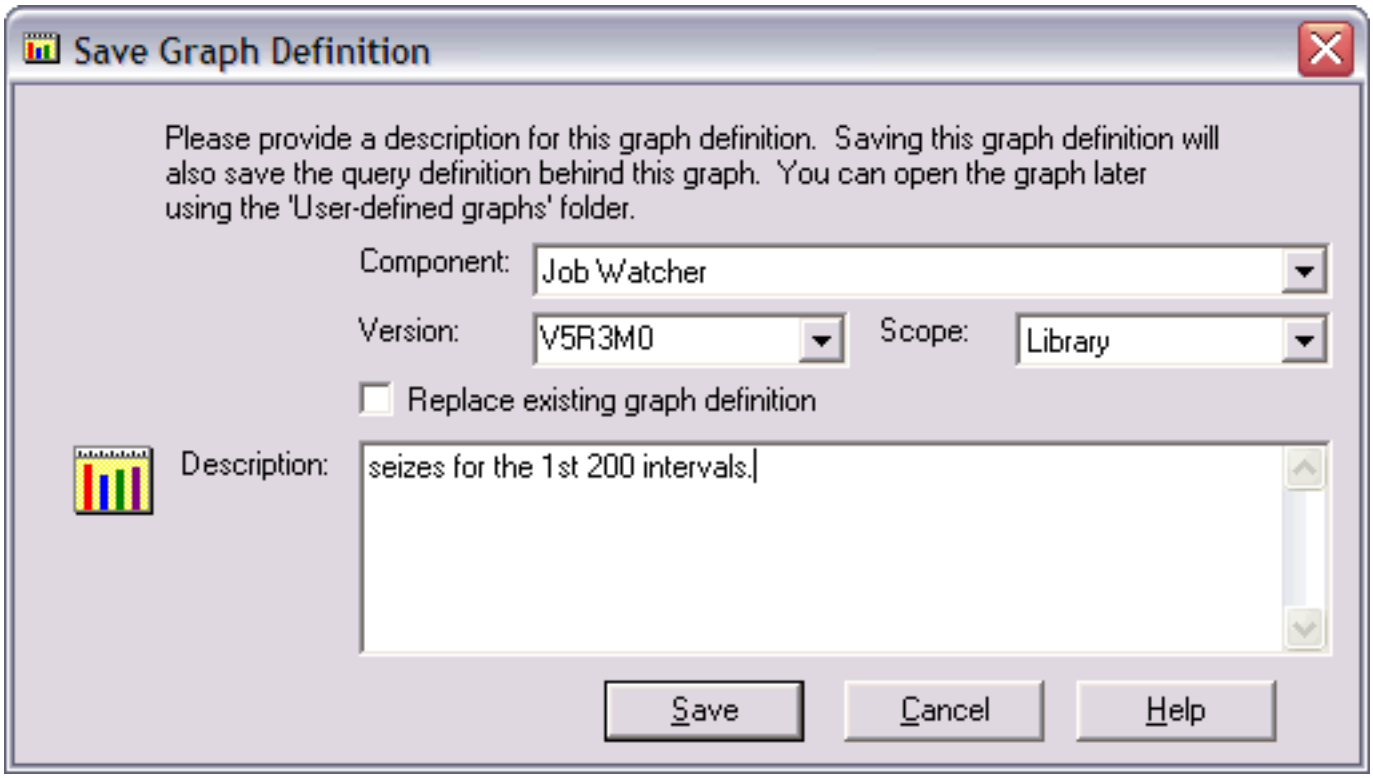
Element name	Description
Secondary Y-axis description	The title to give the secondary Y-axis.
Scale	Allows the secondary Y-axis scale to be set to a specific maximum and/or minimum value. If any numbers are greater or less than the values specified they will not be visible. This feature is optional.
Field 1	The name of the field to use for the 1st line of the secondary Y-axis. This value should be blank (first selection in the list) if no secondary Y-axis is desired.
Description 1	A description of the 1st line of the secondary Y-axis. (50 characters max)
Color 1	The color to use for the 1st line of the secondary Y-axis.
Line width 1	The number of pixels wide to draw the 1st line of the secondary Y-axis.
Field 2	The name of the field to use for the 2nd line of the secondary Y-axis. This value should be blank (first selection in the list) if no secondary Y-axis is desired.
Description 2	A description of the 2nd line of the secondary Y-axis. (50 characters max)
Color 2	The color to use for the 2nd line of the secondary Y-axis.
Line width 2	The number of pixels wide to draw the 2nd line of the secondary Y-axis.

Table of Contents | Previous | Next

1.7.4 Saving

Graph Definitions are saved using the Graph Definition -> Save As... menu for the active graph view. All Graph Definitions are saved into the file QAIDRGPH08. The library where the graph is saved into depends on the scope of the graph.

An example of the Save Graph Definition interface is shown below:



The interface elements within this window are described in more detail below:

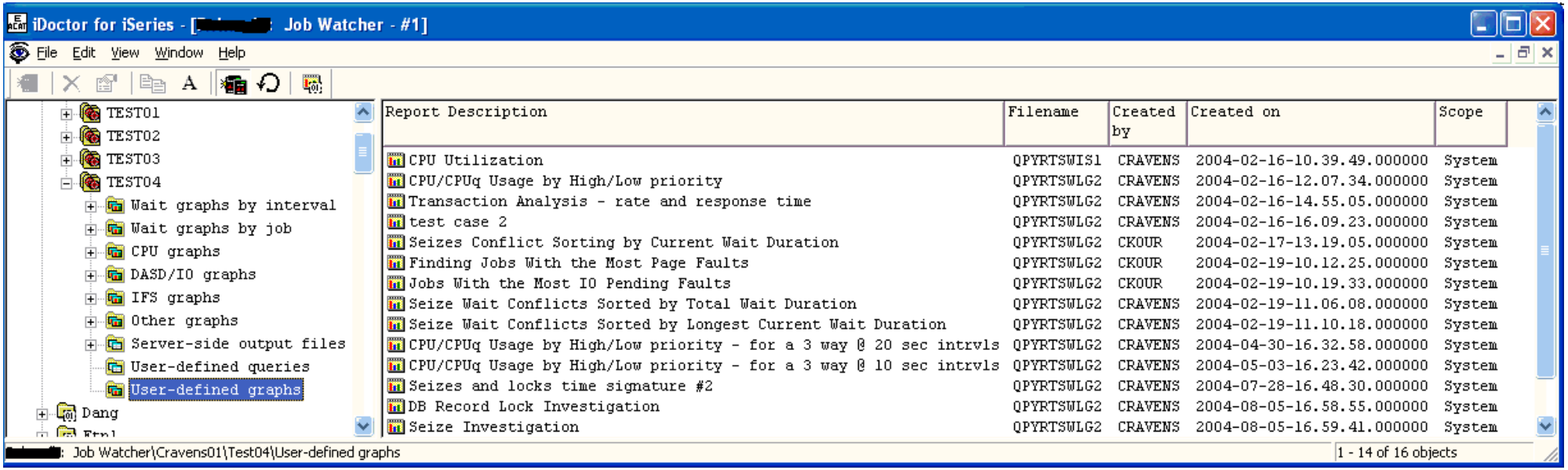
Interface element	Description
Component	The name of the component this graph should be visible in.
Version	The version the graph definition should be visible for. If this is set as V5R3, then this user defined graph will not be visible under a V5R2 collection for the component indicated.

Scope	Use this option to set the scope of the graph. This determines at which level (system, library, or current instance) the graph should be visible. If the graph scope is library or current instance then the file QAIDRGPH08 in the same library the current collection is in will contain the graph definition. System scoped graph definitions are stored in library QUSRSYS. System scoped graphs can be used against any libraries on the system that have the same underlining file(s) that the current graph uses. Graph definitions can be opened from the user-defined graphs folder underneath a collection.
Replace existing graph definition option	Check this box to replace the saved graph definition with the one currently being used. This checkbox is only visible if the graph view was created from a user-defined graph definition.
Description	The description for the Graph Definition. This description can be up to 250 characters long.

1.7.5 Working with Graph Definitions

After a graph definition has been created it will be displayed in the user-defined graphs folder under a collection in any of the iDoctor components.
This folder lists the graph definitions available for the current collection. Only graphs that were saved over the same type of files as the current selection will be visible. For example, PEX Analyzer graphs are not visible when using Job Watcher within the user-defined graphs folder.

An example of the user-defined graphs folder is:



The fields shown in the user-defined graphs folder are:

Report description - name of the graph when it was last saved.
Filename - the primary file (or first file) for the SQL statement used behind the graph
Created by - user profile that last saved this graph definition
Created on - the date and time when the graph definition was last changed or created
Scope - describes the visibility of the graph definition
Current instance - the graph definition is only visible for the library and collection or analysis for which it was created. The graph definition is only visible on the current system.
Library - the graph definition is visible for all collections/analyses in the same library of the same type for which the graph was created. The graph definition is only visible on the current system.
System xyz - the graph definition is visible for all collections/analyses in all libraries of the same type for which the graph was created. The graph definition is only visible on the current system.
All systems - the graph definition is visible for all collections/analyses on all systems.

The menu options available for a graph definition are:

Menu	Description
Open Graph	Opens the selected graph definition as a graph view in a new or existing data viewer.
Delete...	Removes the selected graph definition(s) from the system.
Set Scope	Modifies the scope of the selected graph definition to the specified type.

Properties	Displays the SQL statement behind the selected graph definition.

Table of Contents | Previous | Next

1.8 Find Window

The Find Window allows a user to perform a search over a column in a Table View. Use the Edit | Find... menu or right-click on a Table View and choose the Find... menu to use the Find Window. Find allows the user to search for a text string within a specfic column.

An example of the find window is shown below:

The following table summarizes the interface elements on the Find window.

Interface Element	Field Description
Find what textbox	Enter the string you would like to search on in this textbox.
Column drop down list	This list contains the name of every column in the active Table View. Select the column you would like to search on from this list.
Find Next button	Clicking this button will perform a search over the active Table View for the next occurence (depending on direction up or down) of the string in the Find what textbox in the specified column.
Find All button	Click this button to find and select all matching occurrences of the Find what textbox in the specified column.

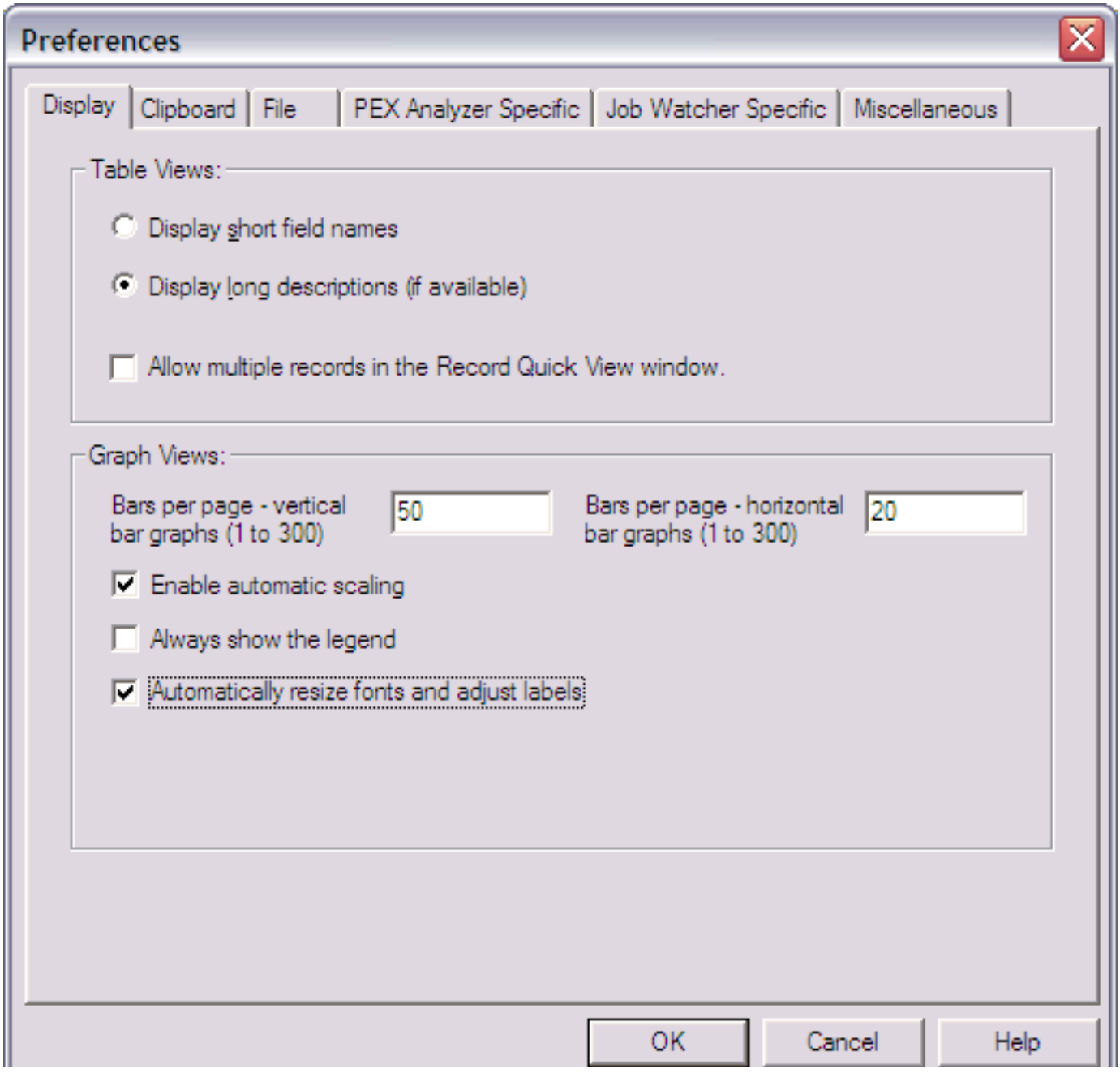
Match whole column only	Check this to indicate that records should only match if the value in the find what text box matches exactly with the searched on column value.
Match case	Check this to perform a case-sensitive search.
Direction (Up/Down)	These fields indicate whether the search should be performed over the remaining data in the Table View or the preceding data.

Table of Contents | Previous | Next

1.9 Preferences

The Preferences window allows a user to work with the customizable options in the iDoctor for iSeries client. Several different categories of options are available and each category is presented on a different page.

The Preferences window is accessible via the Edit | Preferences menu in the Data Viewer or from the iDoctor for Series Main Window. See the next sections for information on each page in the Preferences window.



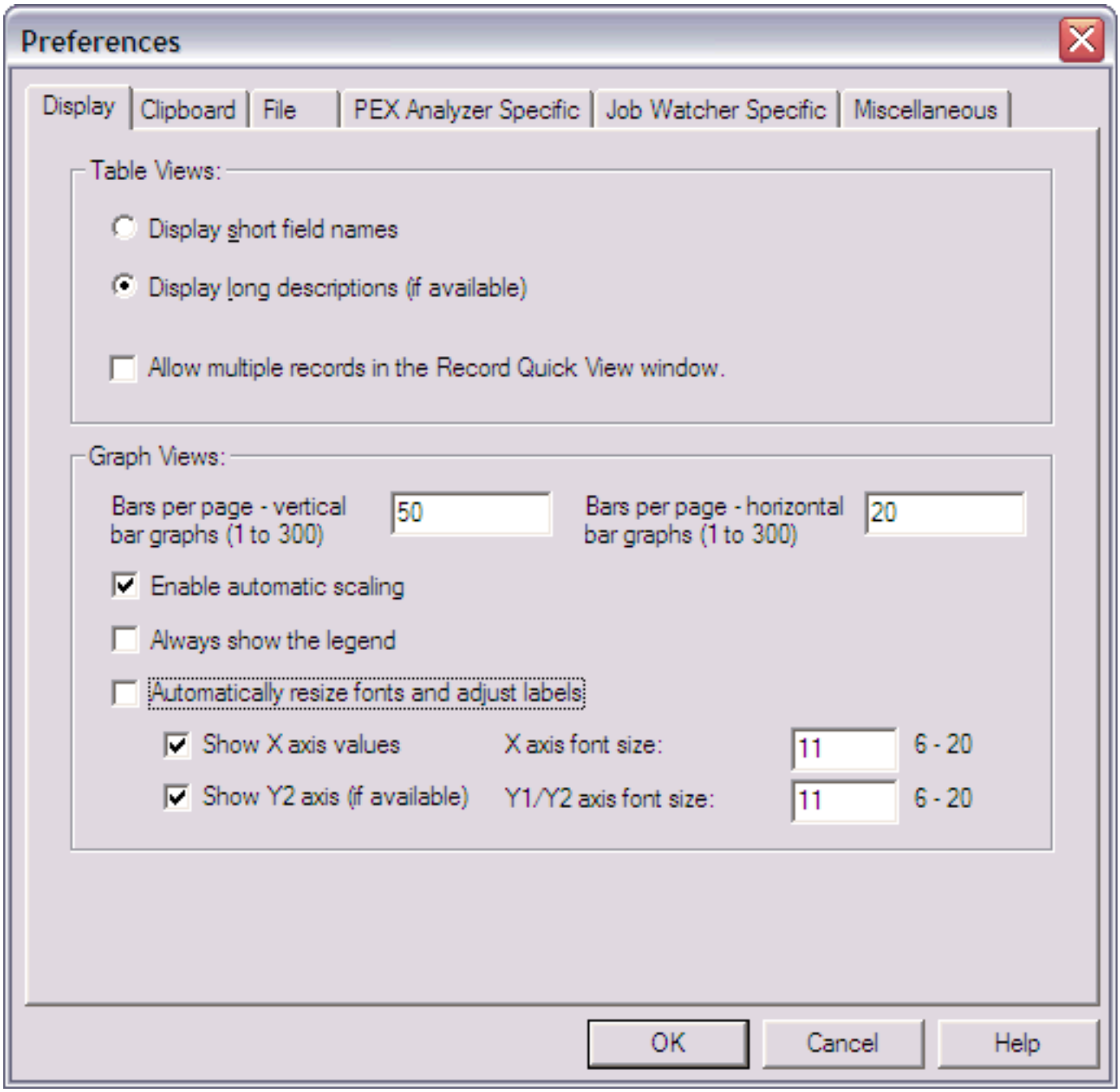

OK
Cancel
Help

Table of Contents | Previous | Next

1.9.1 Display

The Display page on the Preferences window lets the user work with options that effect the visible presentation of table or graph views in the iDoctor for iSeries client.

An example of this interface is shown below:



The options available on this page are summarized in the tables below:

Table View Options	Description
Display short or long name option	This option lets the user decide if long field descriptions should be displayed or short field names in the column headings of all table views. **Note:** Short field names will be displayed if the long descriptions are not available or not defined within the file being viewed.
Allow multiple records in the Record Quick View window	This option allows the user to select multiple records in a table and use the record quick view option to compare multiple records side by side.

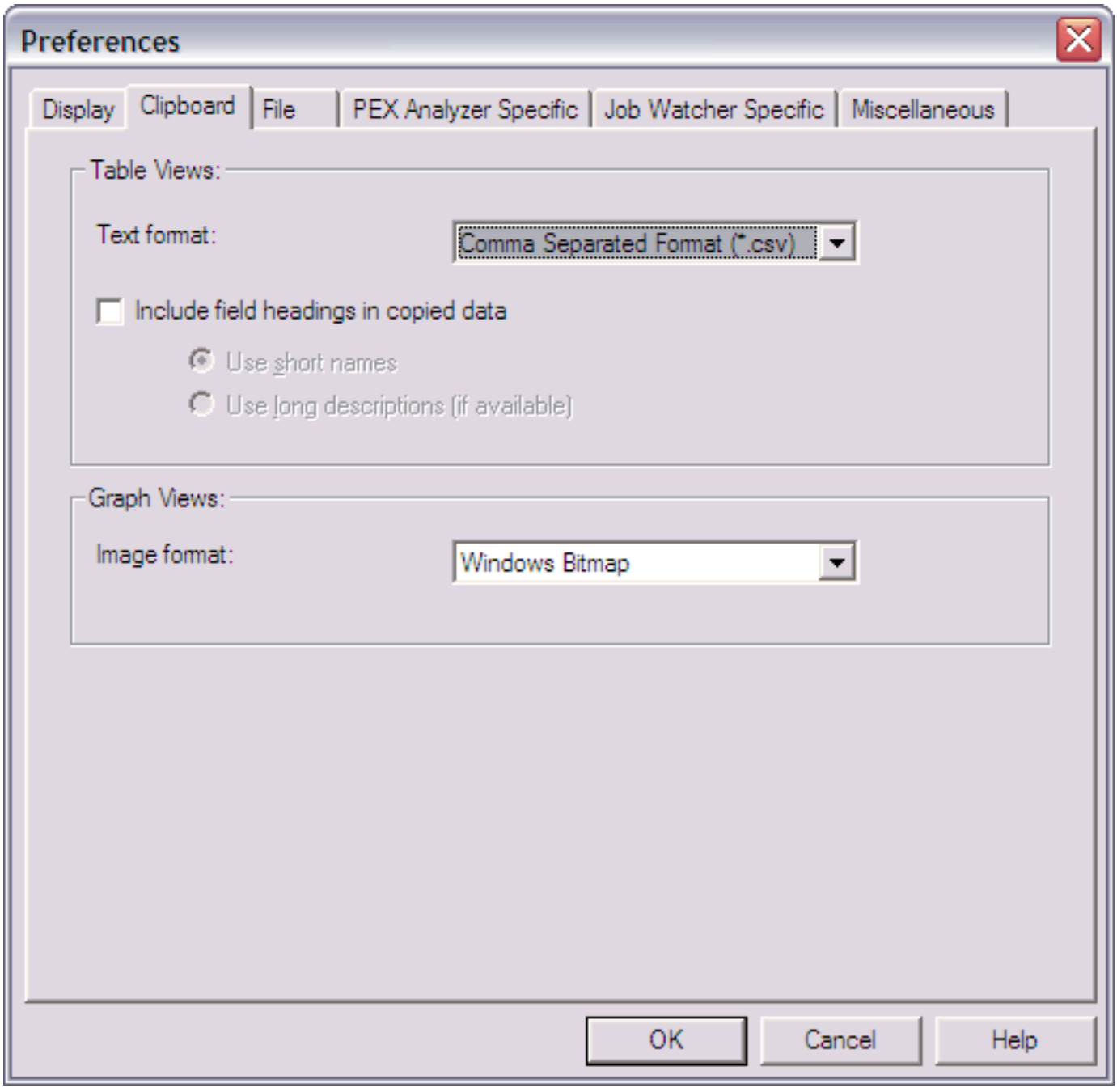
Graph View Options	Description
Bars per page - vertical	Indicates how many bars should be displayed per page in a vertical bar, or stacked vertical bar graph.
Bars per page - horizontal	Indicates how many bars should be displayed per page in a horizontal bar, or stacked horizontal bar graph.
Enable automatic scaling	Indicates if the graph should automaticly resize the scale on the Y-axis each time the current position in the graph changes. If this option is turned off the scale will be fixed based on the maximum and minimum values of the first page of the graph when it is opened.
Always show the legend	Indicates if the graph legend should always be shown. If checked this will override the option in some graph defintions that indicates the graph legend should not be shown.
Automatically resize fonts and adjust labels	This option controls whether or not the fonts and labels should be automatically resized and adjusted (recommended on).
Show X-Axis values	Indicates if labels for the X-Axis values should be displayed.
Show Y2-Axis (if available)	Indicates if the Y2-Axis (the secondary Y-Axis) should be displayed. This axis is not used on all graphs.
X-Axis font size	Indicates the font size to use for values on the X-Axis. The higher the number the larger the font will appear.
Y-Axis font size	Indicates the font size to use for values on the Y-Axis. The higher the number the larger the font will appear.

Table of Contents | Previous | Next

1.9.2 Clipboard

The Clipboard page on the Preferences window lets the user work with the 'Copy to Clipboard' options available for table or graph views in the iDoctor for iSeries client.

An example of this interface is shown below:



The options available on this page are summarized in the tables below:

Table View Options	Description
Text format	Select the desired text format when copying records or cell selections to the clipboard. The possible choices are: comma separated, tab separated and rich text format.
Include field headings in copied data	Check this option to indicate that field headings should be included as the first record of data when copying data to the clipboard. If this option is checked you can choose to use short field names or long descriptions for the copied output.

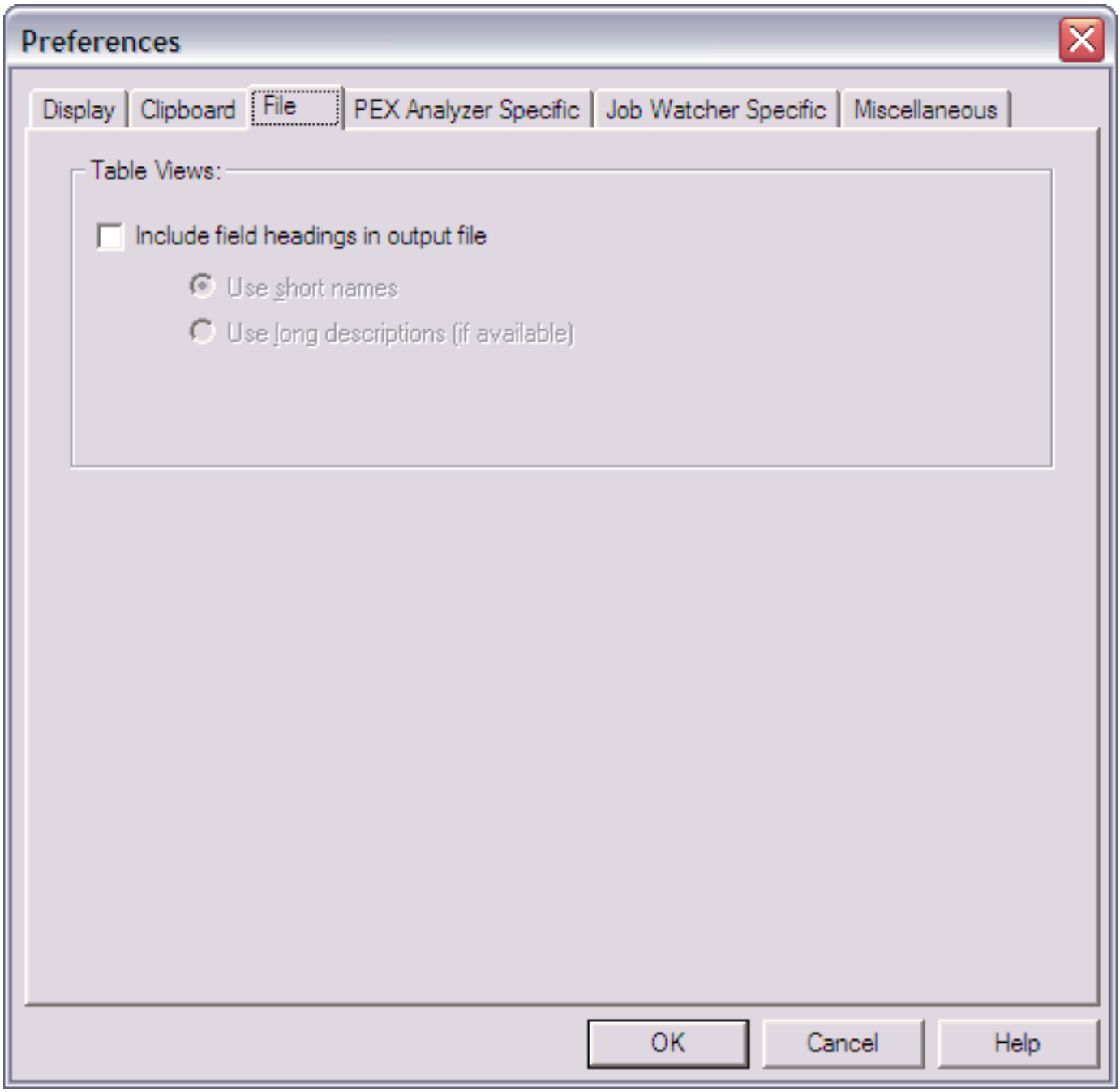
Graph View Options	Description
Image format	Select the desired image format when copying a graph view to the clipboard. This option will copy the currently active graph view to the clipboard as a bitmap.

Table of Contents | Previous | Next

1.9.3 File

The File page on the Preferences window lets the user work with options related to creating output files from a table view's data.

An example of this interface is shown below:



The options available on this page are:

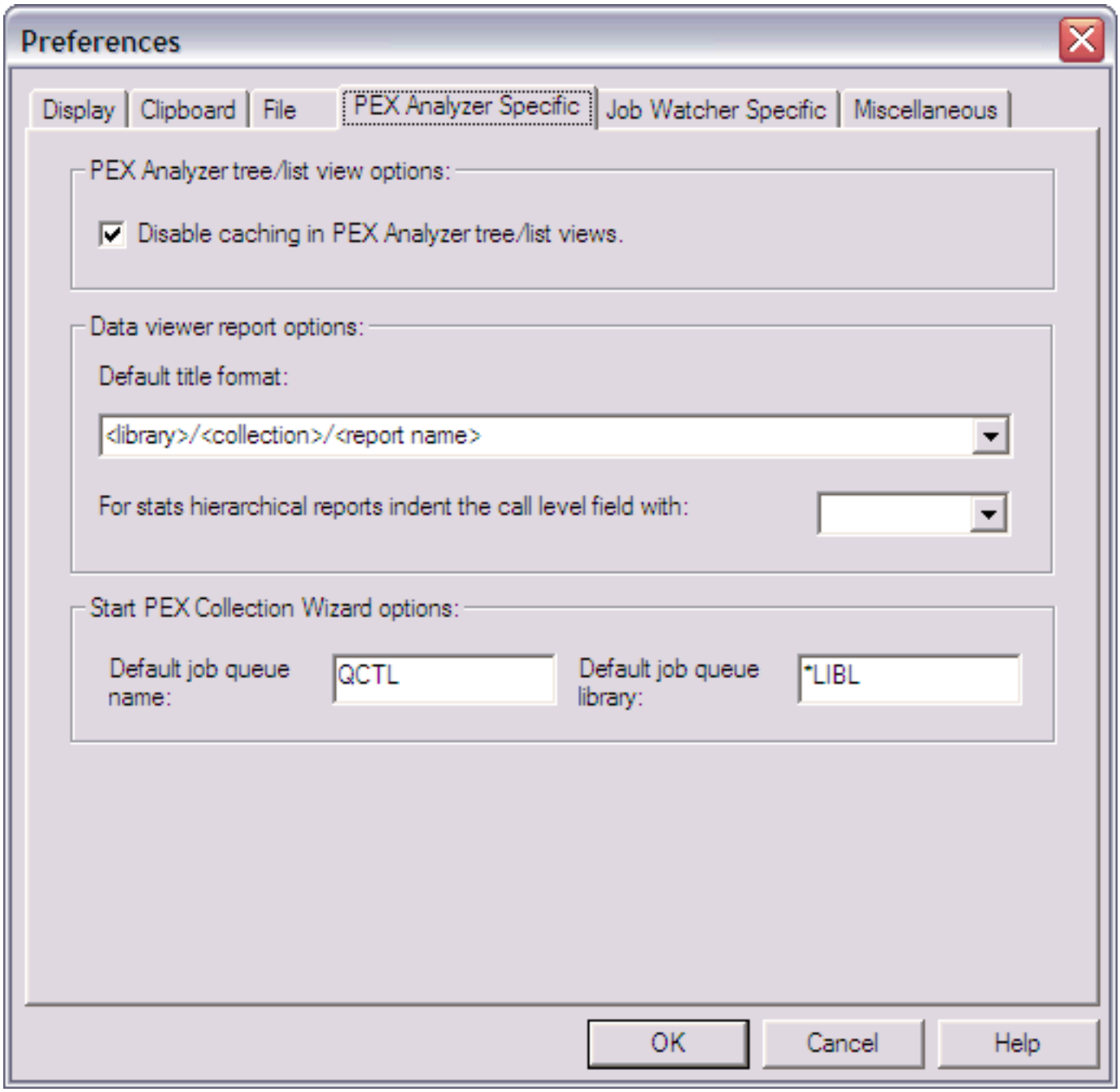
Table View Options	Description
Include field headings in output file	Check this option to indicate that field headings should be included as the first record of data when generating the output file. If this option is checked the user may choose to use short field names or long descriptions in the output file. To generate an output file use the File | Save As... menu when for an active Table View.

Table of Contents | Previous | Next

1.9.4 PEX Analyzer

The PEX Analyzer page on the Preferences window lets the user work with options related to the PEX Analyzer component of iDoctor for iSeries.

An example of this interface is shown below:



The options available on this page are summarized below:

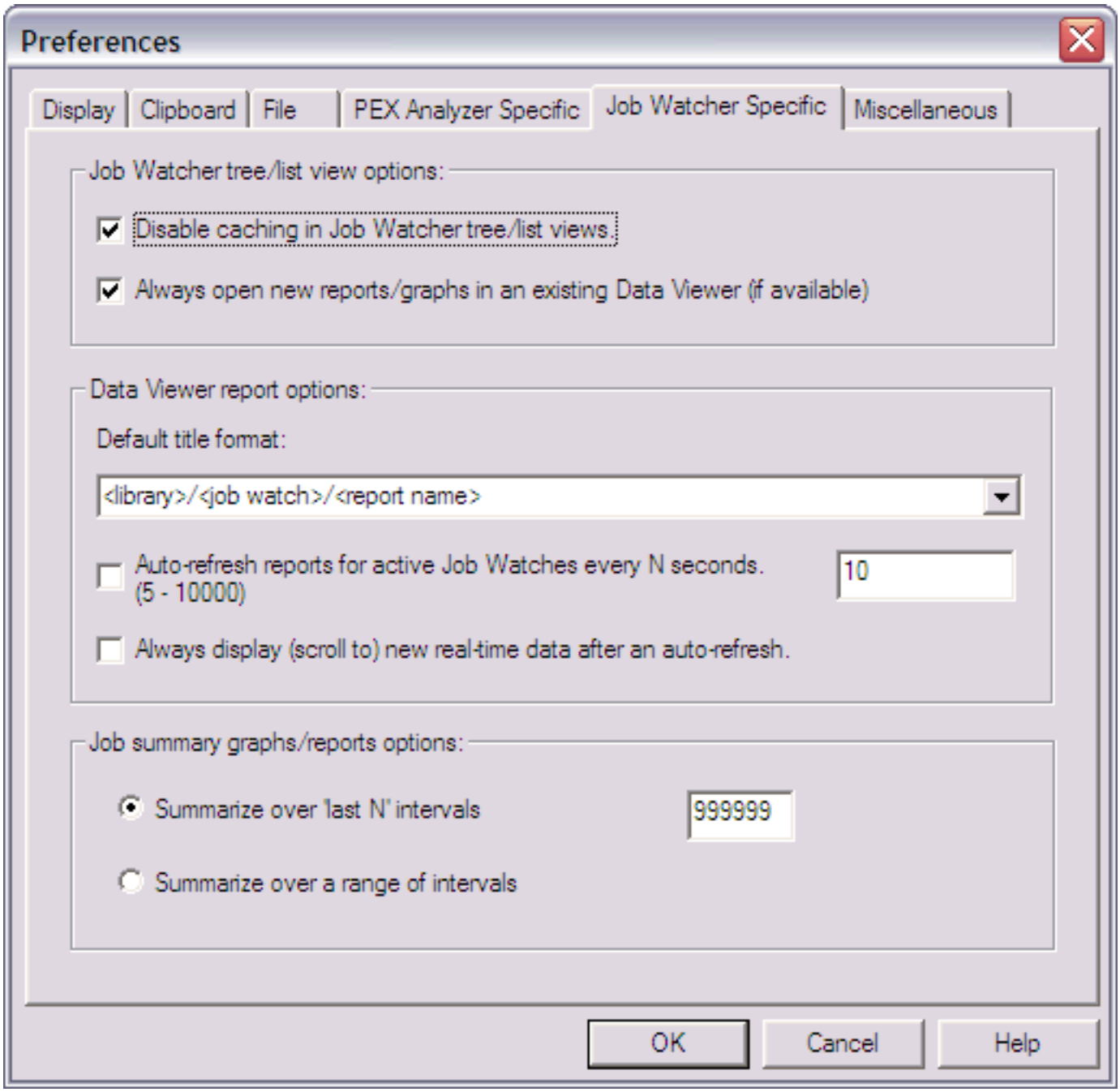
Option	Description
Disable caching	Turn caching on or off in the PEX Analyzer component view. If caching is on and a folder in the tree is clicked that has already been previously clicked it will NOT go to the server to get the latest information. However, this feature can be useful if the system or connection is slow and it's only desired to get data when the user manually refreshes the screen.
Default Title Format	Use this option to identify how the titles of PEX Analyzer reports should be named. The dropdown list contains several different possible name formats. Other possible titles are available by modifying the value in the list and including any of the tabs listed below in <>. <library> - Library name for the collection <collection> - Collection name <file> - The filename for the report. Each PEX Analyzer report is typically over a single file.width=600 <report name> - The long report description. <analysis description> - The analysis description. <analysis member> - The member name for all report files for this analysis. This member name will also match the member of the primary file you are viewing, unless you have changed it using the Query Definition -> Member selection page.
Indent call level with	This option effects the Call Level field for a Statistical hierarchical report (table view). The call level will be indented with the character selected in the drop-down list.
Default job queue name	Indicates the value to use for the default job queue advanced option in the Start PEX Collection Wizard.
Default job queue library	Indicates the value to use for the default job queue library name advanced option in the Start PEX Collection Wizard.

Table of Contents | Previous | Next

1.9.5 Job Watcher

The Job Watcher - Specific page on the Preferences window lets the user work with options only related to the Job Watcher component of iDoctor for iSeries.

An example of this interface is shown below:



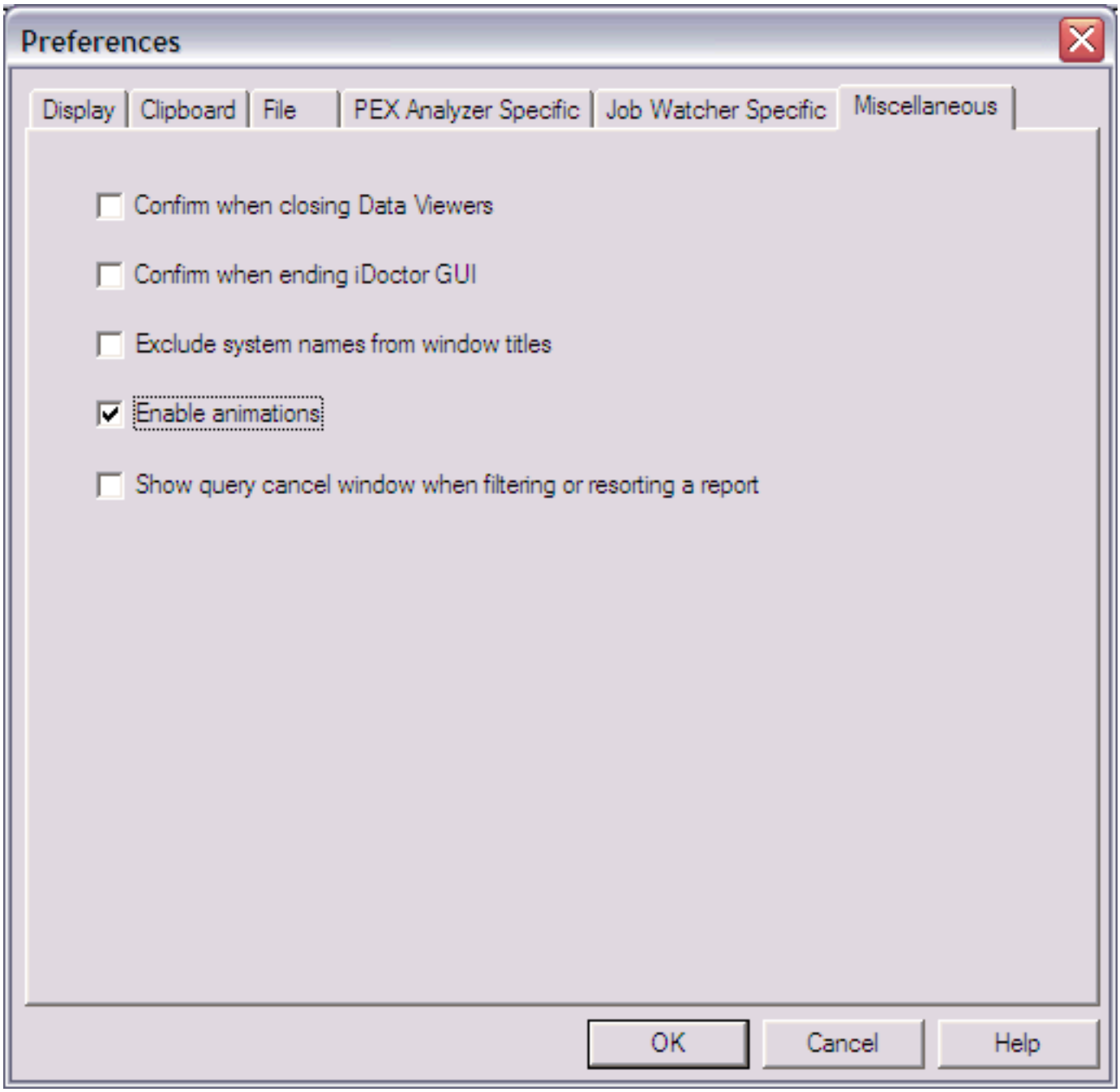
The options available on this page are summarized in the tables below:

Options	Description
Disable caching in tree/list views	Turn caching on or off in the Job Watcher component view. If caching is on and a folder in the tree is clicked that has already been previously clicked it will NOT go to the server to get the latest information. However, this feature can be useful if the system or connection is slow and it's only desired to get data on a "refresh (F5)".
Always open new reports/graphs in an existing Data Viewer	This option lets the user choose to open every Job Watcher report into an existing Data Viewer or into a new Data Viewer.
Default title format	Use this option to identify how Job Watcher reports should be named. The dropdown list contains several different possible name formats. Other possible titles are available by modifying the value in the list and including any of the tabs listed below in <>. <library> - Library name for the watch <job watch> - Job Watch name <report name> - The long report description.
Auto refresh reports for active Job Watches every N seconds	This option lets the user specify how often to auto refresh reports in the Data Viewer that are over currently active Job Watches. **Note:** only the report with the current focus will get refreshed every N seconds.
Always scroll to new data after refresh	This option indicates that after an auto refresh occurs the scrollbar should be adjusted to scroll to the end of the table or graph. This can be useful if new data is consistently being added to the end of the report.
Summarize over a range or last N intervals	Indicates for the Job Watcher job summary graphs and reports, how many intervals should be included in the report. Either the last N intervals or a range of intervals may be specified. This option does not apply to summary graphs/reports by interval, only summarizations done by job. The maximum number of intervals that can be summarized is 999,999. The job summary queries will take much longer to run if this much data is summarized however vs a smaller amount.

Table of Contents | Previous | Next

1.9.6 Miscellaneous

This page contains a set of preferences to control some miscellanous features.



Options	Description

Confirm when closing Data Viewers	Indicates if the user should be warned before closing a Data Viewer. If unchecked and a Data Viewer is closed all views within it are shut down without confirmation.
Confirm when ending iDoctor GUI	Indicates if the user should be warned before closing the iDoctor application. If unchecked and all Data Viewers and views within them are shut down without confirmation.
Exclude system names from window titles	This option is used to remove customer or IBM sensitive system names from the window titles. This is useful when creating presentations or documentation!
Enable animations	Indicates if the graphics shown within the collection wizards should animate.
Show query cancel window	Indicates if the query cancel window should be shown when resorting a report or adding a filter. If you regularly work with very large collections it's a good idea to check this box.

Table of Contents | Previous | Next

1.10 Set Font Dialog

Another feature of the Data Viewer is the ability to customize the font used in the Table Views. The Set Font dialog provides the user with this flexibility. To change the Table View font, right-click on an active Table View and choose the Set Font... menu. The font will be saved for use in future iDoctor for iSeries sessions. In addition to table views this font is used in all tree/list views and list views elsewhere in the application.

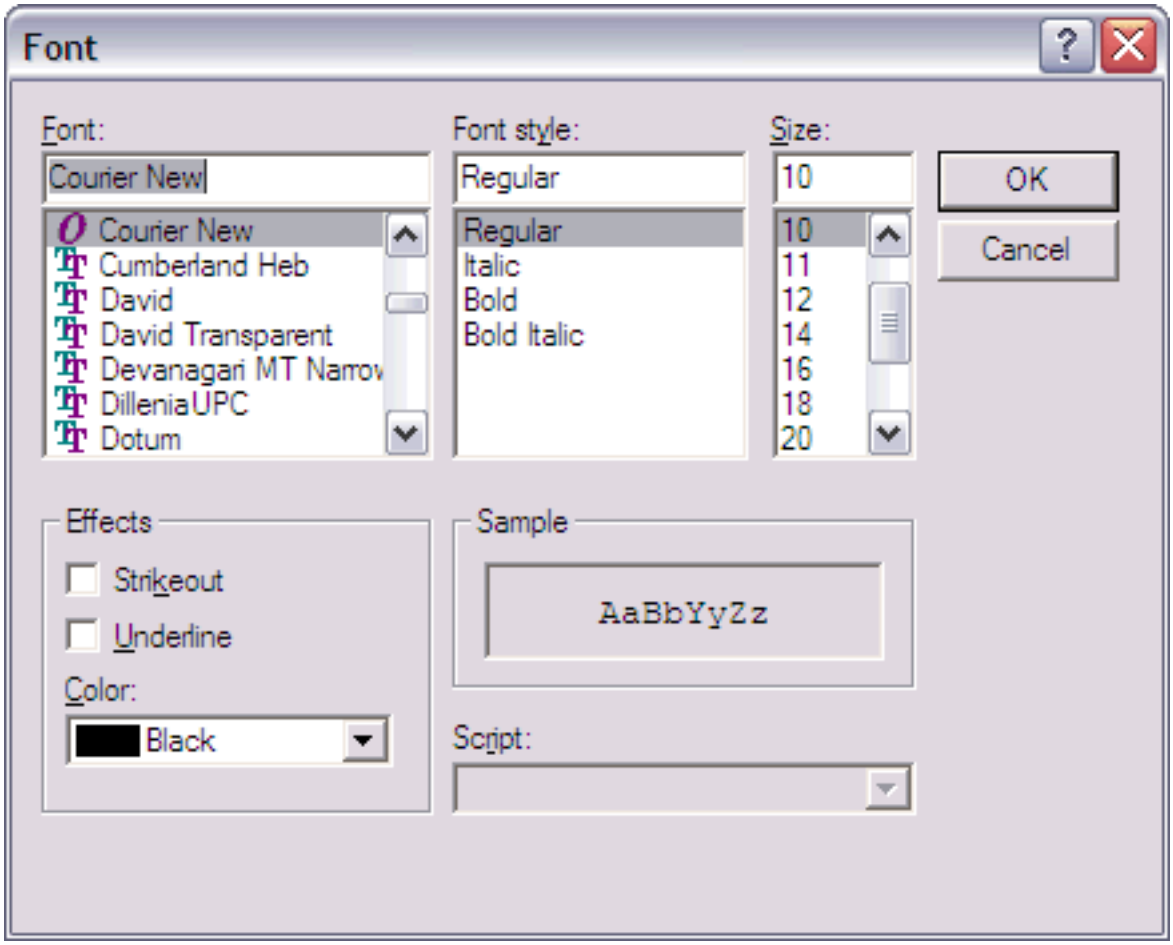


Table of Contents Previous Next

Chapter 2 Job Watcher

This chapter provides an overview of the interfaces within the iDoctor for iSeries - Job Watcher component.

Table of Contents | Previous | Next

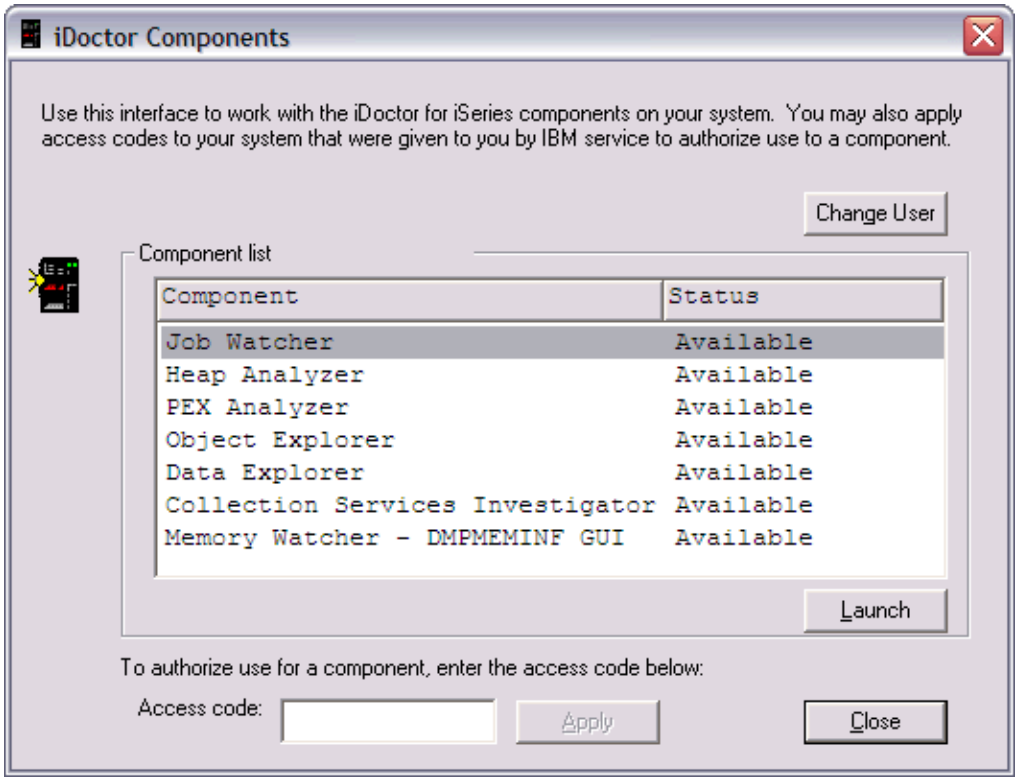
2.1 Job Watcher Basics

The Job Watcher component provides a number of interfaces designed to help the user more easily determine what a job is doing and more importantly why it is not running effectively.

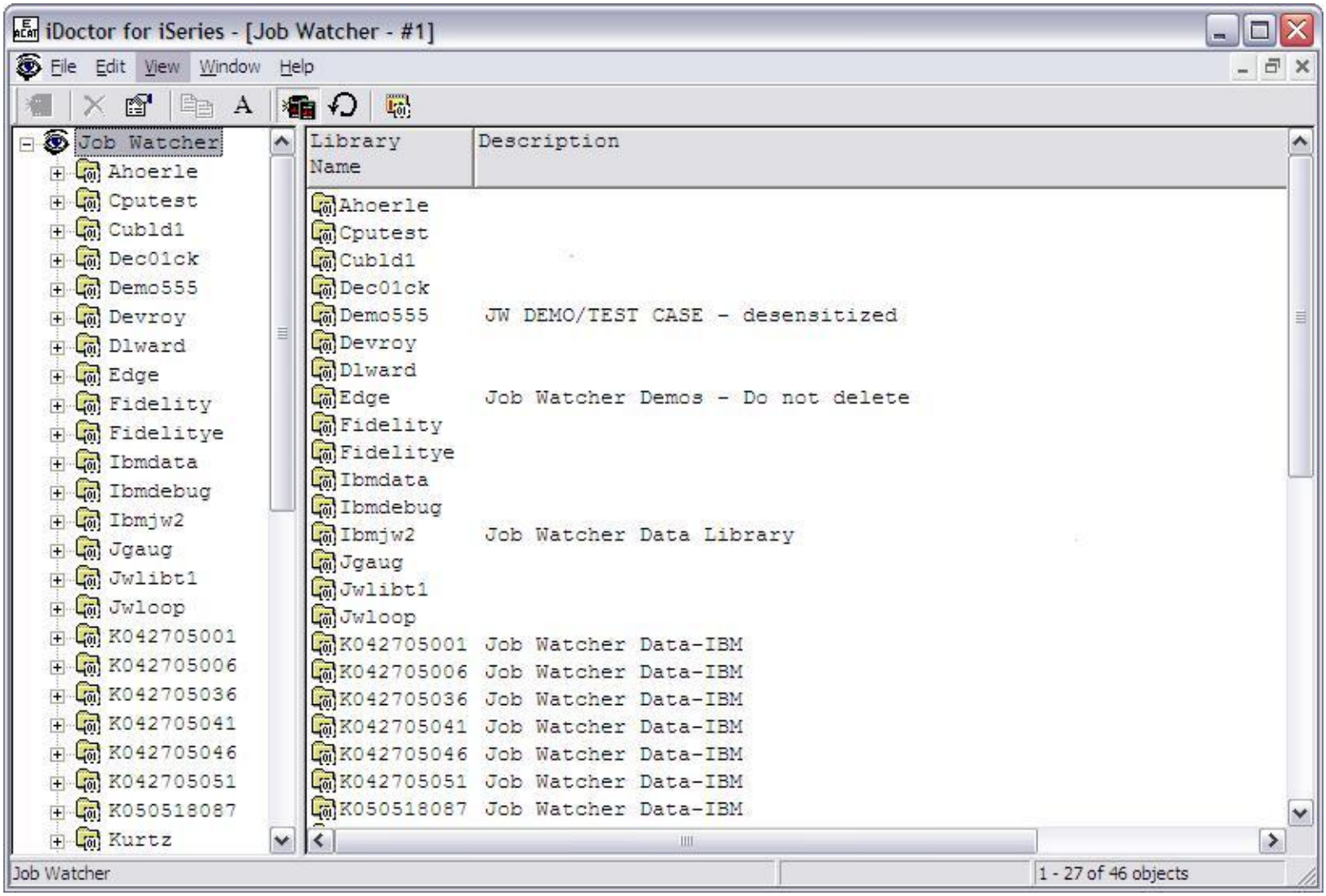
Starting Job Watcher

Job Watcher is a component of the iDoctor for iSeries suite of tools. iDoctor for iSeries can be started using the Start menu: Start->Programs->iDoctor for iSeries. Once the iDoctor for iSeries application appears, the Job Watcher component is started from the Connection List View by double-clicking on the desired system.

A list of available components will appear on the next window. Double-click on the Job Watcher component or select Job Watcher and click the Launch button in order to continue



After clicking the Launch button the following window will appear. This window is called the Job Watcher component view.



The 'Job Watcher' folder contains a list of library folders, each representing a library on the iSeries that contains Job Watcher database files (job watches). The list displays each library's name and description.

Job Watcher Objects

There are four types of objects within the tree/list views of Job Watcher in the following order: **Libraries**, **job watches**, **job watch report folders**, and **reports**. Each of these will covered in more detail in the next sections.

Job Watcher Menu Options

The following menu options are available by right-clicking on the 'Job Watcher' folder in the tree/list view above.

Menu Item	Description
Explore	Displays the contents of the Job Watcher folder (list of libraries on the system containing Job Watcher data) in the right pane of the tree/list window.
Start Job Watch	This menu will open the Start Job Watch Wizard where the user can define and run a Job Watch.
Work with Monitors	Displays the Job Watcher monitors that exist on the current system with a view. From this view new monitors can be started, and existing monitors deleted viewed or restarted.
Open New Data Viewer	Opens a new Data Viewer window. This window is used to display tables and graphs on the system. You can open Job Watcher reports into this window or you can also open any other type of physical file and view as a graph or table.
User-defined reports	This set of menu options allows a user to change the database to use when saving/retrieving user-defined reports or to import definitions into the current user-defined reports database. Definitions can be imported from a server (from library QUSRSYS) or from another user's user-defined reports database.

Properties	Use this menu to display Job Watcher version information installed on the current system.

Table of Contents | Previous | Next

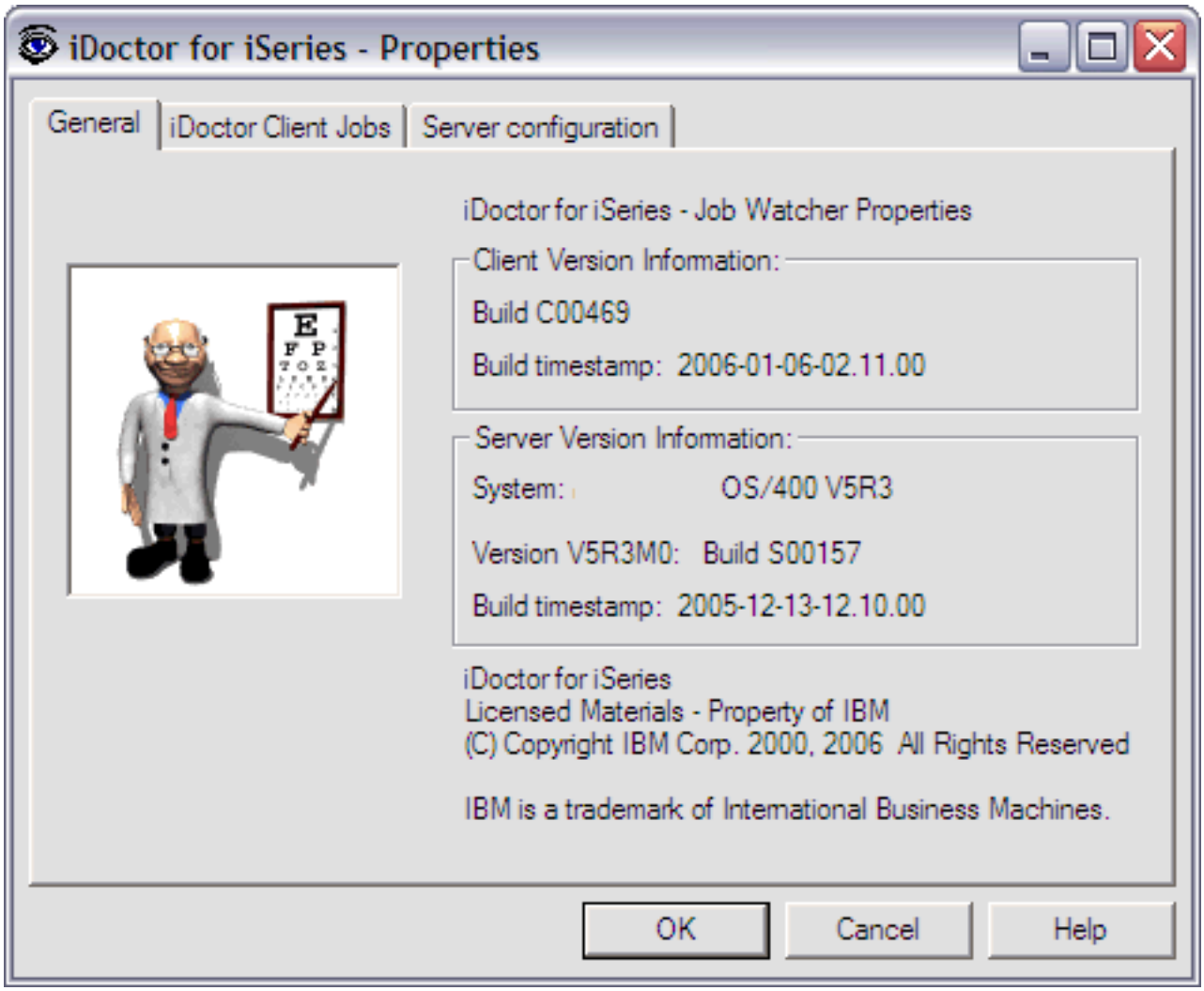
2.2 Job Watcher Component Property Pages

Each component view including Job Watcher has a set of property pages available by right-clicking on the component icon and choosing the Properties... menu. The Job Watcher component property pages contain basic information about the level of Job Watcher installed, as well as the job queue and subsystem that has been configured to run Job Watcher collections on the server.

The next sections cover the different property pages for the Job Watcher component.

Table of Contents | Previous | Next

2.2.1 General



This page displays information about the version of Job Watcher installed on both the client and the server.
The following information is supplied within the General page of this window:

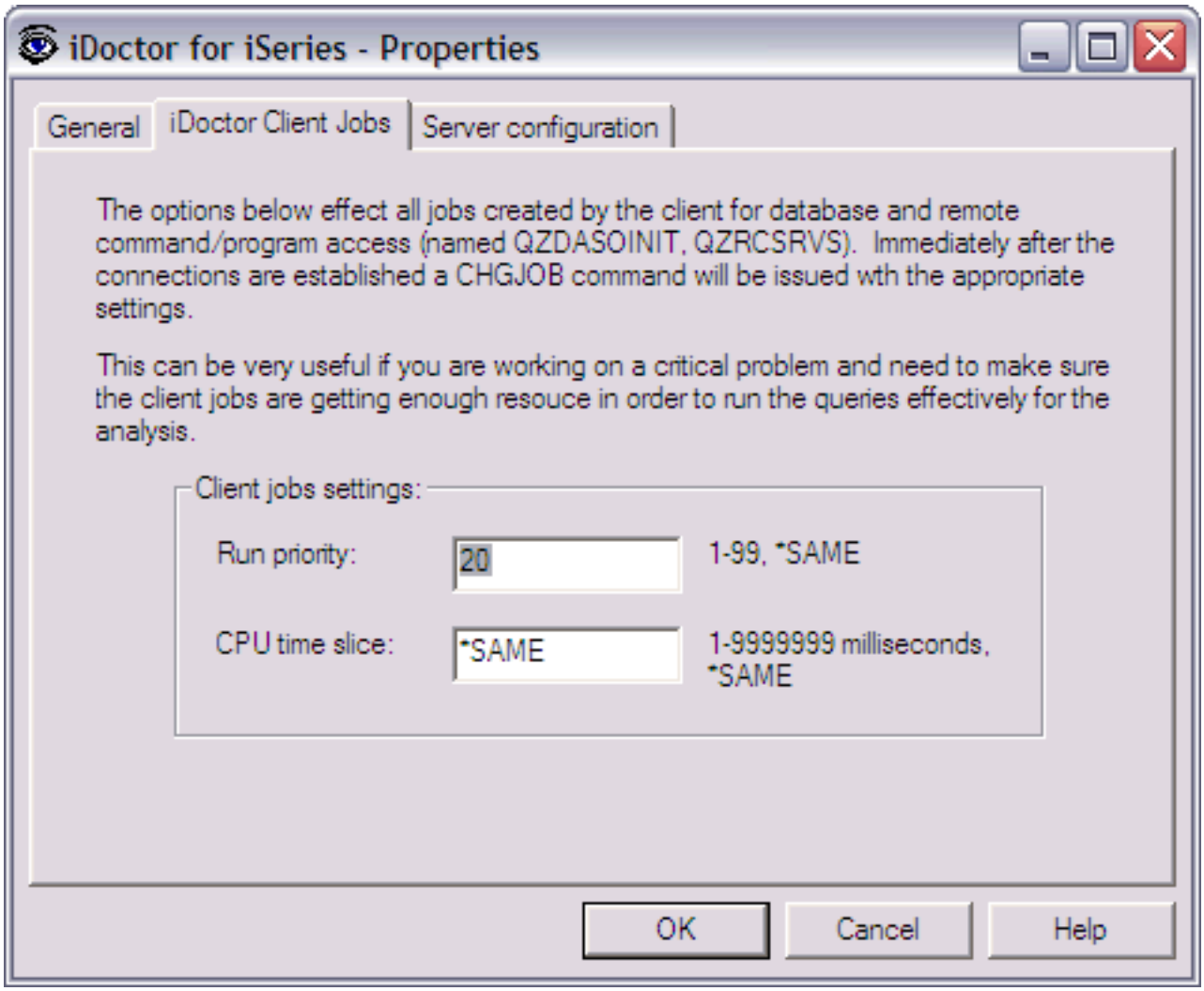
Client Version Information	Description

Build	The build number of the client your are currently running. This number goes up everytime a new build is run for internal (IBM only) testing or for customer use. There will be "internal only" builds resulting in perceived gaps in the build number sequence for customers, but please disregard this.
Build timestamp	The date/time the client build was produced. This value is shown in yyyy-mm-dd-hh.mm.ss format.

Server Version Information	**Description**
System name	The system that the current component view is connected to.
OS/400	The version and release of OS/400 on the active system.
Version	The version and release of Job Watcher on the system. For Job Watcher running at release V5R3 this value is V5R3M0.
Build	Build number of the component installed on the server side
Build timestamp	The date/time the server build was produced. This value is shown in yyyy-mm-dd-hh.mm.ss format.

Table of Contents | Previous | Next

2.2.2 iDoctor Client Jobs

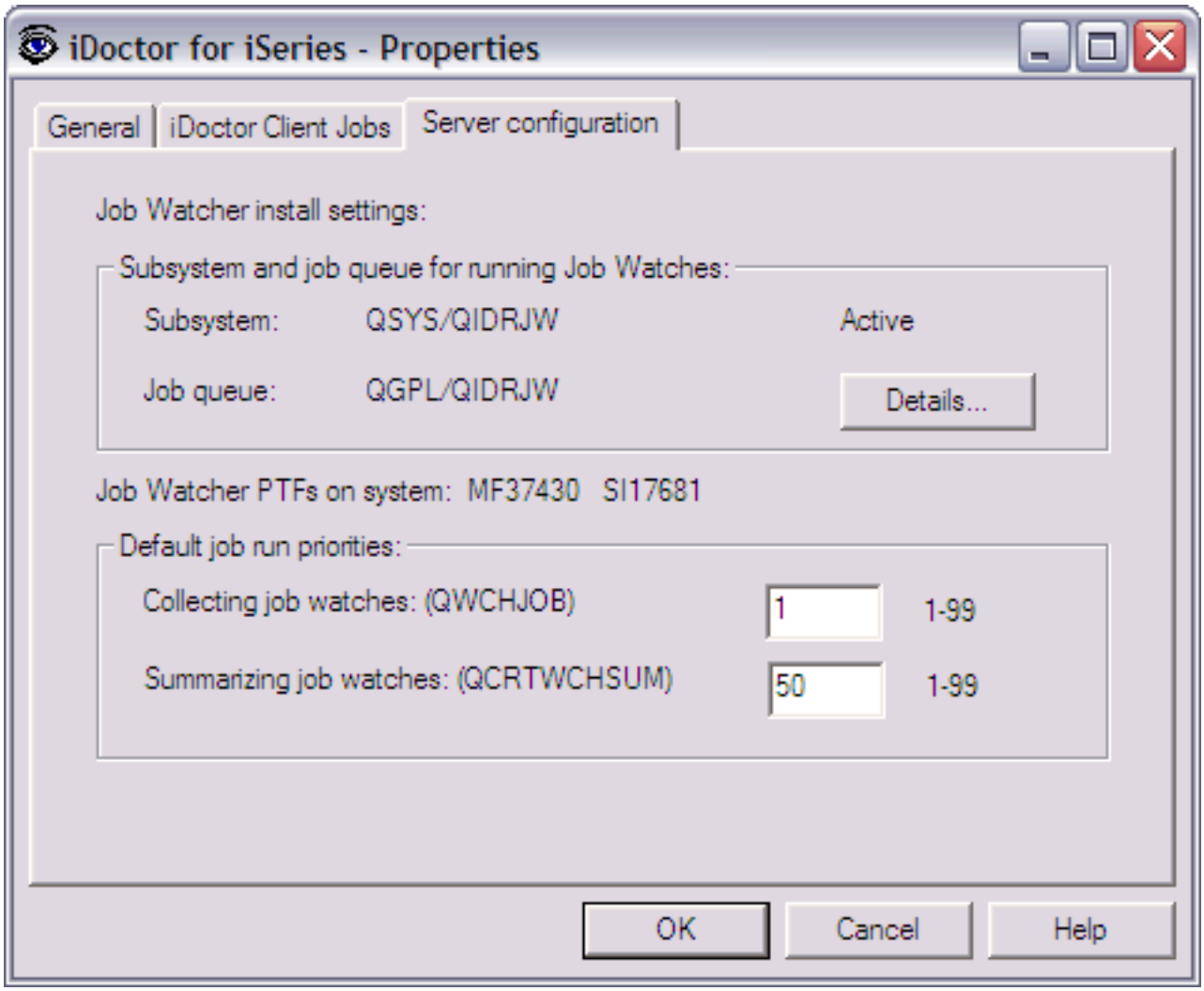


The following information is supplied within the iDoctor Client Jobs page of this window:

This page lets the user set the run priority and CPU time slice of all iDoctor client jobs. This should only be set by advanced users. You must shut down the client and restart in order for these changes to take effect.

Table of Contents | Previous | Next

2.2.3 Server configuration



[Job Watcher component properties displaying the Server configuration page.]

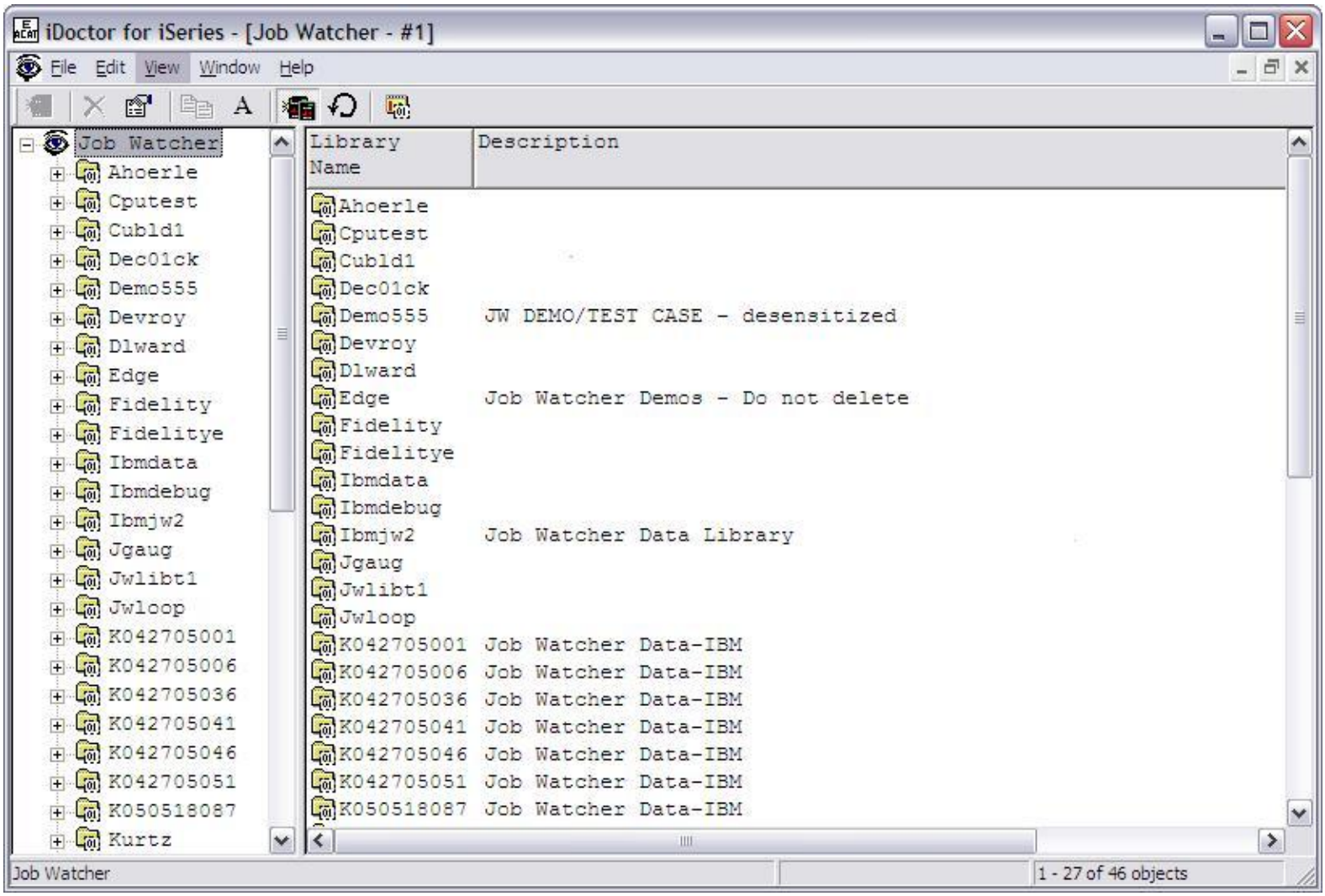
The subsystem and job queue used for running Job Watches is shown on this page. Information about the status of the subsystem and the PTFs installed on the system are also shown on this page.

The run priorities for collecting Job Watcher data and summarizing Job Watcher data can be configured on this page. Doing so modifies the class QIDRBCH (for collecting job watches) or QIDRLOWBCH (for summarizing) in library QIDRWCH. These objects determines the run priority of the jobs when submitted from the GUI on the current system. The numbers shown are retireved from the current system and do not apply to all systems with Job Watcher installed.

Table of Contents | Previous | Next

2.3 Libraries

The 'Job Watcher' folder contains a list of library folders, each representing a library on the iSeries that contains Job Watcher database files. The list displays each library's name and description. By clicking on a library in the tree you will see its contents (a list of job watches).



Library Menu Options

The following menu options are available by right-clicking on a library in the tree/list view above.

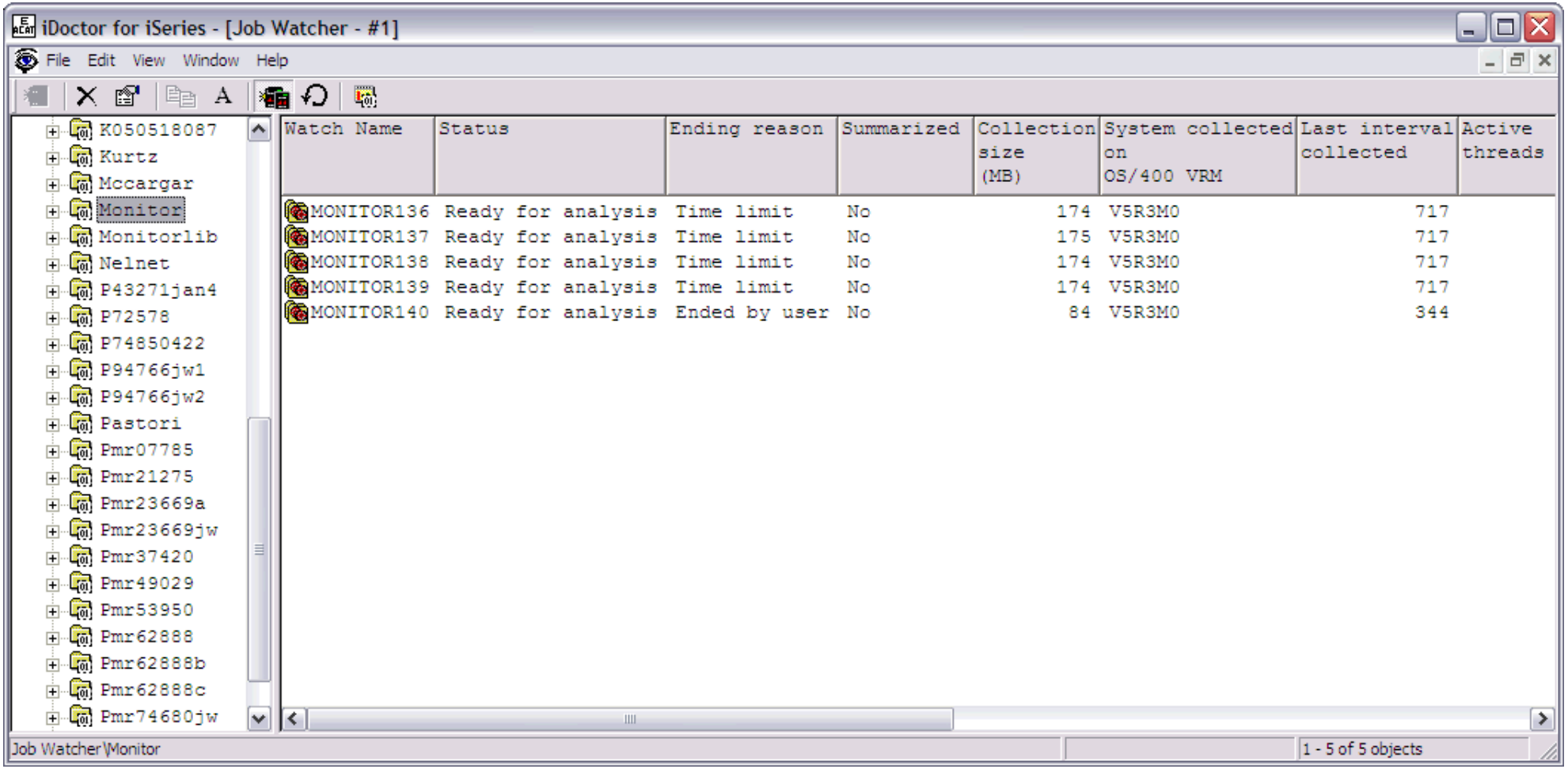
Menu Item	Description
Explore	Displays the contents of the library (list of job watches within the library) in the right pane of the Job Watcher component view.
Select fields...	Brings up a window that lets the user modify what fields are shown when displaying the contents of the library. The contents of a library are Job Watcher collections. This option allows the user to hide/display/reorder fields that are relevant to a Job Watch. This option is only enabled from the tree side of the Job Watcher component view.
Start Job Watch...	This menu will open the Start Job Watch Wizard so the user can create a Job Watch in the selected library.
Copy...	Gives the user the option to copy the library's contents into a new library or into an existing one.

Save...	This option provides the option to save the library's contents into a save file.
Transfer to...	Gives the user the option to transfer a library to another system.
Clear	This option clears a library (deletes all objects in the library).
Delete	Deletes the library.
Rename	Renames the library.
Properties	Use this menu to display the library's property pages. Basic information similar to that provided by DSPOBJD is available through these property pages.

Table of Contents | Previous | Next

2.4 Job Watches

Moving down the tree within each Library folder are one or more job watches that have been created (or are currently being created) within the current library. The green icons indicate active job watches and red icons indicate job watches which have completed. The status field is used to indicate if any errors occurred during collection or the current status of an active job watch.



Job Watch Status

Each watch has a status field indicating whether or not it is currently running. You can also tell the status by the color of the icon: Green = active, Red = not active.

Job Watch Fields

The list of job watches displays the watch name, description, status as well as several additional fields.

Each job watch in the list has a set of fields available which can be optionally reordered and displayed. To change the current field selections for the job watch list, use the Select fields... menu from the library folder. A listing of the available fields and a short description is provided in the table below:

Field	**Description**
Watch name	Name of the job watch. This name matches the member name used in the output files named QAPYJW* that exist on the system.
Status	The status field indicates the status of the job on the system running the job watch (if active) or if not active the status indicates whether or not the watch can be analyzed and/or has data available.
Ending reason	This field indicates what caused the job watch to end.
Summarized	Indicates if the collection has been summarized or not. If this is No, then many of the graphs will take longer to run and may contain an incomplete picture of the data collected. Contributions for jobs that were idle for an interval (used 0 CPU) are not included in the wait graphs. Other possible values are Yes or In progress with a percent complete indicator.
Collection size (MB)	Displays the approximate size of the collection in megabytes.
OS VRM	The version of system that was used to create this job watch. It is possible to view and analyze collections from a previous release through the GUI (to view V5R2 collections on a V5R3 system).
Last interval collected	This value shows the last interval collected. If the job watch is not running, this value indicates the total number of intervals that were collected in the job watch.
Active threads/tasks	The total number of active jobs/threads (meaning used CPU in the last interval) detected in this job watch. This value is only shown when the job watch is actively running.
Description	A description for the job watch specified at creation time.
Start time	The date/time the job watch started..
End time	The date/time the job watch ended (if not active)
Process creating collection	The fully qualified job that created (or is currently creating) the job watch.

Table of Contents | Previous | Next

2.4.1 Menu options

The table below outlines the different types of operations that may be performed by right-clicking on a job watch from the Job Watcher component view.

Menu Item	Description
Explore	Displays the contents of the job watch folder in the right pane of the Job Watcher component view.
Record Quick View	Displays the fields for a Job Watch in the list view vertically for easier viewing. Not available from the tree side, only the list side.
Wait graphs by interval	Contains collection-wide wait summary graphs showing a bar per interval. Each bar is a summary of a particular set of wait times over all jobs for an interval in the job watch. If unsure of where to investigate first, these graphs and the wait graphs by job are a good place to start.
Wait graphs job signatures	These graphs are wait summary graphs showing a bar per job. The jobs are sorted by the type of wait the graph is showing with the job experience the most time spent in the indicated wait at the top of the graph.
Dispatched CPU graphs	This option provides collection-wide summary graphs illustrating CPU and CPU queueing usage by interval.
I/O graphs	This option provides collection-wide summary graphs showing IO operations and disk activity by interval.
I/O graphs by job	This option provides collection-wide summary graphs showing IO operations and disk activity ranked by job.
IFS graphs	This option provides collection-wide summary graphs showing IFS activity by interval.
Other graphs	This option provides collection-wide summary graphs showing other types of information such as state transitions and transactions. Each bar in the graphs represent an interval within the collection.
Search...	Performs a search over the entire collection looking for a specific piece of data specified by the user.

Summarize	Runs the default summary of the collection. This will fill in the idle record wait times and improve performance of many of the graphs by creating an interval summary file.
Copy...	Copies one or more job watches to another library. Selecting multiples is only available from the list side of the Job Watcher component view.
Delete...	Deletes a job watch. Select multiple job watches in order to delete more than one at a time. Selecting multiples is only available from the list side of the Job Watcher component view.
Split	Divides a collection into multiple pieces based on an interval range or a time range. This can be used to focus on a particular set of data or to improve performance of the graphs if the collection is very large.
Transfer to...	FTP one or more job watches to another system. Selecting multiples is only available from the list side of the Job Watcher component view.
Stop	Permanently ends an active job watch by issuing the ENDJOB command for the job that is producing the job watch. Once a job watch is stopped it cannot be restarted again.
Properties	Use this menu to display the property pages for the job watch. The property pages provide quick access to additional information about the job watch.

Table of Contents | Previous | Next

2.4.2 Property Pages

This section covers the property pages for a job watch. Access the property pages by right-clicking on a job watch and choosing the Properties menu.

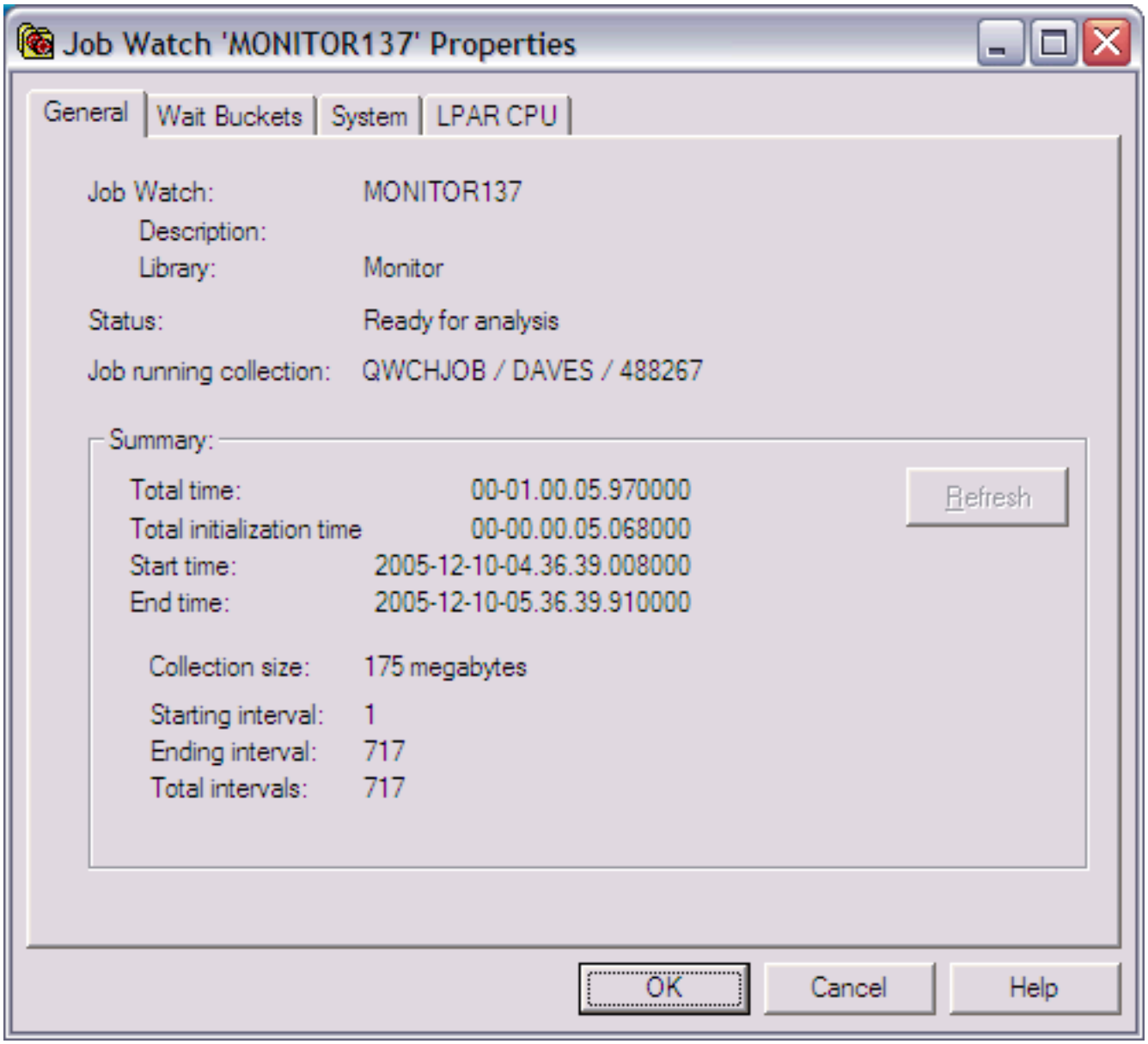
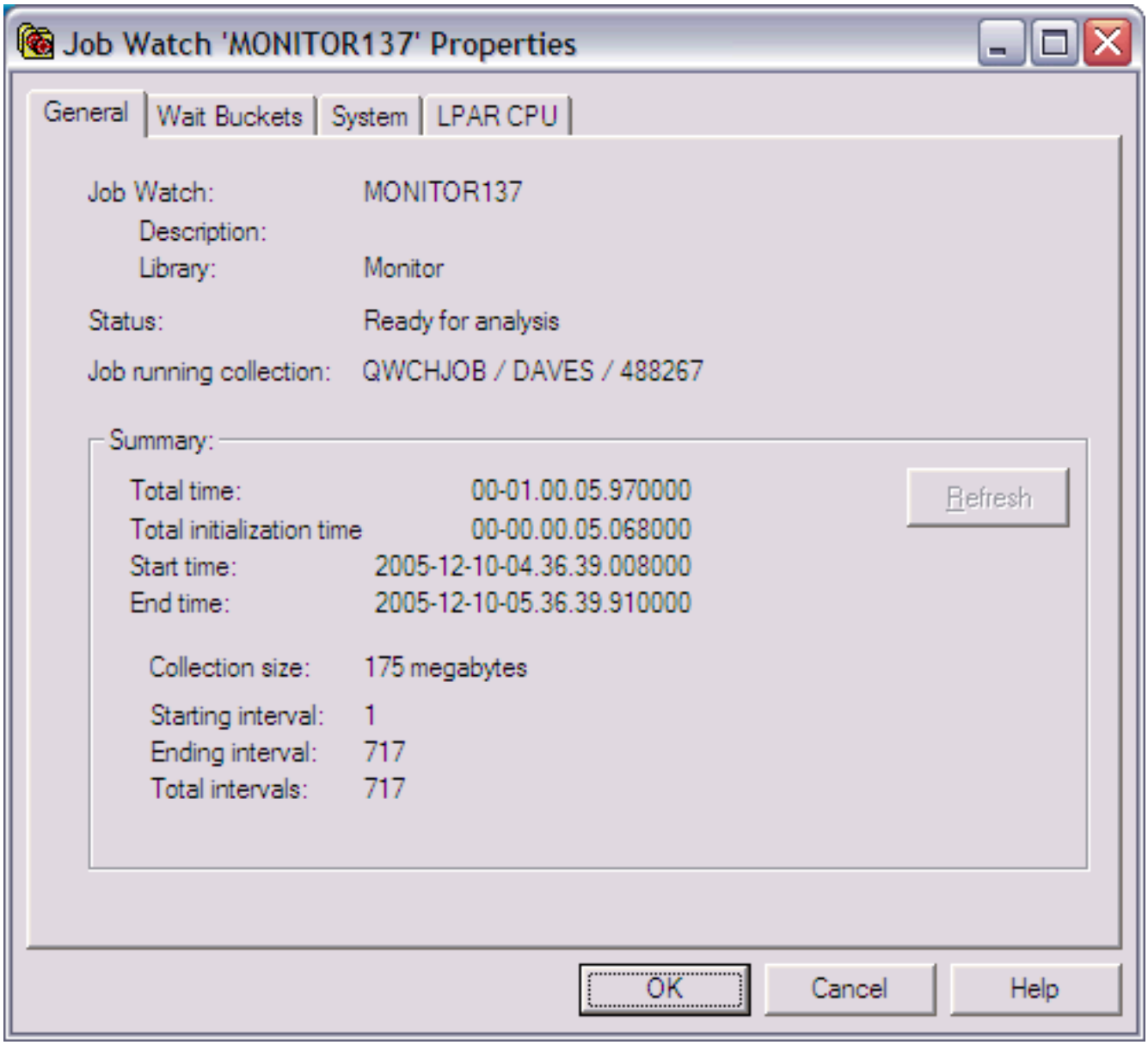


Table of Contents | Previous | Next

2.4.2.1 General

The General property page provides basic information about a job watch.



The following information is displayed on the General property page:

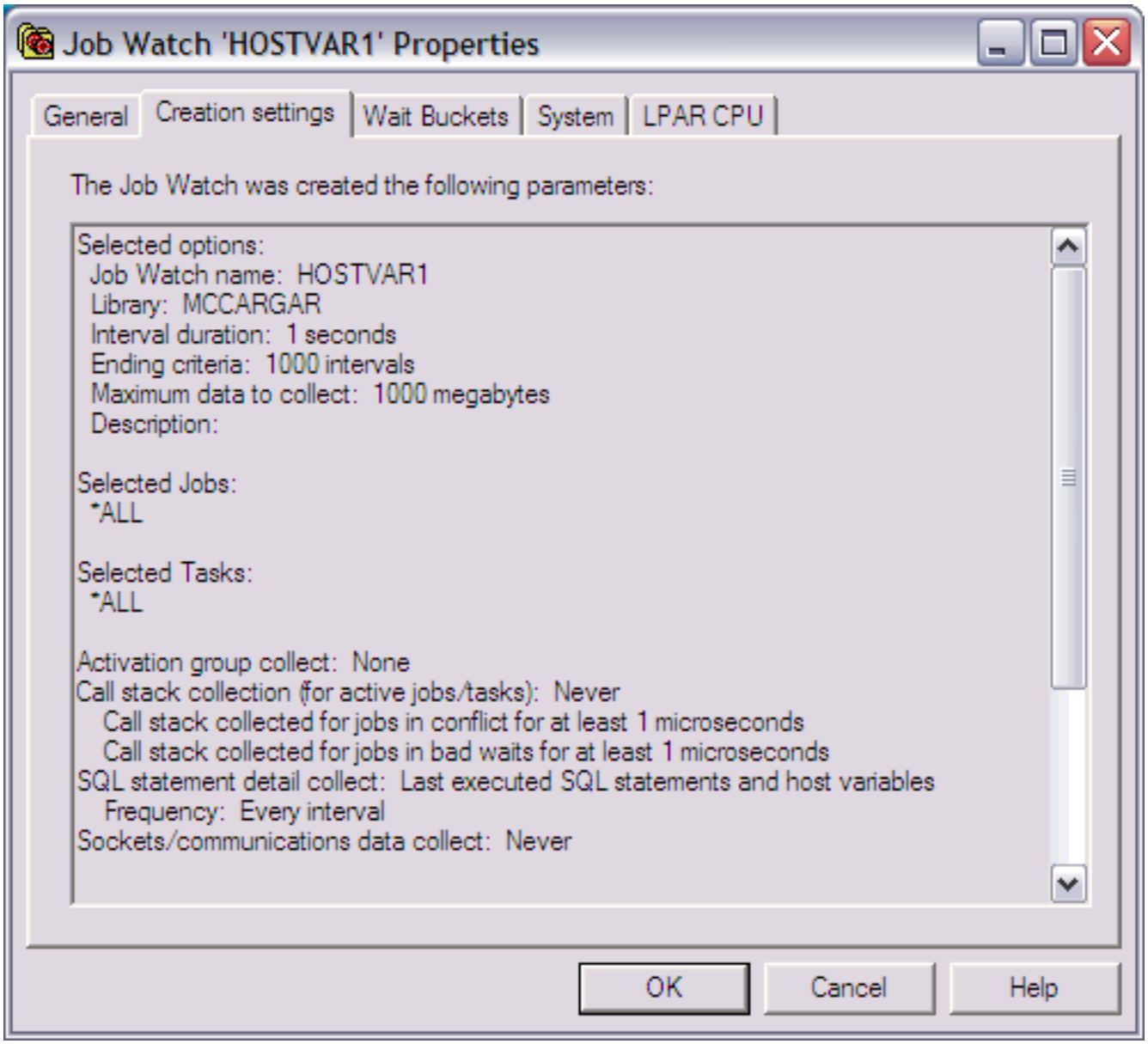
Field Name	Field Description

Job Watch	Name of the job watch. This also matches the member name used in the QAPYJW* files on the server in the library specified.
Description	Description of the job watch set when the job watch was created.
Library	Library the job watch is stored in.
Status	The status of the job watch.
Job running collection	Displays the name of the job that created or is currently creating the collection. If the job log is available a button will be shown to allow it to be displayed.
Total time	Displays the total run time of the collection.
Total initialization time	Displays the estimated initialization time for the collection. This is an estimate of the amount of time it took between the collection being started and the 1st interval of data being collected
Start time	The time the collection was started.
End time	The time the collection ended (if it has ended).
Collection size	The total size of the collection. This number does not include the sizes for Job Watcher summary files. Some of these files can be quite large.
Starting interval	The 1st or lowest interval number detected in the collection.
Ending interval	The last or highest interval number detected in the collection.
Total interval	The total number of intervals found in the collection.

Table of Contents | Previous | Next

2.4.2.2 Creation settings

The Creation settings property page provides details about the parameters on the WCHJOB command that were used to create the Job Watch.



The information shown on this window matches the summary page of the Start Job Watch Wizard when the Job Watch was created.

Note: This page is only shown if the job watch was originally created using the client interface. This window is built using the command string stored in file QAIDRJWDFN which is created by the GUI

when the job watch is submitted.

Table of Contents | Previous | Next

2.4.2.3 Wait buckets

The wait bucket page displays the wait bucket and enums that were used during creation of the collection. These are the building blocks for the job run/wait time signature graphs.

Each specific type of wait is identified by an ENUM number and each enum is given a wait bucket. In Job Watcher we can tell how much time was spent in each wait bucket for each job during each interval. We can also tell what enum (wait) each job was in at the end of interval and how long the job was in that wait (the current wait).

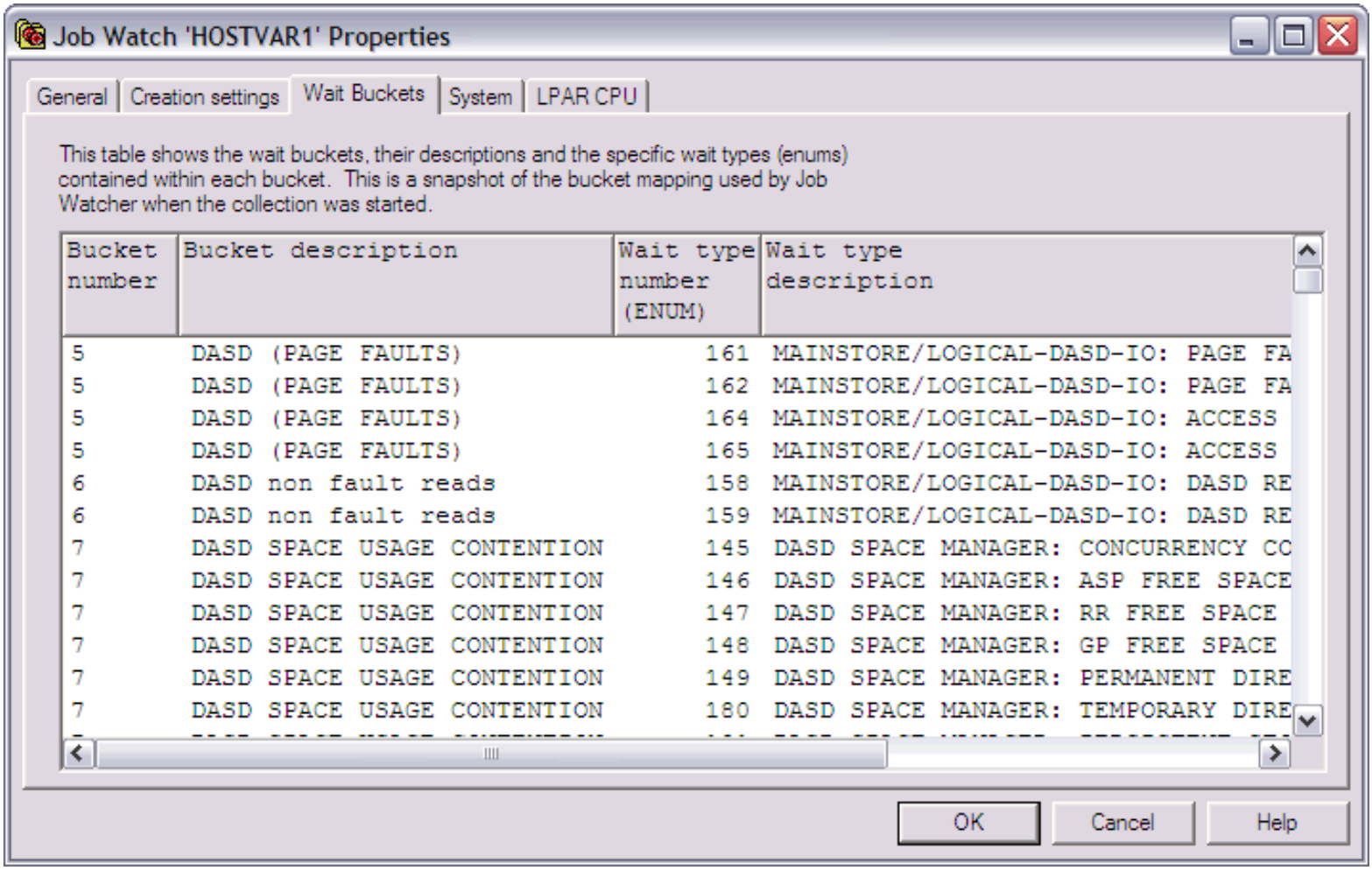


Table of Contents | Previous | Next

2.4.2.4 Rule definition

This page is only available if a rule was defined during creation of the collection.
Details about the rule definition are shown on this page.

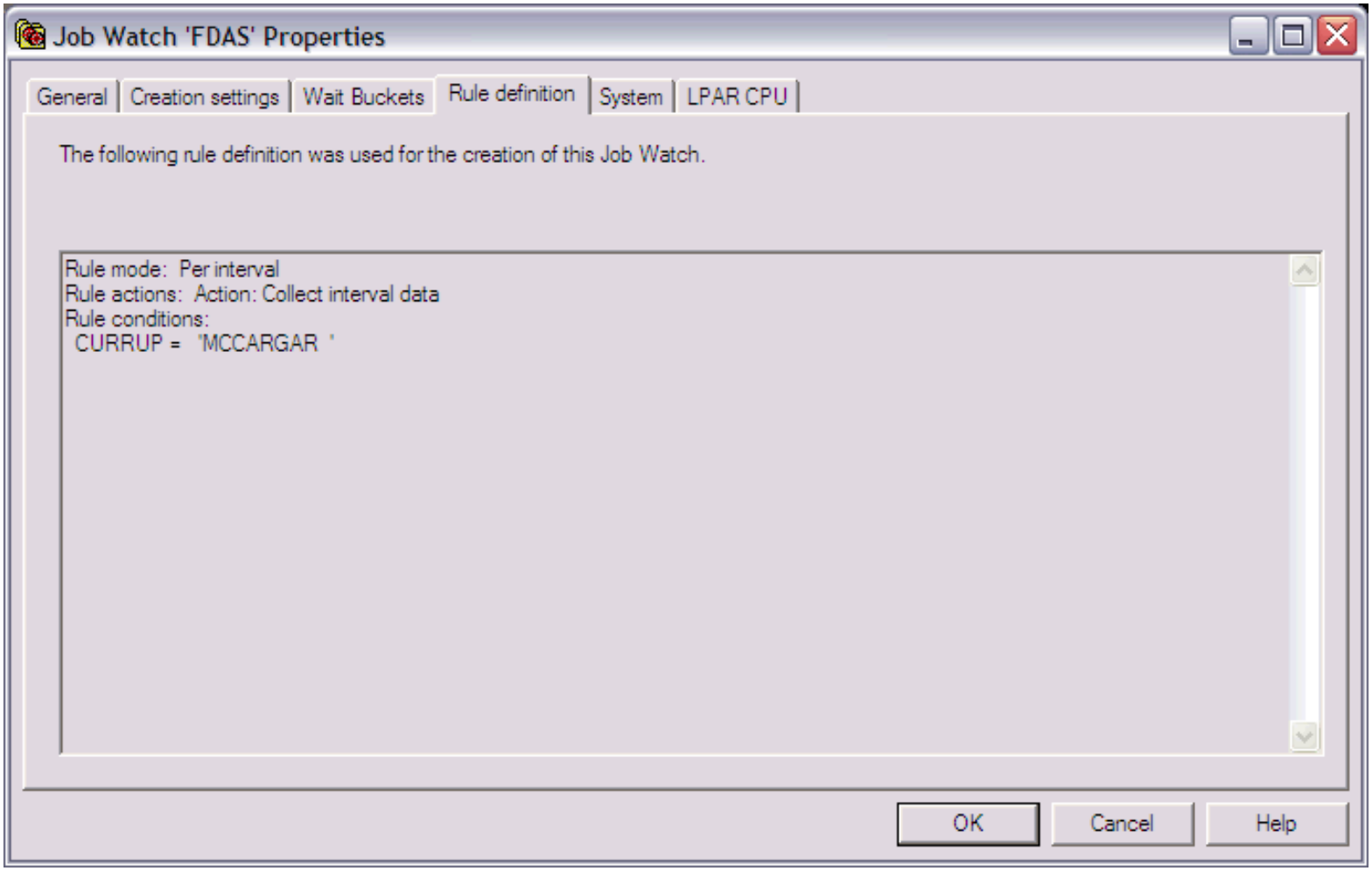


Table of Contents | Previous | Next

2.4.2.5 System

The system property page displays details about the system the collection was created on. This information includes the type, model, operating system VRM and the number of processors.

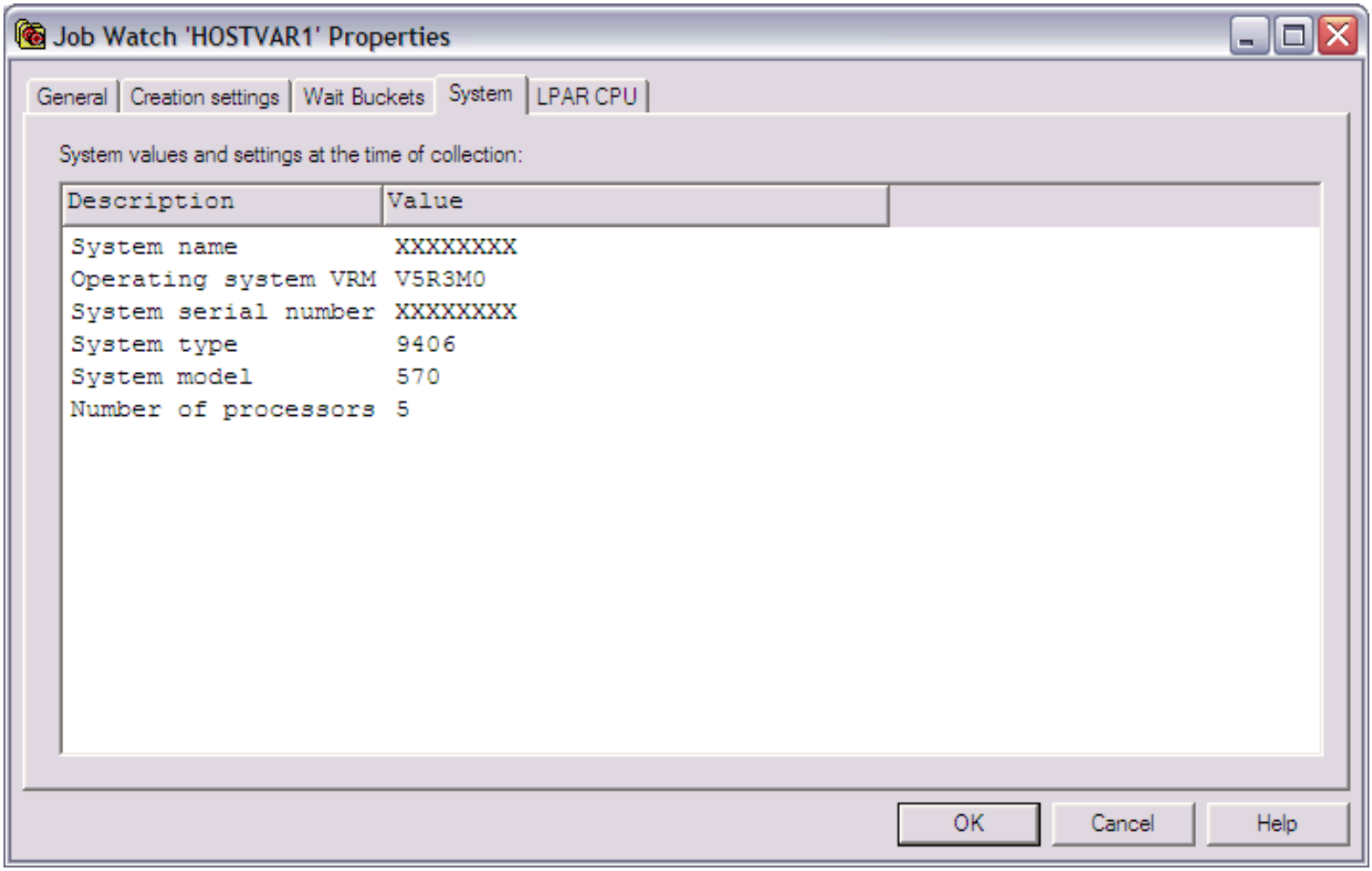


Table of Contents | Previous | Next

2.4.2.6 LPAR CPU

The LPAR CPU property page provides details about the CPU utilization on the system during collection as well as the current processor capacity (CPC) value.

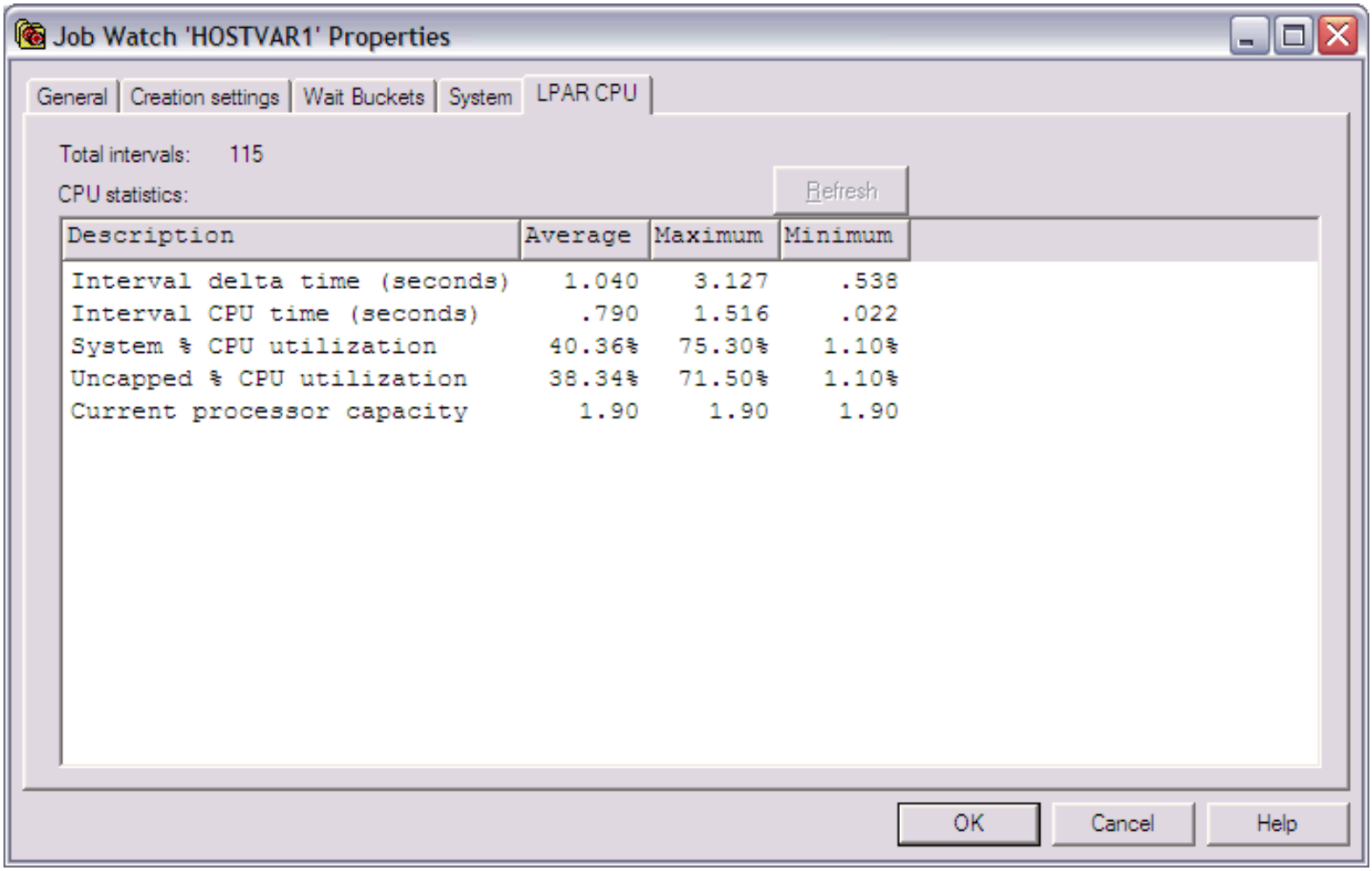


Table of Contents | Previous | Next

2.4.3 Creating - Job Watcher Wizard

Job Watcher provides the capability to collect detailed information about all jobs and tasks on the system.

This section covers the creation of a job watch using the Wizard. To create a new Job Watch use the Start Job Watch Wizard. The Wizard is accessible via the Start Job Watch menu on the Job Watcher or library folder icons. The Job Watch Wizard guides the user step by step through the process of creating the Job Watch. Each page is covered in detailed within the next sections.

An example of the Wizard is shown below:

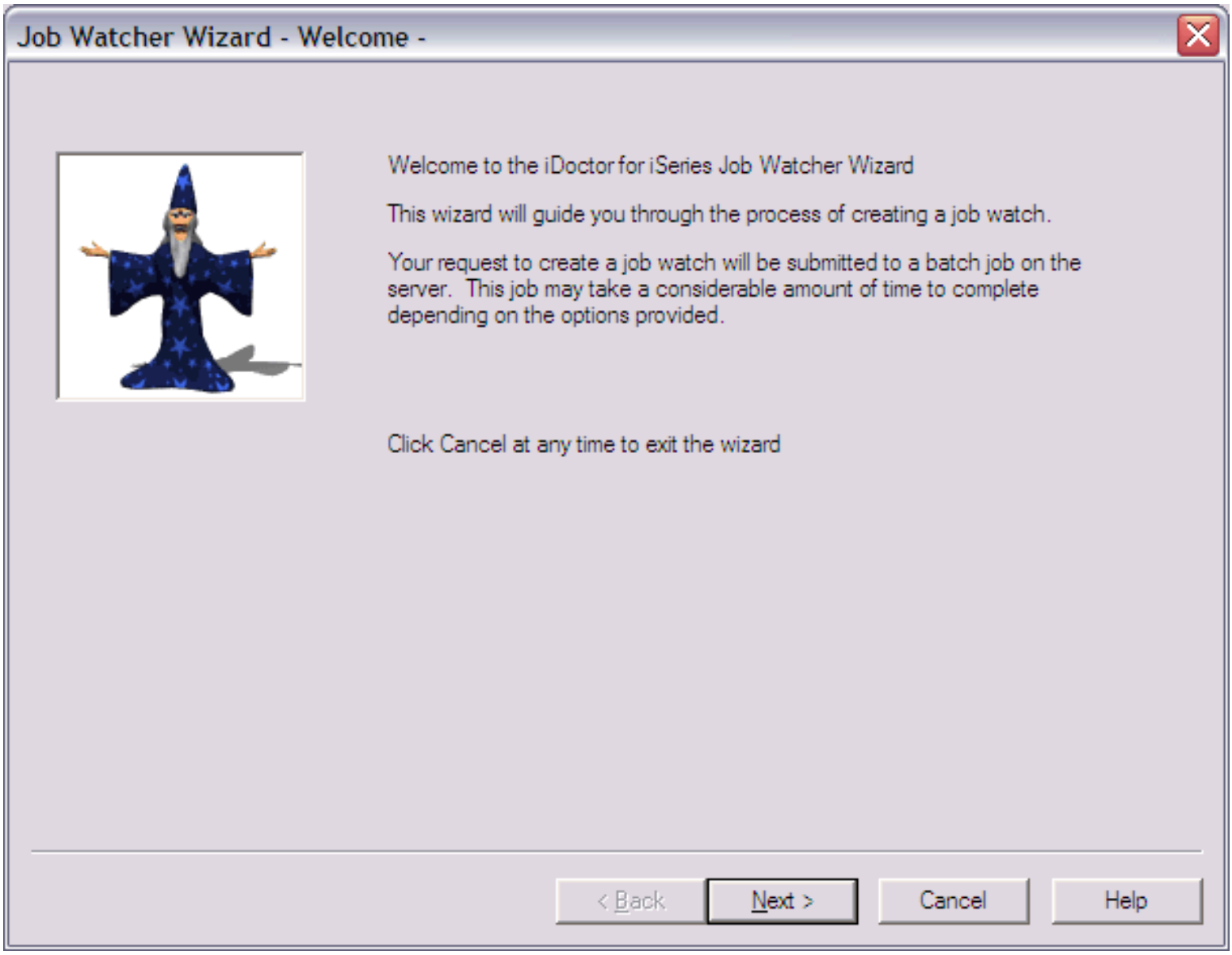


Table of Contents | Previous | Next

2.4.3.1 Welcome

The Welcome page in the Job Watch Wizard introduces the user to the wizard and offers information about what the wizard will do.

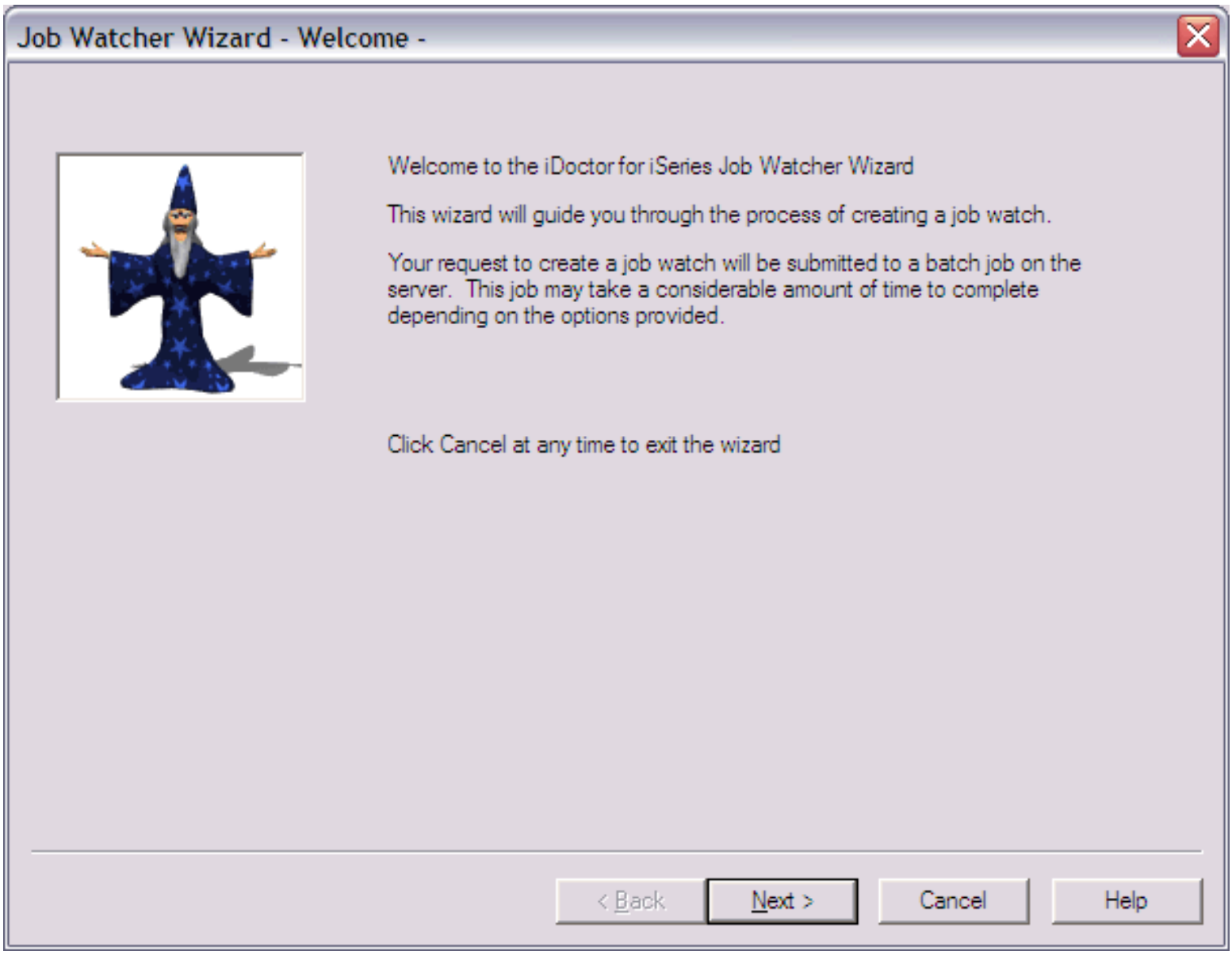
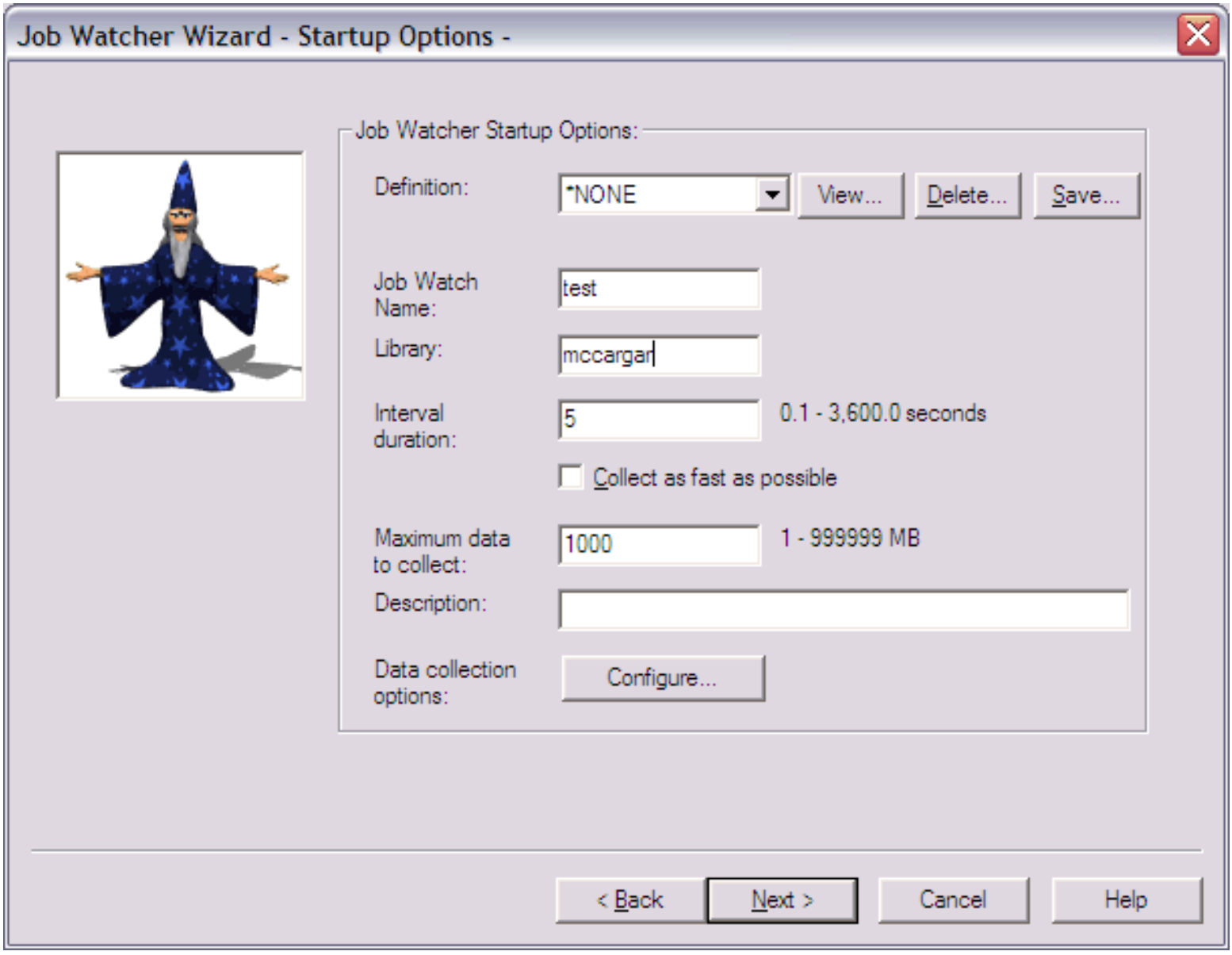


Table of Contents | Previous | Next

2.4.3.2 Options

The Options Page allows the user to specify the most basic pieces of information about a job watch such as its name, library, duration and description. The following is an example of this page of the Job Watcher Wizard.



The following table provides details about each of the parameters on this page:

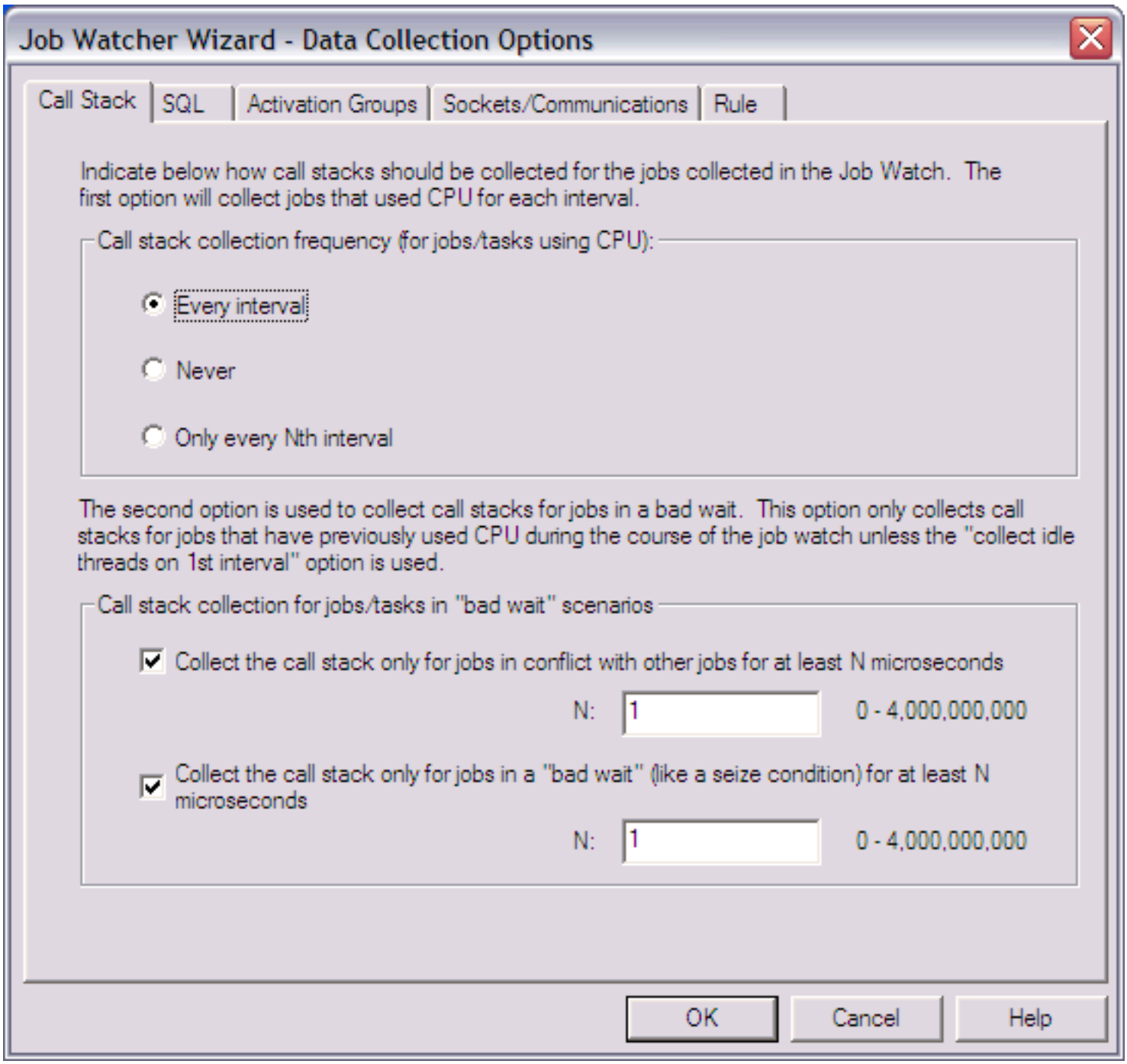
Field	Description

Definition	The name of the definition to create the job watch with. A Job Watcher definition is a set of parameter settings that can be used to more quickly set up and run multiple job watches having the same data collection options. A definition can be created by using the Save As... button on the options page once the desired data collection options have been set on the data collection options section of the Wizard. By selecting a definition from the list of definitions, the parameters for the definition will be loaded into the Wizard for immediate use.
Job Watch	The name of the job watch (10 chars max) to create. This name matches the member name used when creating the output files on the server. The output file names start with QAPYJW* and will exist in the library specified on this page of the Wizard.
Library	The name of the library to create the job watch in. If the library does not exist, the client will ask if it should be created.
Interval duration	The size of each sample of data in seconds. Check the collect as fast as possible button to collect the next snapshot immediately after the previous one finishes (no delay).
Maximum data to collect	This option provides a limit to the total size of the collection (in megabytes). If this limit is reached the collection will automatically end.
Description	A description to give the job watch being created.
Data collection options button	Click this button to select additional types of information to collect such as SQL statements or communications data.

Table of Contents | Previous | Next

2.4.3.3 Data collection options - Call stack page

This page allows the user to specify options for collecting the call stack.



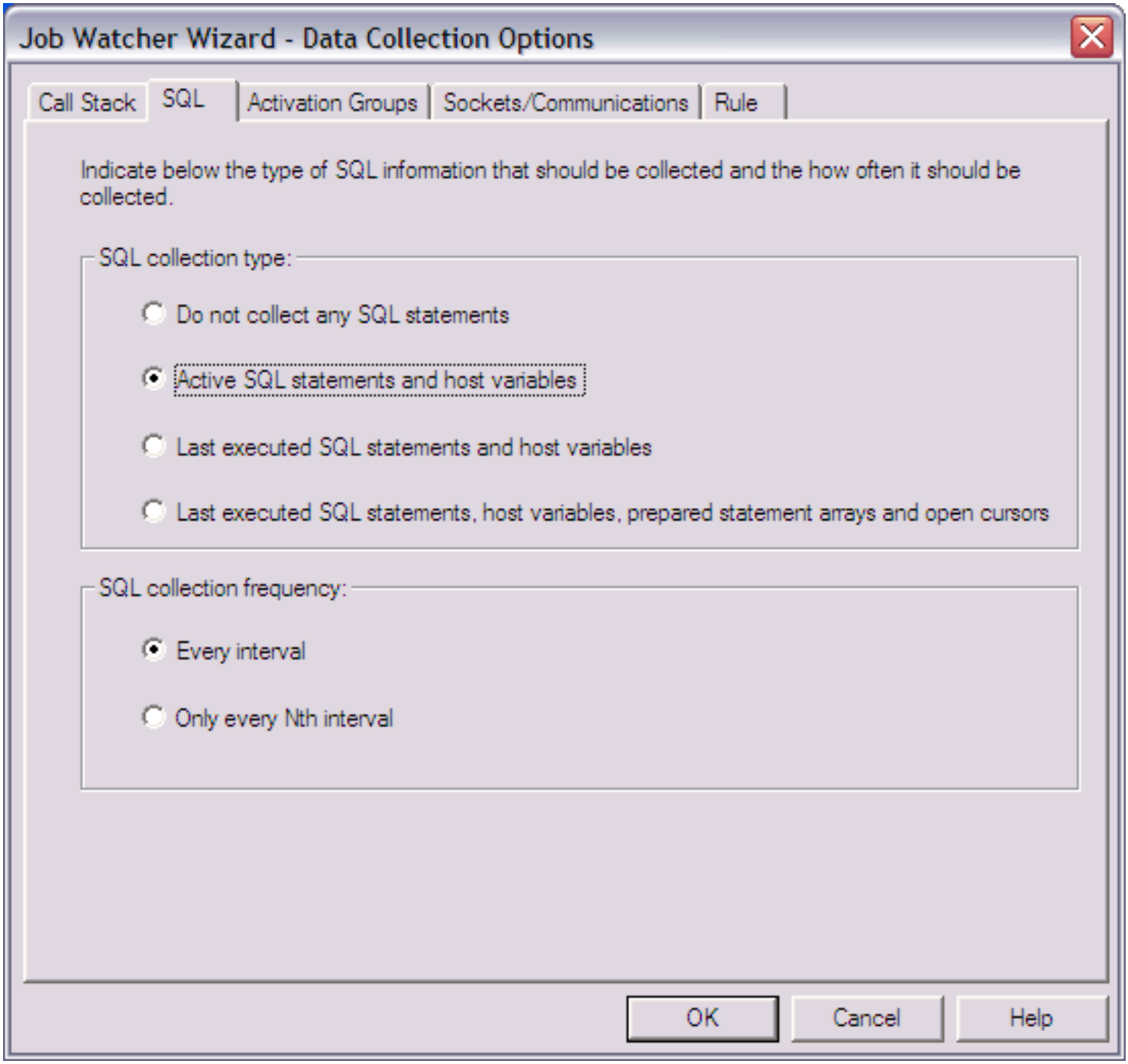
The following table describes the parameters available on this page of the Wizard.

Call Stack Option	Description
Every interval	The call stack will be harvested every interval for every active job in the collection. "Active" jobs are only the jobs that used CPU during each interval.
Never	The call stack will never be harvested during the collection.
Only every Nth interval	The call stack will only be harvested for active jobs every Nth interval. Selecting this option will display a field where the value for N can be entered. "Active" jobs are only the jobs that used CPU during each interval. If the value for N is 5 then only jobs that used CPU every 5th interval of the collection will the call stack be harvested.
Collect call stacks for jobs in conflict	This option indicates if call stacks should be collected for jobs that are in conflict with other jobs. The value N defines how long the the job needs to have been in conflict in order for the call stack to be collected. N is specified in microseconds.
Collect call stacks for jobs in bad waits	This option indicates if call stacks should be collected for jobs that are in bad waits. The value N defines how long the the job needs to have been in a bad wait in order for the call stack to be collected. N is specified in microseconds.

Table of Contents | Previous | Next

2.4.3.4 Data collection options - SQL page

This page allows the user to define the options for collecting SQL statements for jobs included in the job watch.



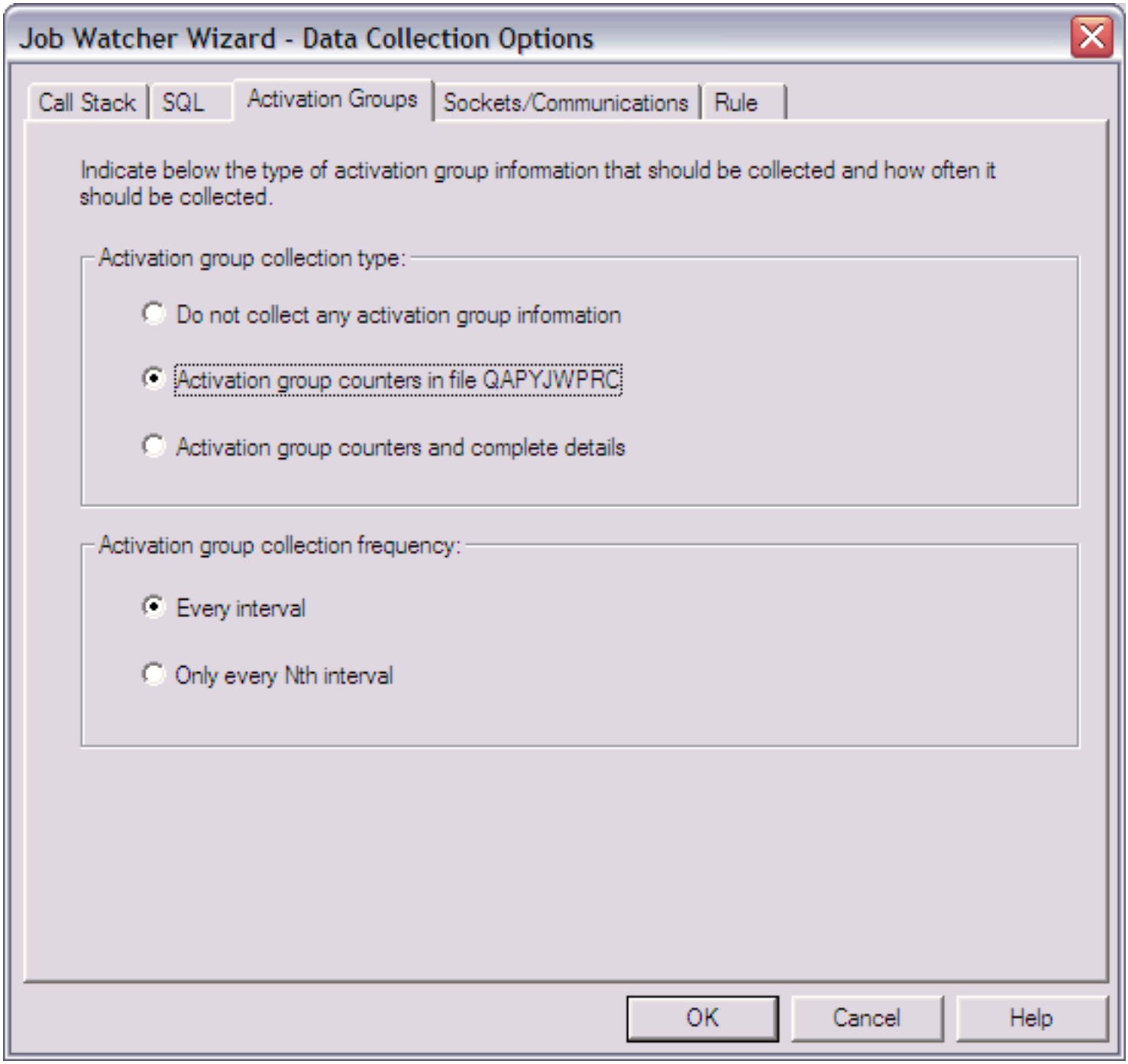
The following table describes the parameters available on this page of the Wizard.

Option	Description
Do not collect any SQL statements	No SQL statements collected
Active SQL statements and host variables	SQL statements will be collected for any jobs that are currently running SQL statements (at the moment each interval is harvested) within the collection. If this option is used it's quite possible not to get any SQL information if the statements that are running complete
Last executed SQL statements and host variables	This option will collect the last executed SQL statement and host variable for every job in the collection, for every interval the job is active.
Last executed SQL statements, host variables, prepared statement arrays...	This option will collect the last executed SQL statement and host variable for every job in the collection, for every interval the job is active. In addition this option will collect information about any prepared statement arrays and open cursors for the job running the SQL statement.
SQL collection frequency	If one of the above SQL collection option is selected, this option allows the user to determine how often the SQL data should be collected.

Table of Contents | Previous | Next

2.4.3.5 Data collection options - Activation groups page

This page allows the user to define the options for collecting activation group information for jobs included in the job watch.



The following table describes the parameters available on this page of the Wizard.

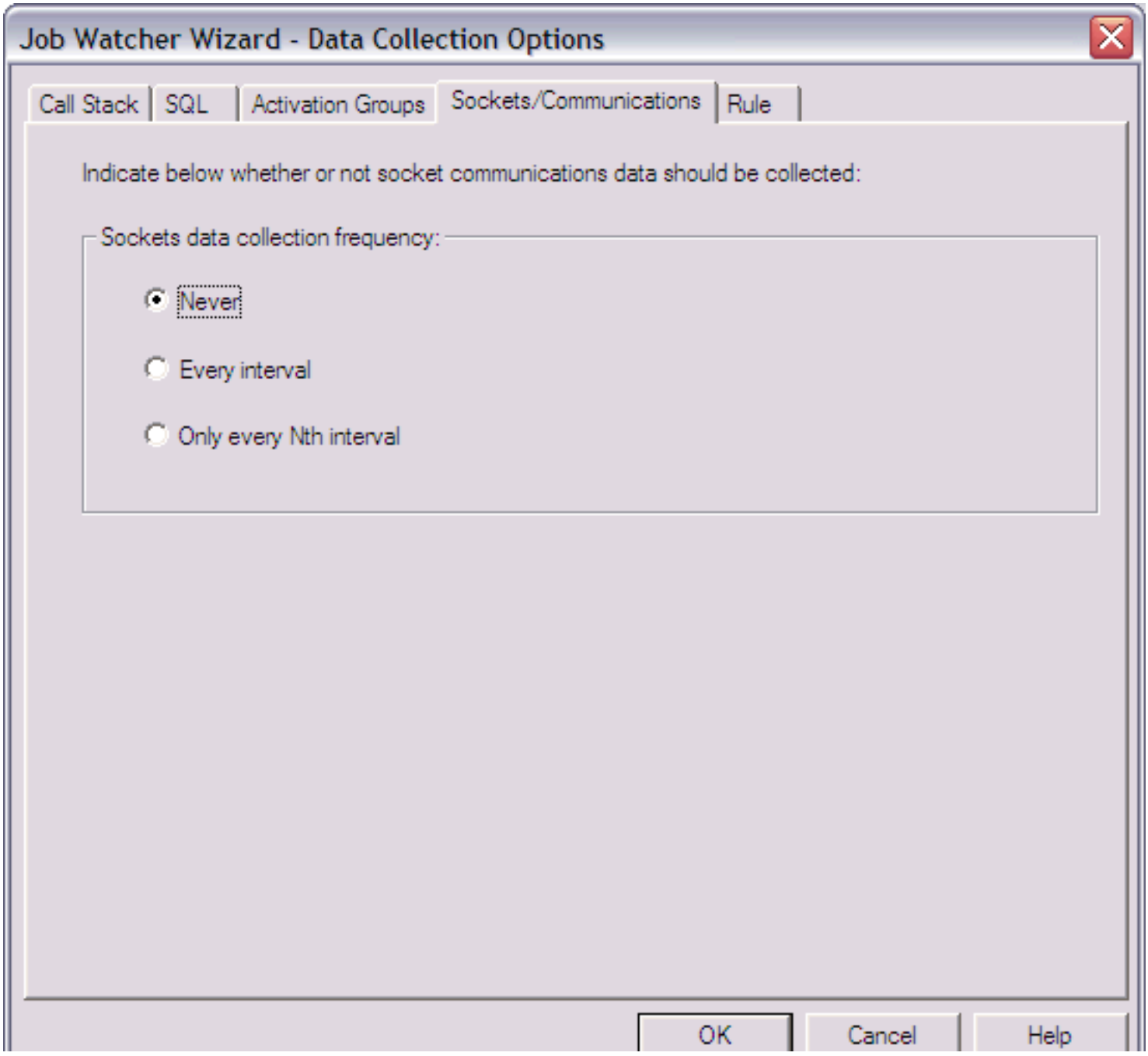
Option	Description
Do not collect any activation group information	No activation group data collected
Activation group counters in file QAPYJWPRC	If this option is selected counters in file QAPYJWPRC (the job/process information file) will be filled. The fields that will be filled are: CURNUMACTG (current number of activation groups) and CURNUMACT (current number of activations)
Activation group counters and complete details	This option will collect the activation group counters in the prior option as well as an additional files containing complete information about the activation groups for all jobs collected in the job watch. The files collected by this option are: QAPYJWAIGP - general activation group information QAPYJWAIHP - activation group heap sizes and counts QAPYJWAIPA - list of programs in each activation group collection
Activation group collection frequency	If one of the above activation group collection option is selected, this option allows the user to determine how often the activation group data should be collected.

Table of Contents | Previous | Next

2.4.3.6 Data collection options - Sockets page

This page allows the user to capture communications and socket information for jobs running in the job watch. Socket data is collected into files QAPYJWSKTC and QAPYJWSKJB.

An example of this window is shown below:



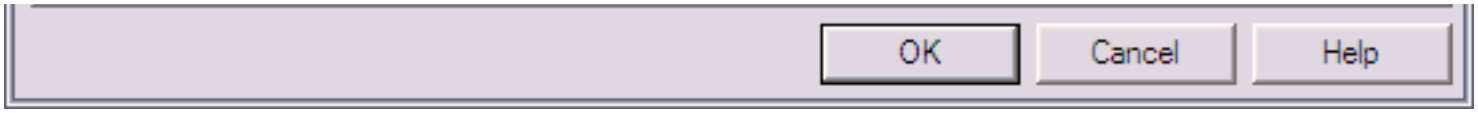
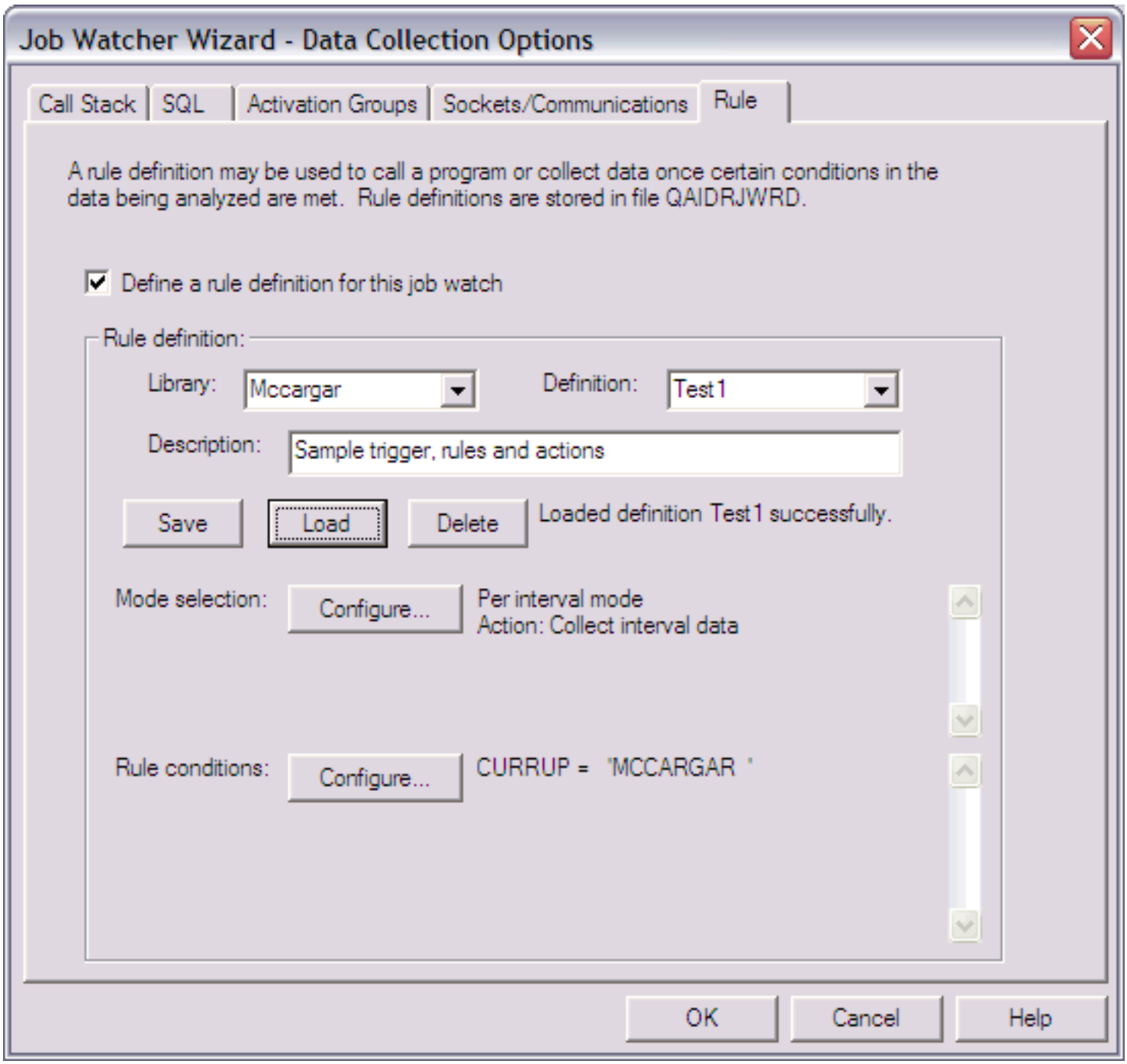

OK
Cancel
Help

Table of Contents | Previous | Next

2.4.3.6a Data collection options - Rule

This page is used to define a rule definition for the collection. A rule definition is used to collect data based on certain criteria over the data encountered during collection. Rule definition are saved into file QAIDRJWRD. An example of creating a rule definition via the green screen is available in file QAIDRJWRD in library QIDRWCH.



Field	Description
Library	List of the libraries found on the system containing existing rule definitions. The value is editable. To save the rule definition into a new library, type the library name into this field before pressing the Save button.
Definition	Within the current library selected, the definition (member) names that were found. The value is editable. To save a new rule definition, provide the name into this field before pressing the Save button.
Description	Description of the rule definition.
Save, Load, Delete	Buttons to save, load or delete a rule definition into the library and definition name specified. Rule definitions are saved into file QAIDRJWRD. When a collection is created that uses a rule definition, a copy of the definition is saved into file QAIDRJWRDB with a member name matching the collection name.
Mode	Displays and configures the type of rule definition.
Rule conditions	Displays and allows configuration of the conditions against the collection data that should be used when evaluation the rule.

Mode Selection

There are two possible modes for rule definitions: per interval and lurk.

Per interval mode is used to allow the collection to only contain data for intervals where the conditions defined are met. In addition a program call can be made each interval the conditions are met.

Lurk mode allows the collection to investigate the collected data without actually storing it until the rule conditions defined are met. Once the conditions are met, a program can be called, data can begin collection or both. In addition historical data can be dumped to the files for a period of time before the conditions were met.

Rule Conditions

Rule conditions are defined within the window below over many of the fields in the Job Watcher data files.

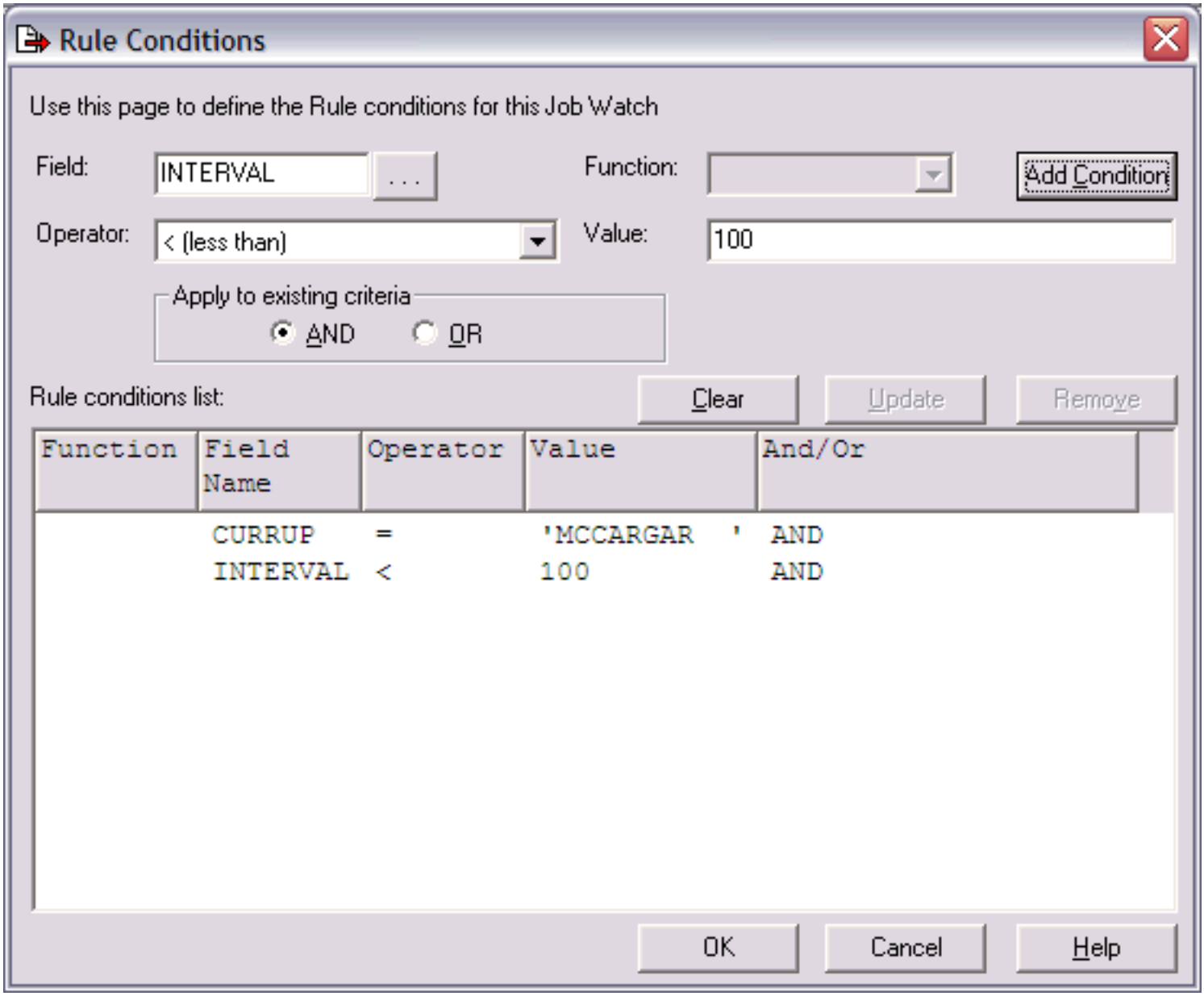
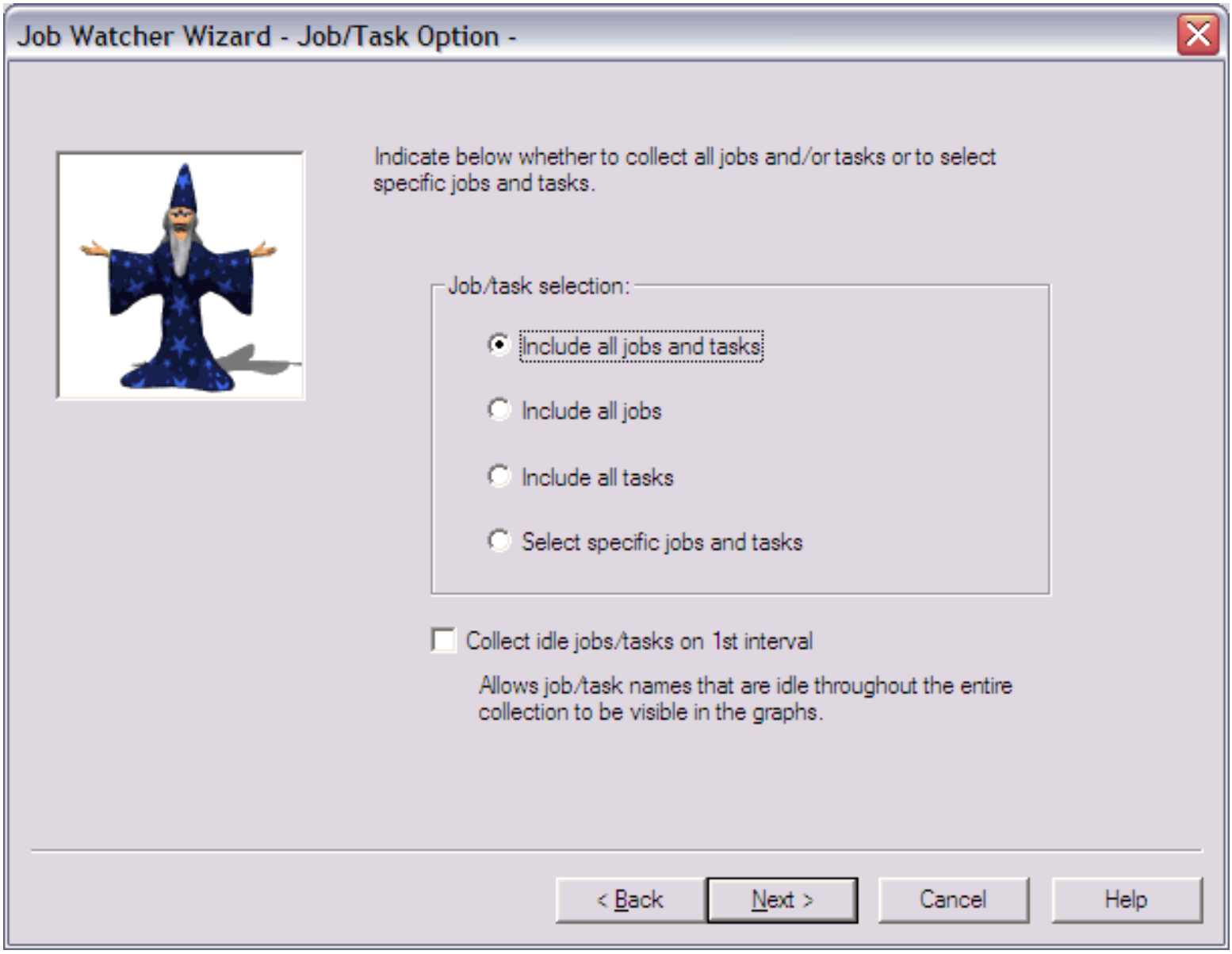

Rule Conditions
Use this page to define the Rule conditions for this Job Watch
Field:
INTERVAL
...
Function:
Add Condition
Operator:
< (less than)
Value:
100
Apply to existing criteria
AND
OR
Rule conditions list:
Clear
Update
Remove
Function
Field Name
Operator
Value
And/Or
CURRUP = 'MCCARGAR ' AND
INTERVAL < 100 AND
OK
Cancel
Help

Table of Contents | Previous | Next

2.4.3.7 Job/task Option

This page allows the user to determine whether all jobs/tasks should be collected, or if specific jobs and tasks should be collected.
If the option "Select specific jobs and tasks" is selected then the job/task selection page will be shown next in order for the user to define which jobs and/or tasks that are running should be collected.

An example of this window is shown below:



The following table describes the parameters available on this page of the Wizard.

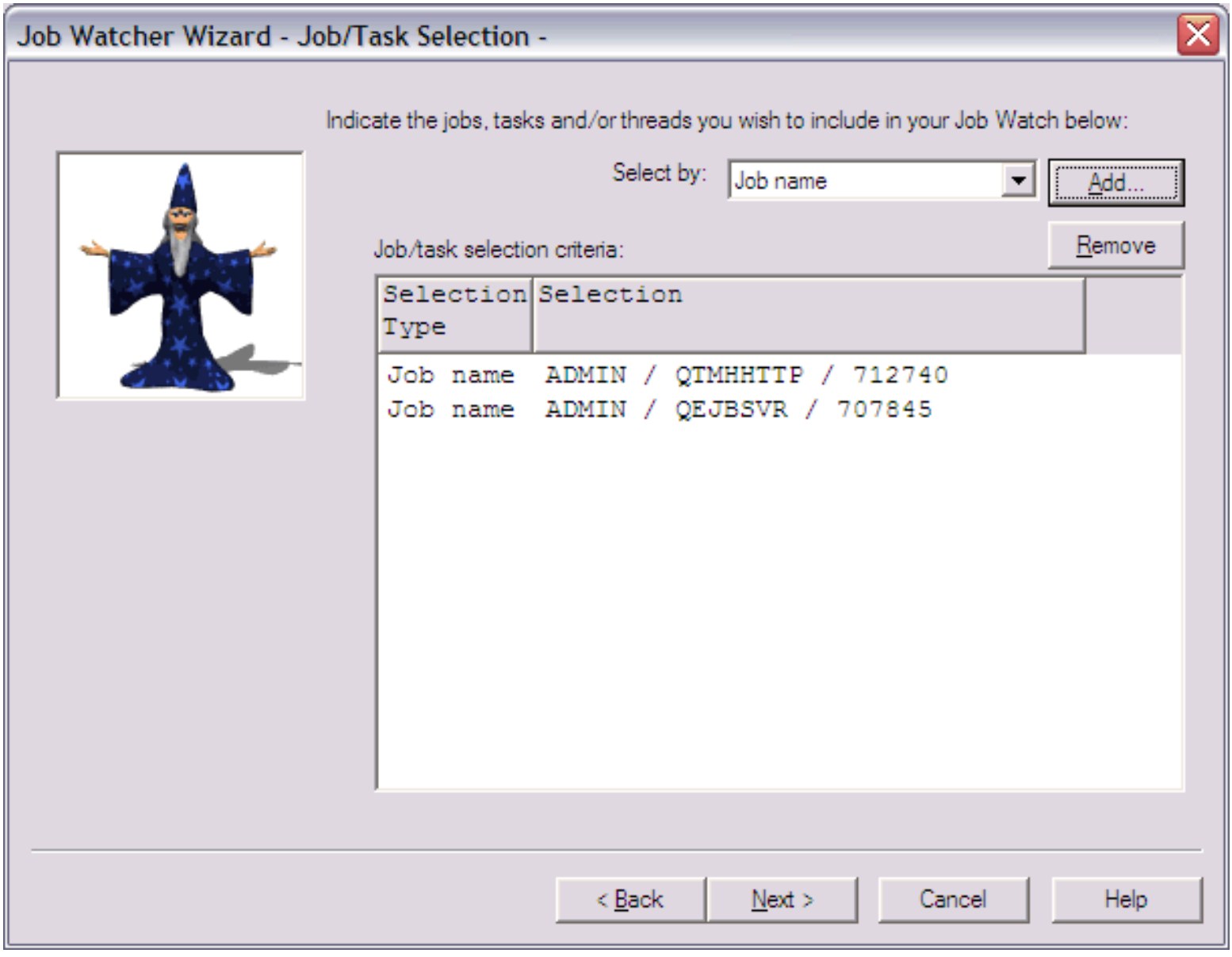
Option	Description
Include all jobs and tasks	All "active" jobs and tasks running on the system will be collected. Active jobs/tasks are defined as those jobs or tasks that used the CPU for each interval collected.
Include all jobs	All "active" jobs running on the system will be collected. Active jobs are defined as jobs that used the CPU for each interval collected.
Include all tasks	All "active" tasks running on the system will be collected. Active tasks are defined as tasks that used the CPU for each interval collected.
Select specific jobs and tasks	Selecting this option will display the Job/task selection page when the 'Next' button on the Wizard is pressed. This window provides many ways to select or filter which jobs/tasks to collect among the jobs/tasks running on the system.
Collect idle jobs/tasks on 1st interval	This option will collect an interval of data for every job/task found within the collection regardless if the job/thread/task used CPU or not. Normally data is not collected for jobs and tasks that did not use CPU during an interval. If a job never uses CPU throughout the entire collection the job name will not be displayable in the reports unless this option is used.

Table of Contents | Previous | Next

2.4.3.8 Job/task Selection

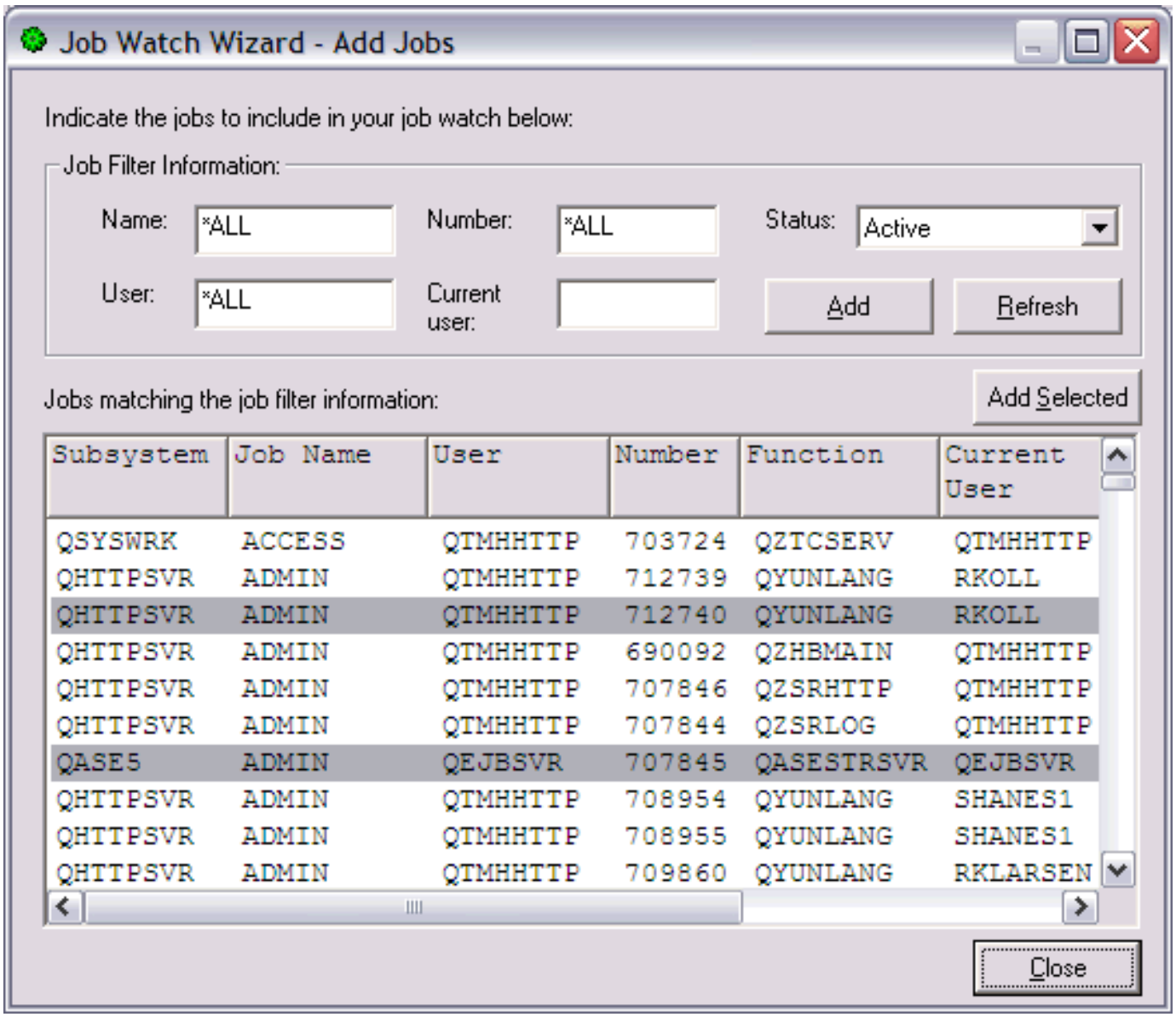
This window provides the user with the ability to select the jobs and tasks to include in the collection. There are six different ways to select the jobs/tasks to use in the collection: Job name, task name, current user profile, subsystem, pool ID, and taskcount. These options are listed within the select by drop down list. After making the selection in the list, pressing the Add... button will display the appropriate interface in order to make the selection and add it to the list of job/task selection criteria.

An example of this page of the Wizard is:



Job name selection

Pressing the Add... button while "Job name" is selected in the Select by drop down list will display the following window.



This window displays the list of jobs on the system and allows the user to add generic or specific job names to the job/task selection criteria list on the job/task selection page of the Wizard.

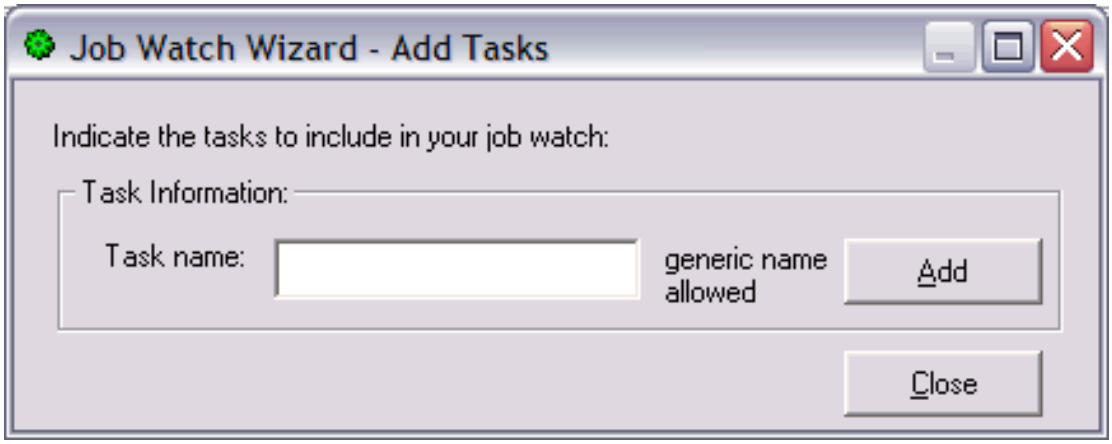
The following table describes the fields on this window:

Option	Description
Job Filter information: Job Name	This field is used to specify a generic job name. This job name may be used to either display a list of active jobs running on the system that match the generic name (by pressing the Refresh button), or add a job/task selection criteria using a generic name (by pressing the Add button).

Job Filter information: Job User	This field is used to specify a generic job user name. This job user name along with the job name filter may be used to either display a list of active jobs running on the system that match the generic job user name (by pressing the Refresh button), or add a job/task selection criteria using a generic job user name (by pressing the Add button).
Job Filter information: Job Number	This field is used to specify the job number to use when either filtering the list of active jobs or adding a job selection criteria to the job/task selection page of the Wizard.
Job Filter information: Current user	Indicates the current user profile to use when displaying the list of active jobs. This option only applies to the "Refresh" button for updating the active list of jobs to select from and does not apply to the Add... button (can't select jobs by current user profile using the Add button). To select all jobs for a specific user profile use the "current user profile" selection type on the Job/Task selection page of the Wizard.
Add	This button will add the currently specified job information filter (job name, job user and job number) to the list of job/task selection criteria on the Job/Task selection page of the Wizard. This option does not apply to the current user field.
Refresh	This button will update the list of "jobs matching the job filter information".
Jobs list	This is the list of jobs matching the job name, job user, job number and current user profile specified. This list may be used to select individual jobs to collect in the job watch.

Task name selection

Pressing the Add... button while "Task name" is selected in the Select by drop down list will display the following window.



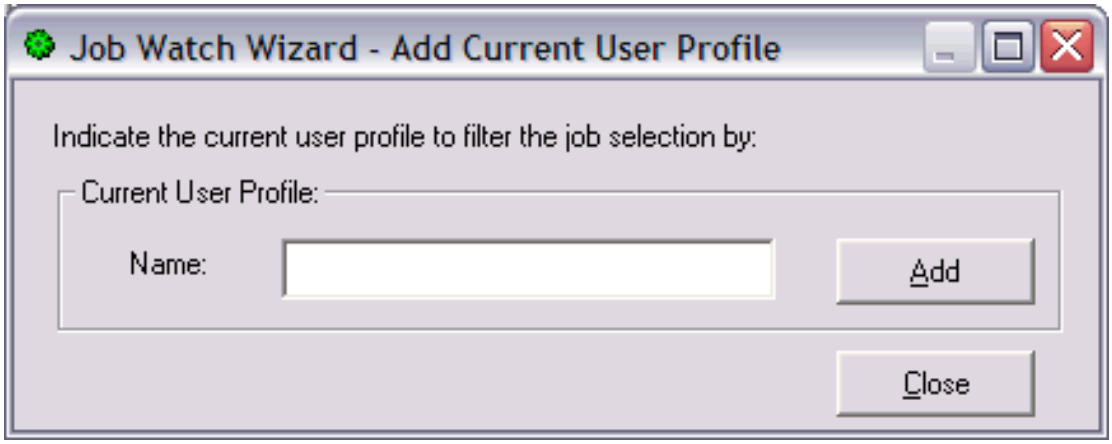
This window displays a field to specify a generic task name to include in the job/task selection criteria list on the job/task selection page of the Wizard.

The following table describes the fields on this window:

Option	Description
Task name	This field is the generic task name. Pressing the Add button will add the generic task name to the list on the Job/task selection page of the Wizard. This field could also contain a specific task name if it is keyed in correctly, but there is not an option to view the list of active tasks from this window.

Current user profile selection

Pressing the Add... button while "Current user profile" is selected in the Select by drop down list will display the following window.



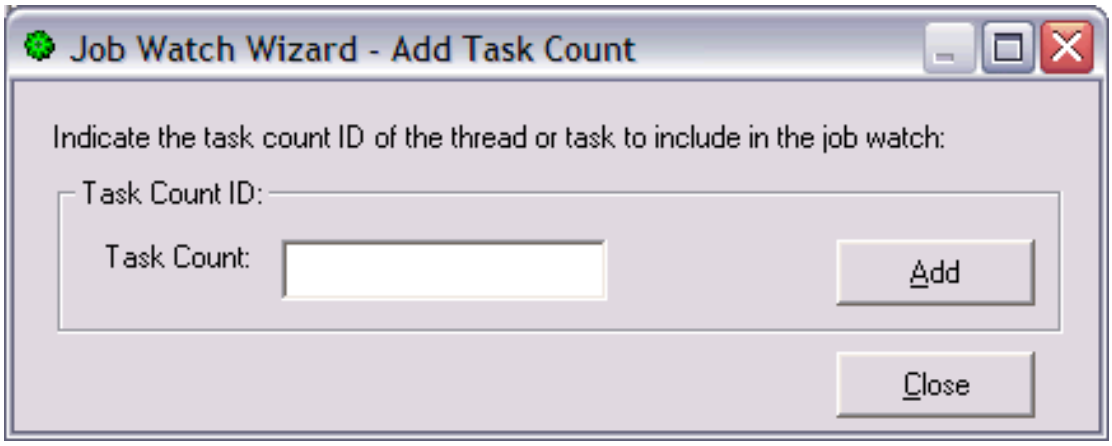
This window displays a field to specify a current user profile name to include in the job/task selection criteria list on the job/task selection page of the Wizard.

The following table describes the fields on this window:

Option	Description
Current user profile name	This field is for entering the current user profile to collect job information for. Generic names are not allowed for this field.

Task count selection

Pressing the Add... button while "Task count (hex)" is selected in the Select by drop down list will display the following window.



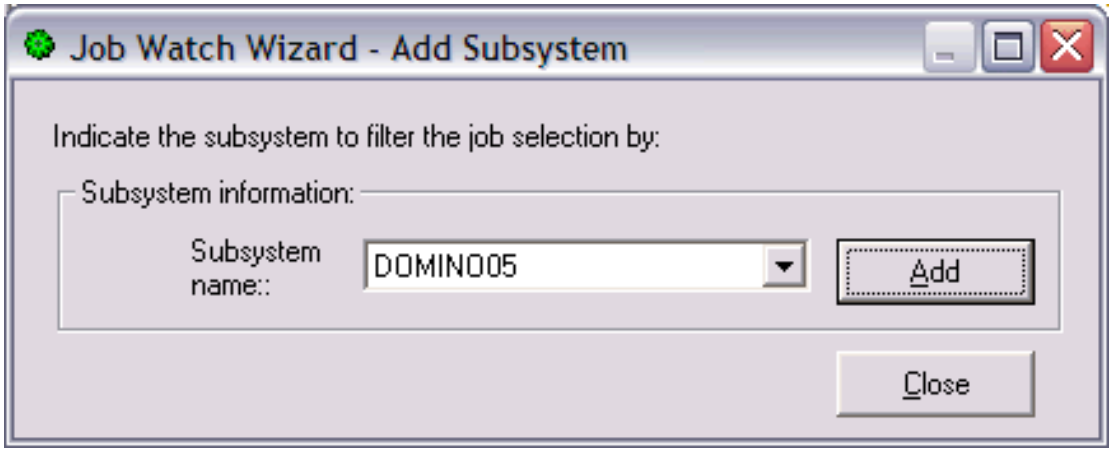
This window displays a field to specify the task count to include in the job/task selection criteria list on the job/task selection page of the Wizard.

The following table describes the fields on this window:

Option	Description
Task count	This field is the task count which uniquely identifies a job/thread or task on a system. The task count must be entered in 16 character HEX format.

Subsystem name selection

Pressing the Add... button while "Subsystem" is selected in the Select by drop down list will display the following window.



This window displays a list of subsystems that are running on the system to select from. By selecting a subsystem this indicates that all jobs that running in that subsystem will be included in the job watch (if not filtered out by other parameters which may also be used).

The following table describes the fields on this window:

Option	Description

Subsystem	Contains a list of active subsystems. Clicking the Add button will add the selected subsystem to the list on the Job/task selection page.

Pool ID selection

Pressing the Add... button while "Pool ID" is selected in the Select by drop down list will display the following window.

This window allows the user to select the jobs/tasks to include in the job watch by the pool the jobs/tasks are running in.

The following table describes the fields on this window:

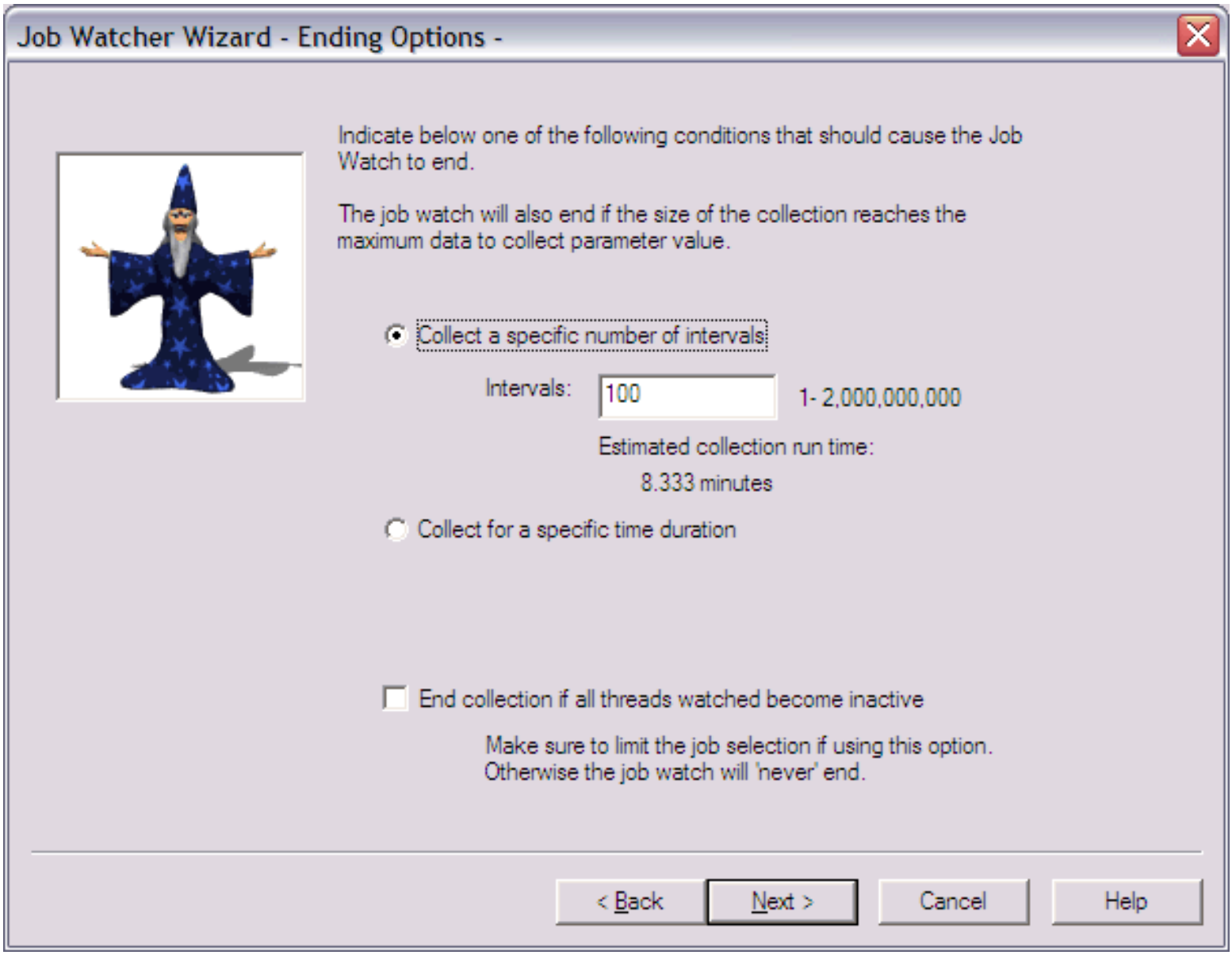
Option	Description
Pool ID	This field contains the desired pool ID to collect job/task/threads from. Clicking the Add button will add the selected pool information to the list on the Job/task selection page of the Wizard.

Table of Contents | Previous | Next

2.4.3.9 Ending criteria

This page of the Wizard allows the user to specify for how long the collection should run either by total number of intervals to collect or by the total time the collection should run.

An example of this page of the Wizard is:



The following table describes the options available on this window:

Option	Description

Collect a specific number of intervals	This option provides the ability to end the collection after a specified number of intervals have been collected.
Collect a specific amount of time	This option allows the user to specify that the collection should run for a specified number of seconds.
End collection if all threads become inactive	This option is not available if all jobs and/or tasks on the system are being watched. This option is useful in order to end the collection once a single job or set of jobs being watched end. The time or interval values above still apply, this option will only allow the collection to be ended early if all threads watched become inactive.

Table of Contents | Previous | Next

2.4.3.10 Scheduling options

This page allows the user to determine a specific date and time for the Job Watch to begin collecting data. By clicking the checkbox the user can optionally include a date/time to schedule the Job Watch. This option will utilize the SCDDATE and SCDTIME parameters on the SBMJOB command when the collection is submitted.

An example of this page of the Wizard is:

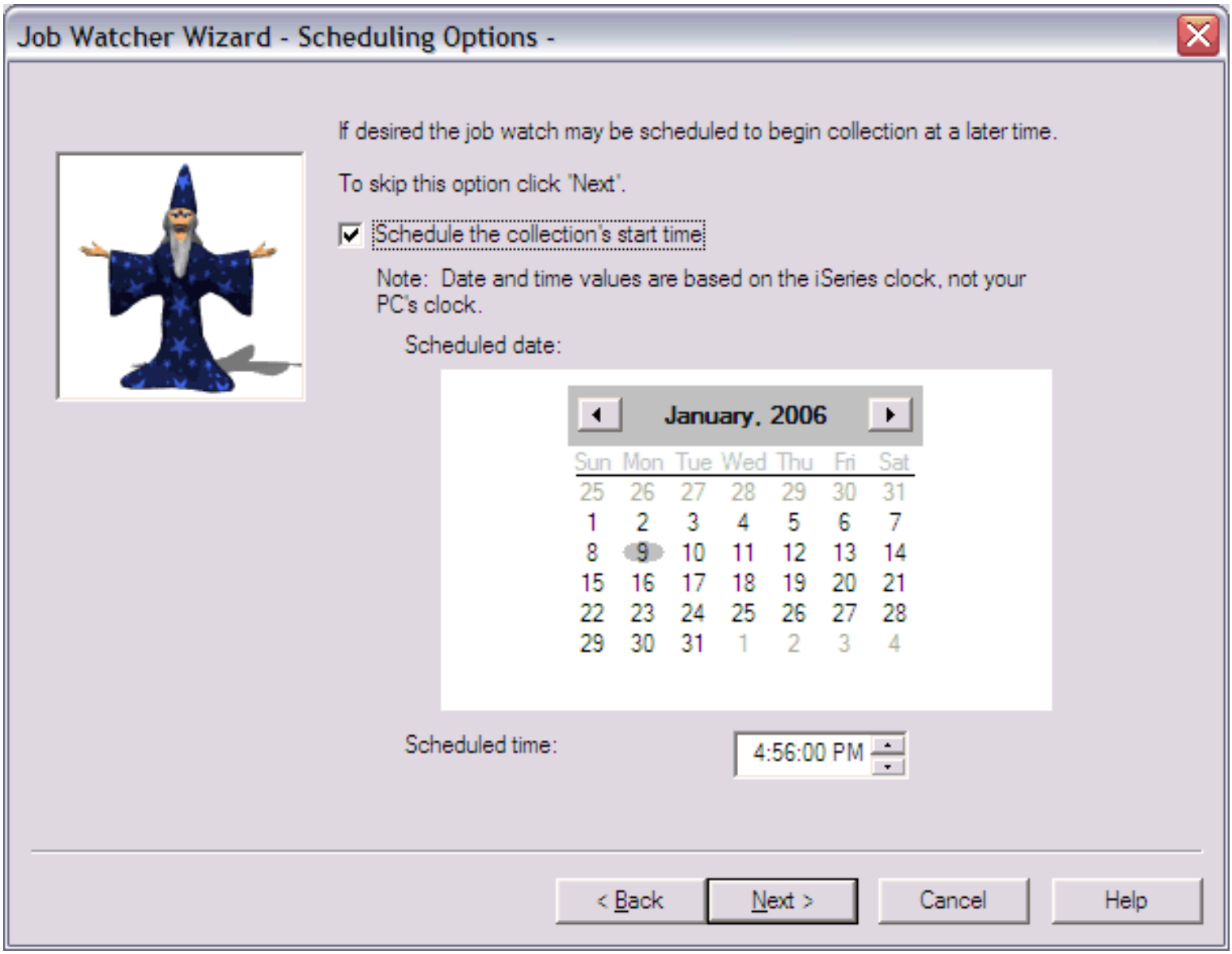


Table of Contents | Previous | Next

2.4.3.11 Summary

The Summary page provides complete details about all selections made in the wizard. If anything listed doesn't look right, use the Back button to go back and make any changes necessary. After clicking 'Finish' a SBMJOB command will be issued to start the job watch. This command is listed at the bottom this page, and can be copied to a green screen session and modified if necessary.

Note that this SBMJOB command contains references to the job queue and subsystem names defined by the current installation on the current system. These parameters may need to be modified if the green screen command is desired to be executed on another system depending on the parameters used during the installation on the other system.

After the job watch is submitted it will take several seconds before anything appears in the GUI while the files are being allocated. Use F5 to refresh the list of job watches in the current library in order to work with the newly created job watch.

Job Watcher Wizard - Summary -

Here is a summary of your selections.

You have selected to create a Job Watch named FDAS in library MCCARGAR

Selected options:
Job Watch name: FDAS
Library: MCCARGAR
Schedule collection to begin on 01/09/06 at 16:56
Interval duration: 5 seconds
Ending criteria: 100 intervals
Maximum data to collect: 1000 megabytes
Description:

Selected Jobs:
*ALL

Selected Tasks:
*ALL

Activation group collect: None
Call stack collection (for active jobs/tasks): Every interval
Call stack collected for jobs in conflict for at least 1 microseconds

To submit your request now click 'Finish'

< Back | Finish | Cancel | Help

Table of Contents | Previous | Next

2.4.4 Copying

A job watch can be copied by using the Copy... menu found by right-clicking on a job watch from the Job Watcher component view.
This option will execute the CPYJWCOL green screen command. Copying a job watch that is still running is not allowed. Multiple job watches can be copied at the same time if desired.

Field	Description
Watch(es)	The job watches to be copied
From library	The library to copy the job watches from.
To library	The library to copy the job watches to. This can either be an existing library from the list or a new library name.

Table of Contents | Previous | Next

2.4.5 Deleting

A job watch can be deleted by using the Delete... menu found by right-clicking on a job watch from the Job Watcher component view.
This option will execute the DLTJWCOL green screen command. Deleting a job watch that is still running is not allowed. The job watch must be ended before trying to delete it.

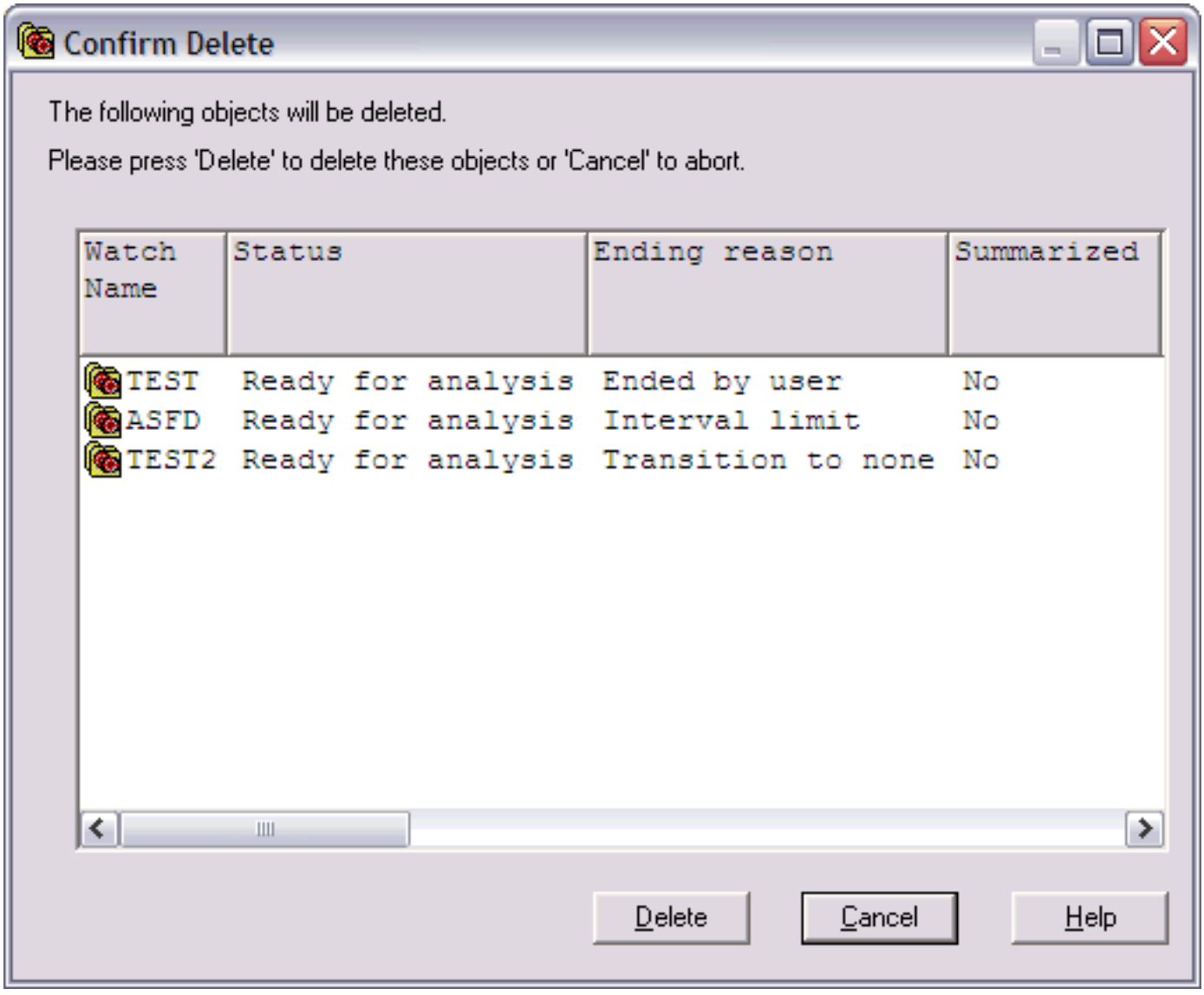
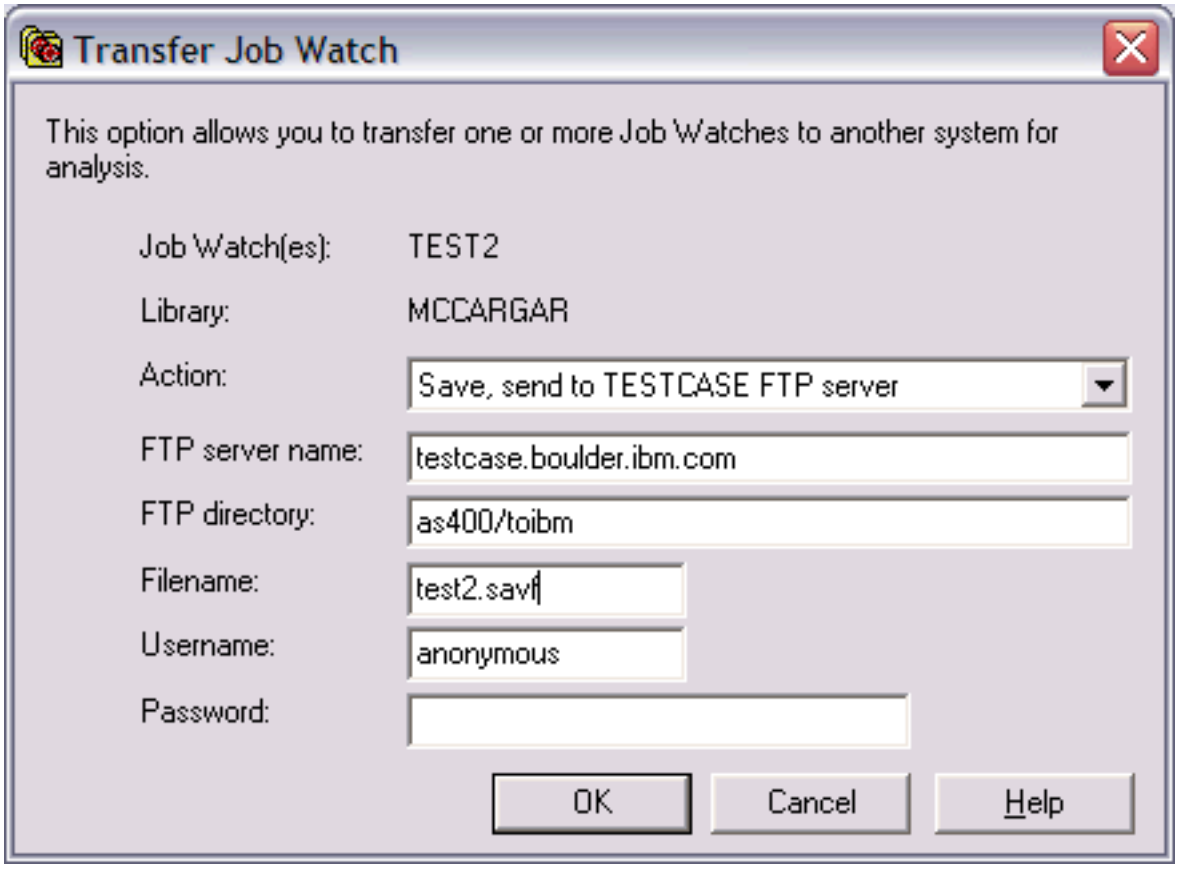


Table of Contents | Previous | Next

2.4.6 Transfering

A job watch can be transfered to another system by using the Transfer to... menu found by right-clicking on a job watch from the Job Watcher component view.
This option will execute the FTPJWCOL green screen command. This option is only available for job watches that are no longer running.

This option is only supported for release V5R3 and higher.



Field	Description
Job Watch(es)	The collections to transfer.
Library	The library the collections to transfer are currently located in.

Action	Indicates the type of transfer. Collections can be sent to another system and restored (if iDoctor is installed on the remote system), or just sent to an FTP server as a save file and not restored.

Action 'Save, send and restore to iSeries library'

Field	Description
Remote system	The name of the remote system (with iDoctor installed) to send the save file to and have it restored. The remote system must have Job Watcher installed at the same release as the current system.
Remote library	The library name on the remote system to restore the collection(s) to.

Action 'Save, send to FTP server' or 'Save, send to TESTCASE FTP server'

Field	Description
FTP server name	The name of the FTP server to send the collection to.
FTP directory	The directory to send the FTP collection to. This directory must already exist.
Filename	The name of the file to save the collection to.
Username	The user name to connect to the FTP server with.
Password	The password to connect ot the FTP server with. If using anonymous FTP, you can supply your email address as a password if desired.

Table of Contents | Previous | Next

2.4.7 Stopping

An active job watch can be stopped by using the Stop menu found by right-clicking on a job watch from the Job Watcher component view.

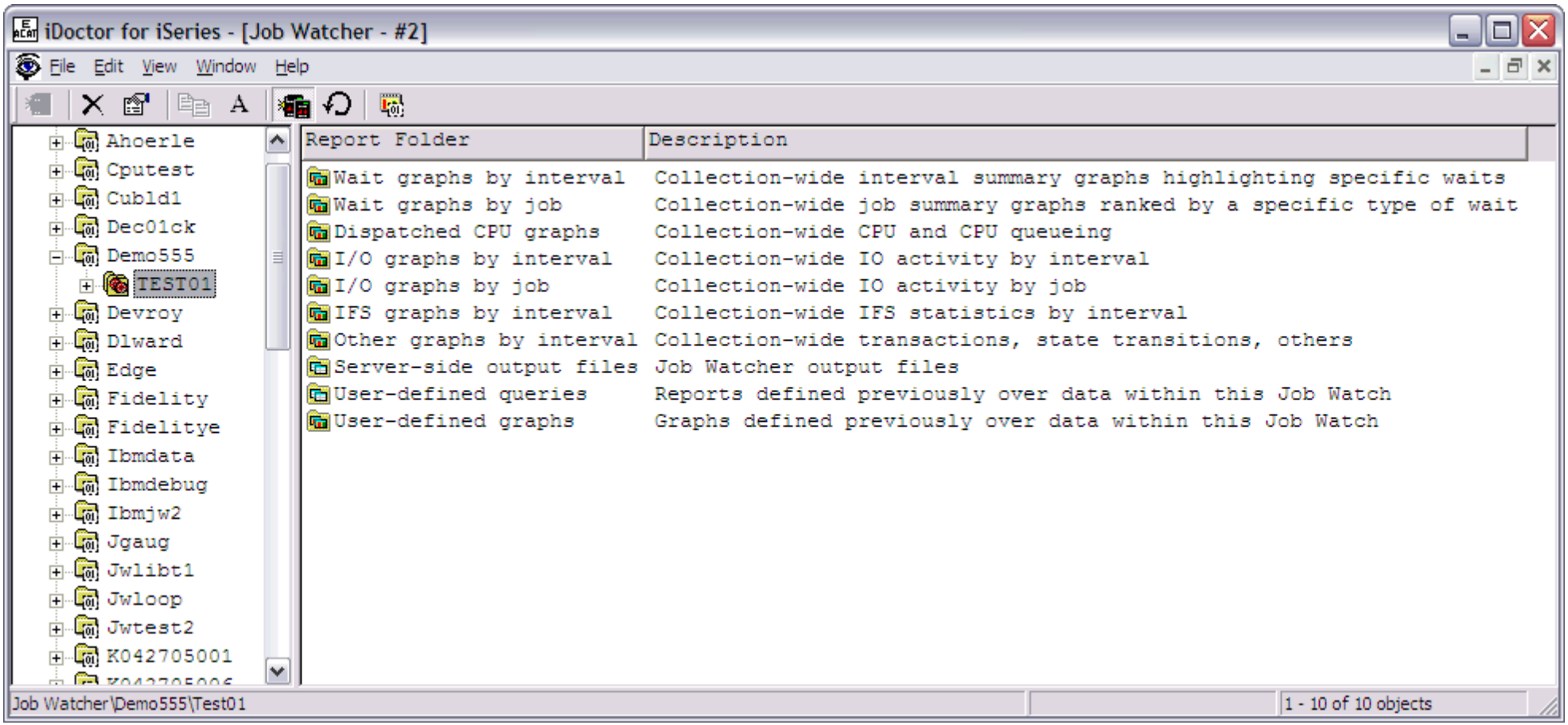
This option will issue an ENDJOB command for the job that is producing the active job watch.

Table of Contents | Previous | Next

2.5 Job Watch summary reports

Moving down the tree within each job watch are several report folders. These folders contain many different ways to look at the Job Watcher data. Right-clicking on a job watch provides options to view graphs over many of the same graph category folders listed underneath a job watch. These popup menu options allow the user to open graphs without needing to expand the graph folders.

An example of the contents of a job watch is shown below:



The following table describes the contents of each report folder:

Folder	Description

Wait graphs by interval	Contains collection-wide wait summary graphs showing a bar per interval. Each bar is a summary of a particular set of wait times over all jobs for an interval in the job watch. If unsure of where to investigate first, these graphs and the wait graphs by job are a good place to start.
Wait graphs by job	These graphs are wait summary graphs showing a bar per job. The jobs are sorted by the type of wait the graph is showing with the job experiencing the most time spent in the indicated wait at the top of the graph.
Dispatched CPU graphs	This folder contains collection-wide summary graphs illustrating CPU and CPU queueing usage by interval.
I/O graphs by interval	This folder contains collection-wide summary graphs showing IO operations and disk activity by interval.
I/O graphs by job	This folder contains collection-wide summary graphs showing IO operations and disk activity ranked by job..
IFS graphs by interval	This folder contains collection-wide summary graphs showing IFS activity by interval.
Other graphs by interval	This folder contains collection-wide summary graphs showing other types of information such as state transitions and transactions.
Server-side output files	A complete list of each Job Watcher output file produced for the collection. These files are queryable through the query definition interface.
User-defined queries	Contains table views over the Job Watcher output files previously defined on the current system. These queries are stored in file QAIDRSQL04 in the current library and library QUSRSYS for system scoped queries.
User-defined graphs	Contains graph views over the Job Watcher output files that were previously defined on the current system. These graphs are stored in file QAIDRGPH08 in the current library and library QUSRSYS for system scoped graphs.

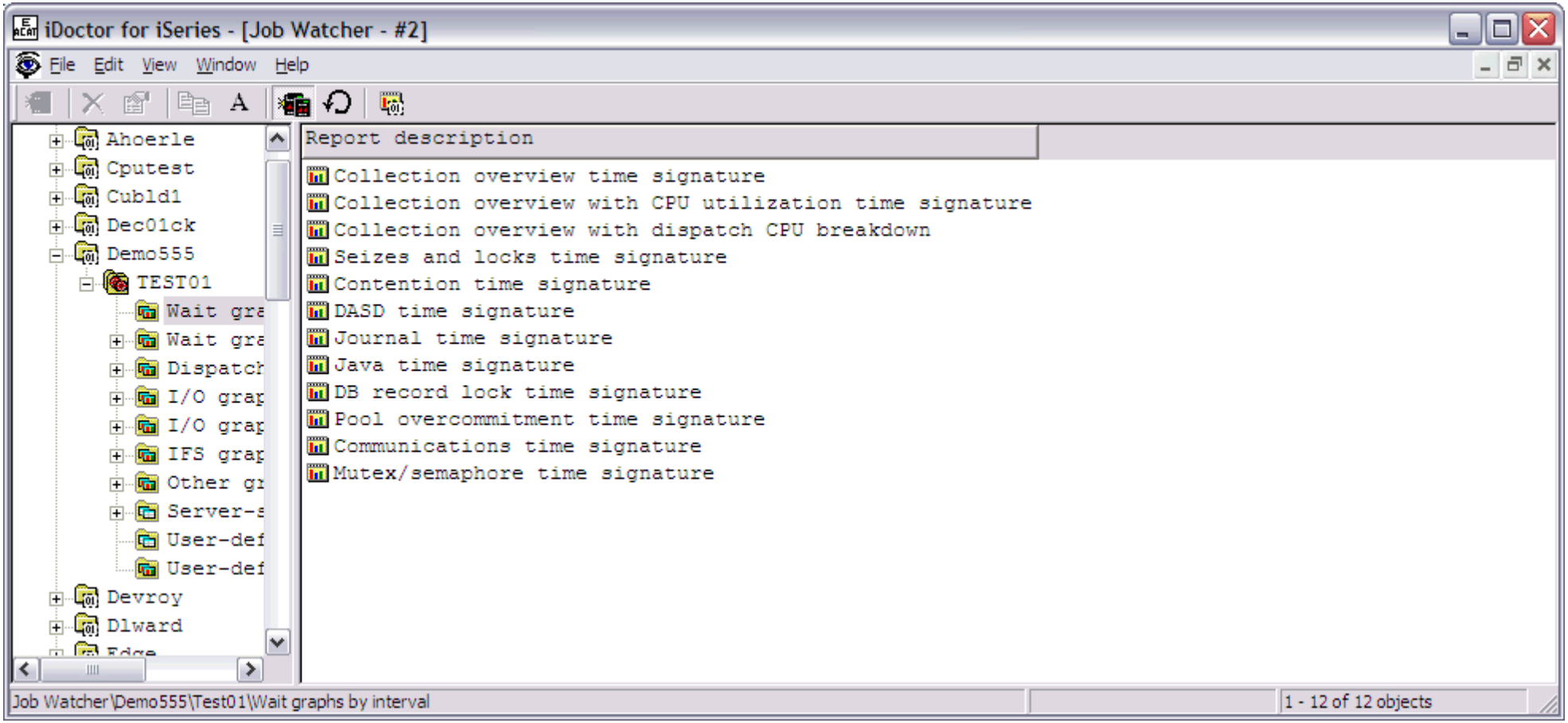
Table of Contents | Previous | Next

2.5.1 Wait graphs by interval

This folder contains a list of graphs which contain summary data over the job watch relating to waits. These graphs display a bar per interval showing the amount of time spent (across all jobs in the collection) in various types of potentially bad waits. High numbers in these graphs do not necessarily indicate a performance problem. A comparison should be made using these graphs from a time when the system is running well, with a time where the system is running poorly in order to determine if there is a problem.

The data within these graphs could look quite different depending on if the collection was summarized or not. The summarized column when viewing a list of collections in a library indicates this. Summarizing a collection will greatly improve the execution time of these graphs and will also show contributions for jobs that were in idle waits.

An example of the contents of the wait graphs by interval folder for a job watch is:



The following table illustrates the menu options available by right-clicking on a graph in the list.

Menu	Description
Open graph(s)	Opens the selected graph(s) into a Data Viewer. If a Data Viewer has already been opened submenus will appear underneath this menu in order to give the option of opening the graph(s) into an already open Data Viewer.

Table of Contents | Previous | Next

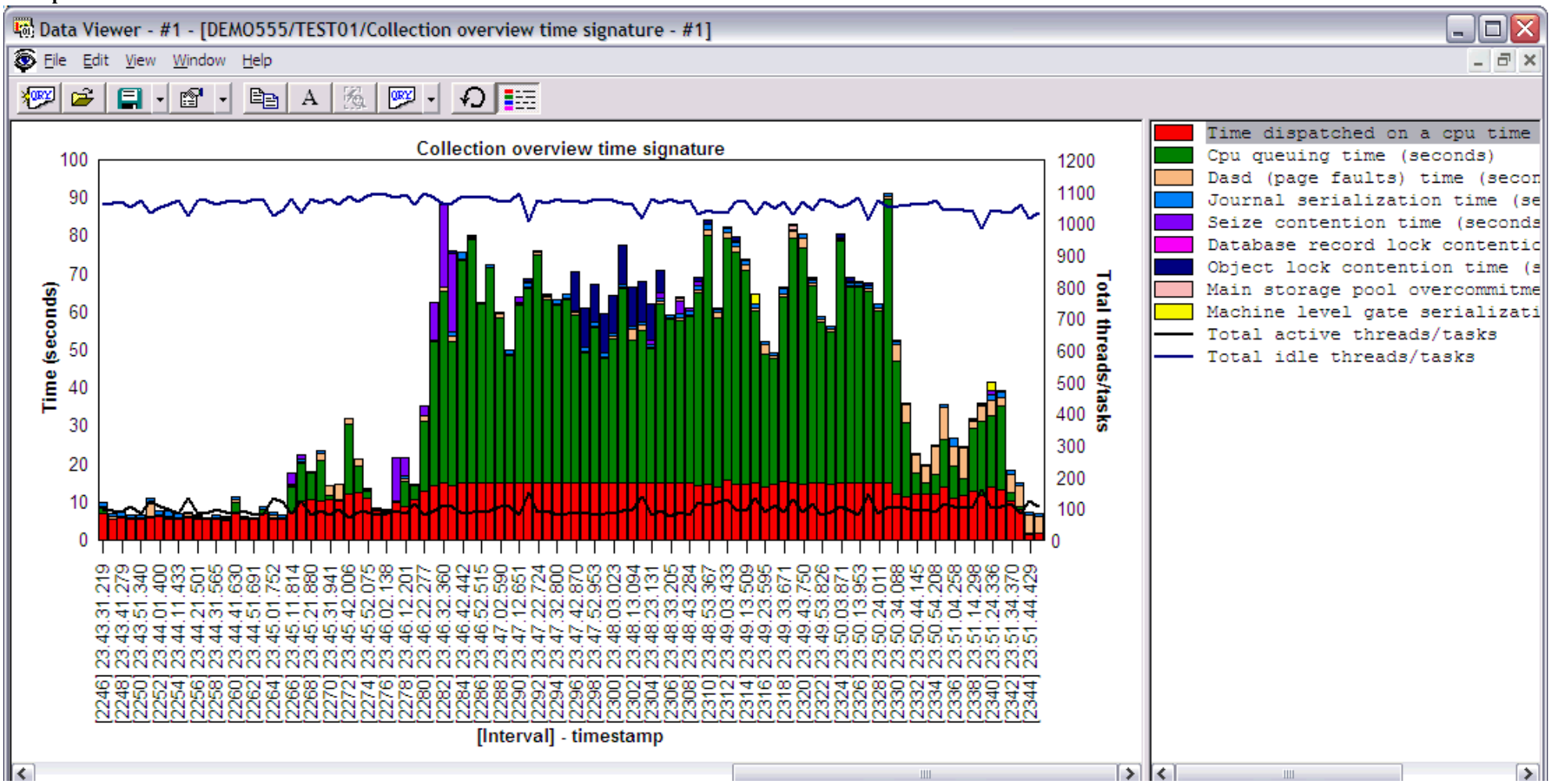
2.5.1.1 Collection overview time signature

Description: This graph shows a **summary** of the time all jobs spent in waits that are commonly associated with performance problems. The graph makes use of the 32 run/wait buckets used in Job Watcher.

This graph is a good starting point because it provides a high-level overview of important wait types such as seizes, and locks, page faults and CPU.

Graph Type: summarized by interval (stacked vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: Each color represents the total amount of time spent across all threads in a particular type of wait. All times are provided in seconds.
Second Y-Axis: The secondary Y-Axis displays the total number of active and idle threads/tasks on the system for each interval. Idle threads are threads that did not use CPU in the interval.

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Run/wait graphs by job	This menu allows the user to view the jobs for a paticular interval or interval range ranked by the desired wait type. By right-clicking on the graph over a particular color (such as dark blue for object locks) the user can drill into an interval to view the jobs that contributed to the selected color.
I/O graphs by job	Displays I/O and disk activity graphs for a particular interval or interval range.
Wait graphs by interval	This menu allows the user to view one of the wait graphs by interval starting at the interval range selected (or right-clicked on).
Detail reports	Displays one of the table view reports for the selected interval within the job watch.
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar in a properties window. This window will also appear by left-clicking on any bar in the graph.

2.5.1.2 Collection overview with CPU utilization time signature

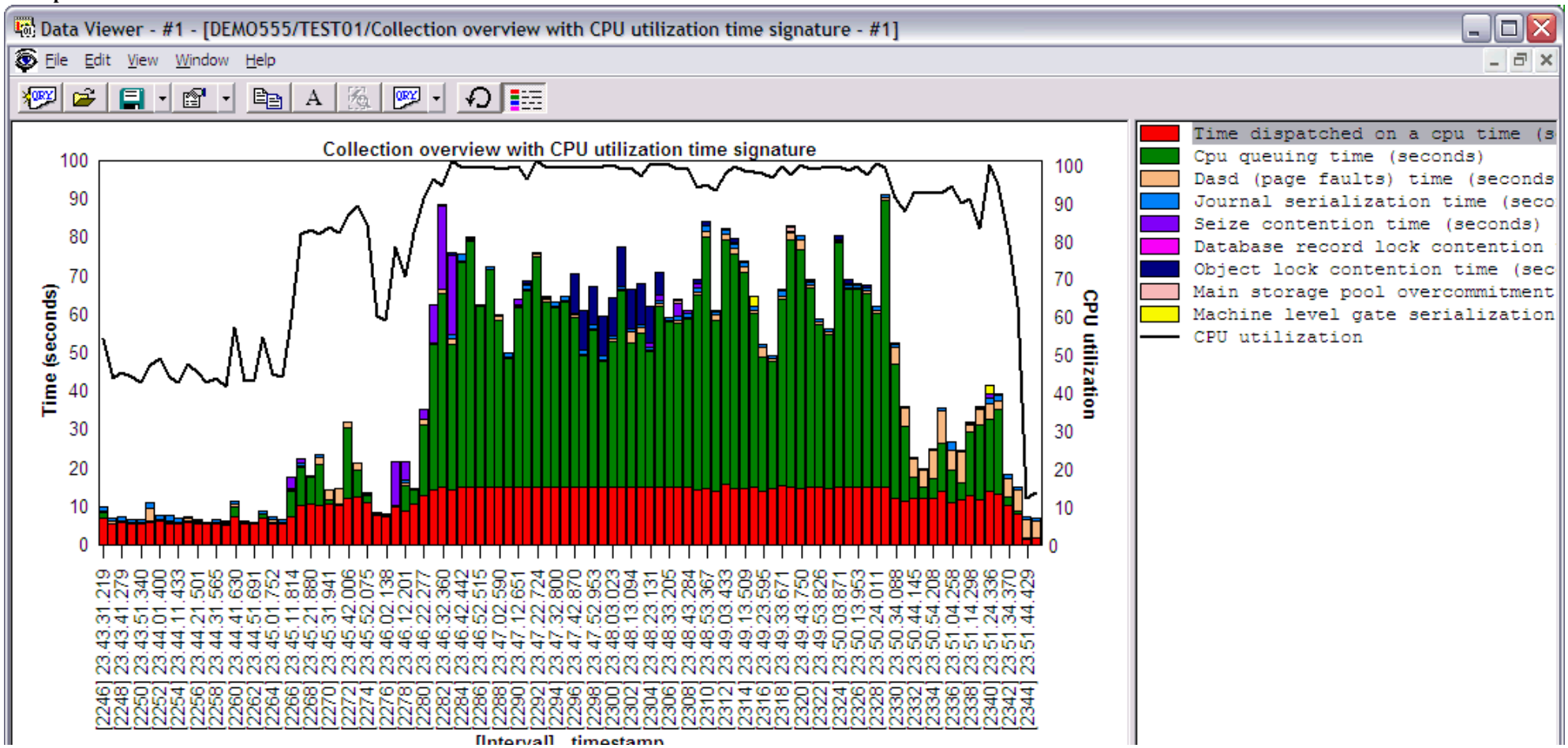
Description: This graph shows a **summary** of the time all jobs spent in waits that are commonly associated with performance problems. The graph makes use of the 32 run/wait buckets used in Job Watcher.

This graph is a good starting point because it provides a high-level overview of important wait types such as seizes, and locks, page faults and CPU.

This graph is only shown if the collection has been summarized.

Graph Type: summarized by interval (stacked vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: Each color represents the total amount of time spent across all threads in a particular type of wait. All times are provided in seconds.
Second Y-Axis: The secondary Y-Axis displays the CPU utilization calculated by Job Watcher for each interval.

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Run/wait graphs by job	This menu allows the user to view the jobs for a paticular interval or interval range ranked by the desired wait type. By right-clicking on the graph over a particular color (such as dark blue for object locks) the user can drill into an interval to view the jobs that contributed to the selected color.
I/O graphs by job	Displays I/O and disk activity graphs for a particular interval or interval range.
Wait graphs by interval	This menu allows the user to view one of the wait graphs by interval starting at the interval range selected (or right-clicked on).
Detail reports	Displays one of the table view reports for the selected interval within the job watch.
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar in a properties window. This window will also appear by left-clicking on any bar in the graph.

2.5.1.3 Collection overview with dispatch CPU breakdown time signature

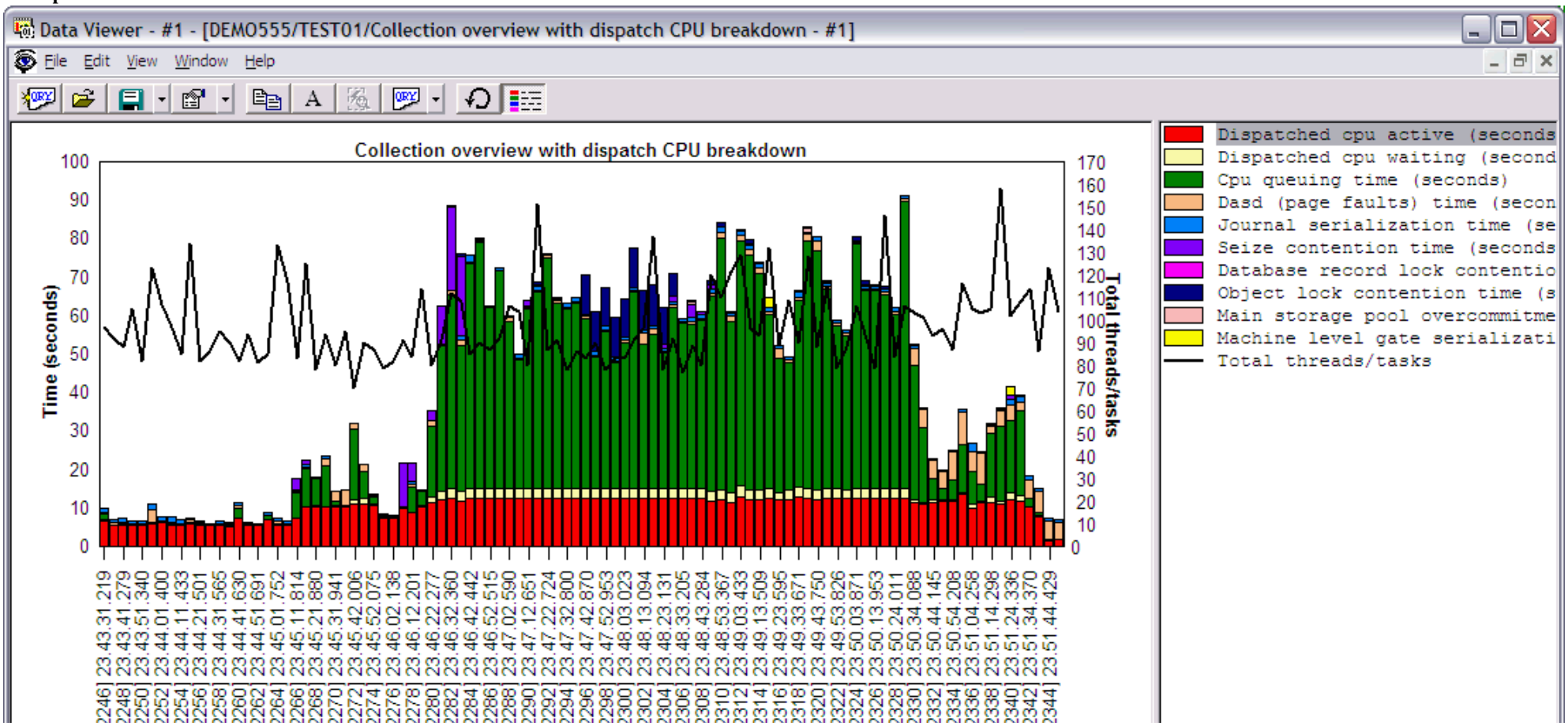
Description: This graph shows a **summary** of the time all jobs spent in waits that are commonly associated with performance problems. The graph makes use of the 32 run/wait buckets used in Job Watcher.
This graph breaks down the Dispatched CPU bucket (QTIME01) into two parts: dispatched CPU active and dispatched CPU waiting.

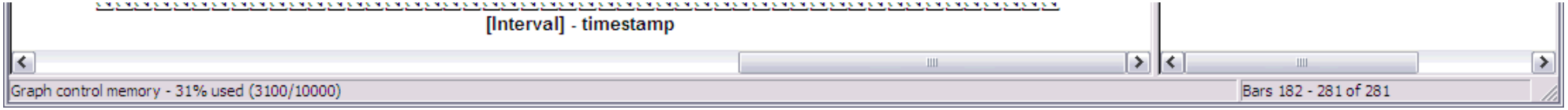
This graph is a good starting point because it provides a high-level overview of important wait types such as seizes, and locks, page faults and CPU.

This graph is only shown if the collection has been summarized.

Graph Type: summarized by interval (stacked vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: Each color represents the total amount of time spent across all threads in a particular type of wait. All times are provided in seconds.
Second Y-Axis: The secondary Y-Axis displays the active threads/tasks for each interval.

Example:





Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Run/wait graphs by job	This menu allows the user to view the jobs for a paticular interval or interval range ranked by the desired wait type. By right-clicking on the graph over a particular color (such as dark blue for object locks) the user can drill into an interval to view the jobs that contributed to the selected color.
I/O graphs by job	Displays I/O and disk activity graphs for a particular interval or interval range.
Wait graphs by interval	This menu allows the user to view one of the wait graphs by interval starting at the interval range selected (or right-clicked on).
Detail reports	Displays one of the table view reports for the selected interval within the job watch.
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar in a properties window. This window will also appear by left-clicking on any bar in the graph.

Table of Contents | Previous | Next

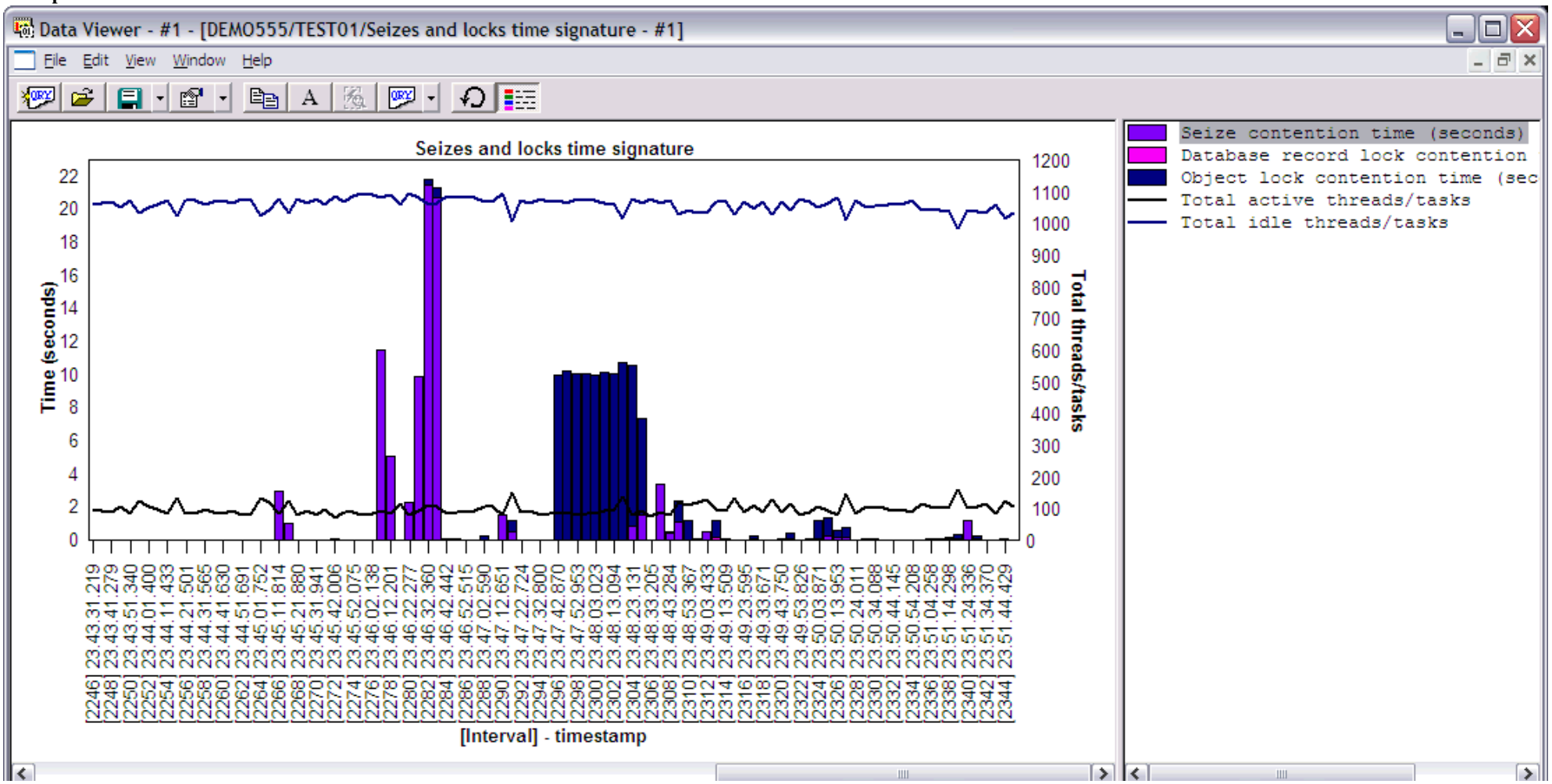
2.5.1.4 Seizes and locks time signature

Description: This graph shows a **summary** of the time all jobs spent in seizes and locks type wait conditions for each interval within the job watch. The graph makes use of the 32 run/wait buckets used in Job Watcher.

This graph is a good starting point because it provides a high-level overview of when the system experienced seizes or object locks.

Graph Type: summarized by interval (stacked vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: Each color represents the total amount of time spent across all threads in a particular type of wait. All times are provided in seconds.
Second Y-Axis: The secondary Y-Axis displays the total number of active and idle threads/tasks on the system for each interval. Idle threads are threads that did not use CPU in the interval.

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Run/wait graphs by job	This menu allows the user to view the jobs for a paticular interval or interval range ranked by the desired wait type. By right-clicking on the graph over a particular color (such as dark blue for object locks) the user can drill into an interval to view the jobs that contributed to the selected color.
I/O graphs by job	Displays I/O and disk activity graphs for a particular interval or interval range.
Wait graphs by interval	This menu allows the user to view one of the wait graphs by interval starting at the interval range selected (or right-clicked on).
Detail reports	Displays one of the table view reports for the selected interval within the job watch.
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar in a properties window. This window will also appear by left-clicking on any bar in the graph.

Table of Contents | Previous | Next

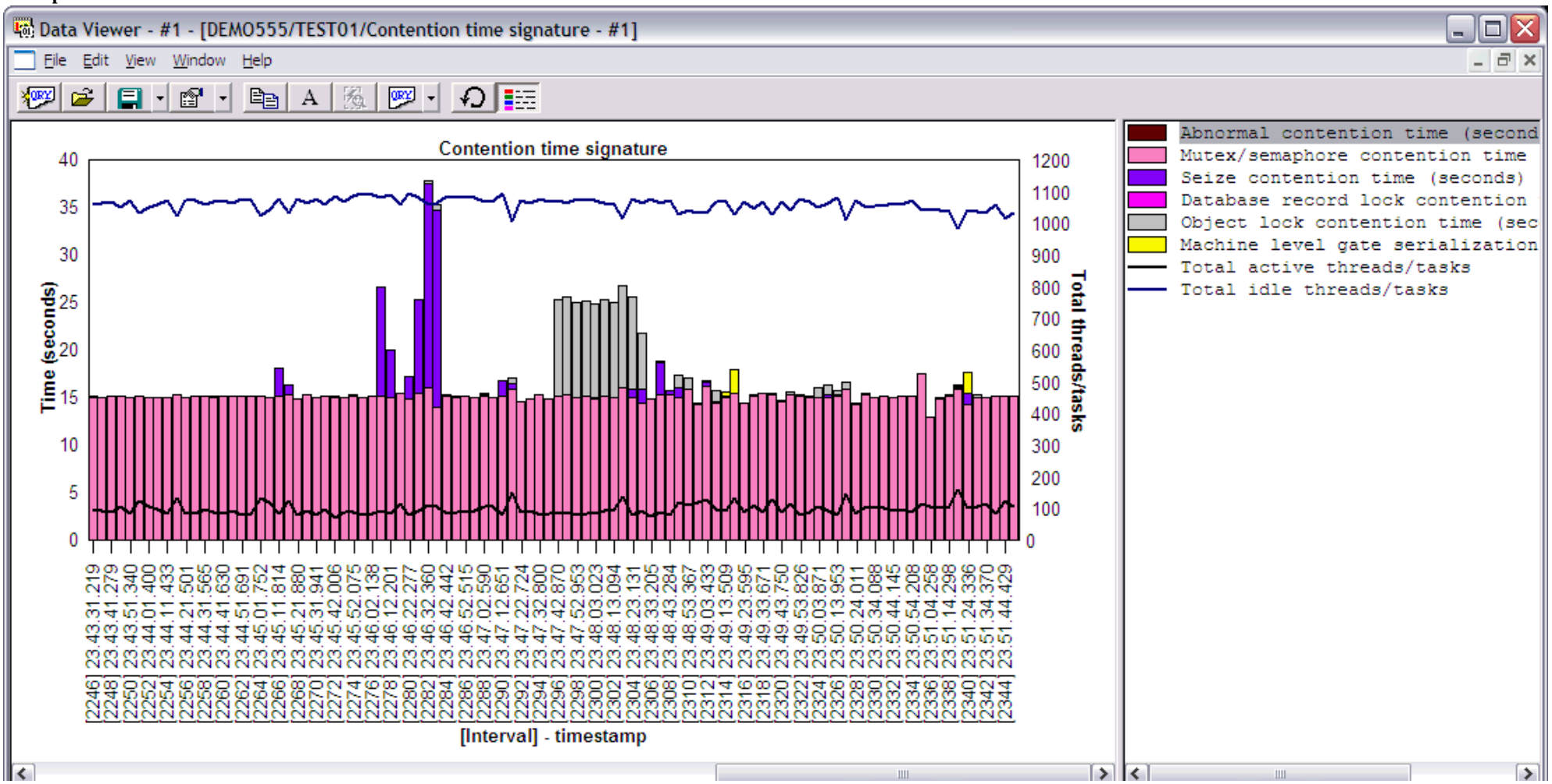
2.5.1.5 Contention time signature

Description: This graph shows a **summary** of contention times for each interval within the job watch. Abnormal contention time is a special type of wait that indicates jobs have been unable to perform work and have made several attempts to do work but failed.

If these numbers are high the system is probably running very slowly.

Graph Type: summarized by interval (stacked vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: Each color represents the total amount of time spent across all threads in a particular type of wait. All times are provided in seconds.
Second Y-Axis: The secondary Y-Axis displays the total number of active and idle threads/tasks on the system for each interval. Idle threads are threads that did not use CPU in the interval.

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Run/wait graphs by job	This menu allows the user to view the jobs for a paticular interval or interval range ranked by the desired wait type. By right-clicking on the graph over a particular color (such as dark blue for object locks) the user can drill into an interval to view the jobs that contributed to the selected color.
I/O graphs by job	Displays I/O and disk activity graphs for a particular interval or interval range.
Wait graphs by interval	This menu allows the user to view one of the wait graphs by interval starting at the interval range selected (or right-clicked on).
Detail reports	Displays one of the table view reports for the selected interval within the job watch.
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar in a properties window. This window will also appear by left-clicking on any bar in the graph.

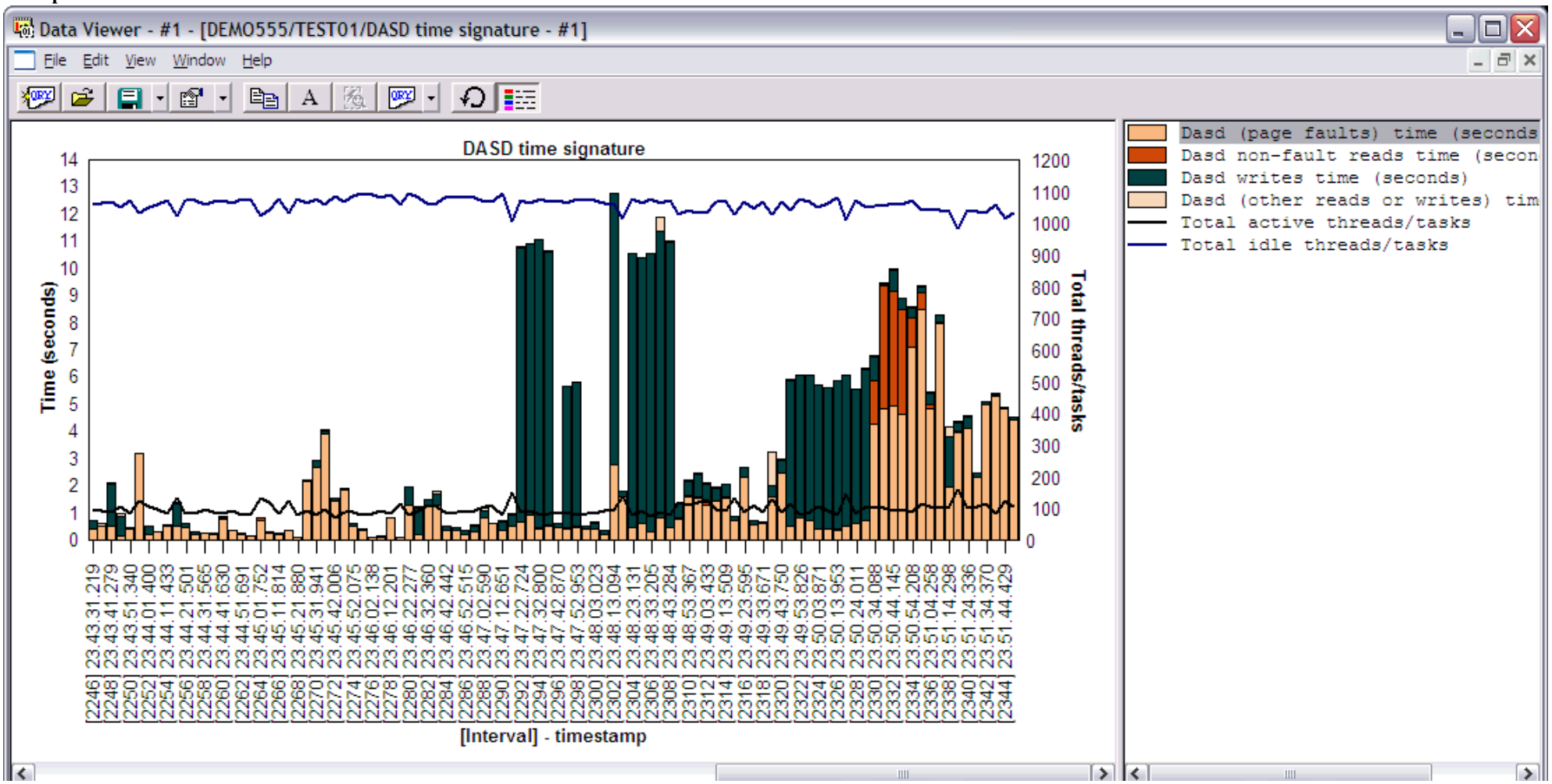
2.5.1.6 DASD time signature

Description: This graph shows a **summary** of the time all jobs spent in DASD waits of various types for each interval within the job watch. The graph makes use of the 32 run/wait buckets used in Job Watcher.

This graph can be used to see a quick summary of when disk use was relatively high or low within the job watch and the types of operations made.

Graph Type: summarized by interval (stacked vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: Each color represents the total amount of time spent across all threads in a particular type of wait. All times are provided in seconds.
Second Y-Axis: The secondary Y-Axis displays the total number of active and idle threads/tasks on the system for each interval. Idle threads are threads that did not use CPU in the interval.

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Run/wait graphs by job	This menu allows the user to view the jobs for a paticular interval or interval range ranked by the desired wait type. By right-clicking on the graph over a particular color (such as dark blue for object locks) the user can drill into an interval to view the jobs that contributed to the selected color.
I/O graphs by job	Displays I/O and disk activity graphs for a particular interval or interval range.
Wait graphs by interval	This menu allows the user to view one of the wait graphs by interval starting at the interval range selected (or right-clicked on).
Detail reports	Displays one of the table view reports for the selected interval within the job watch.
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar in a properties window. This window will also appear by left-clicking on any bar in the graph.

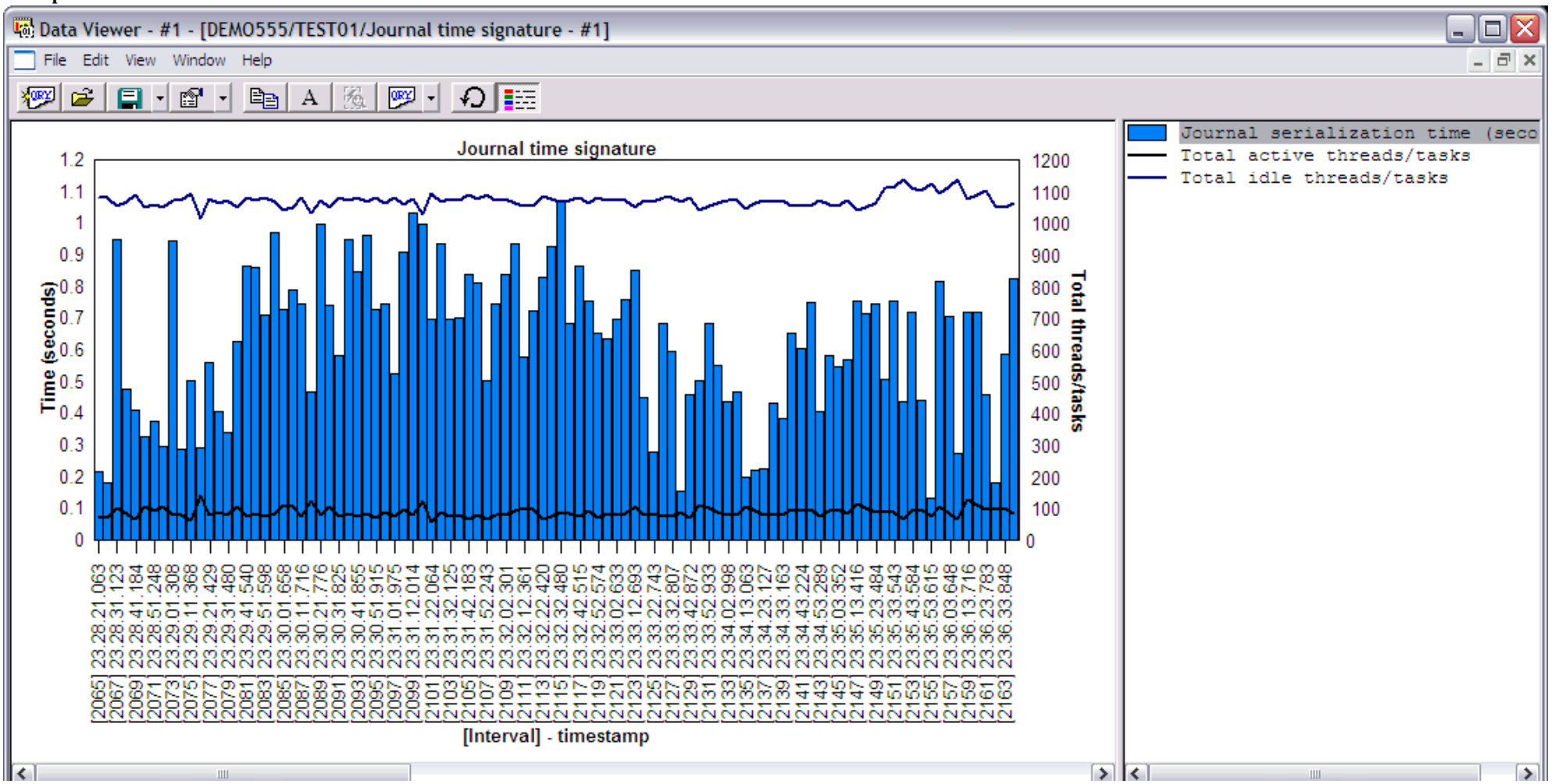
2.5.1.7 Journal time signature

Description: This graph shows a **summary** of the time all jobs spent in journal operations for each interval within the job watch. The graph makes use of the 32 run/wait buckets used in Job Watcher.

This graph can be used to quickly see when journalling was used by the system during the job watch.

Graph Type: summarized by interval (stacked vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: The blue bars represents the total amount of time spent in journaling. All times are provided in seconds.
Second Y-Axis: The secondary Y-Axis displays the total number of active and idle threads/tasks on the system for each interval. Idle threads are threads that did not use CPU in the interval.

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Run/wait graphs by job	This menu allows the user to view the jobs for a paticular interval or interval range ranked by the desired wait type. By right-clicking on the graph over a particular color (such as dark blue for object locks) the user can drill into an interval to view the jobs that contributed to the selected color.
I/O graphs by job	Displays I/O and disk activity graphs for a particular interval or interval range.
Wait graphs by interval	This menu allows the user to view one of the wait graphs by interval starting at the interval range selected (or right-clicked on).
Detail reports	Displays one of the table view reports for the selected interval within the job watch.
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar in a properties window. This window will also appear by left-clicking on any bar in the graph.

Table of Contents | Previous | Next

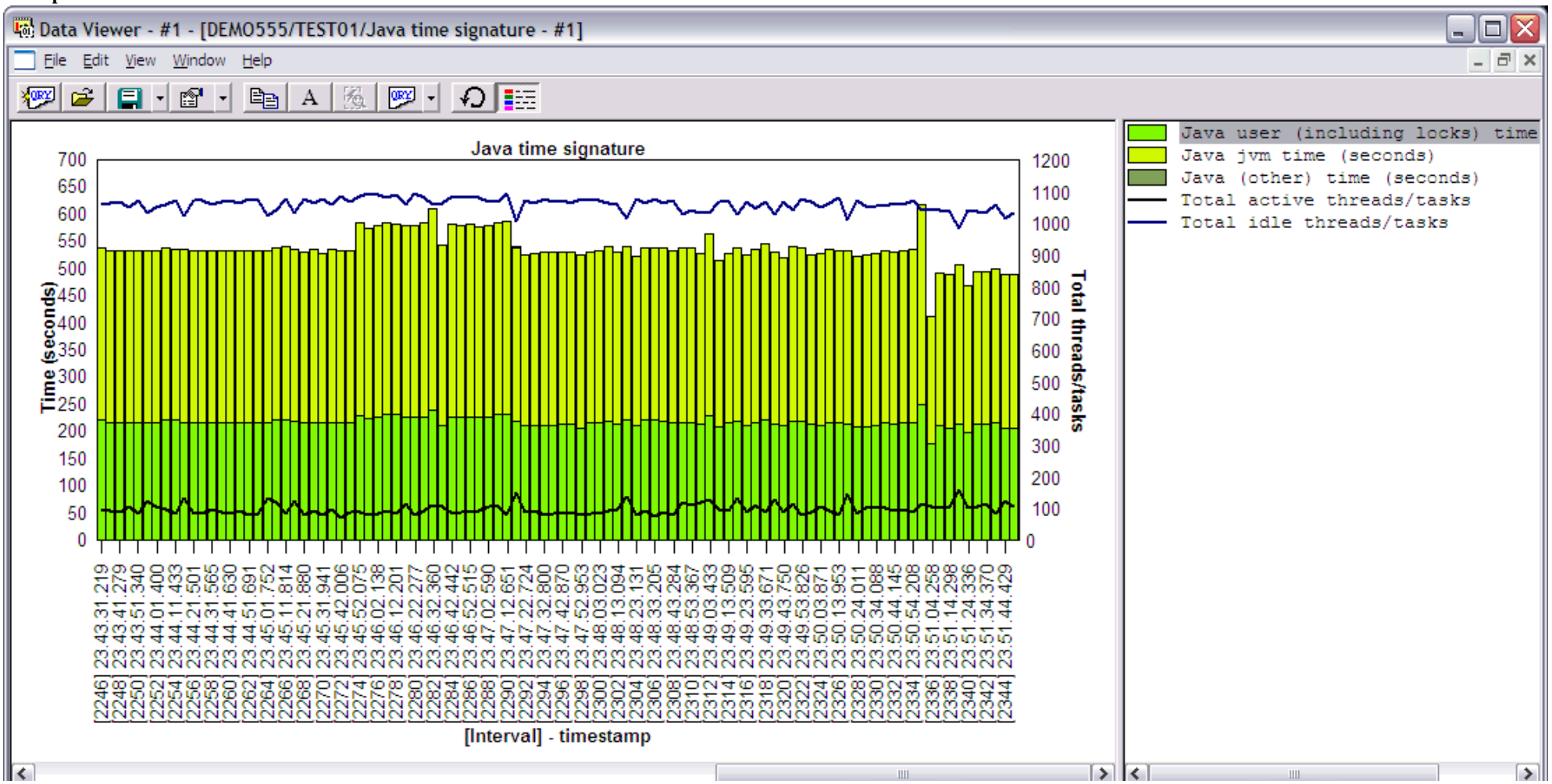
2.5.1.8 Java time signature

Description: This graph shows a **summary** of the time all jobs spent in journal operations for each interval within the job watch. The graph makes use of the 32 run/wait buckets used in Job Watcher.

This graph can be used to quickly see when Java waits occurred for jobs running on the system.

Graph Type: summarized by interval (stacked vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: The bars show the total time each interval all jobs spent in various types of Java waits.. All times are provided in seconds.
Second Y-Axis: The secondary Y-Axis displays the total number of active and idle threads/tasks on the system for each interval. Idle threads are threads that did not use CPU in the interval.

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Run/wait graphs by job	This menu allows the user to view the jobs for a paticular interval or interval range ranked by the desired wait type. By right-clicking on the graph over a particular color (such as dark blue for object locks) the user can drill into an interval to view the jobs that contributed to the selected color.
I/O graphs by job	Displays I/O and disk activity graphs for a particular interval or interval range.
Wait graphs by interval	This menu allows the user to view one of the wait graphs by interval starting at the interval range selected (or right-clicked on).
Detail reports	Displays one of the table view reports for the selected interval within the job watch.
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar in a properties window. This window will also appear by left-clicking on any bar in the graph.

Table of Contents | Previous | Next

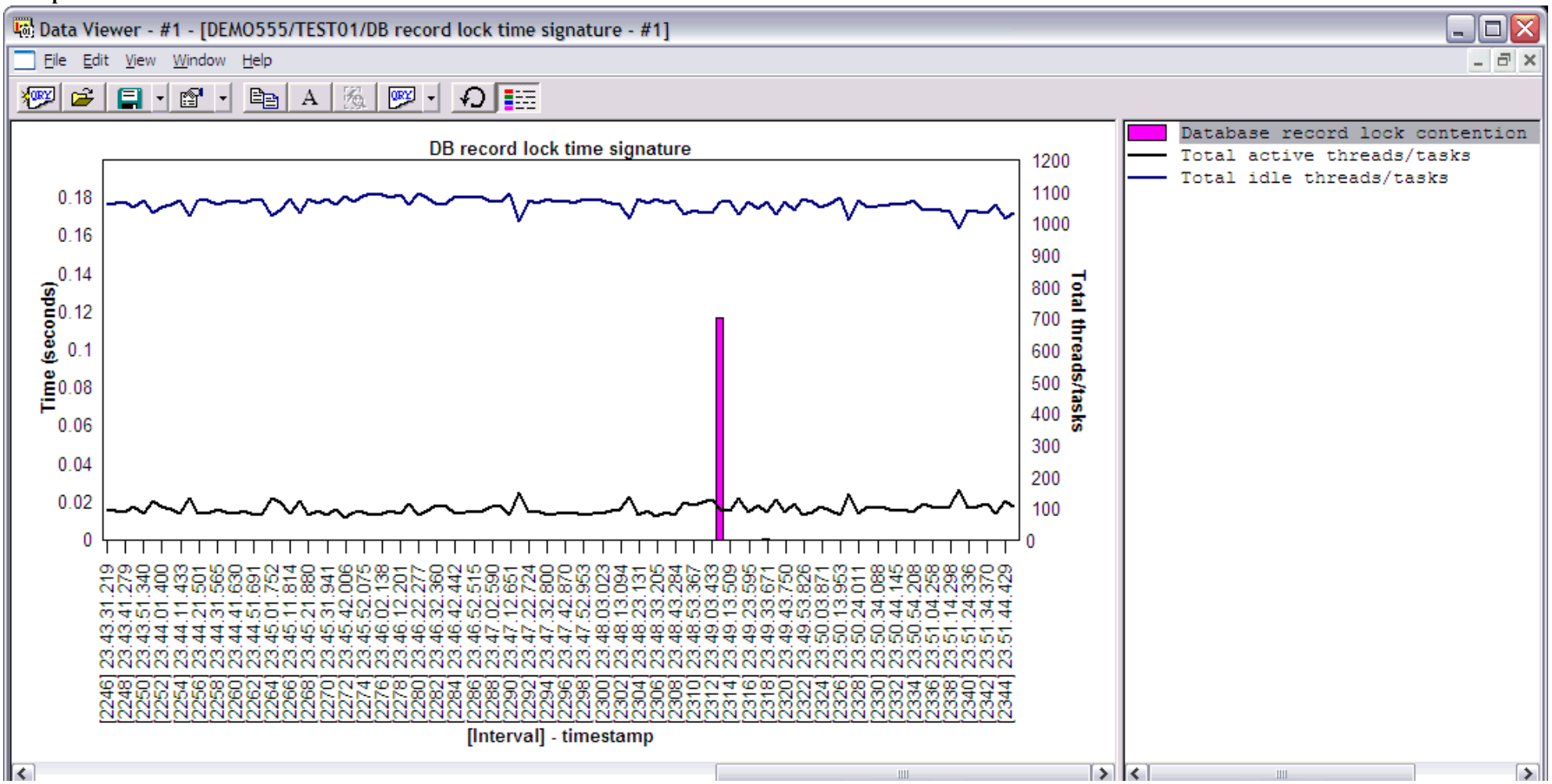
2.5.1.9 DB record lock time signature

Description: This graph shows a **summary** of the time all jobs spent in DB record lock conflicts for each interval within the job watch. The graph makes use of the 32 run/wait buckets used in Job Watcher.

This graph can be used to quickly see when DB record lock conflicts occurred within the job watch.

Graph Type: summarized by interval (stacked vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: The bars show the total time spent in DB record lock time for all jobs each interval. All times are provided in seconds.
Second Y-Axis: The secondary Y-Axis displays the total number of active and idle threads/tasks on the system for each interval. Idle threads are threads that did not use CPU in the interval.

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Run/wait graphs by job	This menu allows the user to view the jobs for a paticular interval or interval range ranked by the desired wait type. By right-clicking on the graph over a particular color (such as dark blue for object locks) the user can drill into an interval to view the jobs that contributed to the selected color.
I/O graphs by job	Displays I/O and disk activity graphs for a particular interval or interval range.
Wait graphs by interval	This menu allows the user to view one of the wait graphs by interval starting at the interval range selected (or right-clicked on).
Detail reports	Displays one of the table view reports for the selected interval within the job watch.
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar in a properties window. This window will also appear by left-clicking on any bar in the graph.

Table of Contents | Previous | Next

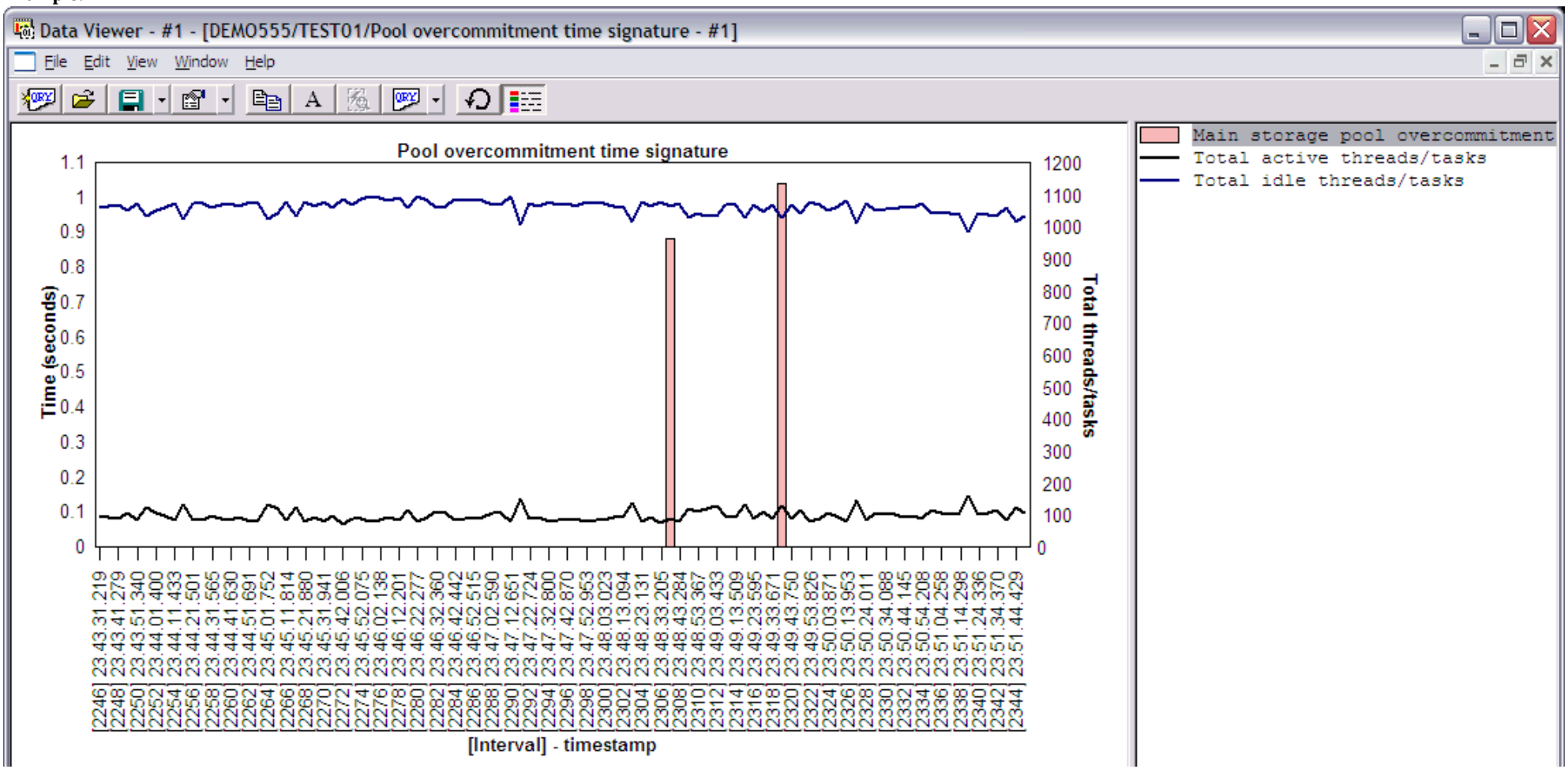
2.5.1.10 Pool overcommitment time signature

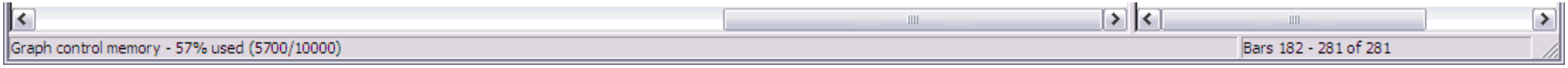
Description: This graph shows a **summary** of the time all jobs spent in pool overcommitment wait for each interval within the job watch. The graph makes use of the 32 run/wait buckets used in Job Watcher.

If these numbers are present on a graph it means that during the intervals specified the maintstore pool used by the job is full. Regular operations like DASD reads or page faults are being delayed in order to locate "free" mainstorage pool space to hold the new data.

Graph Type: summarized by interval (stacked vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: The bars in the graph represent the total amount of time spent across all threads in mainstorage pool overcommitment time. All times are provided in seconds.
Second Y-Axis: The secondary Y-Axis displays the total number of active and idle threads/tasks on the system for each interval. Idle threads are threads that did not use CPU in the interval.

Example:





Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Run/wait graphs by job	This menu allows the user to view the jobs for a paticular interval or interval range ranked by the desired wait type. By right-clicking on the graph over a particular color (such as dark blue for object locks) the user can drill into an interval to view the jobs that contributed to the selected color.
I/O graphs by job	Displays I/O and disk activity graphs for a particular interval or interval range.
Wait graphs by interval	This menu allows the user to view one of the wait graphs by interval starting at the interval range selected (or right-clicked on).
Detail reports	Displays one of the table view reports for the selected interval within the job watch.
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar in a properties window. This window will also appear by left-clicking on any bar in the graph.

Table of Contents | Previous | Next

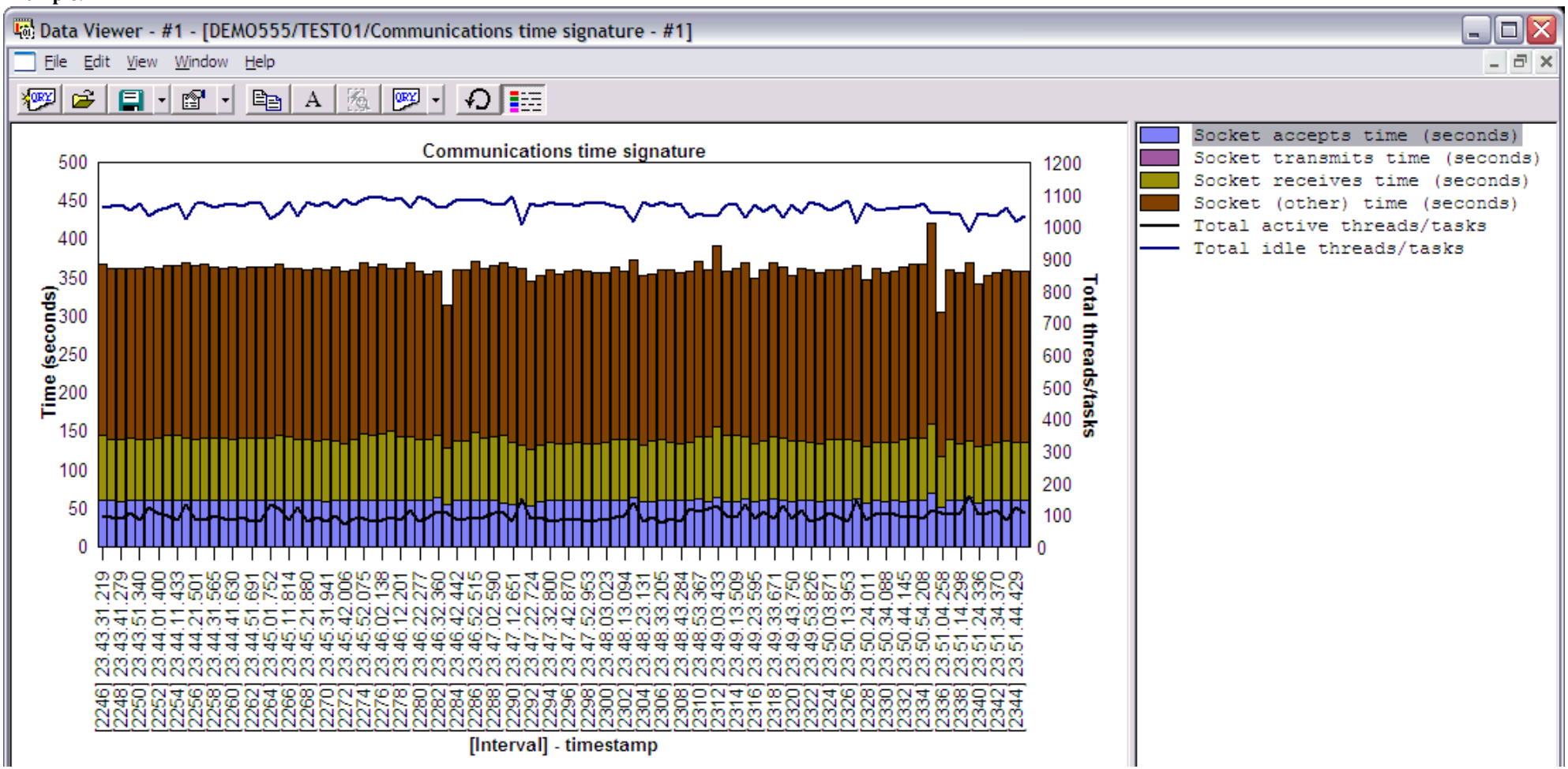
2.5.1.11 Communications time signature

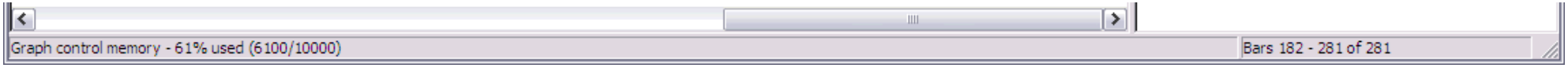
Description: This graph shows a **summary** of the time all jobs spent in communications time for each interval within the job watch. The graph makes use of the 32 run/wait buckets used in Job Watcher.

This graph shows for all jobs the total time spent sending and receiving data over a connection during each interval.

Graph Type: summarized by interval (stacked vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: Each color represents the total amount of time spent across all threads in one of the communication buckets. These buckets include socket sends and socket receives. All times are provided in seconds.
Second Y-Axis: The secondary Y-Axis displays the total number of active and idle threads/tasks on the system for each interval. Idle threads are threads that did not use CPU in the interval.

Example:





Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Run/wait graphs by job	This menu allows the user to view the jobs for a paticular interval or interval range ranked by the desired wait type. By right-clicking on the graph over a particular color (such as dark blue for object locks) the user can drill into an interval to view the jobs that contributed to the selected color.
I/O graphs by job	Displays I/O and disk activity graphs for a particular interval or interval range.
Wait graphs by interval	This menu allows the user to view one of the wait graphs by interval starting at the interval range selected (or right-clicked on).
Detail reports	Displays one of the table view reports for the selected interval within the job watch.
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar in a properties window. This window will also appear by left-clicking on any bar in the graph.

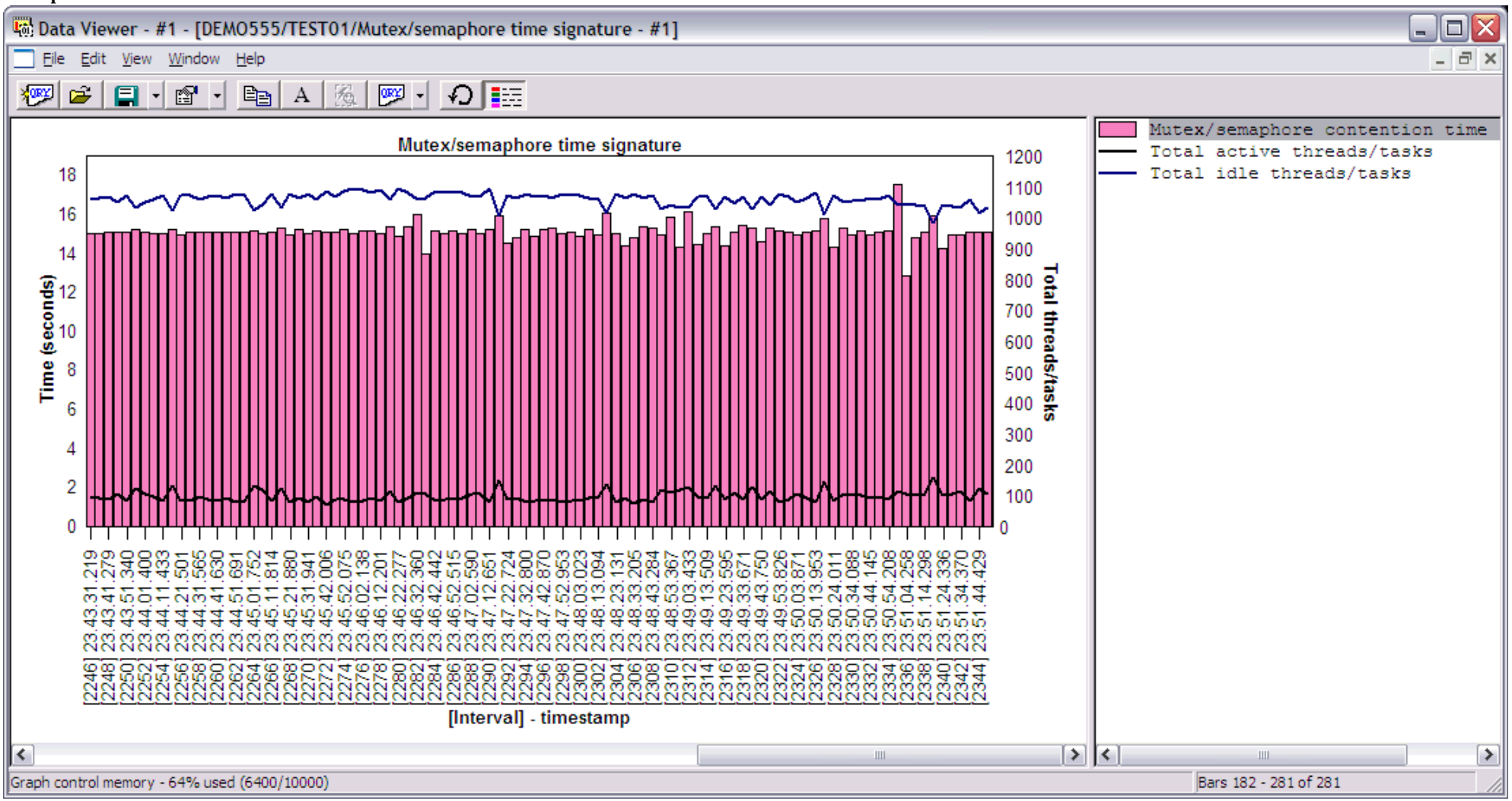
Table of Contents | Previous | Next

2.5.1.12 Mutex/semaphore time signature

Description: This graph shows a **summary** of the time all jobs spent in mutex or semaphore waits for each interval within the job watch. The graph makes use of the 32 run/wait buckets used in Job Watcher.

Graph Type: summarized by interval (stacked vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: Each color represents the total amount of time spent across all threads in the mutex/semaphore bucket. Times are shown in seconds.
Second Y-Axis: The secondary Y-Axis displays the total number of active and idle threads/tasks on the system for each interval. Idle threads are threads that did not use CPU in the interval.

Example:



Right-clicking on a bar in this graph provides the following menu options:

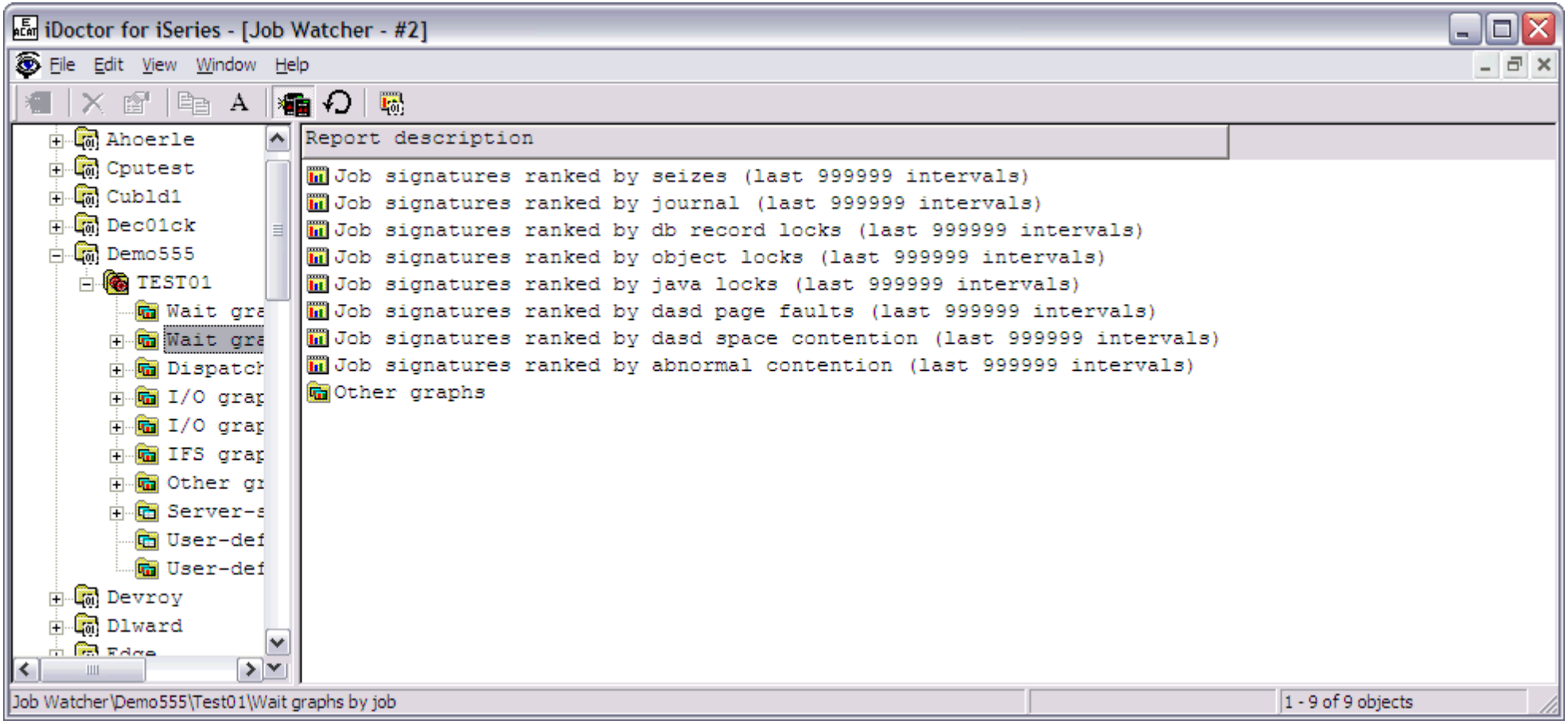
Menu	Description
Run/wait graphs by job	This menu allows the user to view the jobs for a paticular interval or interval range ranked by the desired wait type. By right-clicking on the graph over a particular color (such as dark blue for object locks) the user can drill into an interval to view the jobs that contributed to the selected color.
I/O graphs by job	Displays I/O and disk activity graphs for a particular interval or interval range.
Wait graphs by interval	This menu allows the user to view one of the wait graphs by interval starting at the interval range selected (or right-clicked on).
Detail reports	Displays one of the table view reports for the selected interval within the job watch.
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar in a properties window. This window will also appear by left-clicking on any bar in the graph.

Table of Contents | Previous | Next

2.5.2 Wait graphs by job

This folder contains a list of graphs which contain job summary data over the collection relating to waits. These graphs display a bar per job showing a run/wait time signature sorted by the type of wait indicated in the graph name.

The graphs in this folder are listed with the most commonly used wait types first and the other wait types are available in the other graphs folder within this list. The screenshot below shows this expanded wait graphs by job folder.



The following table illustrates the menu options available by right-clicking on a graph in the list.

Menu	Description

Open graph(s)	Opens the selected graph(s) into a Data Viewer. If a Data Viewer has already been opened submenus will appear underneath this menu in order to give the option of opening the graph(s) into an already open Data Viewer.

Table of Contents | Previous | Next

2.5.2.1 Job signatures ranked by <wait type>

Description: This graph shows a **summary** of all jobs on the system ranked by the total time spent in a certain type of wait (like a sezie condition) over an interval range. Each bar in the graph represents a job and the colors in the bar represent various types of wait time the job spent during its existance in the job watch. The graph makes use of the 32 run/wait buckets used in Job Watcher.

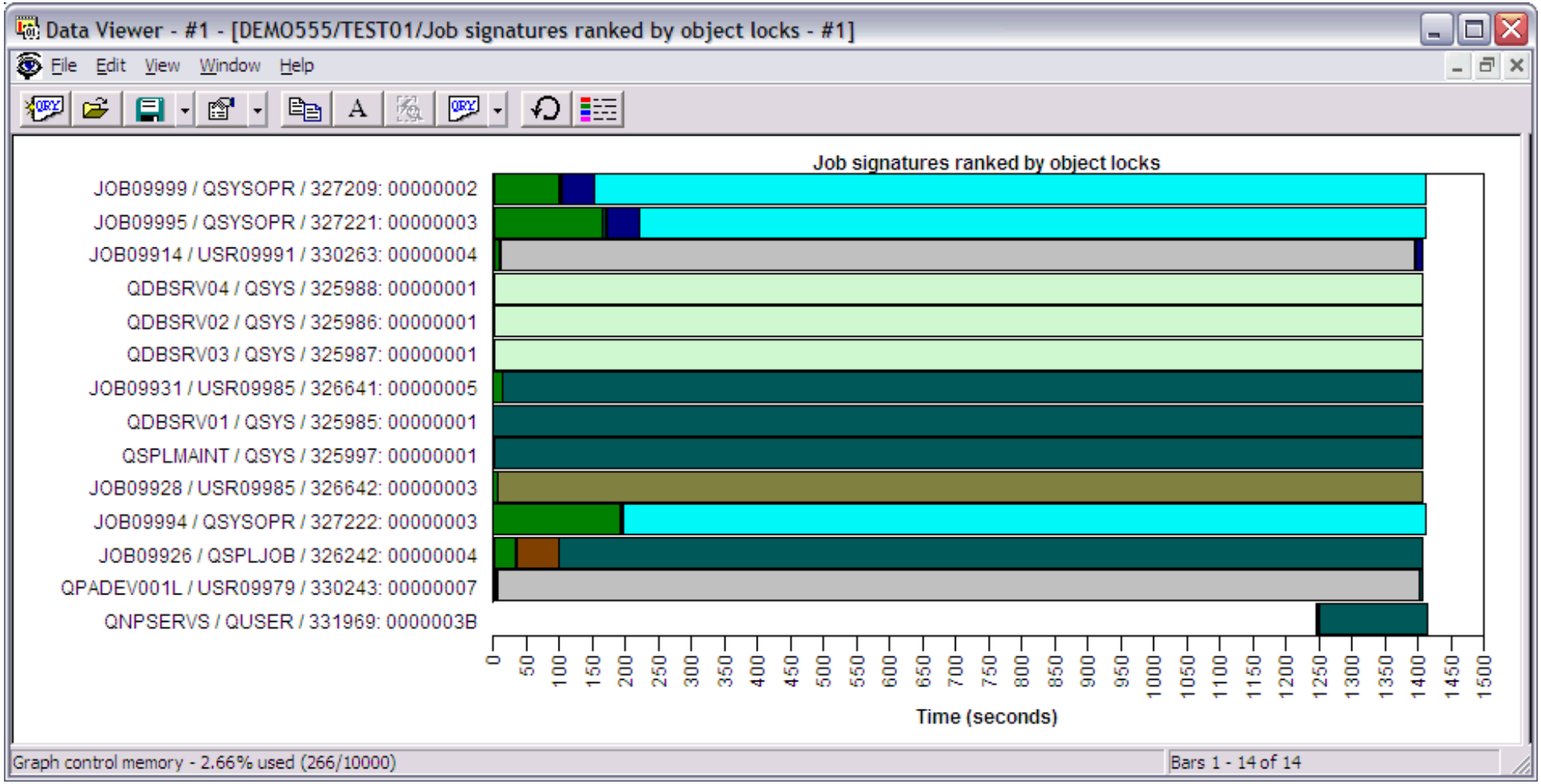
This graph is a good starting point because it provides a high-level overview of which jobs were performing the desired wait type. The total length of the bar indicates the total time the job was collected by the job watch over the interval range specified by the report title.

This graph will look quite different depending on if the collection has been summarized or not. Idle wait times are not included in the wait numbers if the collection has not been summarized. If this is the case bar lengths will be usually much shorter. **Summarizing the collection is highly recommended before viewing this graph.**

Bars that are indented indicate that the job/task began execution later in time in relation to the other jobs shown. Bars are only indented if the collection has been summarized.

Graph Type: summarized by job (stacked horizontal bar)
X-axis: Job name and thread ID or task name and taskcount.
Y-Axis: Each color represents a type of wait. Place the mouse pointer over a bar to see a description of the wait type. All times are provided in seconds.

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Display call stack	Shows the call stack within the interval detail property pages for the job selected.
Wait graphs	Displays one of the detailed wait graphs for the job selected.
Dispatched CPU graphs	Displays one of the detailed CPU graphs for the job selected.
DASD/IO graphs	Displays one of the detailed DASD/IO graphs for the job selected.
IFS graphs	Displays one of the detailed IFS graphs for the job selected.

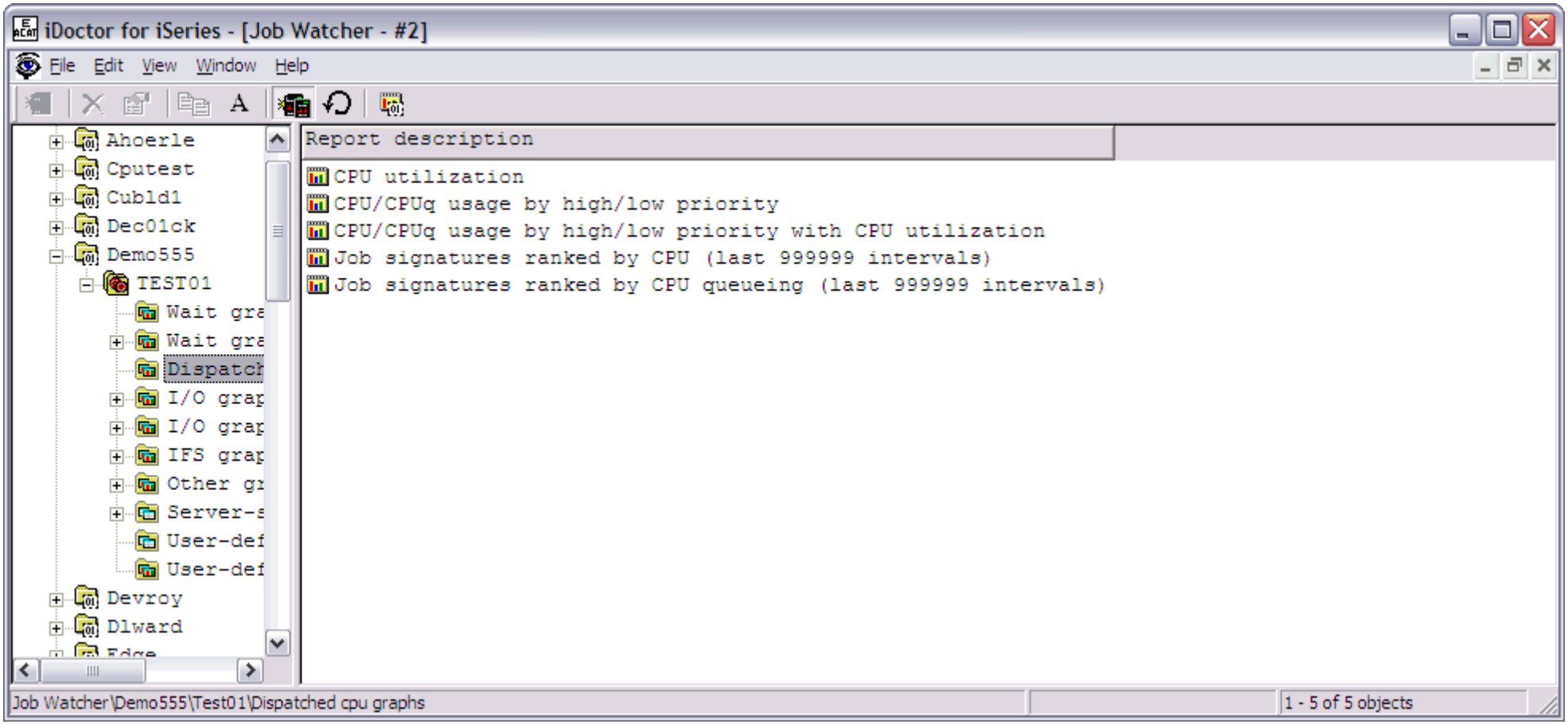
Other graphs	Displays one of the other detailed graphs for the job selected. This category includes transactions and state transitions.
Wait graphs by interval	This menu allows the user to view one of the wait graphs by interval starting at the interval range selected (or right-clicked on).
Detail records	Displays a table view showing all records in the QAPYJWTDE file for the selected job/thread or task.
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar in a properties window. This window will also appear by left-clicking on any bar in the graph.

Table of Contents | Previous | Next

2.5.3 Dispatched CPU graphs

This folder contains a list of graphs which contain summary data over the job watch showing CPU and CPU queueing times. This folder also contains graphs for job run/wait time signatures ranked by CPU and CPU queueing.

An example of the contents of the CPU graphs folder for a job watch is:



The following table illustrates the menu options available by right-clicking on a graph in the list.

Menu	Description

Open graph(s)	Opens the selected graph(s) into a Data Viewer. If a Data Viewer has already been opened submenus will appear underneath this menu in order to give the option of opening the graph(s) into an already open Data Viewer.

Table of Contents | Previous | Next

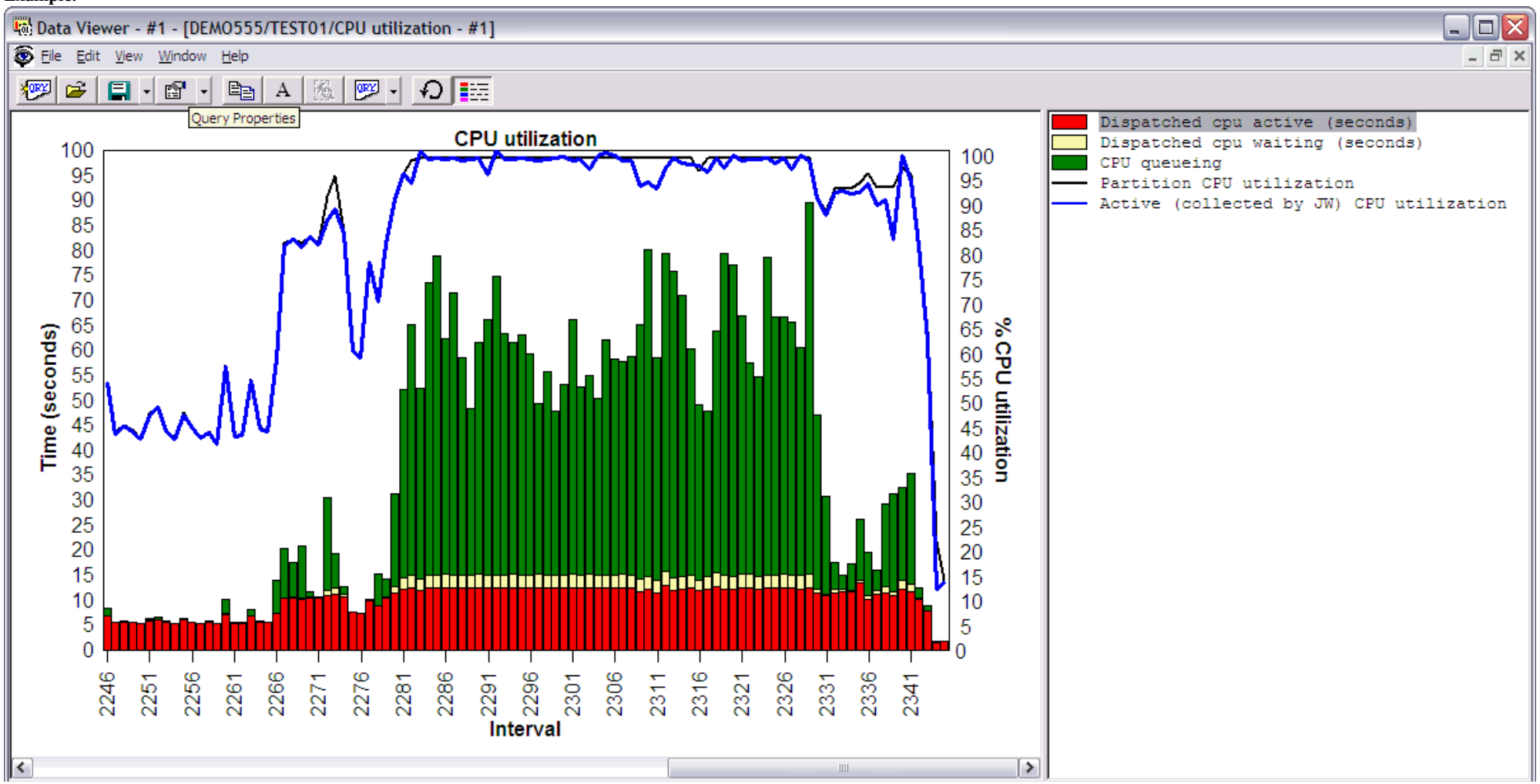
2.5.3.1 CPU utilization

Description: This graph shows a **summary** of the CPU and CPU queueing times as well as CPU utilization during the job watch. The red bars show CPU times and the green bars show CPU queueing times.

This graph is only shown if the collection has been summarized.

Graph Type: summarized by interval (stacked vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: The bars show dispatched CPU active, dispatched CPU waiting and CPU queueing time. All times are provided in seconds.
Second Y-Axis: The secondary Y-Axis displays partition CPU utilization and active CPU utilization. Active CPU utilization is generally not as accurate because it's based only on the jobs sampled by Job Watcher. Job Watcher will not collect information for jobs/tasks the started and ending within a single interval (between snapshots). These missed jobs account for the difference between partition CPU utilization (true CPU utilization) and active CPU utilization.

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Run/wait graphs by job	This menu allows the user to view the jobs for a paticular interval or interval range ranked by the desired wait type. By right-clicking on the graph over a particular color (such as dark blue for object locks) the user can drill into an interval to view the jobs that contributed to the selected color.
I/O graphs by job	Displays I/O and disk activity graphs for a particular interval or interval range.
Wait graphs by interval	This menu allows the user to view one of the wait graphs by interval starting at the interval range selected (or right-clicked on).
Detail reports	Displays one of the table view reports for the selected interval within the job watch.
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar in a properties window. This window will also appear by left-clicking on any bar in the graph.

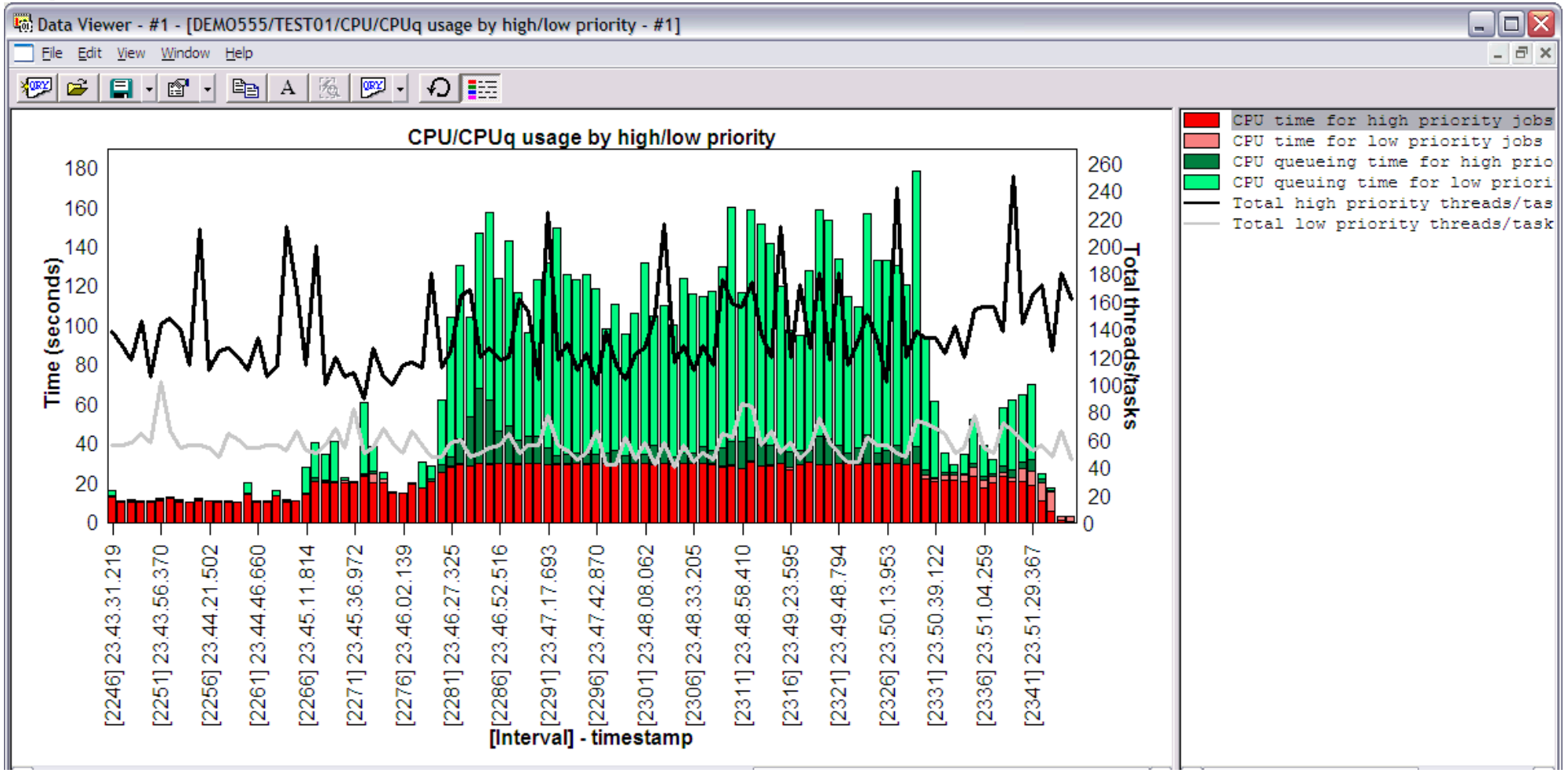
2.5.3.2 CPU/CPUq usage by high/low priority

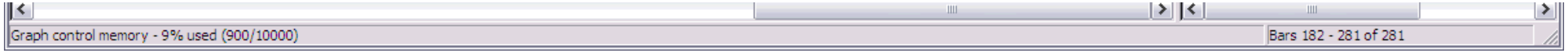
Description: This graph shows a **summary** of the CPU and CPU queueing times used by high priority (<= 29) and low priority (> 29) jobs during the job watch. The red bars show CPU times and the green bars show CPU queueing times.

The priority values are configurable on the field selection page of the query definition interface by modifying the SQL expression for fields CPUHIGH, CPULOW, CPUQHIGH and CPUQLOW once the graph has been opened. The priority field in the data is a LIC priority which is 140 higher than XPF priority. For this reason 169 instead of 29 is used to make the comparison in the SQL expressions in the query used behind the graph.

Graph Type: summarized by interval (stacked vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: The bars show CPU and CPU queueing time. All times are provided in seconds.
Second Y-Axis: The secondary Y-Axis displays the total number of active threads/tasks on the system for each interval.

Example:





Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Run/wait graphs by job	This menu allows the user to view the jobs for a paticular interval or interval range ranked by the desired wait type. By right-clicking on the graph over a particular color (such as dark blue for object locks) the user can drill into an interval to view the jobs that contributed to the selected color.
I/O graphs by job	Displays I/O and disk activity graphs for a particular interval or interval range.
Wait graphs by interval	This menu allows the user to view one of the wait graphs by interval starting at the interval range selected (or right-clicked on).
Detail reports	Displays one of the table view reports for the selected interval within the job watch.
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar in a properties window. This window will also appear by left-clicking on any bar in the graph.

2.5.3.3 CPU/CPUq usage by high/low priority with CPU utilization

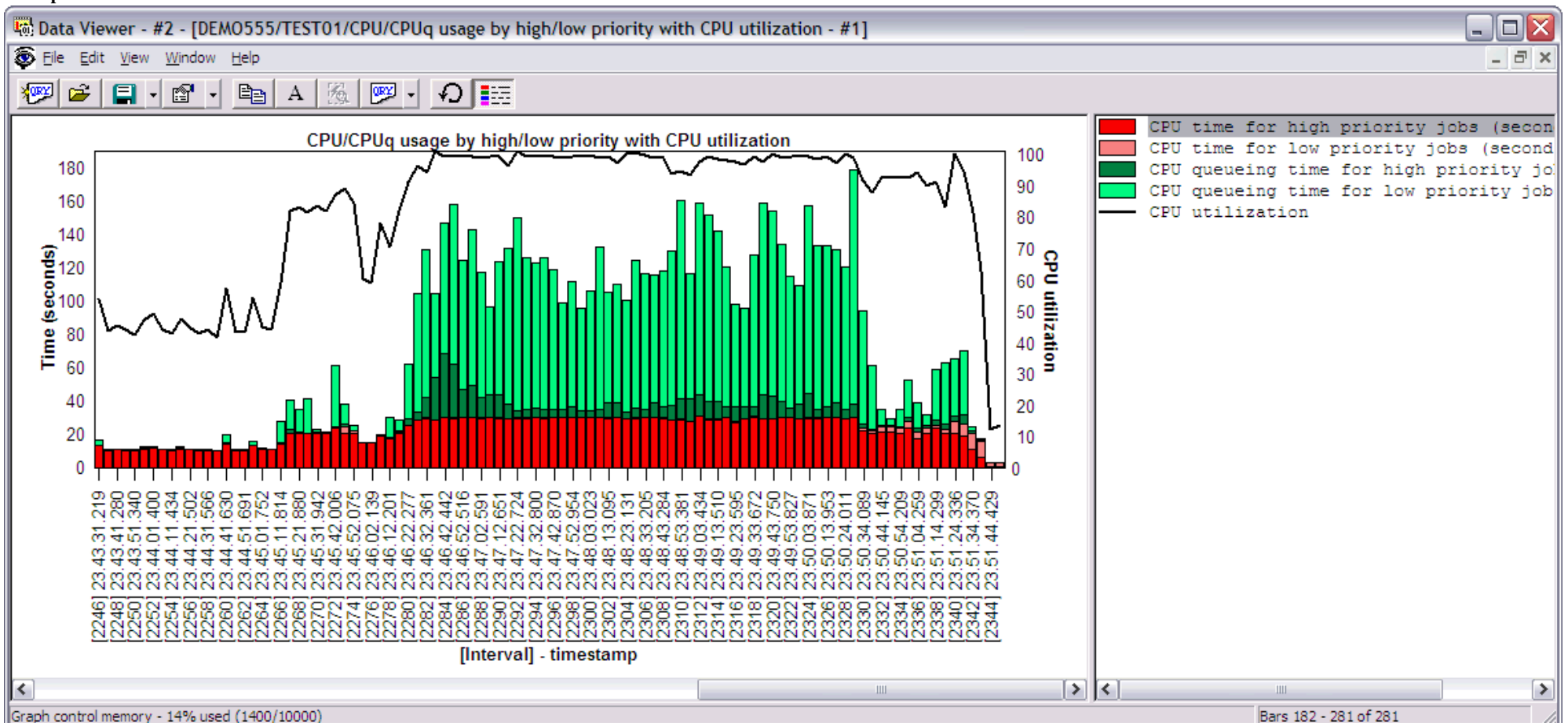
Description: This graph shows a **summary** of the CPU and CPU queueing times used by high priority (<= 29) and low priority (> 29) jobs during the job watch. The red bars show CPU times and the green bars show CPU queueing times.

The priority values are configurable on the field selection page of the query definition interface by modifying the SQL expression for fields CPUHIGH, CPULOW, CPUQHIGH and CPUQLOW once the graph has been opened. The priority field in the data is a LIC priority which is 140 higher than XPF priority. For this reason 169 instead of 29 is used to make the comparison in the SQL expressions in the query used behind the graph.

This graph is only shown if the collection has been summarized.

Graph Type: summarized by interval (stacked vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: The bars show CPU and CPU queueing times. All times are provided in seconds.
Second Y-Axis: The secondary Y-Axis displays the CPU utilization for each interval of the collection.

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Run/wait graphs by job	This menu allows the user to view the jobs for a paticular interval or interval range ranked by the desired wait type. By right-clicking on the graph over a particular color (such as dark blue for object locks) the user can drill into an interval to view the jobs that contributed to the selected color.
I/O graphs by job	Displays I/O and disk activity graphs for a particular interval or interval range.
Wait graphs by interval	This menu allows the user to view one of the wait graphs by interval starting at the interval range selected (or right-clicked on).
Detail reports	Displays one of the table view reports for the selected interval within the job watch.
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar in a properties window. This window will also appear by left-clicking on any bar in the graph.

Table of Contents | Previous | Next

2.5.3.4 Job signatures ranked by CPU

Description: This graph shows a **summary** of the jobs in the job watch that used the most CPU. Each bar in the graph represents a job and the colors in the bar represent various types of wait time the job spent during its existance in the job watch. The graph makes use of the 32 run/wait buckets used in Job Watcher.

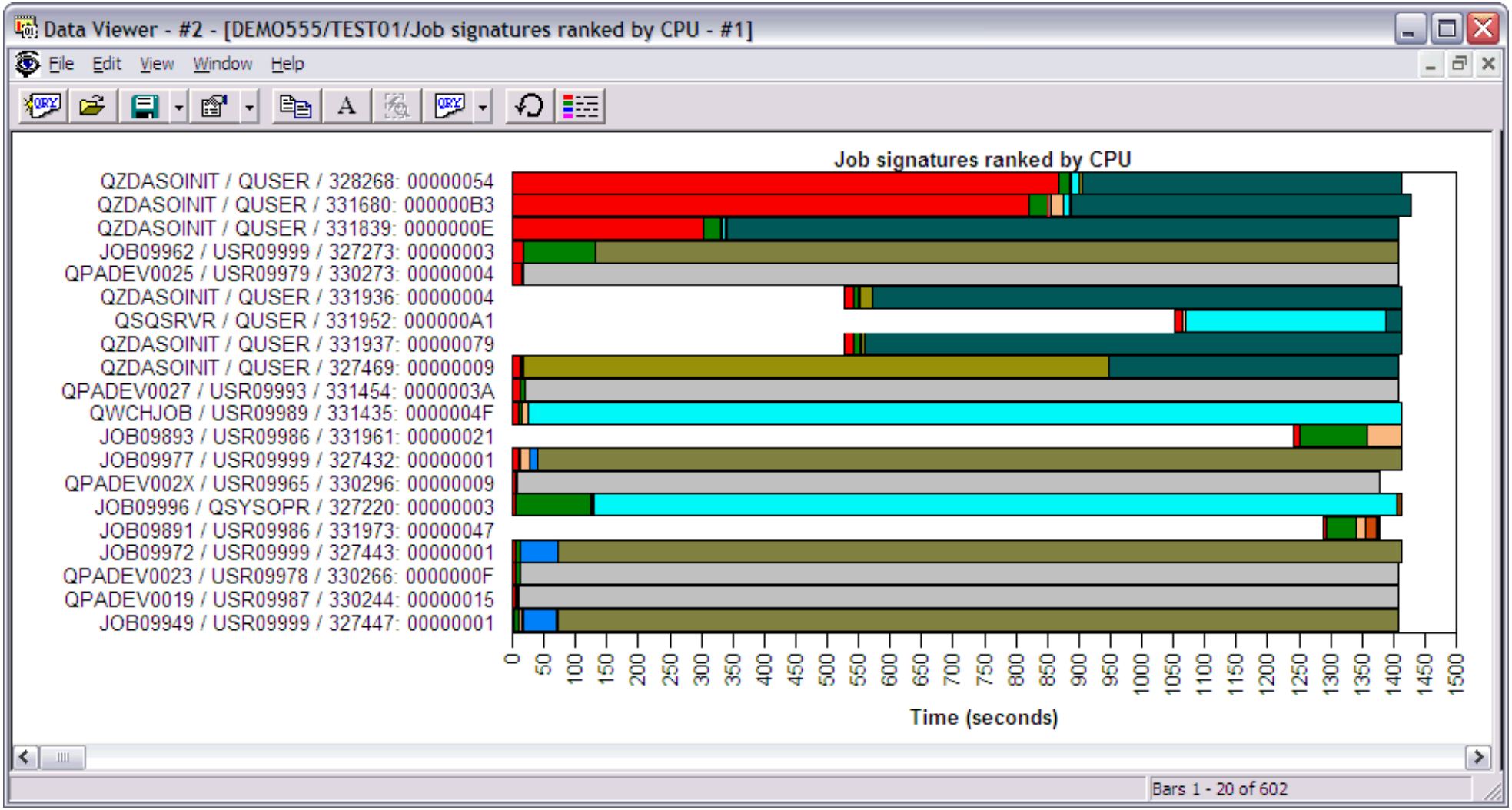
This graph is a good starting point because it provides a high-level overview of which jobs were using the most CPU. The total length of the bar indicates the total time the job was collected by the job watch over the interval range specified by the report title.

This graph will look quite different depending on if the collection has been summarized or not. Idle wait times are not included in the wait numbers if the collection has not been summarized. If this is the case bar lengths will be usually much shorter. **Summarizing the collection is highly recommended before viewing this graph.**

Bars that are indented indicate that the job/task began execution later in time in relation to the other jobs shown. Bars are only indented if the collection has been summarized.

Graph Type: summarized by job (horizontal stacked bar)
X-axis: Job name and thread ID or task name and taskcount.
Y-Axis: Red in the bars indicate CPU time. Other colors show times for other types of waits. Place the mouse pointer over a bar to see a description of the wait type. All times are provided in seconds.

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Display call stack	Shows the call stack within the interval detail property pages for the job selected.
Wait graphs	Displays one of the detailed wait graphs for the job selected.
Dispatched CPU graphs	Displays one of the detailed CPU graphs for the job selected.
DASD/IO graphs	Displays one of the detailed DASD/IO graphs for the job selected.

IFS graphs	Displays one of the detailed IFS graphs for the job selected.
Other graphs	Displays one of the other detailed graphs for the job selected. This category includes transactions and state transitions.
Wait graphs by interval	This menu allows the user to view one of the wait graphs by interval starting at the interval range selected (or right-clicked on).
Detail records	Displays a table view showing all records in the QAPYJWTDE file for the selected job/thread or task.
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar in a properties window. This window will also appear by left-clicking on any bar in the graph.

Table of Contents | Previous | Next

2.5.3.5 Job signatures ranked by CPU queueing

Description: This graph shows a **summary** of the jobs in the job watch that had the most CPU queueing (spent time waiting to use the CPU). Each bar in the graph represents a job and the colors in the bar represent various types of wait time the job spent during its existance in the job watch. The graph makes use of the 32 run/wait buckets used in Job Watcher.

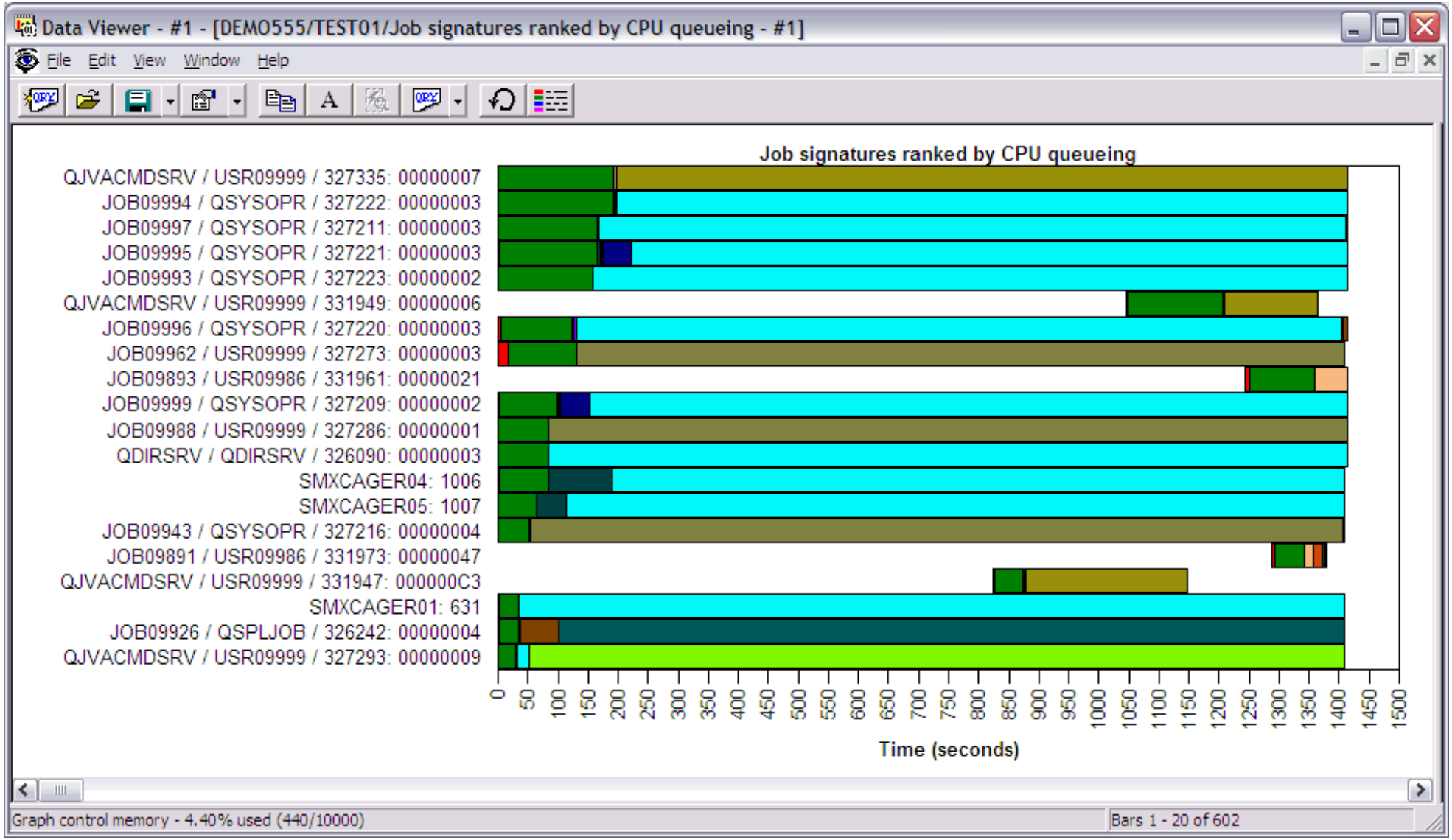
The total length of the bar indicates the total time the job was collected by the job watch over the interval range specified by the report title.

This graph will look quite different depending on if the collection has been summarized or not. Idle wait times are not included in the wait numbers if the collection has not been summarized. If this is the case bar lengths will be usually much shorter. **Summarizing the collection is highly recommended before viewing this graph.**

Bars that are indented indicate that the job/task began execution later in time in relation to the other jobs shown. Bars are only indented if the collection has been summarized.

Graph Type: summarized by job (horizontal stacked bar)
X-axis: Job name and thread ID or task name and taskcount.
Y-Axis: Green in the bars indicate CPU queueing time. Red colors represent CPU time. Other colors show times for other types of waits. Place the mouse pointer over a bar to see a description of the wait type. All times are provided in seconds.

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Display call stack	Shows the call stack within the interval detail property pages for the job selected.
Wait graphs	Displays one of the detailed wait graphs for the job selected.
Dispatched CPU graphs	Displays one of the detailed CPU graphs for the job selected.

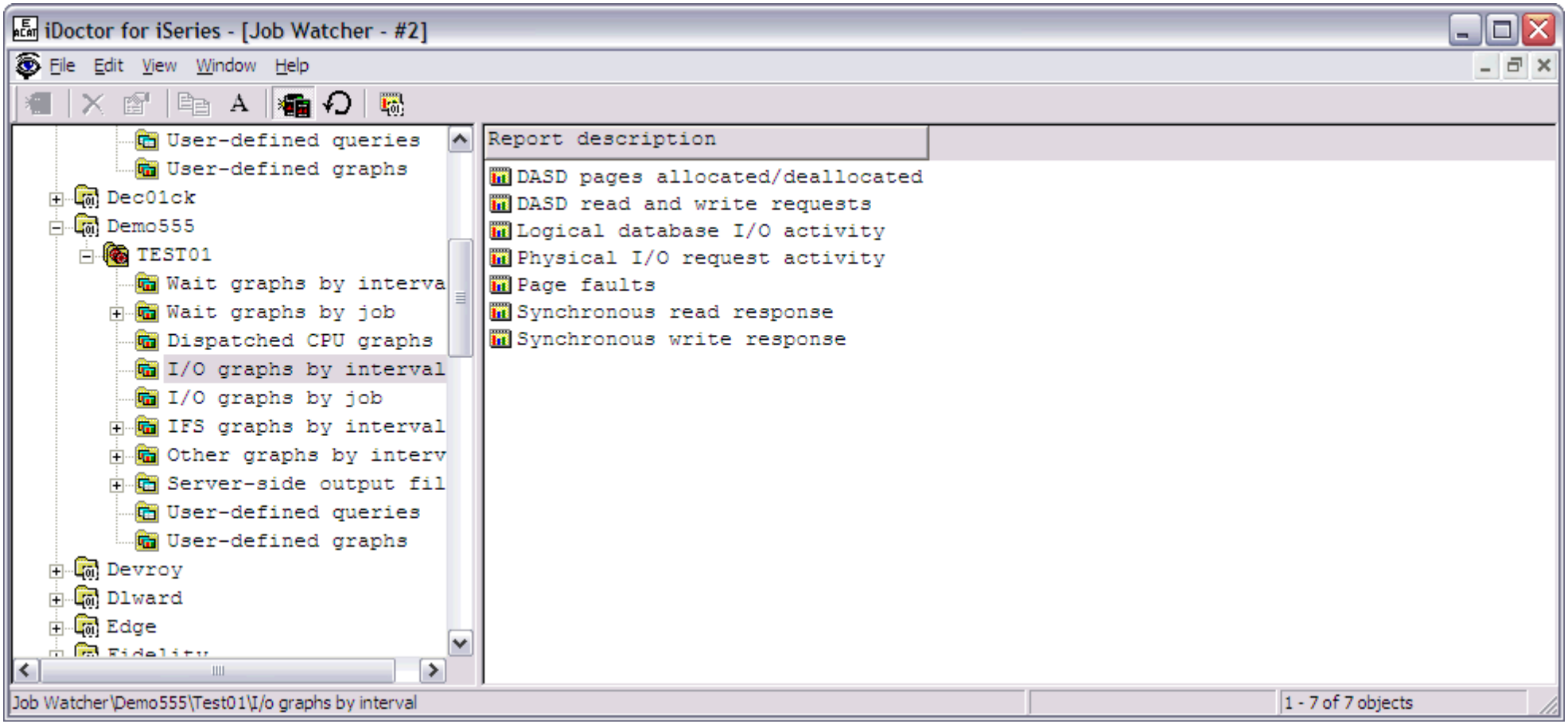
DASD/IO graphs	Displays one of the detailed DASD/IO graphs for the job selected.
IFS graphs	Displays one of the detailed IFS graphs for the job selected.
Other graphs	Displays one of the other detailed graphs for the job selected. This category includes transactions and state transitions.
Wait graphs by interval	This menu allows the user to view one of the wait graphs by interval starting at the interval range selected (or right-clicked on).
Detail records	Displays a table view showing all records in the QAPYJWTDE file for the selected job/thread or task.
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar in a properties window. This window will also appear by left-clicking on any bar in the graph.

Table of Contents | Previous | Next

2.5.4 I/O graphs by interval

This folder contains a list of graphs which contain summary data over the job watch relating to DASD/IO usage. These graphs display a bar per interval showing a type of physical IO or disk activity.

An example of the contents of the I/O graphs by interval folder is:



The following table illustrates the menu options available by right-clicking on a graph in the list.

Menu	Description

Open graph(s)	Opens the selected graph(s) into a Data Viewer. If a Data Viewer has already been opened submenus will appear underneath this menu in order to give the option of opening the graph(s) into an already open Data Viewer.

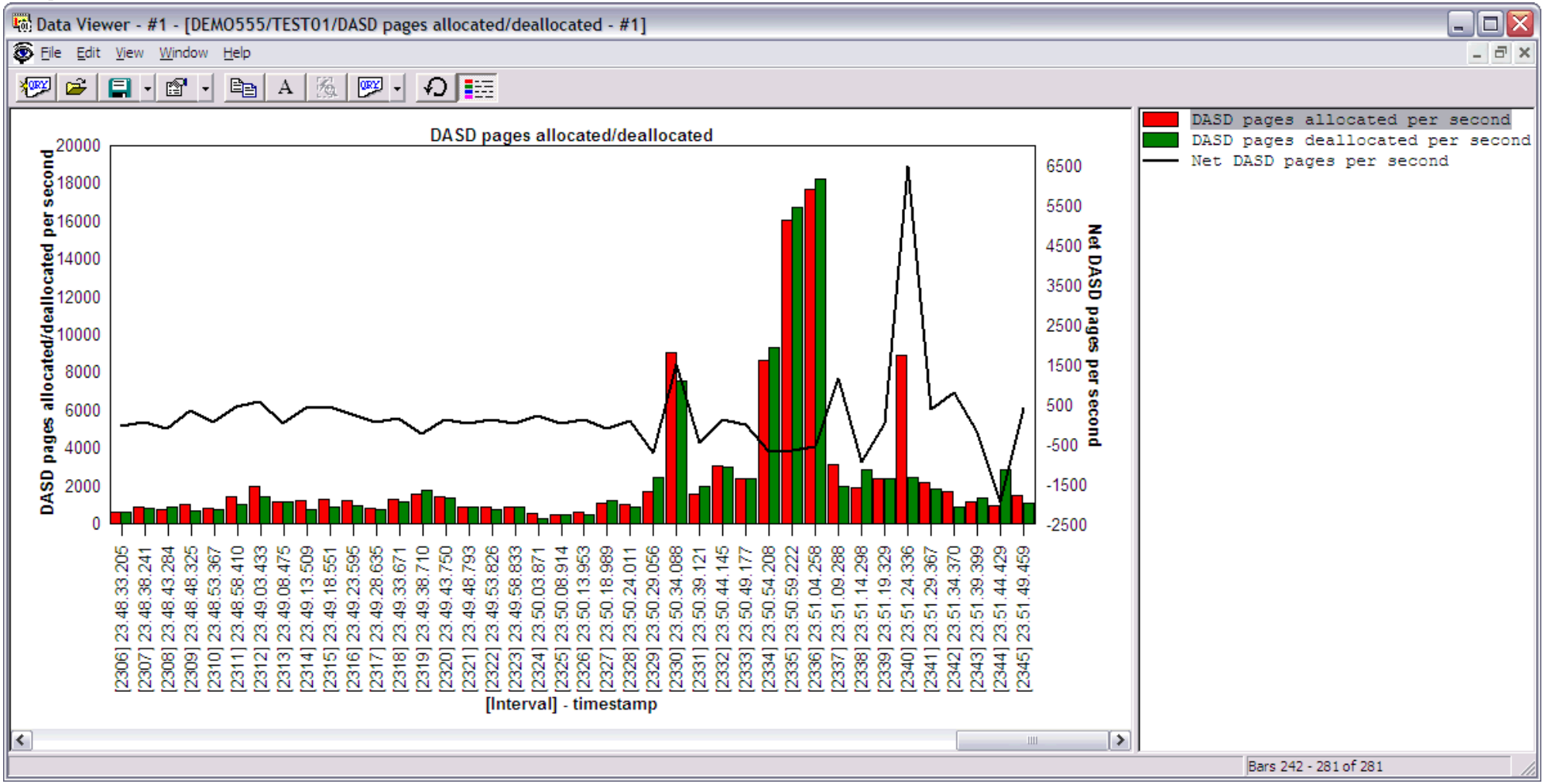
2.5.4.1 DASD pages allocated/deallocated

Description: This graph shows a **summary** of the rates of DASD pages allocated/deallocated each interval in the job watch.. The bars in the graph show rates of pages allocated and deallocated.

This graph can be useful to see when high DASD space allocations or deallocations were made on the system.

Graph Type: summarized by interval (vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: The bars show the rate of 4K DASD pages allocated and deallocated. Red bars show allocations and green bars show deallocations. All rates are 4k DASD pages per second.
Second Y-Axis: The line shows the net rate of pages allocated/deallocated each interval.

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Detail reports	Displays one of the table view reports for the selected interval within the job watch.
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar in a properties window. This window will also appear by left-clicking on any bar in the graph.

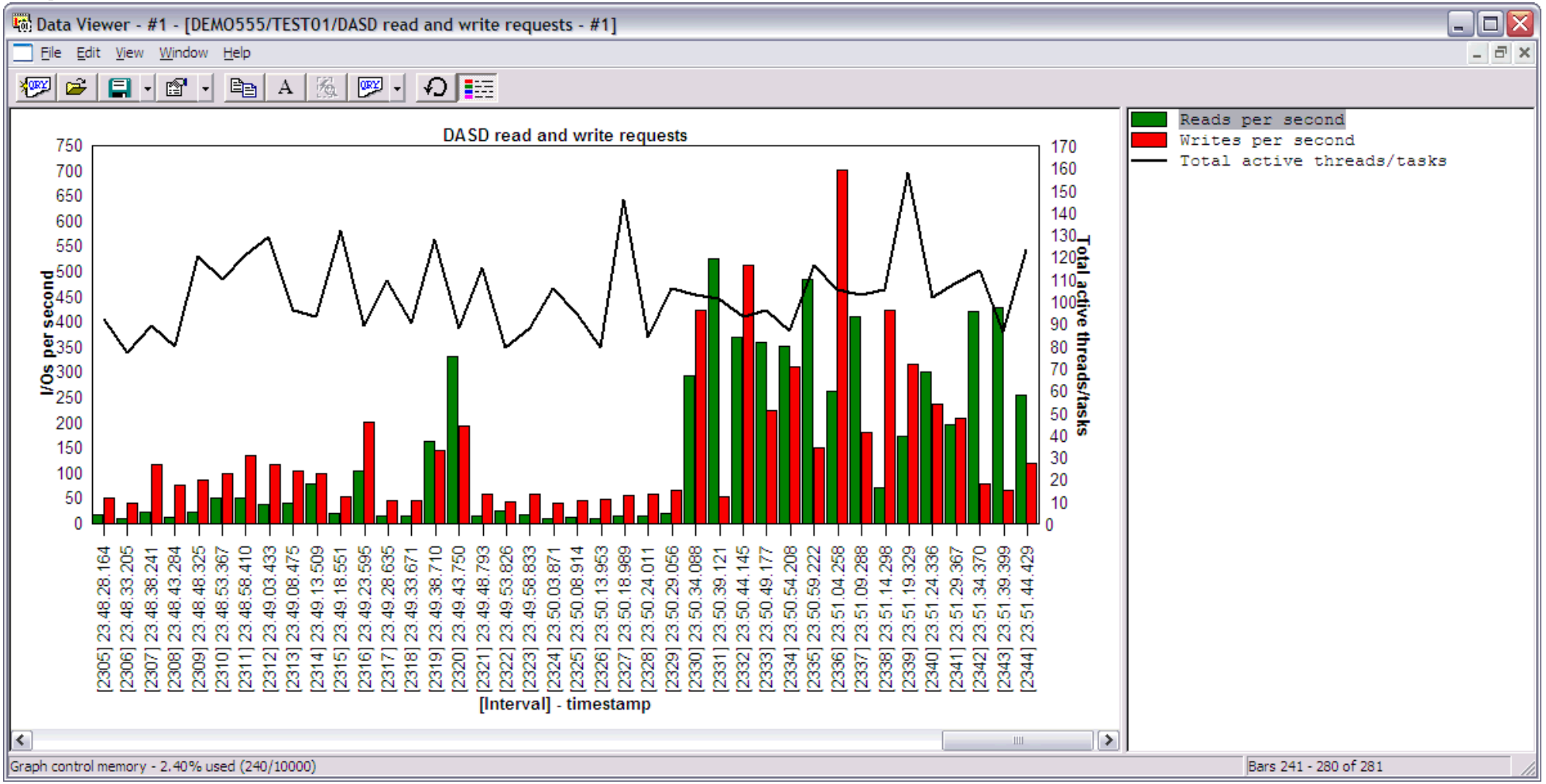
2.5.4.2 DASD reads and write requests

Description: This graph shows a **summary** of the rates of DASD reads and writes that occurred each interval in the job watch. The red bars show reads per second and the green bars show writes per second.

This graph can be useful to see when high disk reads and writes were made on the system.

Graph Type: summarized by interval (vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: The bars show rates of disk reads and writes per second.
Second Y-Axis: The line shows the total active threads/tasks for each interval.

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Detail reports	Displays one of the table view reports for the selected interval within the job watch.
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar in a properties window. This window will also appear by left-clicking on any bar in the graph.

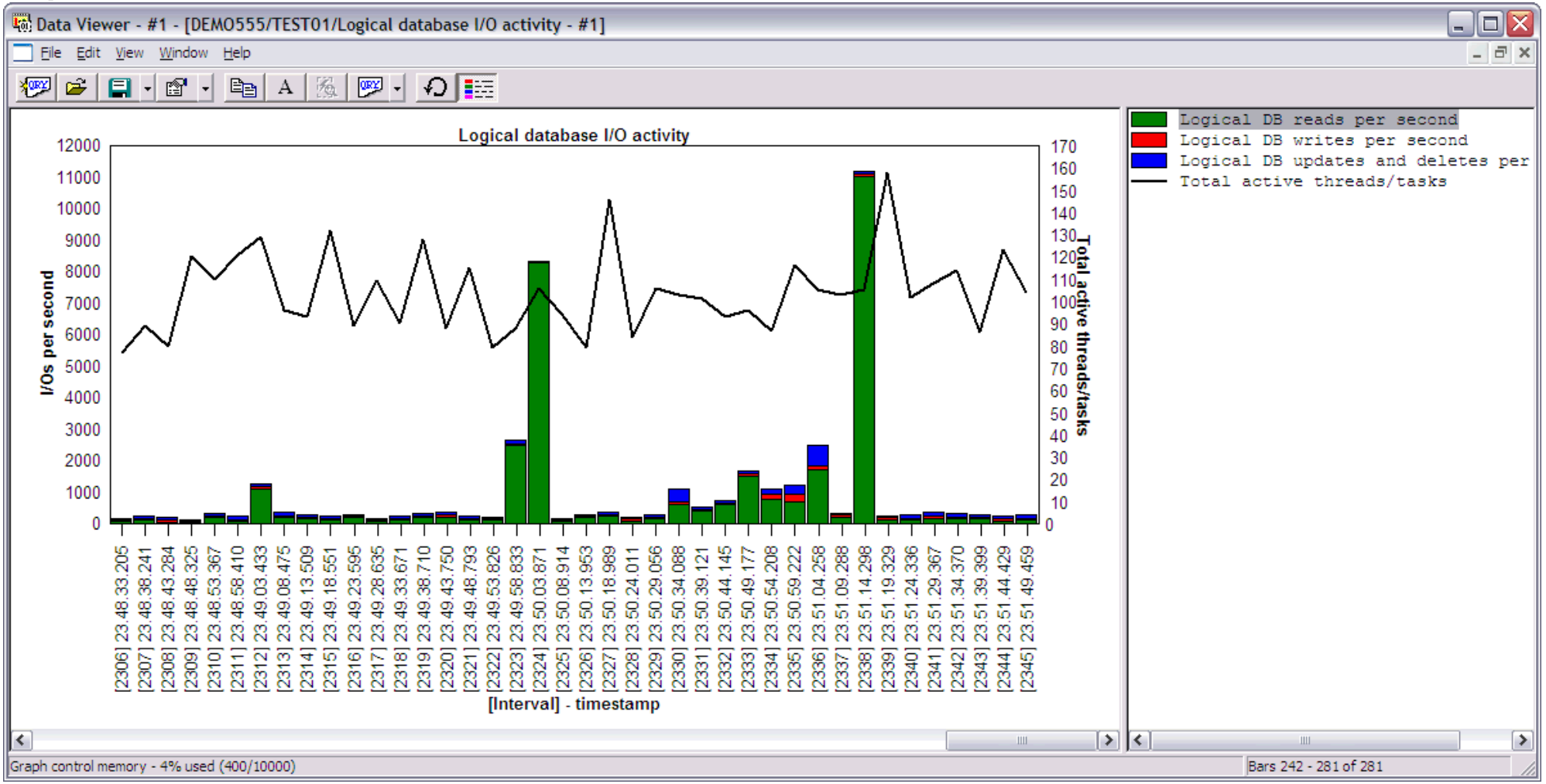
Table of Contents | Previous | Next

2.5.4.3 Logical database I/O activity

Description: This graph shows a **summary** of I/O activity for each interval in the job watch.. The bars in the graph show rates of logical DB reads, writes and updates/deletes per second.

Graph Type: summarized by interval (stacked vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: The bars show various types of logical database IO activity in I/Os per second.
Second Y-Axis: The line shows the total active threads/tasks for each interval.

Example:



Right-clicking on a bar in this graph provides the following menu options:

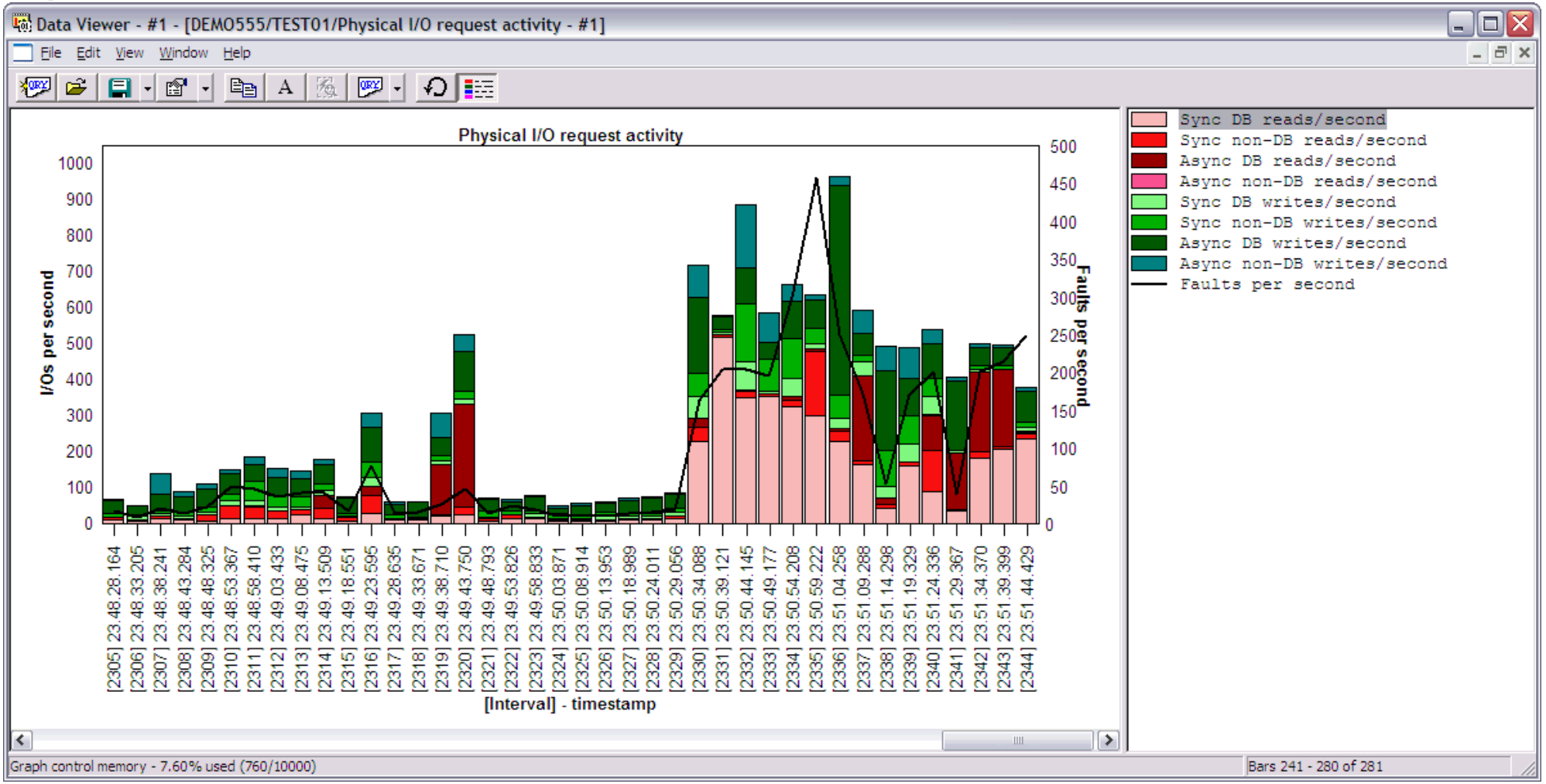
Menu	Description
Detail reports	Displays one of the table view reports for the selected interval within the job watch.
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar in a properties window. This window will also appear by left-clicking on any bar in the graph.

2.5.4.4 Physical I/O request activity

Description: This graph shows a **summary** of the rates of synchronous and asycnhronous, database and non-database reads and writes per second for each interval in the job watch.

Graph Type: summarized by interval (stacked vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: The bars shows collection-wide rates of physical I/O activity.
Second Y-Axis: The secondary Y-Axis displays the faults occuring collection-wide per second.

Example:



Right-clicking on a bar in this graph provides the following menu options:

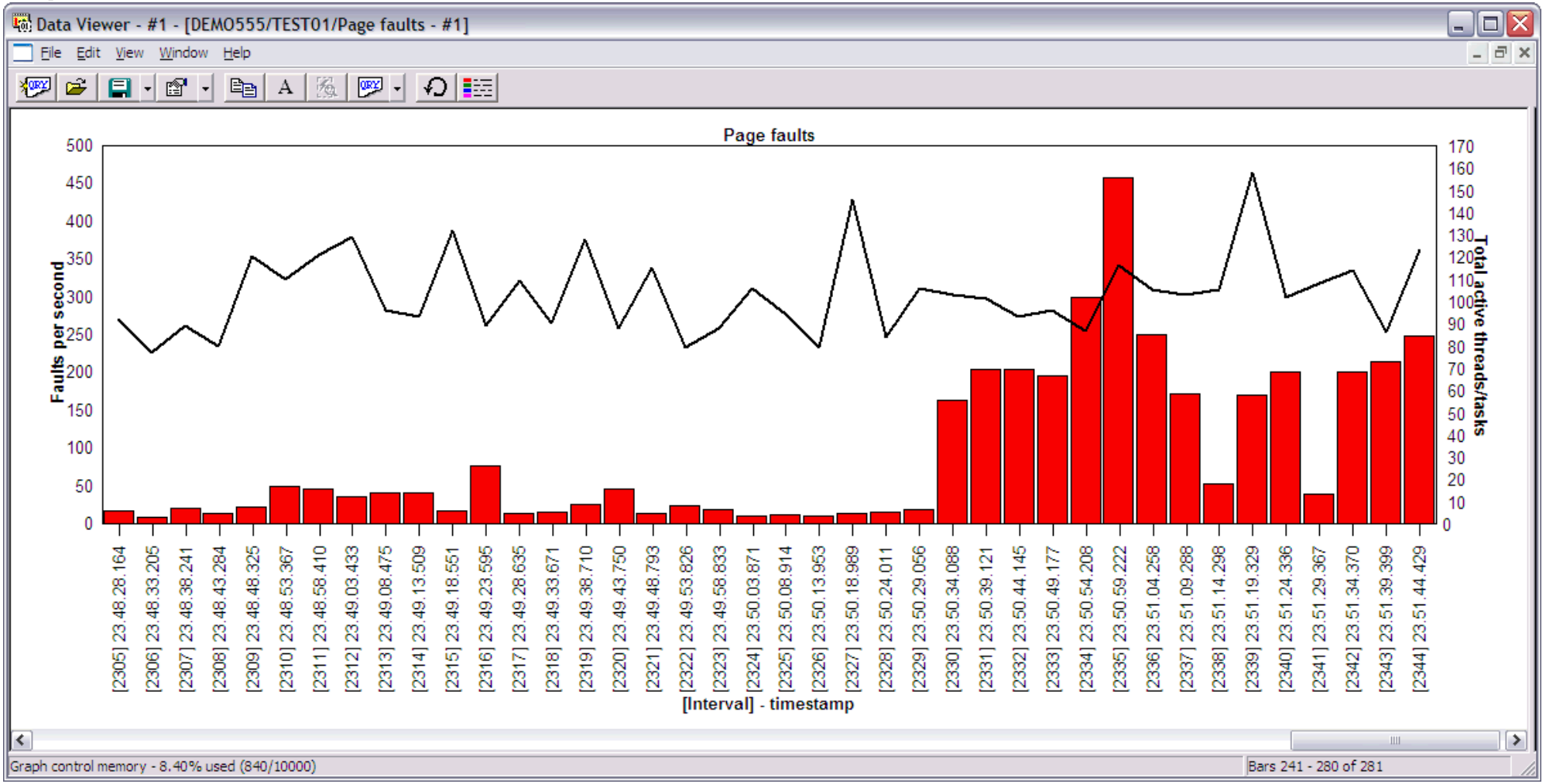
Menu	Description
Detail reports	Displays one of the table view reports for the selected interval within the job watch.
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar in a properties window. This window will also appear by left-clicking on any bar in the graph.

2.5.4.5 Page faults

Description: This graph shows a **summary** of the rates of page faults for each interval in the job watch.

Graph Type: summarized by interval (vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: The bars shows the rates of page faults occuring per second for each interval.
Second Y-Axis: The secondary Y-Axis displays the total active threads/tasks during each interval.

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Detail reports	Displays one of the table view reports for the selected interval within the job watch.
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar in a properties window. This window will also appear by left-clicking on any bar in the graph.

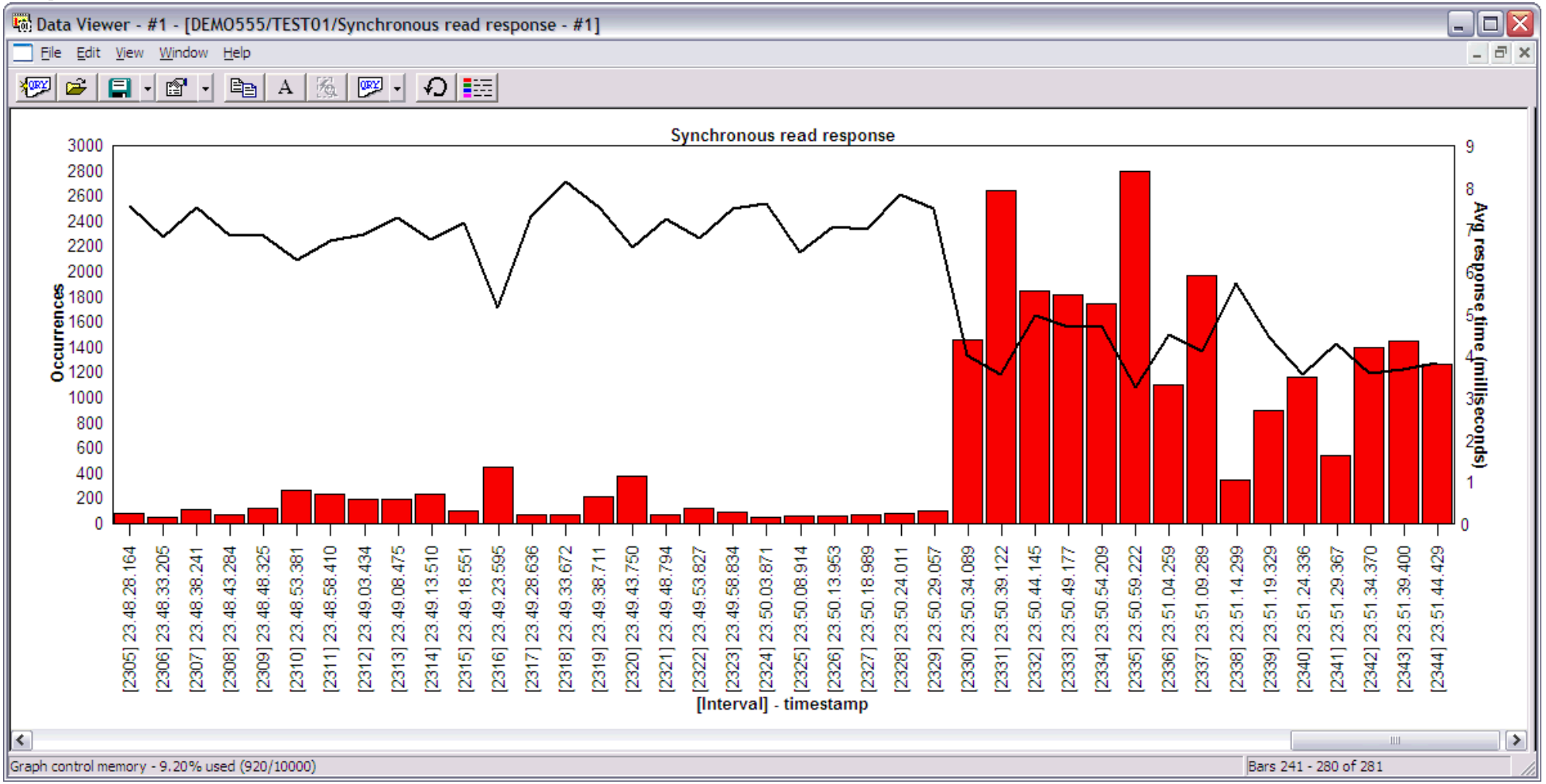
Table of Contents | Previous | Next

2.5.4.6 Synchronous read response

Description: This graph shows a **summary** of the total synchronous reads occuring every interval in the job watch and the average response time.

Graph Type: summarized by interval (vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: The bars shows total number of synchronous reads occurring each interval.
Second Y-Axis: The secondary axis shows the average synchronous reads response time (in milliseconds) for each interval in the job watch.

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Detail reports	Displays one of the table view reports for the selected interval within the job watch.
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar in a properties window. This window will also appear by left-clicking on any bar in the graph.

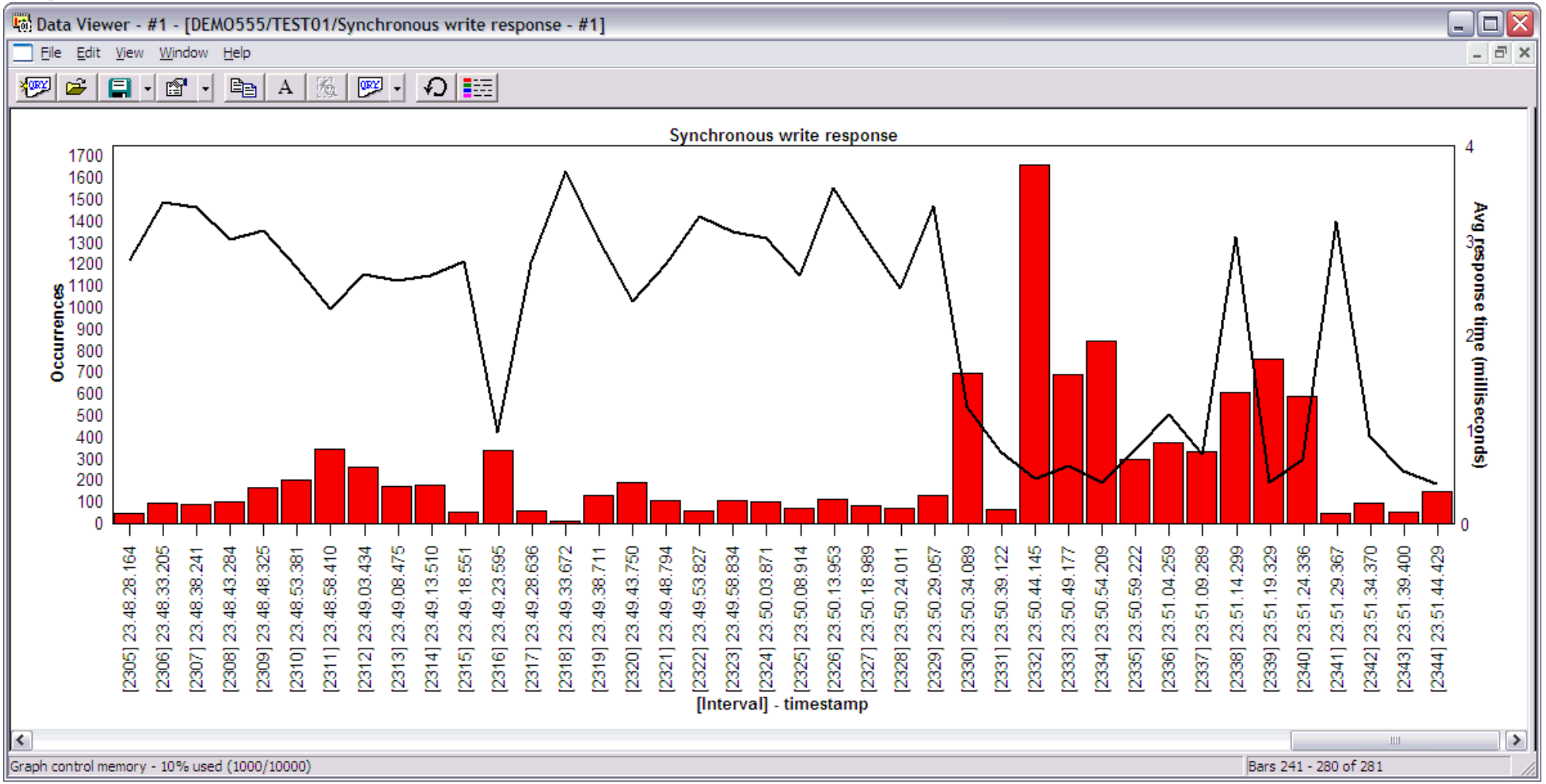
Table of Contents | Previous | Next

2.5.4.7 Synchronous write response

Description: This graph shows a **summary** of the total synchronous writes occuring every interval in the job watch and the average response time.

Graph Type: summarized by interval (vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: The bars shows total number of synchronous writes occurring each interval.
Second Y-Axis: The secondary axis shows the average synchronous write response time (in milliseconds) for each interval in the job watch.

Example:



Right-clicking on a bar in this graph provides the following menu options:

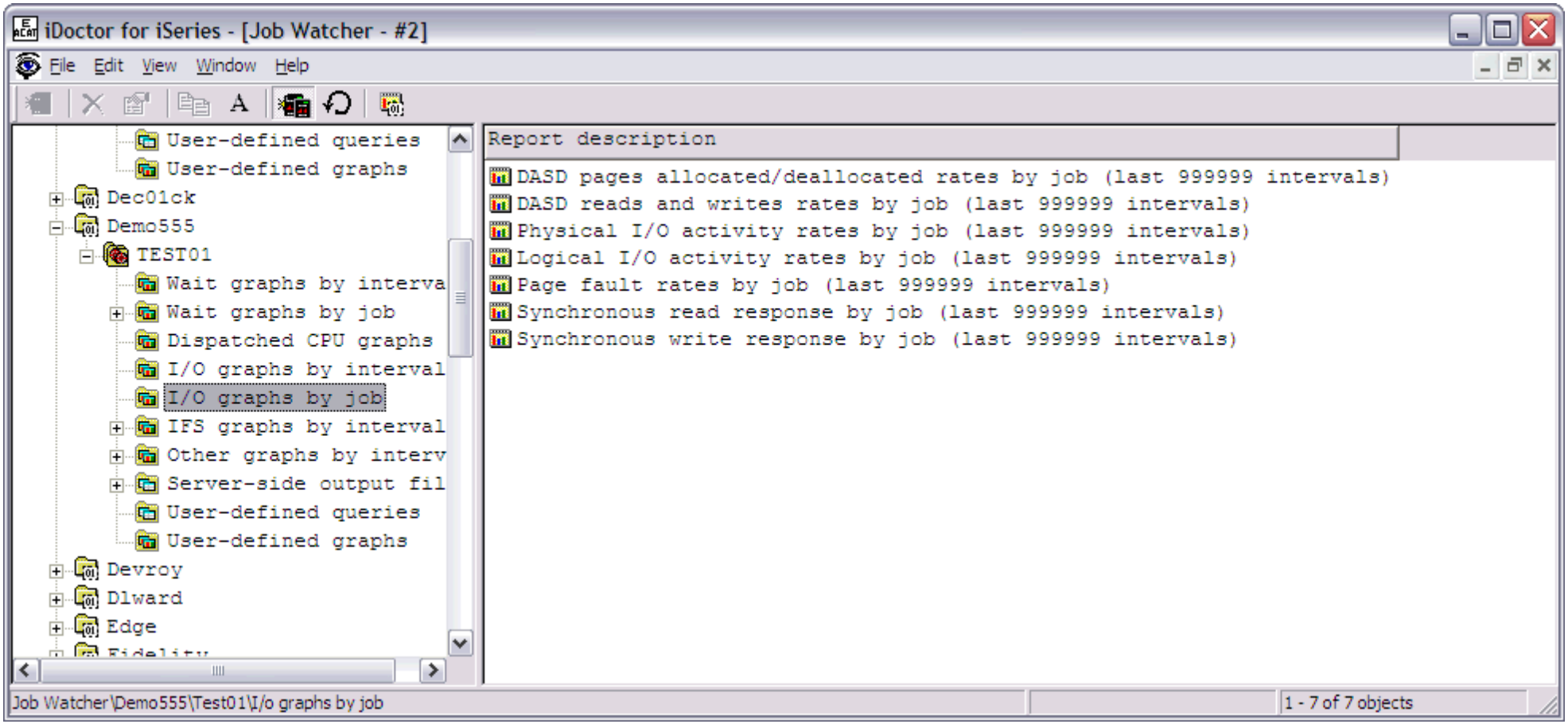
Menu	Description
Detail reports	Displays one of the table view reports for the selected interval within the job watch.
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar in a properties window. This window will also appear by left-clicking on any bar in the graph.

Table of Contents | Previous | Next

2.5.5 I/O graphs by job

This folder contains a list of graphs which contain summary data over the job watch relating to DASD/IO usage. These graphs display a bar per job with the jobs ranked by a statistic related to physical IO or disk activity.

An example of the contents of the I/O graphs by job folder is:



The following table illustrates the menu options available by right-clicking on a graph in the list.

Menu	Description

Open graph(s)	Opens the selected graph(s) into a Data Viewer. If a Data Viewer has already been opened submenus will appear underneath this menu in order to give the option of opening the graph(s) into an already open Data Viewer.

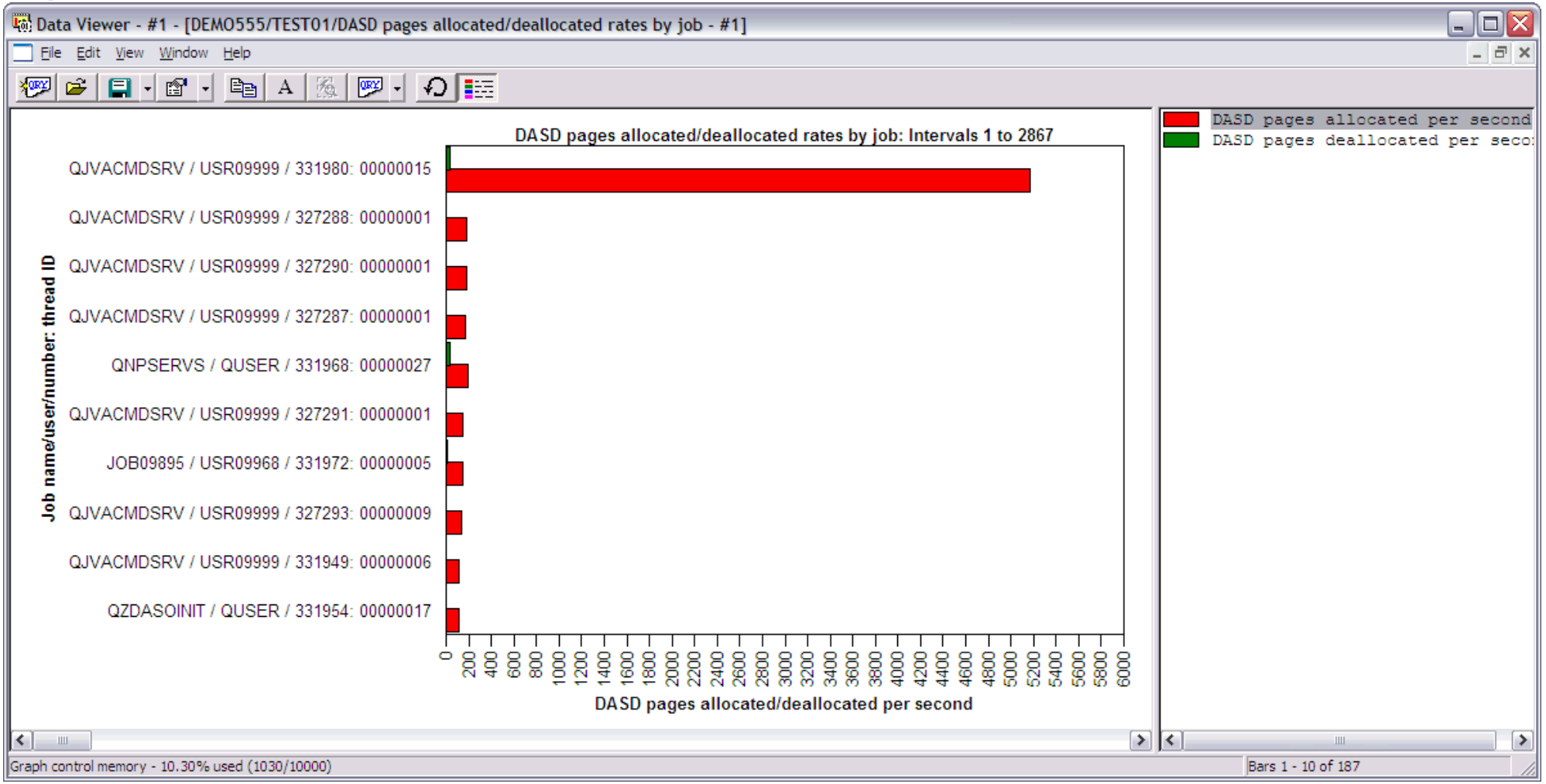
2.5.5.1 DASD pages allocated/deallocated rates by job

Description: This graph shows a **summary** of the rates of DASD pages allocated/deallocated for each job in the collection over the selected interval range. The bars in the graph show rates of pages allocated and deallocated.

This graph can be useful to see which jobs made the highest number of DASD space allocations or deallocations.

Graph Type: summarized by job (horizontal bar)
X-axis: Job name and thread ID or task name and taskcount.
Y-Axis: The bars show the rate of 4K DASD pages allocated and deallocated. Red bars show allocations and green bars show deallocations. All rates are 4k DASD pages per second.

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Detail reports	Displays one of the table view reports for the selected interval within the job watch.
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar in a properties window. This window will also appear by left-clicking on any bar in the graph.

Table of Contents | Previous | Next

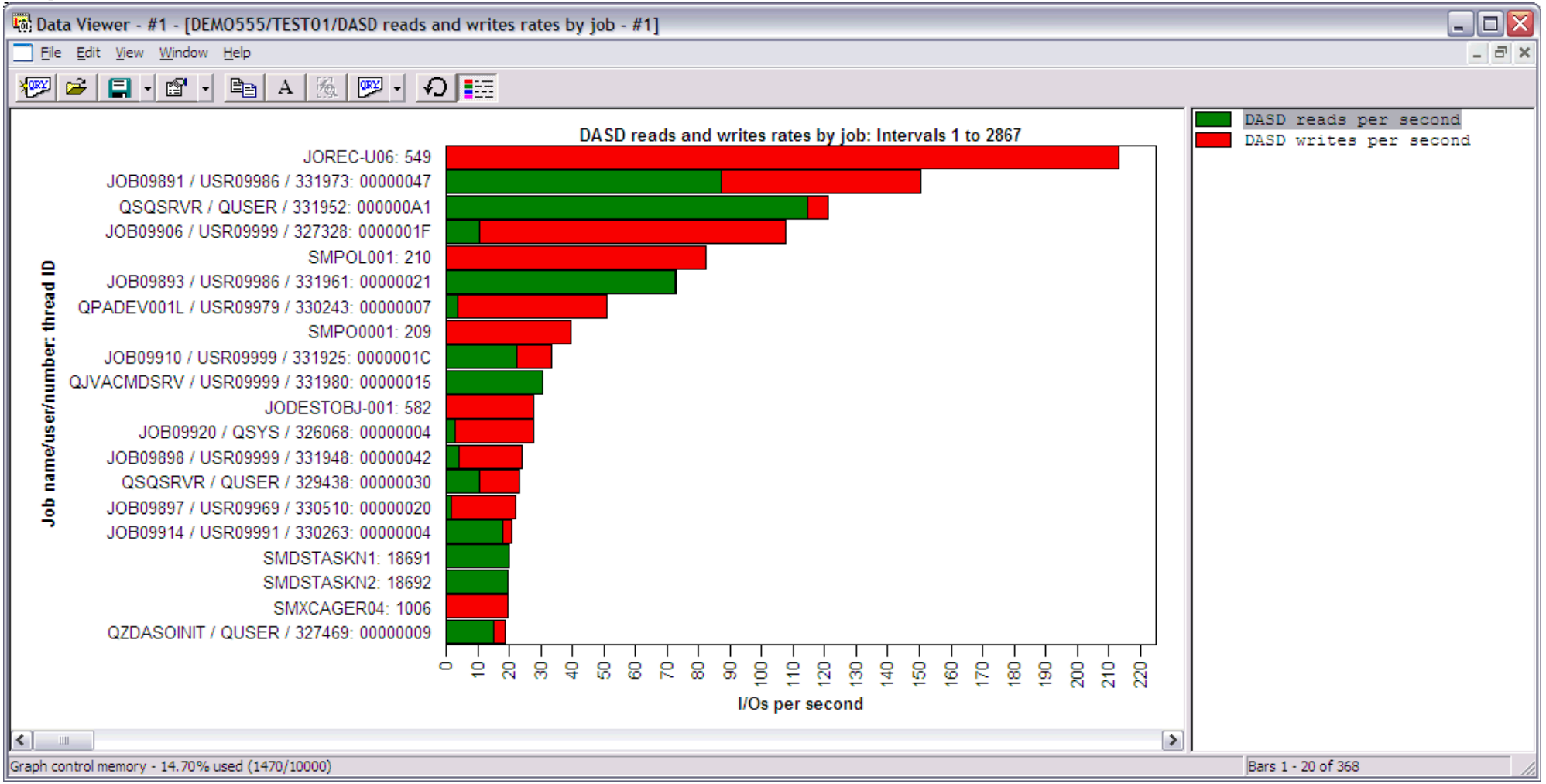
2.5.5.2 DASD reads and write rates by job

Description: This graph shows a **summary** of the rates of DASD reads and writes for each job in the collection over the selected interval range. The red bars show reads per second and the green bars show writes per second.

This graph can be useful to see which jobs had the highest rates of disk reads and writes.

Graph Type: summarized by job (horizontal bar)
X-axis: Job name and thread ID or task name and taskcount.
Y-Axis: The bars show rates of disk reads and writes per second.

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Detail reports	Displays one of the table view reports for the selected interval within the job watch.
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar in a properties window. This window will also appear by left-clicking on any bar in the graph.

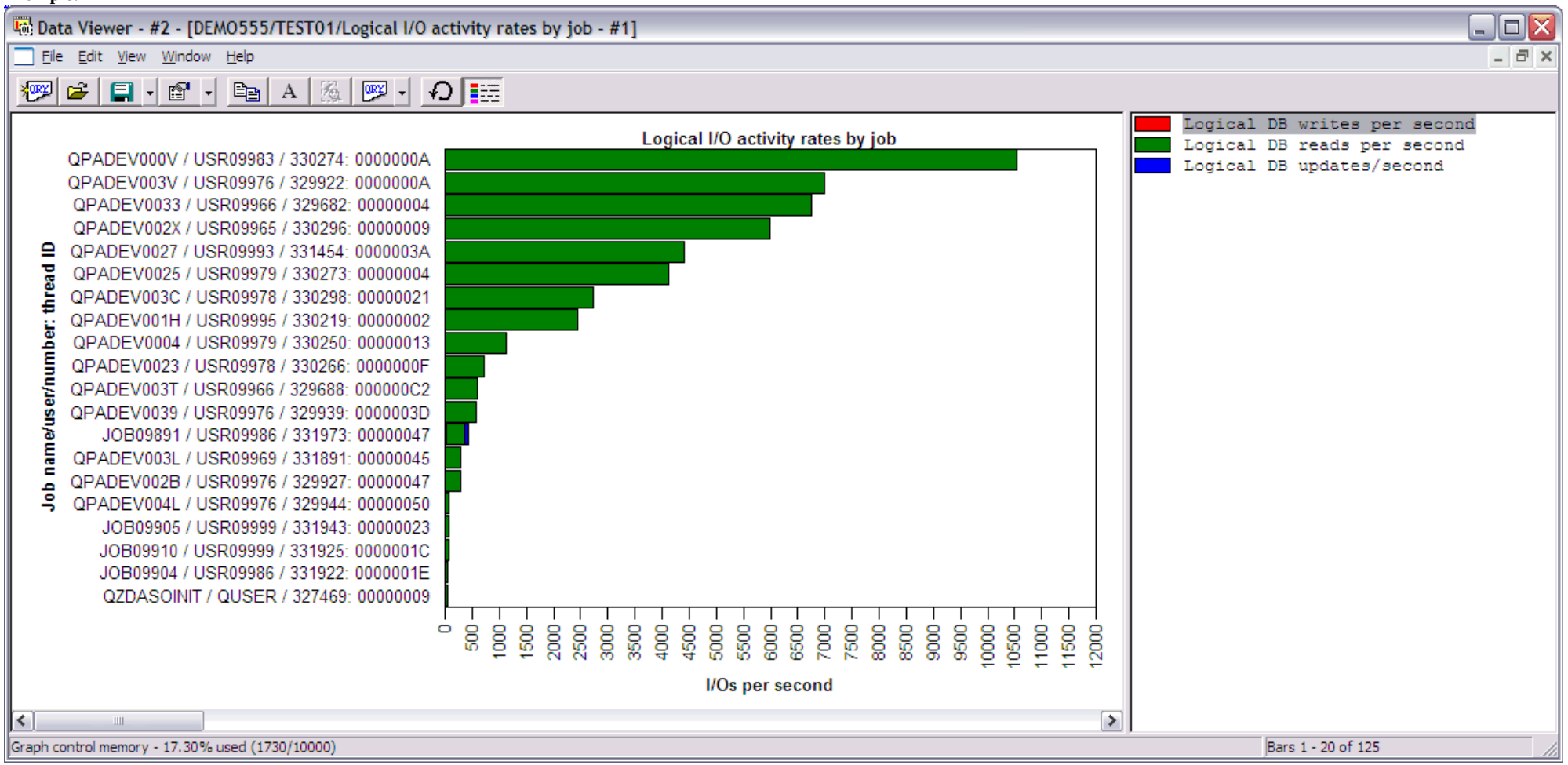
Table of Contents | Previous | Next

2.5.5.3 Logical I/O activity rates by job

Description: This graph shows a **summary** of logical database I/O activity for each job in the collection over the selected interval range. The bars in the graph show rates of logical DB reads, writes and updates/deletes per second.

Graph Type: summarized by job (horizontal bar)
X-axis: Job name and thread ID or task name and taskcount.
Y-Axis: The bars show jobs with the highest rates of logical database IO activity.

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description

Detail reports	Displays one of the table view reports for the selected interval within the job watch.
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar in a properties window. This window will also appear by left-clicking on any bar in the graph.

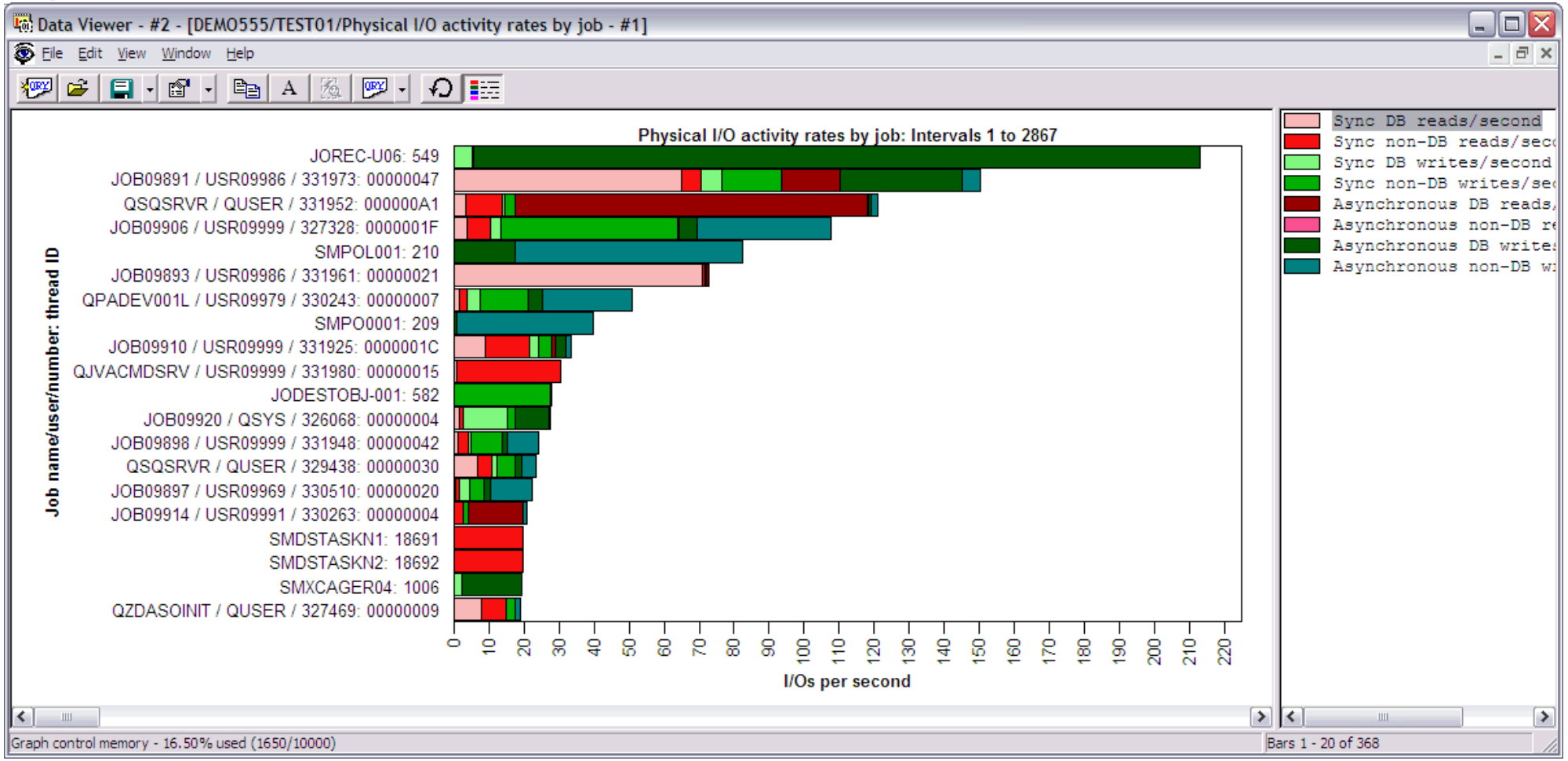
Table of Contents | Previous | Next

2.5.5.4 Physical I/O activity rates by job

Description: This graph shows a **summary** of the rates of synchronous and asycnhronous, database and non-database reads and writes per second for each job in the collection over the selected interval range.

Graph Type: summarized by job (horizontal bar)
X-axis: Job name and thread ID or task name and taskcount.
Y-Axis: The bars shows jobs with the highest rates of physical I/O activity.

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description

Detail reports	Displays one of the table view reports for the selected interval within the job watch.
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar in a properties window. This window will also appear by left-clicking on any bar in the graph.

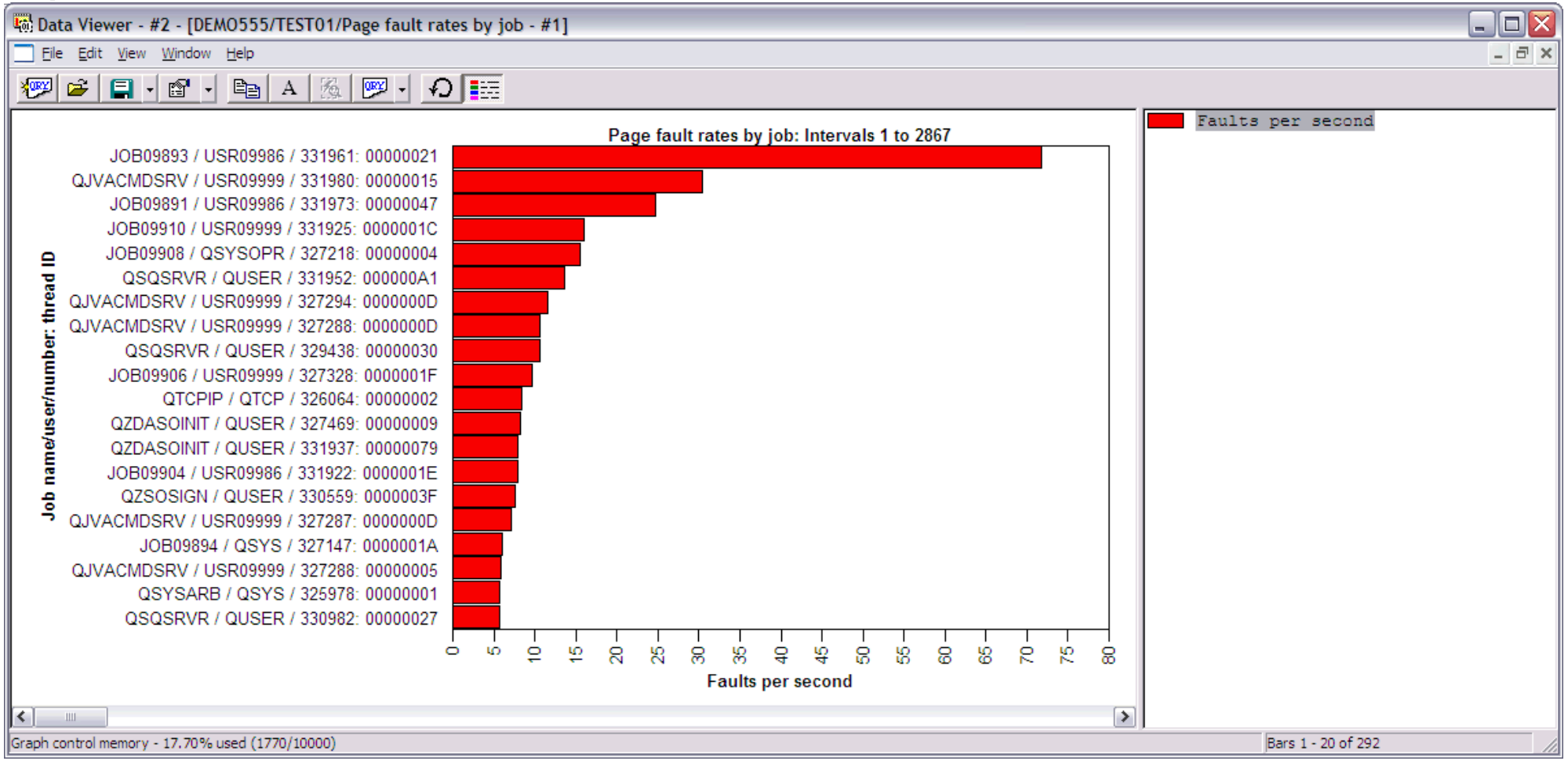
Table of Contents | Previous | Next

2.5.5.5 Page fault rates by job

Description: This graph shows a **summary** of the rates of page faults for each job in the collection over the selected interval range.

Graph Type: summarized by job (horizontal bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: The bars shows the jobs with the highest rates of page faulting.

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description

Detail reports	Displays one of the table view reports for the selected interval within the job watch.
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar in a properties window. This window will also appear by left-clicking on any bar in the graph.

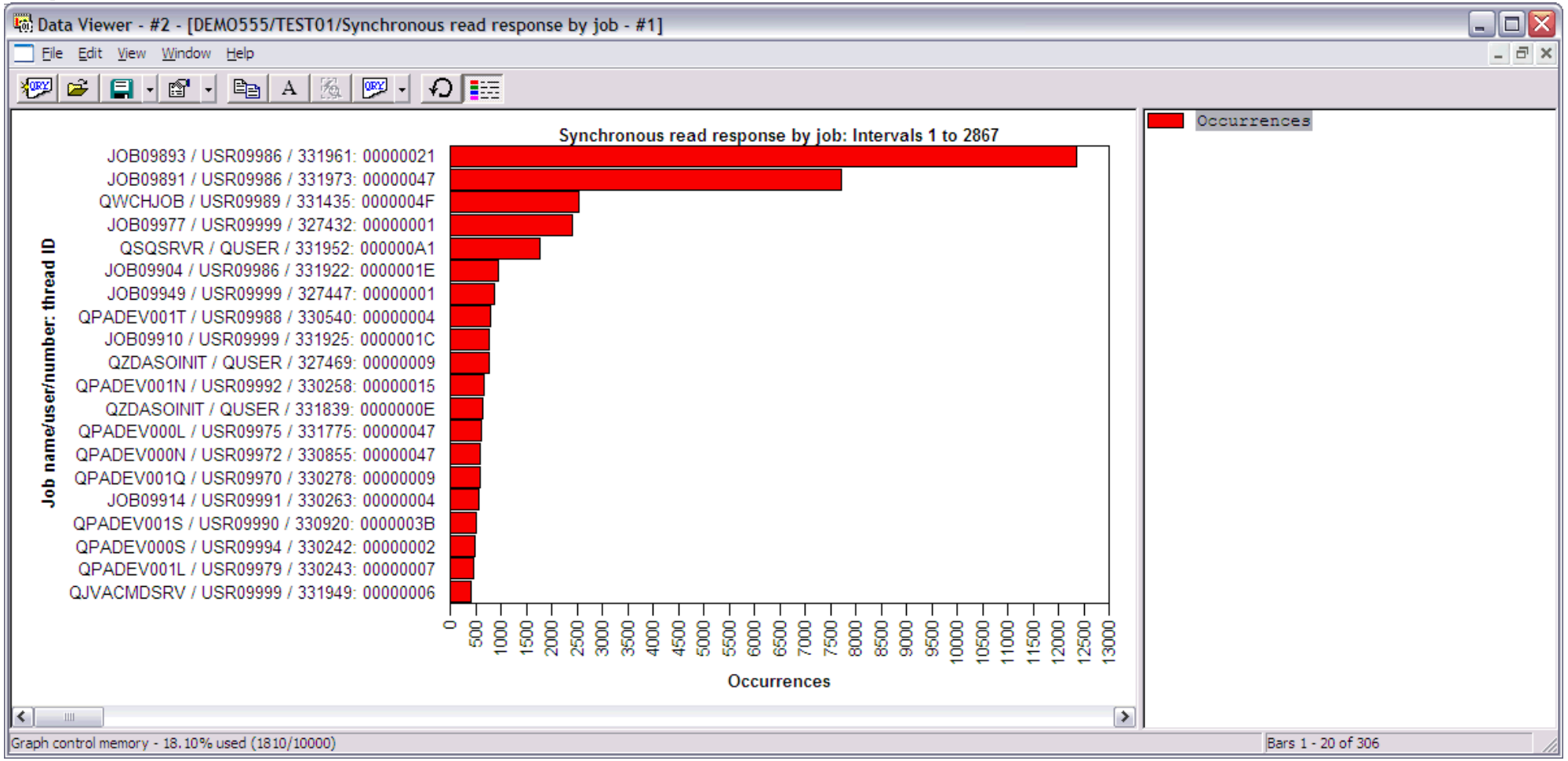
Table of Contents | Previous | Next

2.5.5.6 Synchronous read response by job

Description: This graph shows a **summary** of the jobs with the highest number of synchronous reads.

Graph Type: summarized by job (horizontal bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: The bars shows total number of synchronous reads per job.

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description

Detail reports	Displays one of the table view reports for the selected interval within the job watch.
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar in a properties window. This window will also appear by left-clicking on any bar in the graph.

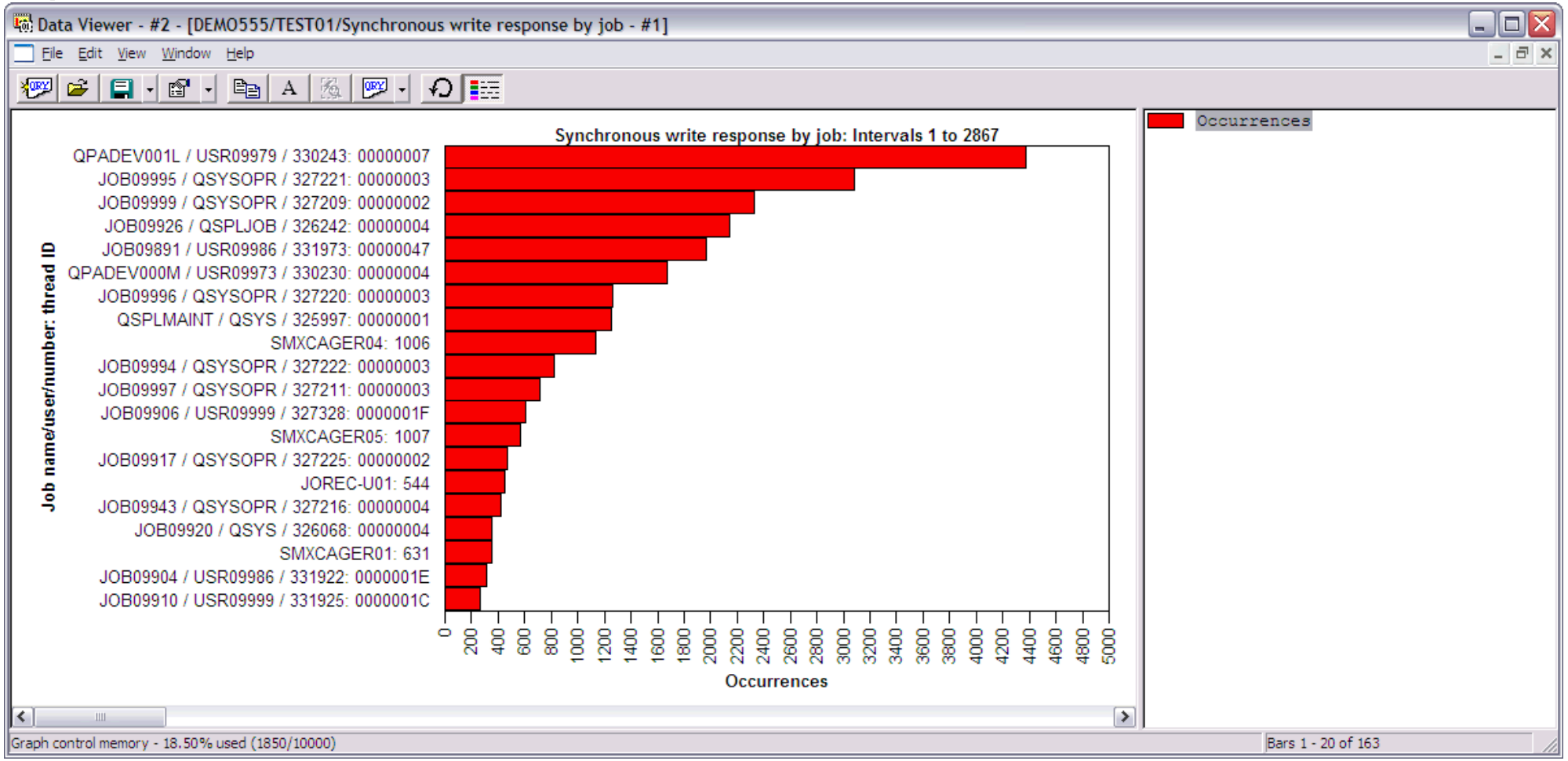
Table of Contents | Previous | Next

2.5.5.7 Synchronous write response by job

Description: This graph shows a **summary** of jobs with the highest number of synchronous writes.

Graph Type: summarized by job (horizontal bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: The bars shows total number of synchronous writes occurring for each job.

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description

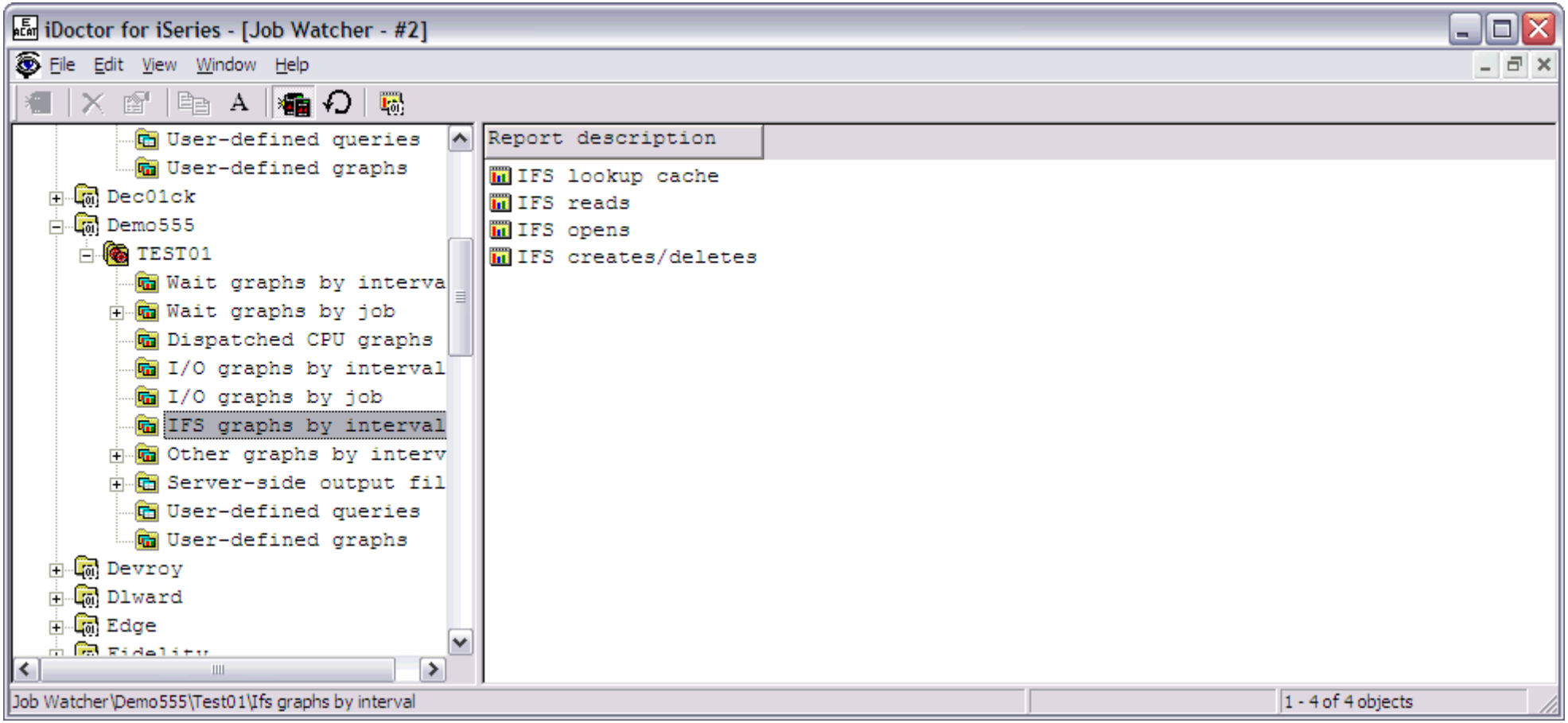
Detail reports	Displays one of the table view reports for the selected interval within the job watch.
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar in a properties window. This window will also appear by left-clicking on any bar in the graph.

Table of Contents | Previous | Next

2.5.6 IFS graphs by interval

This folder contains a list of graphs which contain summary data over the job watch relating to IFS statistics.

An example of the contents of the IFS graphs by interval folder is:



The following table illustrates the menu options available by right-clicking on a graph in the list.

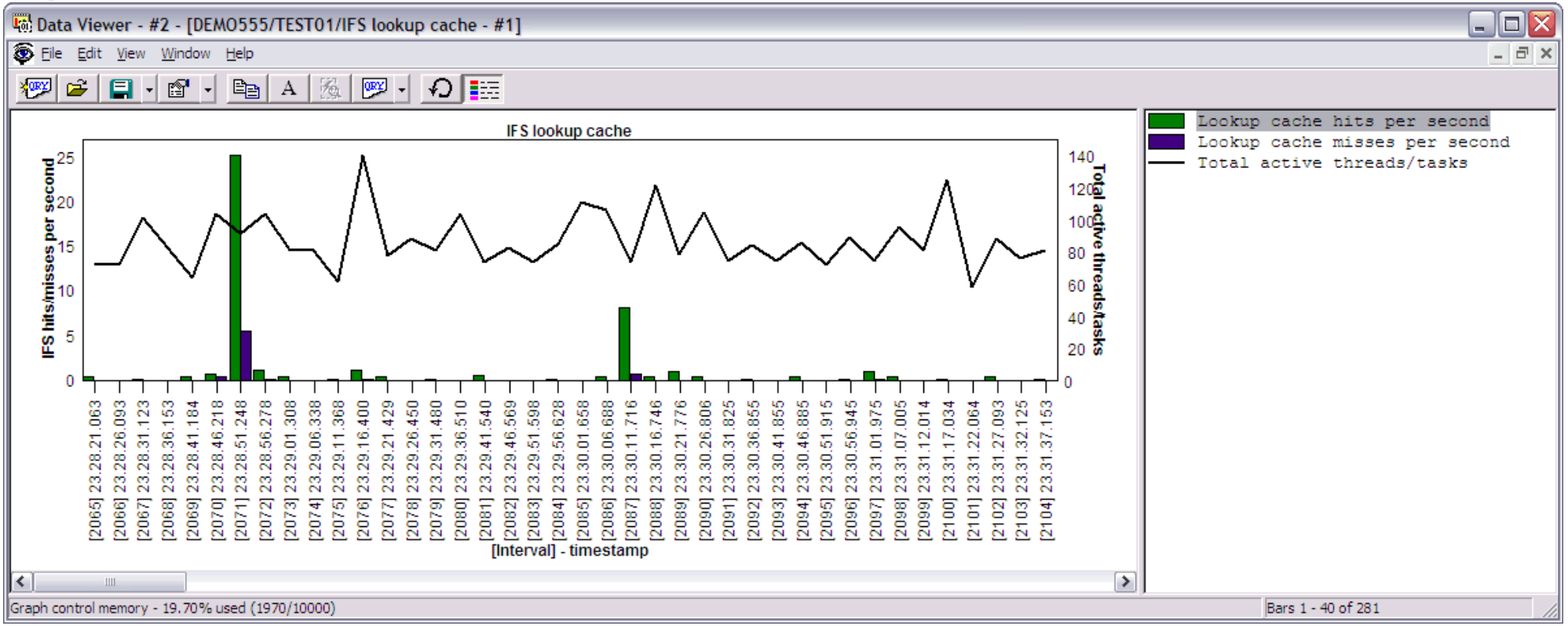
Menu	Description
Open graph(s)	Opens the selected graph(s) into a Data Viewer. If a Data Viewer has already been opened submenus will appear underneath this menu in order to give the option of opening the graph(s) into an already open Data Viewer.

2.5.6.1 IFS lookup cache

Description: This graph shows a **summary** of the rates of IFS lookup cache hits and misses for each interval in the job watch..

Graph Type: summarized by interval (vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: The green bars show the rates of lookup cache hits and the blue bars show the rates of lookup cache misses.
Second Y-Axis: The secondary Y-Axis displays the total number of active threads/tasks on the system for each interval.

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Detail reports	Displays one of the table view reports for the selected interval within the job watch.
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.

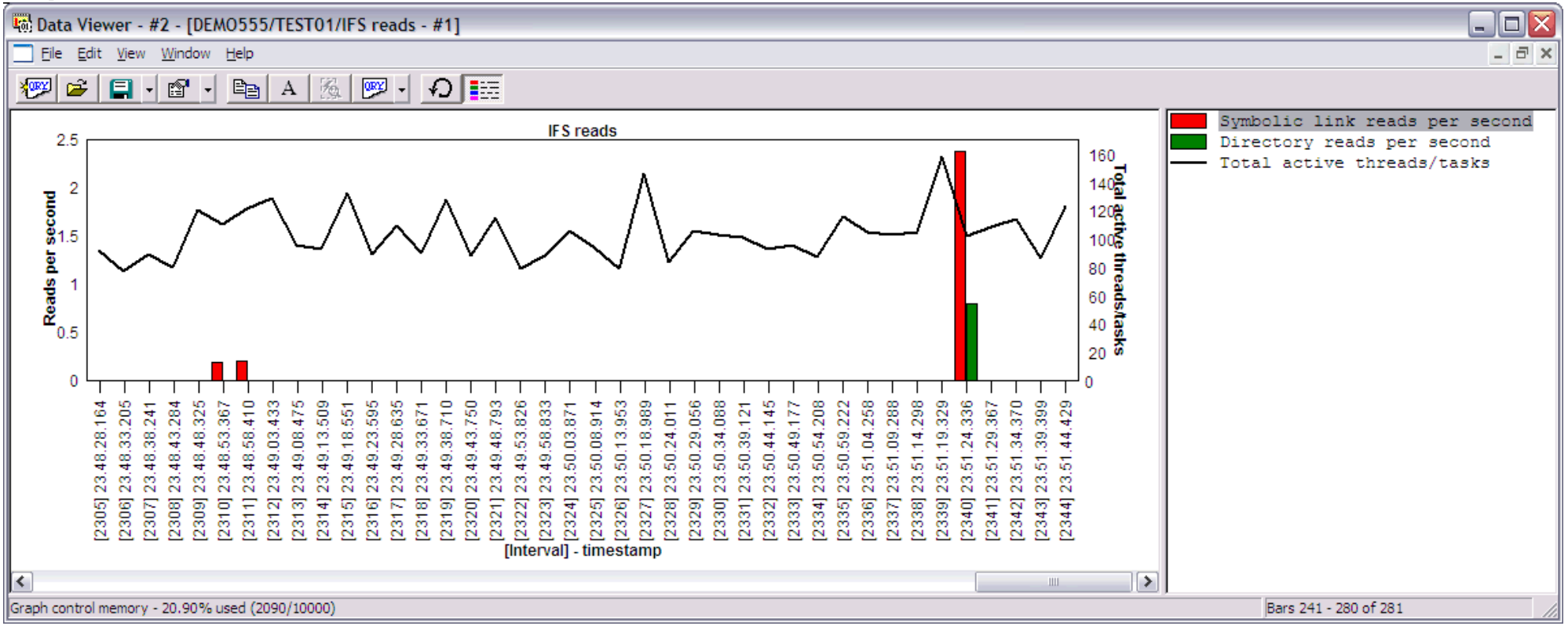
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar in a properties window. This window will also appear by left-clicking on any bar in the graph.

2.5.6.2 IFS reads

Description: This graph shows a **summary** of the rates of IFS reads for each interval in the job watch..

Graph Type: summarized by interval (vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: The red bars show IFS symbolic link reads per second. The green bars show IFS directory reads per second.
Second Y-Axis: The secondary Y-Axis displays the total number of active threads/tasks on the system for each interval.

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Detail reports	Displays one of the table view reports for the selected interval within the job watch.
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.

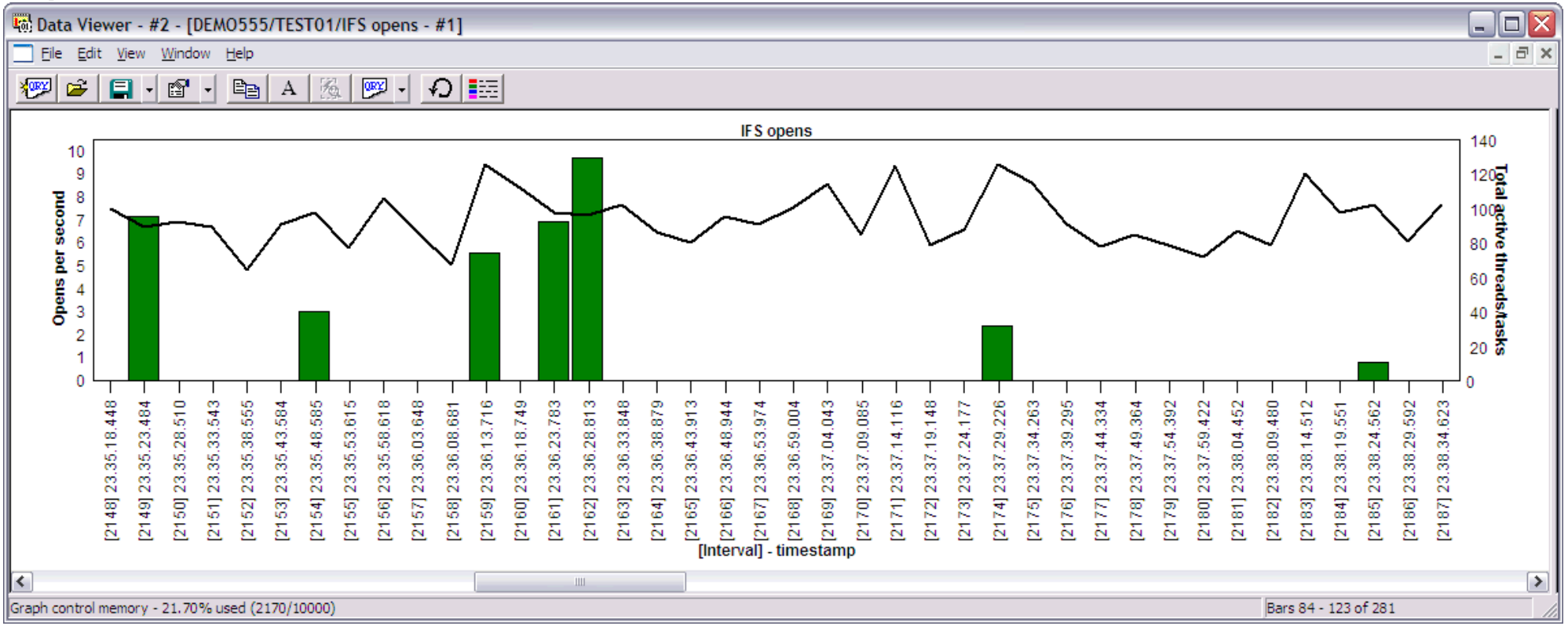
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar in a properties window. This window will also appear by left-clicking on any bar in the graph.

2.5.6.3 IFS opens

Description: This graph shows a **summary** of the rates of IFS file opens during each interval in the job watch..

Graph Type: summarized by interval (vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: The green bars show IFS file opens per second.
Second Y-Axis: The secondary Y-Axis displays the total number of active threads/tasks on the system for each interval.

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Detail reports	Displays one of the table view reports for the selected interval within the job watch.
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.

Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar in a properties window. This window will also appear by left-clicking on any bar in the graph.

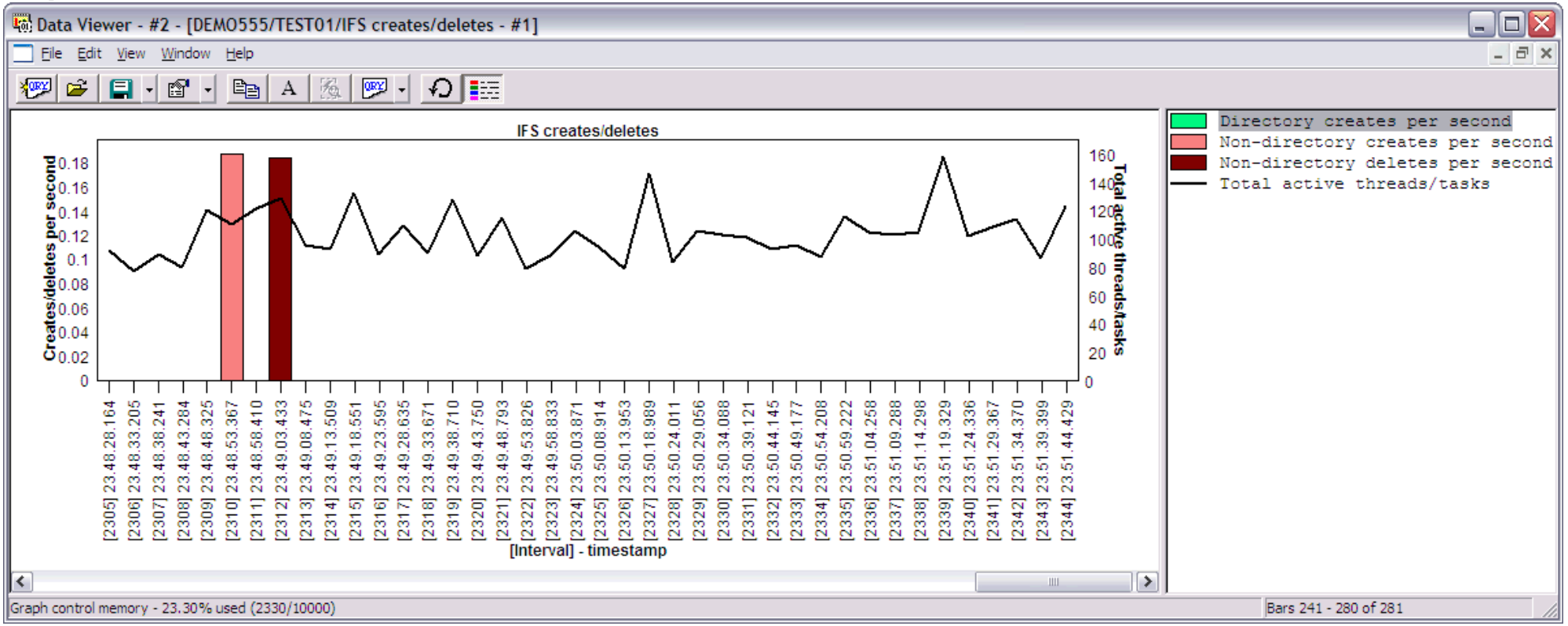
Table of Contents | Previous | Next

2.5.6.4 IFS creates/deletes

Description: This graph shows a **summary** of the rates of IFS directory and non-directory creates and deletes for each interval in the job watch..

Graph Type: summarized by interval (vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: Contains the rates of IFS directory and non-directory creates and deletes per second. Green colors represent directory operations and red colors represent non-directory operations.
Second Y-Axis: The secondary Y-Axis displays the total number of active threads/tasks on the system for each interval.

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Detail reports	Displays one of the table view reports for the selected interval within the job watch.
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.

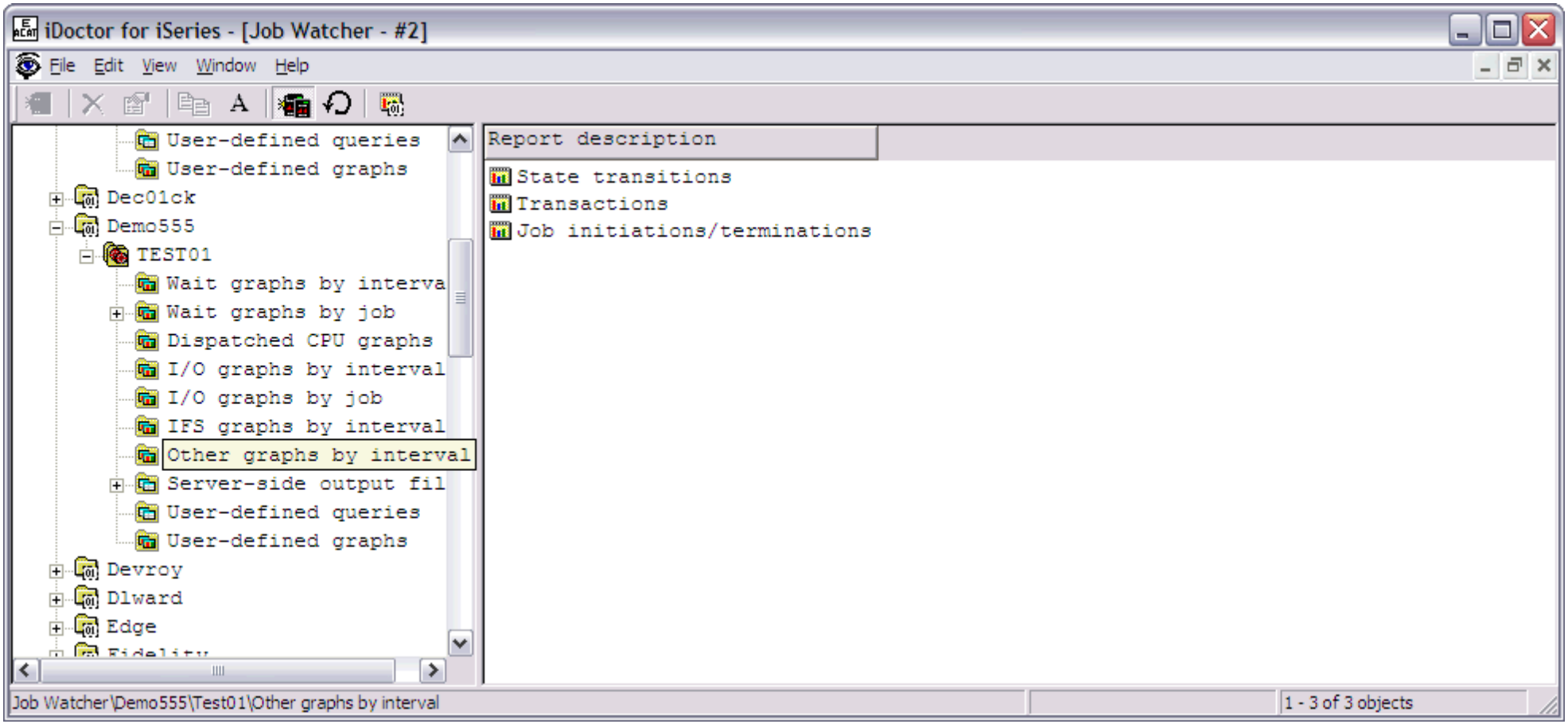
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar in a properties window. This window will also appear by left-clicking on any bar in the graph.

Table of Contents | Previous | Next

2.5.7 Other graphs by interval

This folder contains a list of graphs which contain performance data not covered in the other graph categories. These graphs display a bar (or set of bars) per interval. The types of data available in these graphs include state transitions and transactions.

An example of the contents of the other graphs by interval folder is:



The following table illustrates the menu options available by right-clicking on a graph in the list.

Menu	Description

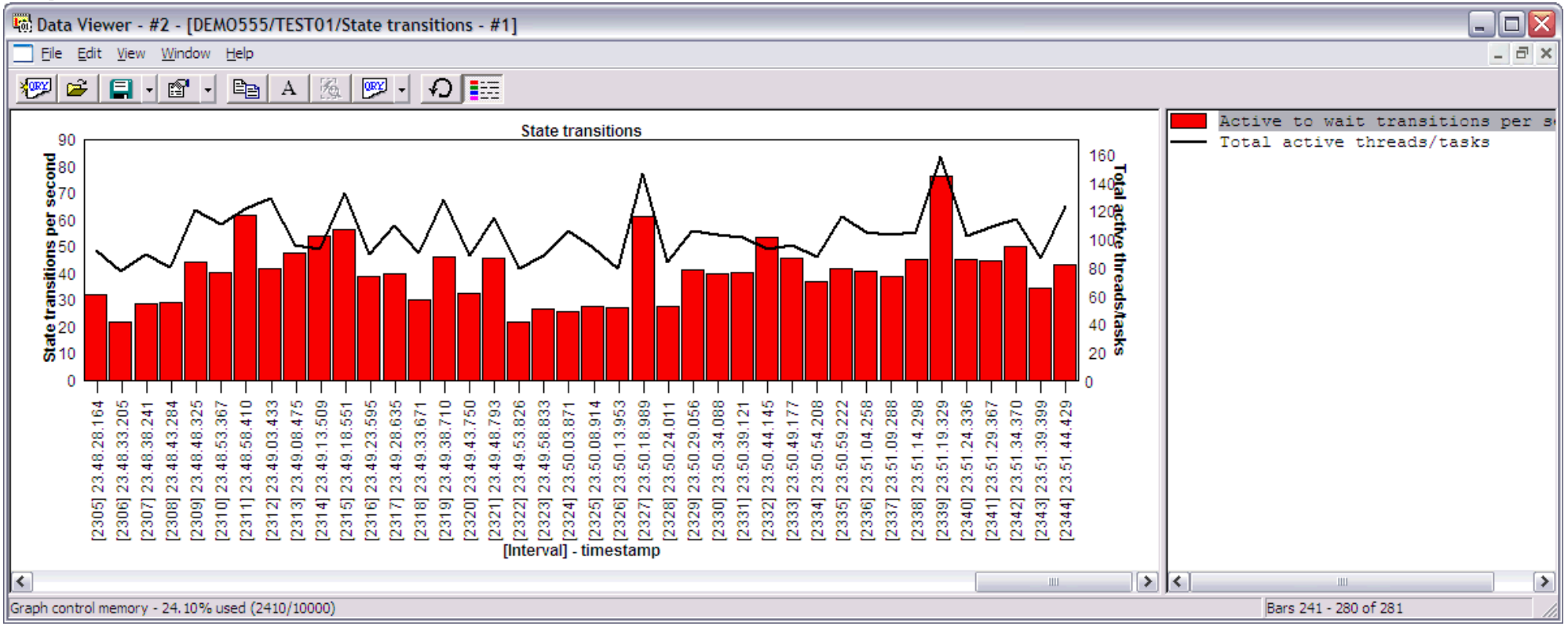
Open graph(s)	Opens the selected graph(s) into a Data Viewer. If a Data Viewer has already been opened submenus will appear underneath this menu in order to give the option of opening the graph(s) into an already open Data Viewer.

2.5.7.1 State transitions

Description: This graph shows a **summary** of the rates of job state transitions per second for each interval in the job watch..

Graph Type: summarized by interval (vertical bar)
X-Axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: Each bar shows the rates of the following types of job state transitions: active to wait, wait to ineligible, and active to ineligible.
Second Y-Axis: The secondary Y-Axis displays the total number of active threads/tasks on the system for each interval.

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Detail reports	Displays one of the table view reports for the selected interval within the job watch.
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.

Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar in a properties window. This window will also appear by left-clicking on any bar in the graph.

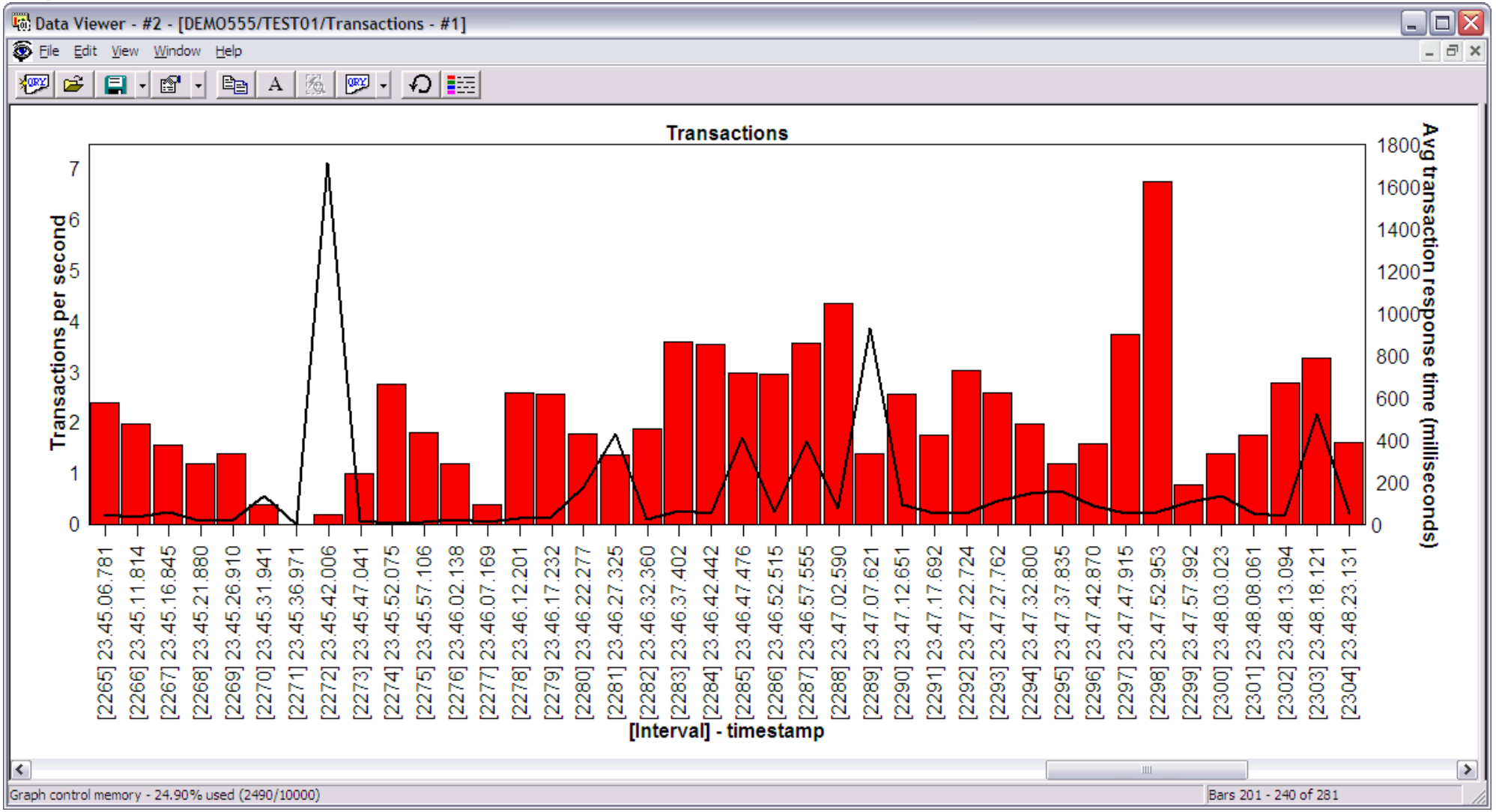
Table of Contents | Previous | Next

2.5.7.2 Transactions

Description: This graph shows a **summary** of the transaction rates per interval and the average transaction response time per interval in the job watch..

Graph Type: summarized by interval (vertical bar)
X-Axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: Each bar shows the transaction per second during each interval.
Second Y-Axis: The line shows the collection-wide average transaction response time (in milliseconds).

Example:



2.5.7.2 Transactions

Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Detail reports	Displays one of the table view reports for the selected interval within the job watch.
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar in a properties window. This window will also appear by left-clicking on any bar in the graph.

Table of Contents | Previous | Next

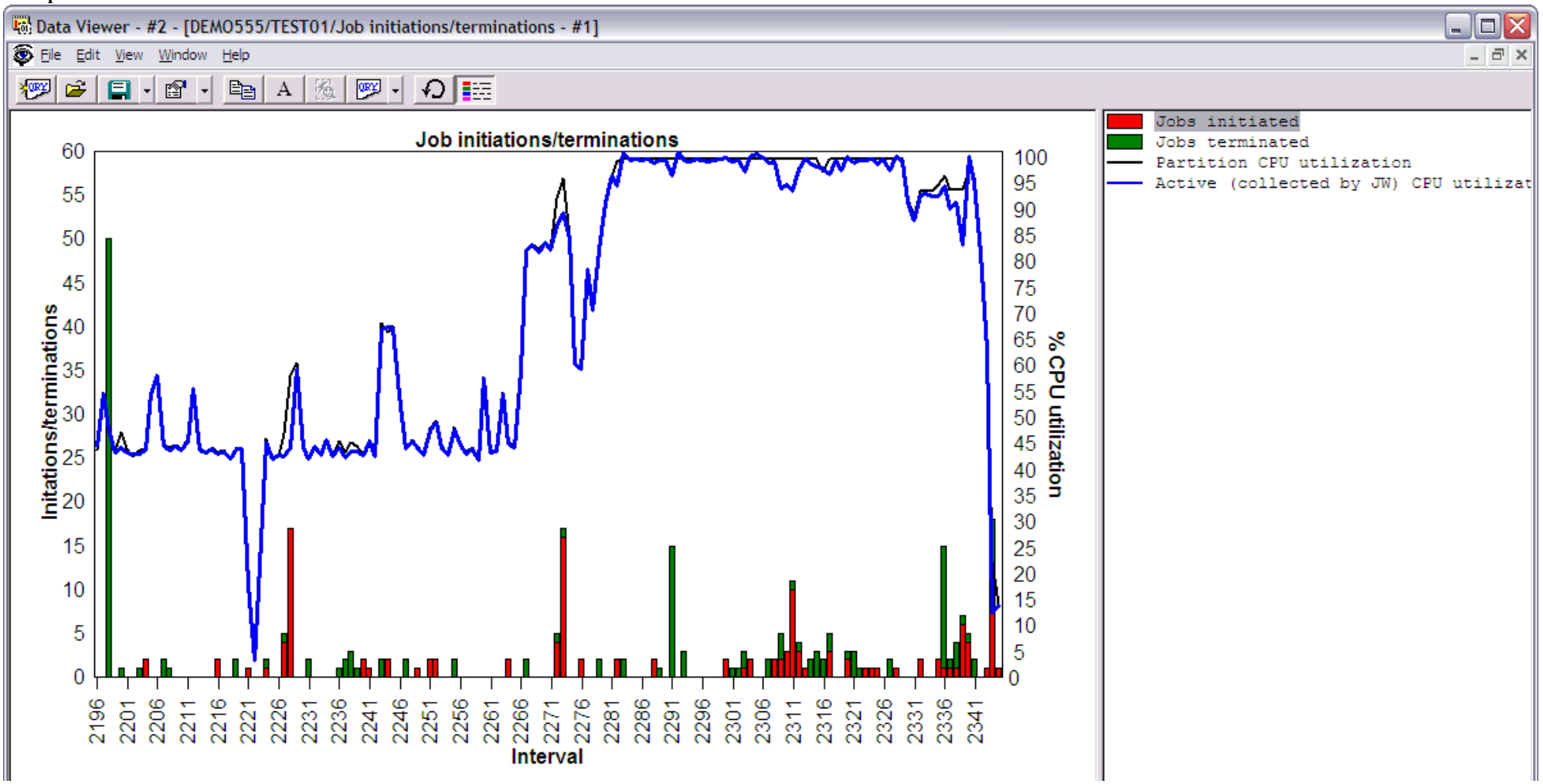
2.5.7.3 Job initiations/terminations

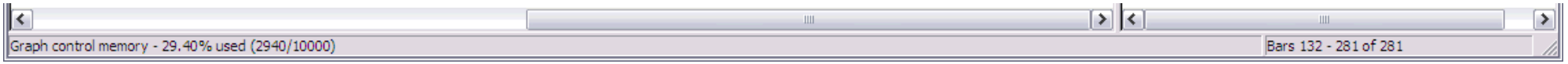
Description: This graph shows a **summary** of the total jobs created and destroyed each interval along with CPU utilization percentages on the secondary Y axis.

This graph is only shown if the collection has been summarized.

Graph Type: summarized by interval (vertical bar)
X-Axis: Interval number.
Y-Axis: Each bar shows the total number of jobs created or destroyed for an interval.
Second Y-Axis: The secondary Y-Axis displays partition CPU utilization and active CPU utilization. Active CPU utilization is generally not as accurate because it's based only on the jobs sampled by Job Watcher. Job Watcher will not collect information for jobs/tasks the started and ending within a single interval (between snapshots). These missed jobs account for the difference between partition CPU utilization (true CPU utilization) and active CPU utilization.

Example:





Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Detail reports	Displays one of the table view reports for the selected interval within the job watch.
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar in a properties window. This window will also appear by left-clicking on any bar in the graph.

Table of Contents | Previous | Next

2.5.8 Collection-wide Interval Properties

The collection-wide interval properties window displays information about a single interval in the job watch across all jobs/tasks/threads. This interface includes information on the objects waited on, holding threads/tasks and wait buckets.

Getting to this interface can be accomplished by clicking on any bar (representing an interval) in one of the summary graphs.

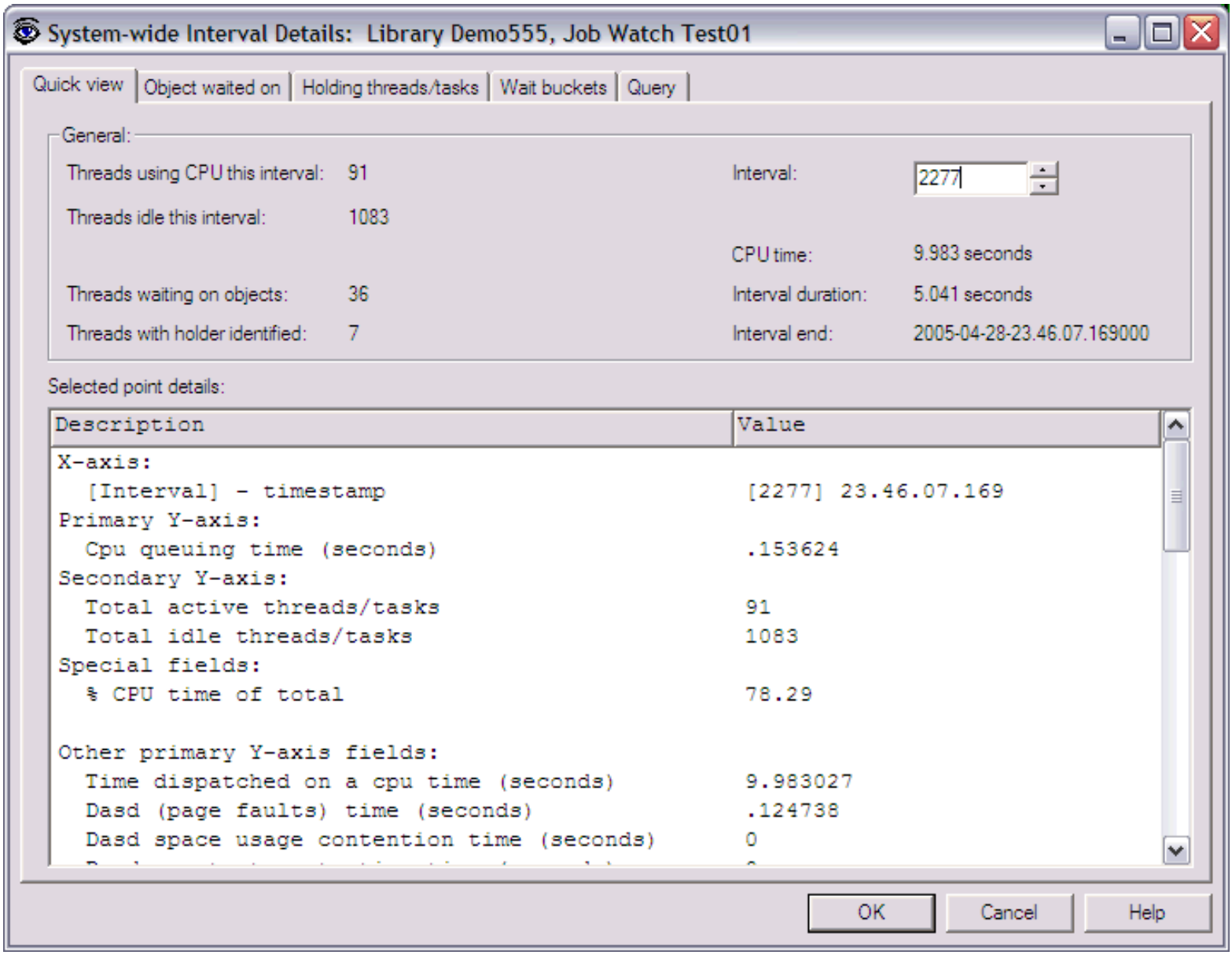
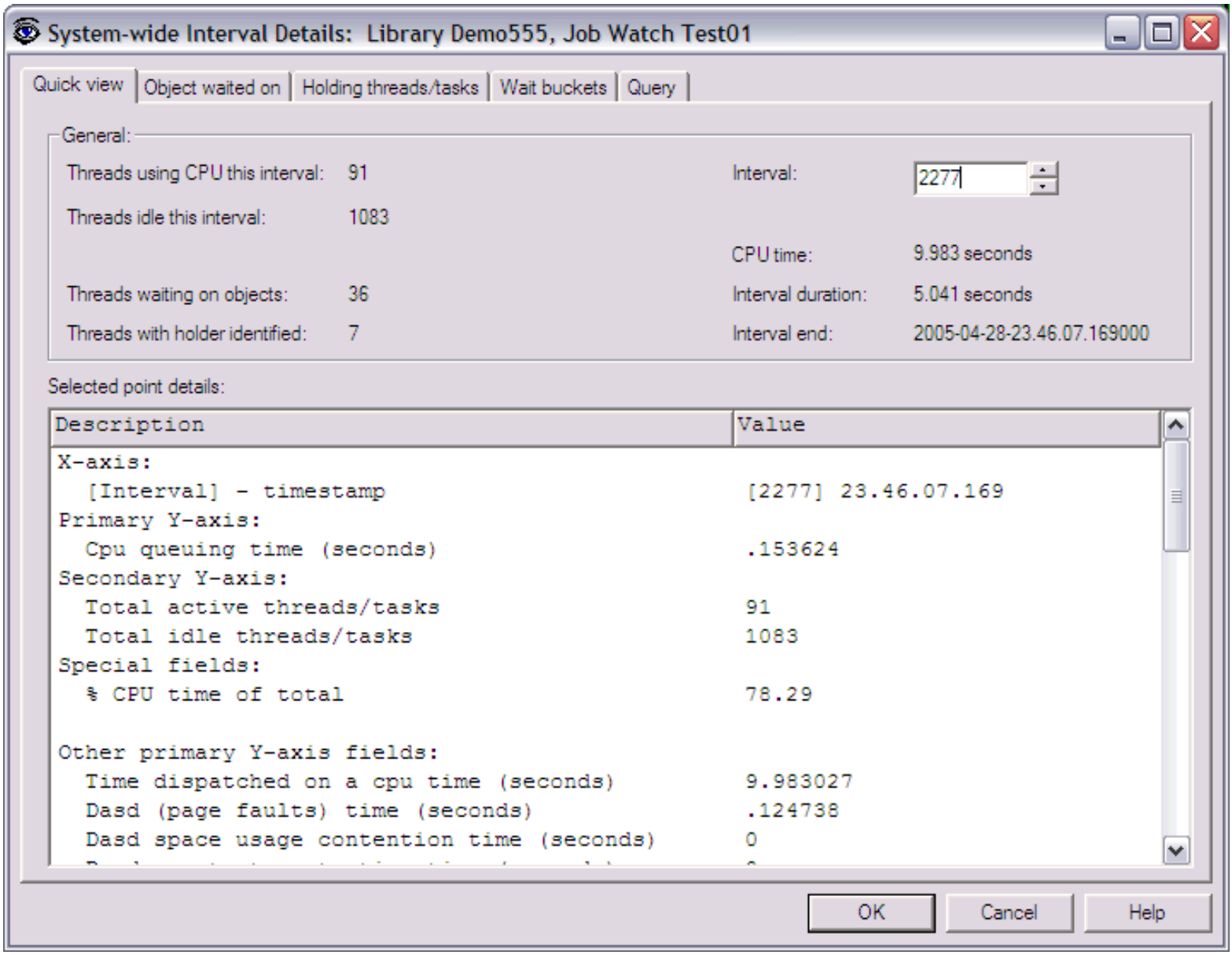


Table of Contents | Previous | Next

2.5.8.1 Quick View

This page shows a summary of information about the selected interval as well as a list of fields for the current interval from the graph this window was launched from. The list of fields shown in the list depends on the SQL statement behind the graph.



This window contains the following fields:

Field name	Description
Threads using CPU this interval	Displays the total threads and tasks that used some CPU during the current interval.

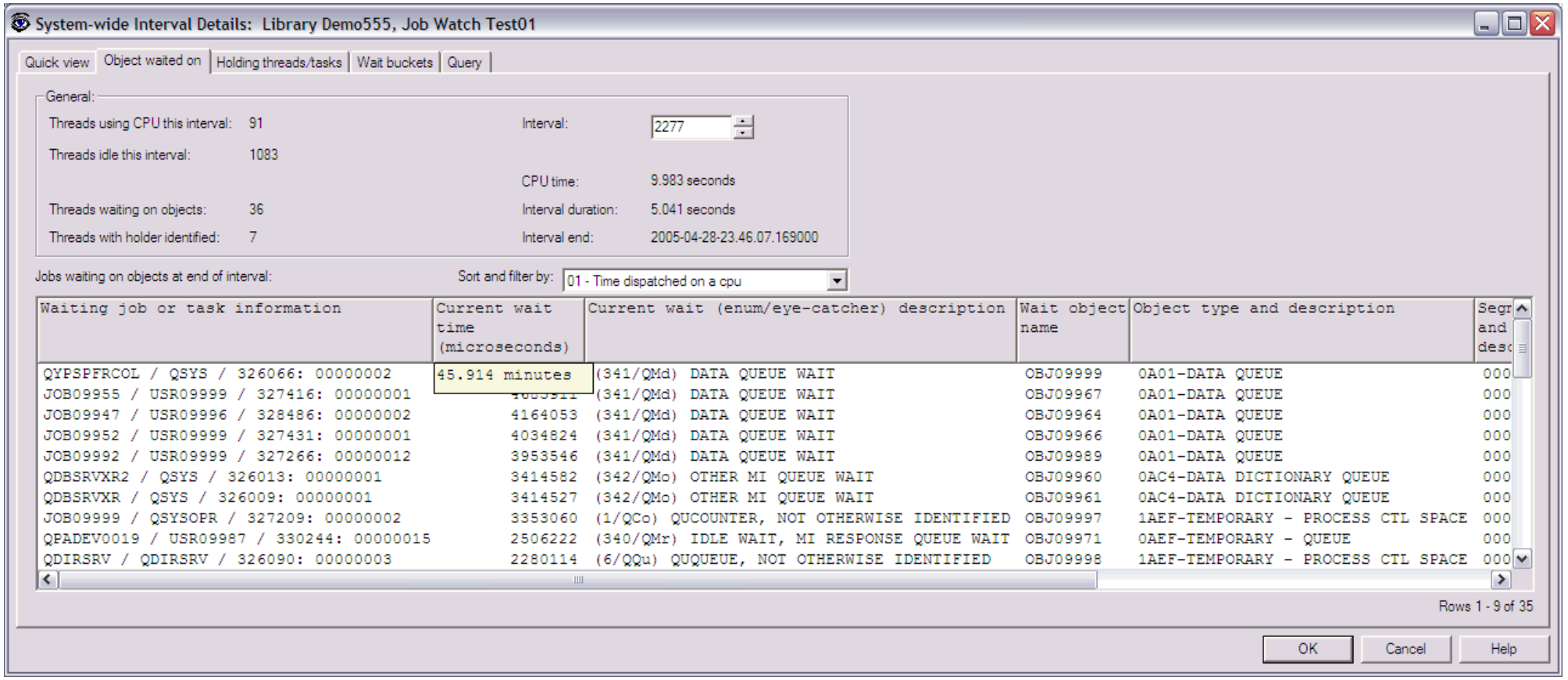
Threads idle this interval	Displays the total threads tasks on the system during the current interval.
Interval	This field displays the current interval. Using the arrow buttons allows the user to change the interval, or a number may be entered into the textbox and the window will be updated to display information about the interval number provided.
CPU time	Displays the total CPU time used across all jobs/thread/tasks for the current interval.
Threads waiting on objects	This number indicates the total threads and tasks that were waiting to use an object at the moment the interval's snapshot was taken.
Threads with holder identified	This number indicates the total threads and tasks that were being held by another job/thread or task at the moment the interval's snapshot was taken.
Interval duration	The total elapsed time from the start to end of the current interval.
Interval end	The date and time the current interval ended.

2.5.8.2 Object waited on

This page shows a summary of information about the selected interval as well as a list of jobs or tasks for the current interval that were waiting on an object.

This list contains information about the object such as its name, object type, segment type, as well as the type of wait and total wait time.

By right-clicking on the list the user can drill one or more jobs to see the run/wait time signatures for the jobs or any of the other detail reports starting at the current interval.



This window contains the following fields:

Field name	Description
Threads using CPU this interval	Displays the total threads and tasks that used some CPU during the current interval.
Threads idle this interval	Displays the total threads tasks on the system during the current interval.
Interval	This field displays the current interval. Using the arrow buttons allows the user to change the interval, or a number may be entered into the textbox and the window will be updated to display information about the interval number provided.
CPU time	Displays the total CPU time used across all jobs/thread/tasks for the current interval.

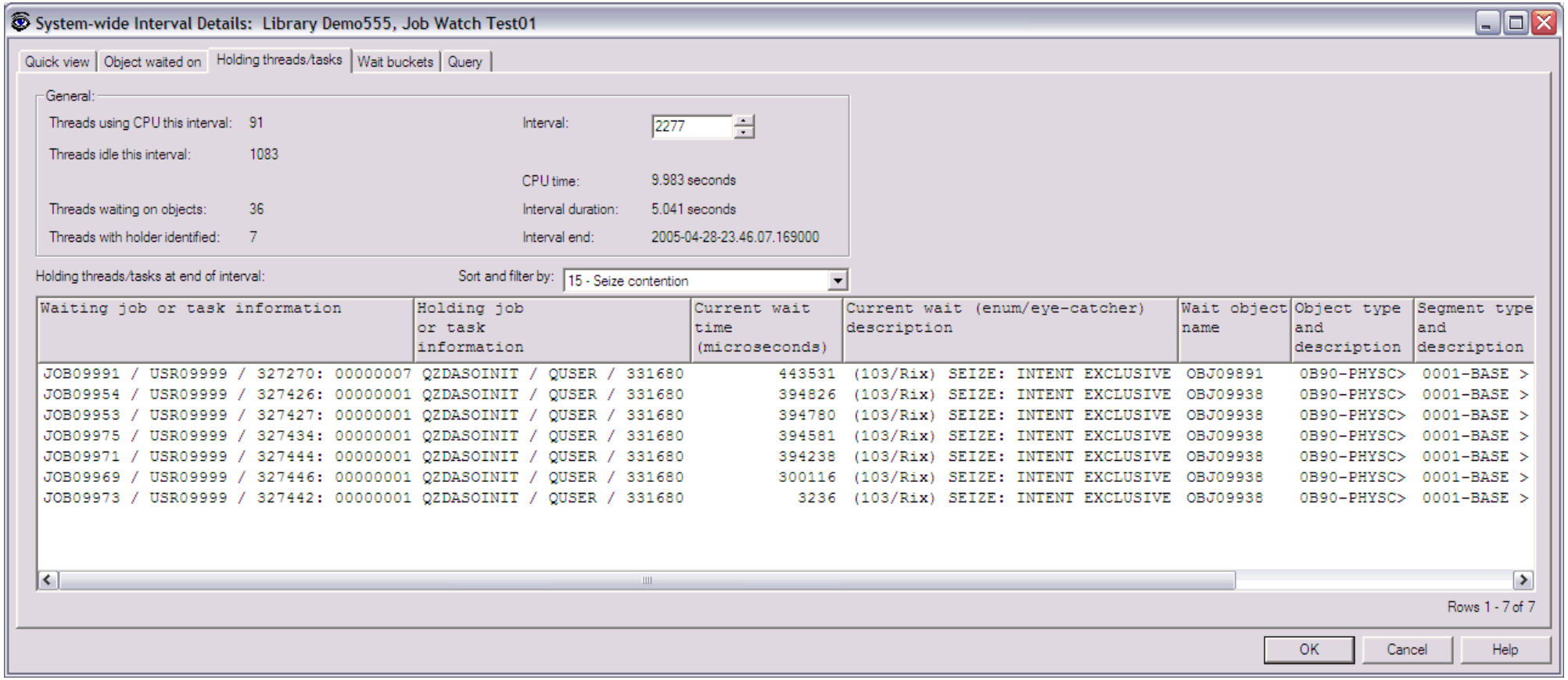
2.5.8.2 Object waited on

Threads waiting on objects	This number indicates the total threads and tasks that were waiting to use an object at the moment the interval snapshot was taken.
Threads with holder identified	This number indicates the total threads and tasks that were being held by another job/thread or task at the moment the interval snapshot was taken.
Interval duration	The total elapsed time from the start to end of the current interval.
Interval end	The date and time the current interval ended.
Sort and filter by	This option is used to filter the list of wait objects by a wait type. If this is set to a wait type greater than 3 the list will be filtered to only show wait objects for the selected wait type. Set the wait type on this window to 1, 2, or 3 in order to show all jobs/tasks with wait objects for the current interval.

2.5.8.3 Holding threads/tasks

This page shows a summary of information about the selected interval as well as a list of jobs being held by another job during the current interval.

By right-clicking on the list the user can drill one or more jobs (either the waiting or holding job) to see the run/wait time signatures for the jobs or any of the other detail reports starting at the current interval.



This window contains the following fields:

Field name	Description
Threads using CPU this interval	Displays the total threads and tasks that used some CPU during the current interval.
Threads idle this interval	Displays the total threads tasks on the system during the current interval.
Interval	This field displays the current interval. Using the arrow buttons allows the user to change the interval, or a number may be entered into the textbox and the window will be updated to display information about the interval number provided.
CPU time	Displays the total CPU time used across all jobs/thread/tasks for the current interval.
Threads waiting on objects	This number indicates the total threads and tasks that were waiting to use an object at the moment the interval snapshot was taken.

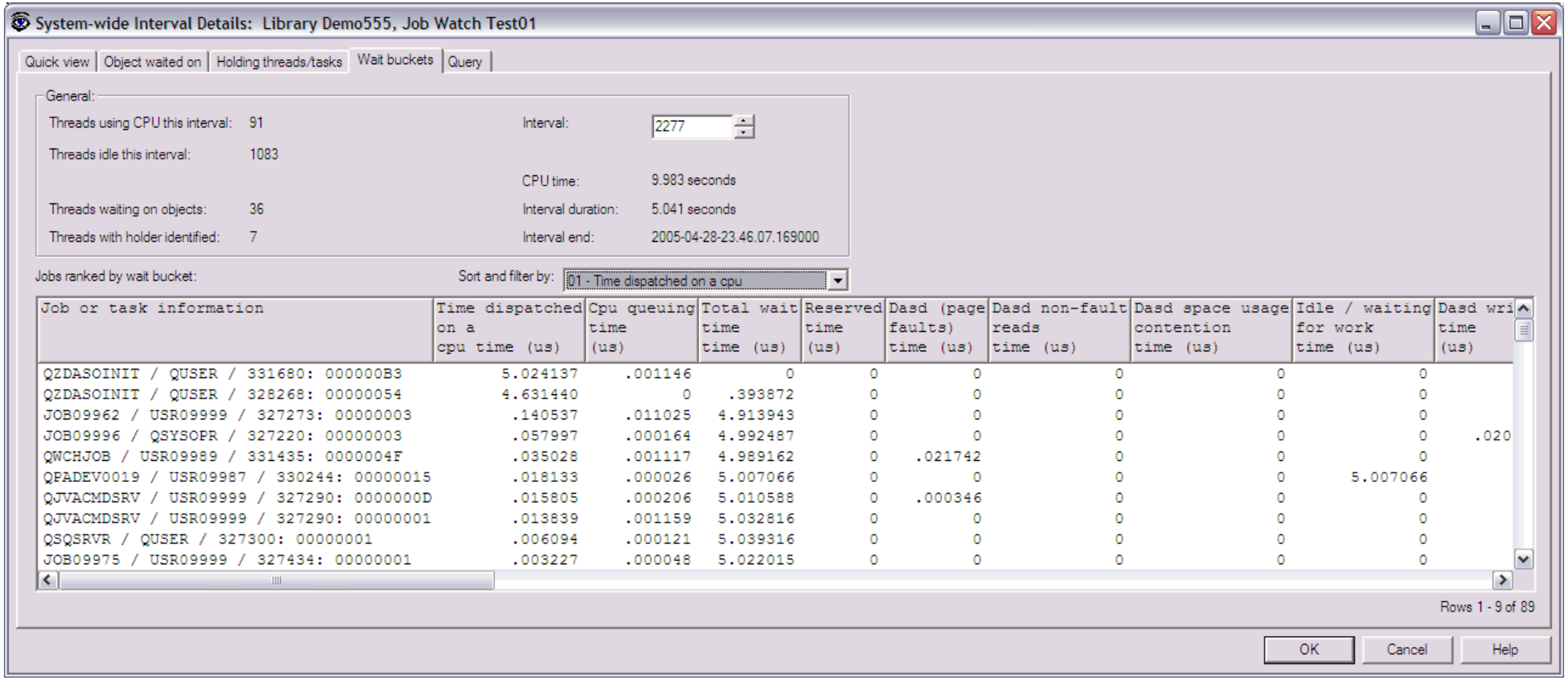
Threads with holder identified	This number indicates the total threads and tasks that were being held by another job/thread or task at the moment the interval snapshot was taken.
Interval duration	The total elapsed time from the start to end of the current interval.
Interval end	The date and time the current interval ended.
Sort and filter by	This option is used to filter the list of holders by a wait type. If this is set to a wait type greater than 3 the list will be filtered to only show holders where the current wait on the waiting job matches the selected wait type. Set the wait type on this window to 1, 2, or 3 in order to show all jobs/tasks with holders for the current interval.

Table of Contents | Previous | Next

2.5.8.4 Wait buckets

This page shows a summary of information about the selected interval as well as a list of the jobs ranked by one of the wait buckets. A drop down list of the possible wait buckets is available so the user can select which bucket they want to use for ranking the list of jobs. The jobs spending the most time in the selected wait will be shown first.

By right-clicking on the list the user can drill one or more jobs (either the waiting or holding job) to see the run/wait time signatures for the jobs or any of the other detail reports starting at the current interval.



This window contains the following fields:

Field name	Description
Threads using CPU this interval	Displays the total threads and tasks that used some CPU during the current interval.
Threads idle this interval	Displays the total threads tasks on the system during the current interval.
Interval	This field displays the current interval. Using the arrow buttons allows the user to change the interval, or a number may be entered into the textbox and the window will be updated to display information about the interval number provided.
CPU time	Displays the total CPU time used across all jobs/thread/tasks for the current interval.

2.5.8.4 Wait buckets

Threads waiting on objects	This number indicates the total threads and tasks that were waiting to use an object at the moment the interval snapshot was taken.
Threads with holder identified	This number indicates the total threads and tasks that were being held by another job/thread or task at the moment the interval snapshot was taken.
Interval duration	The total elapsed time from the start to end of the current interval.
Interval end	The date and time the current interval ended.

Table of Contents Previous Next

2.6 Job Watch detail reports

This section covers the detailed property pages, graphs and tables available in a job watch. These reports are available by selecting any job/thread/task in a table view or graph and right-clicking on the record or bar representing the job. For example, right-clicking on a bar (represent a job) in a summarized graph provides many different detailed reports showing information about just the specific job selected. In most cases each bar in a detail graph represents some value in the job's data reported every interval.

Table of Contents | Previous | Next

2.6.1 Interval Properties

The interval properties window displays information about a single job in a job watch for a specific interval. These properties contain several different panels which vary depending on the data available. Some panels are not shown if optional data has not been collected.

Getting to the interval details panel can be done by clicking on a bar in one of the detail graphs showing the intervals for a single job or by double-clicking a record in a report/table in a Job Watcher output file containing both a taskcount field and an interval field in the record. These two fields identify the job/thread/task and interval to display in the interval property pages.

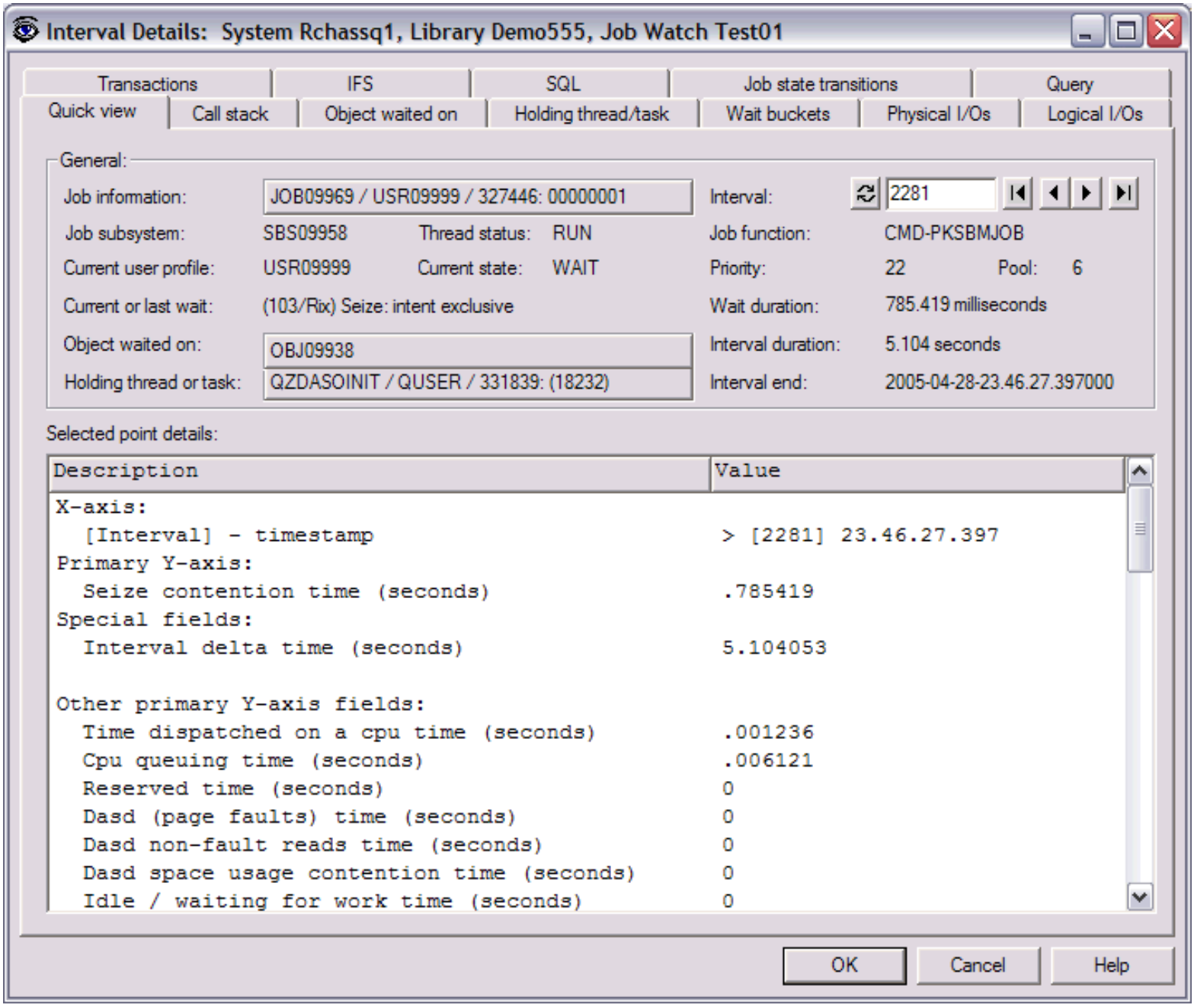


Table of Contents | Previous | Next

2.6.1.1 Record Quick View

This page of the interface is found by double-clicking on one of the Job Watcher output files from a table view. The window displays a record vertically from a table view. The other tabs in the view display additional information about the currently selected job and record . These additional tabs are only shown if a valid taskcount field and interval field is present in the current SQL query behind the table view. The taskcount field identifies the job/thread/task within the job watch.

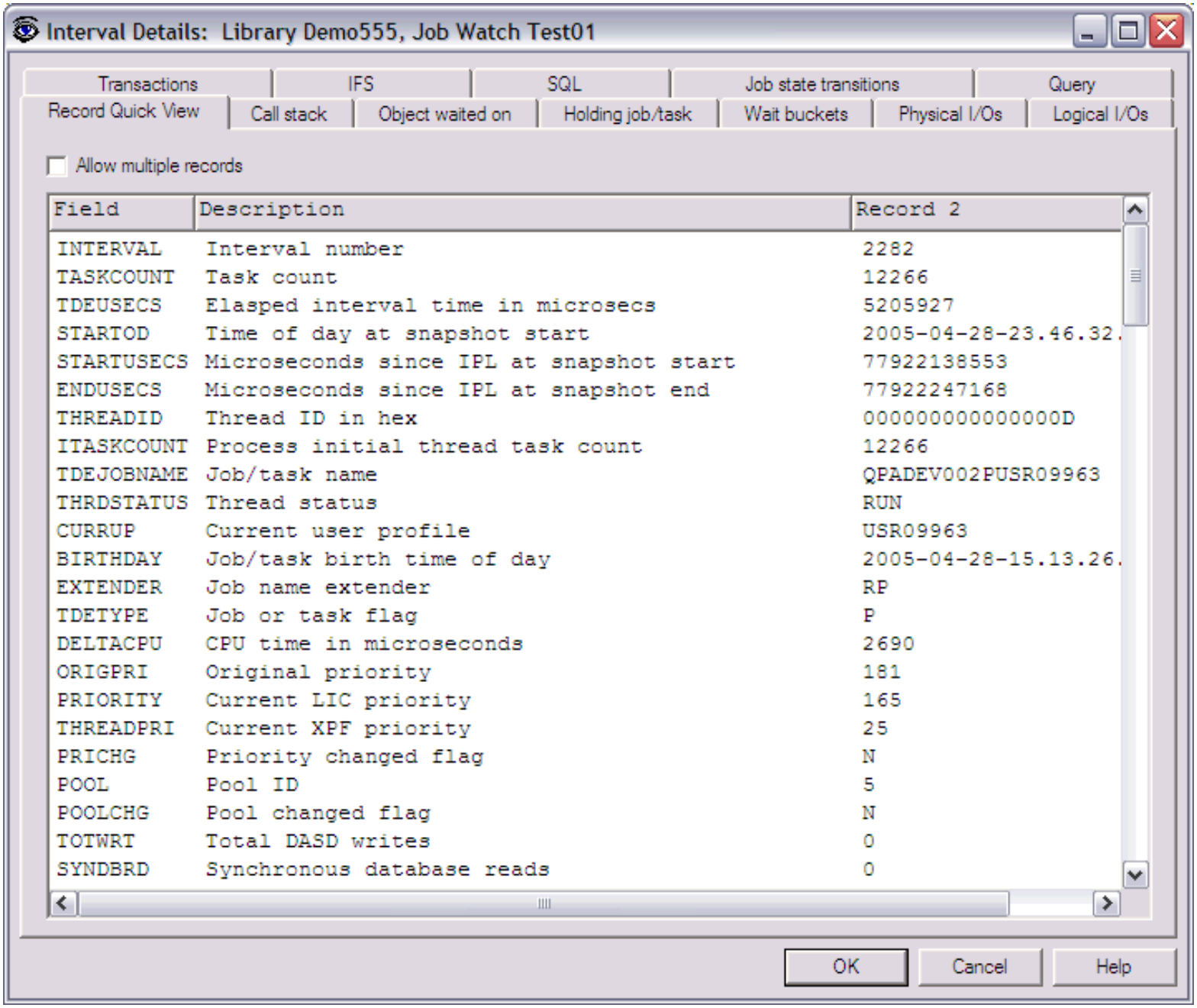
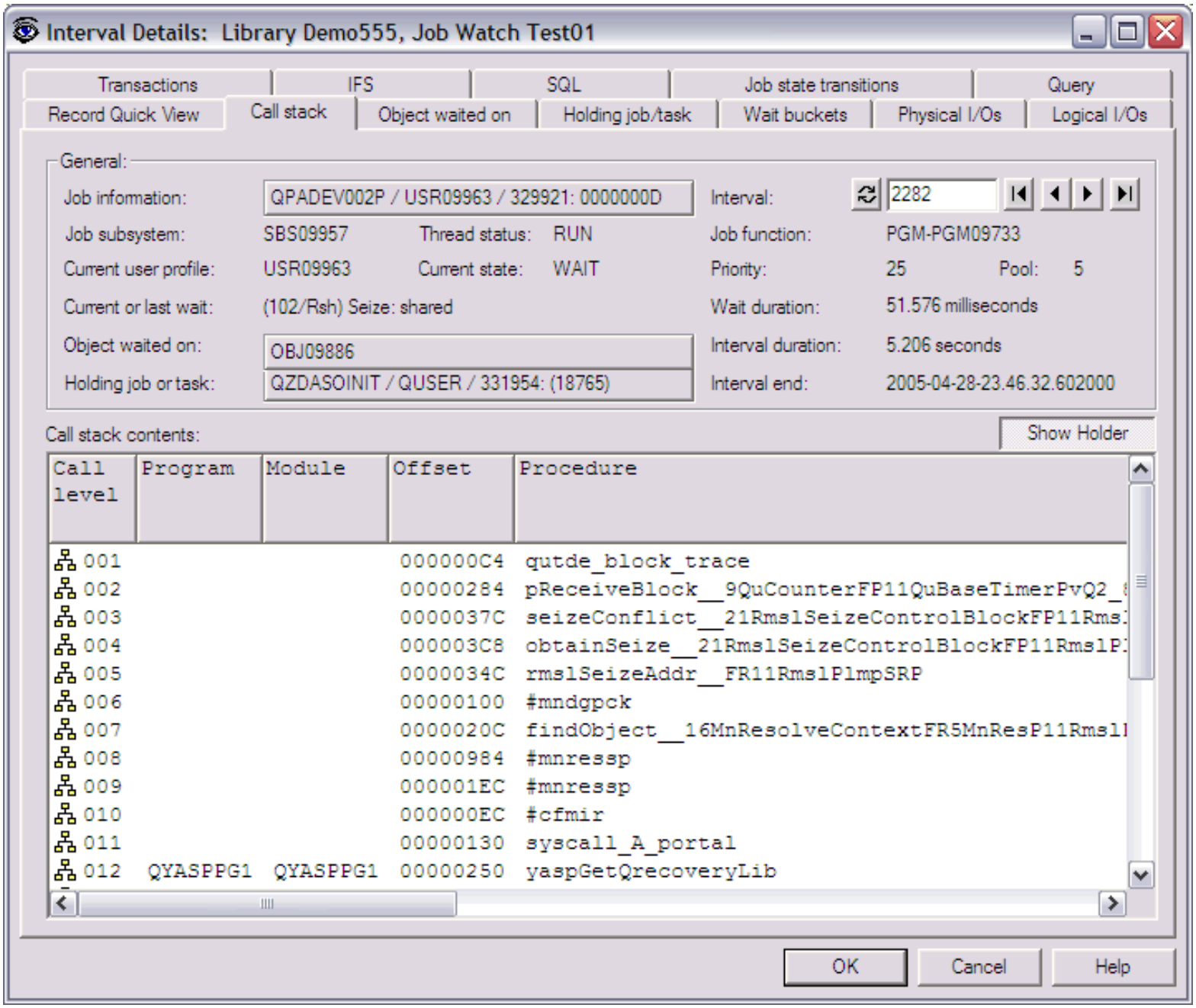


Table of Contents | Previous | Next

2.6.1.2 Call stack

This page shows the call stack (if available) for a job in a job watch at the moment the snapshot for the interval was taken. The call stack can be up to 50 levels deep, and is shown "bottom up". The call stack includes LIC information below the MI. Call stacks are optionally collected depending on the parameters specified during creation of the job watch.



This window contains the following fields:

General Field	Description

Job information	The fully qualified job/thread id or task information this interface applies to. Right-clicking on the job information provides a menu with several detail report options over the job.
Interval	Displays the selected interval the data shown applies to. The buttons allow the user to refresh the window or change the current interval position.
	Refresh the page using the interval number specified in the interval field.
	Moves the interval position to the previous active (used CPU) interval and refreshes the window. If on the call stack or SQL tabs this button will move to the previous interval that contains a call stack or SQL statement instead of returning the previous active interval.
	Decreases the interval number shown in the text field without refreshing the view. Holding down this button will increase the speed of the interval change.
	Increases the interval number shown in the text field without refreshing the view. Holding down this button will increase the speed of the interval change.
	Moves the interval position to the next active (used CPU) interval and refreshes the window. If on the call stack or SQL tabs this button will move to the next interval that contains a call stack or SQL statement instead of returning the next active interval.
Job subsystem	The subsystem the job/thread/task was running in during the current interval.
Thread status	This field displays the status of the job/thread for the current interval. This status is the same status shown for threads in the WRKACTJOB command.
Job function	This field displays the program/function the job/thread was executing for the current interval.
Current user profile	The user profile currently active when the job was running at the interval specified.
Priority	The priority of the job for the specified interval.
Pool	The pool number the job is running in for the specified interval.
Current or last wait	The current wait information contains information about the wait point (numeric identifier and 3 character wait code) and the description of the wait point from files QAPYJWBKT and QAIDRJWENM. This field indicates the type of wait (or CPU) that was occurring when the snapshot to gather information about the job was taken. The wait point shown belongs to a wait bucket. Information about the current bucket is available on the "Wait buckets" panel in this interface.
Wait duration	The total time the job has been in the current wait. Every time the wait type changes or CPU is used, this value is reset at 0. The current wait duration could be days for an idle job doing nothing and this would not indicate any problem. However if the current wait was a seize lock condition and the current wait duration was 30 seconds, this could flag a potential problem. For more information on understanding wait analysis, see the white paper available on the home page of the iDoctor for iSeries website.
Object waited on	This field contains information about the object the job/thread/task waited on for the current interval (if any) as well as the type of object being waited on. If an object is specified on this field, additional information about the object is available on the "Object waited on" panel in this interface. There are also reports available by right-clicking on the wait object name.
Interval duration	The time that elapsed when collecting the selected interval.
Holding thread or task	This field contains information about the job/thread/task that was holding the selected job during the selected interval and preventing it from doing work. If a holding thread or task is specified within this field right-clicking on the holder will provide a menu with additional report options related to the holding job.
Interval end	The date/time the current interval ended for the specified job.

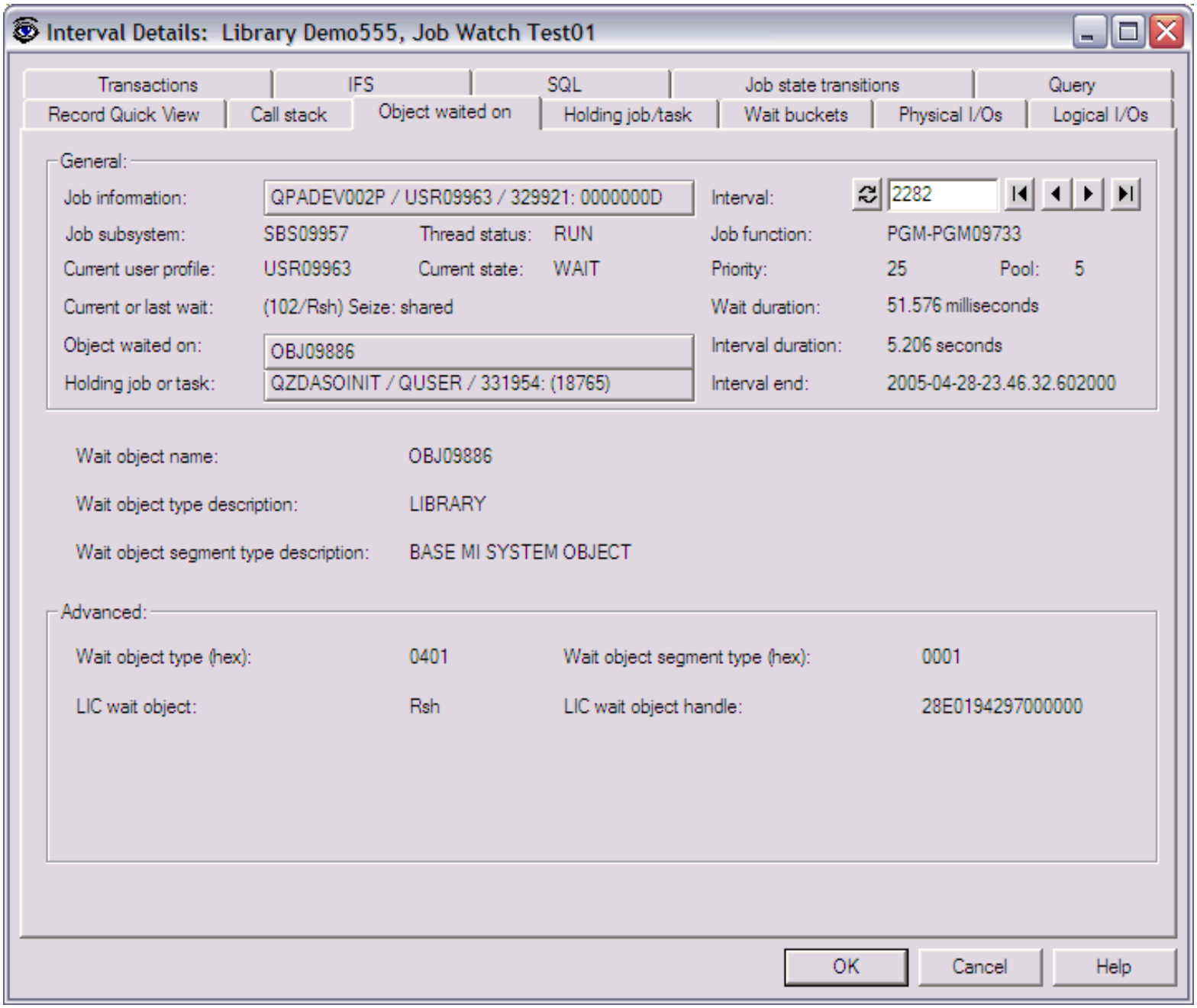
Specific Field	**Description**

Show Holder	This button allows the user to view the call stack for the holder if a holding job was present on the current interval. This button is only shown if the job on the property page had a holder for the current interval.
Show Waiter	This button allows the user to return to the waiter job's call stack after viewing the holding job's call stack. This button is only shown after the user has pressed the Show Holder button on the call stack panel of the interval details property pages.
Call stack	This contains program, module and procedure names within the current job's call stack for the specified interval. The call stack is gathered at the end of the interval. It is a snapshot of what programs/procedures were running in the job at this particular moment in time. The call stack can contain up to 50 levels of information. Right-clicking on a call stack entry will provide a menu with many report options to determine things like how often or for which other jobs the desired call stack entries occurred in the collection. If multiple call stack entries are selected the report will only match results where the same call stack entries occurred.

Table of Contents | Previous | Next

2.6.1.3 Object waited on

This page of the interval details shows information about the object the current job was waiting on at the moment the snapshot for the specified interval was taken.



This window contains the following fields:

General Field	Description

Job information	The fully qualified job/thread id or task information this interface applies to. Right-clicking on the job information provides a menu with several detail report options over the job.
Interval	Displays the selected interval the data shown applies to. The buttons allow the user to refresh the window or change the current interval position.
	Refresh the page using the interval number specified in the interval field.
	Moves the interval position to the previous active (used CPU) interval and refreshes the window. If on the call stack or SQL tabs this button will move to the previous interval that contains a call stack or SQL statement instead of returning the previous active interval.
	Decreases the interval number shown in the text field without refreshing the view. Holding down this button will increase the speed of the interval change.
	Increases the interval number shown in the text field without refreshing the view. Holding down this button will increase the speed of the interval change.
	Moves the interval position to the next active (used CPU) interval and refreshes the window. If on the call stack or SQL tabs this button will move to the next interval that contains a call stack or SQL statement instead of returning the next active interval.
Job subsystem	The subsystem the job/thread/task was running in during the current interval.
Thread status	This field displays the status of the job/thread for the current interval. This status is the same status shown for threads in the WRKACTJOB command.
Job function	This field displays the program/function the job/thread was executing for the current interval.
Current user profile	The user profile currently active when the job was running at the interval specified.
Priority	The priority of the job for the specified interval.
Pool	The pool number the job is running in for the specified interval.
Current or last wait	The current wait information contains information about the wait point (numeric identifier and 3 character wait code) and the description of the wait point from files QAPYJWBKT and QAIDRJWENM. This field indicates the type of wait (or CPU) that was occurring when the snapshot to gather information about the job was taken. The wait point shown belongs to a wait bucket. Information about the current bucket is available on the "Wait buckets" panel in this interface.
Wait duration	The total time the job has been in the current wait. Every time the wait type changes or CPU is used, this value is reset at 0. The current wait duration could be days for an idle job doing nothing and this would not indicate any problem. However if the current wait was a seize lock condition and the current wait duration was 30 seconds, this could flag a potential problem. For more information on understanding wait analysis, see the white paper available on the home page of the iDoctor for iSeries website.
Object waited on	This field contains information about the object the job/thread/task waited on for the current interval (if any) as well as the type of object being waited on. If an object is specified on this field, additional information about the object is available on the "Object waited on" panel in this interface. There are also reports available by right-clicking on the wait object name.
Interval duration	The time that elapsed when collecting the selected interval.
Holding thread or task	This field contains information about the job/thread/task that was holding the selected job during the selected interval and preventing it from doing work. If a holding thread or task is specified within this field right-clicking on the holder will provide a menu with additional report options related to the holding job.
Interval end	The date/time the current interval ended for the specified job.

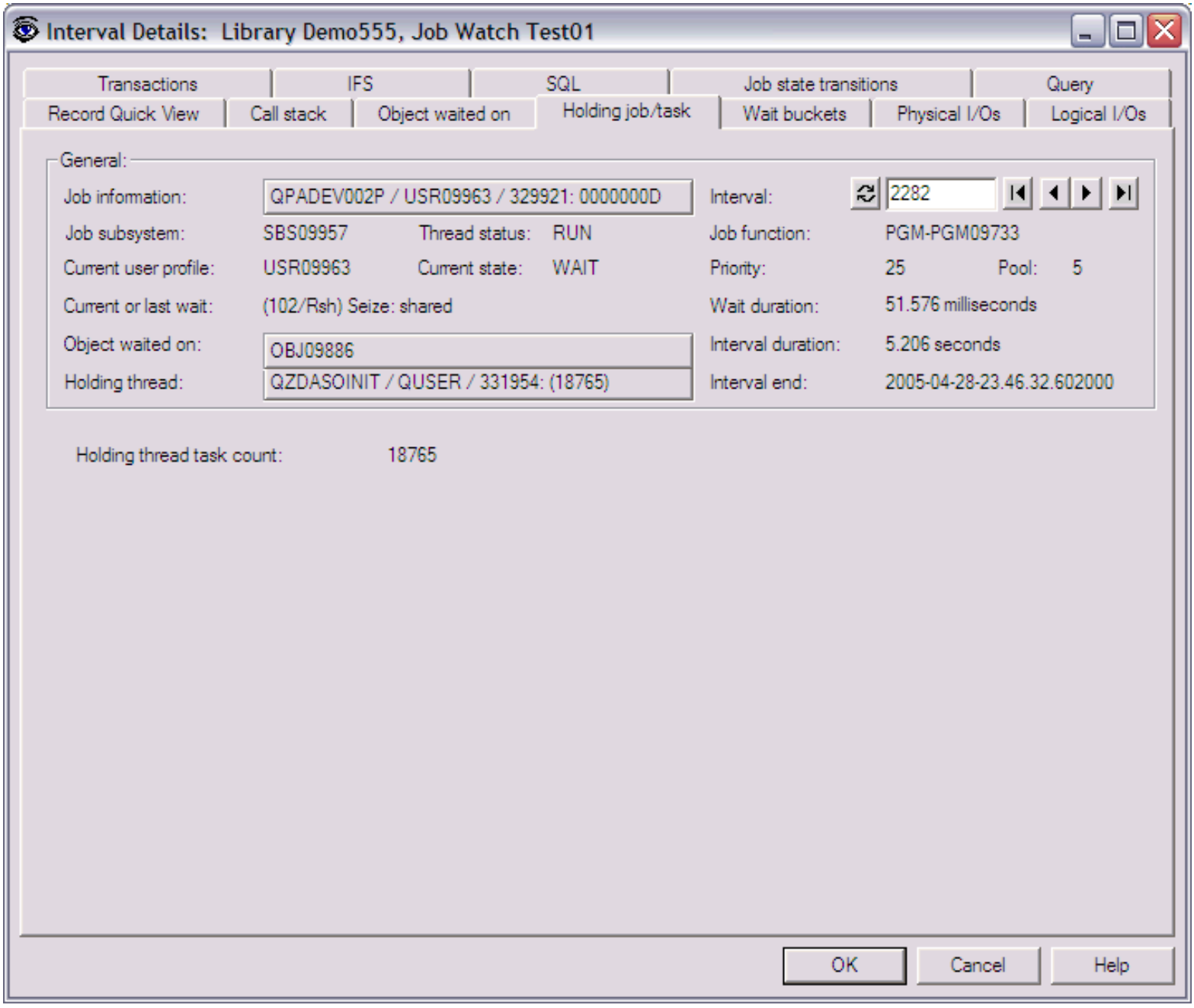
Specific Field	**Description**

Wait object name	The wait object name. This usually matches the object waited on field.
Wait object type description	A description of the wait object type. These descriptions can be found in file QAIDROT in library QIDRGUI.
Wait object segment type description	A description of the wait object segment type. These descriptions can be found in file QAIDRST in library QIDRGUI.
Wait object type (hex)	The 4 character object type that identifies the type of wait object.
Wait object segment type (hex)	The 4 character segment type that identifies the type of wait object segment.
LIC wait object	3 character identifer used in LIC to identify the type of wait.
LIC wait object handle	Address/handle that uniquely identifies the object waited on.

Table of Contents | Previous | Next

2.6.1.4 Holding job/task

This page shows additional information about the job holding the object waited on for the current job.



This window contains the following fields:

General Field	Description
Job information	The fully qualified job/thread id or task information this interface applies to. Right-clicking on the job information provides a menu with several detail report options over the job.

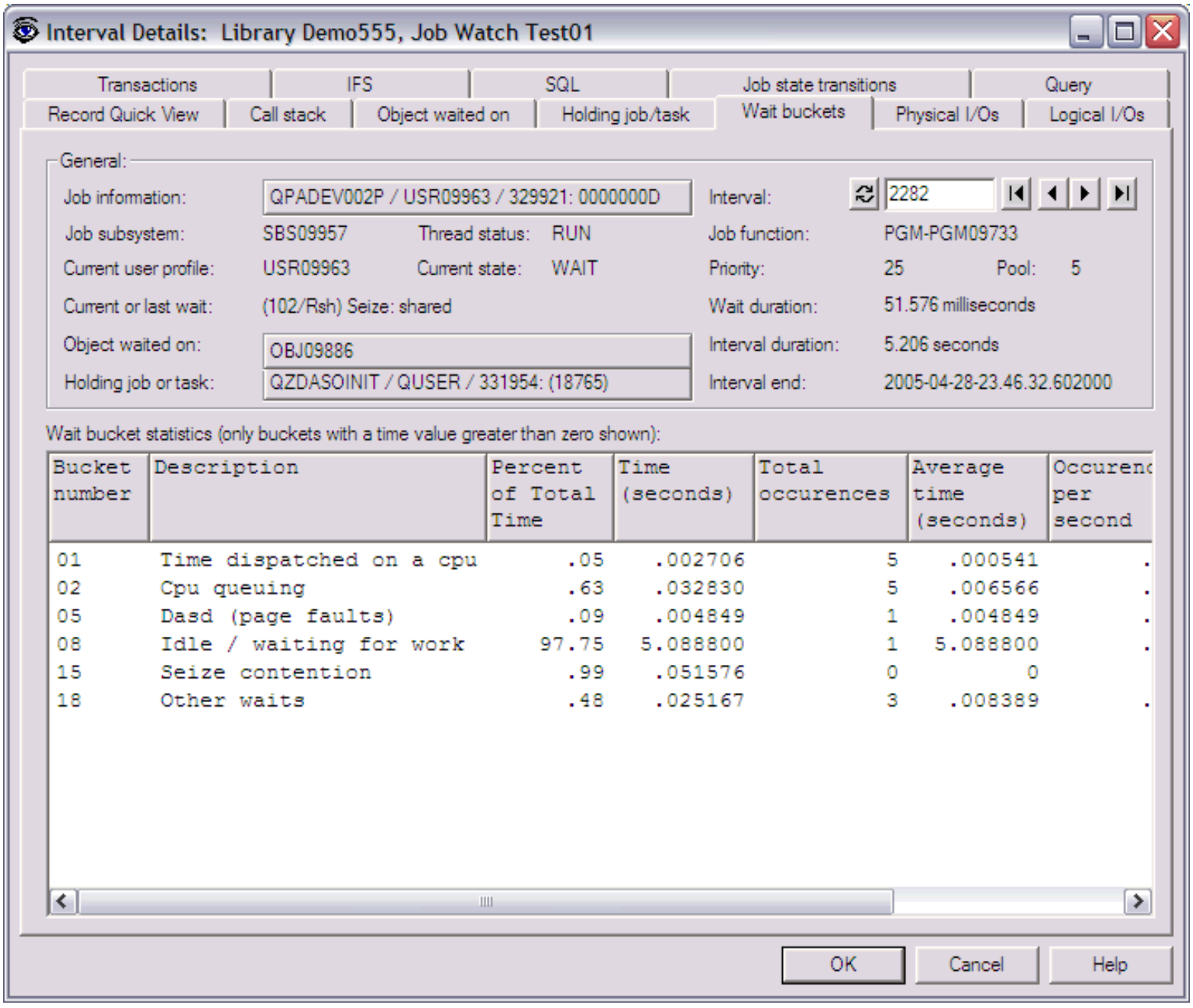
Interval	Displays the selected interval the data shown applies to. The buttons allow the user to refresh the window or change the current interval position.
	Refresh the page using the interval number specified in the interval field.
	Moves the interval position to the previous active (used CPU) interval and refreshes the window. If on the call stack or SQL tabs this button will move to the previous interval that contains a call stack or SQL statement instead of returning the previous active interval.
	Decreases the interval number shown in the text field without refreshing the view. Holding down this button will increase the speed of the interval change.
	Increases the interval number shown in the text field without refreshing the view. Holding down this button will increase the speed of the interval change.
	Moves the interval position to the next active (used CPU) interval and refreshes the window. If on the call stack or SQL tabs this button will move to the next interval that contains a call stack or SQL statement instead of returning the next active interval.
Job subsystem	The subsystem the job/thread/task was running in during the current interval.
Thread status	This field displays the status of the job/thread for the current interval. This status is the same status shown for threads in the WRKACTJOB command.
Job function	This field displays the program/function the job/thread was executing for the current interval.
Current user profile	The user profile currently active when the job was running at the interval specified.
Priority	The priority of the job for the specified interval.
Pool	The pool number the job is running in for the specified interval.
Current or last wait	The current wait information contains information about the wait point (numeric identifier and 3 character wait code) and the description of the wait point from files QAPYJWBKT and QAIDRJWENM. This field indicates the type of wait (or CPU) that was occurring when the snapshot to gather information about the job was taken. The wait point shown belongs to a wait bucket. Information about the current bucket is available on the "Wait buckets" panel in this interface.
Wait duration	The total time the job has been in the current wait. Every time the wait type changes or CPU is used, this value is reset at 0. The current wait duration could be days for an idle job doing nothing and this would not indicate any problem. However if the current wait was a seize lock condition and the current wait duration was 30 seconds, this could flag a potential problem. For more information on understanding wait analysis, see the white paper available on the home page of the iDoctor for iSeries website.
Object waited on	This field contains information about the object the job/thread/task waited on for the current interval (if any) as well as the type of object being waited on. If an object is specified on this field, additional information about the object is available on the "Object waited on" panel in this interface. There are also reports available by right-clicking on the wait object name.
Interval duration	The time that elapsed when collecting the selected interval.
Holding thread or task	This field contains information about the job/thread/task that was holding the selected job during the selected interval and preventing it from doing work. If a holding thread or task is specified within this field right-clicking on the holder will provide a menu with additional report options related to the holding job.
Interval end	The date/time the current interval ended for the specified job.

Specific Field	**Description**
Holding thread taskcount	The taskcount (uniquely identifies a job) for the holding job.

Table of Contents | Previous | Next

2.6.1.5 Wait buckets

This page shows the waits that occurred during the specified interval for the job indicated. The wait buckets are shown along with the percentage of time spent in each bucket. The final column in the list shows which bucket the current wait (listed in the 4th line of the general section) belongs to.



This window contains the following fields:

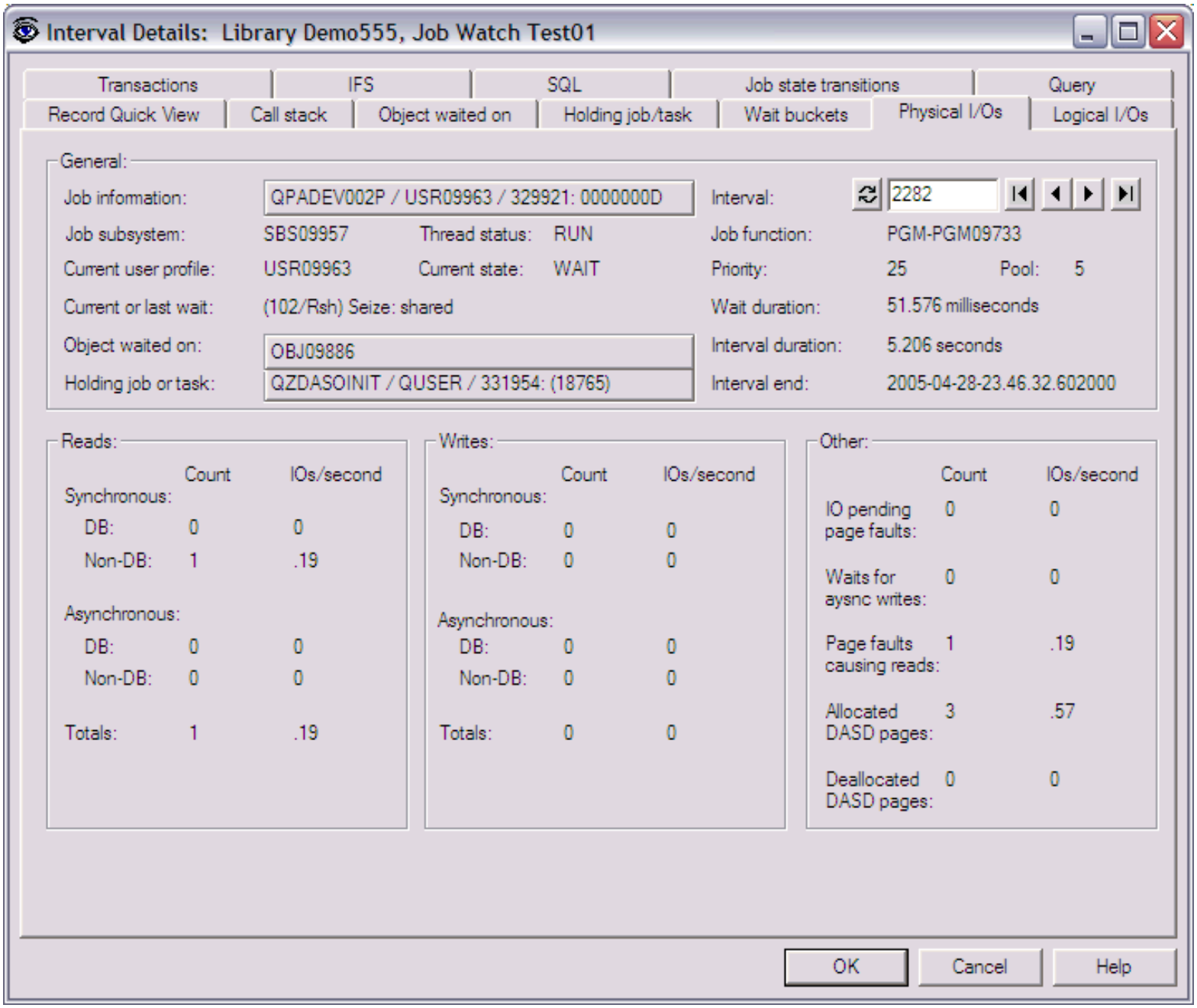
Field name	Description

Job information	The fully qualified job/thread id or task information this interface applies to. Right-clicking on the job information provides a menu with several detail report options over the job.
Interval	Displays the selected interval the data shown applies to. The buttons allow the user to refresh the window or change the current interval position.
	Refresh the page using the interval number specified in the interval field.
	Moves the interval position to the previous active (used CPU) interval and refreshes the window. If on the call stack or SQL tabs this button will move to the previous interval that contains a call stack or SQL statement instead of returning the previous active interval.
	Decreases the interval number shown in the text field without refreshing the view. Holding down this button will increase the speed of the interval change.
	Increases the interval number shown in the text field without refreshing the view. Holding down this button will increase the speed of the interval change.
	Moves the interval position to the next active (used CPU) interval and refreshes the window. If on the call stack or SQL tabs this button will move to the next interval that contains a call stack or SQL statement instead of returning the next active interval.
Job subsystem	The subsystem the job/thread/task was running in during the current interval.
Thread status	This field displays the status of the job/thread for the current interval. This status is the same status shown for threads in the WRKACTJOB command.
Job function	This field displays the program/function the job/thread was executing for the current interval.
Current user profile	The user profile currently active when the job was running at the interval specified.
Priority	The priority of the job for the specified interval.
Pool	The pool number the job is running in for the specified interval.
Current or last wait	The current wait information contains information about the wait point (numeric identifier and 3 character wait code) and the description of the wait point from files QAPYJWBKT and QAIDRJWENM. This field indicates the type of wait (or CPU) that was occurring when the snapshot to gather information about the job was taken. The wait point shown belongs to a wait bucket. Information about the current bucket is available on the "Wait buckets" panel in this interface.
Wait duration	The total time the job has been in the current wait. Every time the wait type changes or CPU is used, this value is reset at 0. The current wait duration could be days for an idle job doing nothing and this would not indicate any problem. However if the current wait was a seize lock condition and the current wait duration was 30 seconds, this could flag a potential problem. For more information on understanding wait analysis, see the white paper available on the home page of the iDoctor for iSeries website.
Object waited on	This field contains information about the object the job/thread/task waited on for the current interval (if any) as well as the type of object being waited on. If an object is specified on this field, additional information about the object is available on the "Object waited on" panel in this interface. There are also reports available by right-clicking on the wait object name.
Interval duration	The time that elapsed when collecting the selected interval.
Holding thread or task	This field contains information about the job/thread/task that was holding the selected job during the selected interval and preventing it from doing work. If a holding thread or task is specified within this field right-clicking on the holder will provide a menu with additional report options related to the holding job.
Interval end	The date/time the current interval ended for the specified job.

Table of Contents | Previous | Next

2.6.1.6 Physical I/Os

This window contains many details about the various types of physical I/Os that occurred for the specified job and interval. Statistics for reads and writes as well as page faults and disk space allocations/deallocations are provided on this panel.



This window contains the following fields:

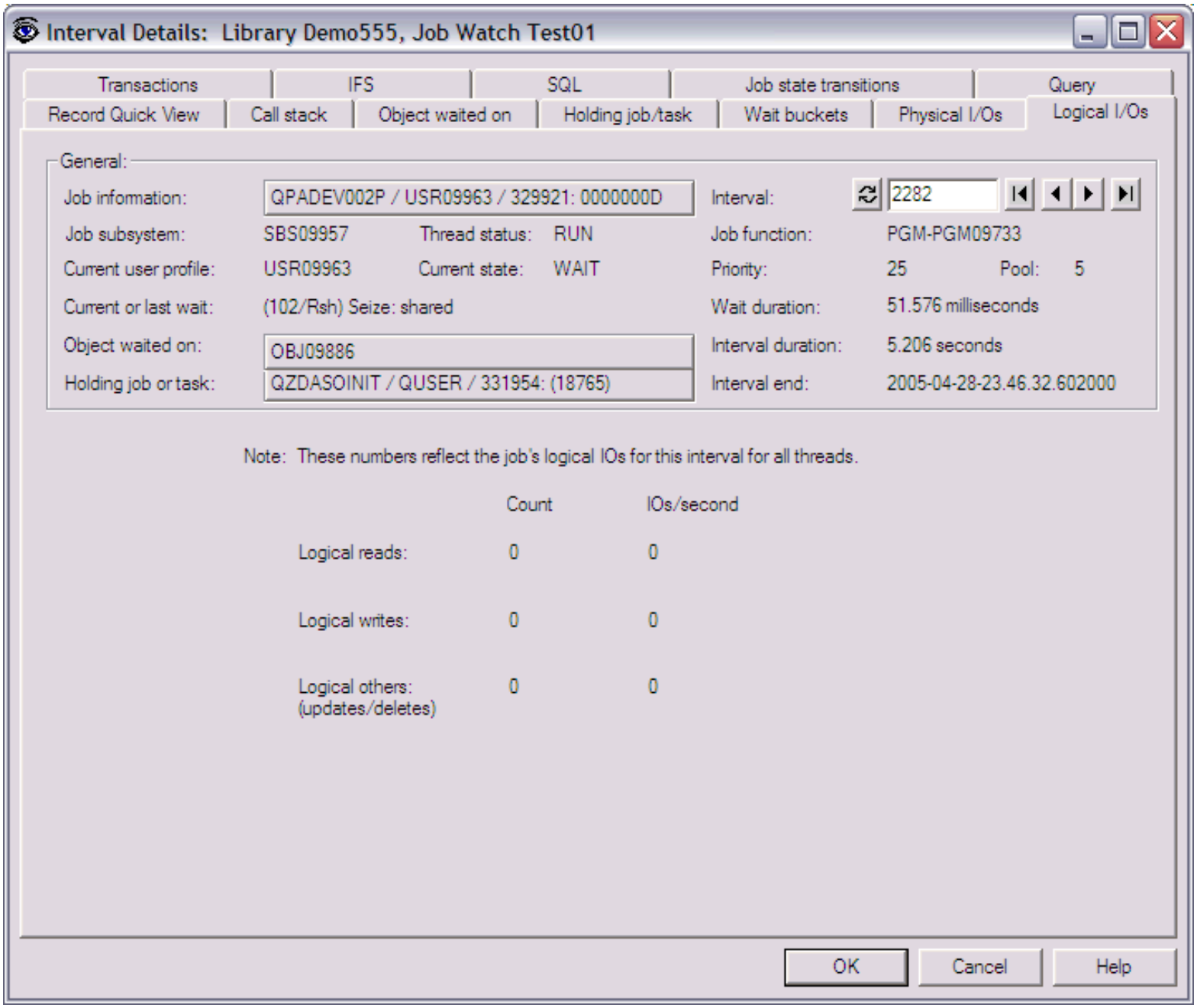
Field name	Description

Job information	The fully qualified job/thread id or task information this interface applies to. Right-clicking on the job information provides a menu with several detail report options over the job.
Interval	Displays the selected interval the data shown applies to. The buttons allow the user to refresh the window or change the current interval position.
	Refresh the page using the interval number specified in the interval field.
	Moves the interval position to the previous active (used CPU) interval and refreshes the window. If on the call stack or SQL tabs this button will move to the previous interval that contains a call stack or SQL statement instead of returning the previous active interval.
	Decreases the interval number shown in the text field without refreshing the view. Holding down this button will increase the speed of the interval change.
	Increases the interval number shown in the text field without refreshing the view. Holding down this button will increase the speed of the interval change.
	Moves the interval position to the next active (used CPU) interval and refreshes the window. If on the call stack or SQL tabs this button will move to the next interval that contains a call stack or SQL statement instead of returning the next active interval.
Job subsystem	The subsystem the job/thread/task was running in during the current interval.
Thread status	This field displays the status of the job/thread for the current interval. This status is the same status shown for threads in the WRKACTJOB command.
Job function	This field displays the program/function the job/thread was executing for the current interval.
Current user profile	The user profile currently active when the job was running at the interval specified.
Priority	The priority of the job for the specified interval.
Pool	The pool number the job is running in for the specified interval.
Current or last wait	The current wait information contains information about the wait point (numeric identifier and 3 character wait code) and the description of the wait point from files QAPYJWBKT and QAIDRJWENM. This field indicates the type of wait (or CPU) that was occurring when the snapshot to gather information about the job was taken. The wait point shown belongs to a wait bucket. Information about the current bucket is available on the "Wait buckets" panel in this interface.
Wait duration	The total time the job has been in the current wait. Every time the wait type changes or CPU is used, this value is reset at 0. The current wait duration could be days for an idle job doing nothing and this would not indicate any problem. However if the current wait was a seize lock condition and the current wait duration was 30 seconds, this could flag a potential problem. For more information on understanding wait analysis, see the white paper available on the home page of the iDoctor for iSeries website.
Object waited on	This field contains information about the object the job/thread/task waited on for the current interval (if any) as well as the type of object being waited on. If an object is specified on this field, additional information about the object is available on the "Object waited on" panel in this interface. There are also reports available by right-clicking on the wait object name.
Interval duration	The time that elapsed when collecting the selected interval.
Holding thread or task	This field contains information about the job/thread/task that was holding the selected job during the selected interval and preventing it from doing work. If a holding thread or task is specified within this field right-clicking on the holder will provide a menu with additional report options related to the holding job.
Interval end	The date/time the current interval ended for the specified job.

Table of Contents | Previous | Next

2.6.1.7 Logical I/Os

This page contains statistics for logical database I/Os occurred for the specified job and interval. The LDIO statistics includes rates and totals of reads, writes and updates/deletes.



This window contains the following fields:

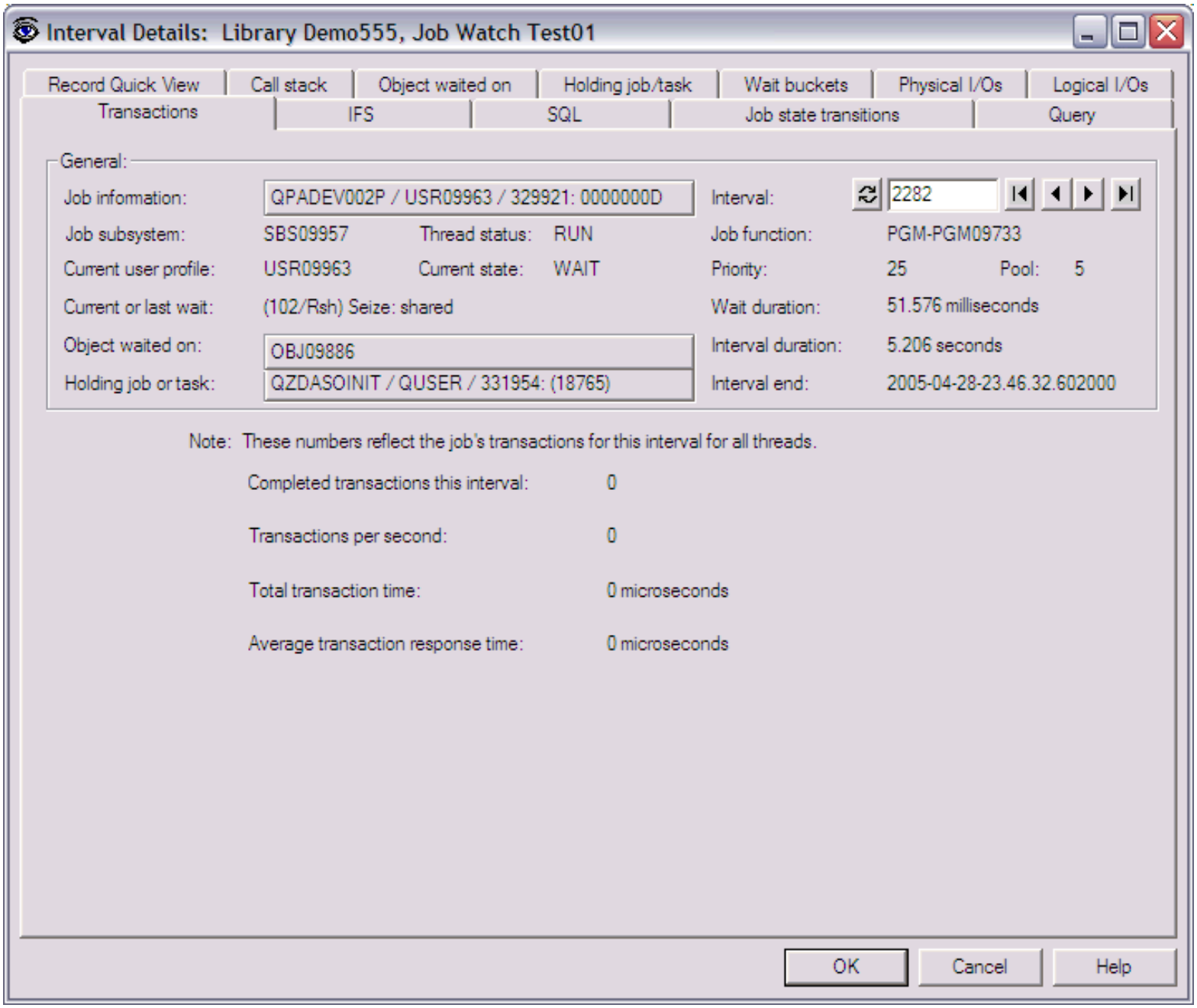
Field name	Description

Job information	The fully qualified job/thread id or task information this interface applies to. Right-clicking on the job information provides a menu with several detail report options over the job.
Interval	Displays the selected interval the data shown applies to. The buttons allow the user to refresh the window or change the current interval position.
	Refresh the page using the interval number specified in the interval field.
	Moves the interval position to the previous active (used CPU) interval and refreshes the window. If on the call stack or SQL tabs this button will move to the previous interval that contains a call stack or SQL statement instead of returning the previous active interval.
	Decreases the interval number shown in the text field without refreshing the view. Holding down this button will increase the speed of the interval change.
	Increases the interval number shown in the text field without refreshing the view. Holding down this button will increase the speed of the interval change.
	Moves the interval position to the next active (used CPU) interval and refreshes the window. If on the call stack or SQL tabs this button will move to the next interval that contains a call stack or SQL statement instead of returning the next active interval.
Job subsystem	The subsystem the job/thread/task was running in during the current interval.
Thread status	This field displays the status of the job/thread for the current interval. This status is the same status shown for threads in the WRKACTJOB command.
Job function	This field displays the program/function the job/thread was executing for the current interval.
Current user profile	The user profile currently active when the job was running at the interval specified.
Priority	The priority of the job for the specified interval.
Pool	The pool number the job is running in for the specified interval.
Current or last wait	The current wait information contains information about the wait point (numeric identifier and 3 character wait code) and the description of the wait point from files QAPYJWBKT and QAIDRJWENM. This field indicates the type of wait (or CPU) that was occurring when the snapshot to gather information about the job was taken. The wait point shown belongs to a wait bucket. Information about the current bucket is available on the "Wait buckets" panel in this interface.
Wait duration	The total time the job has been in the current wait. Every time the wait type changes or CPU is used, this value is reset at 0. The current wait duration could be days for an idle job doing nothing and this would not indicate any problem. However if the current wait was a seize lock condition and the current wait duration was 30 seconds, this could flag a potential problem. For more information on understanding wait analysis, see the white paper available on the home page of the iDoctor for iSeries website.
Object waited on	This field contains information about the object the job/thread/task waited on for the current interval (if any) as well as the type of object being waited on. If an object is specified on this field, additional information about the object is available on the "Object waited on" panel in this interface. There are also reports available by right-clicking on the wait object name.
Interval duration	The time that elapsed when collecting the selected interval.
Holding thread or task	This field contains information about the job/thread/task that was holding the selected job during the selected interval and preventing it from doing work. If a holding thread or task is specified within this field right-clicking on the holder will provide a menu with additional report options related to the holding job.
Interval end	The date/time the current interval ended for the specified job.

Table of Contents | Previous | Next

2.6.1.8 Transactions

This page of the interval details window provides statistics on the transactions that ocurred during the specified job and interval. The total number of completed transactions, transaction rate, and transaction times are provided on this window.



This window contains the following fields:

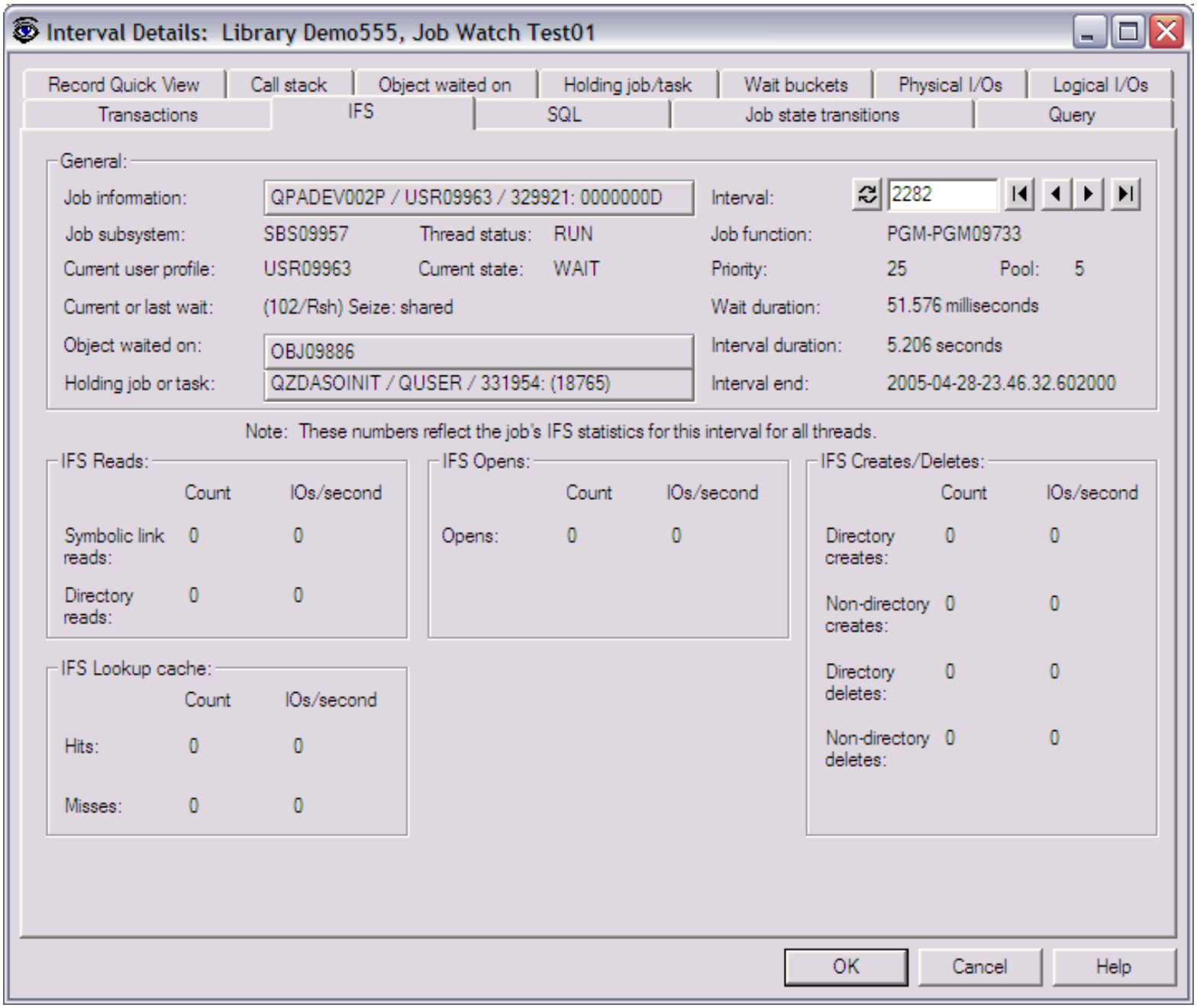
Field name	Description

Job information	The fully qualified job/thread id or task information this interface applies to. Right-clicking on the job information provides a menu with several detail report options over the job.
Interval	Displays the selected interval the data shown applies to. The buttons allow the user to refresh the window or change the current interval position.
	Refresh the page using the interval number specified in the interval field.
	Moves the interval position to the previous active (used CPU) interval and refreshes the window. If on the call stack or SQL tabs this button will move to the previous interval that contains a call stack or SQL statement instead of returning the previous active interval.
	Decreases the interval number shown in the text field without refreshing the view. Holding down this button will increase the speed of the interval change.
	Increases the interval number shown in the text field without refreshing the view. Holding down this button will increase the speed of the interval change.
	Moves the interval position to the next active (used CPU) interval and refreshes the window. If on the call stack or SQL tabs this button will move to the next interval that contains a call stack or SQL statement instead of returning the next active interval.
Job subsystem	The subsystem the job/thread/task was running in during the current interval.
Thread status	This field displays the status of the job/thread for the current interval. This status is the same status shown for threads in the WRKACTJOB command.
Job function	This field displays the program/function the job/thread was executing for the current interval.
Current user profile	The user profile currently active when the job was running at the interval specified.
Priority	The priority of the job for the specified interval.
Pool	The pool number the job is running in for the specified interval.
Current or last wait	The current wait information contains information about the wait point (numeric identifier and 3 character wait code) and the description of the wait point from files QAPYJWBKT and QAIDRJWENM. This field indicates the type of wait (or CPU) that was occurring when the snapshot to gather information about the job was taken. The wait point shown belongs to a wait bucket. Information about the current bucket is available on the "Wait buckets" panel in this interface.
Wait duration	The total time the job has been in the current wait. Every time the wait type changes or CPU is used, this value is reset at 0. The current wait duration could be days for an idle job doing nothing and this would not indicate any problem. However if the current wait was a seize lock condition and the current wait duration was 30 seconds, this could flag a potential problem. For more information on understanding wait analysis, see the white paper available on the home page of the iDoctor for iSeries website.
Object waited on	This field contains information about the object the job/thread/task waited on for the current interval (if any) as well as the type of object being waited on. If an object is specified on this field, additional information about the object is available on the "Object waited on" panel in this interface. There are also reports available by right-clicking on the wait object name.
Interval duration	The time that elapsed when collecting the selected interval.
Holding thread or task	This field contains information about the job/thread/task that was holding the selected job during the selected interval and preventing it from doing work. If a holding thread or task is specified within this field right-clicking on the holder will provide a menu with additional report options related to the holding job.
Interval end	The date/time the current interval ended for the specified job.

Table of Contents | Previous | Next

2.6.1.9 IFS

This page of the interval details window provides statistics on IFS activity that occurred during the specified job and interval. IFS reads, opens, creates/deletes and lookup cache hits and misses are provided on this window.



This window contains the following fields:

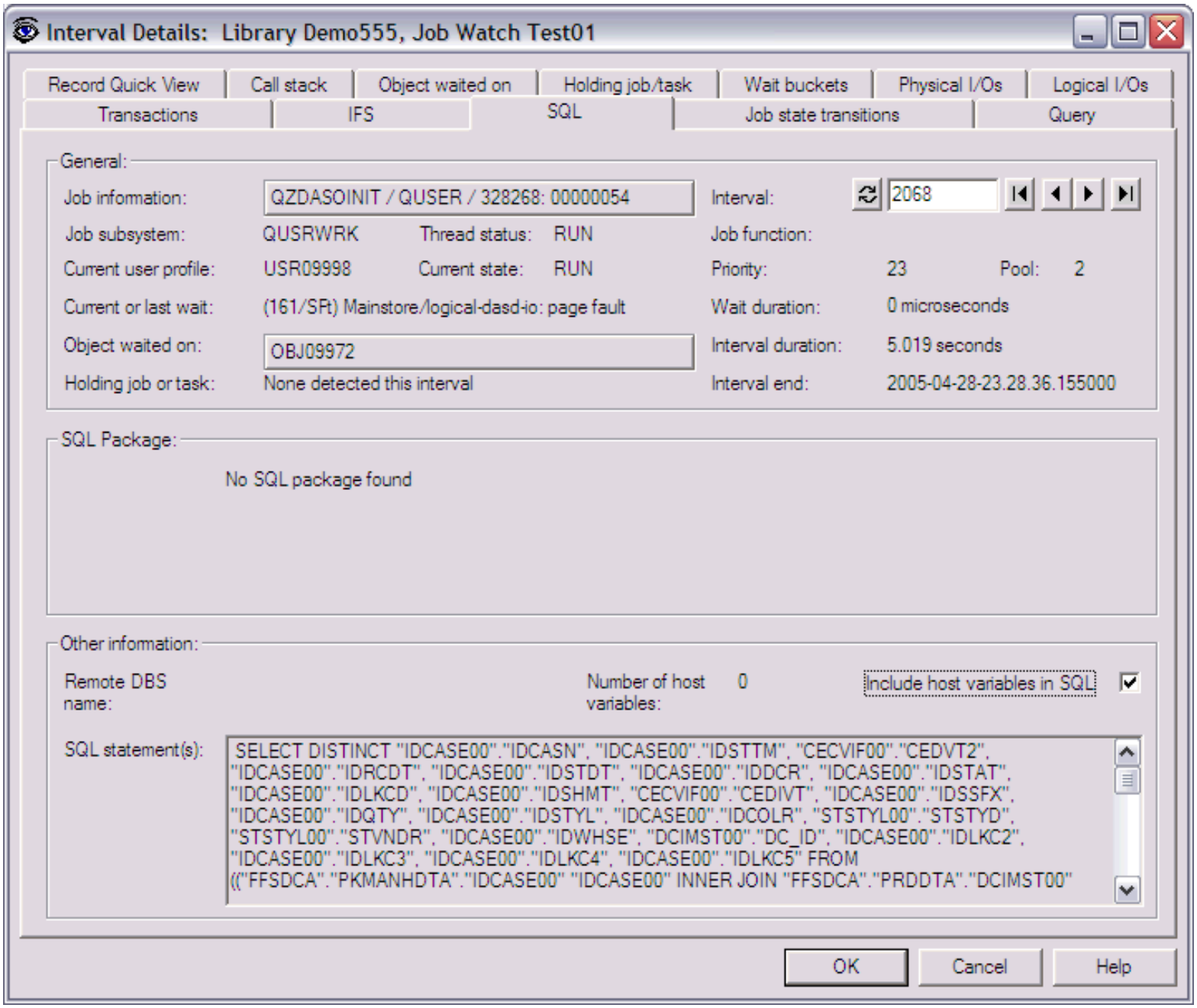
Field name	Description

Job information	The fully qualified job/thread id or task information this interface applies to. Right-clicking on the job information provides a menu with several detail report options over the job.
Interval	Displays the selected interval the data shown applies to. The buttons allow the user to refresh the window or change the current interval position.
	Refresh the page using the interval number specified in the interval field.
	Moves the interval position to the previous active (used CPU) interval and refreshes the window. If on the call stack or SQL tabs this button will move to the previous interval that contains a call stack or SQL statement instead of returning the previous active interval.
	Decreases the interval number shown in the text field without refreshing the view. Holding down this button will increase the speed of the interval change.
	Increases the interval number shown in the text field without refreshing the view. Holding down this button will increase the speed of the interval change.
	Moves the interval position to the next active (used CPU) interval and refreshes the window. If on the call stack or SQL tabs this button will move to the next interval that contains a call stack or SQL statement instead of returning the next active interval.
Job subsystem	The subsystem the job/thread/task was running in during the current interval.
Thread status	This field displays the status of the job/thread for the current interval. This status is the same status shown for threads in the WRKACTJOB command.
Job function	This field displays the program/function the job/thread was executing for the current interval.
Current user profile	The user profile currently active when the job was running at the interval specified.
Priority	The priority of the job for the specified interval.
Pool	The pool number the job is running in for the specified interval.
Current or last wait	The current wait information contains information about the wait point (numeric identifier and 3 character wait code) and the description of the wait point from files QAPYJWBKT and QAIDRJWENM. This field indicates the type of wait (or CPU) that was occurring when the snapshot to gather information about the job was taken. The wait point shown belongs to a wait bucket. Information about the current bucket is available on the "Wait buckets" panel in this interface.
Wait duration	The total time the job has been in the current wait. Every time the wait type changes or CPU is used, this value is reset at 0. The current wait duration could be days for an idle job doing nothing and this would not indicate any problem. However if the current wait was a seize lock condition and the current wait duration was 30 seconds, this could flag a potential problem. For more information on understanding wait analysis, see the white paper available on the home page of the iDoctor for iSeries website.
Object waited on	This field contains information about the object the job/thread/task waited on for the current interval (if any) as well as the type of object being waited on. If an object is specified on this field, additional information about the object is available on the "Object waited on" panel in this interface. There are also reports available by right-clicking on the wait object name.
Interval duration	The time that elapsed when collecting the selected interval.
Holding thread or task	This field contains information about the job/thread/task that was holding the selected job during the selected interval and preventing it from doing work. If a holding thread or task is specified within this field right-clicking on the holder will provide a menu with additional report options related to the holding job.
Interval end	The date/time the current interval ended for the specified job.

Table of Contents | Previous | Next

2.6.1.10 SQL

This page of the interval details window provides the SQL statement running at the time the snapshot was taken for the job in the job watch.



This window contains the following fields:

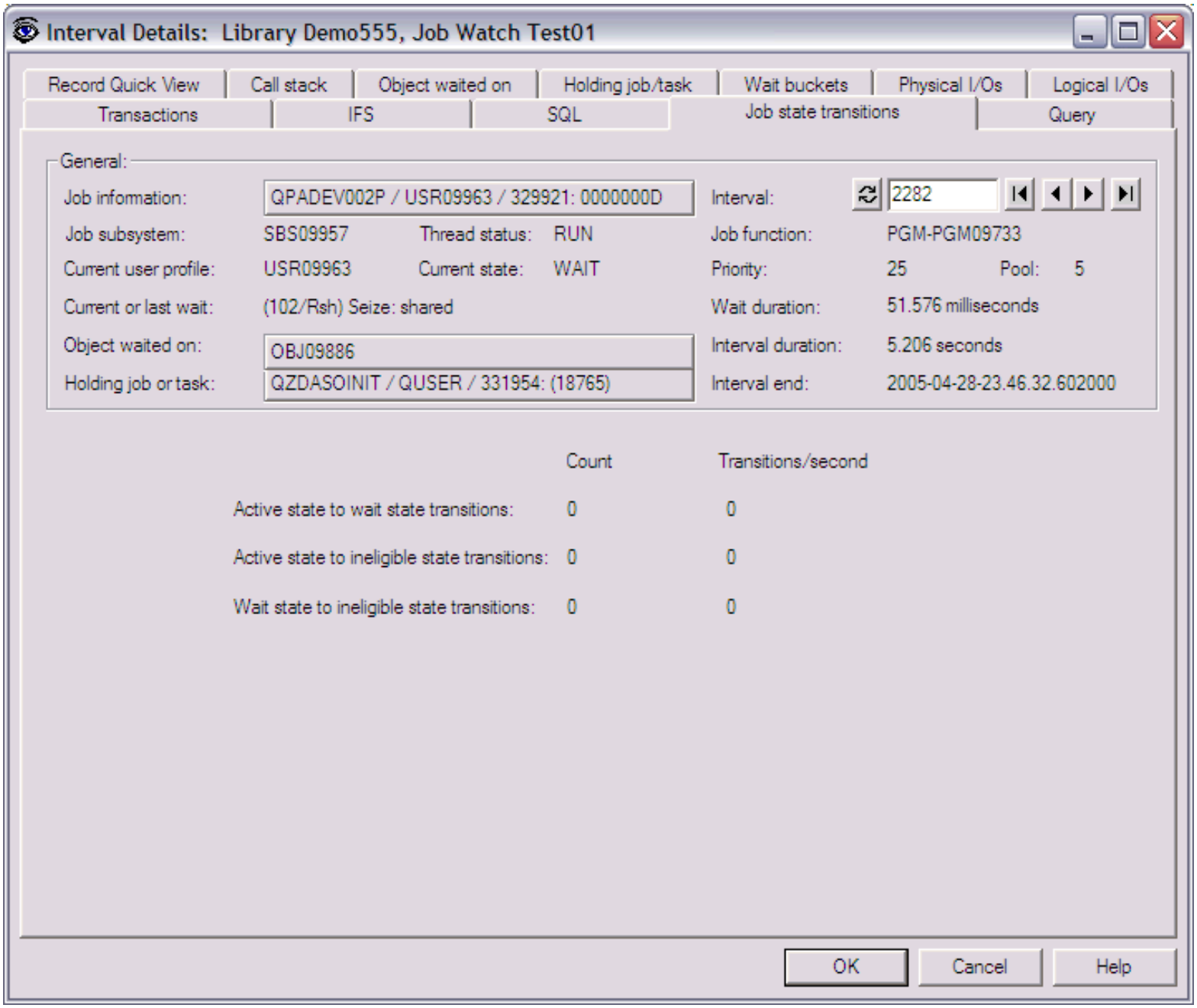
Field name	Description

Job information	The fully qualified job/thread id or task information this interface applies to. Right-clicking on the job information provides a menu with several detail report options over the job.
Interval	Displays the selected interval the data shown applies to. The buttons allow the user to refresh the window or change the current interval position.
	Refresh the page using the interval number specified in the interval field.
	Moves the interval position to the previous active (used CPU) interval and refreshes the window. If on the call stack or SQL tabs this button will move to the previous interval that contains a call stack or SQL statement instead of returning the previous active interval.
	Decreases the interval number shown in the text field without refreshing the view. Holding down this button will increase the speed of the interval change.
	Increases the interval number shown in the text field without refreshing the view. Holding down this button will increase the speed of the interval change.
	Moves the interval position to the next active (used CPU) interval and refreshes the window. If on the call stack or SQL tabs this button will move to the next interval that contains a call stack or SQL statement instead of returning the next active interval.
Job subsystem	The subsystem the job/thread/task was running in during the current interval.
Thread status	This field displays the status of the job/thread for the current interval. This status is the same status shown for threads in the WRKACTJOB command.
Job function	This field displays the program/function the job/thread was executing for the current interval.
Current user profile	The user profile currently active when the job was running at the interval specified.
Priority	The priority of the job for the specified interval.
Pool	The pool number the job is running in for the specified interval.
Current or last wait	The current wait information contains information about the wait point (numeric identifier and 3 character wait code) and the description of the wait point from files QAPYJWBKT and QAIDRJWENM. This field indicates the type of wait (or CPU) that was occurring when the snapshot to gather information about the job was taken. The wait point shown belongs to a wait bucket. Information about the current bucket is available on the "Wait buckets" panel in this interface.
Wait duration	The total time the job has been in the current wait. Every time the wait type changes or CPU is used, this value is reset at 0. The current wait duration could be days for an idle job doing nothing and this would not indicate any problem. However if the current wait was a seize lock condition and the current wait duration was 30 seconds, this could flag a potential problem. For more information on understanding wait analysis, see the white paper available on the home page of the iDoctor for iSeries website.
Object waited on	This field contains information about the object the job/thread/task waited on for the current interval (if any) as well as the type of object being waited on. If an object is specified on this field, additional information about the object is available on the "Object waited on" panel in this interface. There are also reports available by right-clicking on the wait object name.
Interval duration	The time that elapsed when collecting the selected interval.
Holding thread or task	This field contains information about the job/thread/task that was holding the selected job during the selected interval and preventing it from doing work. If a holding thread or task is specified within this field right-clicking on the holder will provide a menu with additional report options related to the holding job.
Interval end	The date/time the current interval ended for the specified job.

Table of Contents | Previous | Next

2.6.1.11 Job state transitions

This page of the interval details window provides statistics on the job state transitions that ocurred during the specified job and interval.



This window contains the following fields:

Field name	Description

Job information	The fully qualified job/thread id or task information this interface applies to. Right-clicking on the job information provides a menu with several detail report options over the job.
Interval	Displays the selected interval the data shown applies to. The buttons allow the user to refresh the window or change the current interval position.
	Refresh the page using the interval number specified in the interval field.
	Moves the interval position to the previous active (used CPU) interval and refreshes the window. If on the call stack or SQL tabs this button will move to the previous interval that contains a call stack or SQL statement instead of returning the previous active interval.
	Decreases the interval number shown in the text field without refreshing the view. Holding down this button will increase the speed of the interval change.
	Increases the interval number shown in the text field without refreshing the view. Holding down this button will increase the speed of the interval change.
	Moves the interval position to the next active (used CPU) interval and refreshes the window. If on the call stack or SQL tabs this button will move to the next interval that contains a call stack or SQL statement instead of returning the next active interval.
Job subsystem	The subsystem the job/thread/task was running in during the current interval.
Thread status	This field displays the status of the job/thread for the current interval. This status is the same status shown for threads in the WRKACTJOB command.
Job function	This field displays the program/function the job/thread was executing for the current interval.
Current user profile	The user profile currently active when the job was running at the interval specified.
Priority	The priority of the job for the specified interval.
Pool	The pool number the job is running in for the specified interval.
Current or last wait	The current wait information contains information about the wait point (numeric identifier and 3 character wait code) and the description of the wait point from files QAPYJWBKT and QAIDRJWENM. This field indicates the type of wait (or CPU) that was occurring when the snapshot to gather information about the job was taken. The wait point shown belongs to a wait bucket. Information about the current bucket is available on the "Wait buckets" panel in this interface.
Wait duration	The total time the job has been in the current wait. Every time the wait type changes or CPU is used, this value is reset at 0. The current wait duration could be days for an idle job doing nothing and this would not indicate any problem. However if the current wait was a seize lock condition and the current wait duration was 30 seconds, this could flag a potential problem. For more information on understanding wait analysis, see the white paper available on the home page of the iDoctor for iSeries website.
Object waited on	This field contains information about the object the job/thread/task waited on for the current interval (if any) as well as the type of object being waited on. If an object is specified on this field, additional information about the object is available on the "Object waited on" panel in this interface. There are also reports available by right-clicking on the wait object name.
Interval duration	The time that elapsed when collecting the selected interval.
Holding thread or task	This field contains information about the job/thread/task that was holding the selected job during the selected interval and preventing it from doing work. If a holding thread or task is specified within this field right-clicking on the holder will provide a menu with additional report options related to the holding job.
Interval end	The date/time the current interval ended for the specified job.

Table of Contents | Previous | Next

2.6.2 Wait graphs - detailed

The detailed wait graphs provide information about wait times that occurred in a job within a job watch.

This section covers the graphs available of this type.

2.6.2.1 Run/wait time signature

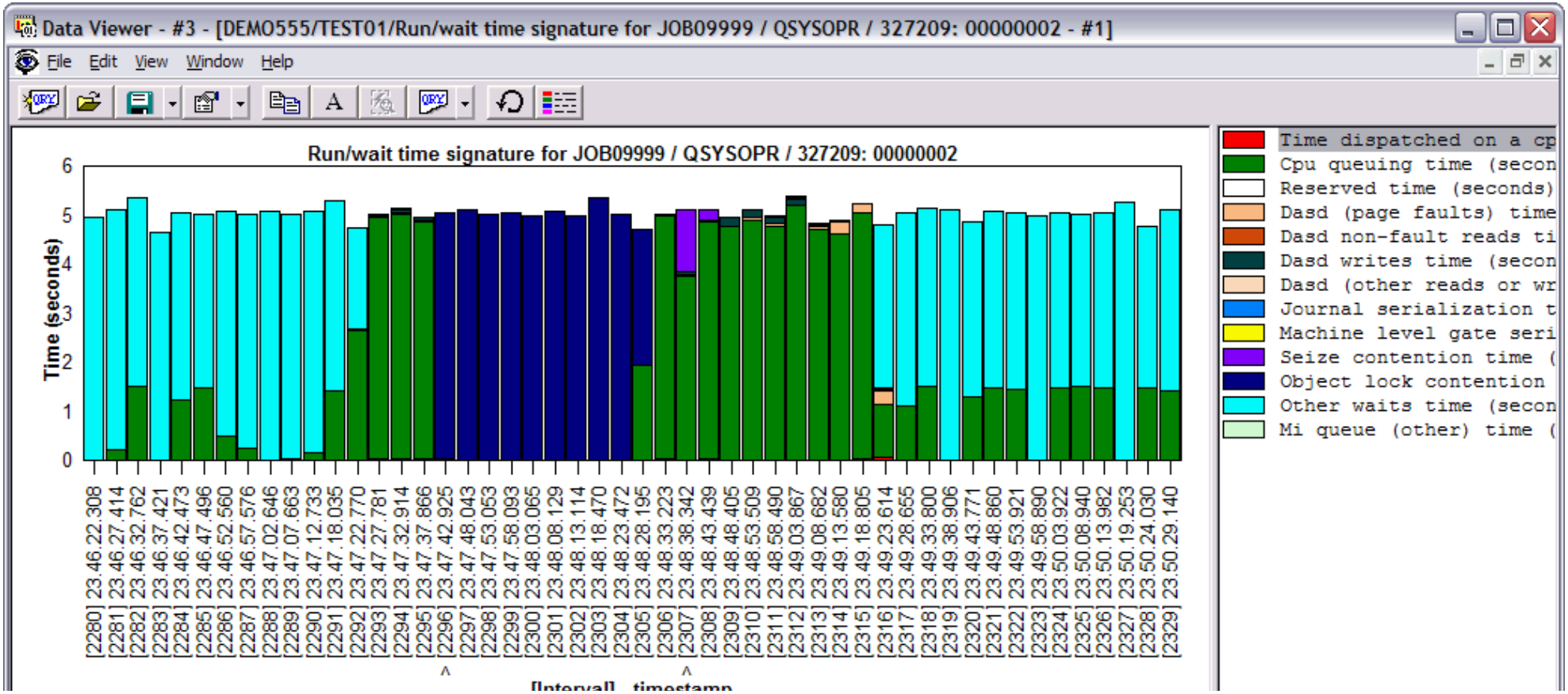
Description: This graph shows a **detailed** look at the time a job spent in various types of waits throughout its existence during the job watch. The graph makes use of the 32 run/wait buckets. To determine which type of wait a color represents, place the mouse pointer of a color/bar of interest and a description of the wait as well as other statistics about the interval/wait will be displayed.

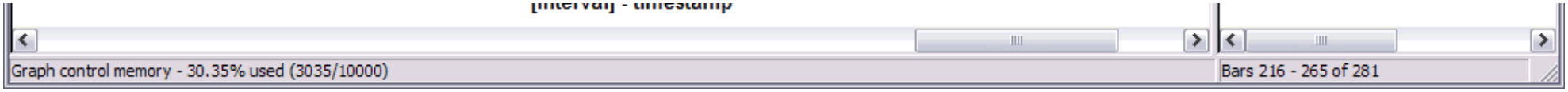
This graph can help pinpoint within a specific job where the job started to perform work and how efficiently it was performing that work. If a interval in the graph suddenly becomes suspicious (high seize time for example) clicking on the bar will provide other valuable information through the interval details window.

The total height of the stacked bar (which includes all waits for that interval and CPU) indicates the elapsed interval time.

Graph Type: detailed for a single job (stacked vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: Each color represents the amount of time the job/thread/task spent in one of the 32 different wait buckets within an interval. Time is listed in seconds.

Example:





Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Holder	Contains menu options for detail reports over the job holding the object being waited on for the selected interval.
Display call stack	Shows the call stack within the interval detail property pages for the interval selected.
Run/wait graphs by job	This menu allows the user to view the jobs for a paticular interval or interval range ranked by the desired wait type. By right-clicking on the graph over a particular color (such as dark blue for object locks) the user can drill into an interval to view the jobs that contributed to the selected color.
I/O graphs by job	Displays I/O and disk activity graphs for a particular interval or interval range.
Wait graphs by interval	This menu allows the user to view one of the wait graphs by interval starting at the interval range selected (or right-clicked on).
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar/job in the interval details window. This window will also appear by left-clicking on any bar in the graph.

Table of Contents | Previous | Next

2.6.2.2 Run/wait time signature with dispatch CPU breakdown

Description: This graph shows a **detailed** look at the time a job spent in various types of waits throughout its existence during the job watch. The graph makes use of the 32 run/wait buckets. To determine which type of wait a color represents, place the mouse pointer of a color/bar of interest and a description of the wait as well as other statistics about the interval/wait will be displayed.

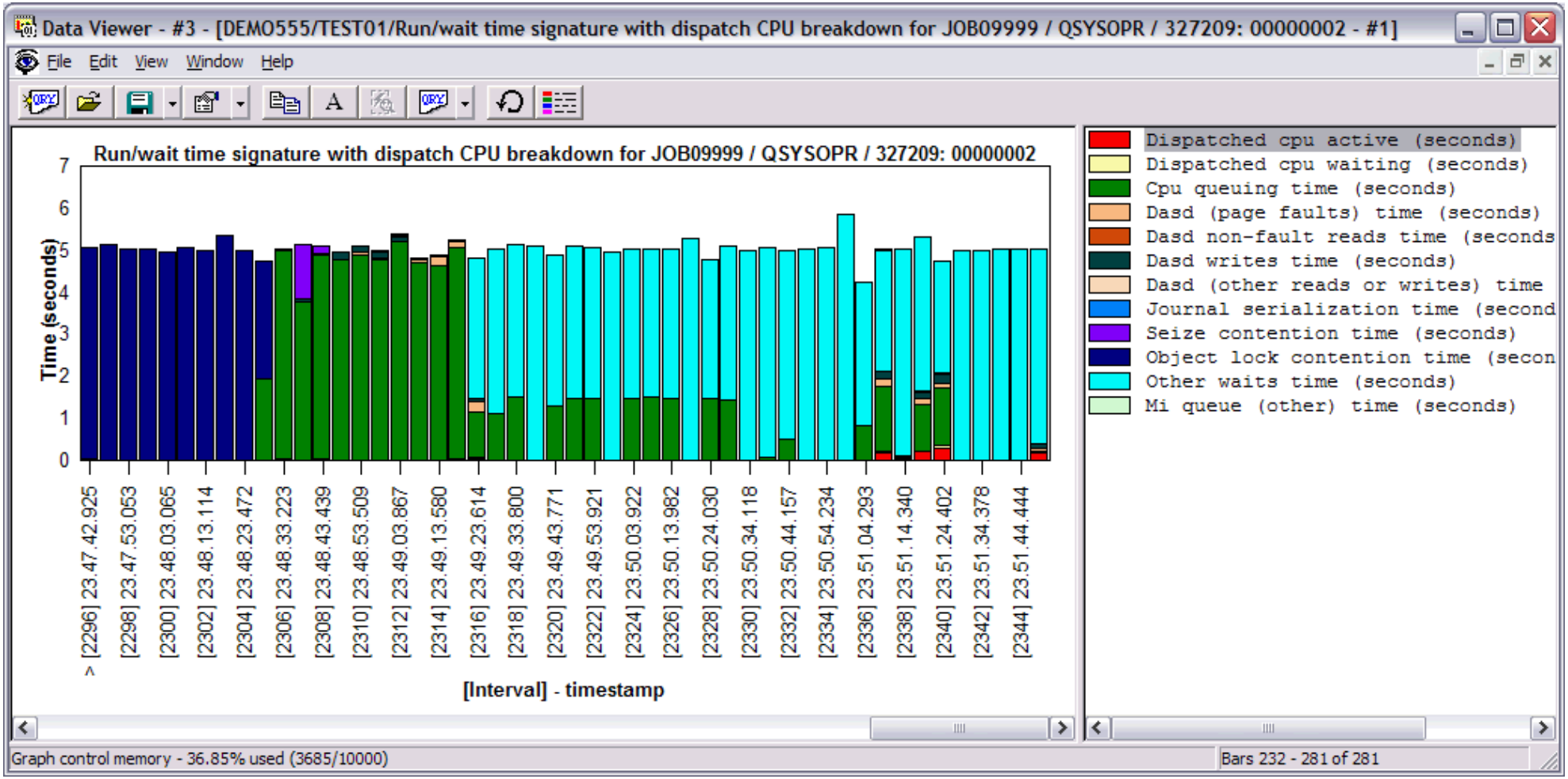
This graph is the same as the Run/wait time signature detail graph except the Dispatch CPU time bucket (QTIME01) is broken down into 2 components: dispatched CPU active and dispatched CPU waiting. Dispatched CPU waiting shows how much time each interval the job was dispatched on the CPU chip but did not actually use the CPU.

This graph can help pinpoint within a specific job where the job started to perform work and how efficiently it was performing that work. If a interval in the graph suddenly becomes suspicious (high seize time for example) clicking on the bar will provide other valuable information through the interval details window.

The total height of the stacked bar (which includes all waits for that interval and CPU) indicates the elapsed interval time.

Graph Type: detailed for a single job (stacked vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: Each color represents the amount of time the job/thread/task spent in one of the 32 different wait buckets within an interval. Time is listed in seconds.

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Holder	Contains menu options for detail reports over the job holding the object being waited on for the selected interval.
Display call stack	Shows the call stack within the interval detail property pages for the interval selected.
Run/wait graphs by job	This menu allows the user to view the jobs for a paticular interval or interval range ranked by the desired wait type. By right-clicking on the graph over a particular color (such as dark blue for object locks) the user can drill into an interval to view the jobs that contributed to the selected color.
I/O graphs by job	Displays I/O and disk activity graphs for a particular interval or interval range.
Wait graphs by interval	This menu allows the user to view one of the wait graphs by interval starting at the interval range selected (or right-clicked on).
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.

Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar/job in the interval details window. This window will also appear by left-clicking on any bar in the graph.

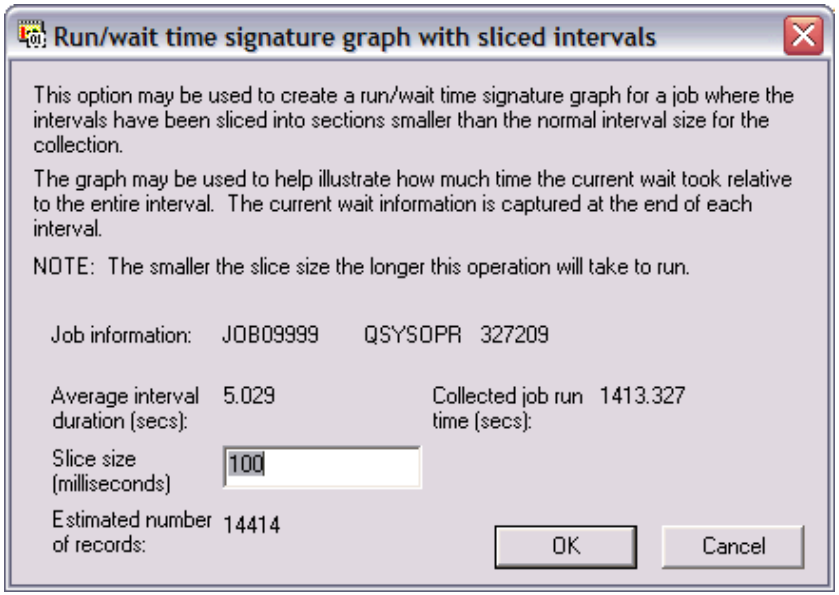
Table of Contents | Previous | Next

2.6.2.3 Run/wait time signature with sliced intervals

Description: This graph shows a **detailed** look at the time a job spent in various types of waits throughout its existence during the job watch. The graph makes use of the 32 run/wait buckets. To determine which type of wait a color represents, place the mouse pointer of a color/bar of interest and a description of the wait as well as other statistics about the interval/wait will be displayed.

This graph shows a breakdown of the wait bucket times as percentages over a specified interval size (which is generally less than 1 second). This advanced feature allows a user to more easily see what type of wait was occuring at the end of each Job Watcher interval and how long the wait was in relation to the rest of the interval.

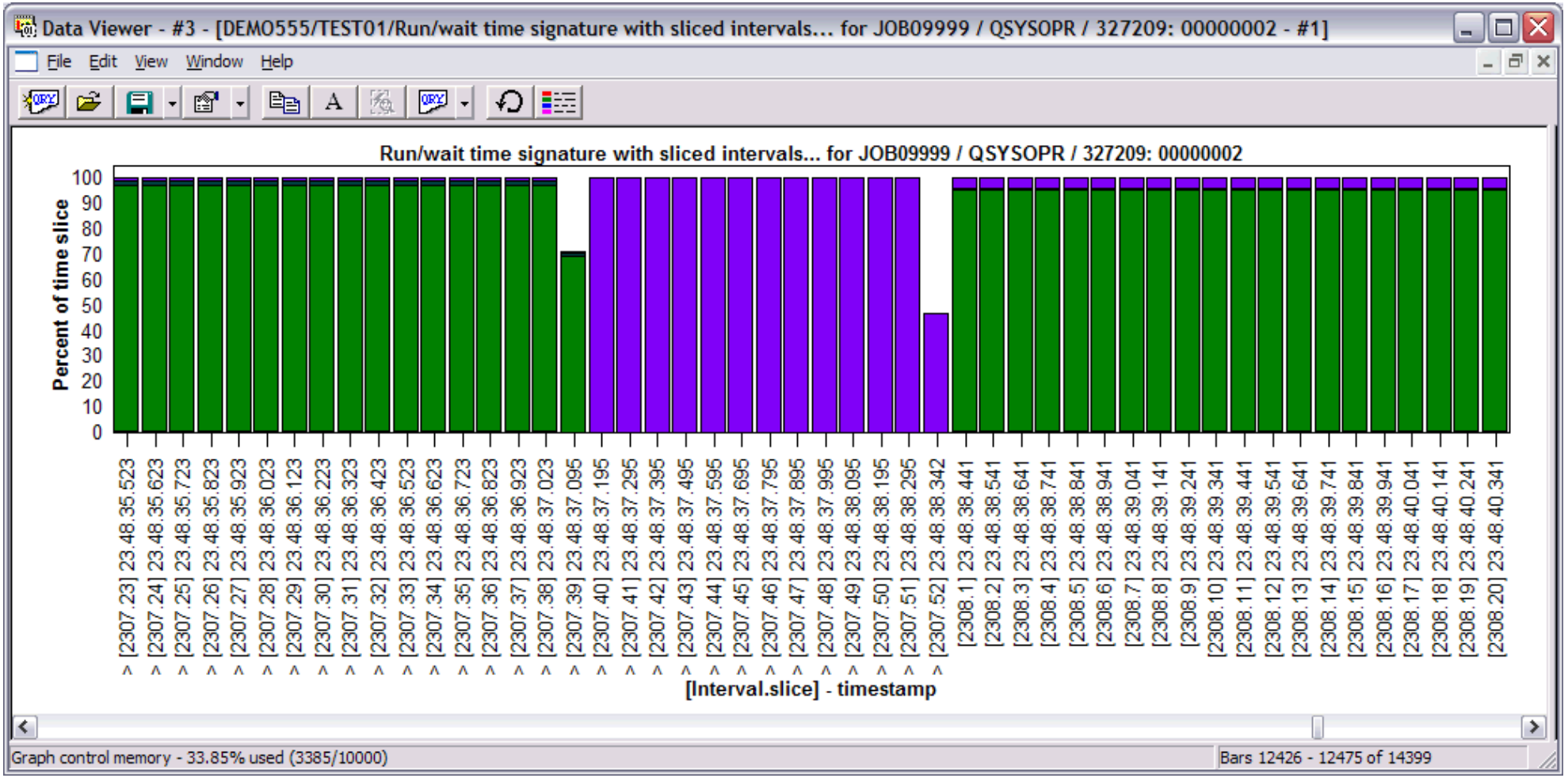
When choosing the menu for this graph the user will be prompted with the following interface which asks for the interval slice size (in milliseconds) The default slice size is 100 milliseconds or 1/10th of a second. Over a 5 second interval this would mean it would take approximately 50 bars in the graph to represent 1 Job Watcher interval.



The total height of the stacked bar (which includes all waits for that interval and CPU) indicates the elapsed interval time.

Graph Type: detailed for a single job (stacked vertical bar)
X-axis: Interval number . slice interval number and time of day. The time of day is listed as hh:mm:ss:xxx.
Y-Axis: Each color represents the amount of time the job/thread/task spent in one of the 32 different wait buckets within an interval. Time is listed in seconds.

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Holder	Contains menu options for detail reports over the job holding the object being waited on for the selected interval.
Display call stack	Shows the call stack within the interval detail property pages for the interval selected.
Run/wait graphs by job	This menu allows the user to view the jobs for a paticular interval or interval range ranked by the desired wait type. By right-clicking on the graph over a particular color (such as dark blue for object locks) the user can drill into an interval to view the jobs that contributed to the selected color.
I/O graphs by job	Displays I/O and disk activity graphs for a particular interval or interval range.
Wait graphs by interval	This menu allows the user to view one of the wait graphs by interval starting at the interval range selected (or right-clicked on).
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.

Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar/job in the interval details window. This window will also appear by left-clicking on any bar in the graph.

Table of Contents | Previous | Next

2.6.2.4 Objected waited on

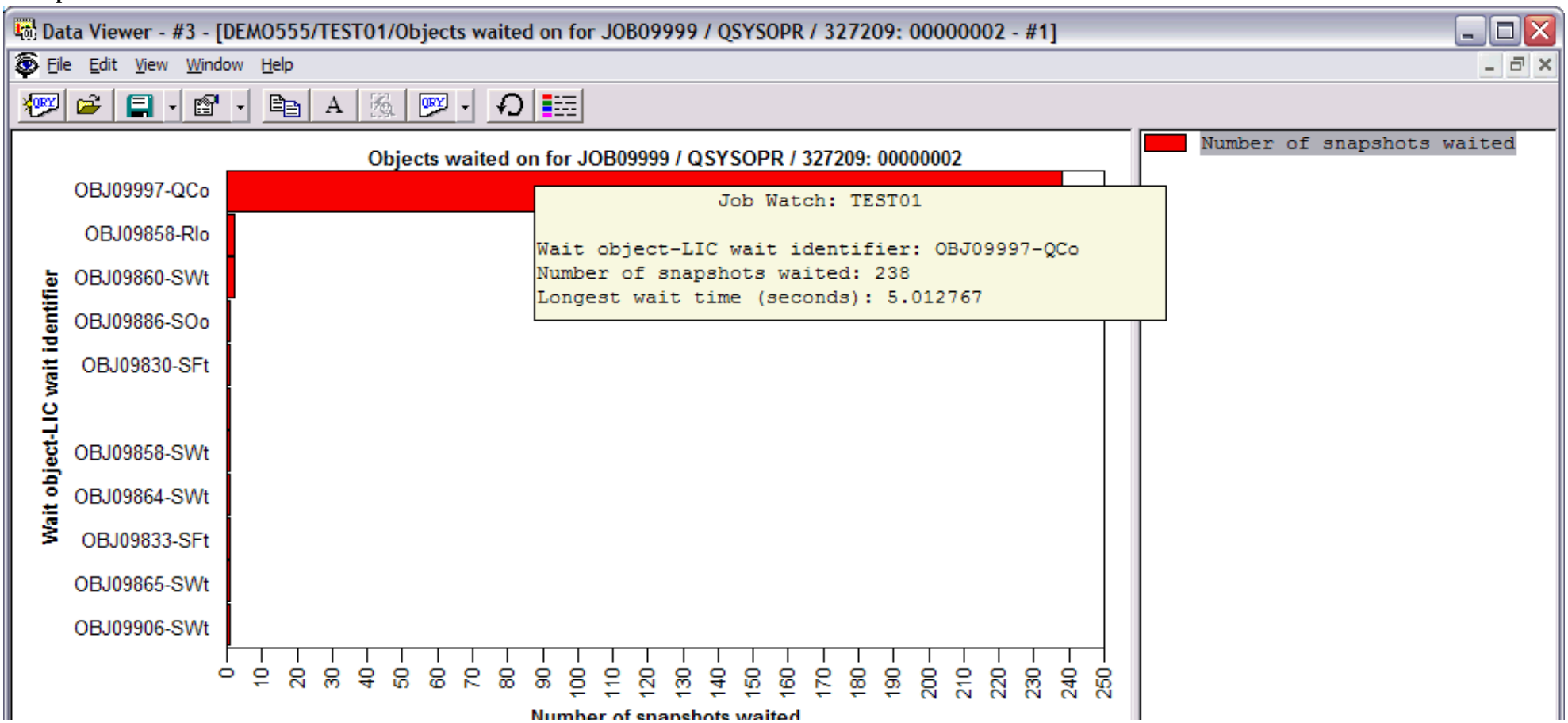
Description: This graph shows the list of objects a job was detected to have waited on throughout its existence during the job watch. The graph shows the number of snapshots taken where the job was waiting to use an object, the name/type information for each object and the type of wait.

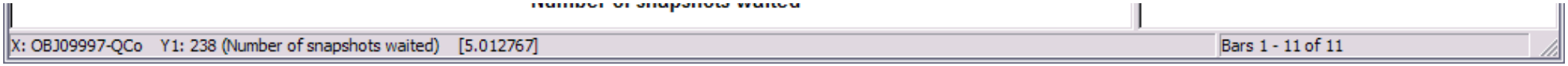
The objects waited on does not necessarily indicate a problem, depending on the type of work the job is performing as well as what is normal for the application. However in other cases especially situations where a program is waiting to use a file, the objected waited on feature can help identify locking conflicts.

Placing the mouse over a bar will also show the longest chunk of time spent waiting on the object during the collection.

Graph Type: summary over a single job (horizontal bar)
X-axis: The name of the object being waited on, and the 3 character LIC wait identifier showing the type of wait that occurred. 1 bar is created per object being waited on for the job.
Y-Axis: The number of snapshots taken where the object waited on was the same.

Example:





Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar in a properties window.

Table of Contents | Previous | Next

2.6.3 Dispatched CPU graphs - detailed

The detailed CPU graphs provide statistics for the CPU and CPU queueing that occurred for a job in a job watch.

This section covers the graphs available of this type.

Table of Contents | Previous | Next

2.6.3.1 Run/wait time signature

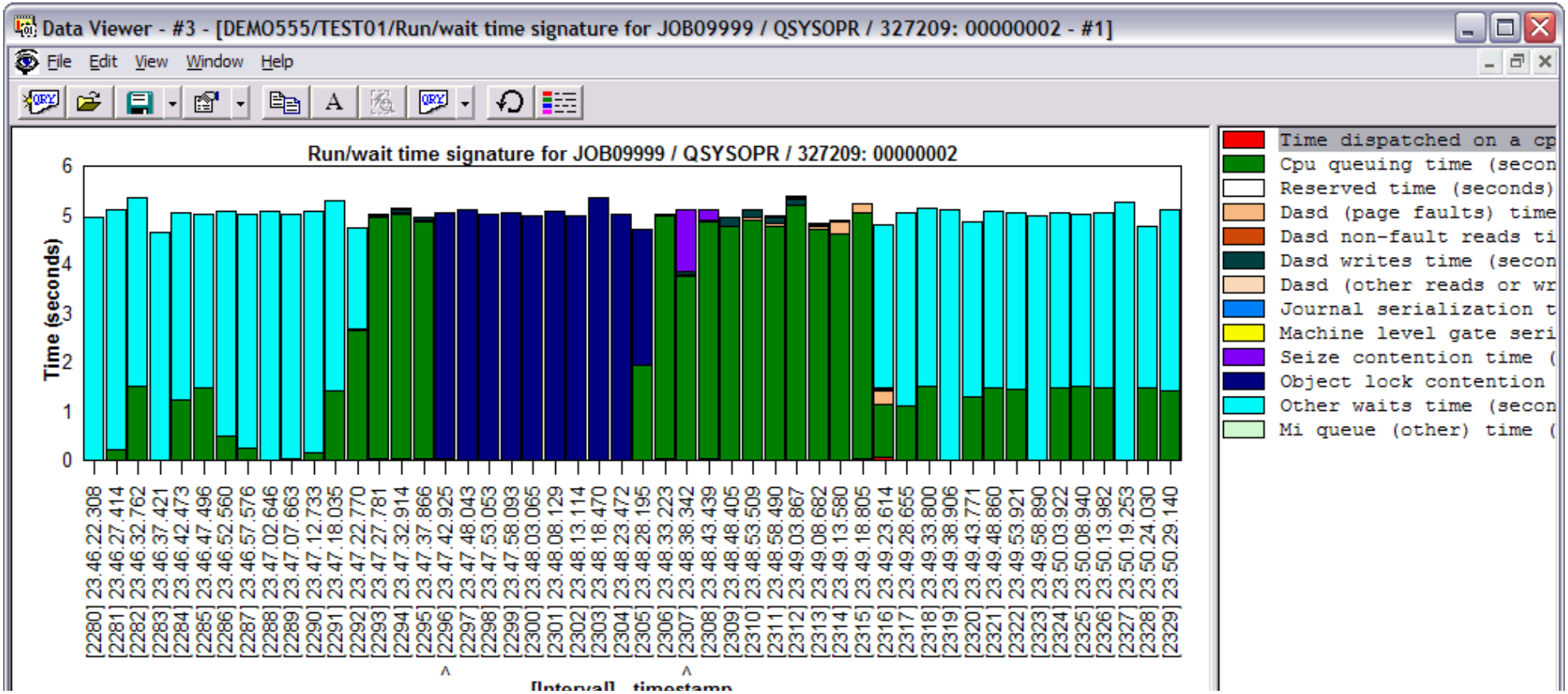
Description: This graph shows a **detailed** look at the time a job spent in various types of waits throughout its existence during the job watch. The graph makes use of the 32 run/wait buckets. To determine which type of wait a color represents, place the mouse pointer of a color/bar of interest and a description of the wait as well as other statistics about the interval/wait will be displayed.

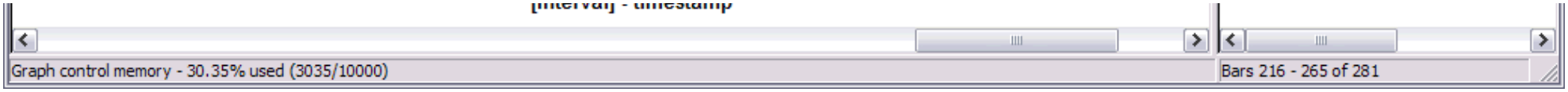
This graph can help pinpoint within a specific job where the job started to perform work and how efficiently it was performing that work. If a interval in the graph suddenly becomes suspicious (high seize time for example) clicking on the bar will provide other valuable information through the interval details window.

The total height of the stacked bar (which includes all waits for that interval and CPU) indicates the elapsed interval time.

Graph Type: detailed for a single job (stacked vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: Each color represents the amount of time the job/thread/task spent in one of the 32 different wait buckets within an interval. Time is listed in seconds.

Example:





Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Holder	Contains menu options for detail reports over the job holding the object being waited on for the selected interval.
Display call stack	Shows the call stack within the interval detail property pages for the interval selected.
Run/wait graphs by job	This menu allows the user to view the jobs for a paticular interval or interval range ranked by the desired wait type. By right-clicking on the graph over a particular color (such as dark blue for object locks) the user can drill into an interval to view the jobs that contributed to the selected color.
I/O graphs by job	Displays I/O and disk activity graphs for a particular interval or interval range.
Wait graphs by interval	This menu allows the user to view one of the wait graphs by interval starting at the interval range selected (or right-clicked on).
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar/job in the interval details window. This window will also appear by left-clicking on any bar in the graph.

Table of Contents | Previous | Next

2.6.3.2 Run/wait time signature with dispatch CPU breakdown

Description: This graph shows a **detailed** look at the time a job spent in various types of waits throughout its existence during the job watch. The graph makes use of the 32 run/wait buckets. To determine which type of wait a color represents, place the mouse pointer of a color/bar of interest and a description of the wait as well as other statistics about the interval/wait will be displayed.

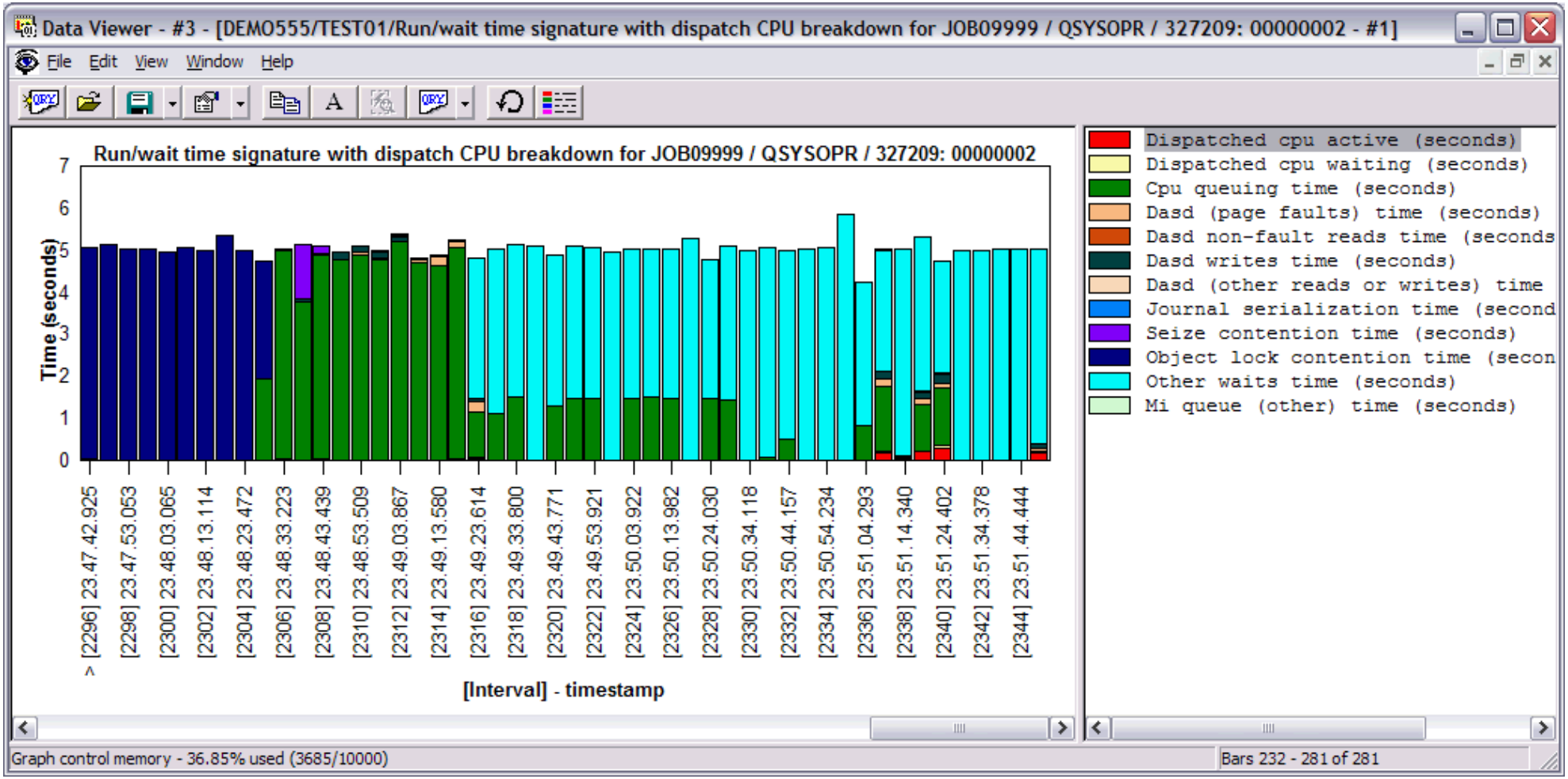
This graph is the same as the Run/wait time signature detail graph except the Dispatch CPU time bucket (QTIME01) is broken down into 2 components: dispatched CPU active and dispatched CPU waiting. Dispatched CPU waiting shows how much time each interval the job was dispatched on the CPU chip but did not actually use the CPU.

This graph can help pinpoint within a specific job where the job started to perform work and how efficiently it was performing that work. If a interval in the graph suddenly becomes suspicious (high seize time for example) clicking on the bar will provide other valuable information through the interval details window.

The total height of the stacked bar (which includes all waits for that interval and CPU) indicates the elapsed interval time.

Graph Type: detailed for a single job (stacked vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: Each color represents the amount of time the job/thread/task spent in one of the 32 different wait buckets within an interval. Time is listed in seconds.

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Holder	Contains menu options for detail reports over the job holding the object being waited on for the selected interval.
Display call stack	Shows the call stack within the interval detail property pages for the interval selected.
Run/wait graphs by job	This menu allows the user to view the jobs for a paticular interval or interval range ranked by the desired wait type. By right-clicking on the graph over a particular color (such as dark blue for object locks) the user can drill into an interval to view the jobs that contributed to the selected color.
I/O graphs by job	Displays I/O and disk activity graphs for a particular interval or interval range.
Wait graphs by interval	This menu allows the user to view one of the wait graphs by interval starting at the interval range selected (or right-clicked on).
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.

Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar/job in the interval details window. This window will also appear by left-clicking on any bar in the graph.

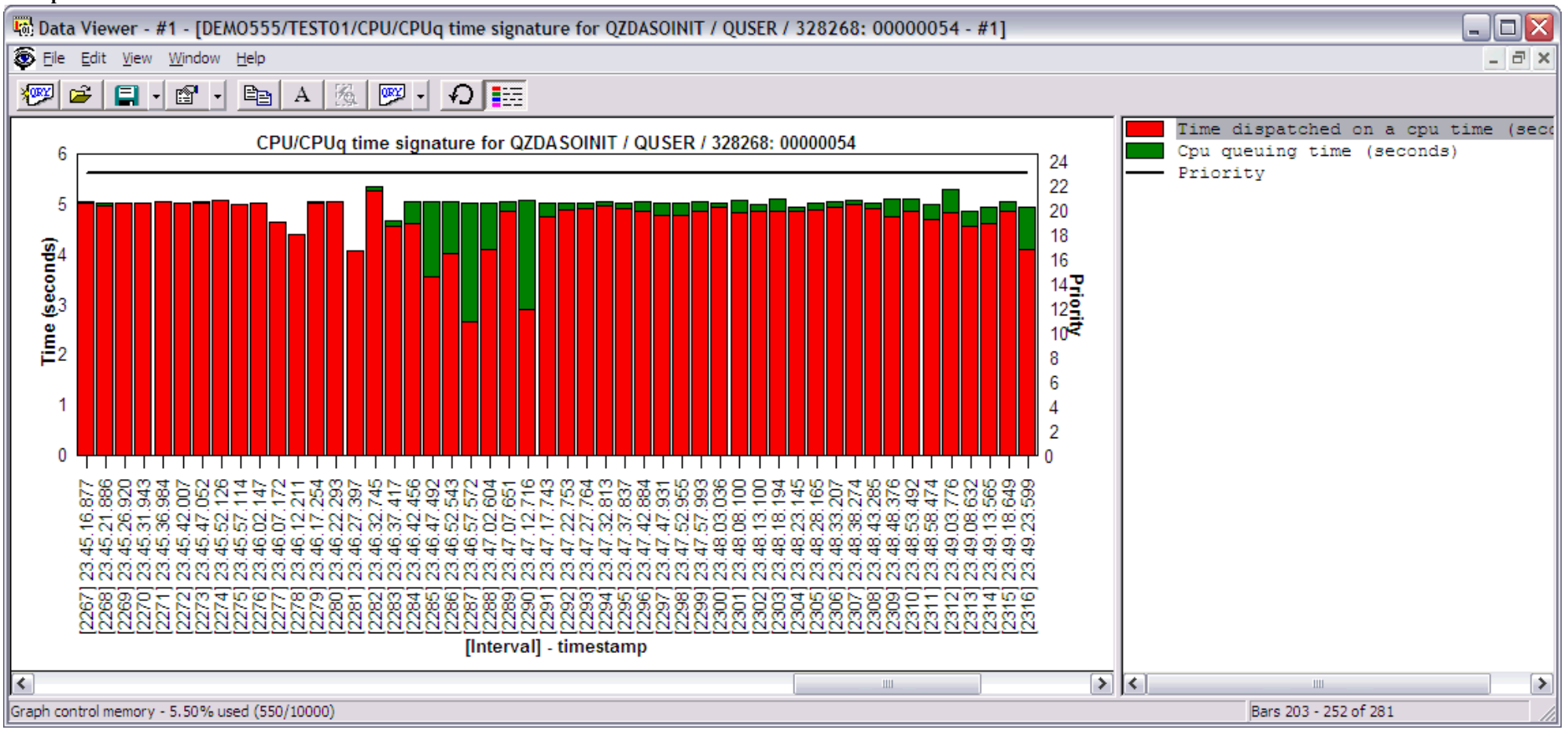
2.6.3.3 CPU/CPUq time signature

Description: This graph shows a **detailed** look at the time the job spent using CPU and waiting for CPU throughout its existence during the job watch. Red colors indicate CPU use, and green colors represent CPU queueing time.

This graph can help locate when high CPU, or CPU queueing is occurring for a specific job.

Graph Type: detailed for a single job (stacked vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: Each color represents the amount of time the job/thread/task spent using CPU or CPU queueing. All times are listed in seconds.
Second Y-Axis: The line shows the job's priority during the collection.

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Holder	Contains menu options for detail reports over the job holding the object being waited on for the selected interval.
Display call stack	Shows the call stack within the interval detail property pages for the interval selected.
Run/wait graphs by job	This menu allows the user to view the jobs for a paticular interval or interval range ranked by the desired wait type. By right-clicking on the graph over a particular color (such as dark blue for object locks) the user can drill into an interval to view the jobs that contributed to the selected color.
I/O graphs by job	Displays I/O and disk activity graphs for a particular interval or interval range.
Wait graphs by interval	This menu allows the user to view one of the wait graphs by interval starting at the interval range selected (or right-clicked on).
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar/job in the interval details window. This window will also appear by left-clicking on any bar in the graph.

Table of Contents | Previous | Next

2.6.4 DASD/IO graphs - detailed

The detailed DASD/IO graphs provide statistics for disk space allocation/deallocations, as well all physical and logical IOs that occurred for a job in a job watch.

This section covers the graphs available of this type.

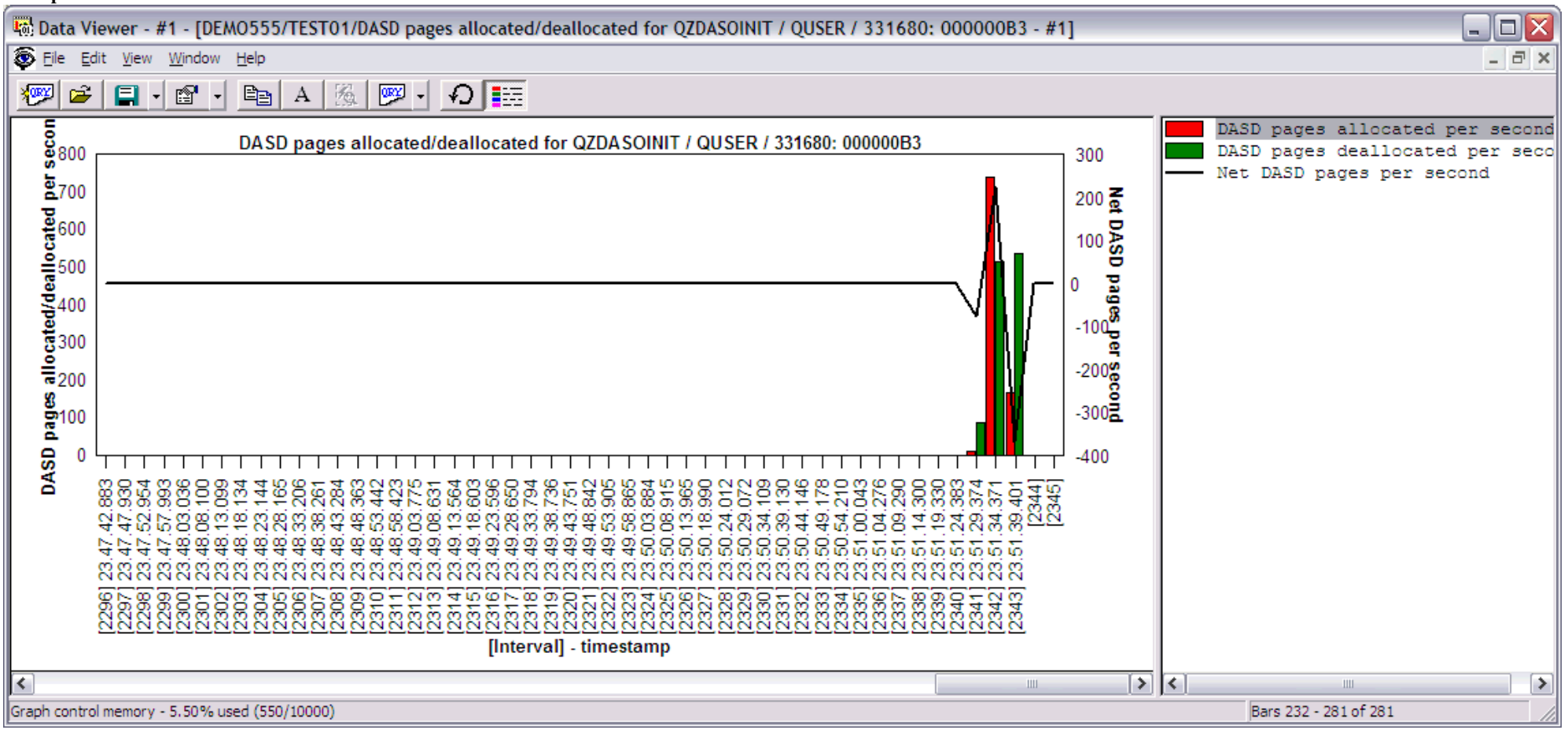
2.6.4.1 DASD pages allocated/deallocated

Description: This graph shows a **detailed** look at the amount of disk space allocated and deallocated by a job throughout its existence during the job watch. Red colors indicate space allocations and green colors indicate space deallocations. Allocations and deallocations listed are the number of 4K dasd pages allocated/deallocated per second. A "DASD page" is a 4k (4096 bytes) block of data.

This graph can help locate where DASD page allocations or deallocations were highest for a specific job.

Graph Type: detailed for a single job (vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: Contains the rate of DASD pages allocations and deallocations for each interval.
Second Y-Axis: This line represents the rate of allocations minus the rate of deallocations per interval (the net dasd pages per second)

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Holder	Contains menu options for detail reports over the job holding the object being waited on for the selected interval.
Display call stack	Shows the call stack within the interval detail property pages for the interval selected.
Run/wait graphs by job	This menu allows the user to view the jobs for a paticular interval or interval range ranked by the desired wait type. By right-clicking on the graph over a particular color (such as dark blue for object locks) the user can drill into an interval to view the jobs that contributed to the selected color.
I/O graphs by job	Displays I/O and disk activity graphs for a particular interval or interval range.
Wait graphs by interval	This menu allows the user to view one of the wait graphs by interval starting at the interval range selected (or right-clicked on).
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar/job in the interval details window. This window will also appear by left-clicking on any bar in the graph.

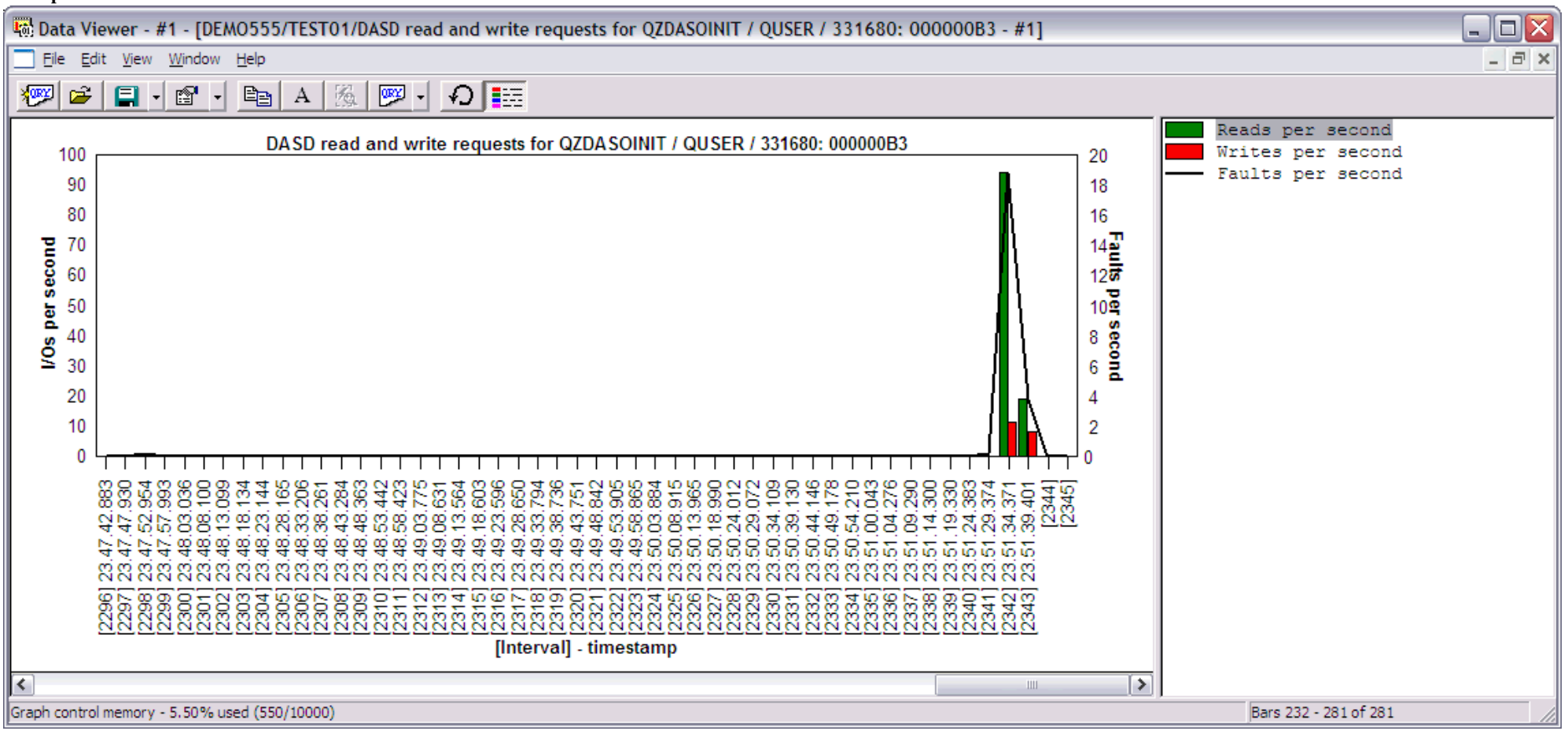
Table of Contents | Previous | Next

2.6.4.2 DASD read and write requests

Description: This graph shows a **detailed** look at the disk reads/writes and page faulting occuring for a job throughout its existence during the job watch. Each color in the graph represents a different type of IO operation.

Graph Type: detailed for a single job (stacked vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: Red colors indicate writes per second and green colors show reads per second.
Second Y-Axis: This black line shows the faults per second over time.

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Holder	Contains menu options for detail reports over the job holding the object being waited on for the selected interval.
Display call stack	Shows the call stack within the interval detail property pages for the interval selected.
Run/wait graphs by job	This menu allows the user to view the jobs for a paticular interval or interval range ranked by the desired wait type. By right-clicking on the graph over a particular color (such as dark blue for object locks) the user can drill into an interval to view the jobs that contributed to the selected color.
I/O graphs by job	Displays I/O and disk activity graphs for a particular interval or interval range.
Wait graphs by interval	This menu allows the user to view one of the wait graphs by interval starting at the interval range selected (or right-clicked on).
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar/job in the interval details window. This window will also appear by left-clicking on any bar in the graph.

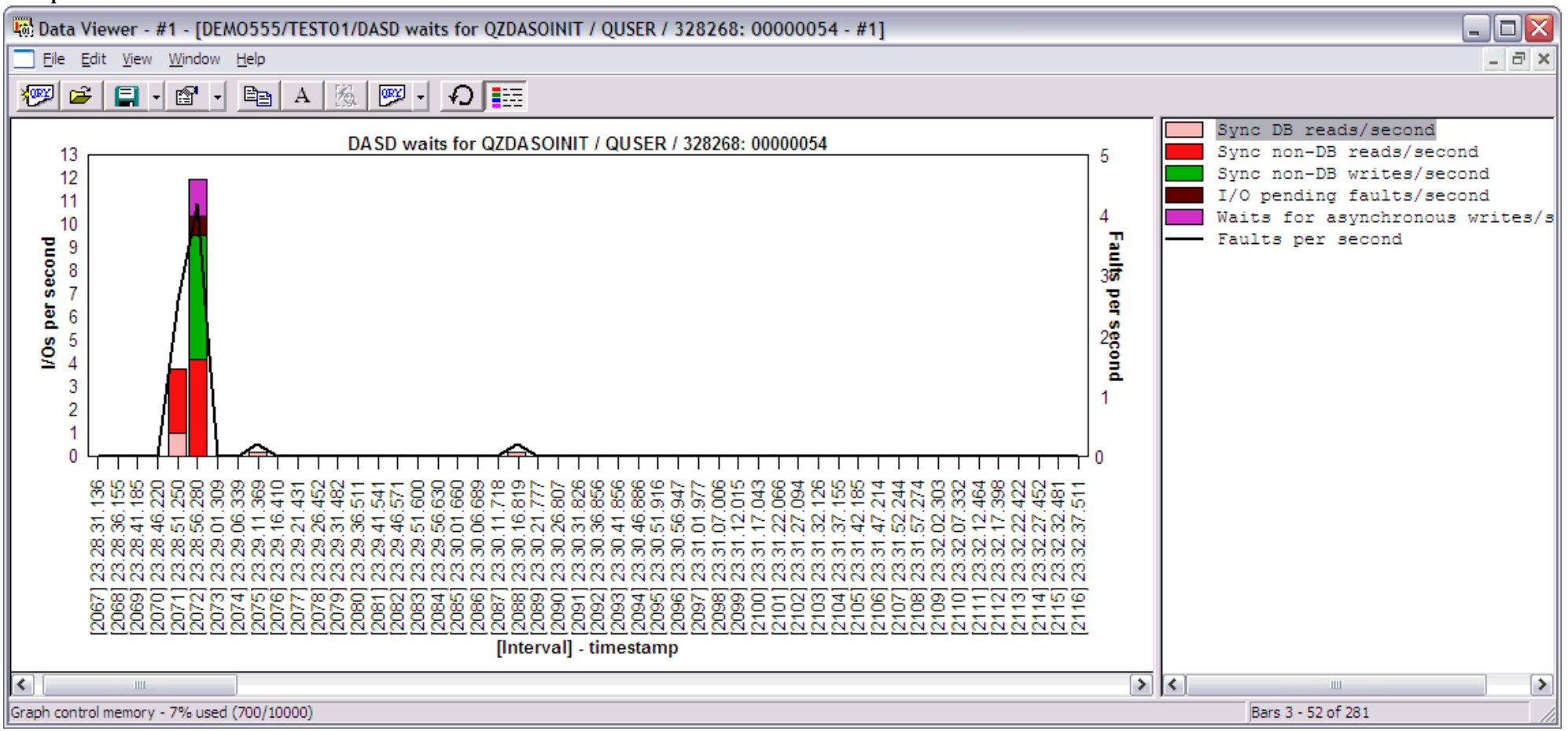
Table of Contents | Previous | Next

2.6.4.3 DASD waits

Description: This graph shows a **detailed** look at the disk IO operations and page faulting occuring by a job throughout its existence during the job watch. Each color in the graph represents a different type of IO operation.

Graph Type: detailed for a single job (stacked vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: Contains rates of various types of disk operations for a specific job.
Second Y-Axis: This black line shows the faults per second over time.

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Holder	Contains menu options for detail reports over the job holding the object being waited on for the selected interval.
Display call stack	Shows the call stack within the interval detail property pages for the interval selected.
Run/wait graphs by job	This menu allows the user to view the jobs for a paticular interval or interval range ranked by the desired wait type. By right-clicking on the graph over a particular color (such as dark blue for object locks) the user can drill into an interval to view the jobs that contributed to the selected color.
I/O graphs by job	Displays I/O and disk activity graphs for a particular interval or interval range.
Wait graphs by interval	This menu allows the user to view one of the wait graphs by interval starting at the interval range selected (or right-clicked on).
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar/job in the interval details window. This window will also appear by left-clicking on any bar in the graph.

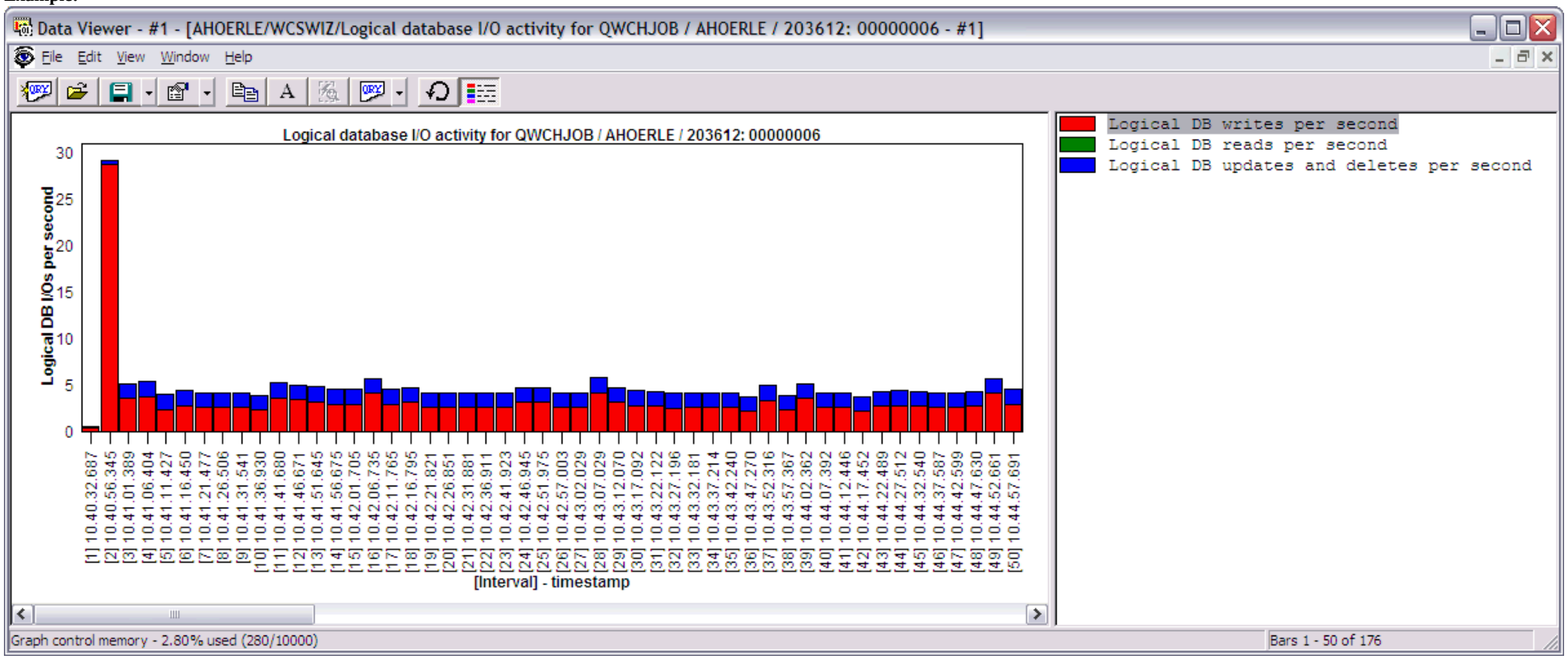
2.6.4.4 Logical database I/O activity

Description: This graph shows a **detailed** look at the rates of logical database I/O activity including writes, reads and updates/deletes for a job in the job watch.

This graph may be used to locate where LDIO activity is occurring within a specific job.

Graph Type: detailed for a single job (stacked vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: Contains the rate of logical database I/Os for writes, reads and updates/deletes.

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Holder	Contains menu options for detail reports over the job holding the object being waited on for the selected interval.
Display call stack	Shows the call stack within the interval detail property pages for the interval selected.

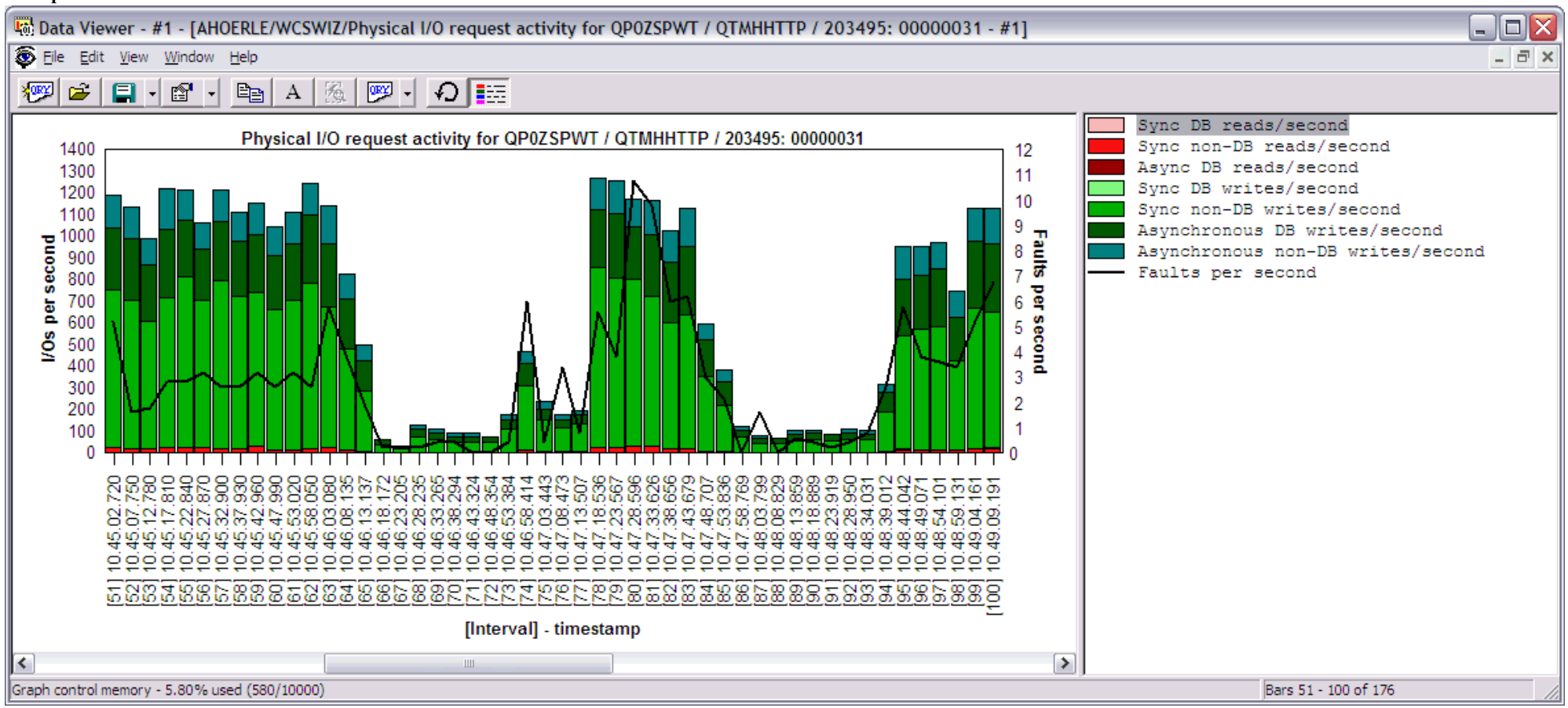
Run/wait graphs by job	This menu allows the user to view the jobs for a paticular interval or interval range ranked by the desired wait type. By right-clicking on the graph over a particular color (such as dark blue for object locks) the user can drill into an interval to view the jobs that contributed to the selected color.
I/O graphs by job	Displays I/O and disk activity graphs for a particular interval or interval range.
Wait graphs by interval	This menu allows the user to view one of the wait graphs by interval starting at the interval range selected (or right-clicked on).
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar/job in the interval details window. This window will also appear by left-clicking on any bar in the graph.

2.6.4.5 Physical I/O request activity

Description: This graph shows a **detailed** look at the physical I/Os that were made by a job throughout its existence during the job watch. The graph provides rates of database and non-database reads and writes. Page fault rates are listed on the secondary Y-axis.

Graph Type: detailed for a single job (stacked vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: Contains the rate of async and synchronous DB and non-DB reads and writes.
Second Y-Axis: The black line represents the page fault rate for the job over time.

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Holder	Contains menu options for detail reports over the job holding the object being waited on for the selected interval.

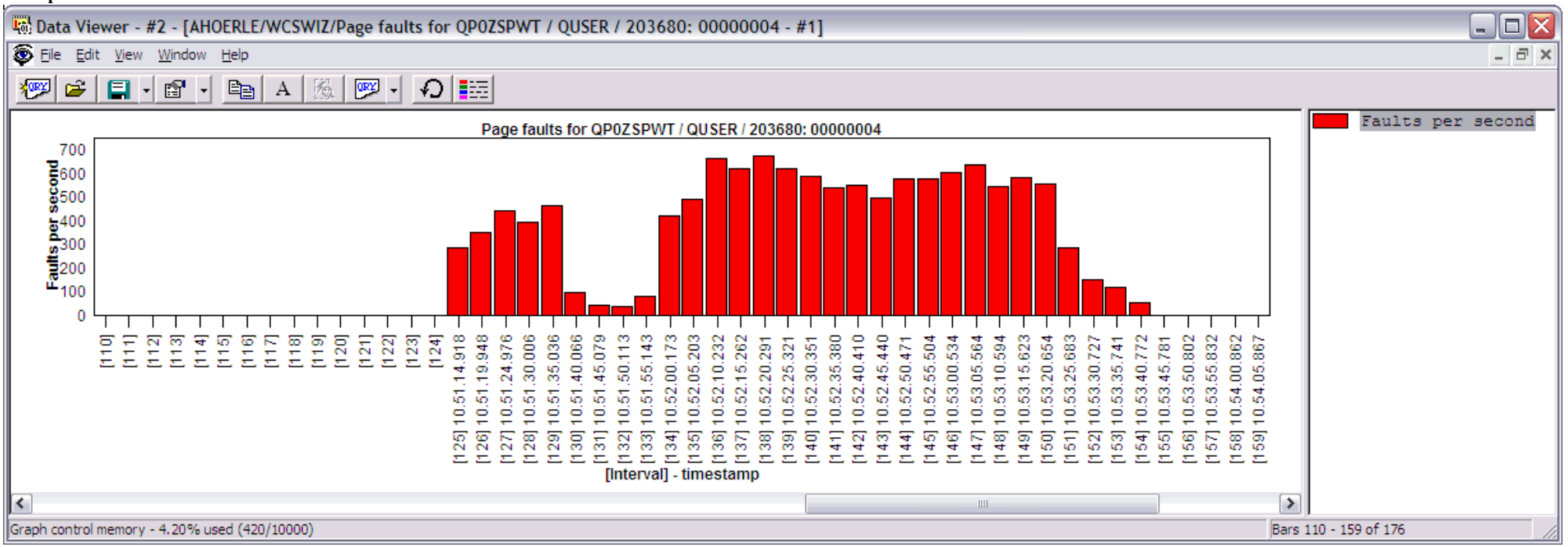
Display call stack	Shows the call stack within the interval detail property pages for the interval selected.
Run/wait graphs by job	This menu allows the user to view the jobs for a paticular interval or interval range ranked by the desired wait type. By right-clicking on the graph over a particular color (such as dark blue for object locks) the user can drill into an interval to view the jobs that contributed to the selected color.
I/O graphs by job	Displays I/O and disk activity graphs for a particular interval or interval range.
Wait graphs by interval	This menu allows the user to view one of the wait graphs by interval starting at the interval range selected (or right-clicked on).
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar/job in the interval details window. This window will also appear by left-clicking on any bar in the graph.

2.6.4.6 Page faults

Description: This graph shows a **detailed** look at the page fault rate per second for a job in a job watch.

Graph Type: detailed for a single job (vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: Each bar displays the page fault rate for the job each interval.

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Holder	Contains menu options for detail reports over the job holding the object being waited on for the selected interval.
Display call stack	Shows the call stack within the interval detail property pages for the interval selected.
Run/wait graphs by job	This menu allows the user to view the jobs for a paticular interval or interval range ranked by the desired wait type. By right-clicking on the graph over a particular color (such as dark blue for object locks) the user can drill into an interval to view the jobs that contributed to the selected color.
I/O graphs by job	Displays I/O and disk activity graphs for a particular interval or interval range.
Wait graphs by interval	This menu allows the user to view one of the wait graphs by interval starting at the interval range selected (or right-clicked on).
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.

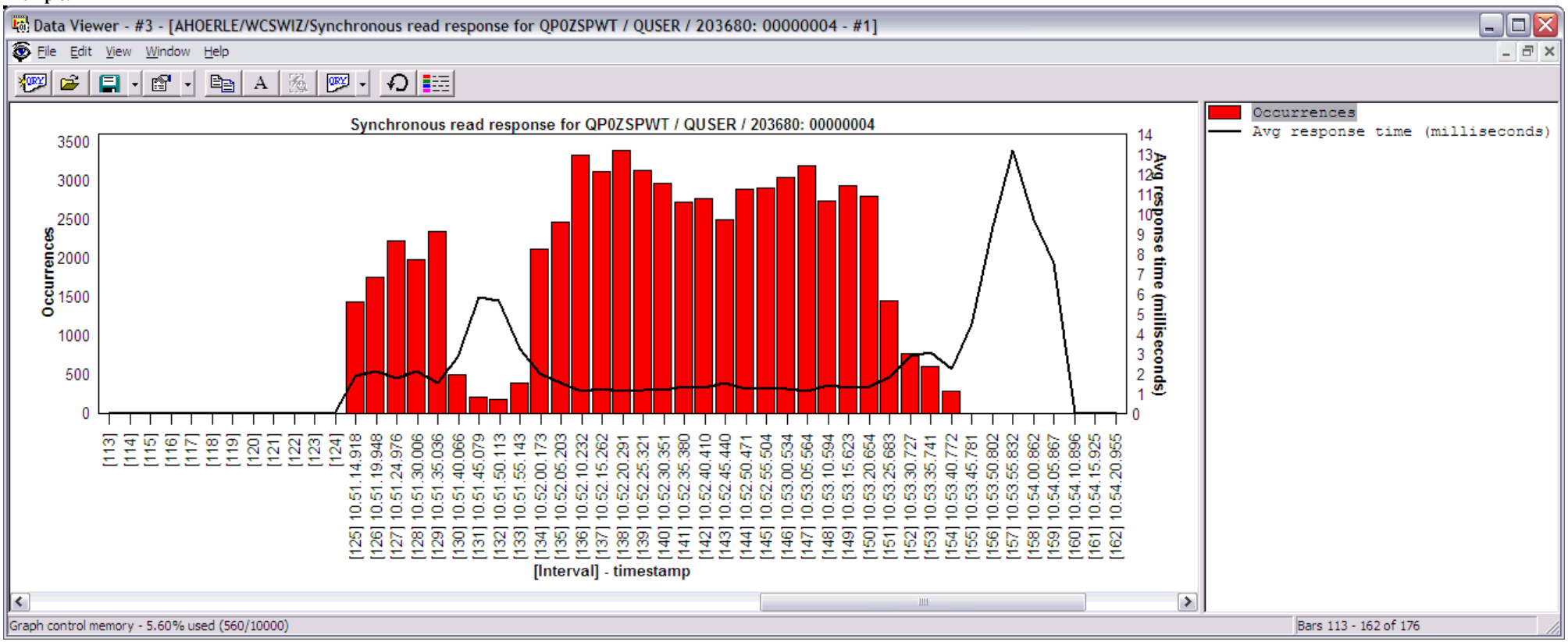
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar/job in the interval details window. This window will also appear by left-clicking on any bar in the graph.

2.6.4.7 Synchronous read response time

Description: This graph shows a **detailed** look at the total synchronous reads and average synchronous read response times for a job in a job watch.

Graph Type: detailed for a single job (vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: Each bar displays the total synchronous reads each interval.
Second Y-Axis: This line shows the average synchronous read response time (in milliseconds).

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Holder	Contains menu options for detail reports over the job holding the object being waited on for the selected interval.
Display call stack	Shows the call stack within the interval detail property pages for the interval selected.
Run/wait graphs by job	This menu allows the user to view the jobs for a paticular interval or interval range ranked by the desired wait type. By right-clicking on the graph over a particular color (such as dark blue for object locks) the user can drill into an interval to view the jobs that contributed to the selected color.
I/O graphs by job	Displays I/O and disk activity graphs for a particular interval or interval range.
Wait graphs by interval	This menu allows the user to view one of the wait graphs by interval starting at the interval range selected (or right-clicked on).

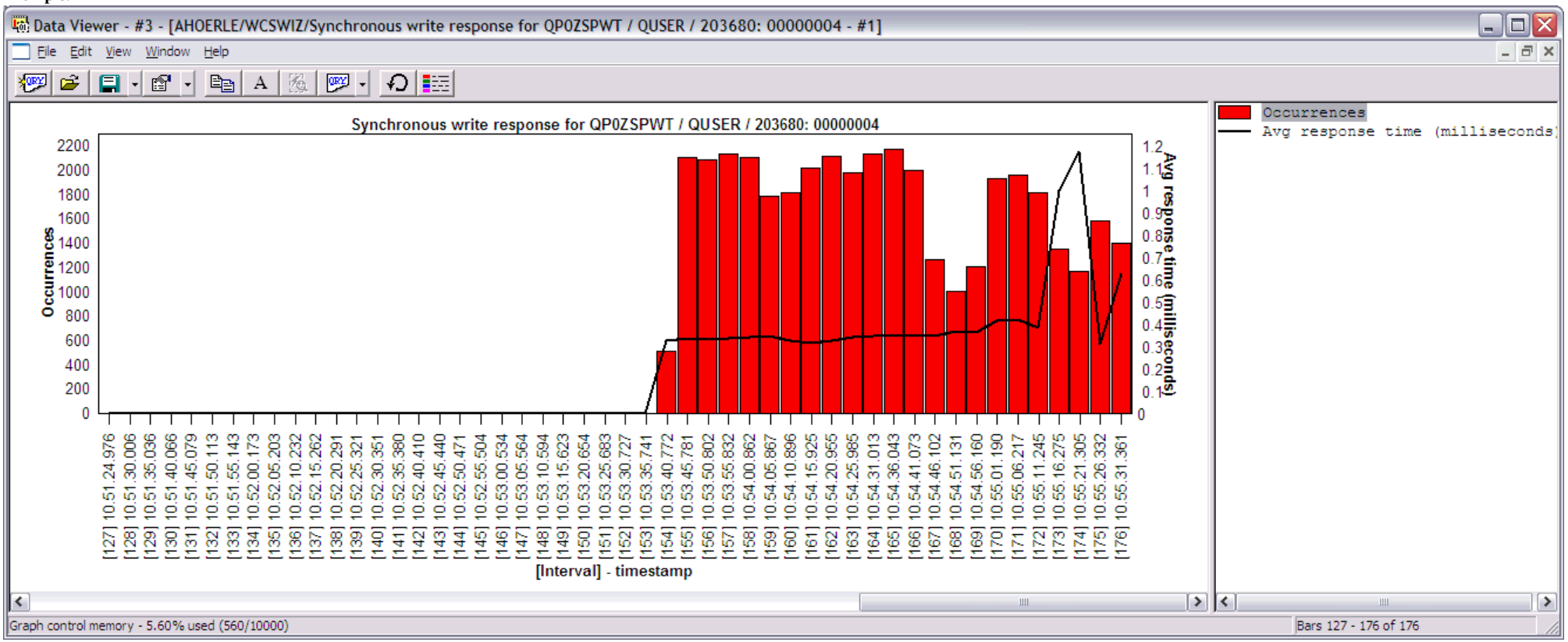
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar/job in the interval details window. This window will also appear by left-clicking on any bar in the graph.

2.6.4.8 Synchronous write response time

Description: This graph shows a **detailed** look at the total synchronous writes and average synchronous write response times for a job in a job watch.

Graph Type: detailed for a single job (vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: Each bar displays the total synchronous writes each interval.
Second Y-Axis: This line shows the average synchronous write response time (in milliseconds).

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Holder	Contains menu options for detail reports over the job holding the object being waited on for the selected interval.
Display call stack	Shows the call stack within the interval detail property pages for the interval selected.
Run/wait graphs by job	This menu allows the user to view the jobs for a paticular interval or interval range ranked by the desired wait type. By right-clicking on the graph over a particular color (such as dark blue for object locks) the user can drill into an interval to view the jobs that contributed to the selected color.
I/O graphs by job	Displays I/O and disk activity graphs for a particular interval or interval range.
Wait graphs by interval	This menu allows the user to view one of the wait graphs by interval starting at the interval range selected (or right-clicked on).

2.6.4.8 Synchronous write response time

Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar/job in the interval details window. This window will also appear by left-clicking on any bar in the graph.

Table of Contents | Previous | Next

2.6.5 IFS graphs - detailed

The detailed IFS graphs provide IFS statistics for a job in a job watch.

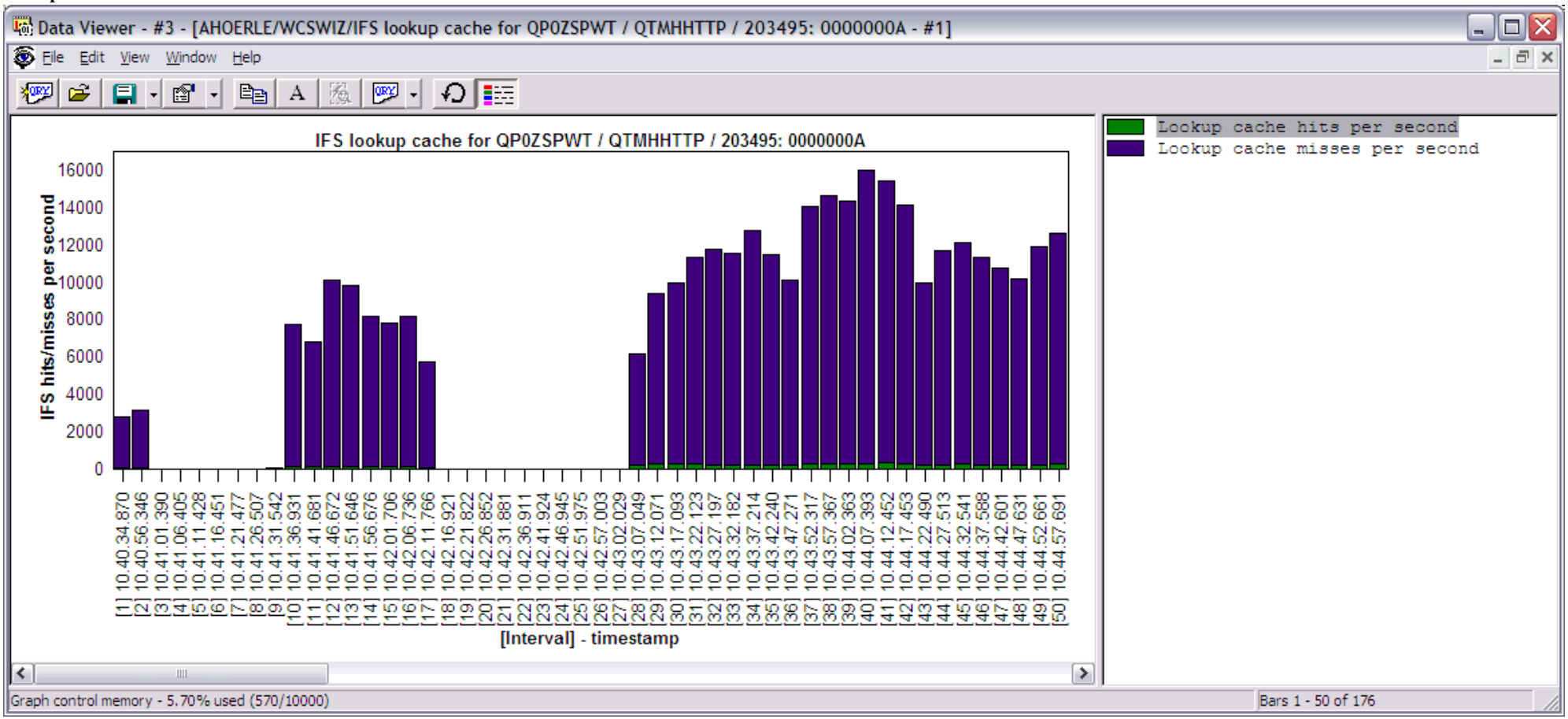
This section covers the graphs available of this type.

2.6.5.1 IFS lookup cache

Description: This graph shows a **detailed** look at the rate of hits/misses for the IFS lookup cache by a job throughout its existence during the job watch. Lookup cache hit rate is indicated in green and misses are shown in blue.

Graph Type: detailed for a single job (stacked vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: Contains the rate of lookup cache hits and misses.

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Holder	Contains menu options for detail reports over the job holding the object being waited on for the selected interval.

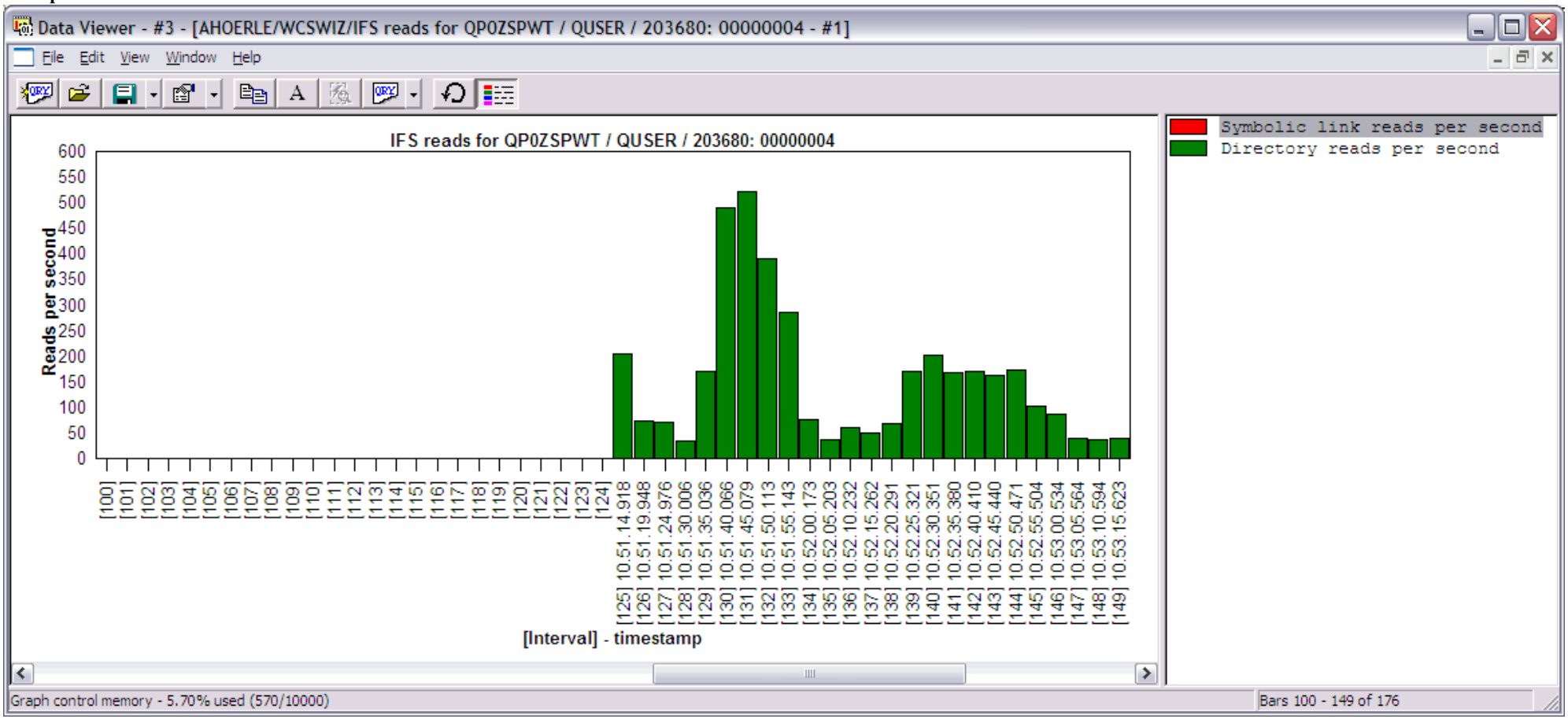
Display call stack	Shows the call stack within the interval detail property pages for the interval selected.
Run/wait graphs by job	This menu allows the user to view the jobs for a paticular interval or interval range ranked by the desired wait type. By right-clicking on the graph over a particular color (such as dark blue for object locks) the user can drill into an interval to view the jobs that contributed to the selected color.
I/O graphs by job	Displays I/O and disk activity graphs for a particular interval or interval range.
Wait graphs by interval	This menu allows the user to view one of the wait graphs by interval starting at the interval range selected (or right-clicked on).
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar/job in the interval details window. This window will also appear by left-clicking on any bar in the graph.

2.6.5.2 IFS reads

Description: This graph shows a **detailed** look at the rate of IFS directory reads and symbolic link reads for a job throughout its existence during the job watch.

Graph Type: detailed for a single job (stacked vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: Contains the rate of symbolic link reads per second in red and directory reads per second in green.

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Holder	Contains menu options for detail reports over the job holding the object being waited on for the selected interval.

Display call stack	Shows the call stack within the interval detail property pages for the interval selected.
Run/wait graphs by job	This menu allows the user to view the jobs for a paticular interval or interval range ranked by the desired wait type. By right-clicking on the graph over a particular color (such as dark blue for object locks) the user can drill into an interval to view the jobs that contributed to the selected color.
I/O graphs by job	Displays I/O and disk activity graphs for a particular interval or interval range.
Wait graphs by interval	This menu allows the user to view one of the wait graphs by interval starting at the interval range selected (or right-clicked on).
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar/job in the interval details window. This window will also appear by left-clicking on any bar in the graph.

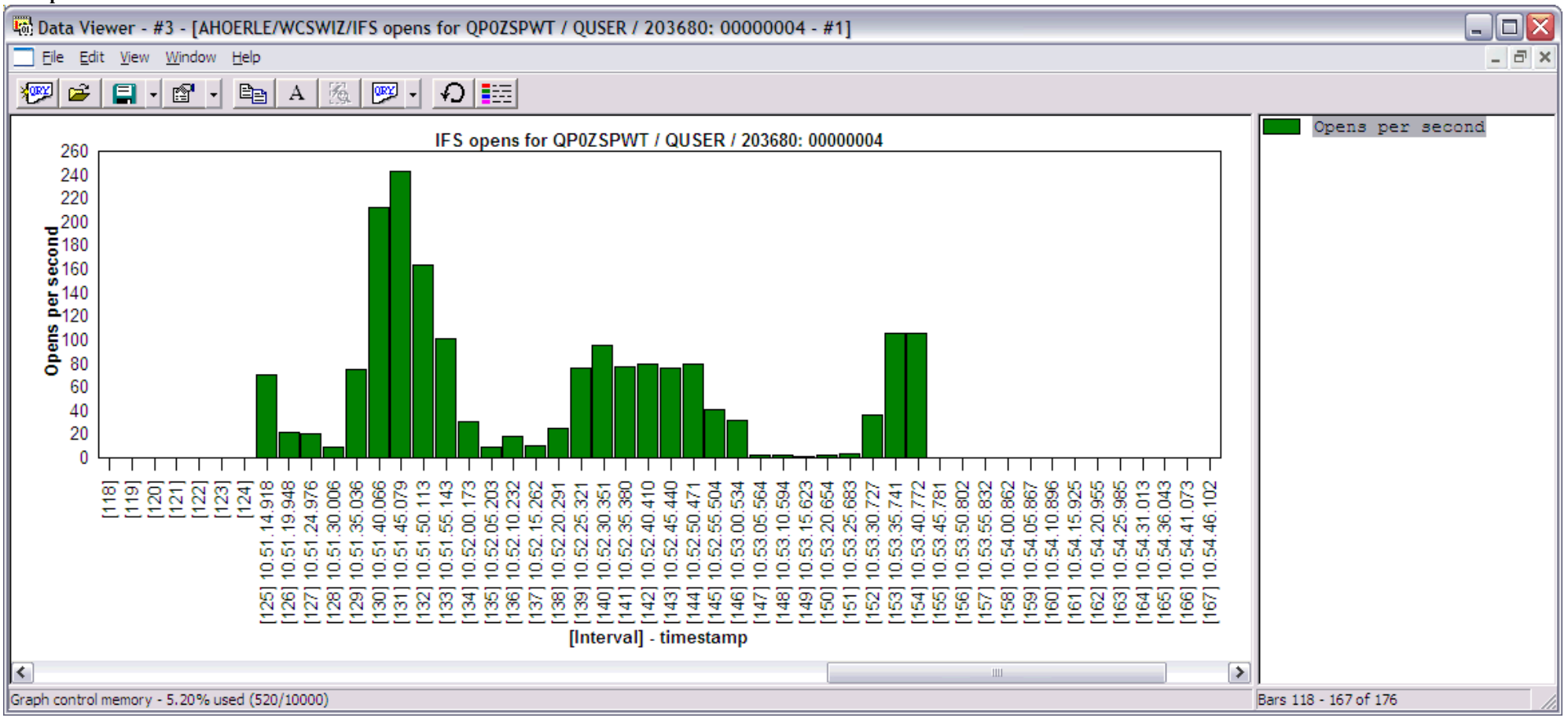
Table of Contents | Previous | Next

2.6.5.3 IFS opens

Description: This graph shows a **detailed** look at the rate of IFS file opens by a job throughout its existence during the job watch.

Graph Type: detailed for a single job (vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: Contains the number of IFS file opens occuring for each interval per second.

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Holder	Contains menu options for detail reports over the job holding the object being waited on for the selected interval.

Display call stack	Shows the call stack within the interval detail property pages for the interval selected.
Run/wait graphs by job	This menu allows the user to view the jobs for a paticular interval or interval range ranked by the desired wait type. By right-clicking on the graph over a particular color (such as dark blue for object locks) the user can drill into an interval to view the jobs that contributed to the selected color.
I/O graphs by job	Displays I/O and disk activity graphs for a particular interval or interval range.
Wait graphs by interval	This menu allows the user to view one of the wait graphs by interval starting at the interval range selected (or right-clicked on).
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar/job in the interval details window. This window will also appear by left-clicking on any bar in the graph.

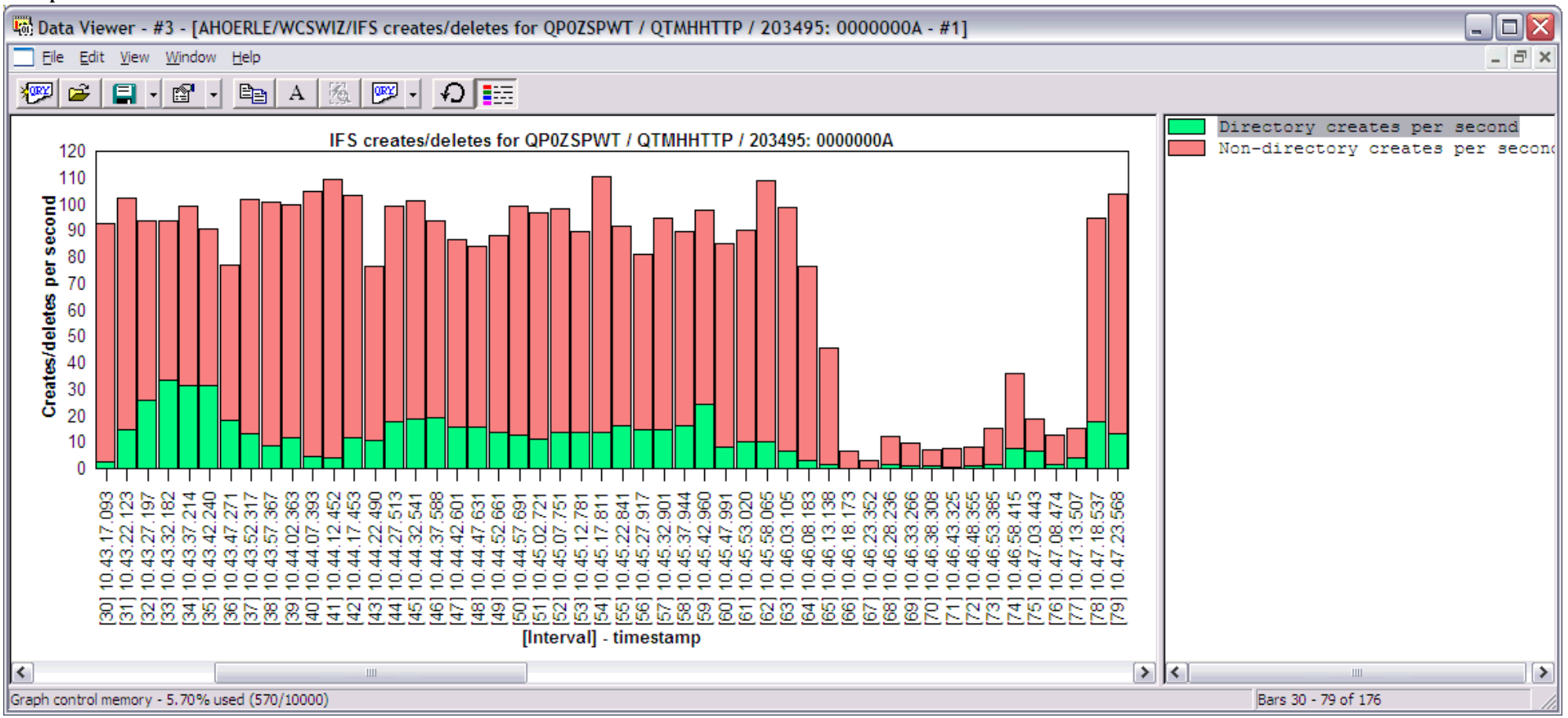
Table of Contents | Previous | Next

2.6.5.4 IFS creates/deletes

Description: This graph shows a **detailed** look at the rate of hits/misses for the IFS creates and deletes performed by a job throughout its existence during the job watch. The creates and deletes are categorized into two types: directory and non-directory.

Graph Type: detailed for a single job (stacked vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: Contains the rates of IFS directory and non-directory creates and deletes per second. Green colors represent directory operations and red colors represent non-directory operations.

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Holder	Contains menu options for detail reports over the job holding the object being waited on for the selected interval.

Display call stack	Shows the call stack within the interval detail property pages for the interval selected.
Run/wait graphs by job	This menu allows the user to view the jobs for a paticular interval or interval range ranked by the desired wait type. By right-clicking on the graph over a particular color (such as dark blue for object locks) the user can drill into an interval to view the jobs that contributed to the selected color.
I/O graphs by job	Displays I/O and disk activity graphs for a particular interval or interval range.
Wait graphs by interval	This menu allows the user to view one of the wait graphs by interval starting at the interval range selected (or right-clicked on).
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar/job in the interval details window. This window will also appear by left-clicking on any bar in the graph.

Table of Contents | Previous | Next

2.6.6 Other graphs - detailed

The detailed "other" graphs provide graphs that were not covered in the other categories for a job in a job watch.

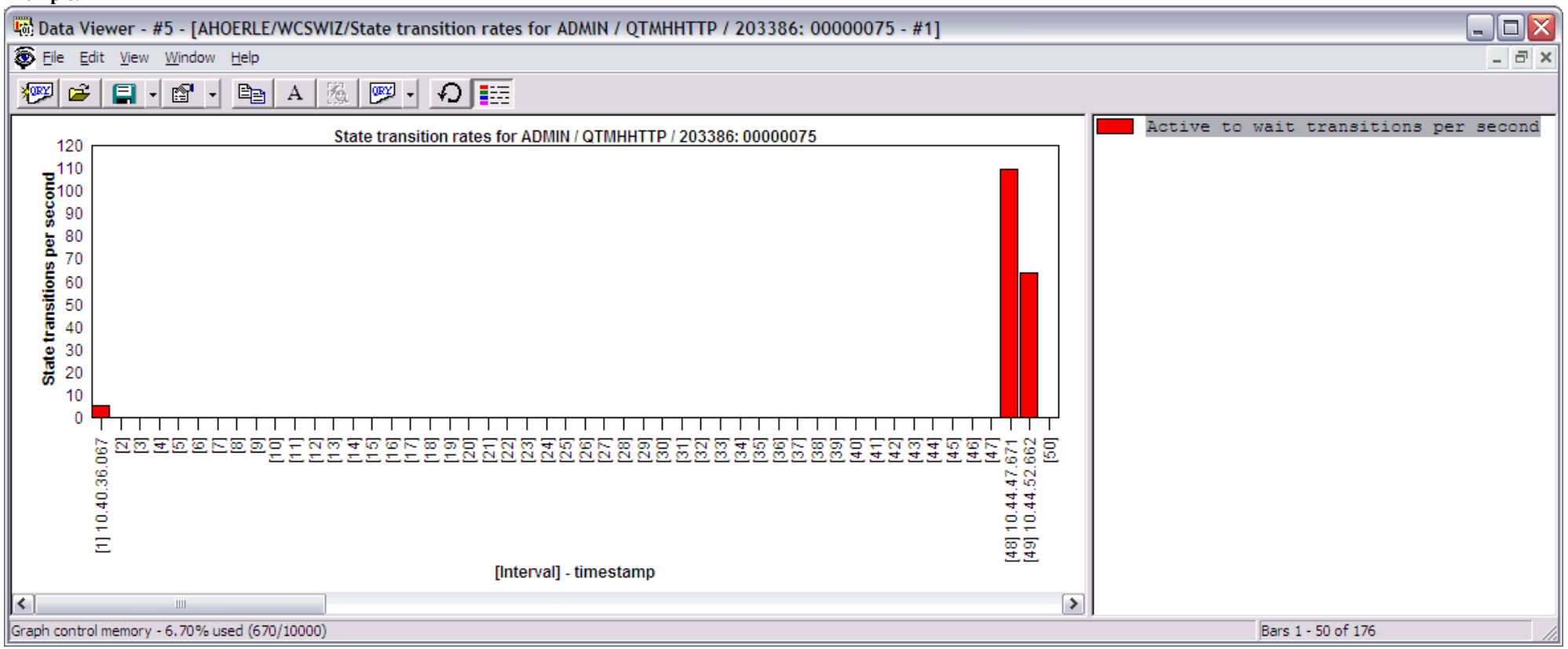
This section covers the graphs available of this type.

2.6.6.1 State transitions

Description: This graph shows a **detailed** look at the job state transitions occurring for a job in the job watch.

Graph Type: detailed for a single job (stacked vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: Each bar shows the rates of the following types of job state transitions: active to wait, wait to ineligible, and active to ineligible.

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Holder	Contains menu options for detail reports over the job holding the object being waited on for the selected interval.
Display call stack	Shows the call stack within the interval detail property pages for the interval selected.
Run/wait graphs by job	This menu allows the user to view the jobs for a paticular interval or interval range ranked by the desired wait type. By right-clicking on the graph over a particular color (such as dark blue for object locks) the user can drill into an interval to view the jobs that contributed to the selected color.

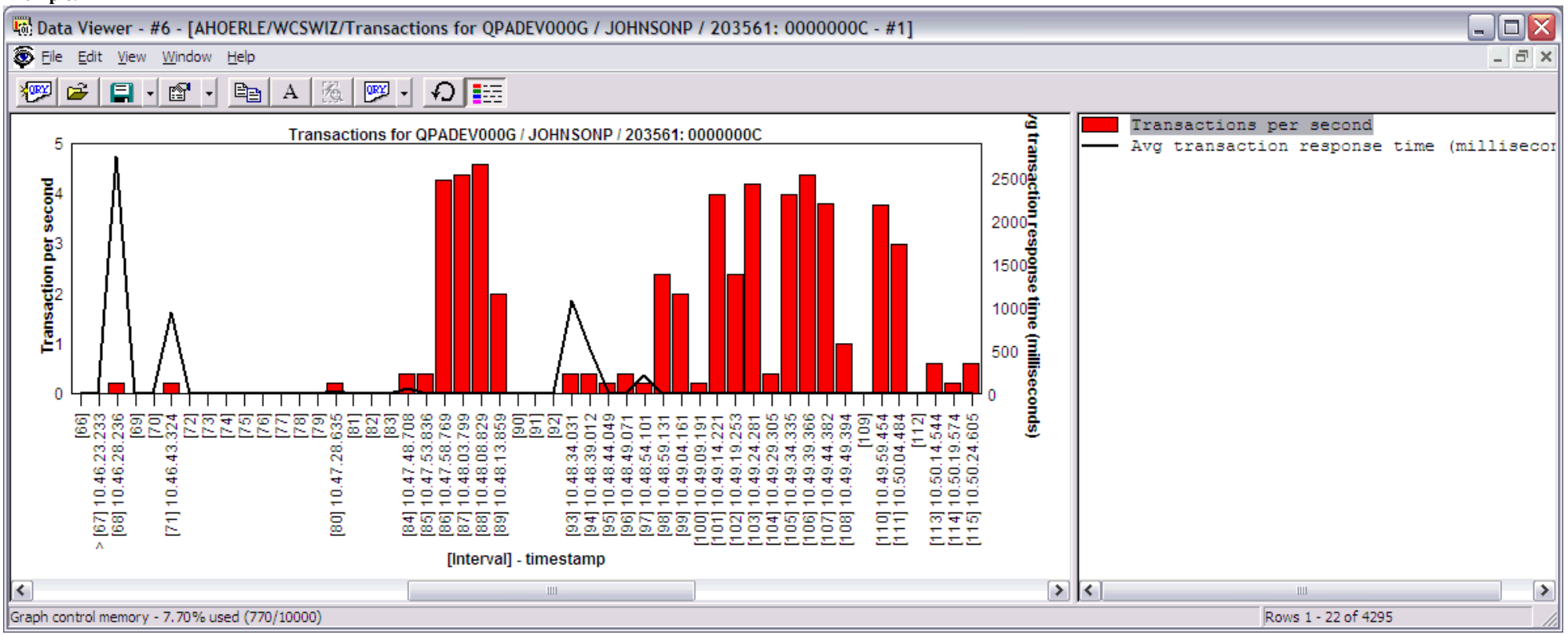
I/O graphs by job	Displays I/O and disk activity graphs for a particular interval or interval range.
Wait graphs by interval	This menu allows the user to view one of the wait graphs by interval starting at the interval range selected (or right-clicked on).
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar/job in the interval details window. This window will also appear by left-clicking on any bar in the graph.

2.6.6.2 Transactions

Description: This graph shows a **detailed** look at the transactions occurring within a job in the job watch.

Graph Type: detailed for a single job (vertical bar)
X-axis: Interval number and time of day. The time of day is listed as hh:mm:ss:xxx
Y-Axis: Each bar shows the transactions occurring per second within each interval.
Second Y-Axis: This axis shows the average transaction response time for each interval (in milliseconds).

Example:



Right-clicking on a bar in this graph provides the following menu options:

Menu	Description
Holder	Contains menu options for detail reports over the job holding the object being waited on for the selected interval.
Display call stack	Shows the call stack within the interval detail property pages for the interval selected.
Run/wait graphs by job	This menu allows the user to view the jobs for a paticular interval or interval range ranked by the desired wait type. By right-clicking on the graph over a particular color (such as dark blue for object locks) the user can drill into an interval to view the jobs that contributed to the selected color.

I/O graphs by job	Displays I/O and disk activity graphs for a particular interval or interval range.
Wait graphs by interval	This menu allows the user to view one of the wait graphs by interval starting at the interval range selected (or right-clicked on).
Copy	Makes a copy of the graph view to the clipboard as a bitmap image. This image can be pasted into applications that accept images from the clipboard.
Save As...	Provides the option to create a .JPG image from the current graph view.
Preferences...	Displays the Preferences interface.
Graph definition...	Displays an interface allowing customization of the graph definition used to build the graph.
Query definition...	Displays an interface allowing customization of the query definition used to build the graph.
Properties	Displays details about the selected bar/job in the interval details window. This window will also appear by left-clicking on any bar in the graph.

Table of Contents | Previous | Next

2.6.7 Detail reports

Table of Contents Previous Next

Chapter 4 PEX Analyzer

This chapter provides an overview of the interfaces within the iDoctor for iSeries PEX Analyzer component.

Table of Contents | Previous | Next

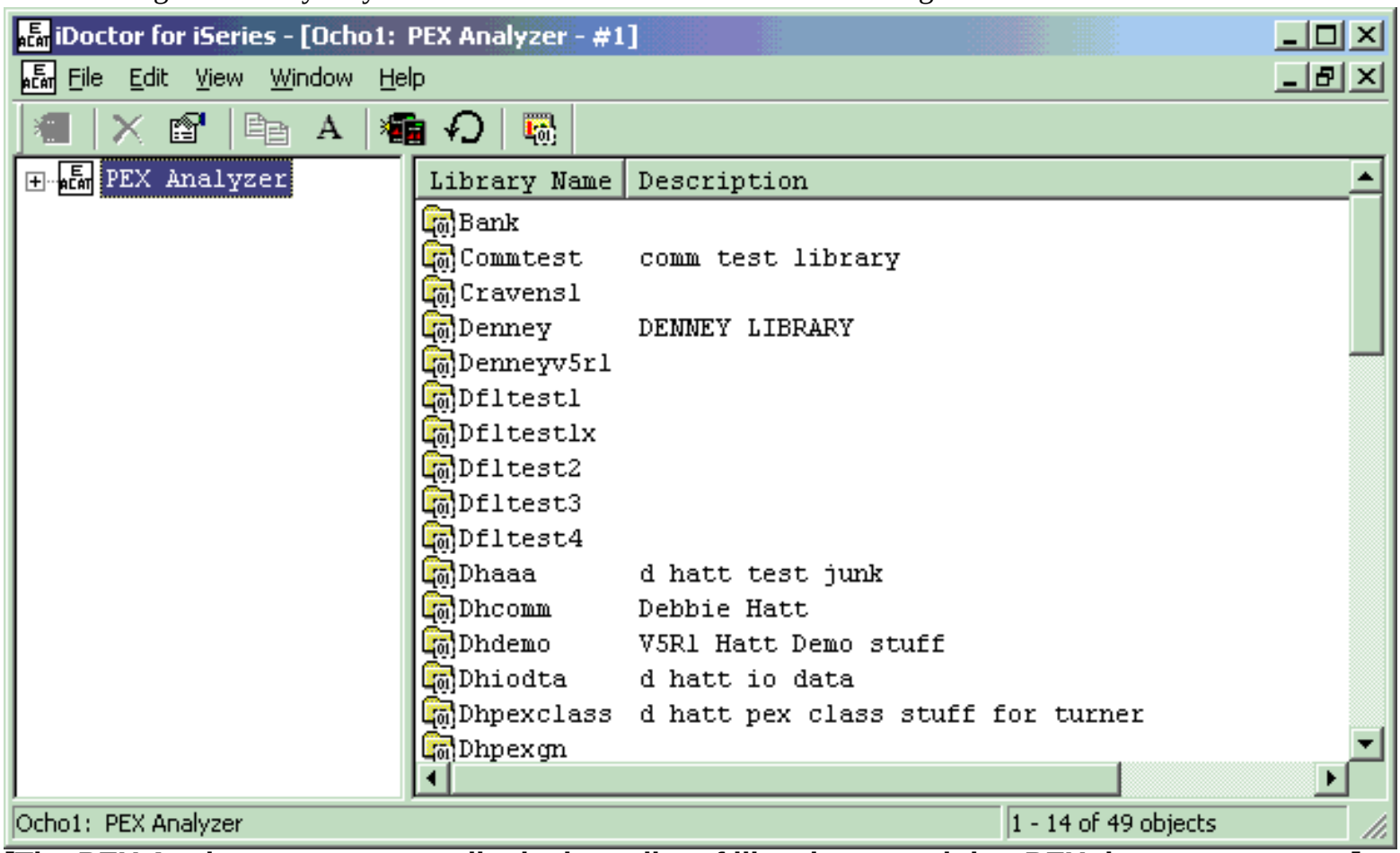
4.1 PEX Analyzer Basics

The PEX Analyzer tool provides a number of interfaces that allows a user to work with PEX collections, and also analyze them.

Starting PEX Analyzer

PEX Analyzer is a component of the iDoctor for iSeries suite of tools. iDoctor for iSeries can be started using the Start menu: Start->Programs->iDoctor for iSeries. Once the iDoctor for iSeries application appears, the PEX Analyzer component is started from the Connection List View by right-clicking on a system name and choosing the PEX Analyzer menu.

After starting PEX Analyzer you will see a window similar to the following:



[The PEX Analyzer component displaying a list of libraries containing PEX data on a system.]

The 'PEX Analyzer' folder contains a list of library folders, each representing a library on the iSeries that contains PEX database files (collections). The list displays each library's name and description. To be more specific a library is displayed in this list if the QAYPERUNI file was found in the library or if the library contains one or more collections currently in progress that have not produced their output files yet.

PEX Analyzer Objects

There are four main types of objects within the tree/list views of PEX Analyzer in the following order: **Libraries**, **collections**, **analyses**, and **reports**. Each of these will covered in more detail in the next sections.

PEX Analyzer Menu Options

The following menu options are available by right-clicking on the 'PEX Analyzer' folder in the tree/list view above.

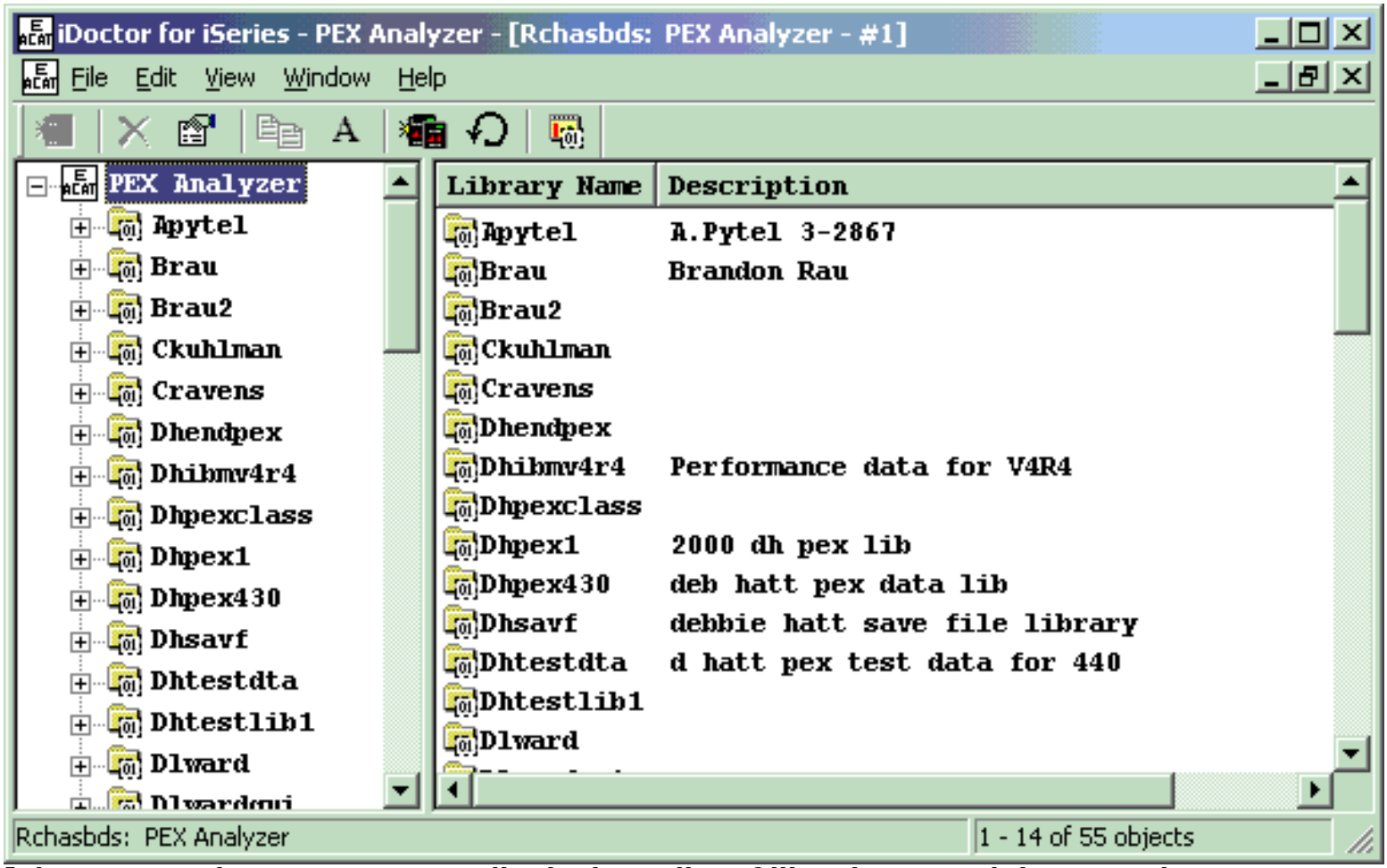
Menu Item	Description
Explore	Displays the contents of the PEX Analyzer folder (a list of libraries on the system containing PEX data) in the right pane of the tree/list window.
Create PEX Definition...	Displays the PEX Definition Wizard which lets you create a PEX definition on your system. This is an interface over the ADDPEXDFN command.
Work with PEX Definitions	Displays the PEX Definitions View which is a list of all the PEX definitions on your system. You may change a PEX definition, delete, or create PEX collections using an existing PEX definitions using this view.
Create PEX Collection...	Displays the PEX Collection Wizard which lets you create a PEX collection on the system. PEX collections contain detailed performance information and can be analyzed via PEX Analyzer.
Open New Data Viewer	Opens a new Data Viewer window. This window is used to display tables and graphs on the system. You can open PEX Analyzer reports into this window or you can also open any other type of physical file and view as a graph or table.
User-defined reports	This set of menu options allows a user to change the database to use when saving/retrieving user-defined reports or to import definitions into the current user-defined reports database. Definitions can be imported from a server (from library QUSRSYS) or from another user's user-defined reports database.
Properties	Use this menu to display the PEX Analyzer property pages. The PEX property pages contain version information and display the job queue and subsystem settings to use when running PEX analysis jobs.

Table of Contents | Previous | Next

4.2 Libraries

The 'PEX Analyzer' folder contains a list of library folders, each representing a library on the iSeries that contains PEX database files. The list displays each library's name and description.

By clicking on a library in the tree you will see its contents (a list of collections).



[The PEX Analyzer component displaying a list of libraries containing PEX data on system 'Rchasbds'.]

Table of Contents | Previous | Next

4.2.1 Menu Options

A library folder in PEX Analyzer has a basic set of menu options that are available within all components as well as some options specific to PEX Analyzer.

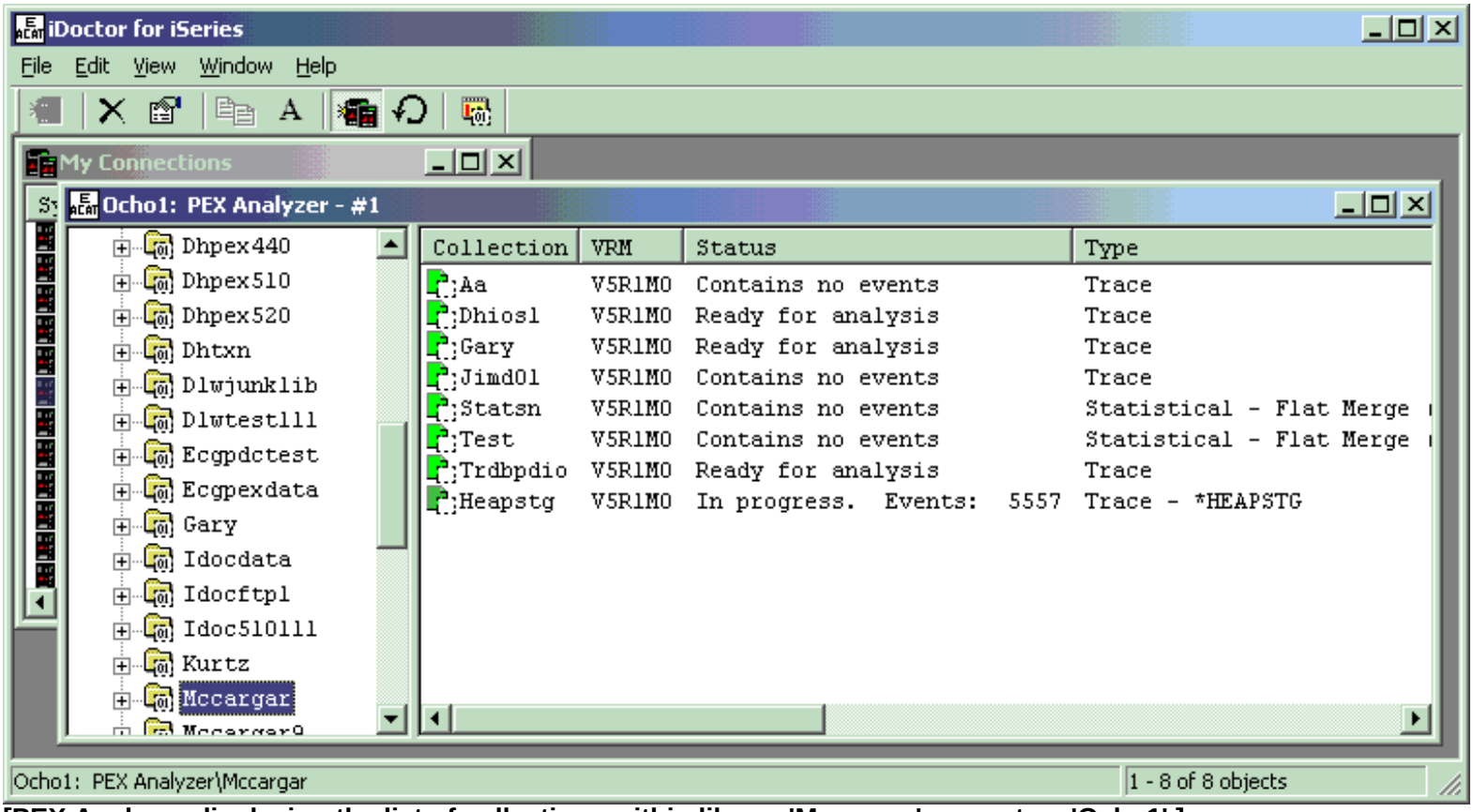
The following are the options for a library folder that are only available in PEX Analyzer.

Menu	Description
Create PEX Collection...	Displays the PEX Collection Wizard which lets you create a PEX collection on the system. PEX collections contain detailed performance information and can be analyzed via PEX Analyzer.
Delete all PEX Analyses...	Use this option to delete all PEX analyses in the current library.
Delete all PEX Collections and Analyses...	Use this option to delete all PEX collections and analyses in the current library.

Table of Contents | Previous | Next

4.3 PEX Collections

Moving down the tree within each library folder are one or more PEX collections. PEX collections that are in the process of being created will be shown in this list as well as the collections that have already been created and are ready for analysis.



[PEX Analyzer displaying the list of collections within library 'Mccargar' on system 'Ocho1'.]

Collection Status

Each collection has a status field indicating if and how a collection may be used. The status is reflected by the type of icon shown for each collection and is summarized in the table below.

Collection Icon	What it means
Green	Collection may be analyzed (using the Analysis Wizard) or deleted. The properties of the collection are viewable.
Dark Green	Collection is either in the process of being created or failed to be created. The status field indicates the status of the job used for creating the collection. The status field for in progress collections will show the number of events that have been collected so far.
Green and Red	Collection is read-only. The collection properties and any already created analyses may be viewed. The collection cannot be deleted.
Red	The collection is not useable. This usually indicates that the PEX data VRM is not supported by the tool. It could also indicate that the PEX collection did not complete successfully or is corrupt in some way (missing files).

Table of Contents | Previous | Next

4.3.1 Menu Options

PEX Collections have a different set of menu options depending on if the collection is in progress or if it has finished collecting and is ready for analysis.

Collections that are in progress have the following set of menu options:

Menu	Description
Stop -> Create DB Files	Stop the collection prematurely and immediately begin creating DB files. Use this after your test has completed and you don't need to collect data any longer.
Stop -> Create Service File	Use this option to generate a single file for the collection. This was used in the past to transfer a single file collection to another system where it would be expanded to a useable multiple file collection there (using the CVTPEXDTA command). You can now FTP collections via the GUI automatically which should reduce the need for this option. If you use this option the service file created is not visible via the GUI. A physical file will be created having the same name as the collection name.
Delete...	Stops collecting immediately and destroys the data that has been collected so far.
Restart	Destroys the data that has been collected so far and then restarts the collection using the same settings.
Resume	Resumes a suspended collection. Use this option to start a collection that was created in 'Standby mode'.
Properties	Displays the basic collection properties like the name and type. From collection properties you can view the job log of the IDOCCOL job running the PEX collection in progress.

Collections that are complete and ready for analysis have these menu options:

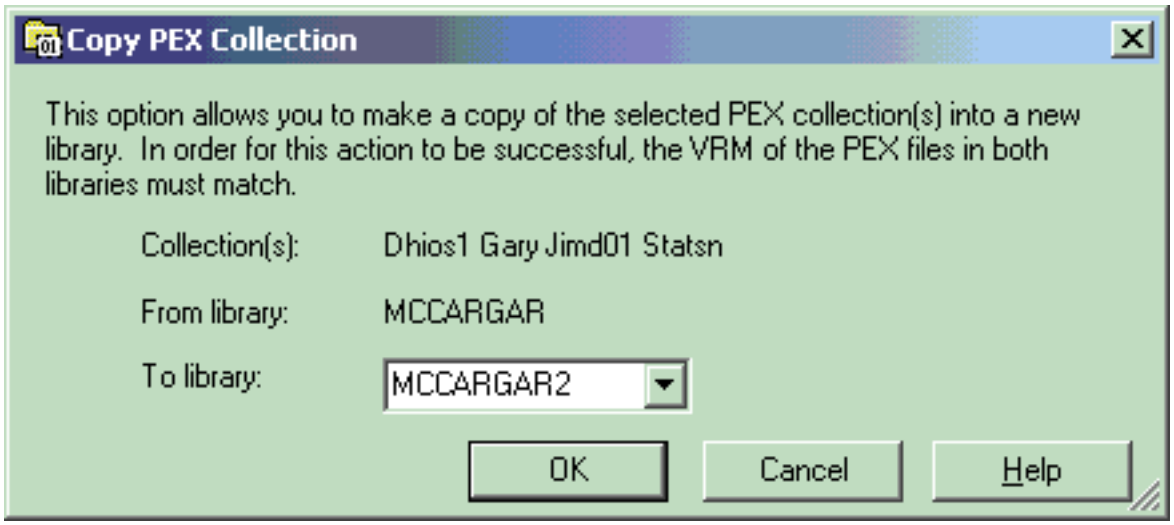
Menu	Description
Explore	Displays the analyses within the PEX collection in the list portion of the PEX Analyzer component view.

Record Quick View	Displays the fields pertaining to the PEX collection vertically. This can help reduce the amount of scrolling required to see all the fields.
Select fields...	Displays an interface that lets you modify which fields are shown for collections. You can optionally include fields such as number of events or number of jobs to get additional information about a list of collections without needing to go into 'Properties'.
Work with Graphs...	Displays an interface that lets you work with the graphs that have been created over analysis files in the current collection.
Work with Queries...	Displays an interface that lets you work with the queries that have been created over analysis files in the current collection.
Copy...	Copy one or more collections to a new library.
Delete...	Delete one or more collections in the current library.
Transfer to...	FTP the selected collection(s) to a remote system.
Properties	Displays the detailed collection properties.

Table of Contents | Previous | Next

4.3.2 Copying

PEX collections can be easily copied to a new library by using the Copy... menu found by right-clicking on a collection. You can select more than one collection at a time to copy multiple collections to a new library in one step.



Field	Description
Collections	List of collections in the 'from library' to be copied to the 'to library'
From library	Name of the library the PEX collections will be copied from.
To library	The name of the library that will receive the PEX collections. By clicking the down arrow you can choose from a list of all libraries on the system. You may also specify a new library that does not exist and it will be created.

Table of Contents | Previous | Next

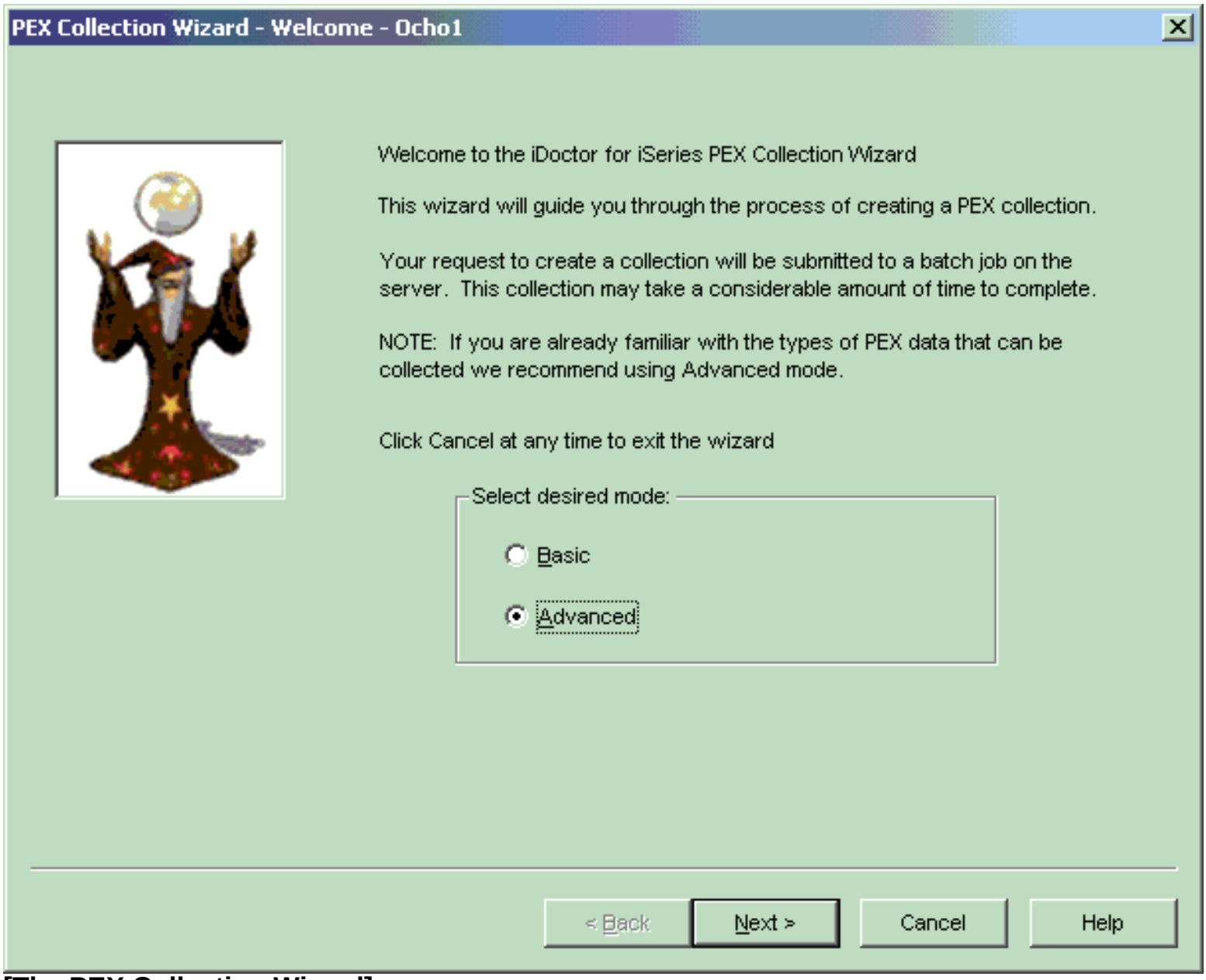
4.3.3 Creating - The PEX Collection Wizard

Within iDoctor for iSeries there are two ways to create a new PEX collection. You can either use the PEX Collection Wizard in the client or you can use the QIDRPA/STRPACOL green screen command. This section covers the PEX Collection Wizard in the GUI.

PEX Collections are created using a PEX definition. Definitions can be created using the green screen ADDPEXDFN command or via the PEX Definition Wizard, also available in the GUI. iDoctor ships several commonly used PEX definitions called 'iDoctor-supplied' PEX definitions. There are several different iDoctor-supplied definitions which cover the most basic problem types. PEX definitions are used to define the specific types of events to capture in your iSeries application or across the entire system and store them in database files when creating a PEX collection.

You can access the PEX Collection Wizard using the Create PEX Collection popup menu when right-clicking on either the PEX Analyzer icon or a library icon.

PEX Collections are created from a batch job on the server (job name IDOCCOL). Depending on the type of data collected, the number of events collected, and the size of ths system, the collection could take anywhere from 30 seconds to hours or even days to complete. For this reason it is important to keep the total time of collection (the Duration parameter) as small as needed.

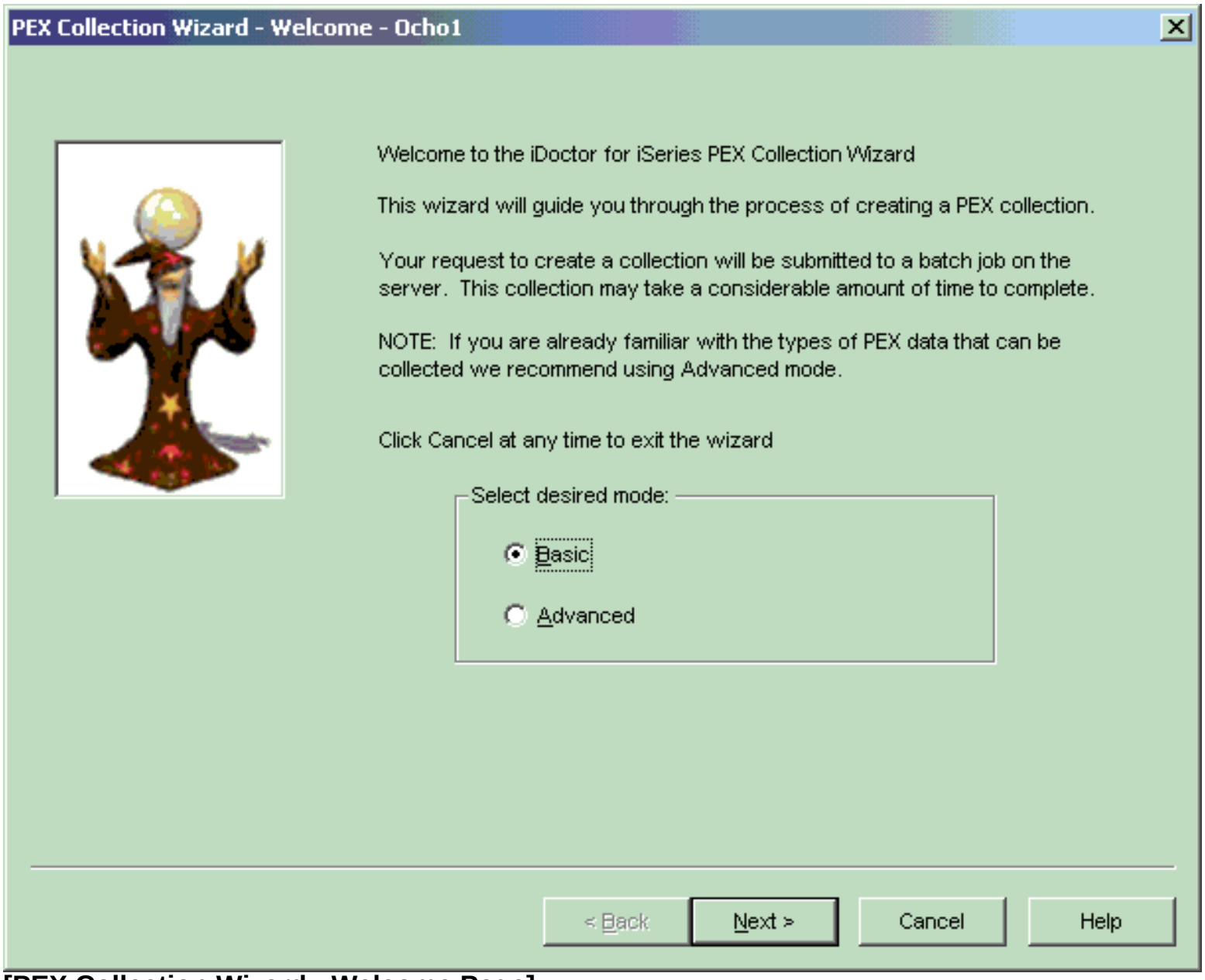


[The PEX Collection Wizard]

Table of Contents | Previous | Next

4.3.3.1 Welcome

The Welcome page in the PEX Collection Wizard introduces the user to the wizard and offers information about what the wizard will do. From here a user can decide which mode to run the wizard in: basic or advanced. Basic mode will follow with a series of questions designed to help a user determine what type of iDoctor-supplied definition best fits the type of performance problem they are having. Advanced mode skips the questions and goes right into the Collection Options page.



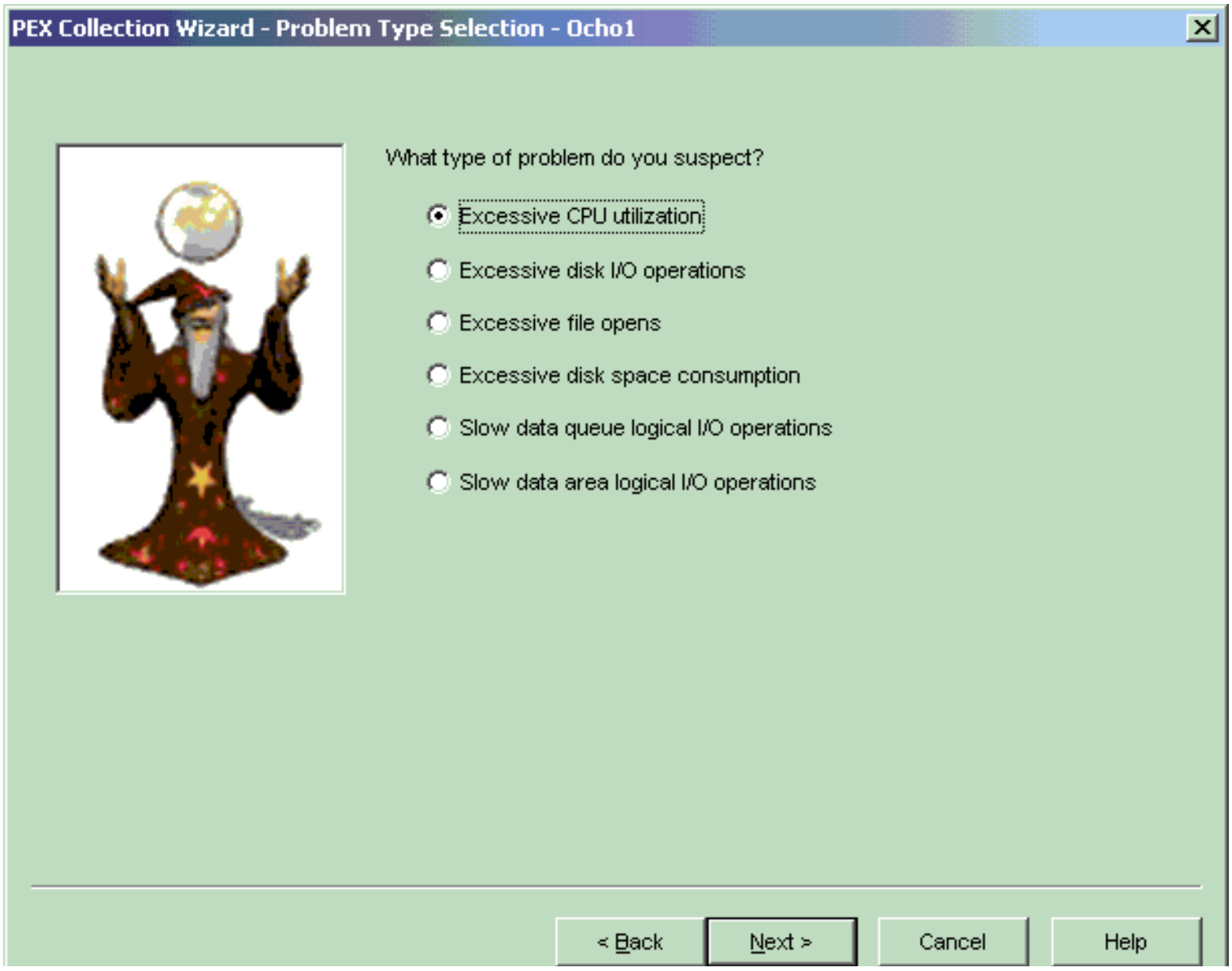
[PEX Collection Wizard - Welcome Page]

Table of Contents | Previous | Next

4.3.3.2 Problem Type Question Pages

The Problem Type Question Pages present a series of questions designed to help a user more easily determine the type of iDoctor-supplied definition to use. These question pages are only shown when running the Wizard in Basic mode. An expert would typically use Advanced mode and skip these questions. Each set of responses on these pages leads to a single iDoctor-supplied definition. This definition will be automatically selected on the Collection Options Page once this iDoctor-supplied definition type is determined.

The questions will follow a flow from general categories of problems like the page shown below down to very specific questions that when answered will determine exactly which iDoctor-supplied definition best fits the situation.

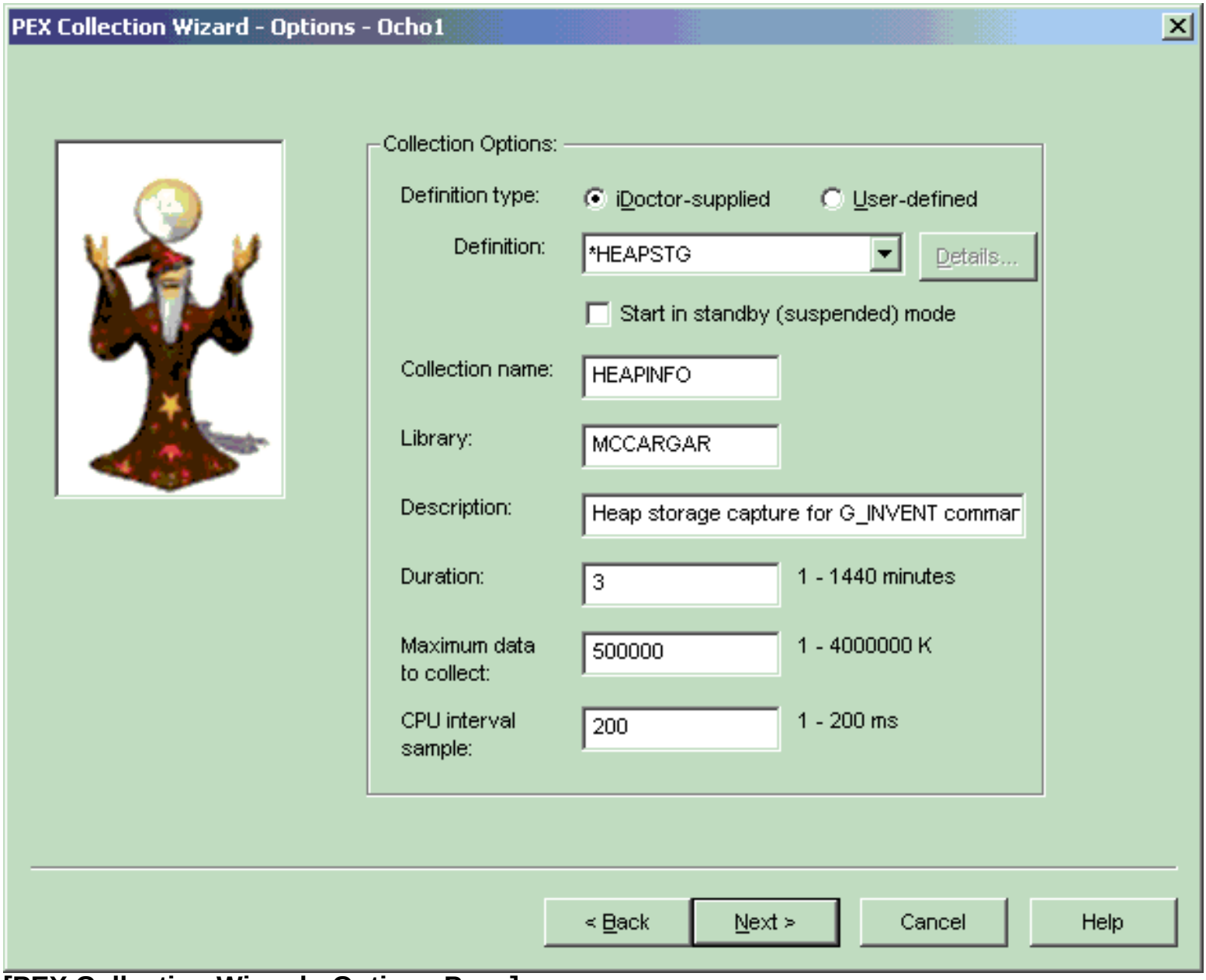


[PEX Collection Wizard - One of the Problem Type question pages]

Table of Contents | Previous | Next

4.3.3.3 Options

The Options Page allows the user to specify the most basic pieces of information about a collection like the PEX definition to use when creating the collection, the name of the collection, library to store the collection in, and more.



[PEX Collection Wizard - Options Page]

The following table provide more information about each of the criteria available on this page:

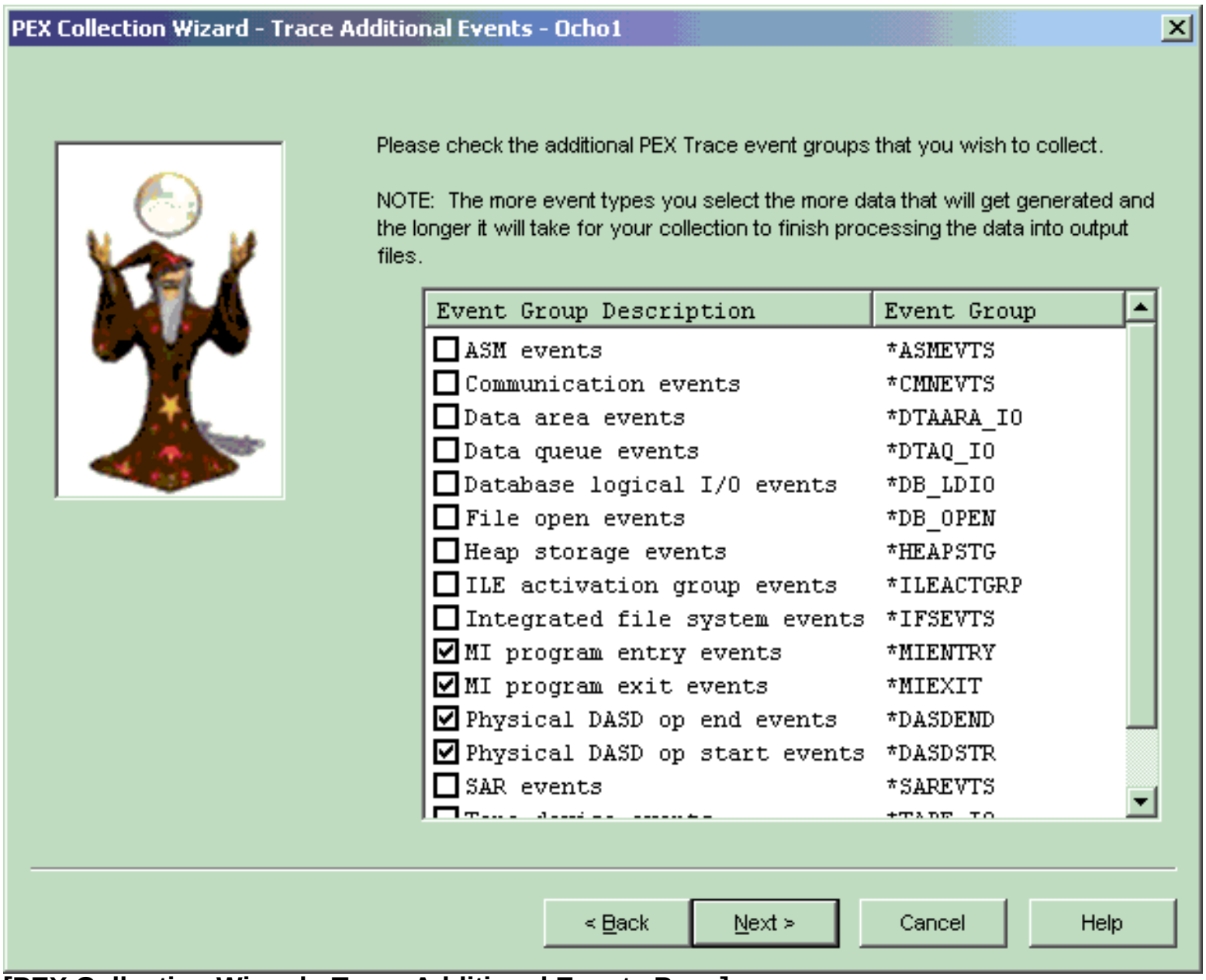
Field	Description
Definition Type	This indicates if the PEX definition will be iDoctor-supplied or user-defined. You can define your own PEX definition using the PEX Definition Wizard or via the ADDPEXDFN command. If you select the user-defined option you can click the Details.. button to quickly see all the details for the PEX definition in a format similar to that provided via PRTPEXRPT command. When using a user-defined PEX definition the rest of the selection pages in the Wizard like Job selection and Task selection are skipped. This is because any Job or Task criteria will come from the PEX definition.
Definition	The name of the iDoctor-supplied or user-defined PEX definition. You must select a value from the list. In Basic mode this field will be preselected based on the answers to the problem type questions.
Details button	Displays the properties for the selected user-defined definition.
Start in standby mode option	Check this box to create the collection but to have it be initially in suspended mode. This option is useful if you need to start the collection at a more exact time (right after a test program is called perhaps) because resuming a suspended collection is much faster than starting a new one
Collection name	The name of the PEX collection. The collection name matches the member name created in each of the PEX files stored in the library.
Library name	The name of the library to create the PEX collection in.
Duration (minutes)	The total amount of time to spend collecting data. This value is listed in minutes and must have a value from 1 to 1440. Certain definition types like task switch can generate many million events (records) in a relatively short amount of time. Make sure this value is not too large to avoid ending up with much more data than desired.
Maximum data to collect	The maximum amount of disk space this collection should use in kilobytes. The default value is 500,000. When using a user-defined PEX definition this parameter is ignored because it is provided within the PEX definition.
CPU interval sample	Specifies the size of the interval which CPU samples are taken of the program. A low interval will cause a high number of samples to be taken, and will also cause higher overhead. A low interval will also provide relatively more data. This parameter will be grayed out if it does not apply to the selected iDoctor-supplied definition.

Table of Contents | Previous | Next

4.3.3.4 Trace Additional Events

This page is only shown when the iDoctor-supplied definition is one that generates a PEX Trace collection. You will not see this page when creating a collection using a user-defined PEX definition.

The list contains additional event groups that can be added to the collection. The more event types you select the more data that will get generated and the longer it will take for your collection to finish processing the data into output files.

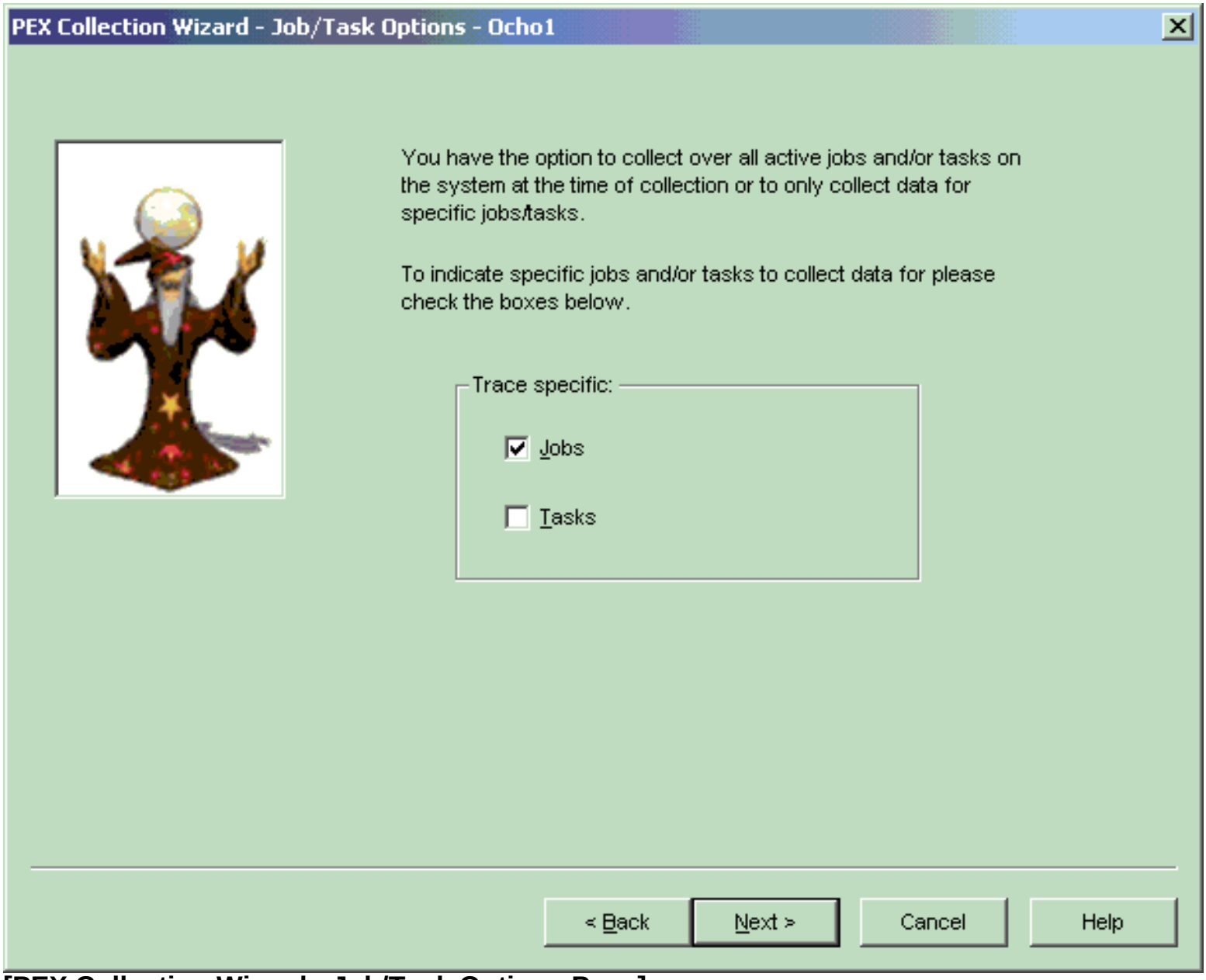


[PEX Collection Wizard - Trace Additional Events Page]

Table of Contents | Previous | Next

4.3.3.5 Job/Task Options

On this page you may decide if you would like to select specific jobs or tasks to include in the PEX collection. Selecting specific jobs and tasks is optional, but is necessary when you only want to collect data for the job(s) or task(s) you are interested in.

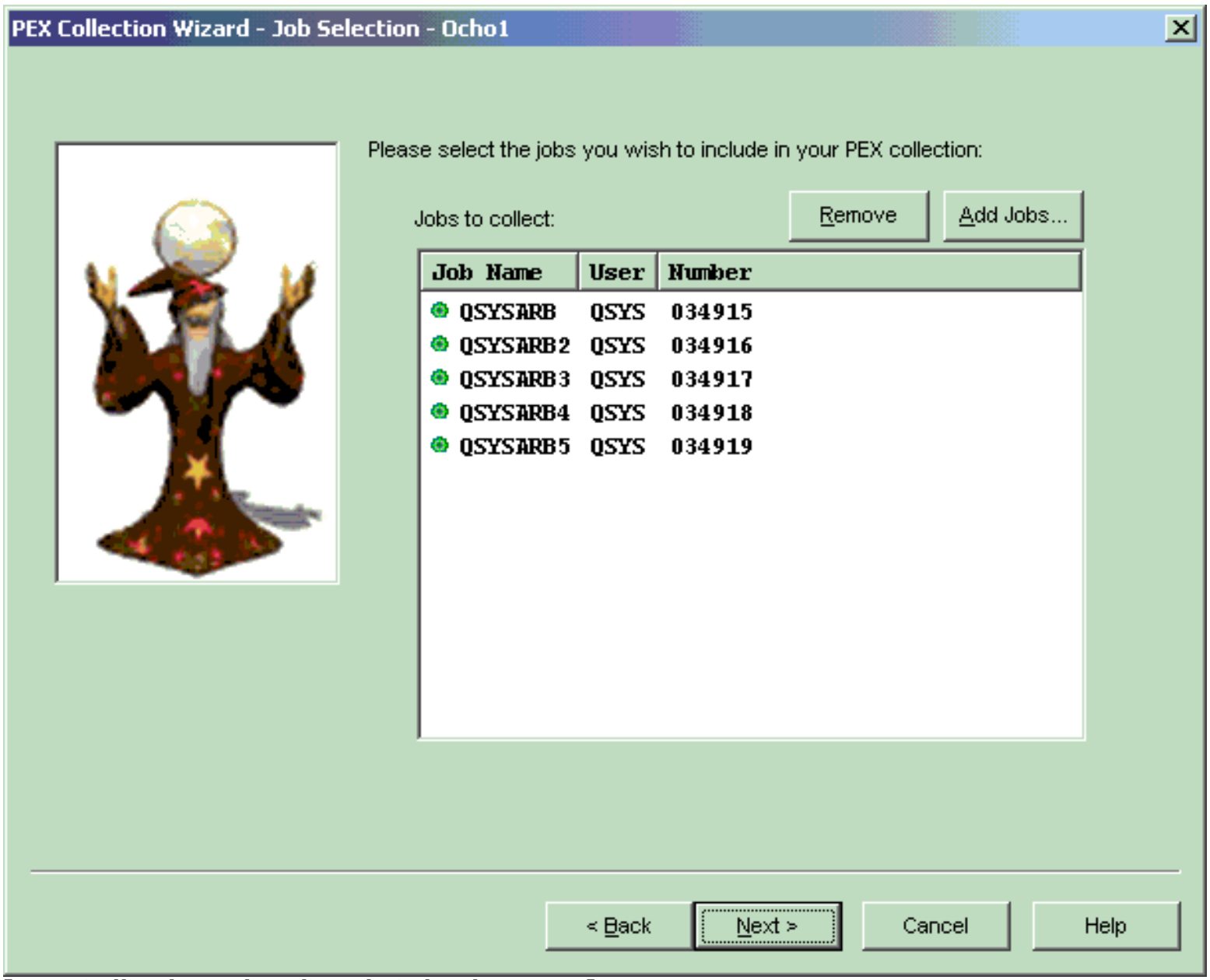


[PEX Collection Wizard - Job/Task Options Page]

Table of Contents | Previous | Next

4.3.3.6 Job Selection

The job selection page displays a list of selected job information to be captured in the PEX collection. There are also two buttons on this page used to add or remove jobs from the list.



[PEX Collection Wizard - Job Selection Page]

The table below summarizes the different elements on this page:

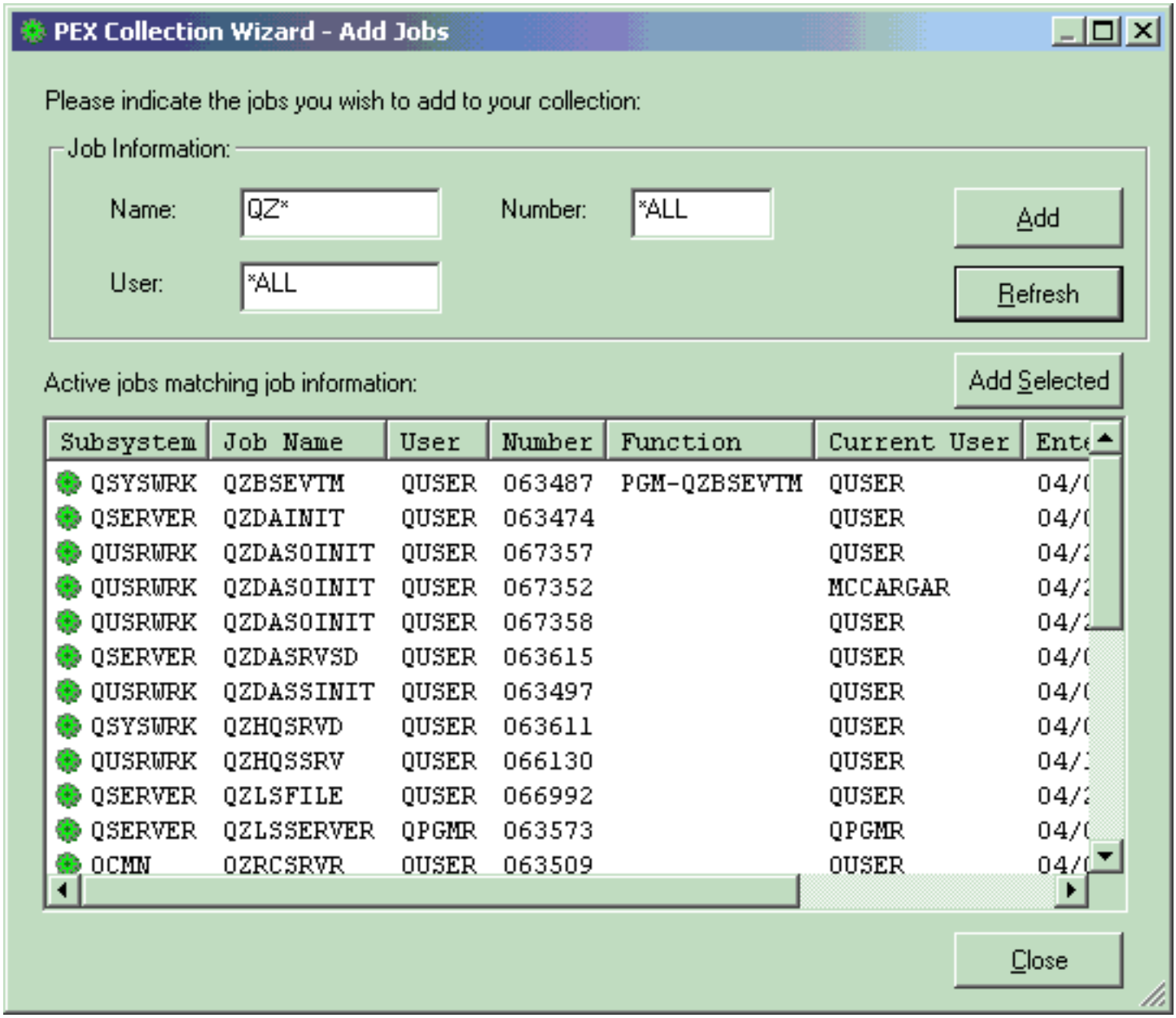
Field	Description
Jobs list	A list of jobs to collect information about in the PEX collection.
Remove button	This button removes the selected jobs from the list.
Add Jobs button	Use this button to open the Add Jobs Window (discussed in the next section). This window is used to select and add additional jobs to the list.

Table of Contents | Previous | Next

4.3.3.7 Add Jobs Window

The add jobs window allows a user to add jobs to the Job Selection page in the wizard. Job information can be of two types: generic job name/generic job user/generic job number -or- job name/job user/job number.

The "Job Information" portion of the window includes text fields used to define a generic job to add to the Job Selection Page or to use as a filter when refreshing the list of jobs shown in the window. The Add button will add the current generic job to the Job Selection page and the Add Selected button will add the selected jobs from the active jobs list to the Job Selection page.



[PEX Collection Wizard - Add Jobs Window]

The table below summarizes the different elements on this page:

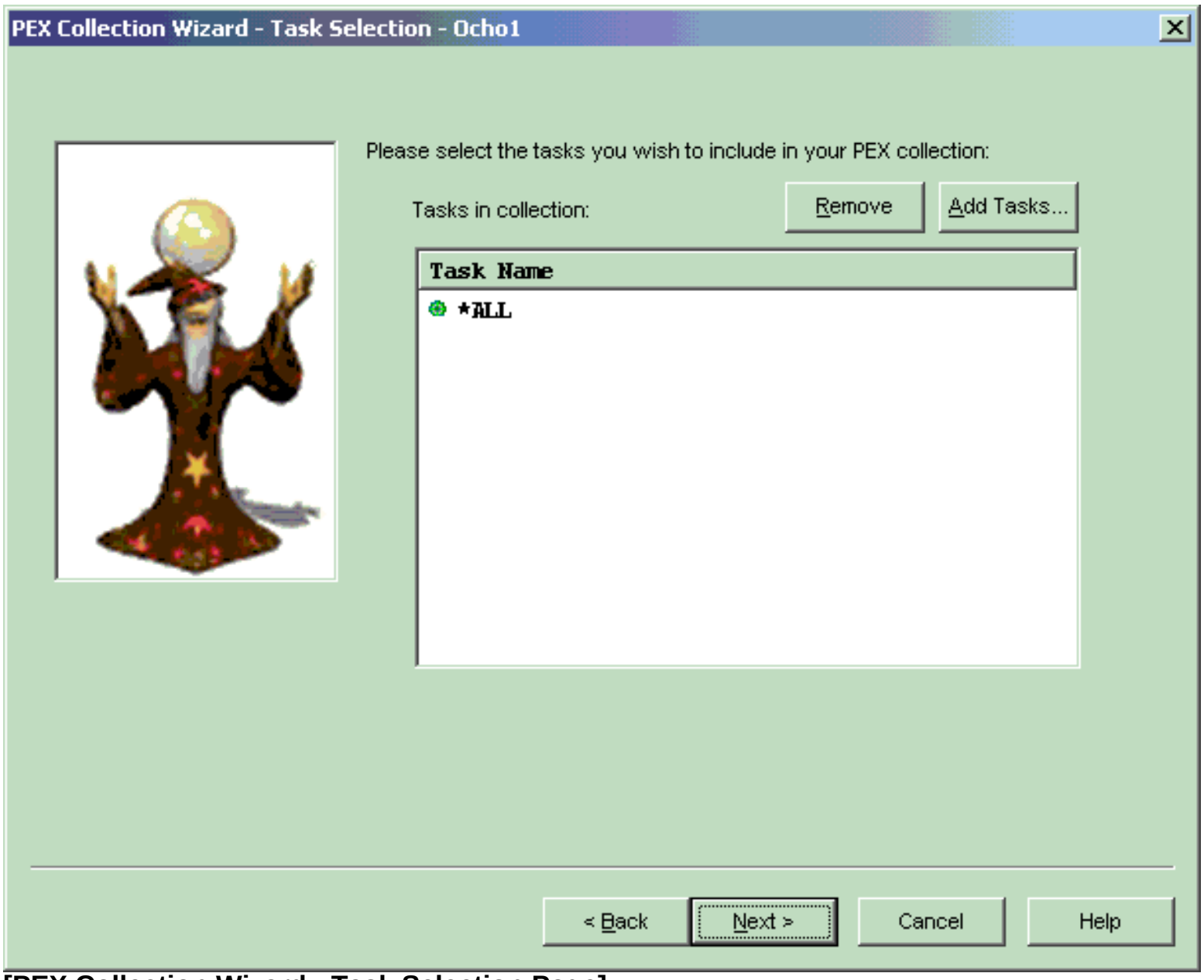
Field	Description
Name	A text field for entering a generic or specific job name. When specifying a generic name use a * at the end of the job name.
User	A text field for entering a generic or specific job user. When specifying a generic name use a * at the end of the job user name.
Number	A text field for entering a specific job number or *ALL.

Add button	This button will add the current job name/user/number values in the text fields to the Job Selection page. This can be used to add a generic job name/user/number value such as QZ*/MCCARGAR/*ALL This value indicates all job names starting with QZ, for job user MCCARGAR.
Refresh button	This button is used to refresh the active jobs list based on the current values specified in the name, user and number text fields.
Add Selected button	Use this button to add the selected active jobs to the Job Selection Page.
Active jobs matching job information list	This list shows all active jobs on the system matching the current Job information specified. When this window is first open the list will show all active jobs on the system.

Table of Contents | Previous | Next

4.3.3.8 Task Selection

The task selection page displays a list of selected tasks to be captured in the PEX collection. There are also two buttons on this page used to add or remove tasks from the list.



[PEX Collection Wizard - Task Selection Page]

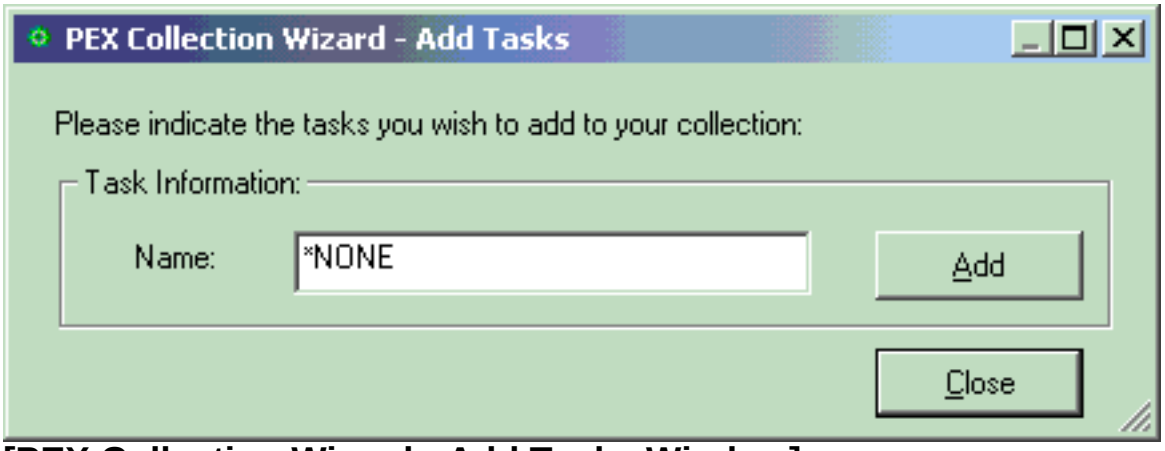
The table below summarizes the different elements on this page:

Field	Description
Tasks list	A list of tasks that will be captured in the PEX collection.
Remove button	This button removes the selected tasks from the list.
Add Tasks button	Use this button to open the Add Tasks Window (discussed in the next section). This window is used to add task information to the task list.

Table of Contents | Previous | Next

4.3.3.9 Add Tasks Window

The add tasks window allows a user to add tasks to the Task Selection page in the wizard. The task name can either be *ALL, *NONE, a generic task name like Q*, or a specific task name. Change the task name field and click the add button for each task that you would like to capture in your PEX collection.



[PEX Collection Wizard - Add Tasks Window]

The table below summarizes the different elements on this page:

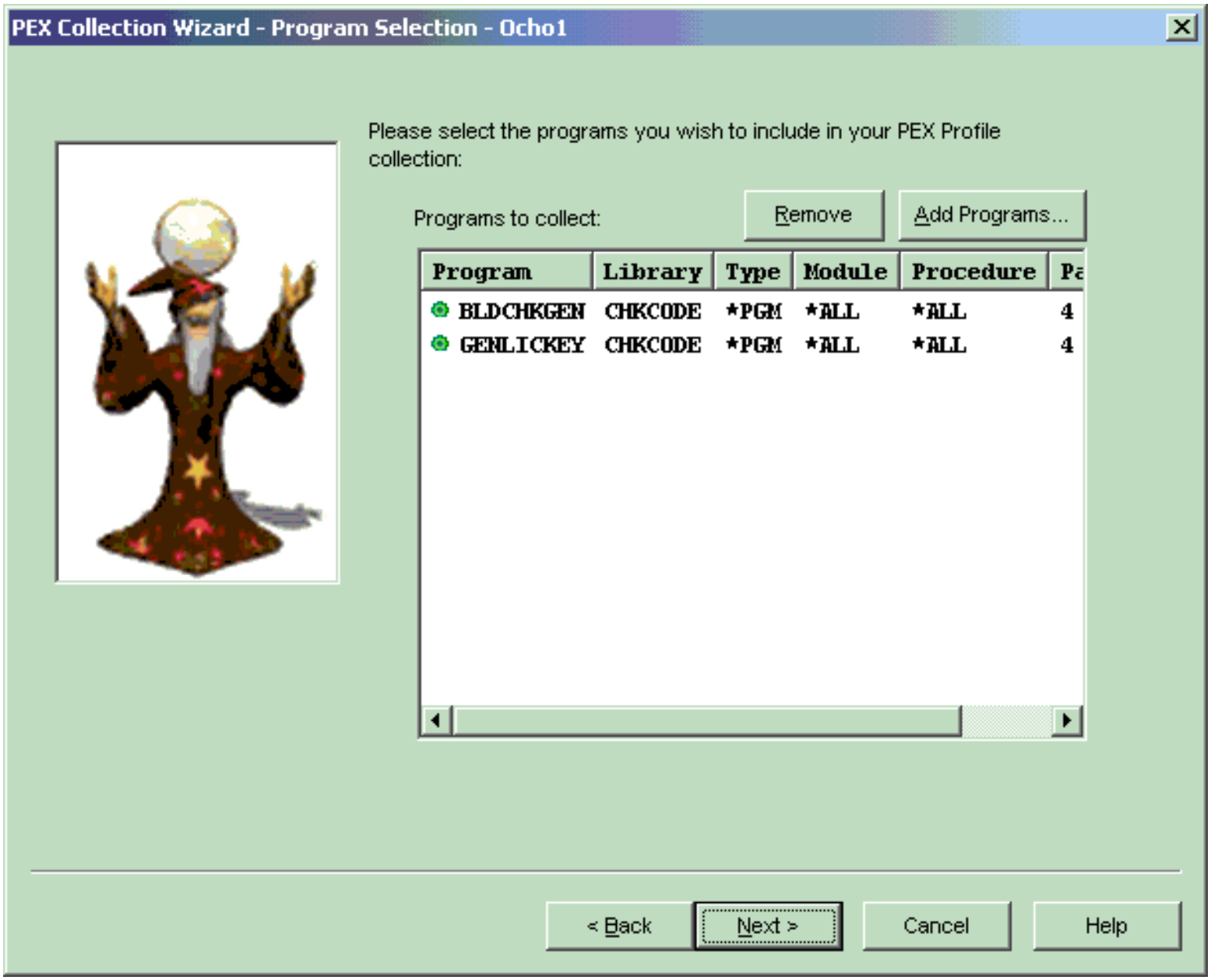
Field	Description
Name	A text field for entering a generic or specific task name. This value can also be *ALL or *NONE. When specifying a generic name use a * at the end of the task name.
Add button	This button will add the current task information to the Task Selection page.
Close button	Closes this window

Table of Contents | Previous | Next

4.3.3.10 Program Selection

The program selection page allows the user to select up to 16 program/module/procedure entries when creating a PEX Profile collection. This window is only displayed when using the iDoctor-supplied definition *PROFILE.

This page displays a list of selected program information to be captured in the PEX collection. Note that in order to get any useful information from this definition type you must select programs that have the enable for profiling flag turned on. There are also two buttons on this page used to add or remove programs from the list.



[PEX Collection Wizard - Program Selection Page]

The table below summarizes the different elements on this page:

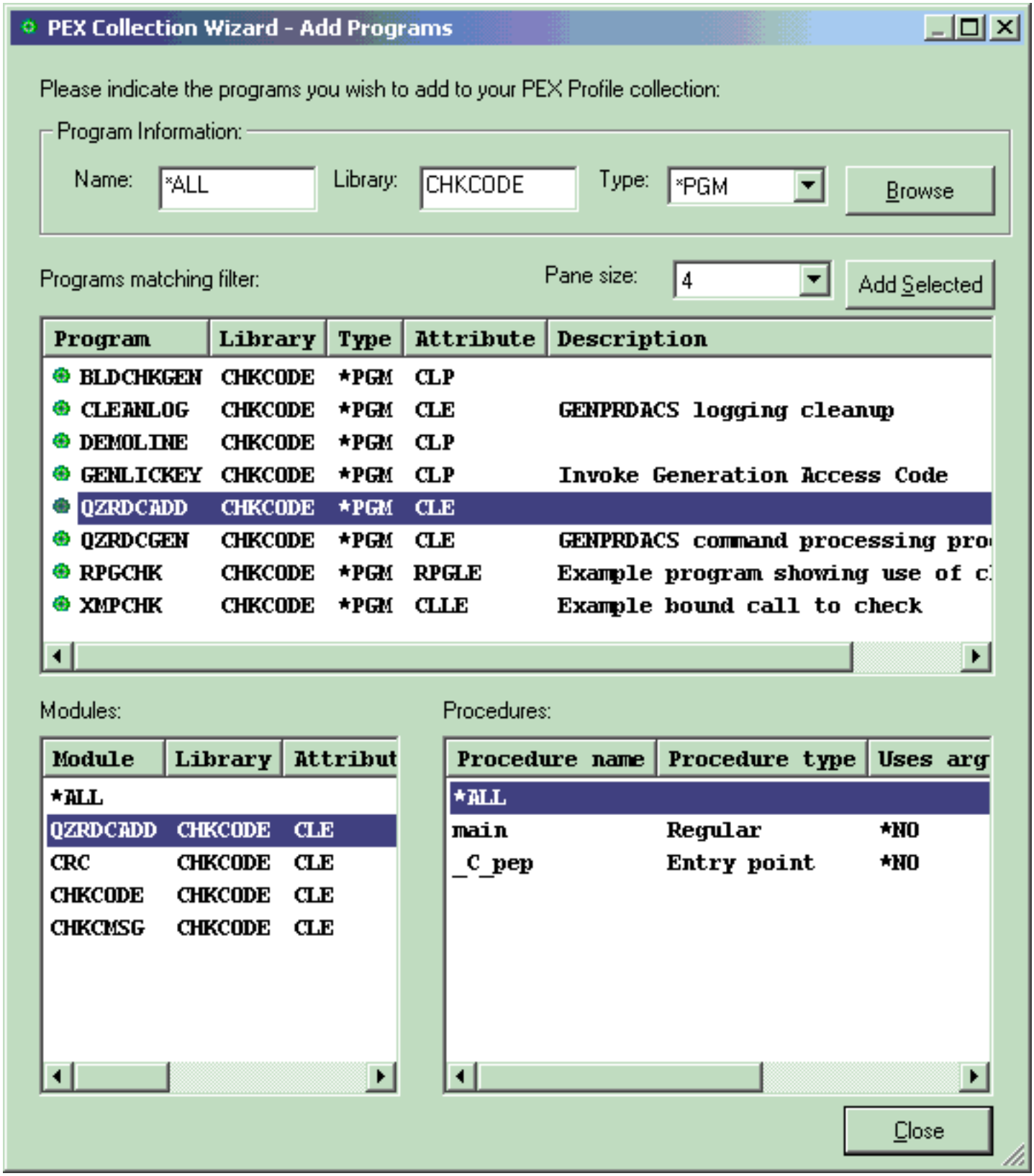
Field	Description
Programs list	A list of program information that will be captured in the PEX Profile collection.
Remove button	This button removes the selected program information from the Programs list.
Add Programs button	Use this button to open the Add Programs Window (discussed in the next section). This window is used to select and add additional program information to the Programs list.

Table of Contents | Previous | Next

4.3.3.11 Add Programs Window

The add programs window allows a user to browse any programs/service programs on the system using generic program and library names for the purpose of adding them to a PEX Profile collection. After finding the programs you want to add to the collection click the Add Selected button to add the selected program/module/procedure to the list. If a program is an ILE program you will see the modules contained within the program in the modules list. If desired select on these modules to see procedure entries found in the module. By selecting a specific program/module/procedure combination you can collect information only about the procedure(s) you are interested in.

The enable profiling flag must be turned on in the program and module you select in order to add the program/module/procedure information to the Program Selection Page.



[PEX Collection Wizard - Add Programs Window]

The table below summarizes the different elements on this page:

Field	Description

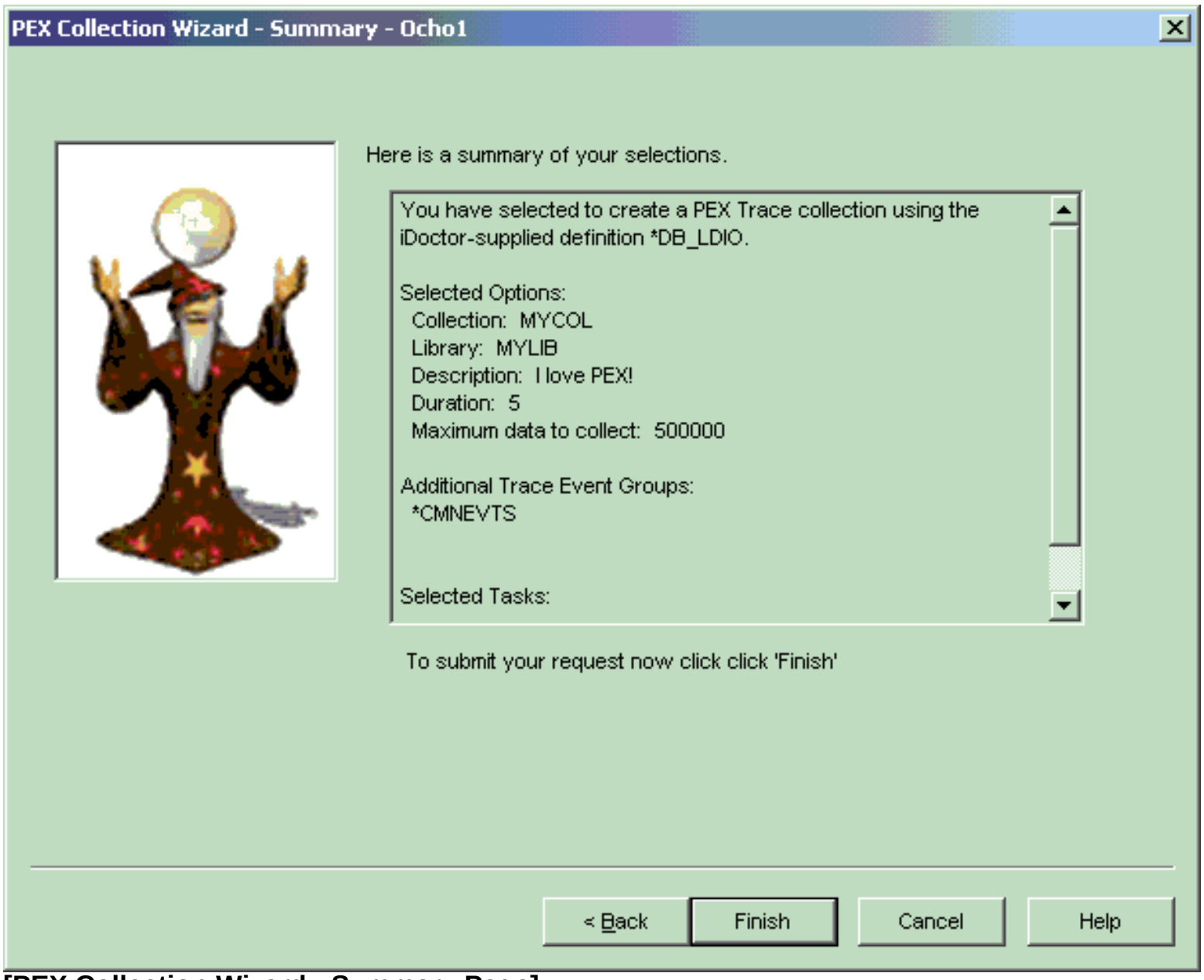
Name	A text field for entering a generic or specific program name. When specifying a generic name use a * at the end of the program name.
Library	A text field for entering a generic or specific library name. When specifying a generic name use a * at the end of the library name.
Type	This drop down lists contains the values *PGM and *SRVPGM. This offers the user the choice of viewing programs or service program objects.
Browse button	The browse button updates the contents of the programs list.
Pane size	The pane size is the number of consecutive program instruction addresses assigned to each counter. The smaller the pane size, the more fine-grained the program profile information will be.
Add Selected button	This button will add the selected program, module and procedure to the Program Selection Page. The program added must be enabled for profiling.
Programs List	The list of programs matching the program information listed above. This list can display programs or service programs. Changing your selected program in the list will immediately refresh the module list showing the module information found within the seleted program.
Module List	The list of modules found within the selected program in the Programs List. Changing the selected module in the list will immediately refresh the procedure list showing the procedure information found within the seleted program.
Procedure List	The list of procedures found within the selected module.
Close button	Close the Add Programs window.

Table of Contents | Previous | Next

4.3.3.12 Summary

The summary page of the PEX Collection Wizard presents a summarization of all of the input provided in the wizard. It lists all of the details about the selected jobs/tasks/programs as well as information from the Options page like collection name and duration.

To submit the job to create the iDoctor PEX Collection click on the Finish button. After submitting your collection go to the library that the collection is going to be created in to see the status of the PEX collection in progress.



[PEX Collection Wizard - Summary Page]

Table of Contents | Previous | Next

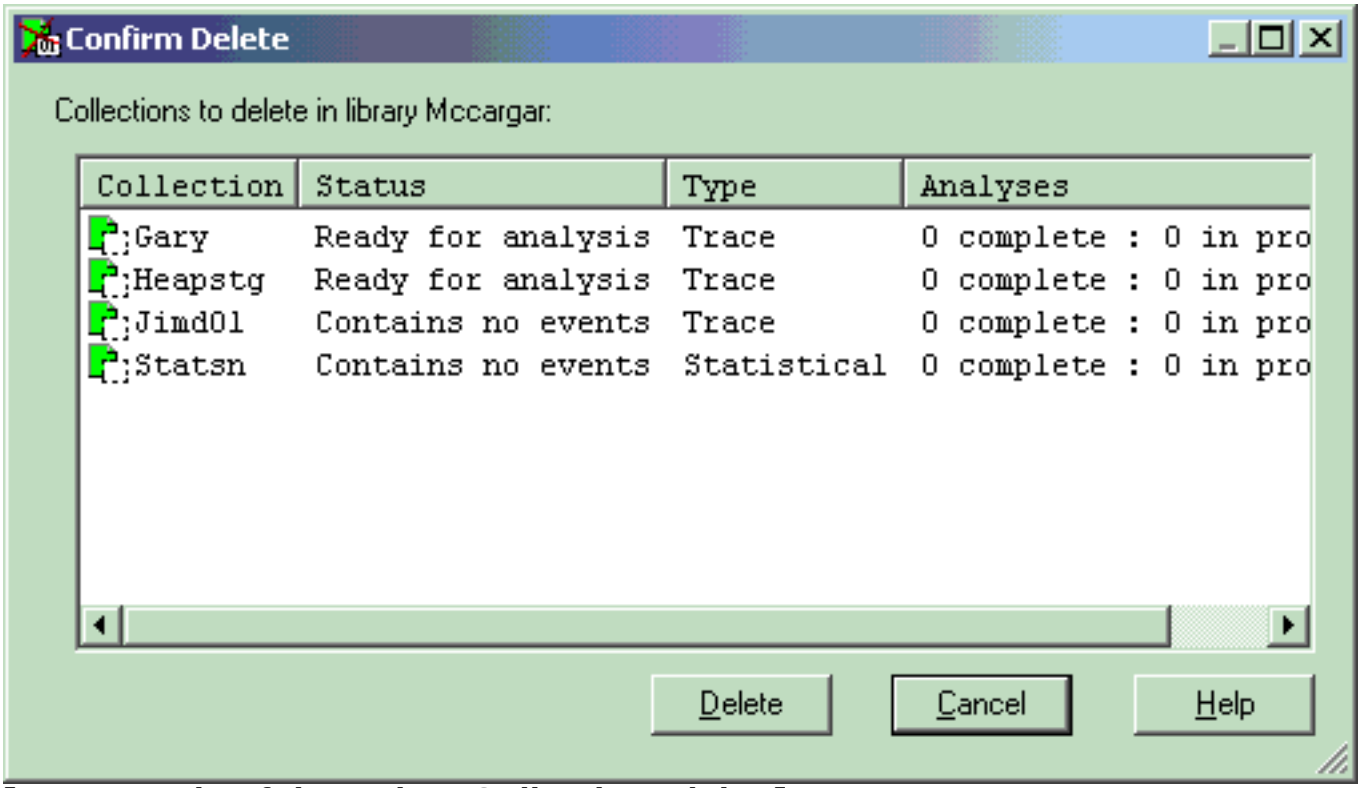
4.3.4 Defining

The events to capture in a PEX collection are defined using a PEX definition. To learn more visit the section on PEX Definitions.

Table of Contents | Previous | Next

4.3.5 Deleting

A user may choose to delete one or more collections using the Delete... menu found by right-clicking on a collection. If you have trouble deleting a collection there may be someone else using it (possibly you if you have some of its data open in the Data Viewer). Ensure that all table or graph views over the collections you want to delete have been closed down before trying to delete them.



[An example of the Delete Collection Dialog]

Table of Contents | Previous | Next

4.3.6 Generating analyses

A PEX collection may be analyzed by right-clicking on it and choosing the Analyze Data... menu. The types of analyses available depends on the type of data has been collected. Visit the documentation on PEX Analyses for more information.

Table of Contents | Previous | Next

4.3.7 Property Pages

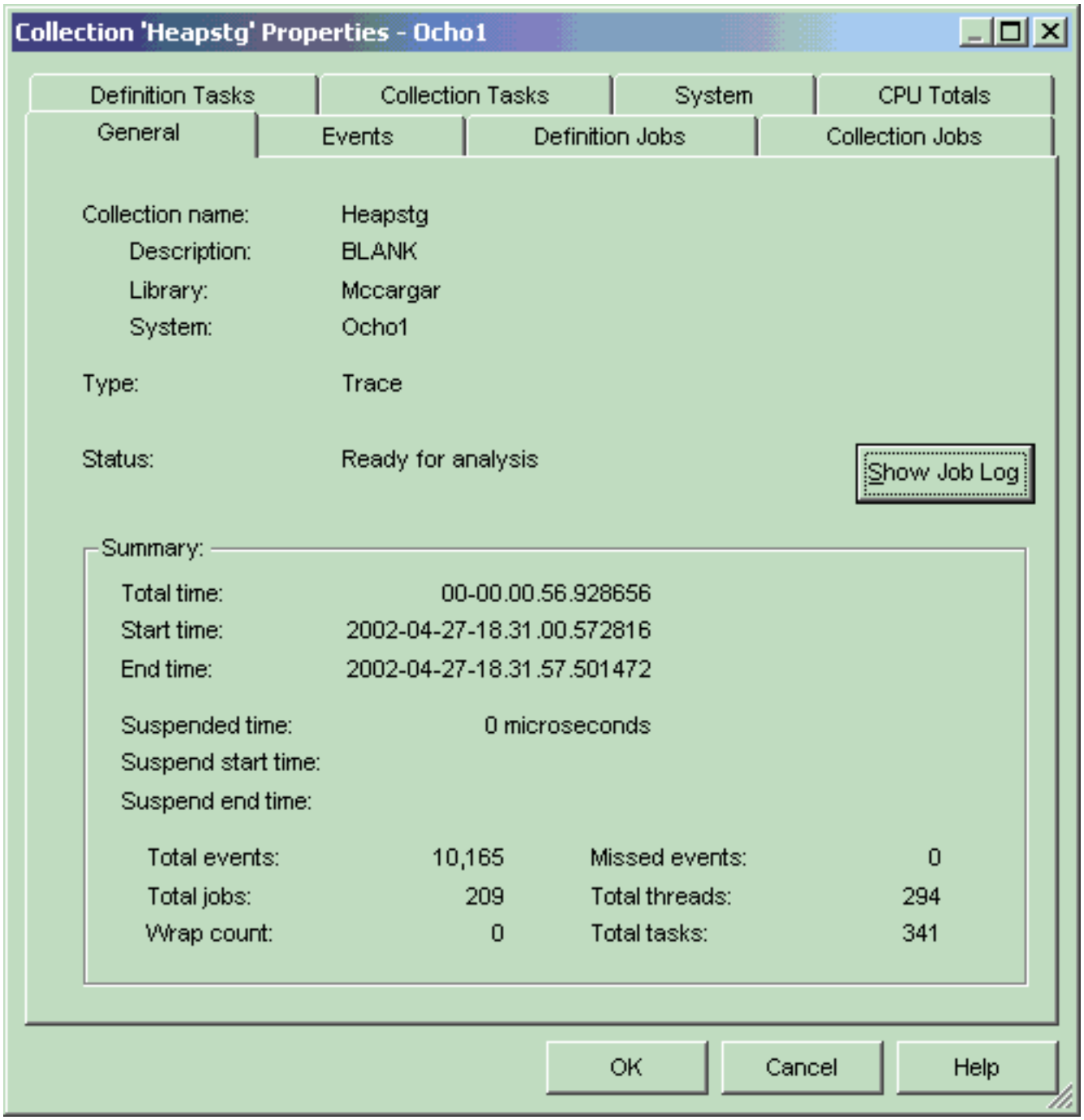
A PEX collection contains a huge amount of performance information which is summarized within the PEX collection property pages. A user may invoke the property pages by right-clicking on the desired collection and choosing Properties.

There are three primary types of collections: Trace, Profile and Statistical. The property pages available to the user varies slightly based on the type of collection used. Each property page will be discussed next along with any differences that exist based on the collection type involved.

Table of Contents | Previous | Next

4.3.7.1 General

The General property page provides a summary of the most basic information about a collection.



[General Property Page for collection 'Heapstg']

The following information is displayed on the General property page:

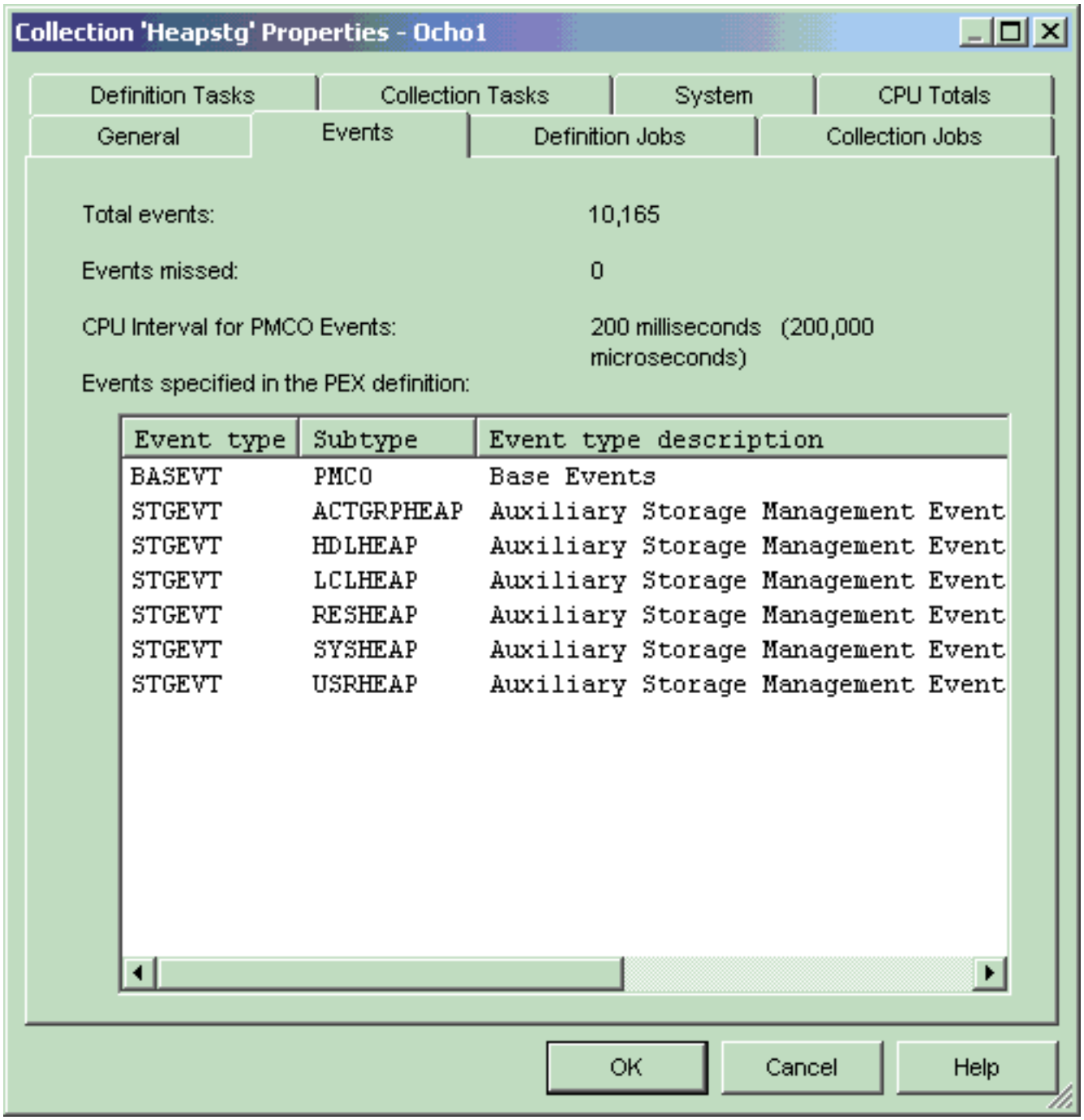
Field Name	Field Description
Collection name	PEX collection name. The collection name is a unique database file member name in the specified library.
Description	Collection description. A user-defined description given to a collection when it is created.
Library	Library containing the collection.
System	The current system name.
Type	Trace, statistical, and profile are the three primary types of PEX data. There are two subtypes of statistical - hierarchical and flat.
Status	Indicates whether or not the collection is useable by iDoctor or indicates the status of the job creating the collection if the collection is currently in the process of being created.
Show Job Log button	This button is used to display the job log of the job used for creating the PEX collection. If the job log is not found on the system the button will not be visible.
Total time	Total elapsed time the collection was gathering data.
Start time	Date and time the collection was created.
End time	Date and time the collection stopped gathering data.
Suspended time (us)	Total number of microseconds the collection was suspended during data gathering. If this field is 0 the collection was not suspended.
Suspend start time	Date and time the collection was suspended. If this field is blank the collection was not suspended.
Suspend end time	Date and time the collection was resumed. If this field is blank the collection was not suspended.
Total events	Number of individual events that occurred within the course of the collection.
Total jobs	Number of unique jobs captured in the collection.
Missed events	Number of events missed. If this is greater than zero your collection data may be corrupt.
Total threads	Number of unique threads captured in the collection.
Total tasks	Number of unique tasks captured in the collection.

Wrap count	Indicates if the collection data was wrapped.

Table of Contents | Previous | Next

4.3.7.2 Events

The Events property page provides information about the events that were captured within the collection.



[An example of a collection's Events Property Page]

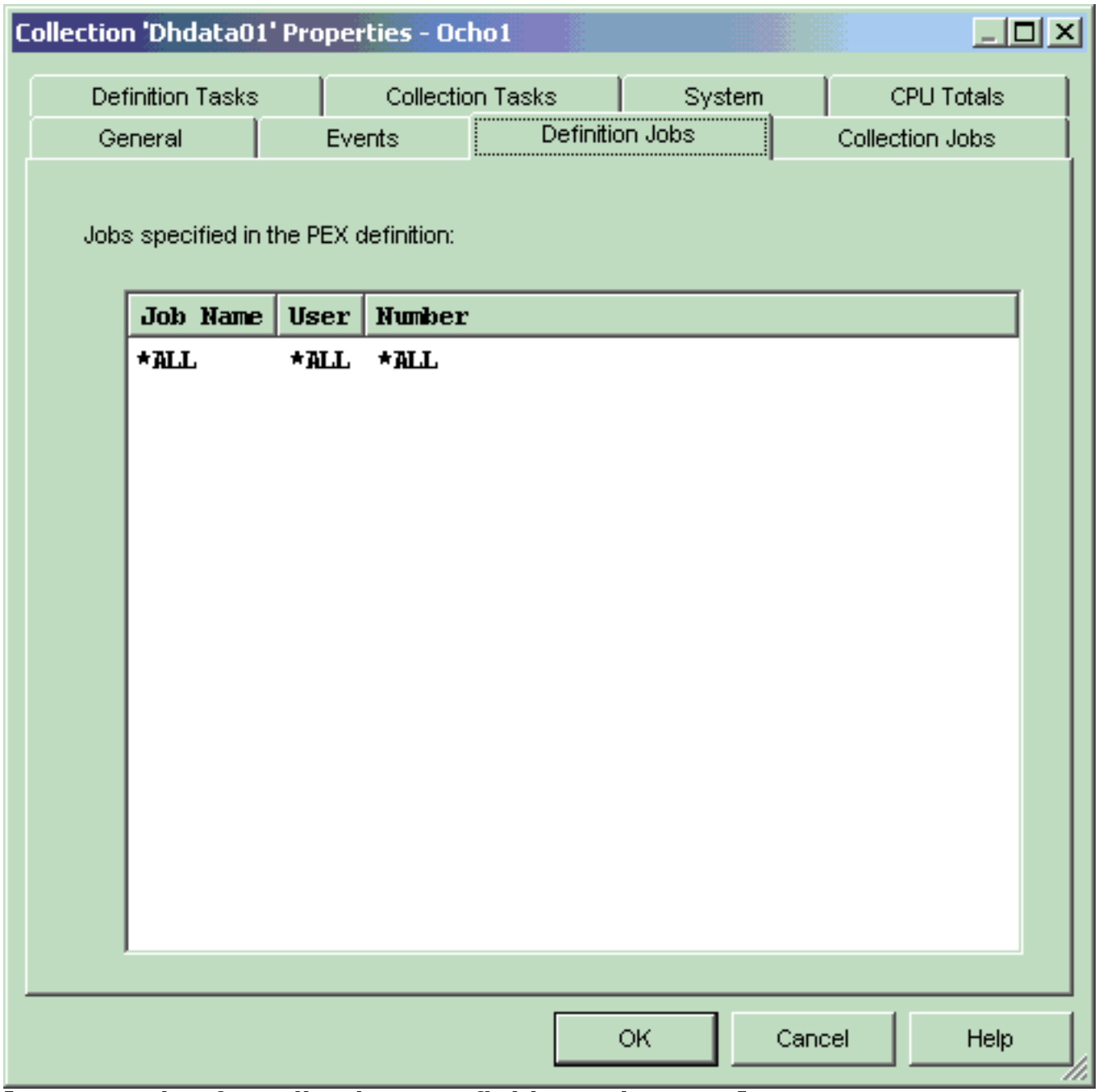
The following information is displayed on the Events property page:

Field Name	Field Description
Total events	The total number of instances of requested events that exist within the collection.
Events missed	The total number of events not captured within the collection.
CPU Interval for PMCO Events	A value used to determine the frequency of Performance Measurement Counter Overflow (PMCO) events. A low value generates more PMCO events than a high value.
Events specified in the PEX definition list	A list of events requested to be included in the collection data. An event is an event type/subtype combination. This combination of event type and subtype is referred to as a PEX trace event.

Table of Contents | Previous | Next

4.3.7.3 Definition Jobs

The Definition Jobs property page provides information about the jobs the PEX definition was set up to capture.



[An example of a collection's Definition Jobs Page]

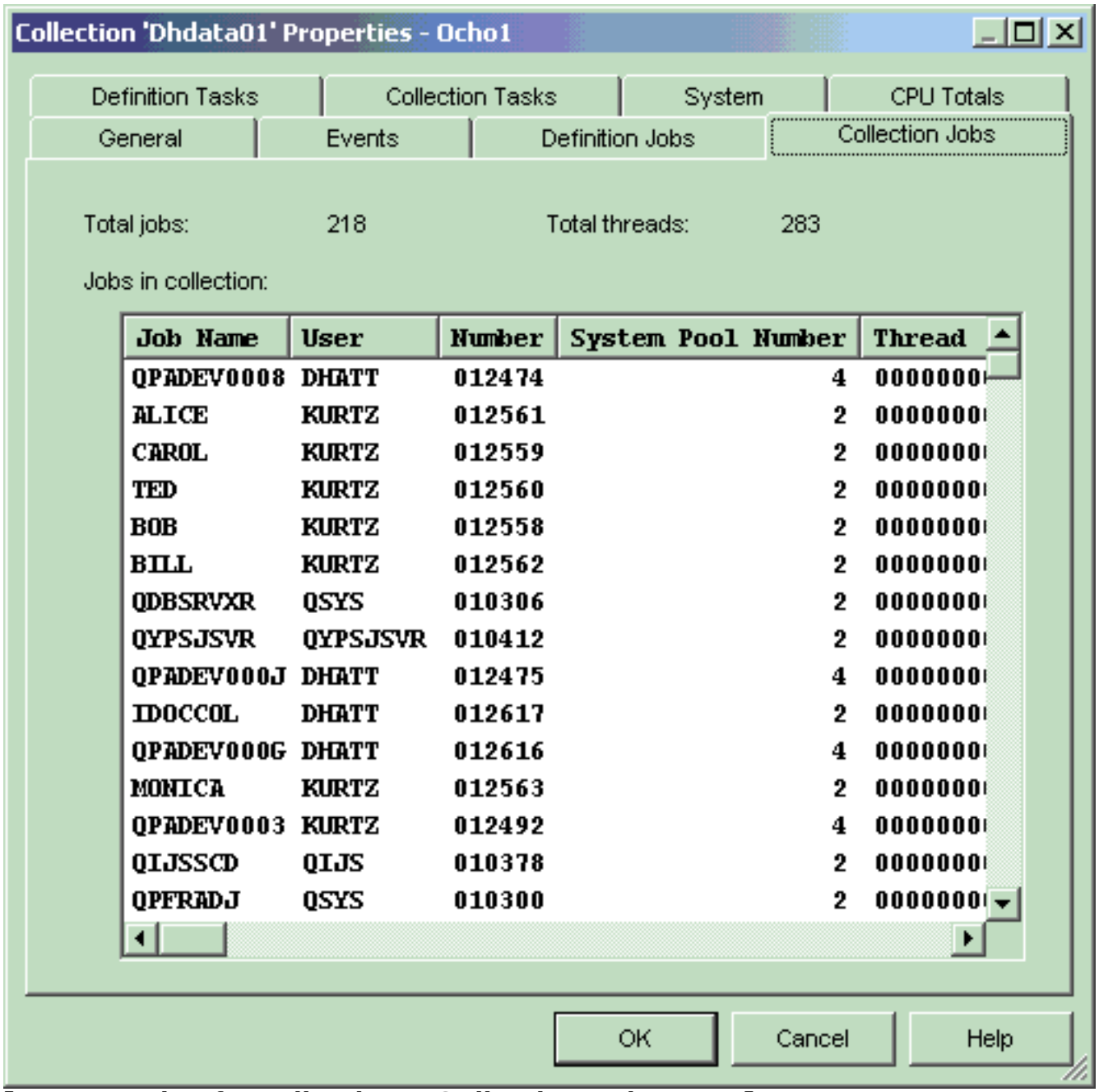
The following information is displayed on the Definition Jobs property page:

Field Name	Field Description
Jobs specified in the PEX definition list	A list of all jobs that were defined to be included in the collection when the collection is started using that definition. These jobs may or may not be actually present within the collection data itself.

Table of Contents | Previous | Next

4.3.7.4 Collection Jobs

The Collection Jobs property page lists every job and thread that was captured in the PEX collection.



[An example of a collection's Collection Jobs Page]

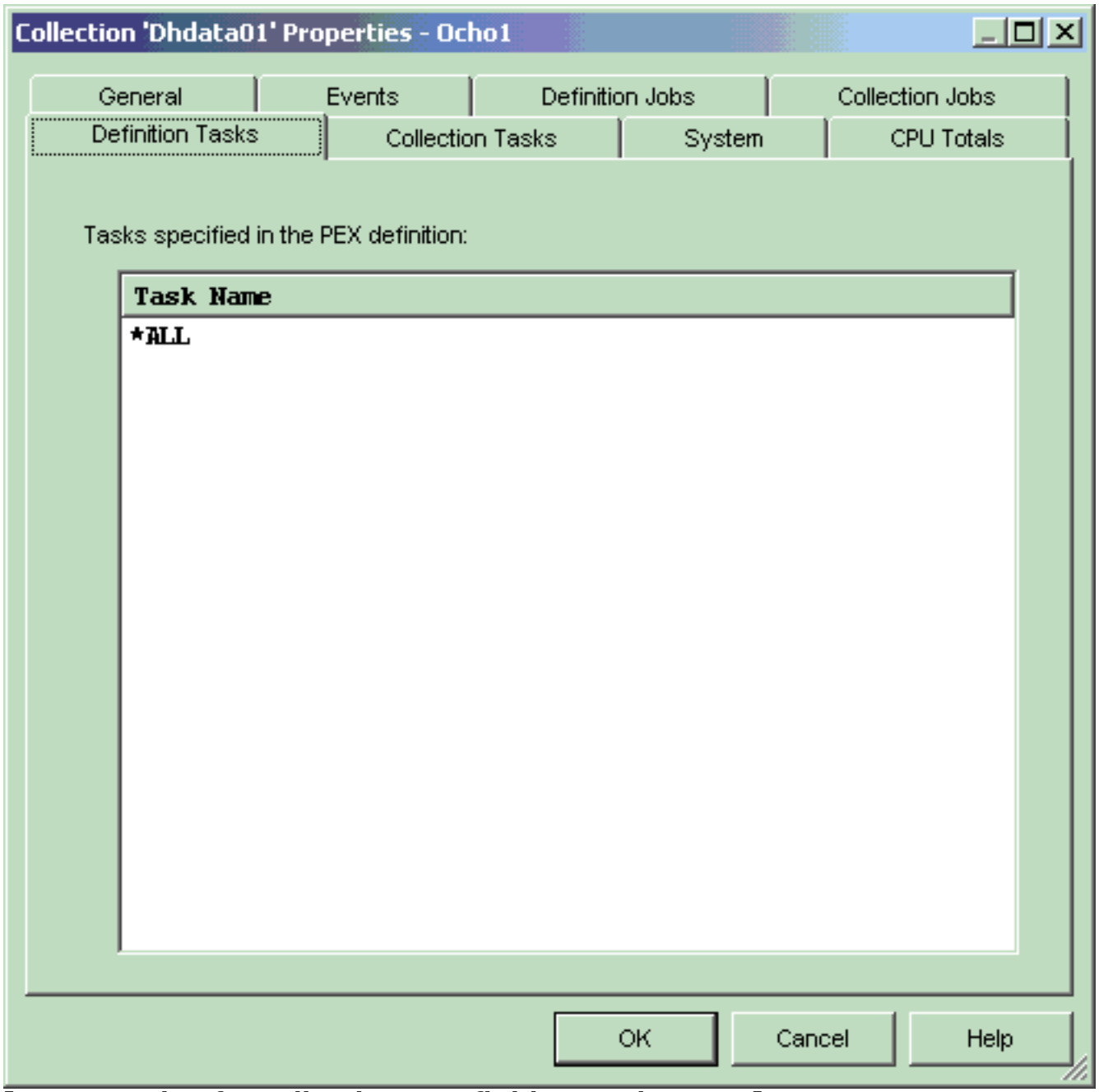
The following information is displayed on the Collection Jobs property page:

Field Name	Field Description
Total jobs	The total number of jobs in the collection.
Total threads	The total number of threads in the collection.
Jobs in collection list	A list of all jobs included in the collection. Due to the potential for huge amounts of data, a menu is available by right-clicking on the list providing additional features. The popup menu offers copy and paste to the clipboard, customizable font, and search capabilities.

Table of Contents | Previous | Next

4.3.7.5 Definition Tasks

The Definition Tasks property page provides information about the tasks the PEX definition was set up to capture.



[An example of a collection's Definition Tasks Page]

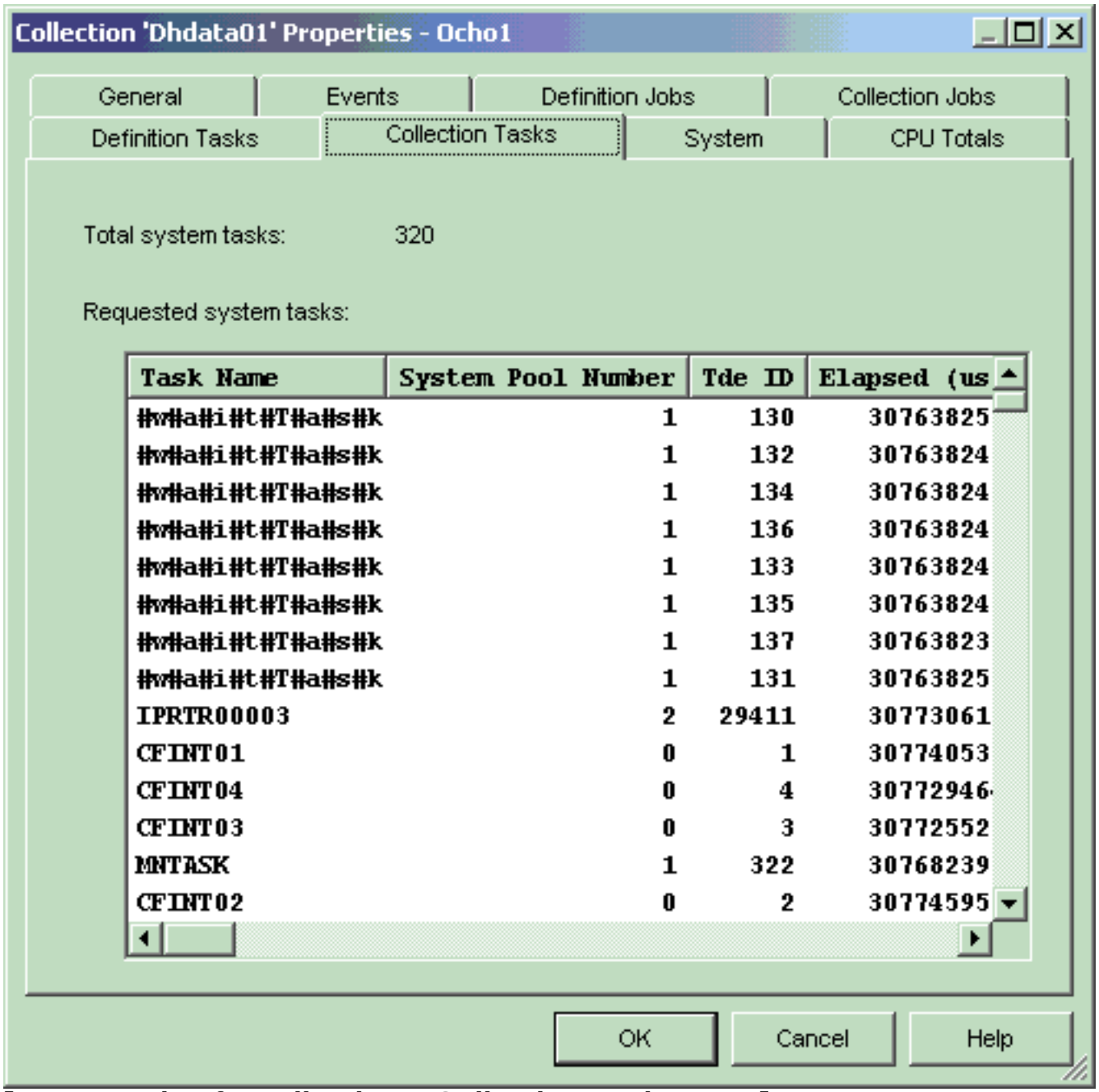
The following information is displayed on the Definition Tasks property page:

Field Name	Field Description
Tasks specified in the PEX definition list	A list of all tasks requested to be included in the collection by the PEX definition.

Table of Contents | Previous | Next

4.3.7.6 Collection Tasks

The Collection Tasks property page lists every task that was captured in the PEX collection. This page also lists the total number of system tasks captured.



[An example of a collection's Collection Tasks Page]

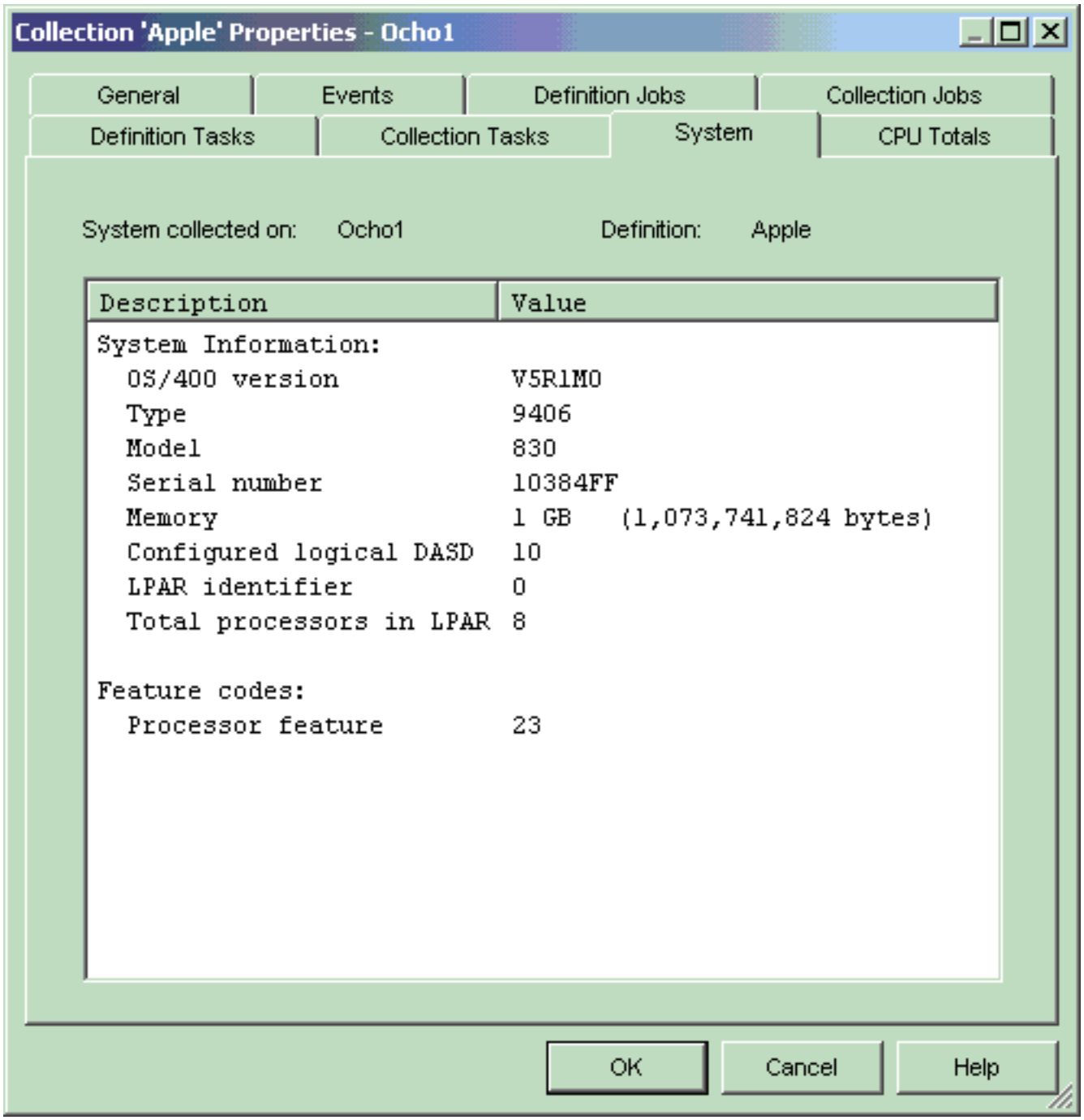
The following information is displayed on the Collection Tasks property page:

Field Name	Field Description
Total system tasks	The total number of system tasks included in the collection. Tasks named CFINT01...CFINTnn where nn equals the number of processors on the system are not included in this total.
Requested system tasks list	A list of all tasks included in the collection. You will always get tasks of name CFINT01...CFINTnn where nn equals number of processors on the system. Due to the potential for huge amounts of data, a menu is available by right-clicking on the list providing additional features. The popup menu offers copy and paste to the clipboard, customizable font, and search capabilities.

Table of Contents | Previous | Next

4.3.7.7 System Page

The System property page provides basic information about the system the collection was created on.



[An example of a collection's System Property Page]

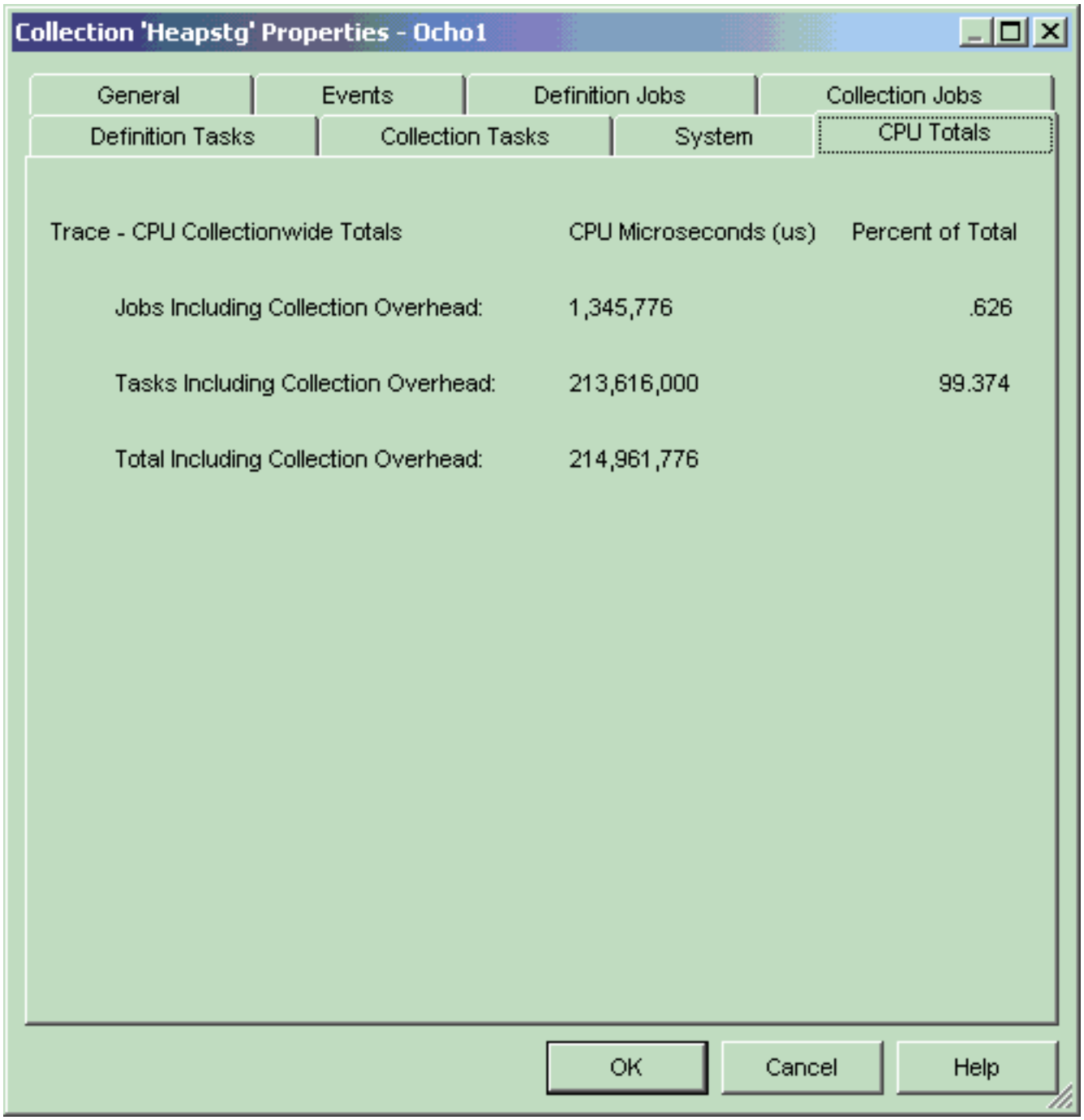
The following information is displayed on the System property page:

Field Name	Field Description
System collected on	iSeries the data was collected on.
OS/400 Version	Version/Release/Mod Level of OS/400.
Type	iSeries type
Model	iSeries model
Serial Number	Unique system serial number identifier for the iSeries.
Memory	Total system memory.
Configured logical DASD	Total number of operational DASD.
Total processors in LPAR	Total number of CPU processors.
LPAR Identifier	Logical Partition identifier. A value of 0 indicates that this is not an LPAR system.

Table of Contents | Previous | Next

4.3.7.8 Trace CPU Totals

The Trace CPU Totals property page provides information about summarized CPU for a Trace collection.



[An example of a collection's Trace CPU Property Page]

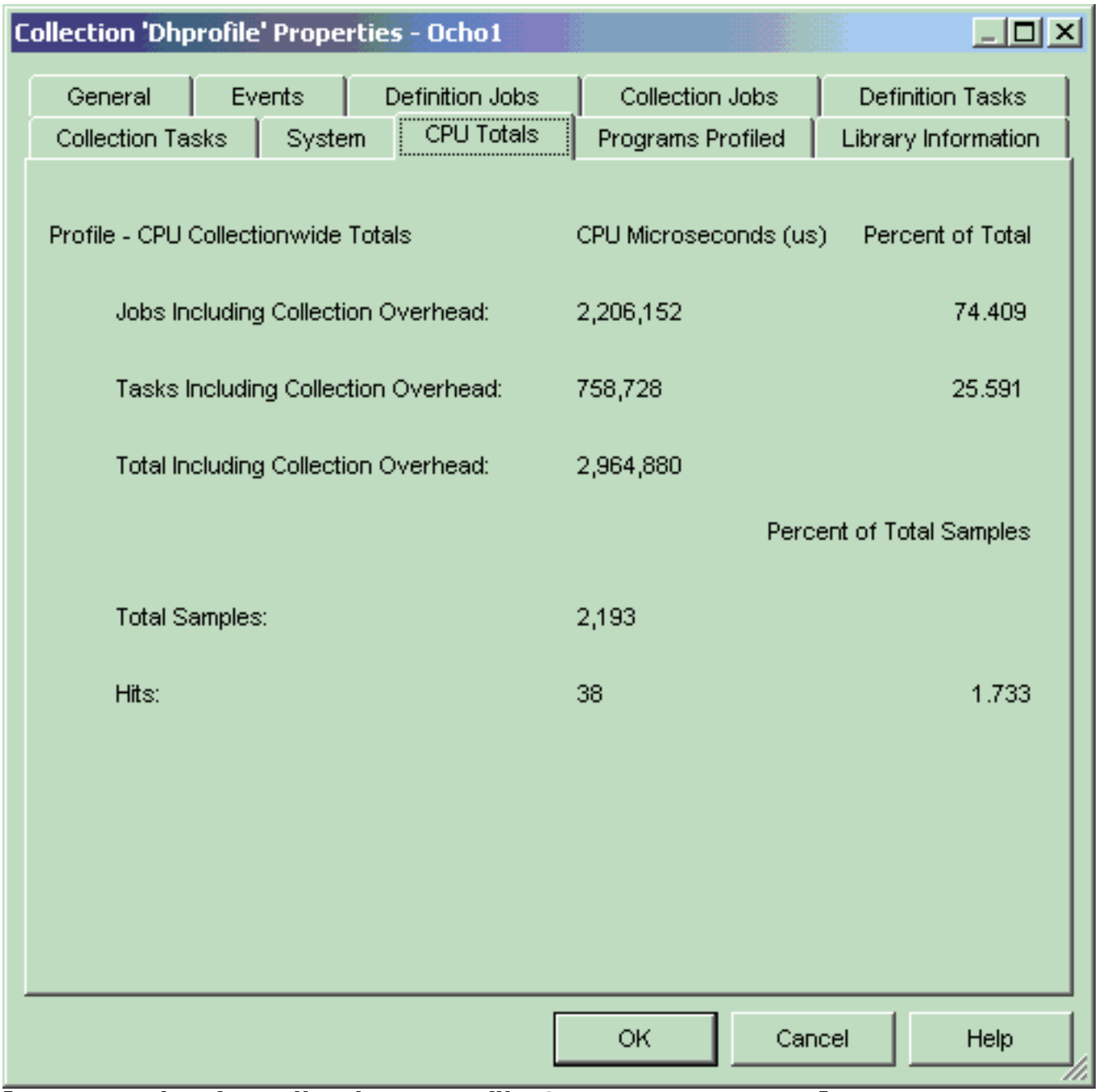
The following information is displayed on the Trace CPU Totals property page:

Field Name	Field Description
Jobs including collection overhead (us)	Total number of CPU microseconds including pex collection overhead for all jobs in the collection.
Jobs including collection overhead (% of total)	Percentage of Total CPU used by jobs in the collection.
Tasks including collection overhead (us)	Total number of CPU microseconds including pex collection overhead for all tasks in the collection.
Tasks including collection overhead (% of total)	Percentage of Total CPU used by tasks in the collection.
Total including collection overhead	The total CPU microseconds for the collection including collection overhead.

Table of Contents | Previous | Next

4.3.7.9 Profile CPU Totals

The Profile CPU Totals property page provides information about summarized CPU for a Profile collection.



[An example of a collection's Profile CPU Property Page]

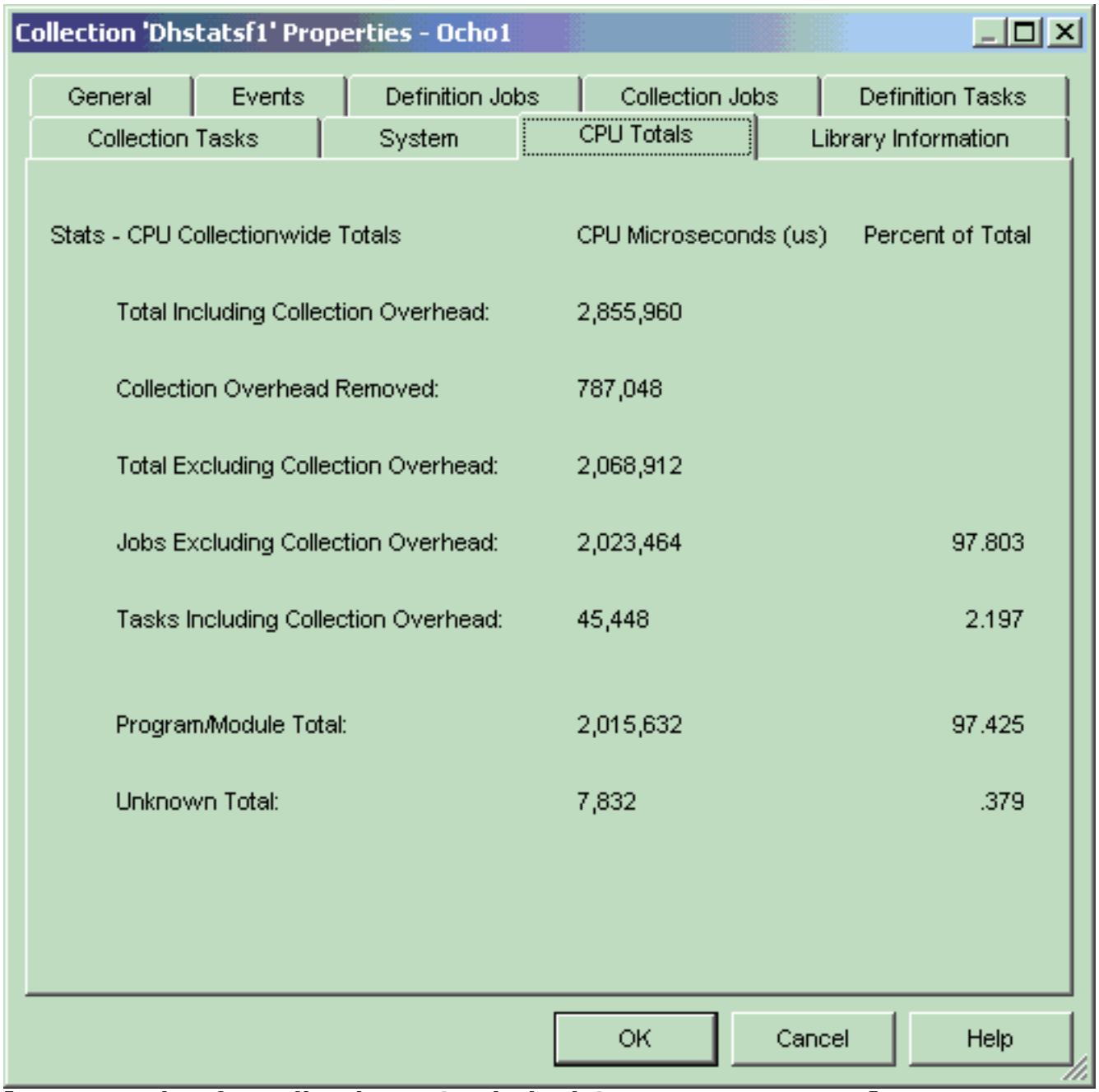
The following information is displayed on the Profile CPU Totals property page:

Field Name	Field Description
Jobs including collection overhead (us)	Total number of CPU microseconds including pex collection overhead for all jobs in the collection.
Jobs including collection overhead (% of total)	Percentage of Total CPU used by jobs in the collection.
Tasks including collection overhead (us)	Total number of CPU microseconds including pex collection overhead for all tasks in the collection.
Tasks including collection overhead (% of total)	Percentage of Total CPU used by tasks in the collection.
Total including collection overhead	The total CPU microseconds for the collection including collection overhead.
Total samples	Total sample count.
Hits	Hit count from the total samples.
Hits (% of total samples)	Percentage of hit CPU microseconds used over the total samples in the collection.

Table of Contents | Previous | Next

4.3.7.10 Statistical CPU Totals

The Stats CPU Totals property page provides information about summarized CPU for a Statistical collection.



[An example of a collection's Statistical CPU Property Page]

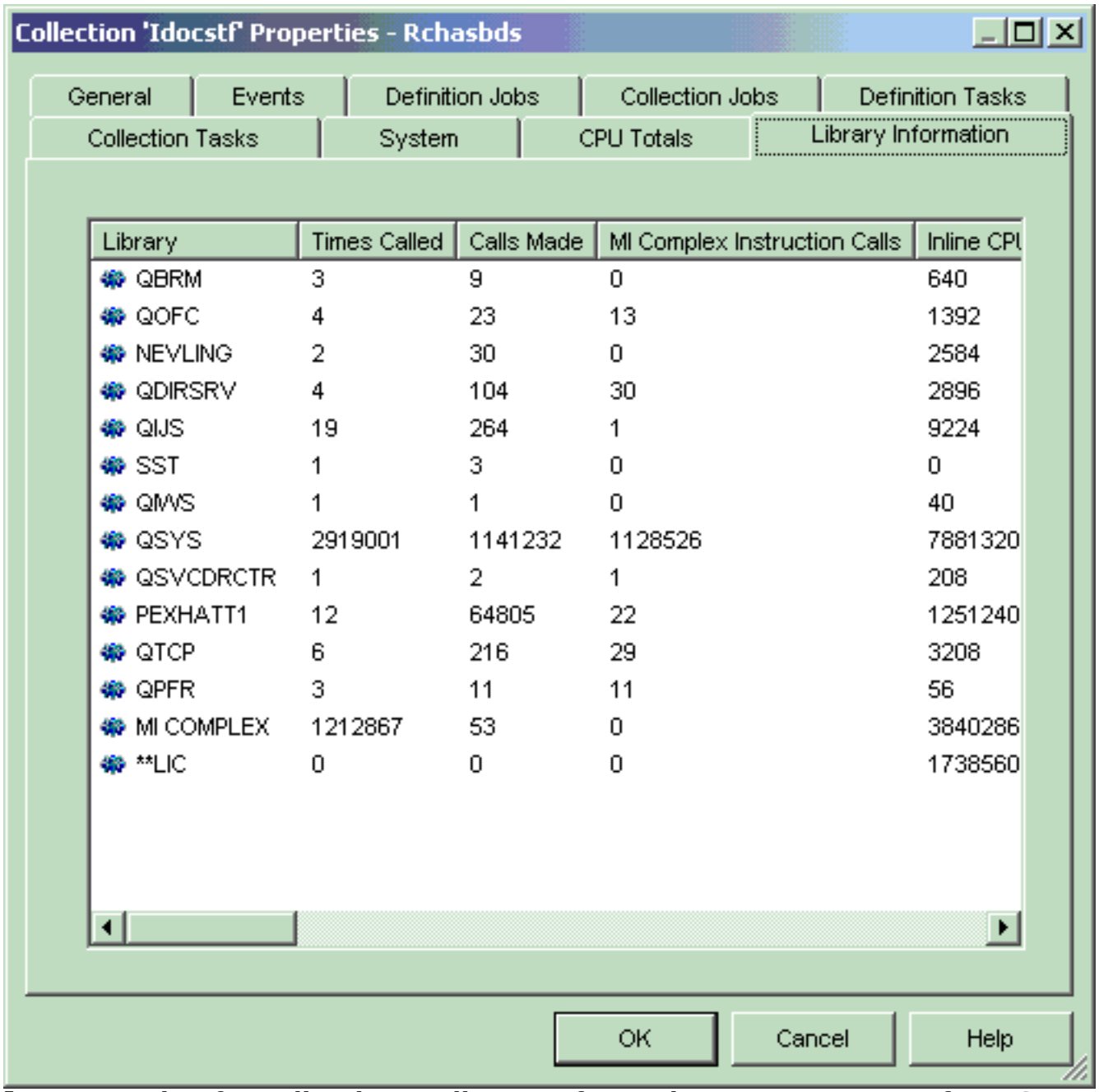
The following information is displayed on the Stats CPU Totals property page:

Field Name	Field Description
Total including collection overhead (us)	The total CPU microseconds for the collection including collection overhead.
Collection overhead removed (us)	The total removed collection overhead in CPU microseconds.
Total excluding collection overhead (us)	The total CPU microseconds for the collection excluding collection overhead.
Jobs excluding collection overhead (us)	Total number of CPU microseconds excluding PEX collection overhead for all jobs in the collection.
Jobs excluding collection overhead (% of total)	Percentage of Total CPU excluding collection overhead used by jobs in the collection.
Tasks excluding collection overhead (us)	Total number of CPU microseconds excluding PEX collection overhead for all tasks in the collection.
Tasks excluding collection overhead (% of total)	Percentage of Total CPU excluding collection overhead used by tasks in the collection.
Program/Module (us)	Total number of CPU microseconds used by programs and modules.
Program/Module (% of total)	Percentage of CPU microseconds devoted to program/module activity of the total CPU.
Unknown (us)	Total number of CPU microseconds which cannot be attributed to specific programs or modules.
Unknown (% of total)	Percentage of CPU microseconds devoted to unknown activity of the total CPU.

Table of Contents | Previous | Next

4.3.7.11 Library Information

The Library Information property page provides information about the libraries in the PEX collection. This page is only shown for collections of type Statistical or Profile.



[An example of a collection's Library Information Property Page for a Stats type collection]

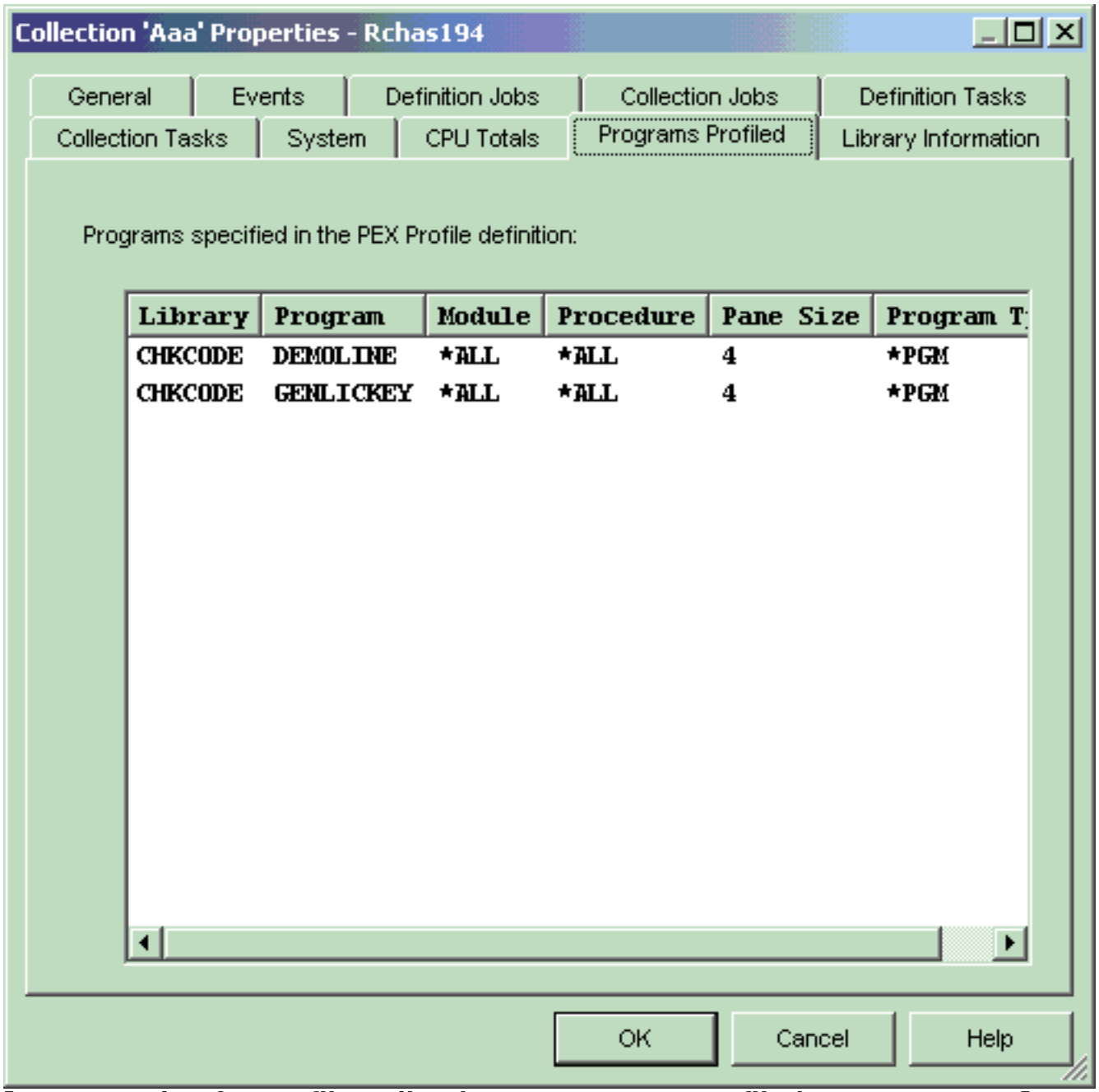
The following information is displayed on the Library Information property page:

Field Name	Field Description
Library Information List	List of libraries with additional supporting information. The fields in the list will vary depending on the type of collection: Stats or Profile.

Table of Contents | Previous | Next

4.3.7.12 Programs Profiled

The Programs Profiled property page provides information about the programs included in the PEX definition for a PEX Profile collection.



[An example of a Profile collection's Programs Profiled Property Page]

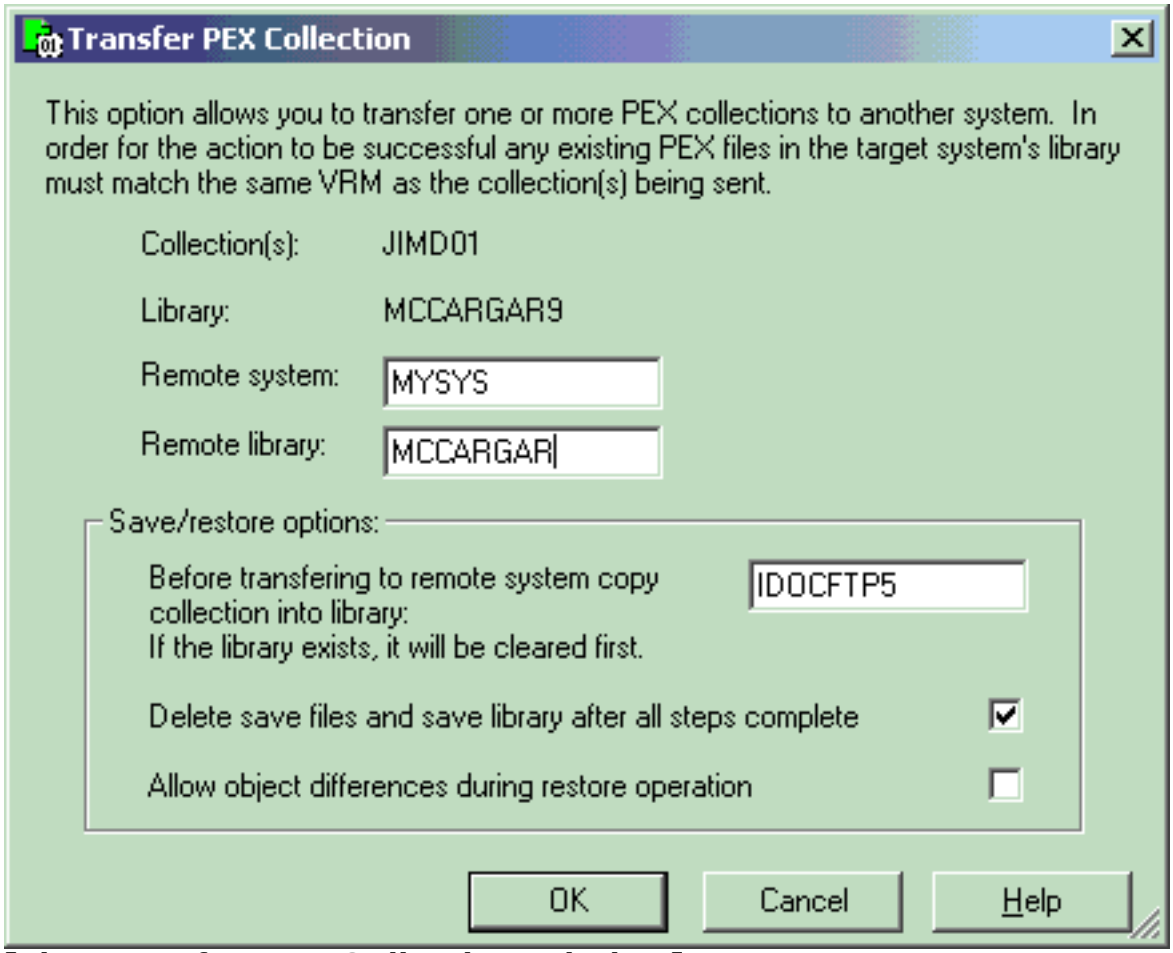
Name	Description
Programs Profiled List	List of programs and/or service programs from the PEX definition used for this collection.

Table of Contents | Previous | Next

4.3.8 Transferring to another system

A user may move one or more PEX collections to another system using PEX Analyzer. Access this feature by selecting one or more collections, right-clicking and choosing the 'Transfer to...' menu.

Moving PEX collections can be very time saving because it eliminates several manual steps that would normally be required. Transferring a collection is accomplished using a series of remote command to save the PEX data to a save file, FTP it to the remote system and the restore the PEX data to the remote library specified.



[The Transfer PEX Collection Window]

The options available on this page are summarized in the following table:

Field	Description
Collection(s)	Lists the collection(s) that are to be transferred from the current system.
Library	The library on the current system the collection(s) will be transferred from.
Remote system	The remote system name or IP address to send the collection(s) to.
Remote library	The library on the remote system to send the collection(s) to.
Save library	The library on the current system the collection(s) should first be saved to in order to create the save file.
Delete save files and save library after all steps complete option	Check this box to get rid of the objects specified. This option is useful if you are transferring a very large collection and you want to make sure the save file doesn't get deleted in case the FTP or some other step fails during the process.
Allow object differences option	By checking this option the ALWOBJDIF parameter on the RSTLIB command will be specified as *ALL. Otherwise this value is set to *NONE. Note: If you check this option and the collection saved has a different file format for the PEX files than the files in the remote library, you will end up with your existing collections renamed via OS/400 by the RSTLIB command and they will no longer be visible to PEX Analyzer.

Table of Contents | Previous | Next

4.4 PEX Definitions

A **PEX definition** is a member in a specific system database (QUSRSYS/QAPEXDFN), and it controls most of the aspects of making a PEX collection except the following:

- When to begin making the collection
- When to end the collection
- The library where collection data is stored

A PEX definition also controls:

- Which subset of the hundreds of event types are to be activated
- Granularity of CPU sampling
- Maximum amount of data to be collected
- Subset of jobs or system tasks (or all jobs and all tasks) that are to be traced

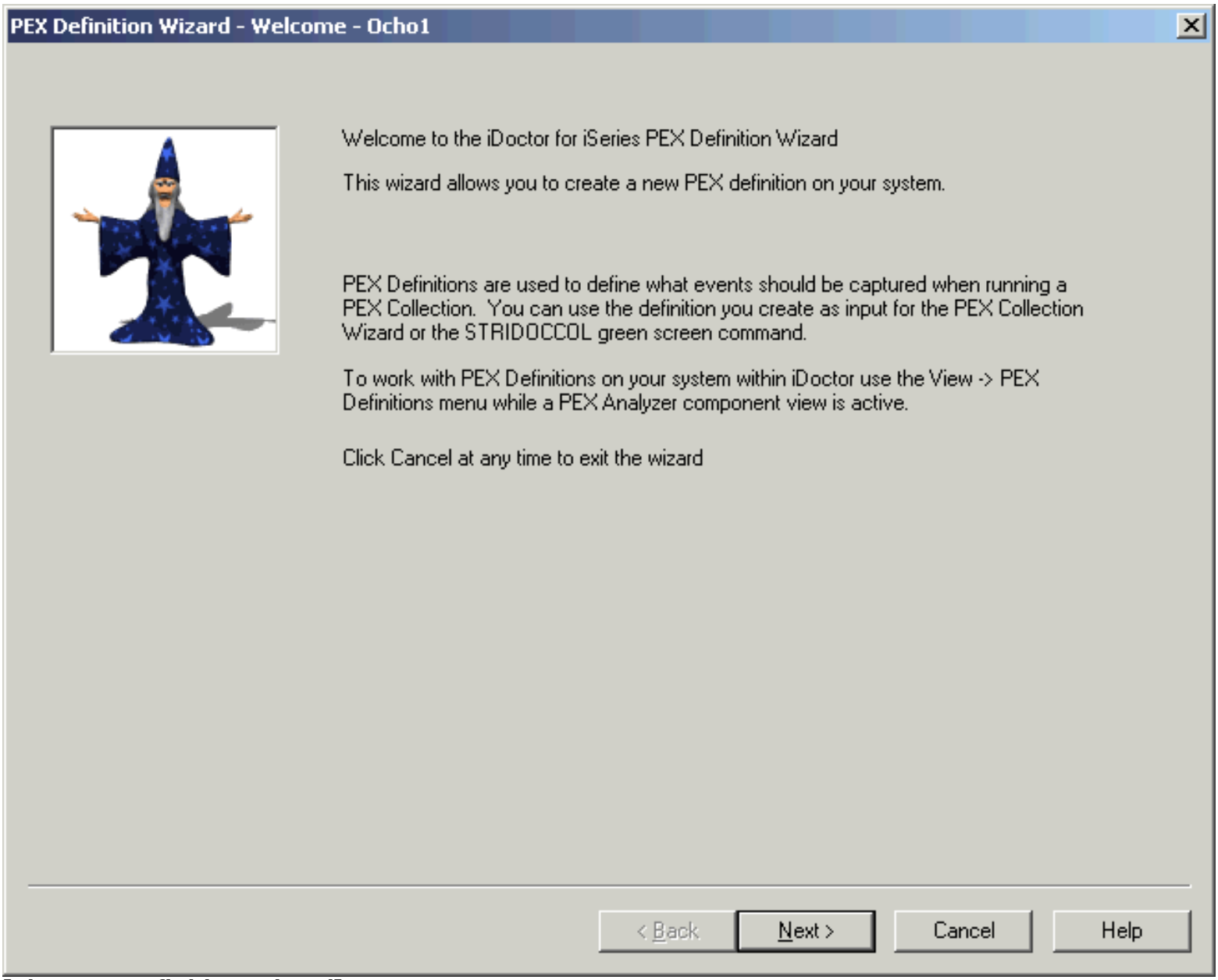
To create or change a PEX definition, use the PEX Definition Wizard discussed within this section.

Table of Contents | Previous | Next

4.4.1 Creating or Changing - The PEX Definition Wizard

Within the iDoctor for iSeries client you can create new PEX definitions or easily change an existing PEX definition using the PEX Definition Wizard. You can access this wizard via the PEX Definitions view.

The PEX Definition Wizard is a full-featured interface over the ADDPEXDFN command.

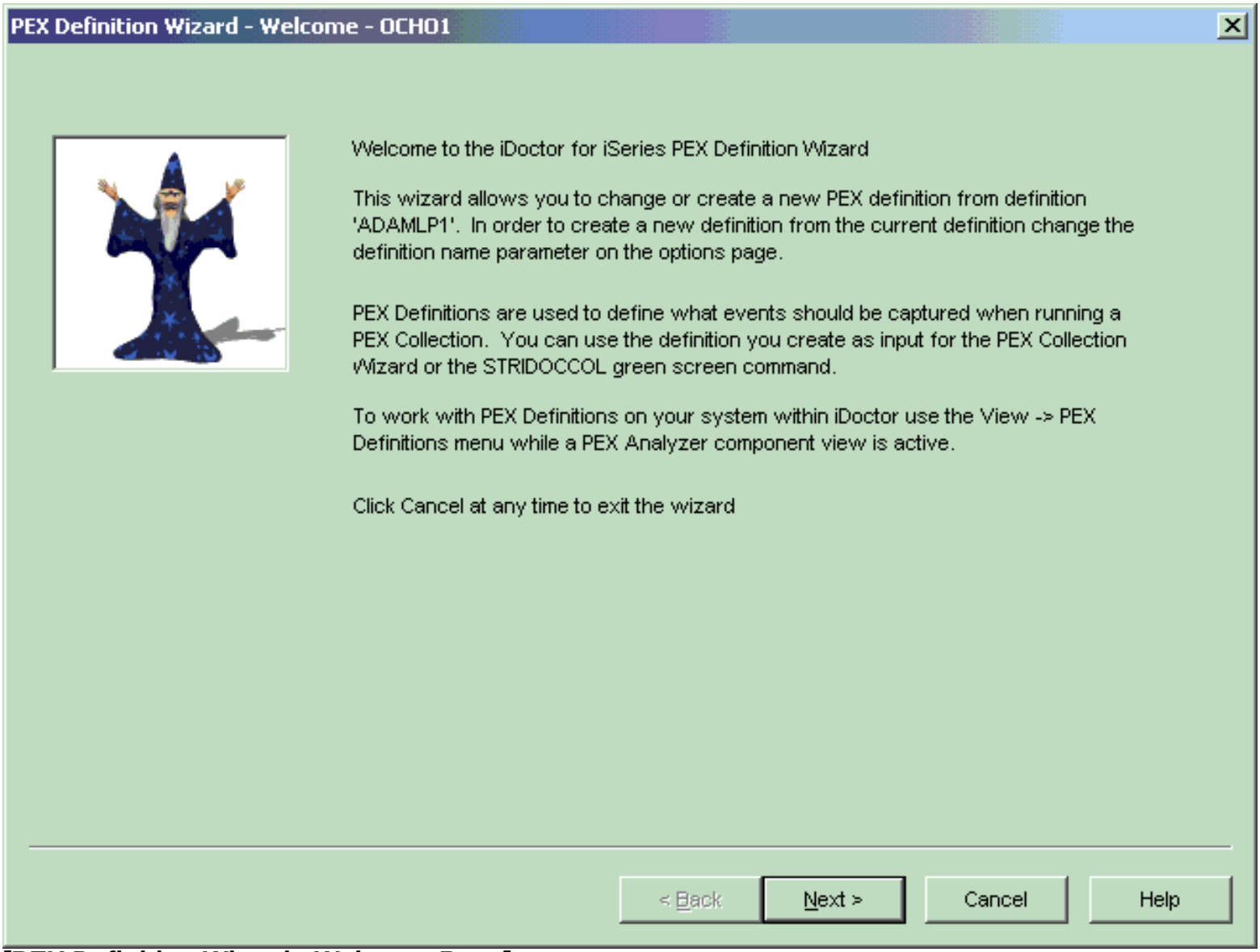


[The PEX Definition Wizard]

Table of Contents | Previous | Next

4.4.1.1 Welcome

The Welcome page in the PEX Definition Wizard introduces the user to the wizard and offers information about what the wizard will do. When changing an existing PEX definition all of the current values will be prefilled into the appropriate places within the PEX Definition interface.

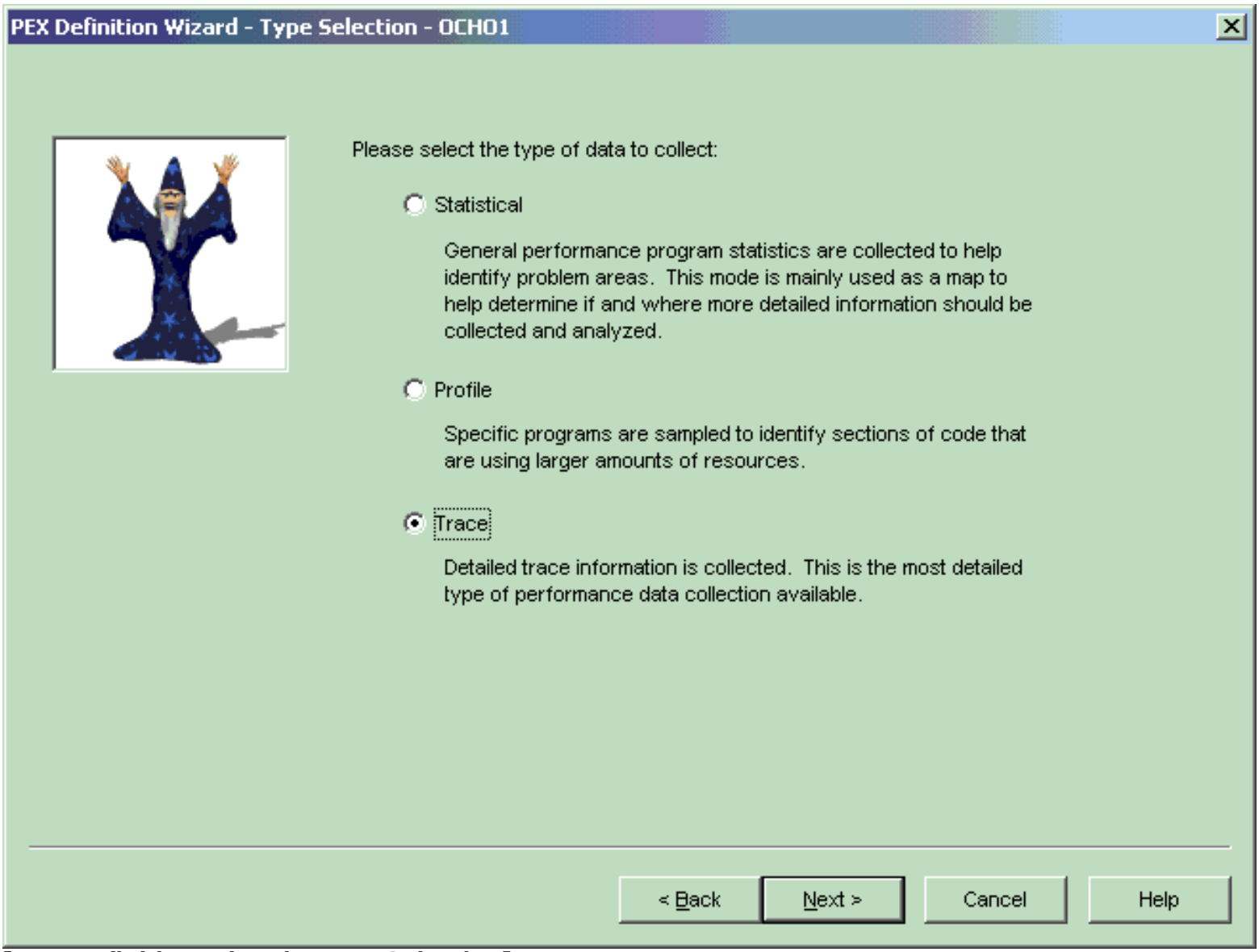


[PEX Definition Wizard - Welcome Page]

Table of Contents | Previous | Next

4.4.1.2 Type Selection

The Type Selection page in the PEX Definition Wizard lets a user decide the most important characteristic about the definition, its type. The value choosen (Statistical, Profile, or Trace), influences whats seen on the following pages. For example, several options that are only available for trace collections will only be shown if Trace is selected.



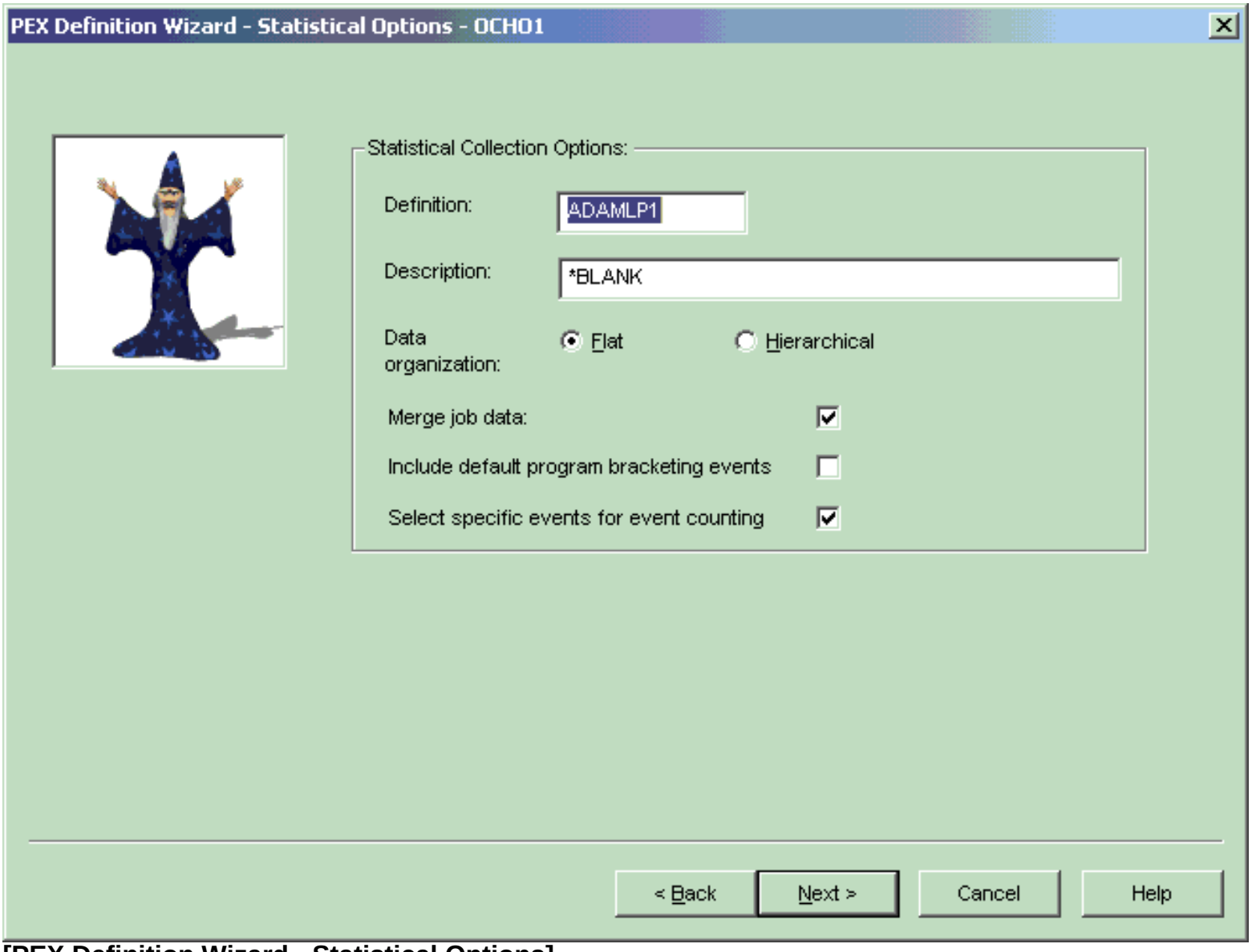
[PEX Definition Wizard - Type Selection]

Table of Contents | Previous | Next

4.4.1.3 Options

The Options page in the PEX Definition Wizard lets the user decide the most basic parameters for the definition. This page is different depending on the type selection parameter specified on the previous page.

An example of the Statistical Options Page is shown below:



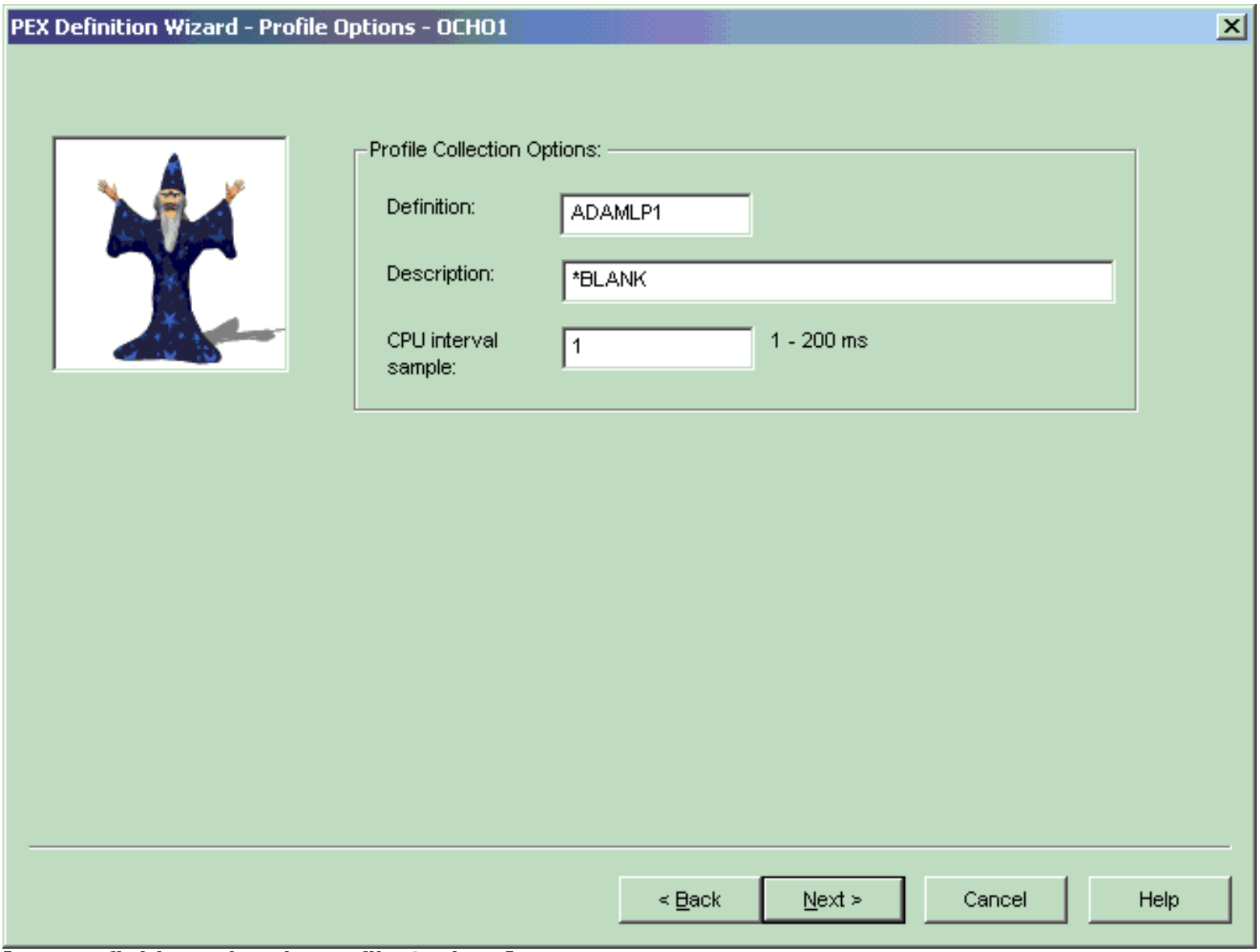
[PEX Definition Wizard - Statistical Options]

This table defines the inputs available on this page:

Field	Description

Definition	Name of the PEX definition. This value matches the member name of the definition stored in the QUSRSYS/QAPEXDFN file.
Description	Specifies the text that briefly describes the PEX definition. The possible values are: *BLANK Text is not specified. 'description' Specify no more than 50 characters of text**.**
Data Organization	Specifies how the data is organized. The possible values are: Flat The performance explorer tool will not collect data for a parent-child relationship. Hierarchical The performance explorer tool will collect data for a parent-child relationship.
Merge Job Data	Specifies if data from different jobs should be merged together or not. This parameter is only available if the Data Organization parameter value is Flat.
Include default program bracketing events	Determines whether or not to include default program bracketing events. Leave this box unchecked if you want to specify other program bracketing events.
Select specific events for counting	Statistical definitions allow you to define event counters which tell you within the Statistical data the total number of events that occurred within each counter bucket.

An example of the Profile Options Page is shown below:



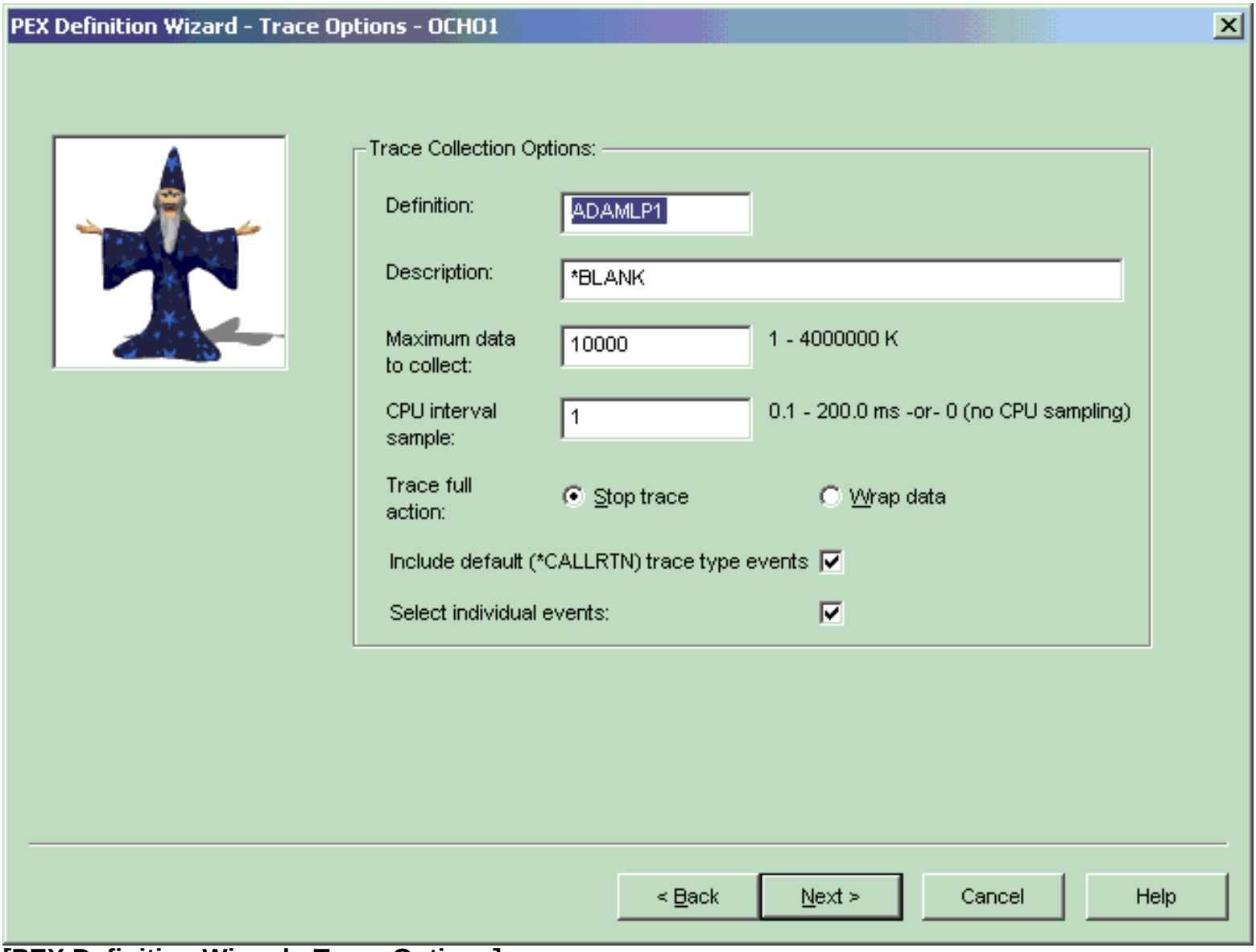
[PEX Definition Wizard - Profile Options]

This table defines the inputs available on this page:

Field	Description
Definition	Name of the PEX definition. This value matches the member name of the definition stored in the QUSRSYS/QAPEXDFN file.
Description	Specifies the text that briefly describes the PEX definition. The possible values are: *BLANK Text is not specified. 'description' Specify no more than 50 characters of text**.**

CPU Interval Sample	Specifies the size of the interval which CPU samples are taken of the program. A low interval will cause a high number of samples to be taken, and will also cause higher overhead. A low interval will also provide relatively more data. This parameter will be grayed out if it does not apply to the selected iDoctor-supplied definition.

An example of the Trace Options Page is shown below:



[PEX Definition Wizard - Trace Options]

This table defines the inputs available on this page:

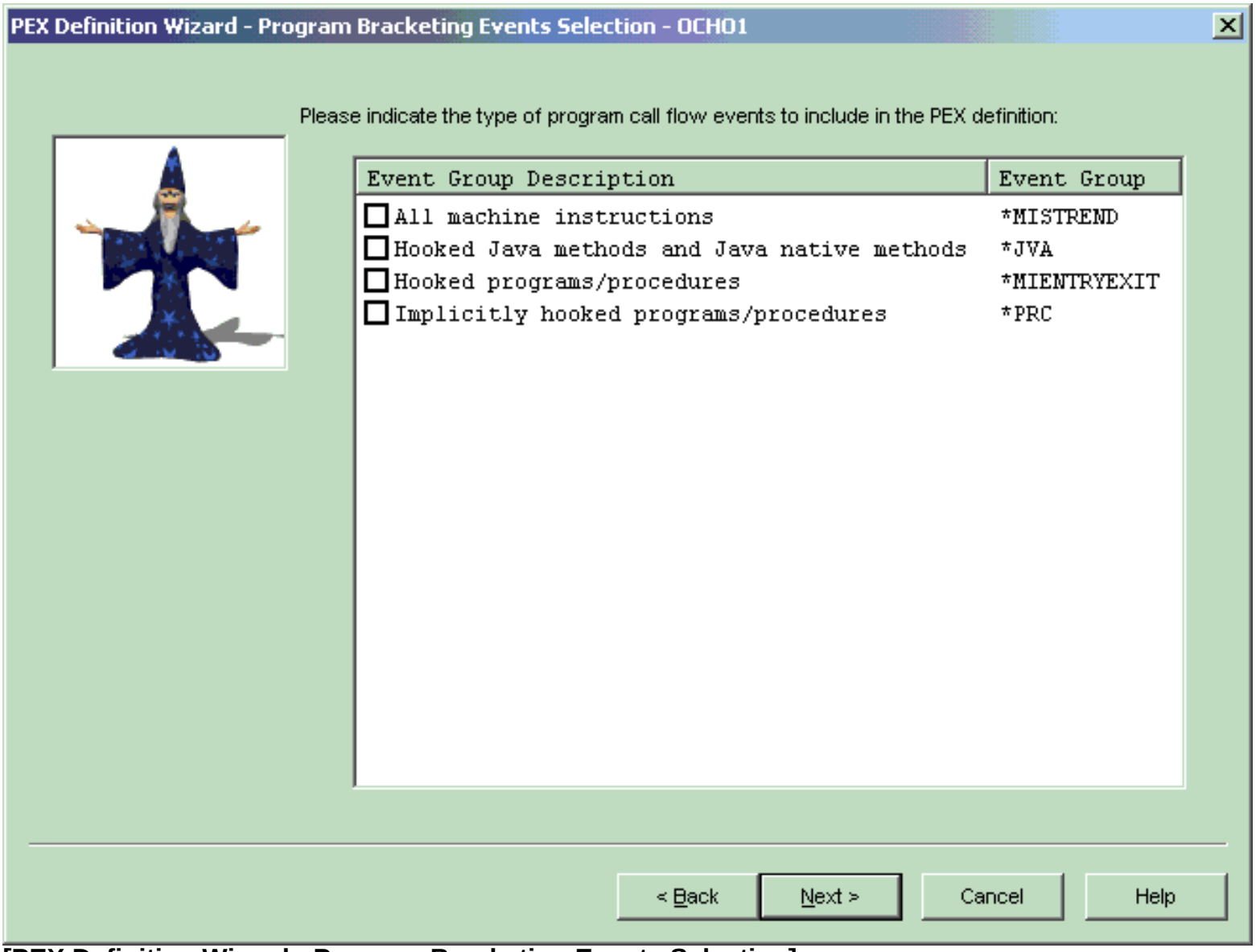
Field	Description
Definition	Name of the PEX definition. This value matches the member name of the definition stored in the QUSRSYS/QAPEXDFN file.

Description	Specifies the text that briefly describes the PEX definition. The possible values are: *BLANK Text is not specified. 'description' Specify no more than 50 characters of text**.**
Data Organization	Specifies how the data is organized. The possible values are: Flat The performance explorer tool will not collect data for a parent-child relationship. Hierarchical The performance explorer tool will collect data for a parent-child relationship.
Maximum data to collect	The maximum amount of disk space this collection should use in kilobytes. The default value is 500,000. When using a user-defined PEX definition this parameter is ignored because it is provided within the PEX definition.
CPU interval sample	Specifies the size of the interval which CPU samples are taken of the program. A low interval will cause a high number of samples to be taken, and will also cause higher overhead. A low interval will also provide relatively more data. This parameter will be grayed out if it does not apply to the selected iDoctor-supplied definition.
Trace full action	The action to take if the maximum data to collect value is reached. The choices are to either suspend/stop the collection or wrap the data. If the data is wrapped the oldest trace records will be overwritten with the newest ones.
Include *CALLRTN trace type events	This option indicates whether or not you wish to specify event groups to collect This is an interface over the TRCTYPE parameter.
Select individual events	Use this option to select individual events to collect data for.

Table of Contents | Previous | Next

4.4.1.4 Program Bracketing Events Selection

This page lets you decide which program call flow events are included in the type Statistical definition. Check the box next to each type of program call flow events to include them in the definition.



[PEX Definition Wizard - Program Bracketing Events Selection]

The following table summarizes the possible program bracketing event groups:

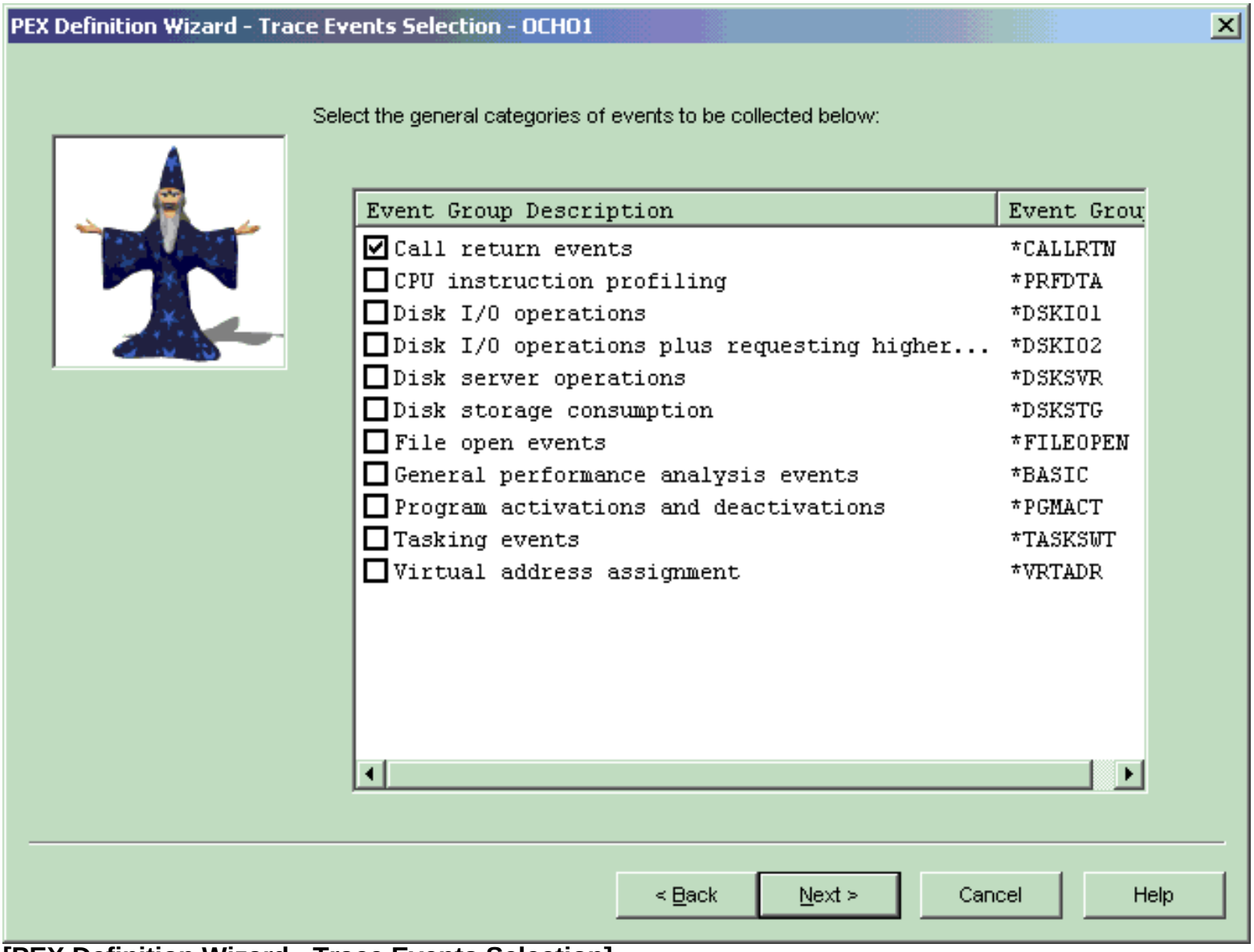
Event Group	Description
*MISTREND	Statistics are to be collected on all machine instructions.

*MIENTRYEXIT	Statistics are to be collected on programs and procedures that have been explicitly hooked via the ENBPFRCOL parameter on the various compile and change job commands.
*JVA	Statistics are to be collected on Java methods and Java native methods that have been explicitly hooked via the ENBPFRCOL parameter (or its equivalents) on the Java and JIT compile commands.
*PRC	Statistics are to be collected on programs and procedures that have been implicitly hooked. This includes any program that has been compiled at optimization level 30 or below. Optimization level 40 programs require explicit compiler options which activate the trace job (trcjob) style hooks. *MIENTRYEXIT and *PRC are mutually exclusive.

Table of Contents | Previous | Next

4.4.1.5 Trace Events Selection

This page lets you decide which trace type event groups to include in the Trace definition. Check the box next to each event group to include it in the definition.



[PEX Definition Wizard - Trace Events Selection]

The following table summarizes the possible trace event groups:

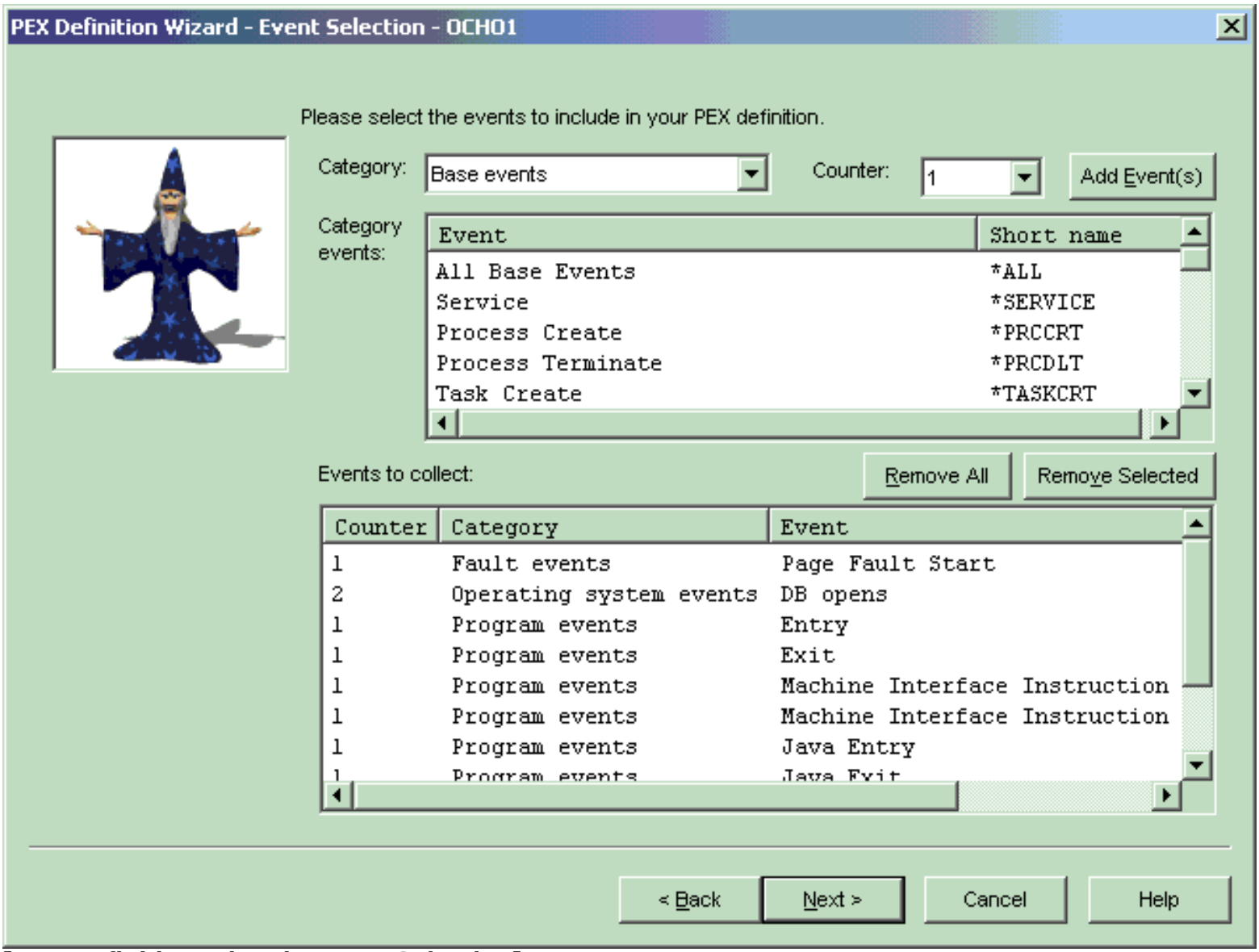
Event Group	Description

*CALLRTN	Specifies that call return events are included in the trace definition. Call return events occur when a program is entered and exited as well as when certain machine instruction are started and completed.
*BASIC	Specifies that events relative to general performance analysis are included in the trace definition. This option should be used when it is unclear as to what type of performance problem determination is necessary.
*DSKIO1	Specifies that events associated with disk input/output operations are included in the trace definition.
*DSKIO2	Specifies that events associated with disk input/output operations plus higher level requests to do input/output operations are included in the trace definition.
*DSKSVR	Specifies that events associated with disk server operations are included in the trace definition.
*DSKSTG	Specifies that events associated with disk storage consumption are included in the trace definition.
*VRTADR	Specifies that events associated with virtual address assignment are included in the trace definition.
*PGMACT	Specifies that events associated with program activations and deactivations are included in the trace definition.
*FILEOPEN	Specifies that events associated with file (*FILE) opens are included in the trace definition.
*PRFDTA	Specifies that events associated CPU instruction profiling are included in the trace definition.
*TASKSWT	Specifies that events associated with tasking are included in the trace definition.

Table of Contents | Previous | Next

4.4.1.6 Event Selection

This page lets you decide which specifc events to include in the definition. This window is available when defining a Statistical or Trace definition. When defining a Statistical definition there is a counter field which lets you assign events to specific counter buckets. The purpose of the counter bucket is to provide the total occurrences of all the events specified in each bucket.



[PEX Definition Wizard - Event Selection]

The following table summarizes the inputs available on this window:

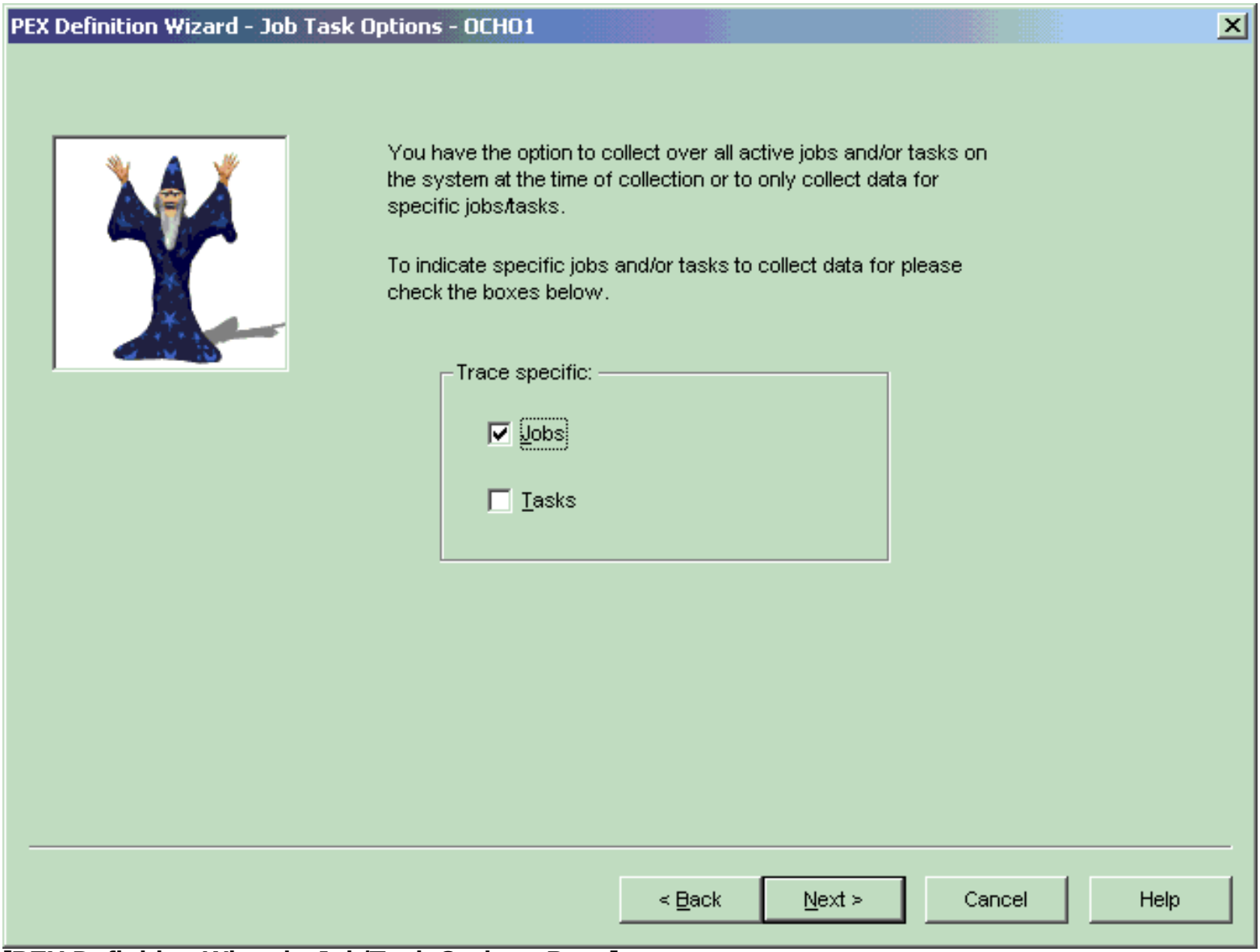
Field	Description

Category	This drop down lists contains the possible categories of events. Change the value and the list of events will be update to show the events within the selected category.
Counter	This drop down lists contains counter buckets to select from. The selected counter is used when adding the selected category events to the Events to collect list. This field is only visible when defining a statistical collection.
Category Events list	Displays the list of events to select from within the selected category.
Add Events button	Adds the selected events from the category events list to the events to collect list.
Events to collect list	Displays the current events to collect for this PEX definition. This list shows each specific category and event name to be collected.
Remove All button	Clears the Events to collect list.
Remove Selected button	Removes the selected events from the Events to collect list.

Table of Contents | Previous | Next

4.4.1.7 Job/Task Options

On this page you may decide if you would like to select specific jobs or tasks to include in the PEX definition. Selecting specific jobs and tasks is optional, but is necessary when you only want to collect data for the job(s) or task(s) you are interested in.

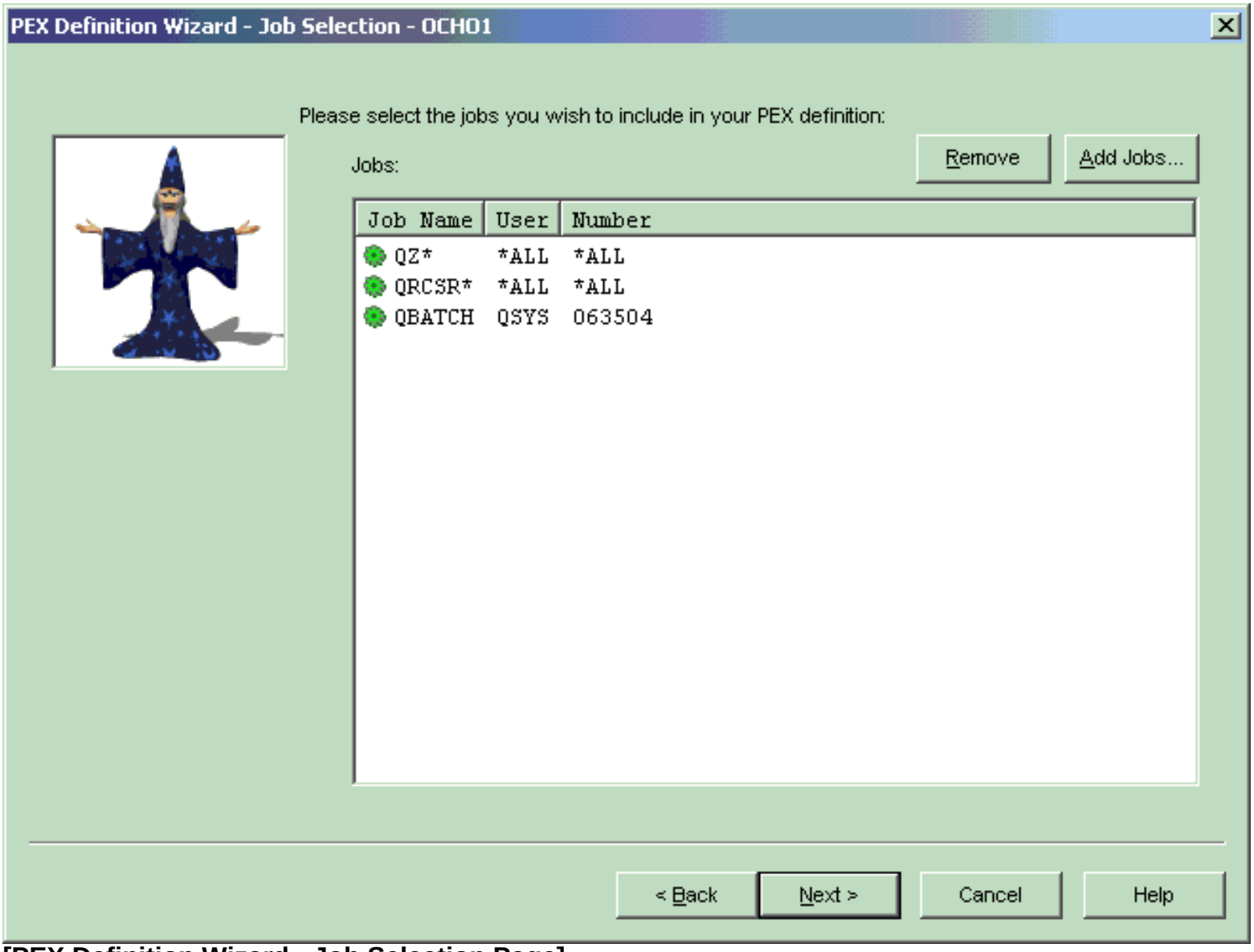


[PEX Definition Wizard - Job/Task Options Page]

Table of Contents | Previous | Next

4.4.1.8 Job Selection

The job selection page displays a list of selected job information to use in the PEX definition. There are also two buttons on this page used to add or remove jobs from the list.



[PEX Definition Wizard - Job Selection Page]

The table below summarizes the different elements on this page:

Field	Description
Jobs list	A list of jobs to collect information about in the PEX definition.

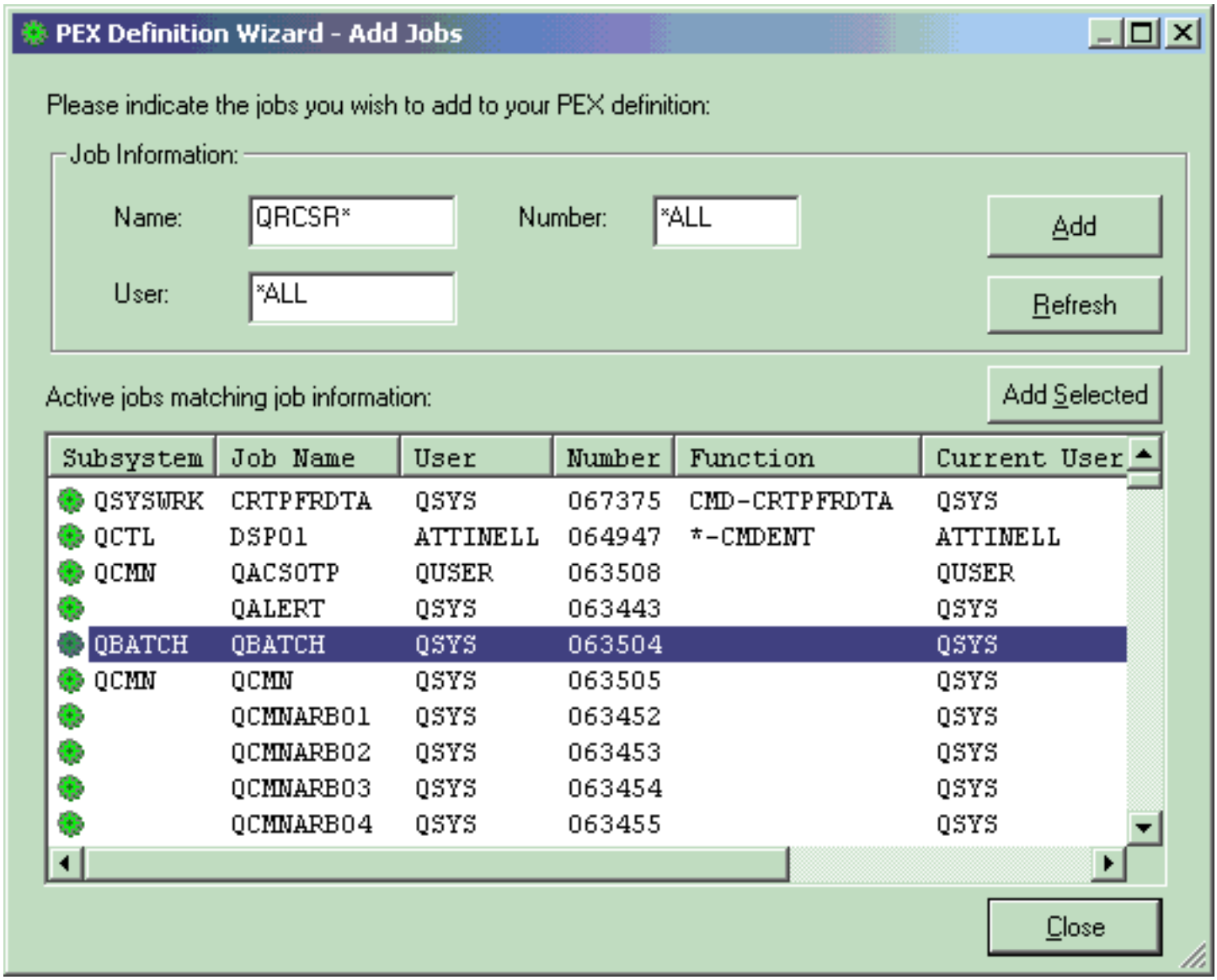
Remove button	This button removes the selected jobs from the list.
Add Jobs button	Use this button to open the Add Jobs Window (discussed in the next section). This window is used to select and add additional jobs to the list.

Table of Contents | Previous | Next

4.4.1.9 Add Jobs Window

The add jobs window allows a user to add jobs to the Job Selection page in the wizard. Job information can be of two types: generic job name/generic job user/generic job number -or- job name/job user/job number.

The "Job Information" portion of the window includes text fields used to define a generic job to add to the Job Selection Page or to use as a filter when refreshing the list of jobs shown in the window. The Add button will add the current generic job to the Job Selection page and the Add Selected button will add the selected jobs from the active jobs list to the Job Selection page.



[PEX Definition Wizard - Add Jobs Window]

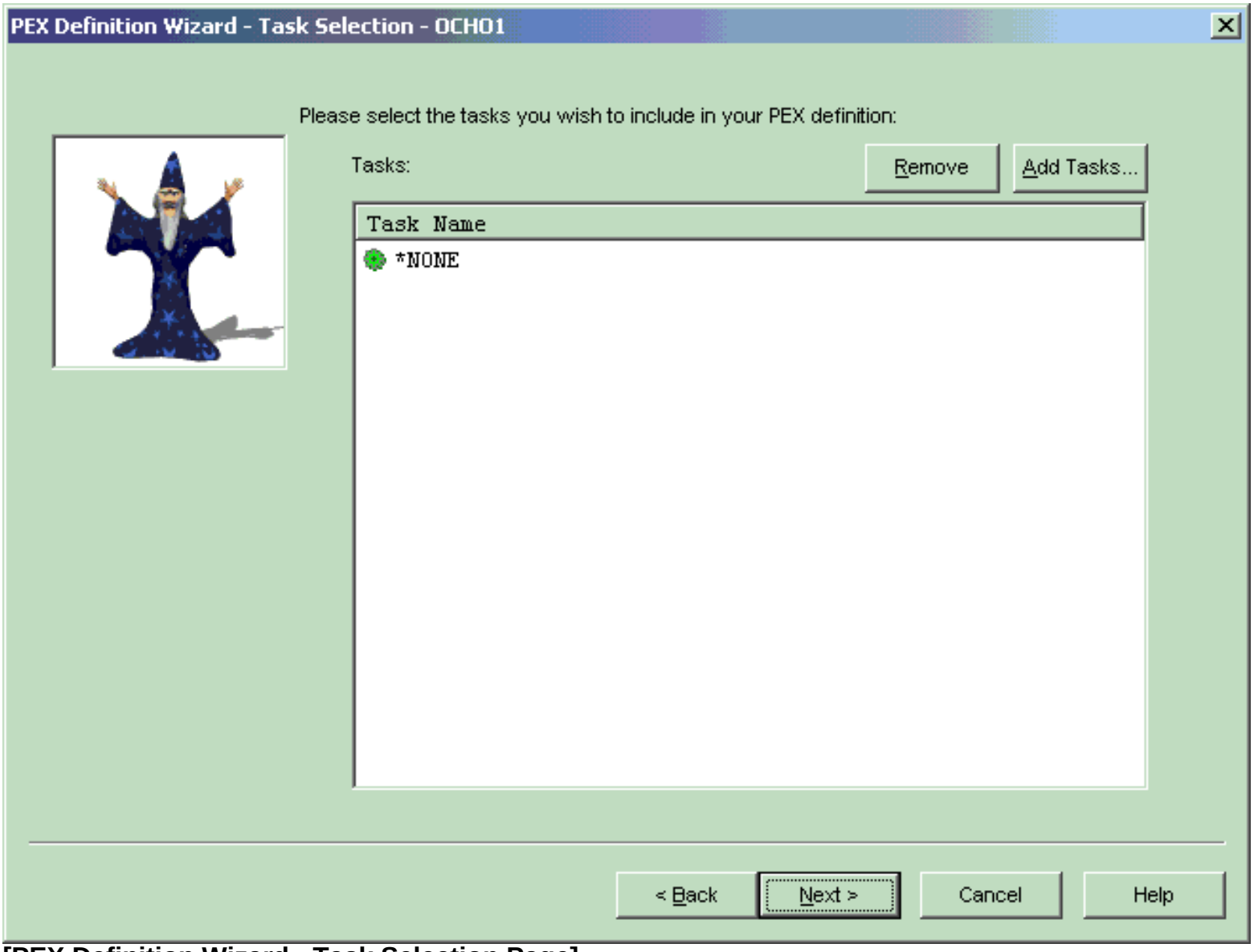
The table below summarizes the different elements on this page:

Field	Description
Name	A text field for entering a generic or specific job name. When specifying a generic name use a * at the end of the job name.
User	A text field for entering a generic or specific job user. When specifying a generic name use a * at the end of the job user name.
Number	A text field for entering a specific job number or *ALL.
Add button	This button will add the current job name/user/number values in the text fields to the Job Selection page. This can be used to add a generic job name/user/number value such as QZ*/MCCARGAR/*ALL This value indicates all job names starting with QZ, for job user MCCARGAR.
Refresh button	This button is used to refresh the active jobs list based on the current values specified in the name, user and number text fields.
Add Selected button	Use this button to add the selected active jobs to the Job Selection Page.
Active jobs matching job information list	This list shows all active jobs on the system matching the current Job information specified. When this window is first open the list will show all active jobs on the system.

Table of Contents | Previous | Next

4.4.1.10 Task Selection

The task selection page displays a list of selected tasks to include in the PEX definition. There are also two buttons on this page used to add or remove tasks from the list.



[PEX Definition Wizard - Task Selection Page]

The table below summarizes the different elements on this page:

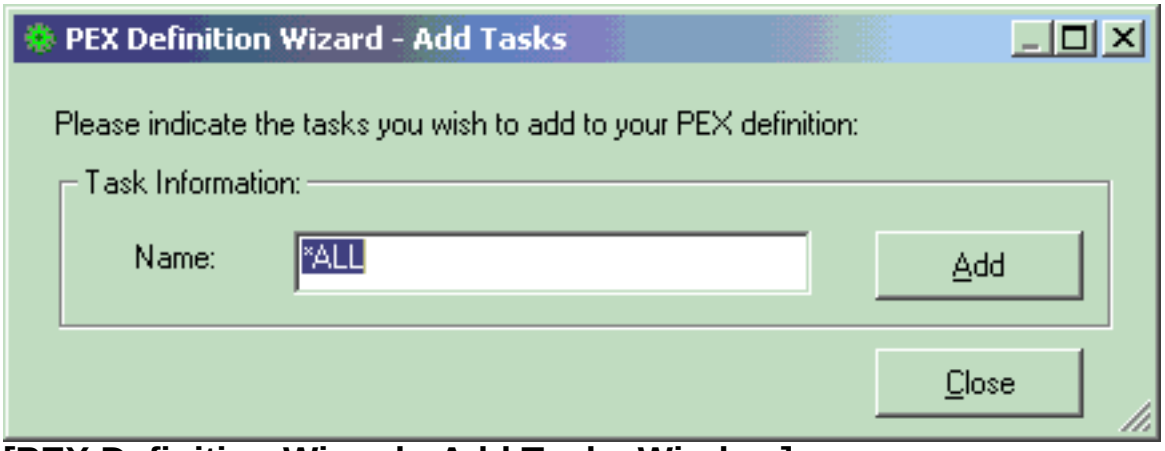
Field	Description
Tasks list	A list of tasks to include in the PEX definition.

Remove button	This button removes the selected tasks from the list.
Add Tasks button	Use this button to open the Add Tasks Window (discussed in the next section). This window is used to add task information to the task list.

Table of Contents | Previous | Next

4.4.1.11 Add Tasks Window

The add tasks window allows a user to add tasks to the Task Selection page in the wizard. The task name can either be *ALL, *NONE, a generic task name like Q*, or a specific task name. Change the task name field and click the add button for each task that you would like to include in your PEX definition



[PEX Definition Wizard - Add Tasks Window]

The table below summarizes the different elements on this page:

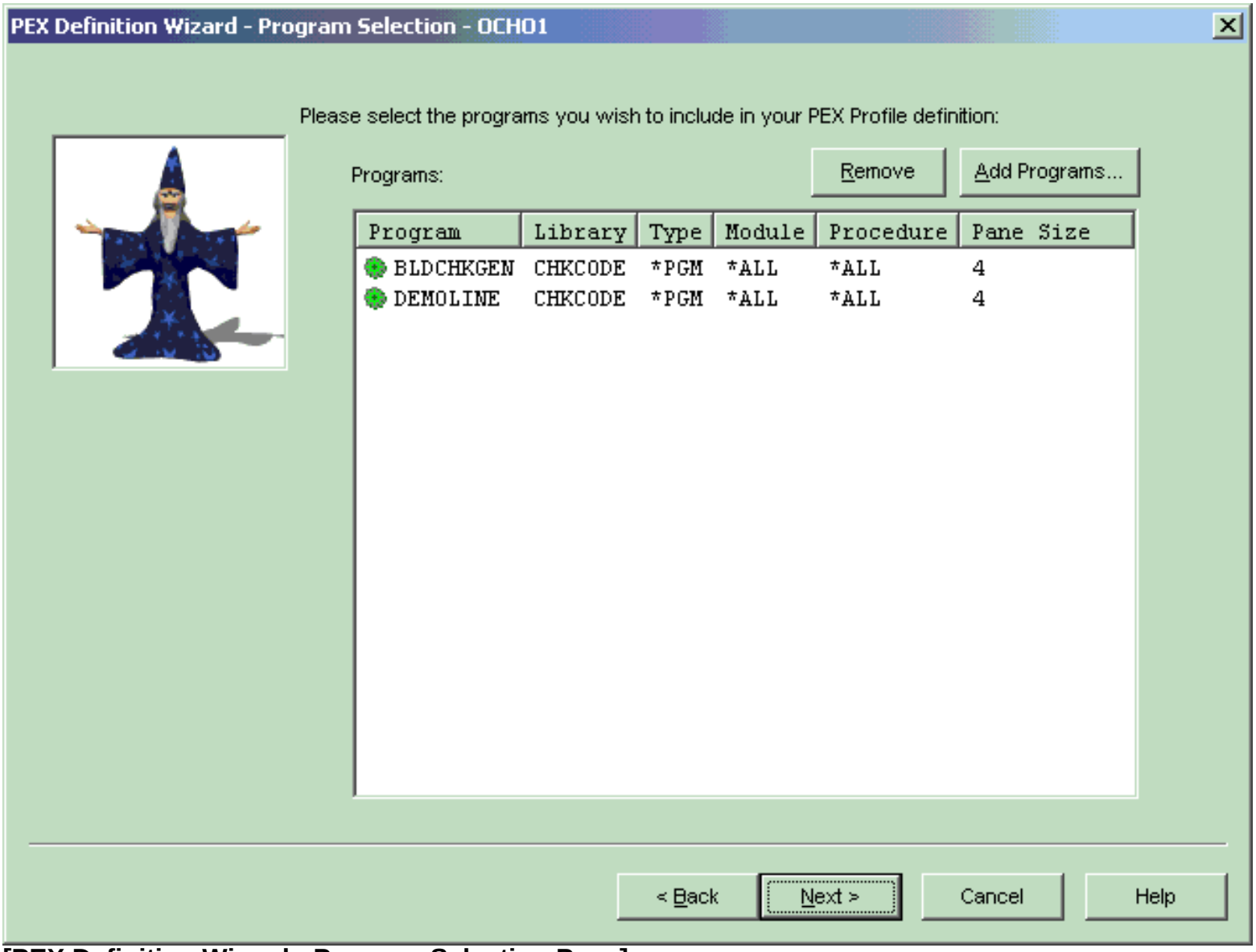
Field	Description
Name	A text field for entering a generic or specific task name. This value can also be *ALL or *NONE. When specifying a generic name use a * at the end of the task name.
Add button	This button will add the current task information to the Task Selection page.
Close button	Closes this window

Table of Contents | Previous | Next

4.4.1.12 Program Selection

The program selection page allows the user to select up to 16 program/module/procedure entries when creating a PEX Profile definition.

This page displays a list of selected program information to be captured in the PEX definition. Note that in order to get any useful information from this definition type you must select programs that have the enable for profiling flag turned on. There are also two buttons on this page used to add or remove programs from the list.



[PEX Definition Wizard - Program Selection Page]

The table below summarizes the different elements on this page:

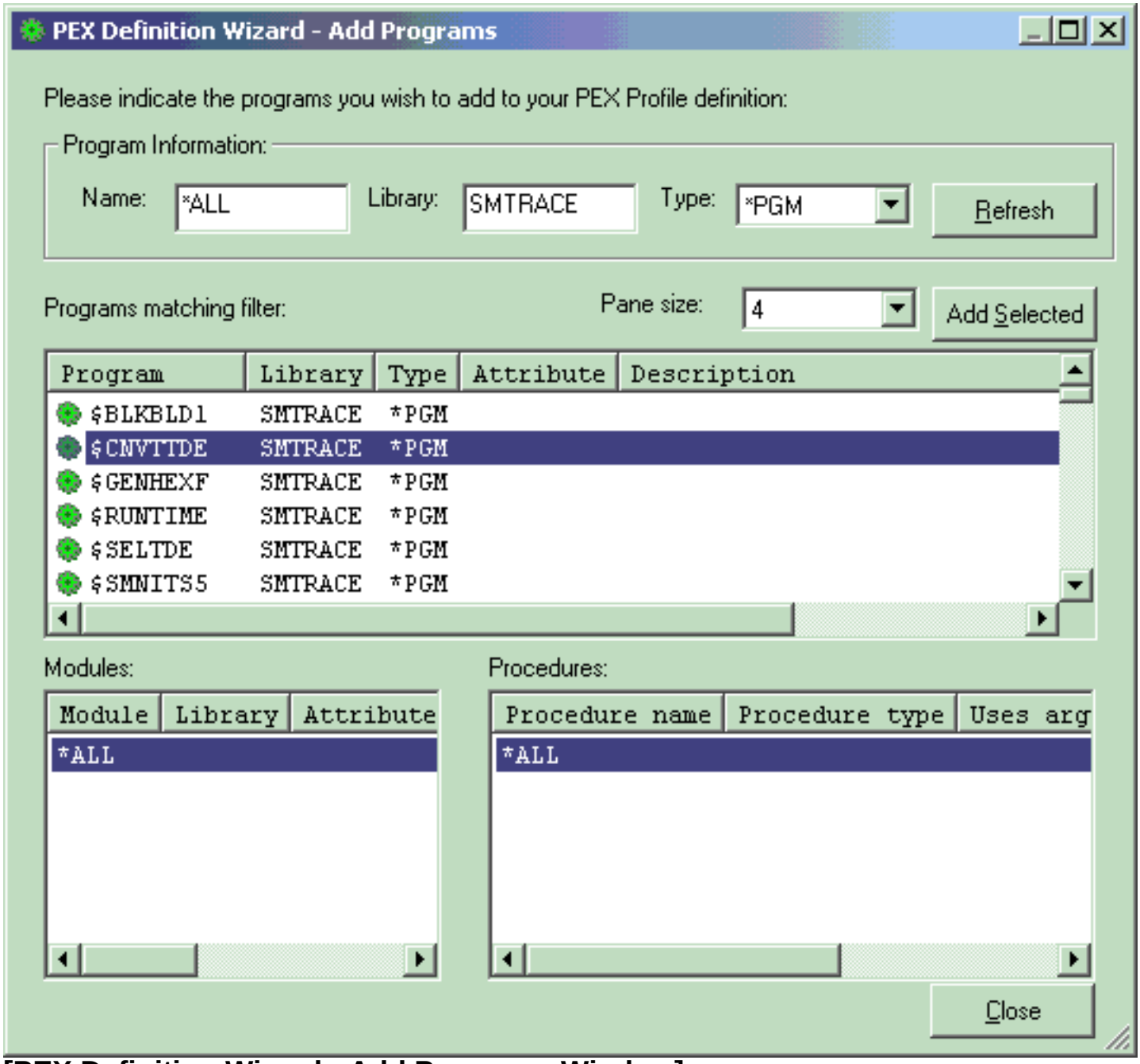
Field	Description
Programs list	A list of program information that will be included in the PEX definition.
Remove button	This button removes the selected program information from the Programs list.
Add Programs button	Use this button to open the Add Programs Window (discussed in the next section). This window is used to select and add additional program information to the Programs list.

Table of Contents | Previous | Next

4.4.1.13 Add Programs Window

The add programs window allows a user to browse any programs/service programs on the system using generic program and library names for the purpose of adding them to a PEX Profile definition. After finding the programs you want to add to the collection click the Add Selected button to add the selected program/module/procedure to the list. If a program is an ILE program you will see the modules contained within the program in the modules list. If desired select on these modules to see procedure entries found in the module. By selecting a specific program/module/procedure combination you can collect information only about the procedure(s) you are interested in.

The enable profiling flag must be turned on in the program and module you select in order to add the program/module/procedure information to the Program Selection Page.



[PEX Definition Wizard - Add Programs Window]

The table below summarizes the different elements on this page:

Field	Description
Name	A text field for entering a generic or specific program name. When specifying a generic name use a * at the end of the program name.
Library	A text field for entering a generic or specific library name. When specifying a generic name use a * at the end of the library name.
Type	This drop down lists contains the values *PGM and *SRVPGM. This offers the user the choice of viewing programs or service program objects.

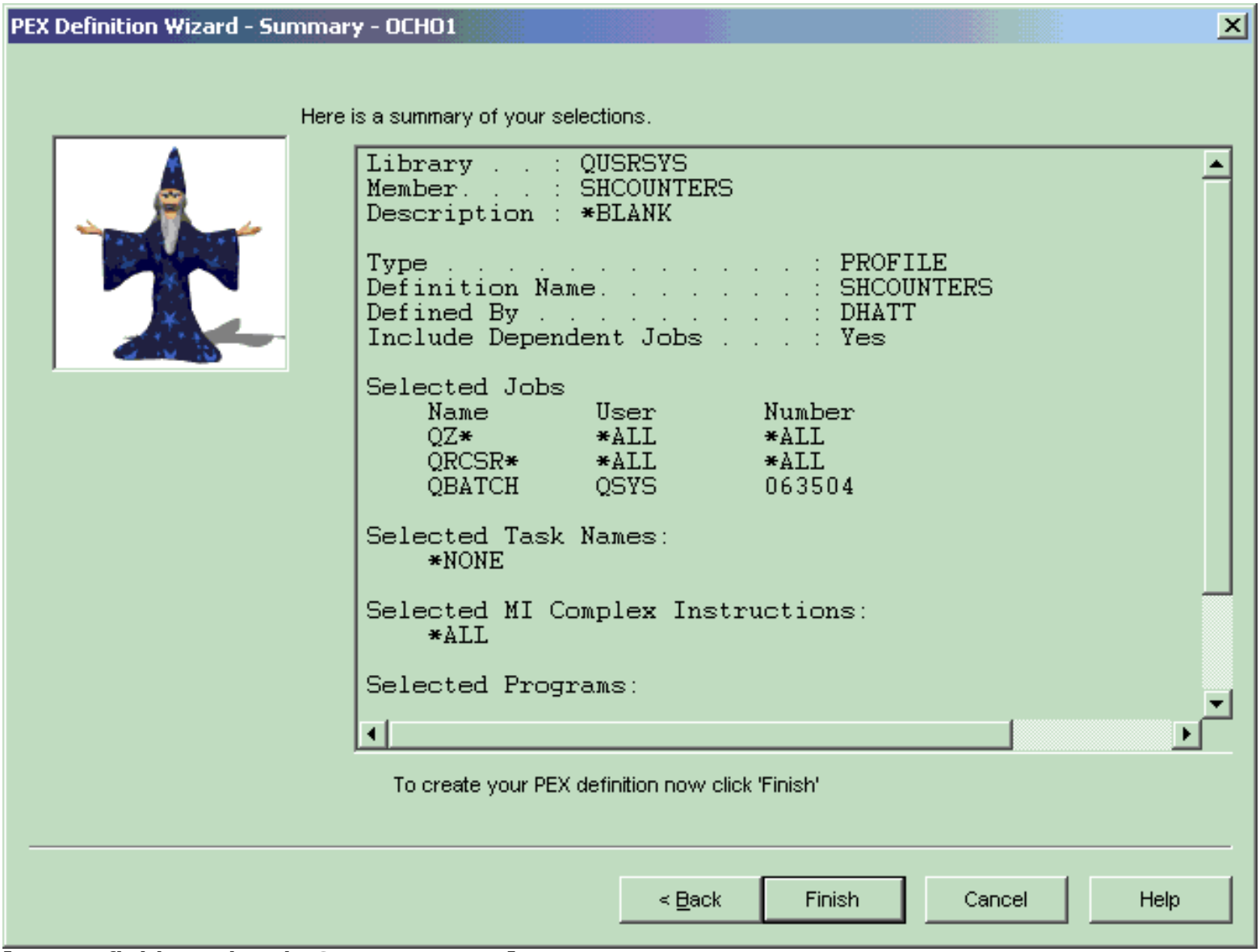
Browse button	The browse button updates the contents of the programs list.
Pane size	The pane size is the number of consecutive program instruction addresses assigned to each counter. The smaller the pane size, the more fine-grained the program profile information will be.
Add Selected button	This button will add the selected program, module and procedure to the Program Selection Page. The program added must be enabled for profiling.
Programs List	The list of programs matching the program information listed above. This list can display programs or service programs. Changing your selected program in the list will immediately refresh the module list showing the module information found within the seleted program.
Module List	The list of modules found within the selected program in the Programs List. Changing the selected module in the list will immediately refresh the procedure list showing the procedure information found within the seleted program.
Procedure List	The list of procedures found within the selected module.
Close button	Close the Add Programs window.

Table of Contents | Previous | Next

4.4.1.14 Summary

The summary page of the PEX Definition Wizard presents a summarization of all of the input provided in the wizard. It lists all of the details about the type of PEX definition to create or change, as well as the selected jobs or tasks, and the events to include

To create the PEX definition as defined click on the Finish button. After creating your definition you can use the PEX Collection Wizard to create a PEX collection using your new PEX definition.



[PEX Definition Wizard - Summary Page]

Table of Contents | Previous | Next

4.4.2 Deleting

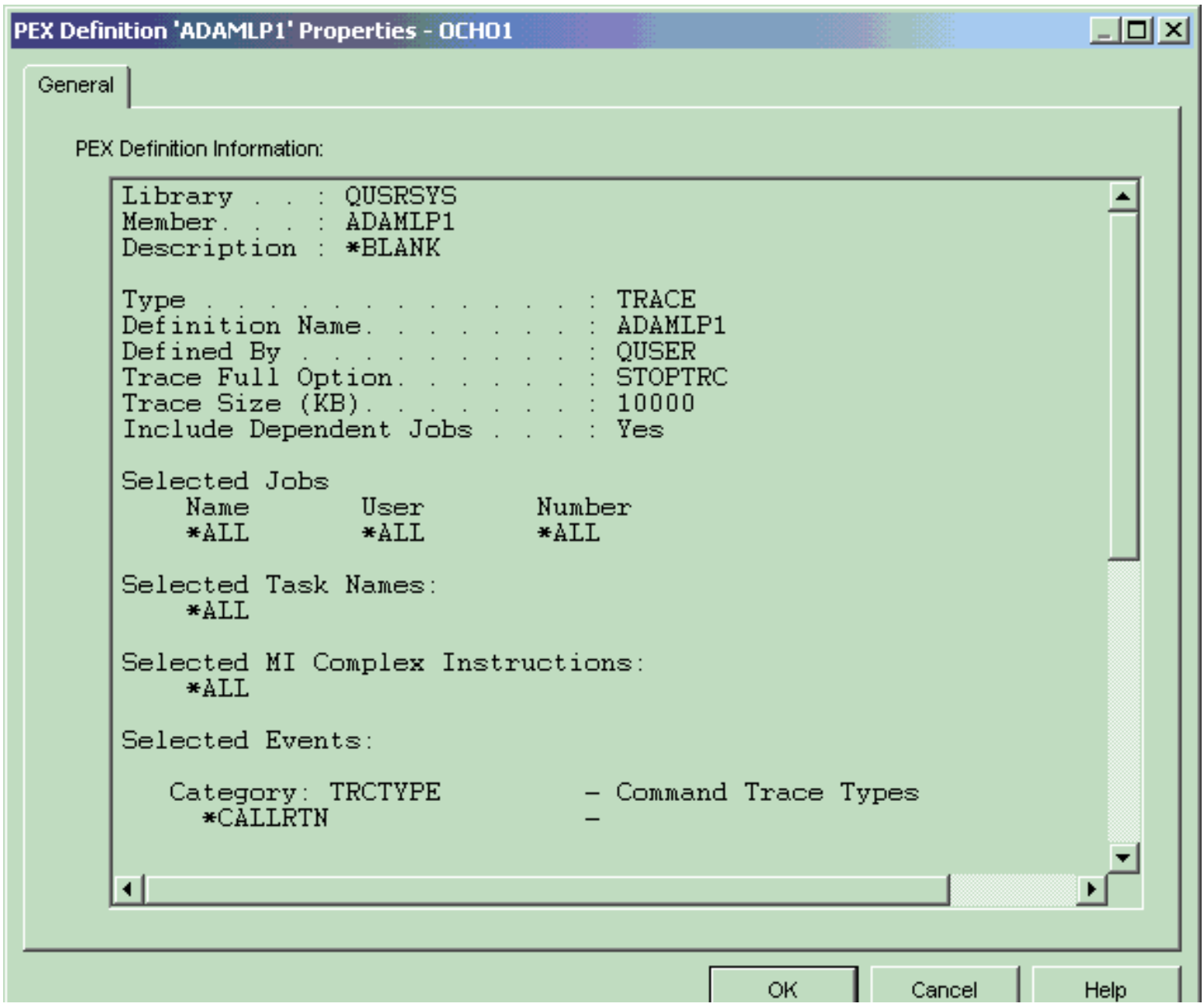
PEX Definitions can be deleted from within the PEX Definitions View. Right-click on one or more PEX definitions and choose the 'Delete...' menu to get rid of any unwanted definitions.

Table of Contents | Previous | Next

4.4.3 Properties

PEX Definition properties can be accessed from within the PEX definitions view. Double-click on any PEX definition to view its properties or right-click and choose the properties menu after selecting a PEX definition.

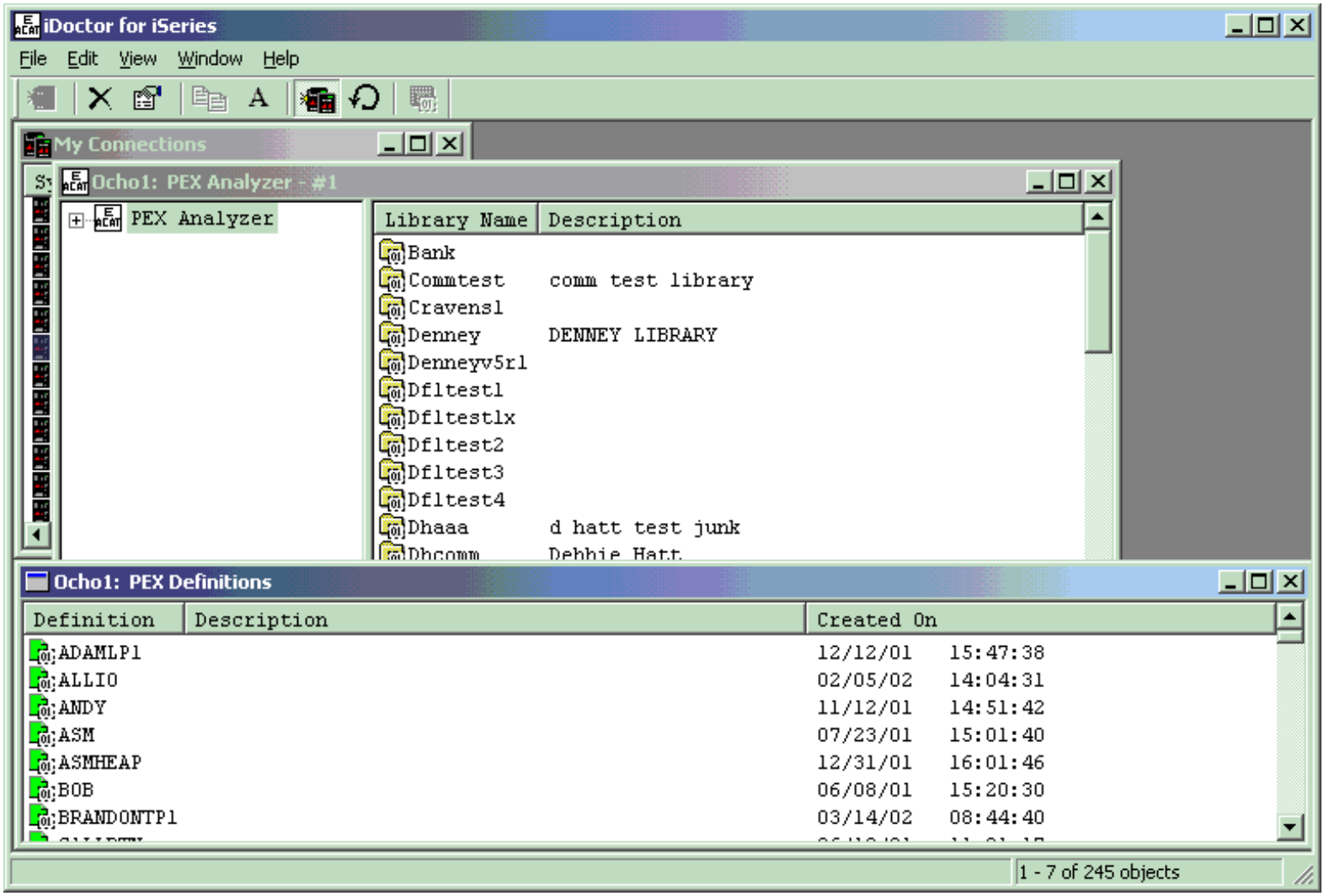
The properties shows all of the information about the PEX definition consist with that provided via the PRTPEXRPT command but conveniently via the GUI. You can also access a definitions properties within the PEX Collection Wizard by clicking the 'Details...' button within the Options page of the wizard.




OK
Cancel
Help

4.4.4 Working With

PEX Definitions can be viewed and manipulated from the PEX Definitions View. Access this view by using the 'View -> PEX Definitions' menu whenever a PEX Analyzer component view is active and has the current focus. The view contains a list of all PEX definitions defined on the system for the associated PEX Analyzer view.



Initially the list is sorted by name, by you can sort by time last changed if desired. Click one of the column headers to modify the sort sequence.

Menu Options

There are several options available for PEX definitions summarized in the table below:

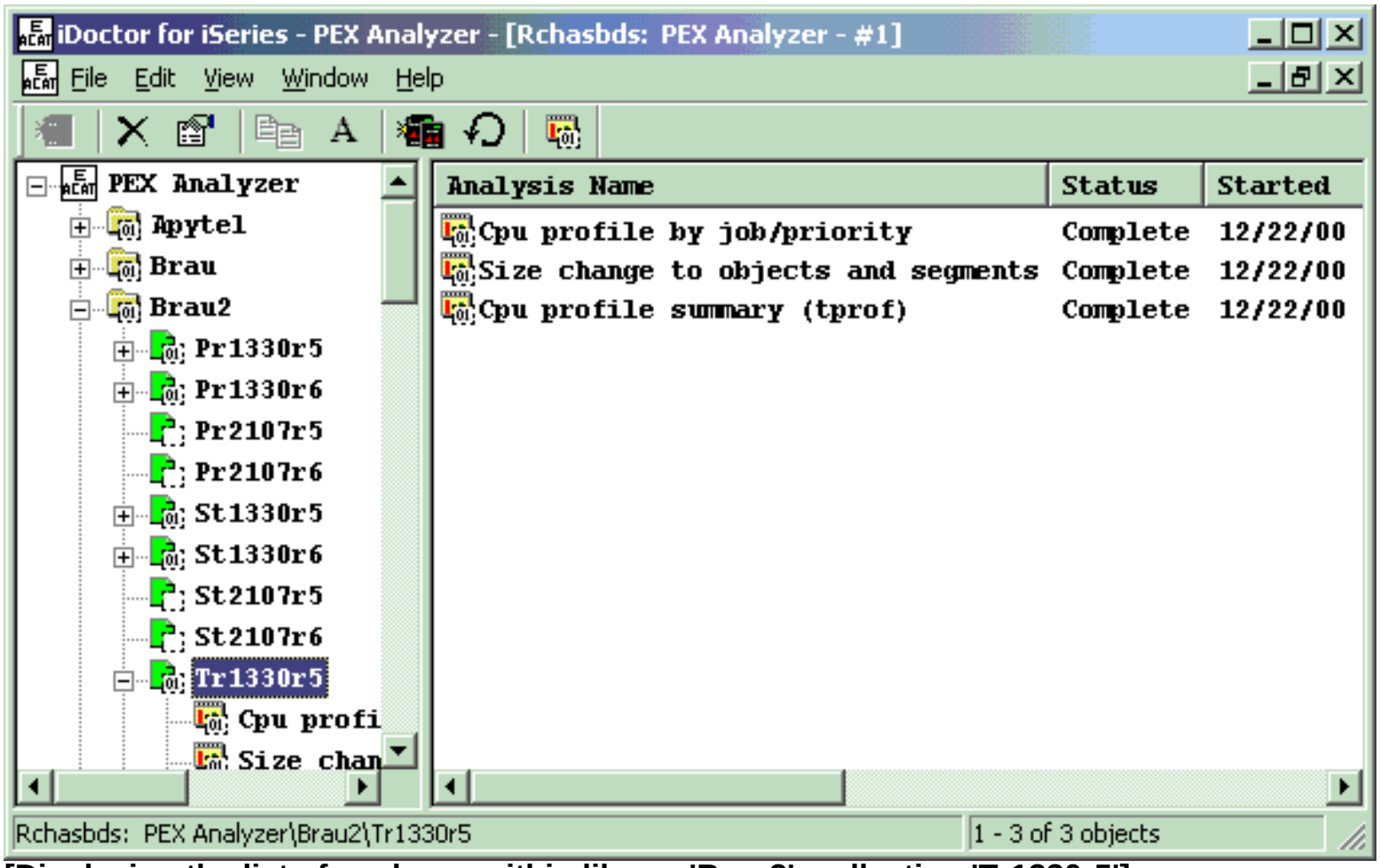
Menu	Description
Create PEX Definition...	Create a new PEX definition using the PEX Definition Wizard.
Change PEX Definition...	Change the PEX definition settings using the PEX Definition Wizard.

Create PEX Collection...	Lets you create a PEX collection within the PEX Collection Wizard using the selected PEX definition.
Delete...	Deletes the selected PEX definitions
Properties	Shows the PEX definition's properties.

Table of Contents | Previous | Next

4.5 Analyses

Every collection may contain zero or more analyses. Analyses may or may not be useable by the tool depending on the analysis status. The list of analyses displays to the user the analysis type, status, start time, and any subsetting criteria that was applied on the collection data when the analysis was created.



[Displaying the list of analyses within library 'Brau2', collection 'Tr1330r5']

Table of Contents | Previous | Next

4.5.1 Menu Options

PEX Analyses have the following menu options available:

Menu	Description
Explore	Displays the contents of the analysis (one or more reports) in the right pane of the tree/list window.
Work with Graphs...	Displays an interface that lets you work with the graphs that have been created over this analysis.
Work with Queries...	Displays an interface that lets you work with the queries that have been created over this analysis.
Delete...	Deletes an analysis. Select multiple analyses in order to delete more than one analysis at a time.
Properties	Use this menu to display the propery pages for an analysis. The property pages contain general information about the analysis type and status as well as the subsetting information that was applied to the collection when the analysis was created. Intervalized analyses will also have a property page displaying the interval size and start and end time of the analysis.

Table of Contents | Previous | Next

4.5.2 Creating - The Analysis Wizard

The Analysis Wizard offers the iDoctor for iSeries user a convenient way of subsetting a collection and formatting collection data in a way that best suits their particular needs. Selecting the Analyze Data menu for a selected collection invokes the Analysis Wizard. A collection must have a status of 'Ready for analysis' in order for the wizard to be available. The Analysis Wizard gathers information such as what type of analysis to create and how the collection data should be subsetted. This information is used to execute a server-side program call which will create the analysis so it may be viewed in iDoctor.

Depending on the type of collection and what types of analysis and subsetting are possible on the PEX collection the content of the Analysis Wizard will vary. The next sections will list the common and specialized screens within the wizard and provide a discussion of their basic use.

Table of Contents | Previous | Next

4.5.2.1 Welcome Page

The Welcome Page simply informs the user that the wizard allows them to perform analyses over their PEX collection. The content of this page will differ slightly depending on the type of the PEX collection being analyzed.

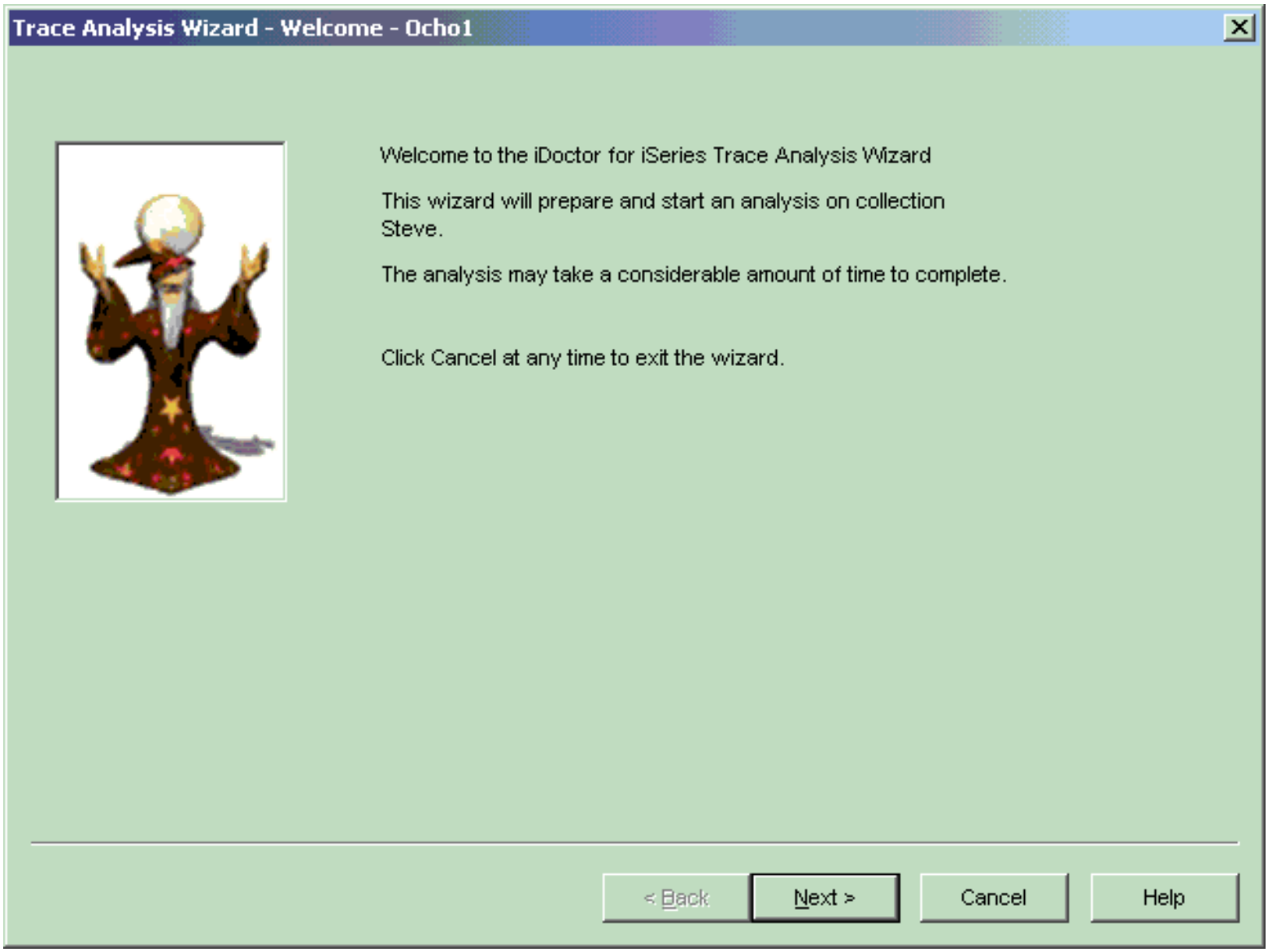


Table of Contents | Previous | Next

4.5.2.2 Select Analysis To Run

The Select Analysis To Run Page is used to select the desired analysis to run on the current collection. The available analysis types are determined when the PEX collection was created on the iSeries. To help make your selection a detailed description of the currently selected analysis is displayed underneath the list of available analysis types.

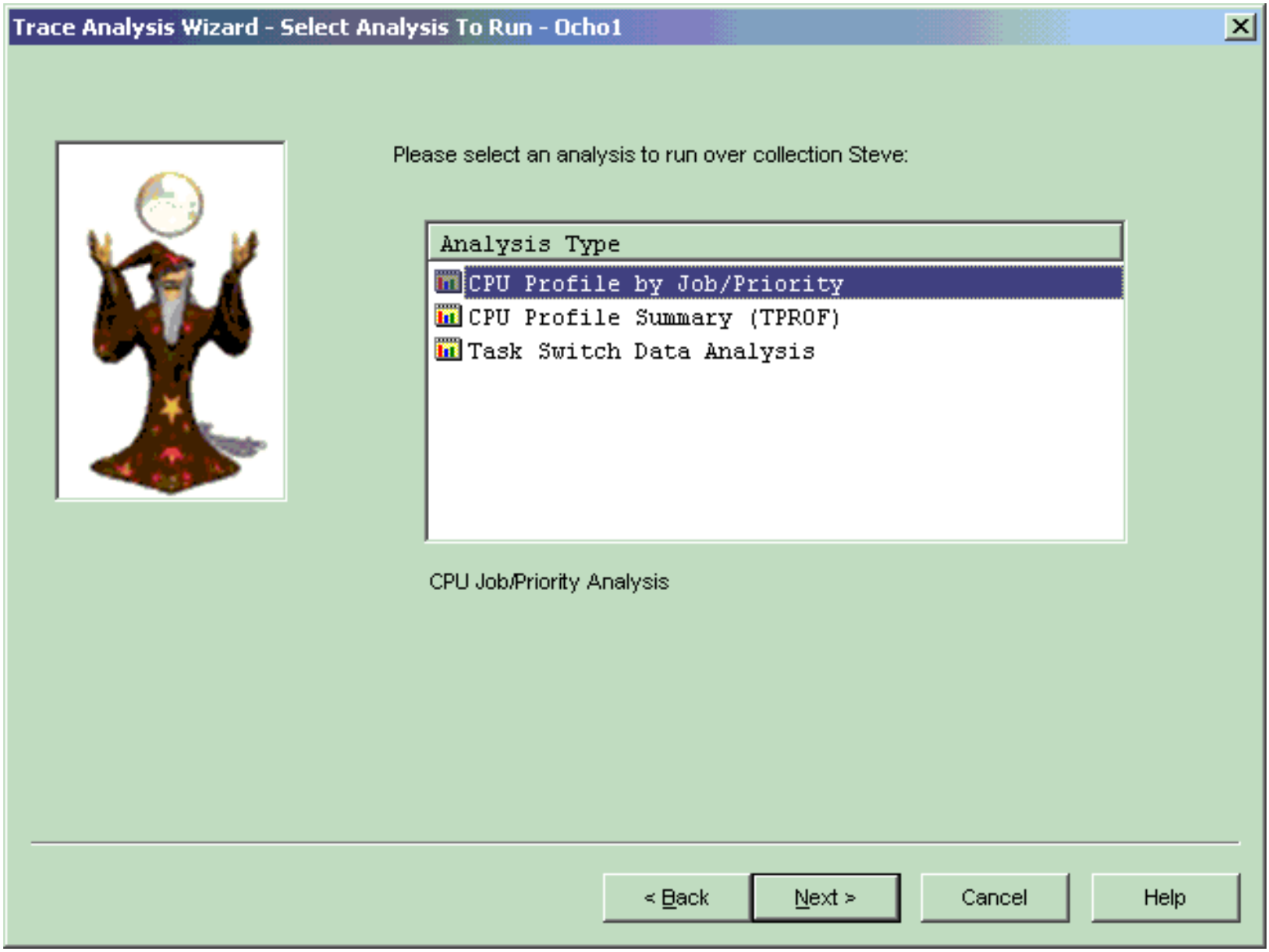


Table of Contents | Previous | Next

4.5.2.3 Trace Subset Options

The Trace Subset Options Page presents the possible ways the PEX Trace collection may be analyzed. Note: The determination for which options should be enabled is made when the PEX Trace collection is created on the iSeries. Selecting one or more subset options on this page will cause their corresponding screens to be displayed next in the Analysis Wizard.

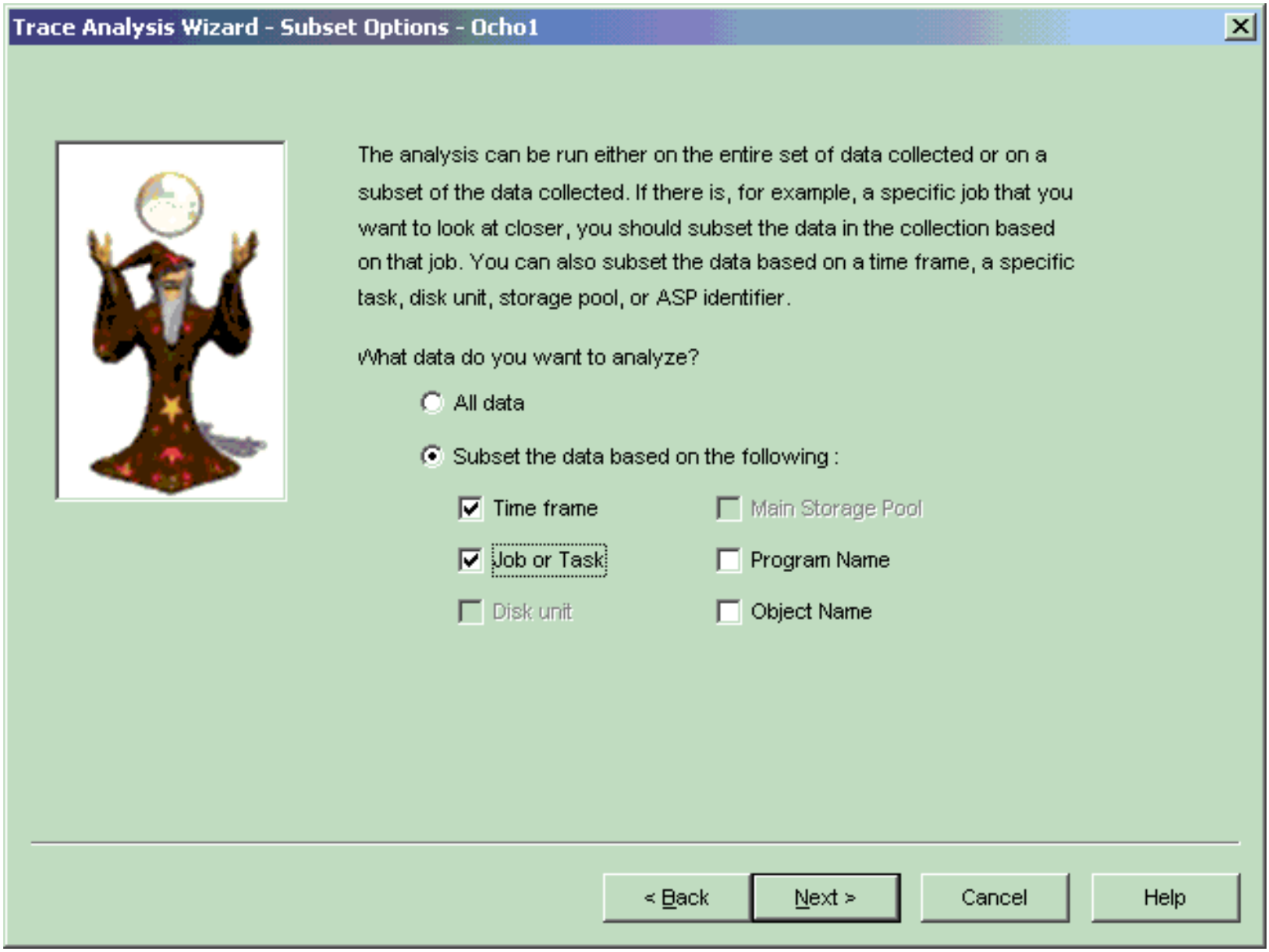


Table of Contents | Previous | Next

4.5.2.4 Profile Presentation Options

The Profile Presentation Options Page allows the user to group and sort the PEX Profile collection data to suit their preferences. The filter percentage field is used to remove any hit count activity below the given percentage.

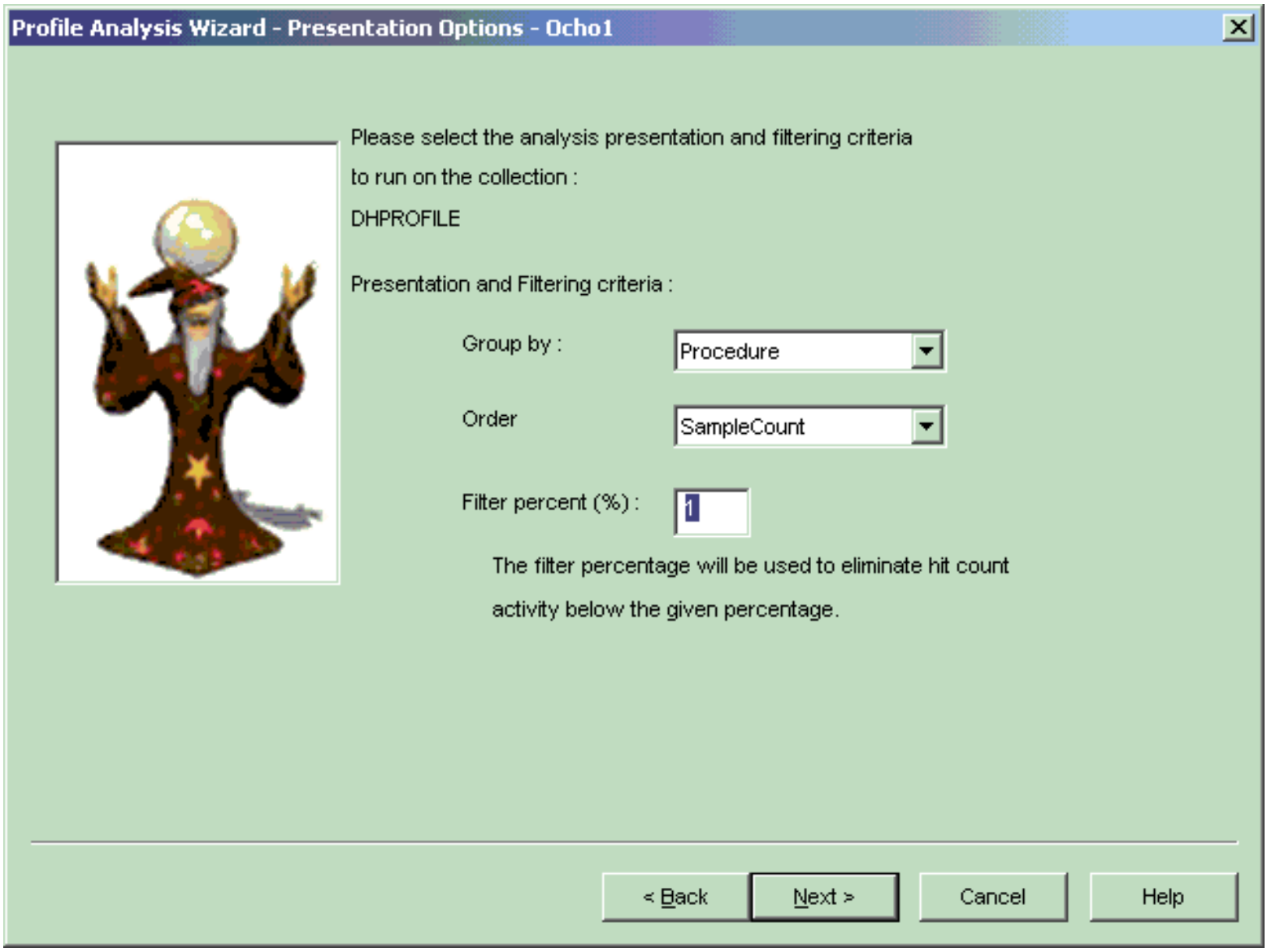


Table of Contents | Previous | Next

4.5.2.5 Statistical Subset Options

The Statistical Subset Options Page allows the user to summarize the PEX Stats collection data to suit their preferences. A "group by" option is used to categorize the analysis data by either module name or program name. An option to include collection overhead is also available. Stats hierarchial collections do not have the option to group by program or module but Stats flat collections have that option.

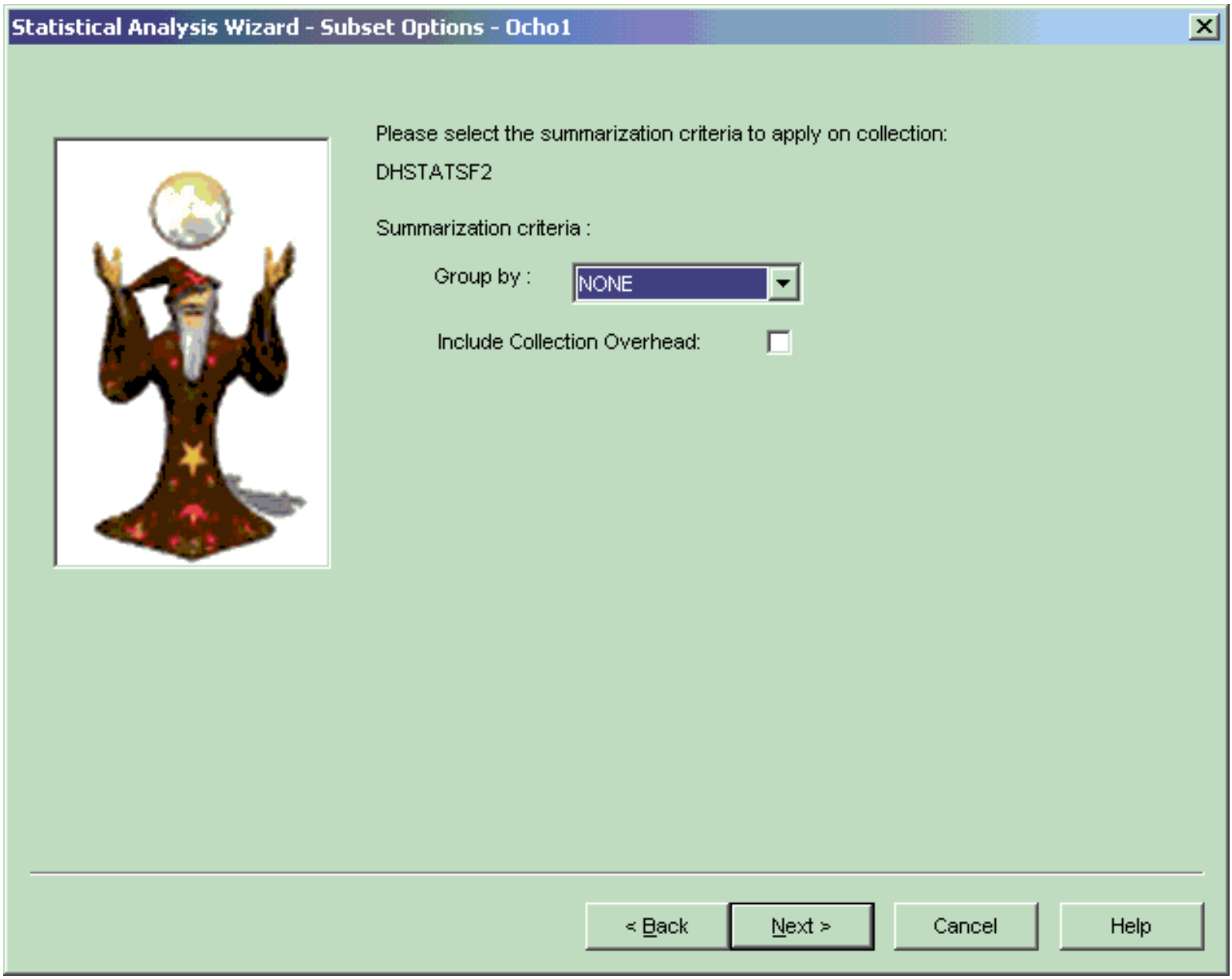


Table of Contents | Previous | Next

4.5.2.6 Subset Time

The Subset Time Page allows the user to subset a PEX collection by time in order to look at a specific block of time in the collection. When this page first loads the collection start time and collection end time are shown. Modify the time values to be a subset of the start/end time range in order to subset the PEX collection by time. The Reset button may be used to restore the initial start time and end time values. This type of subsetting is only available for PEX Trace collections.

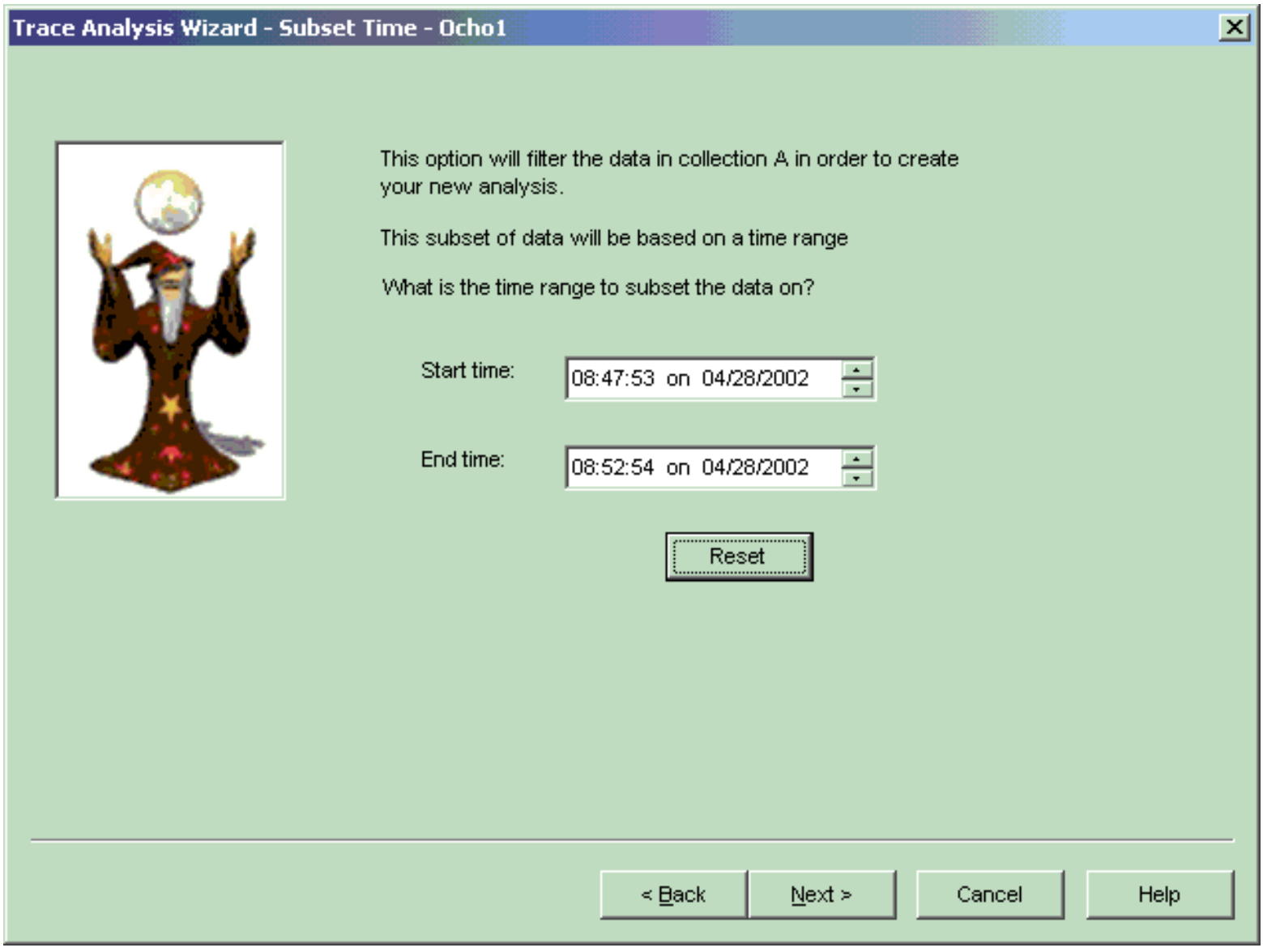


Table of Contents | Previous | Next

4.3.2.7 Subset Job or Task

The Subset Job or Task Page allows the user to subset a PEX collection by a specific job or task within the PEX collection. Task subsetting is only available on PEX Trace collections. Job subsetting is available on Trace, Statistical Hierarchial, and Statistical Flat Merge Jobs *NO collections.

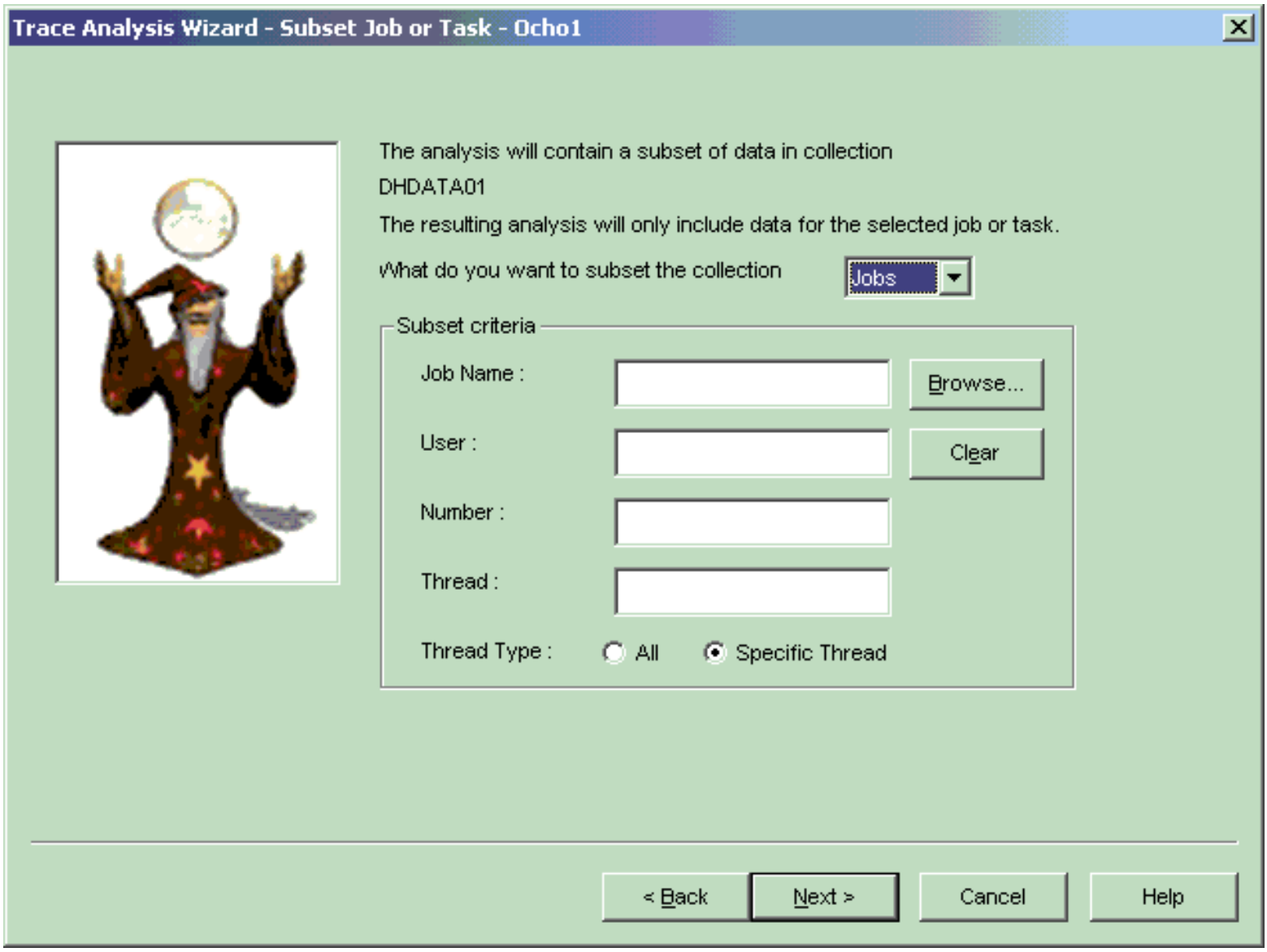


Table of Contents | Previous | Next

4.3.2.8 Browse Jobs Dialog

The Browse Jobs Dialog allows a user to select the job they would like to subset the colleciton over. This window is displayed when the user has chosen to subset a collection by job and they click the Browse... button from the Subset Job or Task page in the Analysis Wizard. Due to the potential for huge amounts of data, a menu is available by right-clicking on the list providing additional features. The popup menu offers copy and paste to the clipboard, customizable font, and search capabilities.

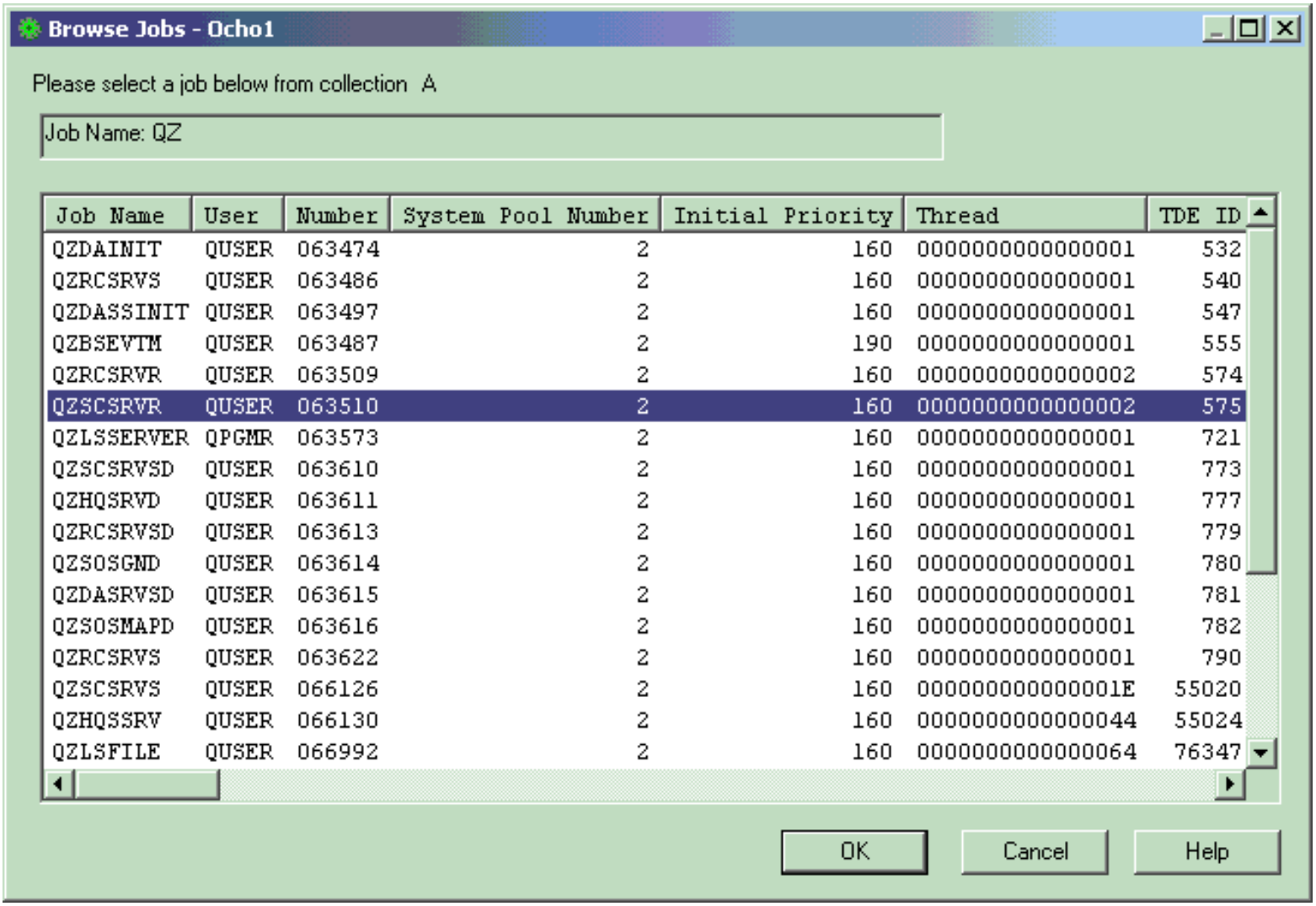


Table of Contents | Previous | Next

4.5.2.9 Browse Tasks Dialog

The Browse Tasks Dialog allows a user to select the task they would like to subset the collection over. This window is displayed when the user has chosen to subset a collection by task and they click the Browse... button from the Subset Job or Task page in the Analysis Wizard. Due to the potential for huge amounts of data, a menu is available by right-clicking on the list providing additional features. The popup menu offers copy and paste to the clipboard, customizable font, and search capabilities.

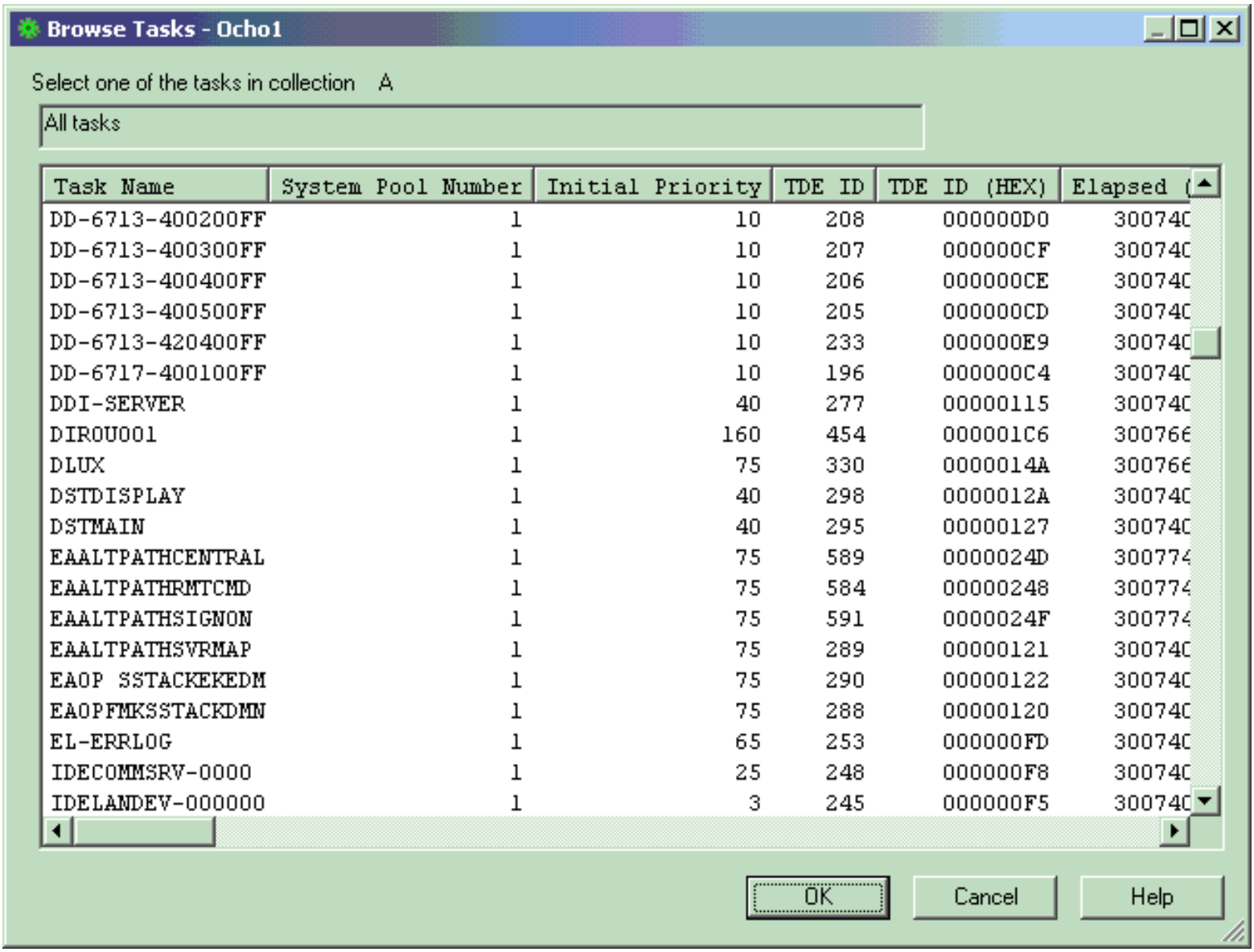


Table of Contents | Previous | Next

4.5.2.10 Subset Storage Pool Page

The Subset Storage Pool Page allows a user to subset by main storage pool. If an invalid storage pool is entered no data will be available in the resulting analysis. This type of subsetting is only available for PEX Trace collections.

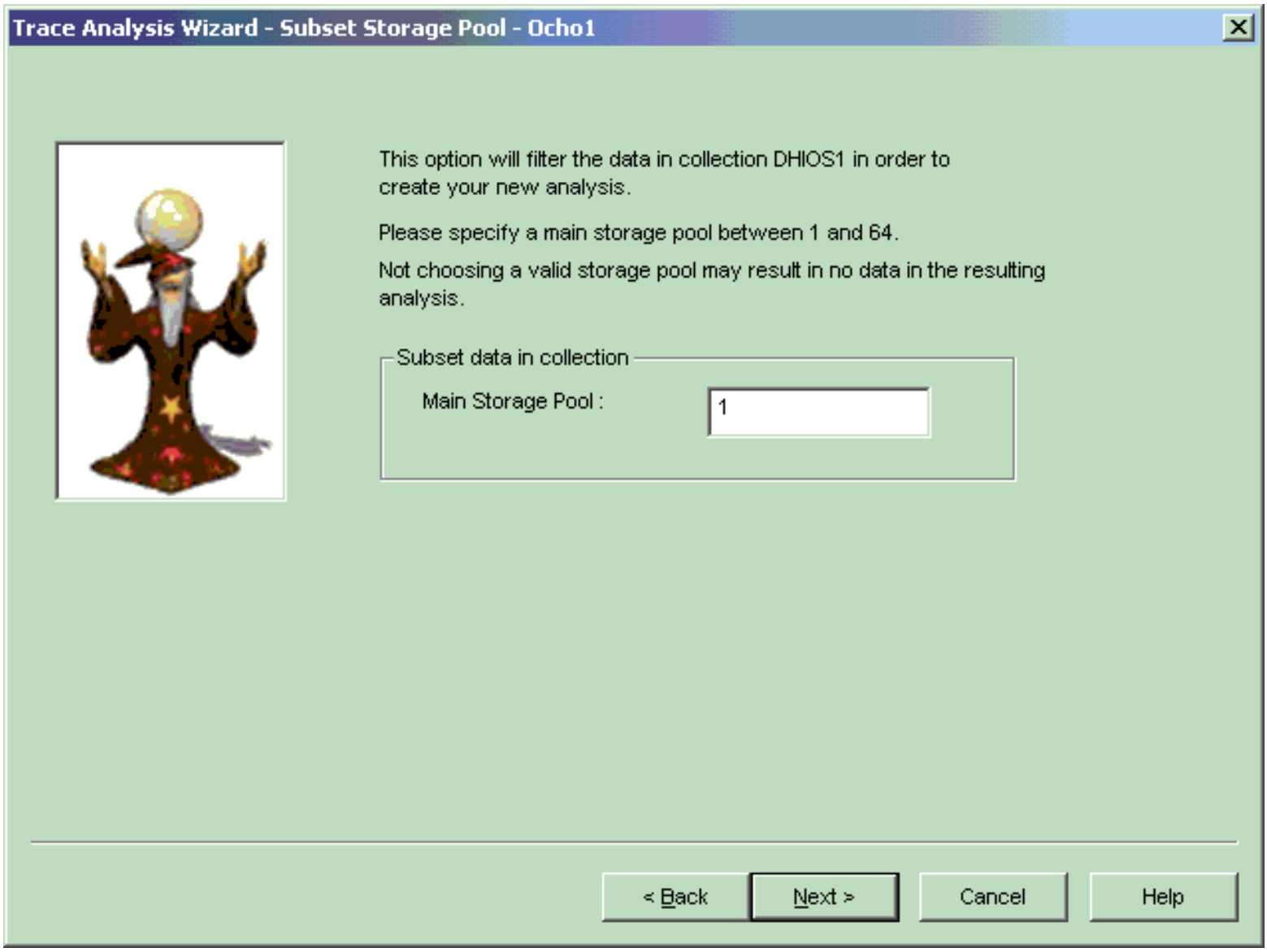


Table of Contents | Previous | Next

4.5.2.11 Subset Disk Unit Page

The Subset Disk Unit Page allows the collection data to subsetted by a specific disk unit. If an invalid disk unit is entered no data will be available in the resulting analysis. This type of subsetting is only available for PEX Trace collections.

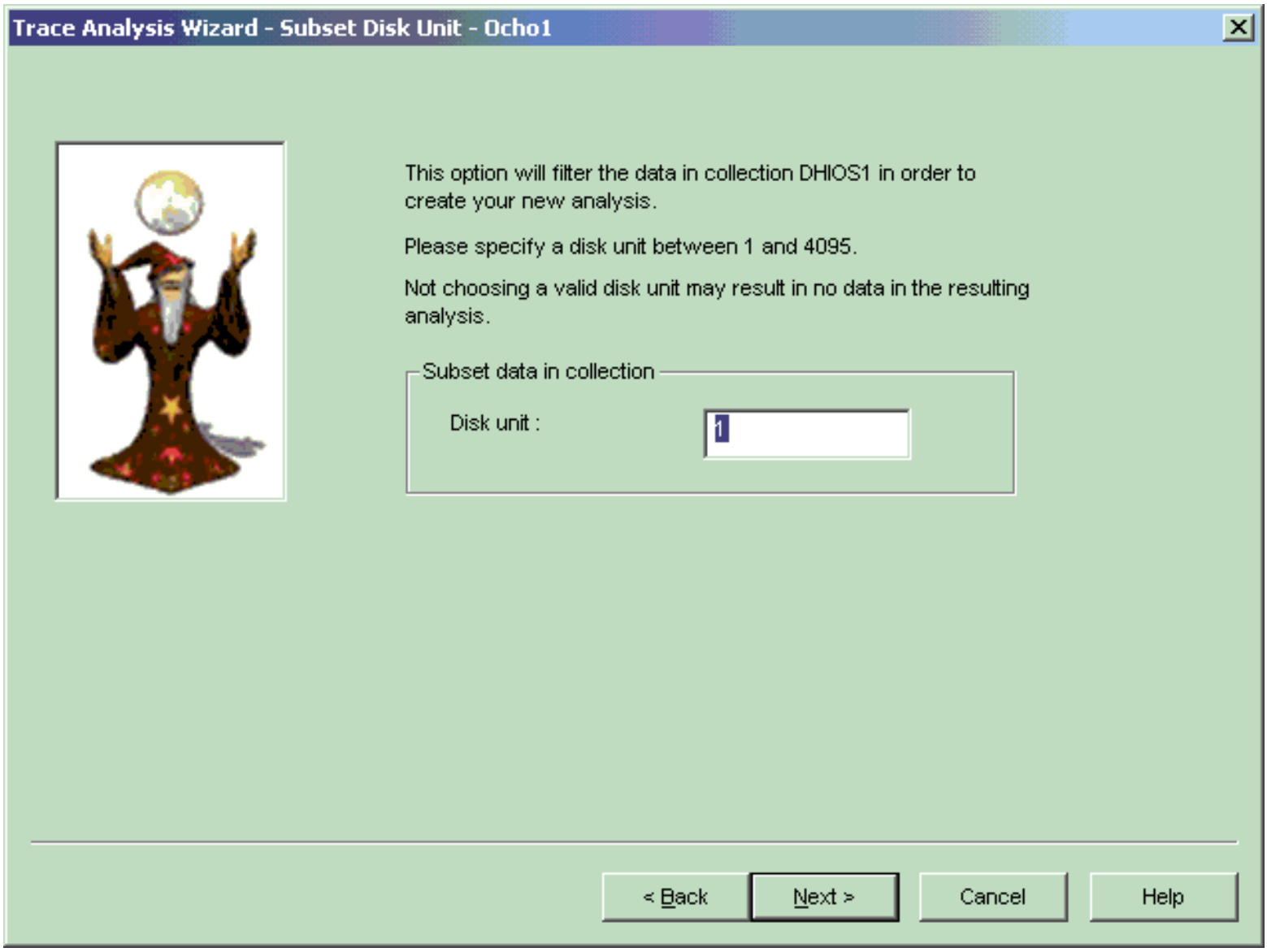


Table of Contents | Previous | Next

4.5.3.12 Subset Program Page

The Subset Program Page allows a user to subset on a specific application program name. Choosing All or leaving the field blank will include all programs in the resulting analysis. This type of subsetting is only available for PEX Trace collections.

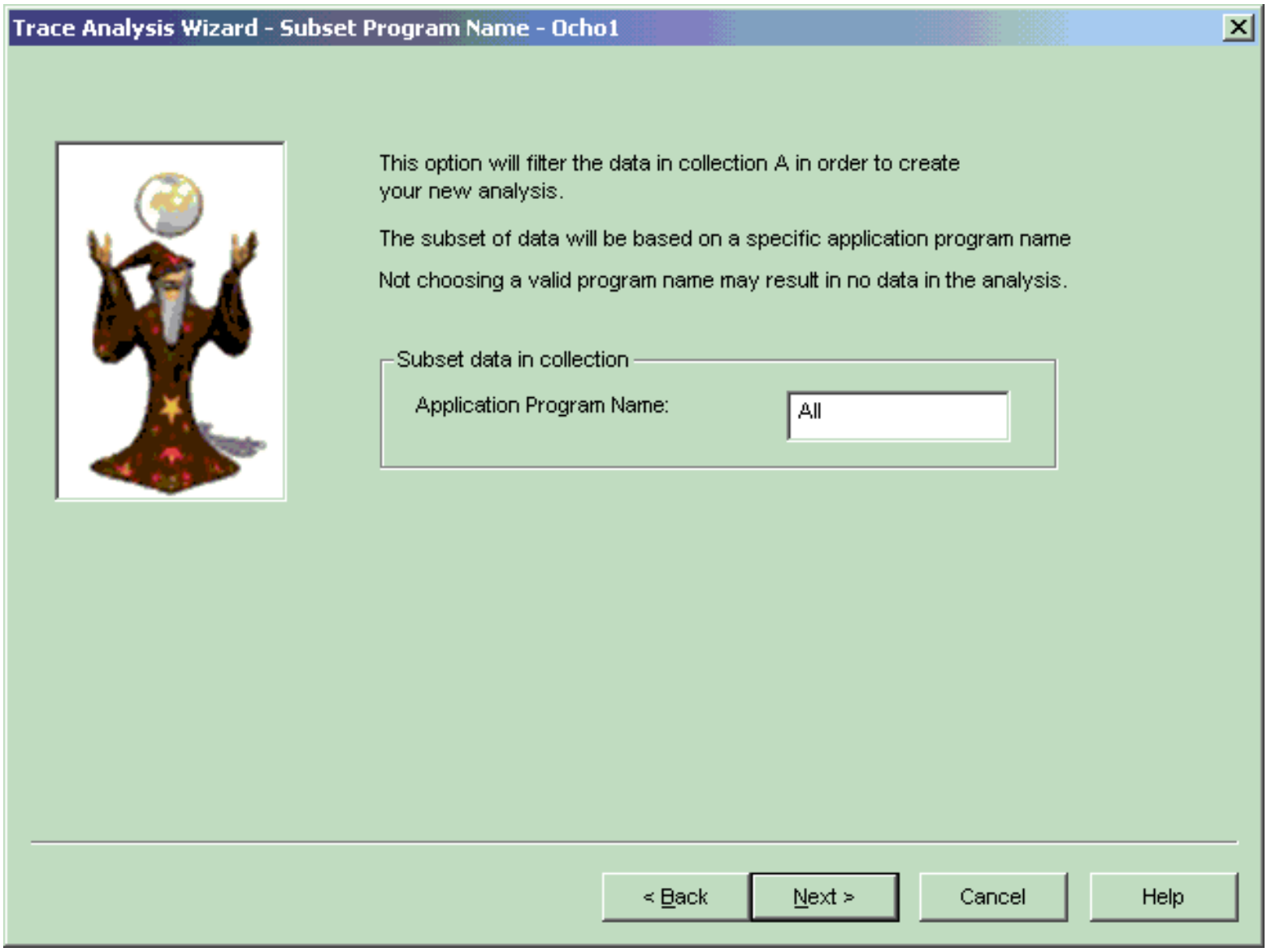


Table of Contents | Previous | Next

4.5.3.13 Subset Object Page

The Subset Object Page allows a user to subset on a specific object filename. If an invalid library, file, member combination is entered no data will be available in the resulting analysis. This type of subsetting is only available for PEX Trace collections.

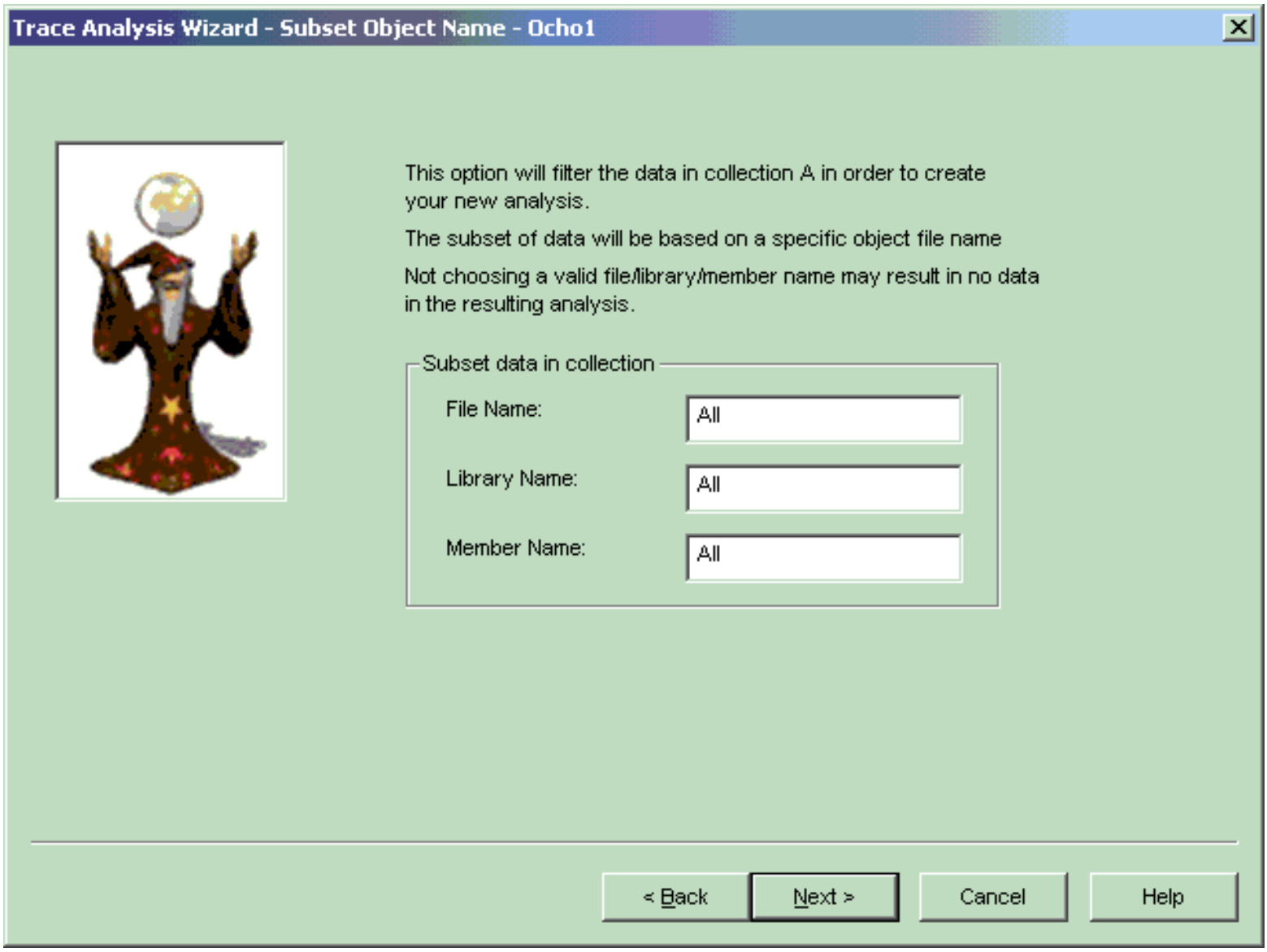


Table of Contents | Previous | Next

4.5.3.14 Time Interval Options Dialog

The Time Interval Options Dialog allows a user to indicate how the resulting time frames within the analysis should be divided. Intervals may be divided into by a certain number of seconds or by setting a fixed number of intervals to divide the entire analysis.

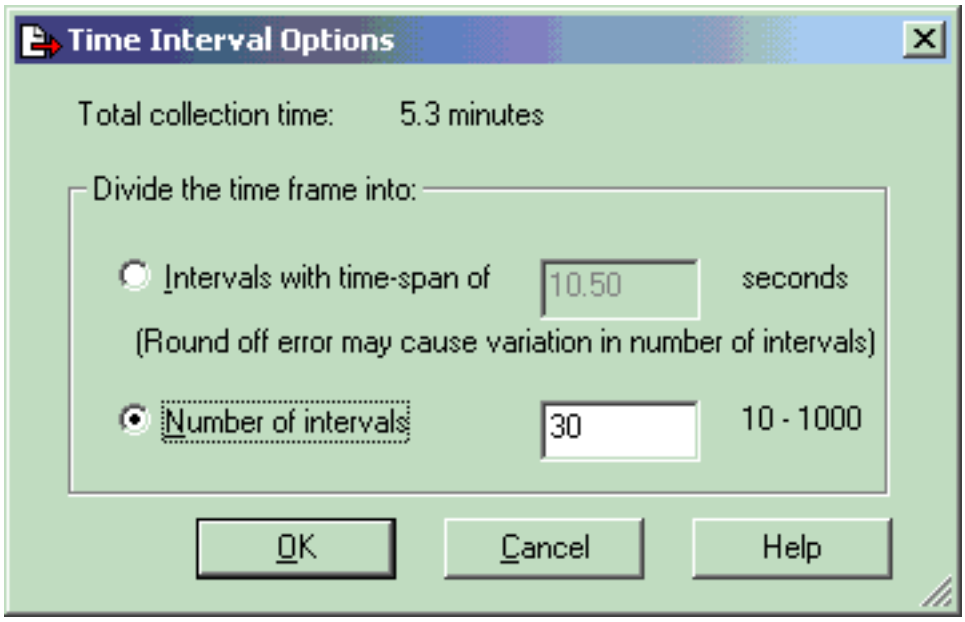


Table of Contents | Previous | Next

4.5.3.15 Summary Page

The Summary Page lists all of the users selection criteria identifying how the analysis will be created. Clicking Finish will submit the analysis request to the iSeries. If the current collection is a PEX Trace collection the Advanced button is available for setting the optional value for the number of intervals the data will be subsetted into.

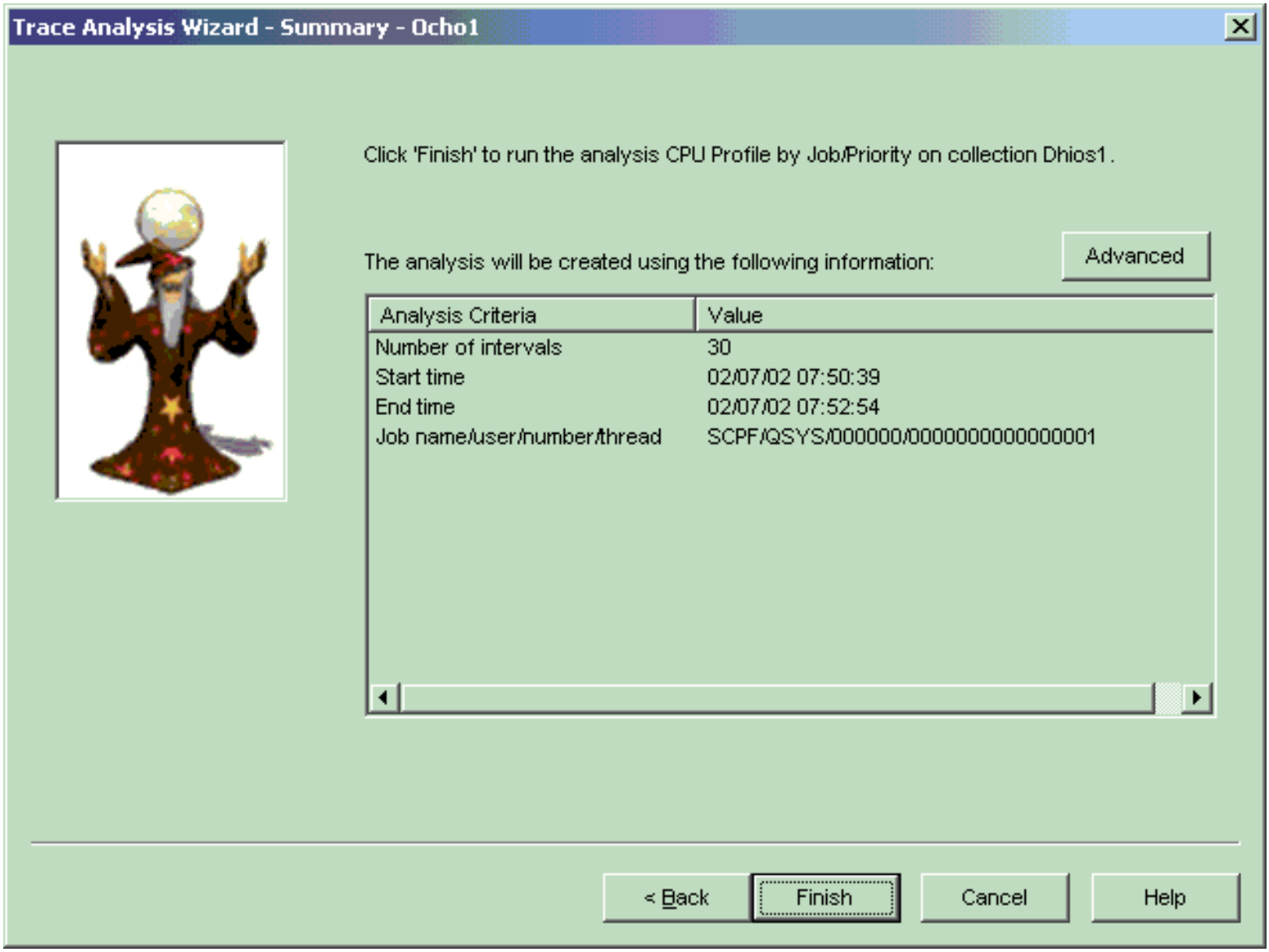
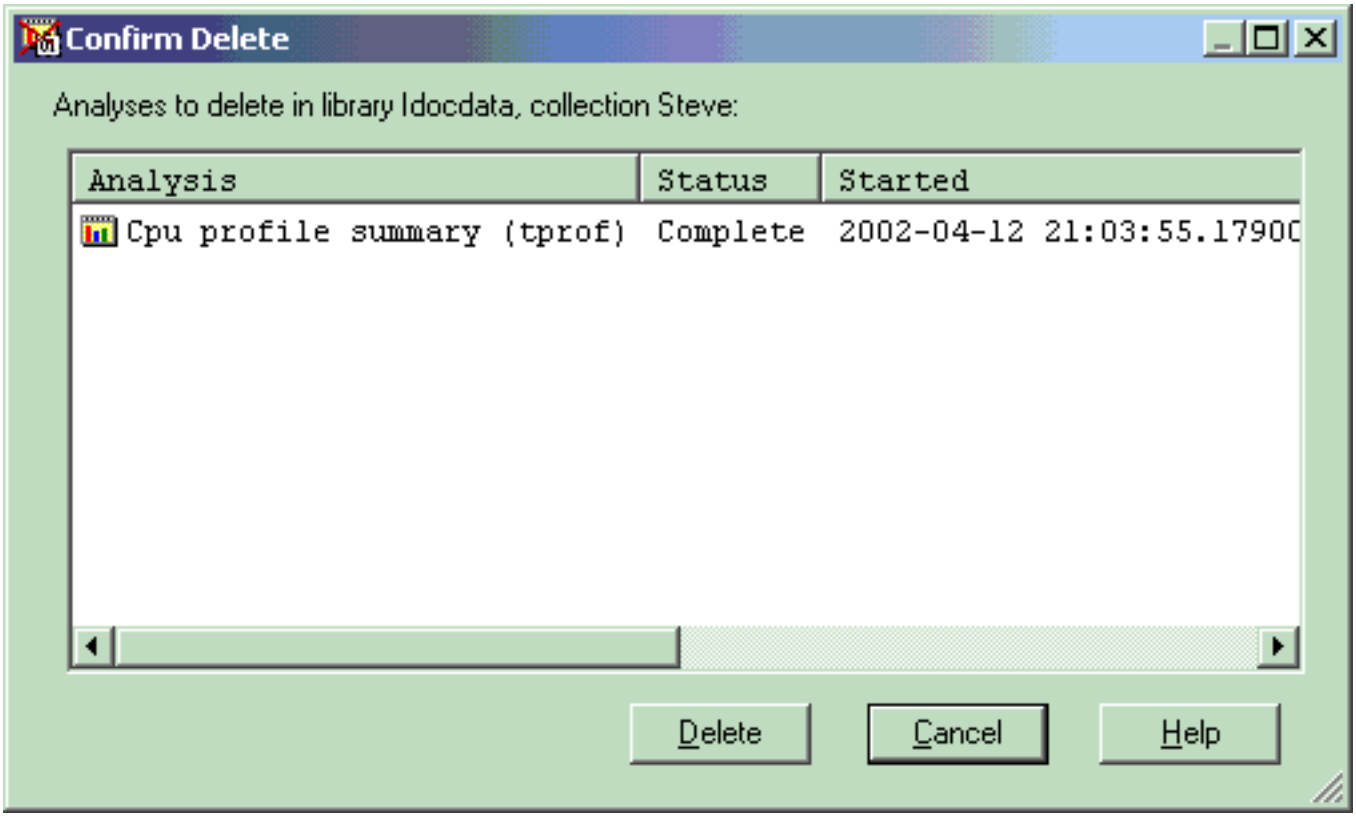


Table of Contents | Previous | Next

4.5.3 Deleting

A user may choose to delete one or more analyses. Select the analyses you would like to delete and choose the 'Delete...' menu. If you have trouble deleting an analysis there may be someone else using it (possibly yourself). Ensure that all table or graph views over the analysis to be deleted have been closed down.



[An example of the Delete Analysis Dialog]

Table of Contents | Previous | Next

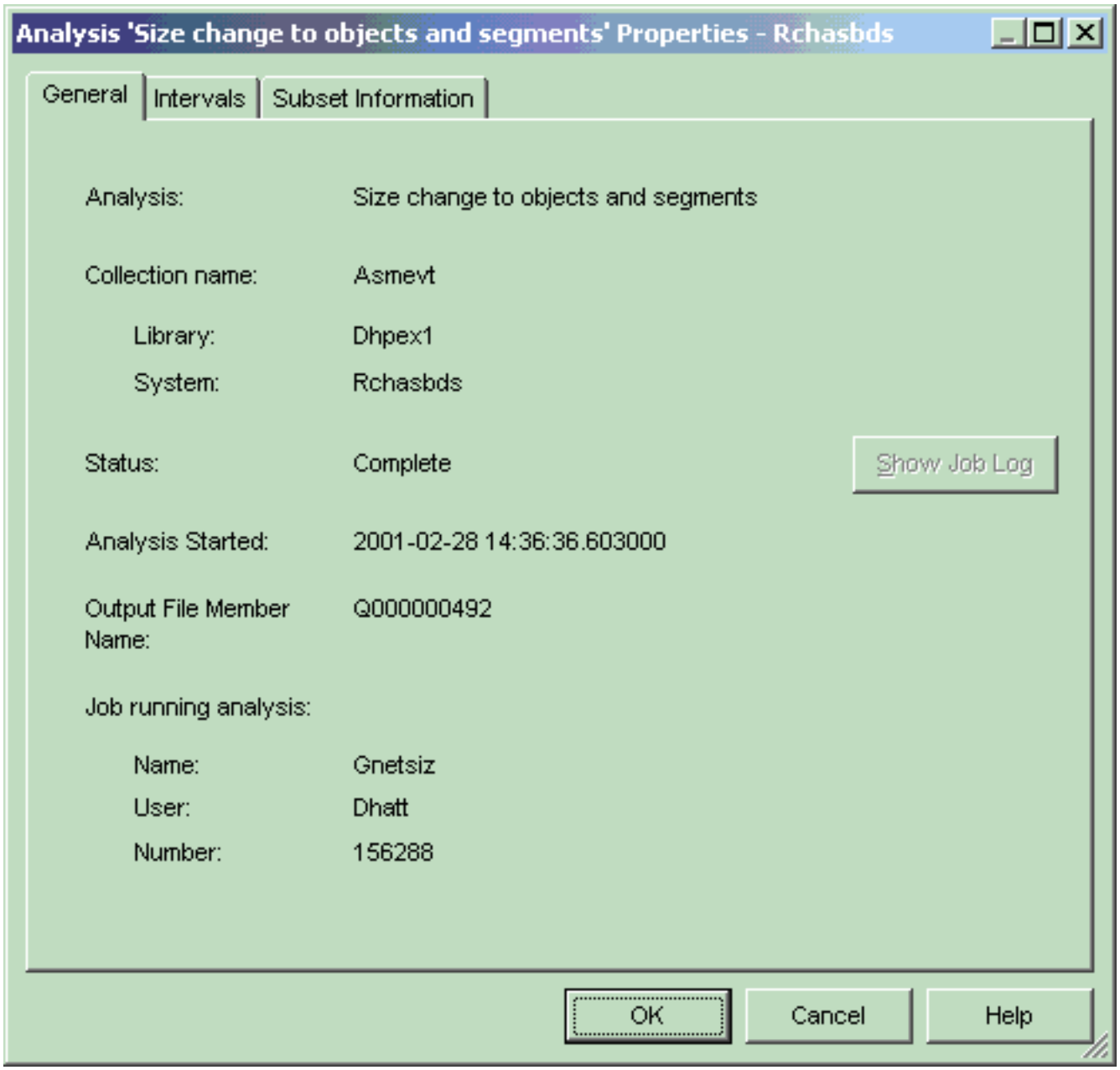
4.5.4 Properties

An analysis is a subset of performance data within a collection. There are several property pages for analyses which are defined in this section. A user may invoke the property pages by right-clicking on the desired analysis and choosing Properties.

Table of Contents | Previous | Next

4.5.4.1 General

The General property page provides a summary of the most basic information about an analysis.



[General Property Page for an analysis]

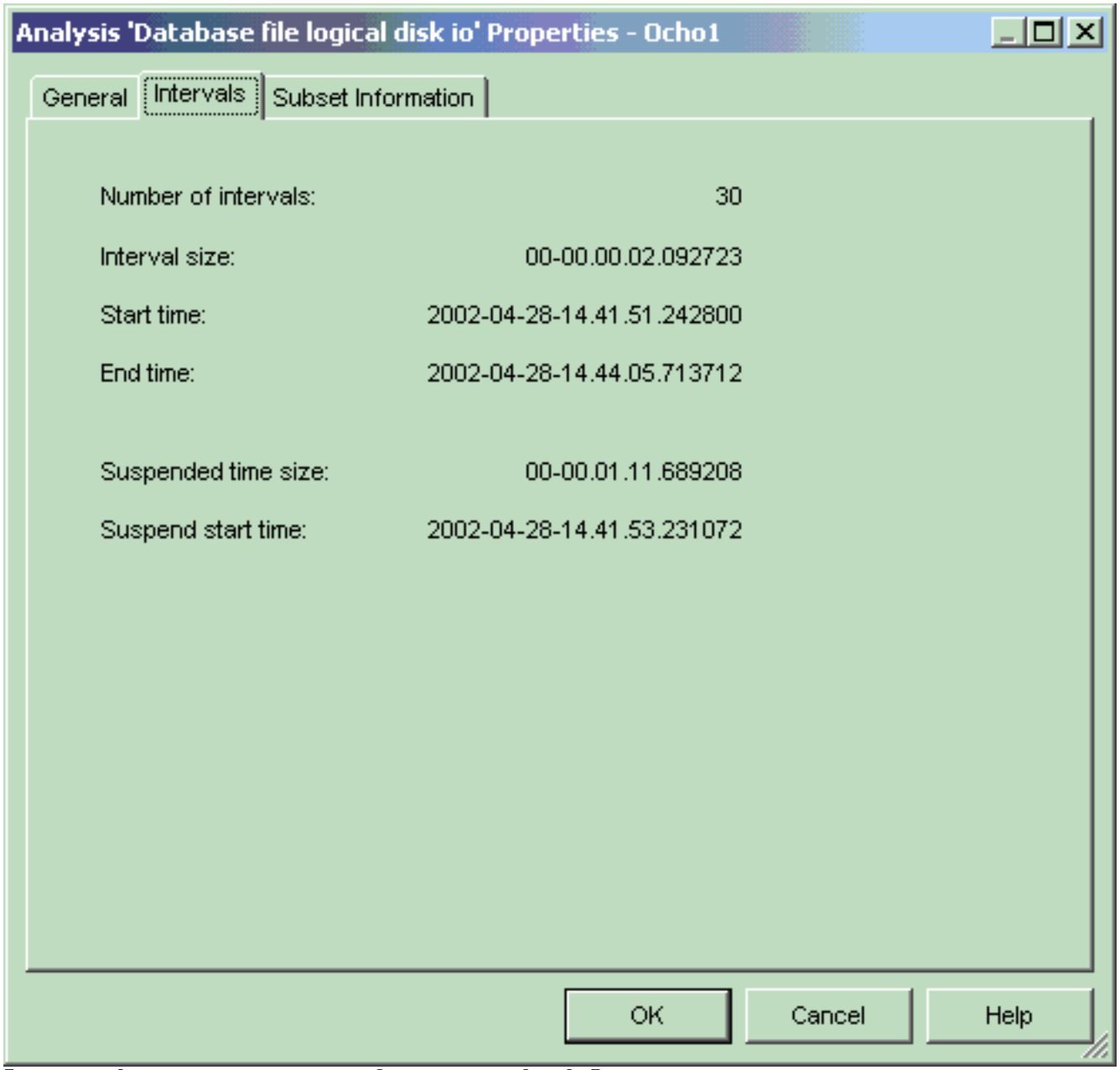
The following information is displayed on the General property page:

Field Name	Field Description
Analysis	The name of the PEX analysis. An analysis is a consolidation of data in a more useable format to help identify a specific problem type. An analysis can contain from 1 to n different reports. Some of these reports are graphable but all of the reports are viewable as table.
Collection name	PEX collection name.
Library	The library containing the collection and this analysis.
System	The system that the collection was initially created on. This can very likely be a different system than what you are currently connected to.
Status	Indicates whether or not the analysis is useable by iDoctor for iSeries. If the status is not 'Complete' then the analysis cannot be viewed in the data viewer.
Analysis Started	Date/time the analysis was started.
Output Database/File Member Name	This is the member name assigned to all reports for this analysis. Each analysis creates one or more G_* files in the collection library.
Show Job Log Button	If the analysis job is currently active, this button will be enabled. Click this button to display the active job log of the analysis creation job being executed on the server.
Job running analysis	The job name/user/number identifying the job currently processing (or that has finished processing) the analysis. If an error occurs in the analysis creation remote program call, you may look at the job log using the job information listed for error reporting purposes.

Table of Contents | Previous | Next

4.5.4.2 Intervals

The Intervals property page provides information about the total size of an interval in an analysis, the start and ending time of the analysis and information about the time the collection was suspended. If the collection was suspended the resulting analysis will have a gap in the data where information was not collected. The total suspended time is also provided on this page.



[Intervals Property Page for an analysis]

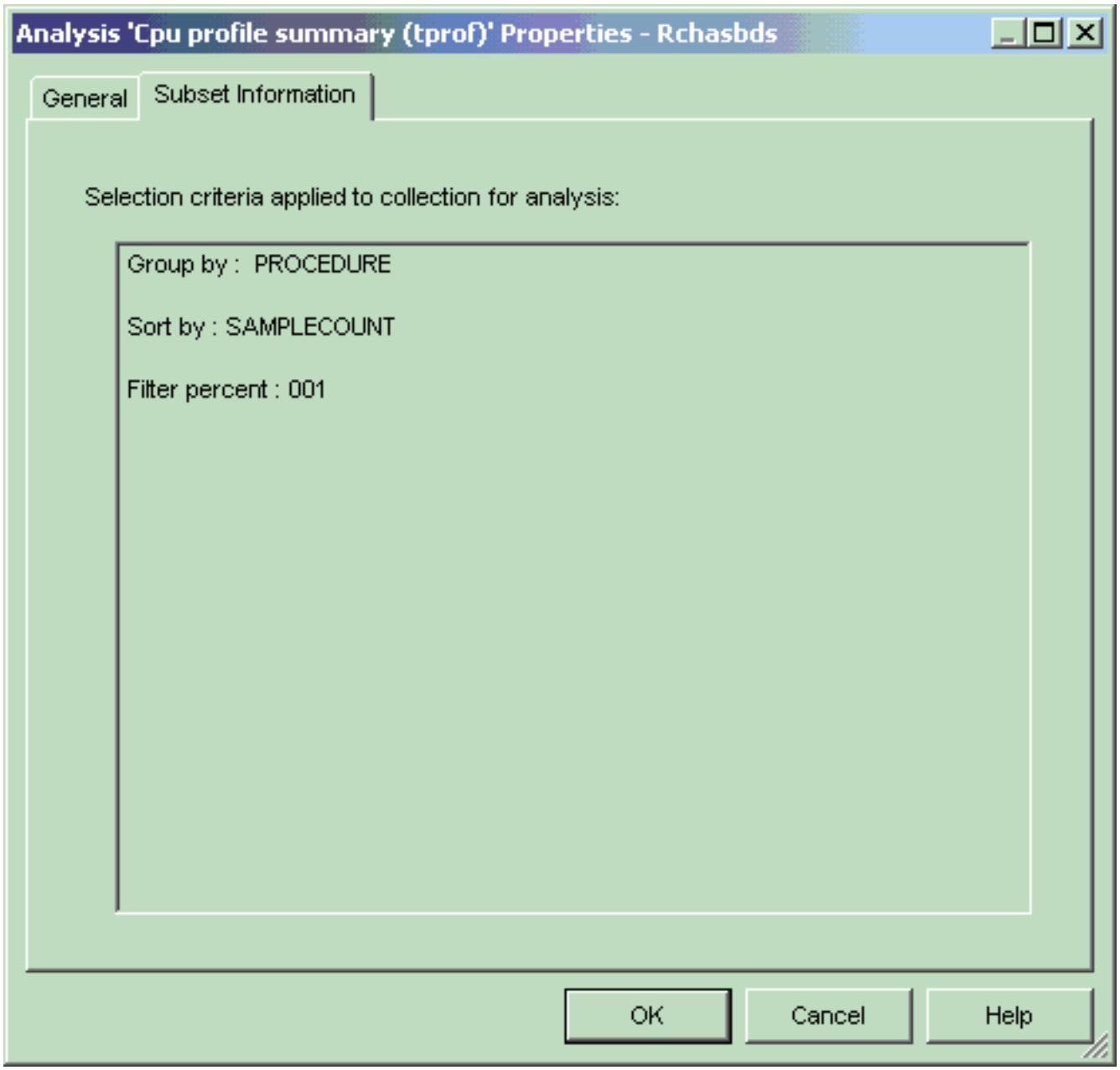
The following fields are available on this page:

Field Name	Field Description
Number of intervals	The total number of time intervals the analysis data has been broken up into.
Inteval size	The size of each time interval.
Start time	The start time for the data in the current analysis. This is either the subsetted start time or the collection start time.
End time	The end time for the data in the current analysis. This is either the subsetted end time or the collection end time.
Suspended time size	The total number of seconds or microseconds where the collecting of data did not occur.
Suspend start time	The time that the collection stopped collecting data.

Table of Contents | Previous | Next

4.5.4.3 Subset Information

The Subset Information property page provides detailed information about how an analysis was defined.



[Subset Information Property Page for an analysis]

The following information is displayed on the Subset Information property page:

Field Name	Field Description
List of selection criteria	This is a list of all selection criteria that took place for the analysis. It represents how the data in the collection was broken down to produce the analysis data.

Table of Contents | Previous | Next

4.6 Statistical Hierarchial Child Analysis Wizard

The Statistical Hierarchical Child Analysis Wizard allows a user to subset an existing analysis by drilling down further on a specific program, procedure or MI complex instruction that has been identified as a heavy resource consumer. This Wizard may be activated in the Data Viewer using the File | Create Child Analysis... menu. This option is only available when viewing the analysis type 'Stats hierarchical for one job/thread' in the Data Viewer.

Table of Contents | Previous | Next

4.6.1 Welcome Page

The Welcome Page simply informs the user that the wizard allows them to subset the statistical hierarchical analysis for a specific program, procedure or MI complex instruction. This page also lists information about the current analysis.

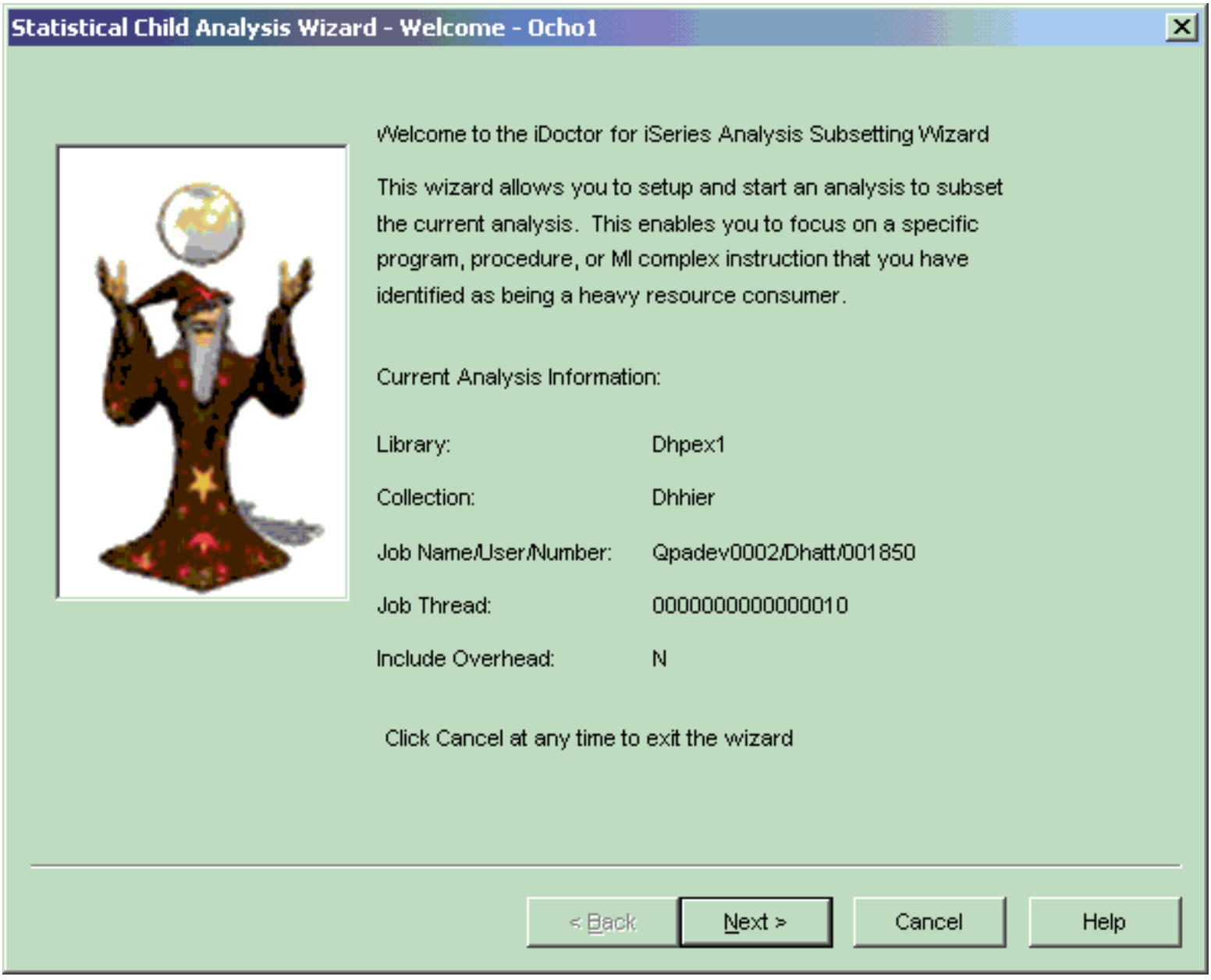


Table of Contents | Previous | Next

4.6.2 Select Analysis To Run Page

Child Analysis Wizard - Select Analysis To Run Page

The Select Analysis To Run Page is used to select the desired analysis to run on the current collection. At the time of this writing, the only available analysis type in the list is 'Statistics Hierarchical Subset on Pgm/Prc/MIInst'.

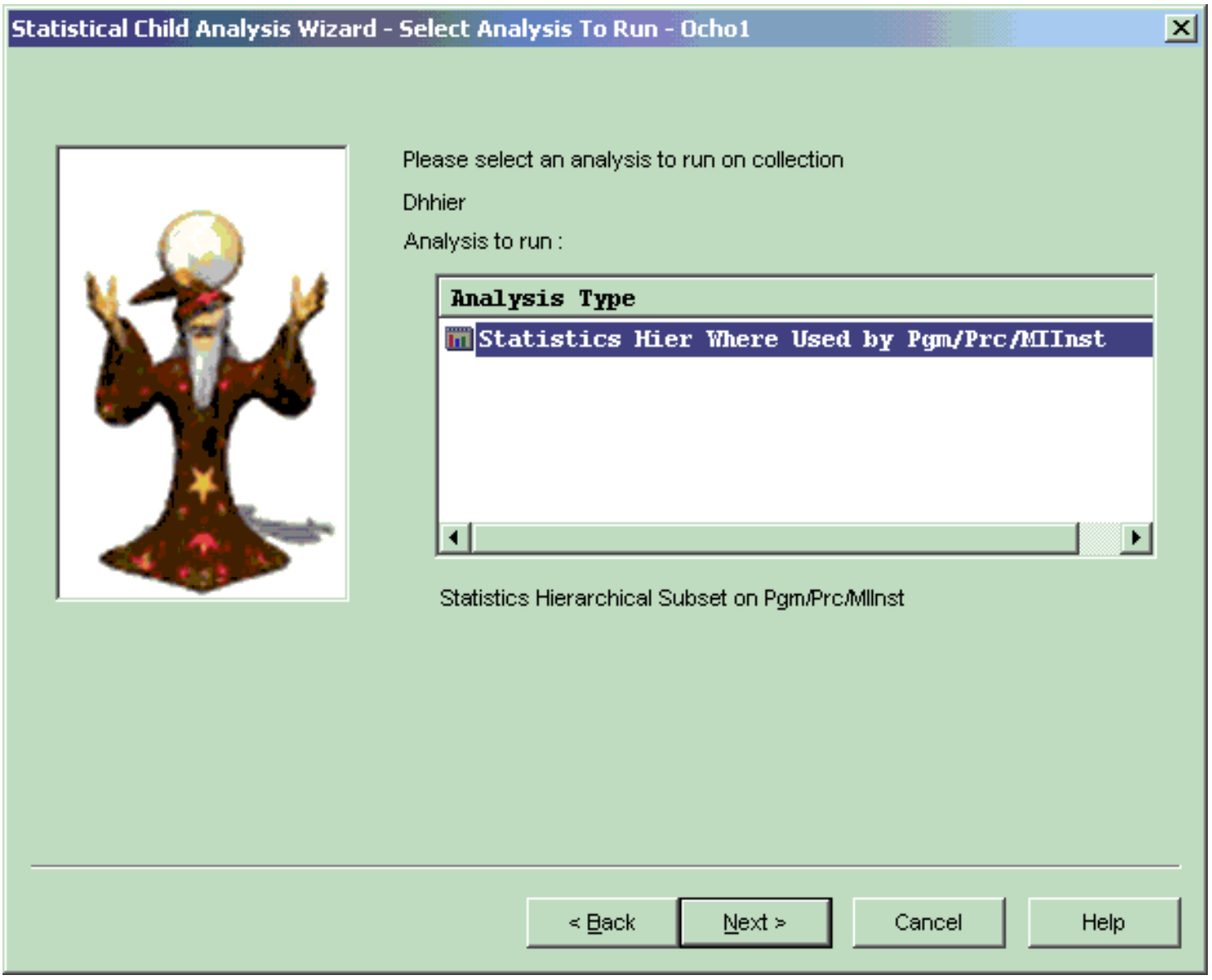


Table of Contents | Previous | Next

4.6.3 Subset Options Page

Child Analysis Wizard - Subset Options Page

The Subset Options Page is used to specify what program/procedure/MI Complex Instruction will be used to subset the analysis on. Use this page to specify the number of call levels in the resulting analysis and what filter percent is desired.

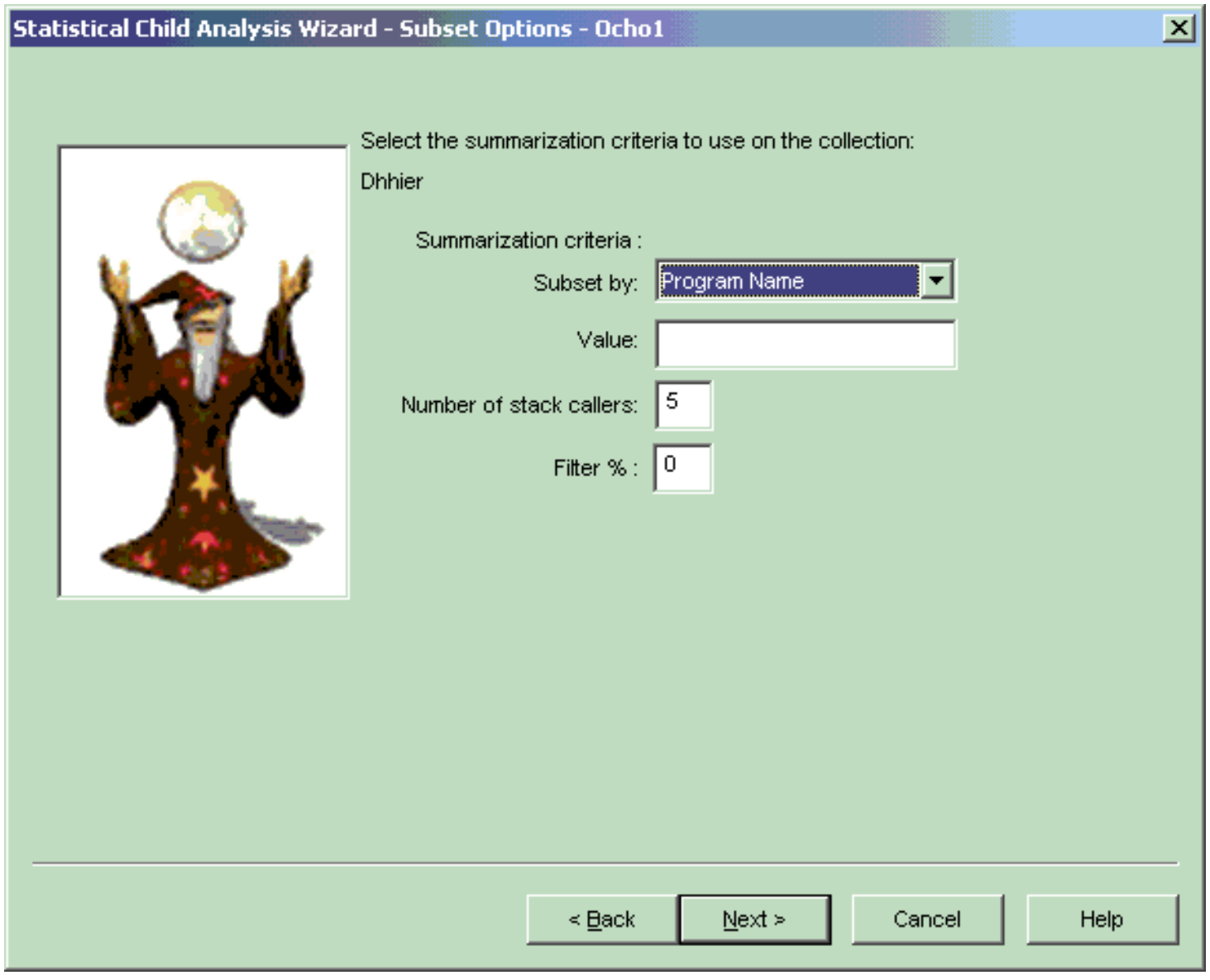


Table of Contents | Previous | Next

4.6.4 Summary Page

Child Analysis Wizard - Summary Page

The Summary Page lists all of the user's selection criteria identifying how the analysis will be created. Clicking Finish will submit the analysis request to the iSeries.

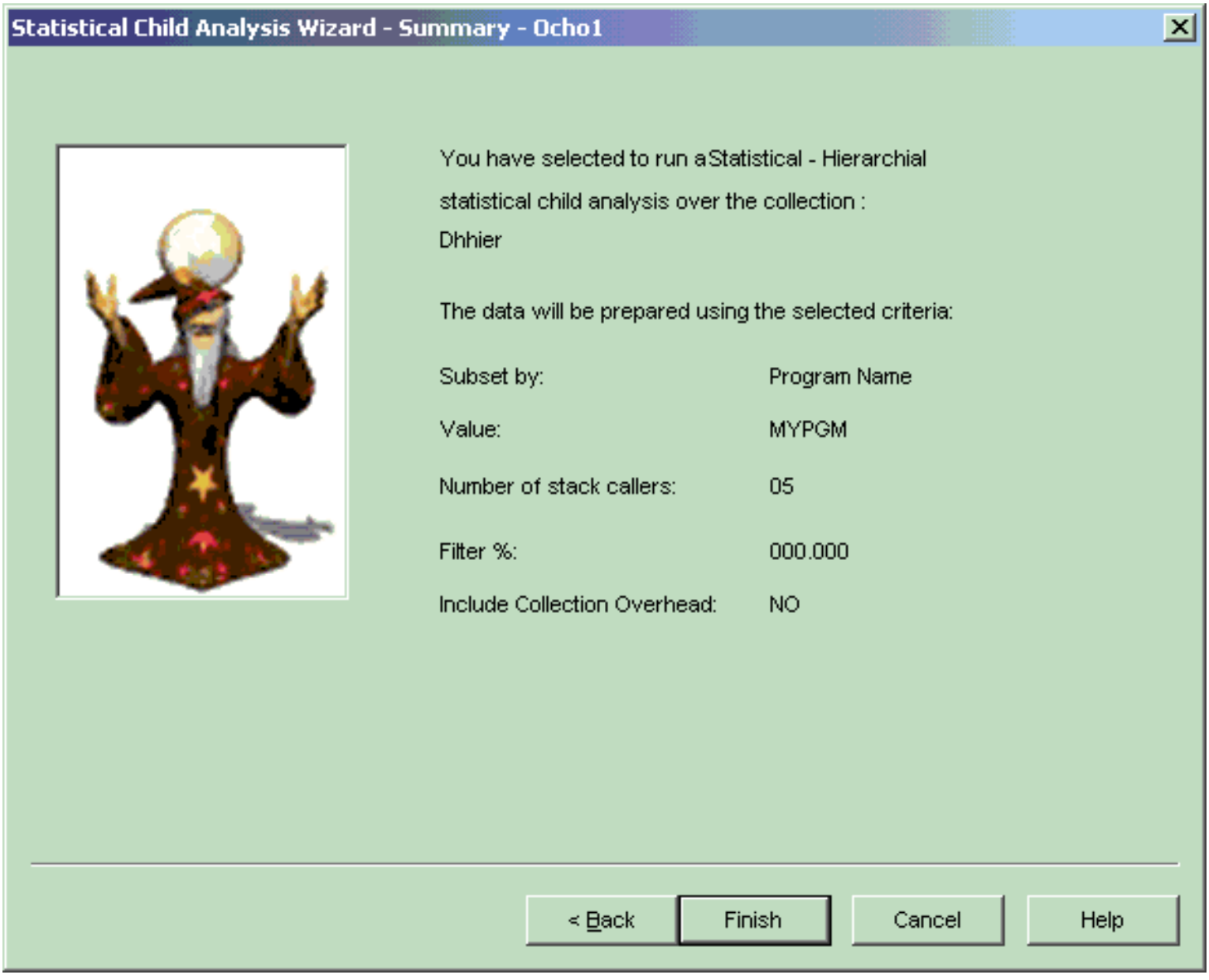


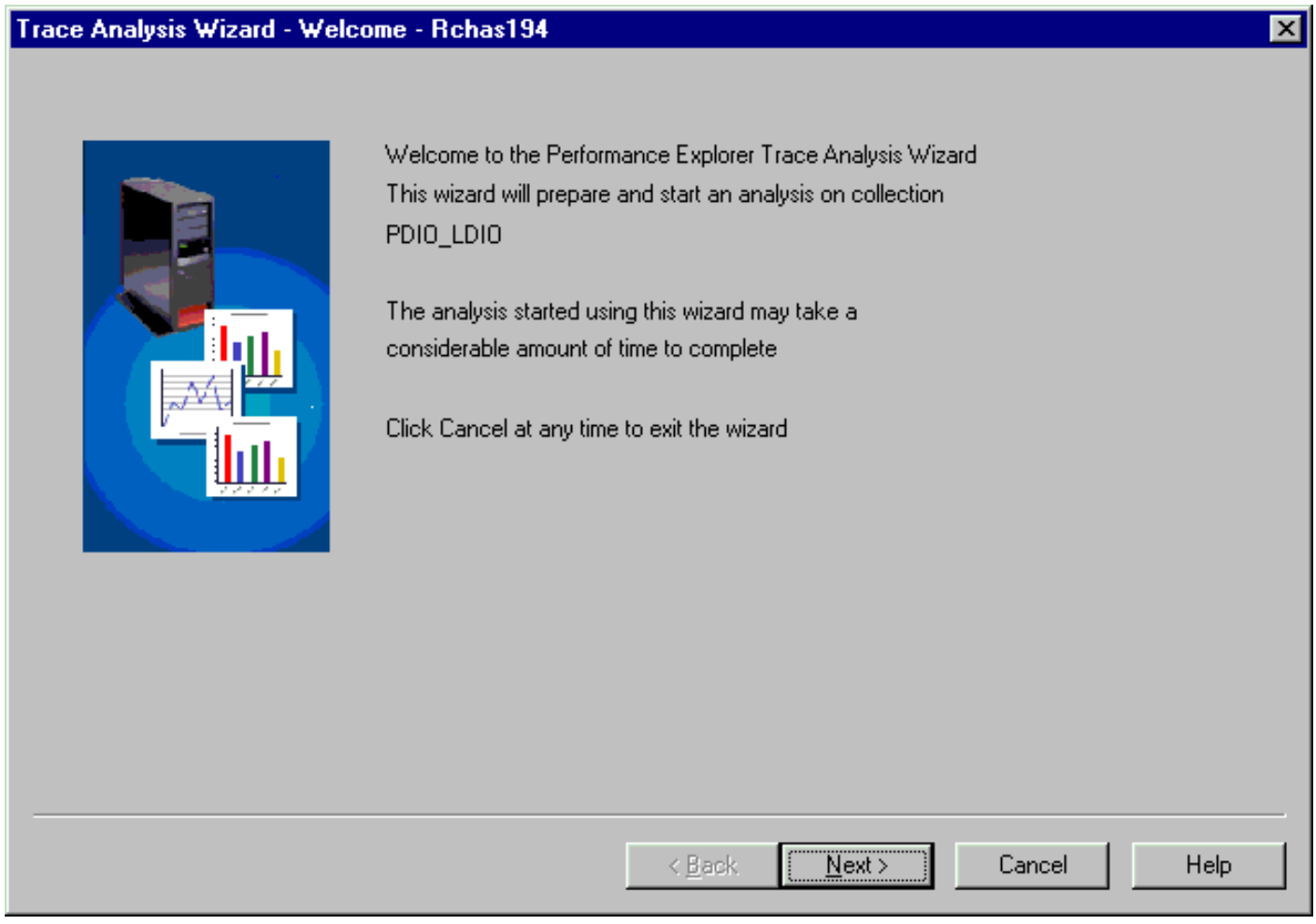
Table of Contents | Previous | Next

4.7 Create Trace Analysis Example

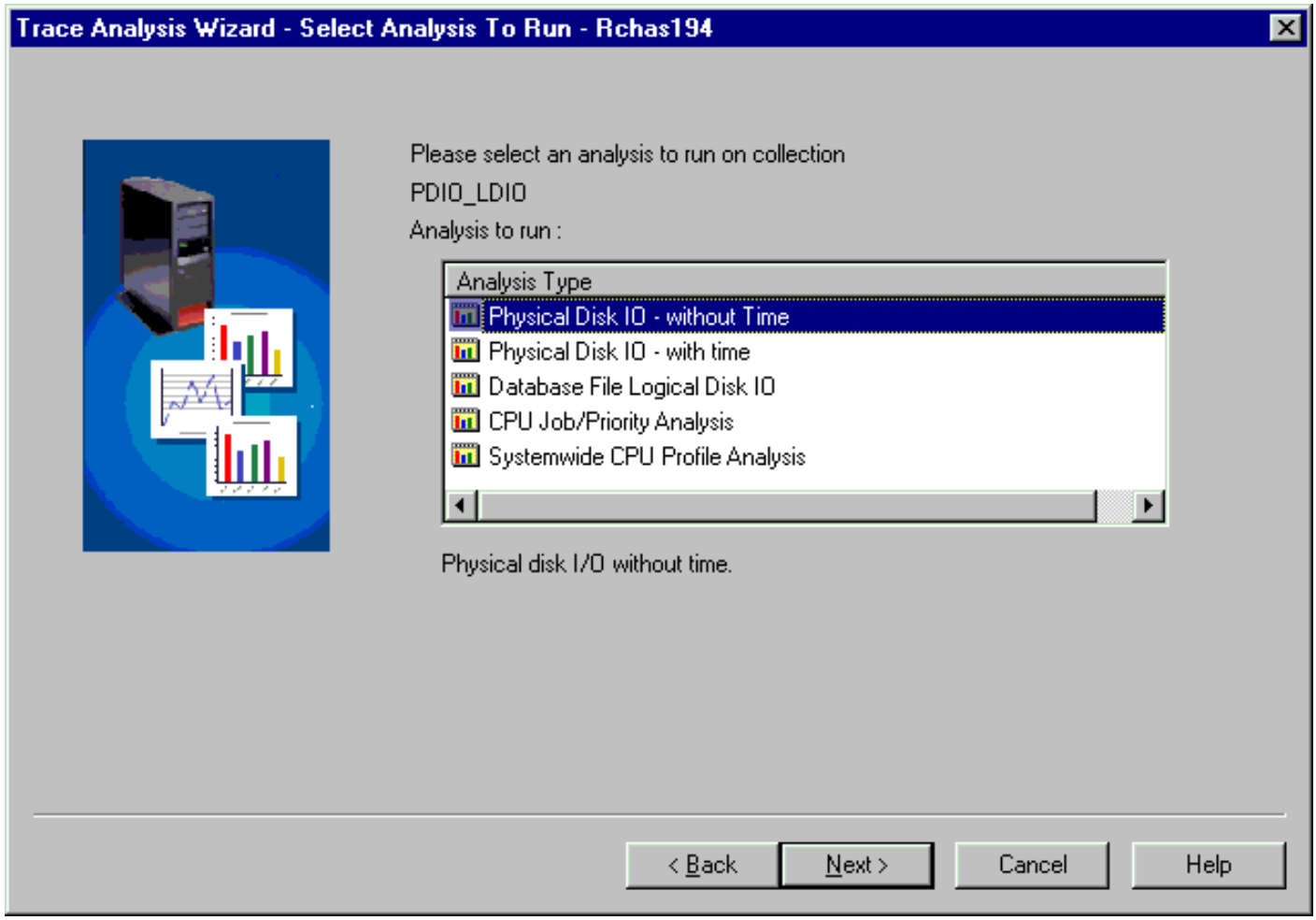
Introduction

The following example will walk through the steps necessary to create an analysis over a PEX Trace collection. The example assumes that a PEX Trace collection has been selected for analysis and the Trace Analysis Wizard has been started. This example will show how to create a Physical Disk IO - without Time analysis over a collection with subsetting for a specific job.

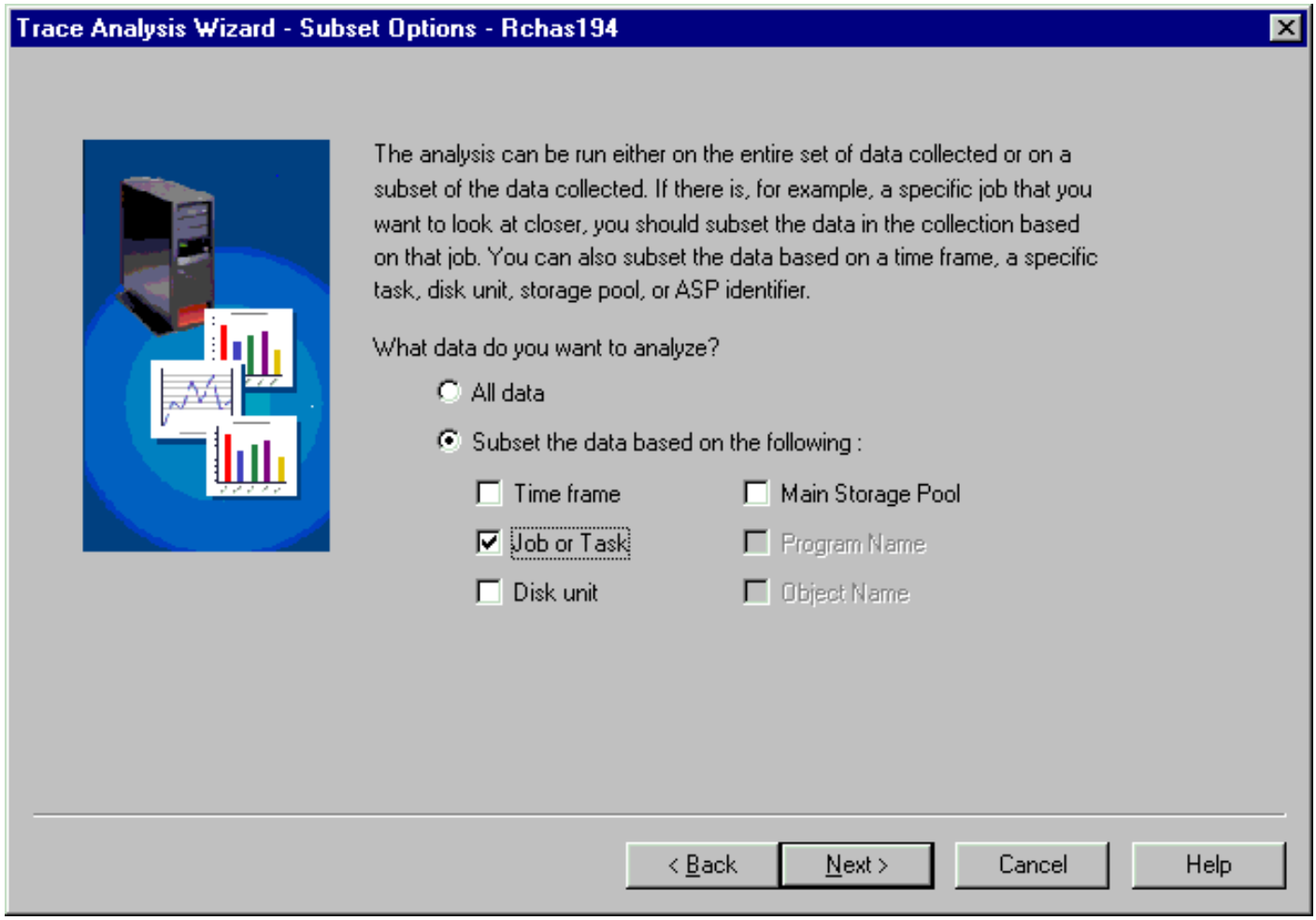
Step-by-step Instructions



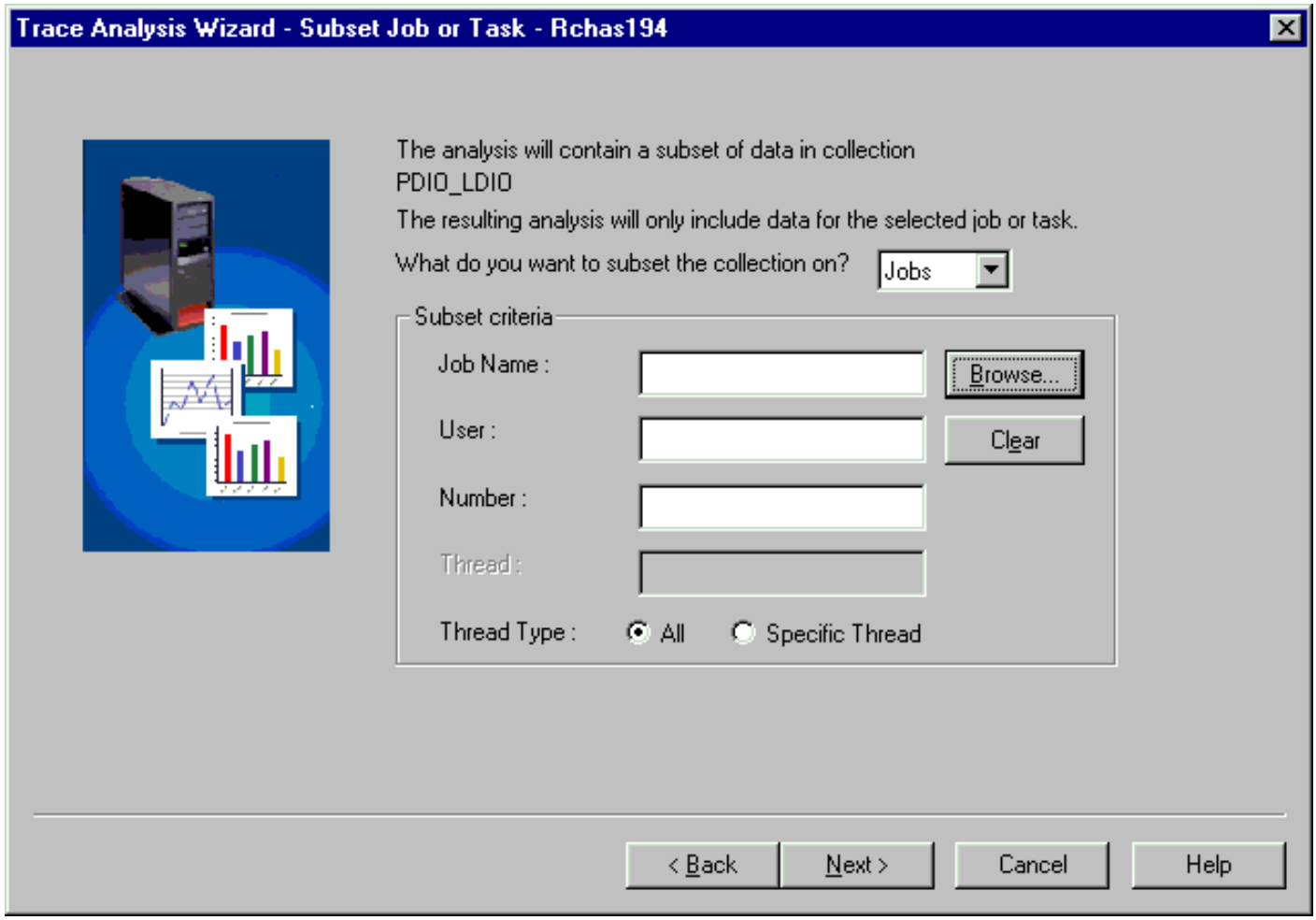
Step 1: Click the Next button to continue the wizard to the next page.



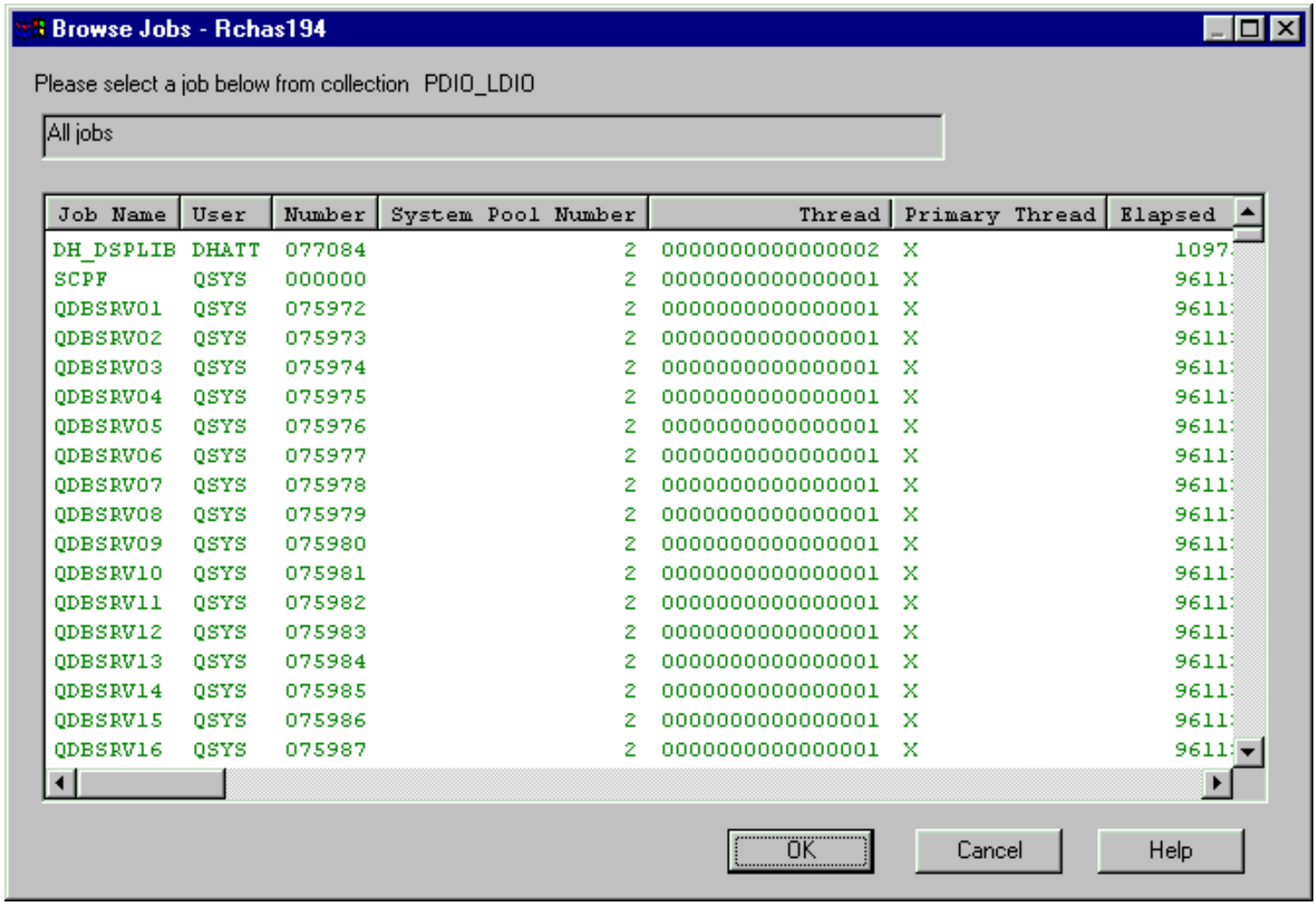
Step 2: Select the type of analysis to run on the collection. A description will be displayed underneath the list of analysis types as each analysis type in the list is selected. Click the next button to continue the Analysis Wizard. This example will show the creation of the analysis 'Physical Disk IO - without Time'.



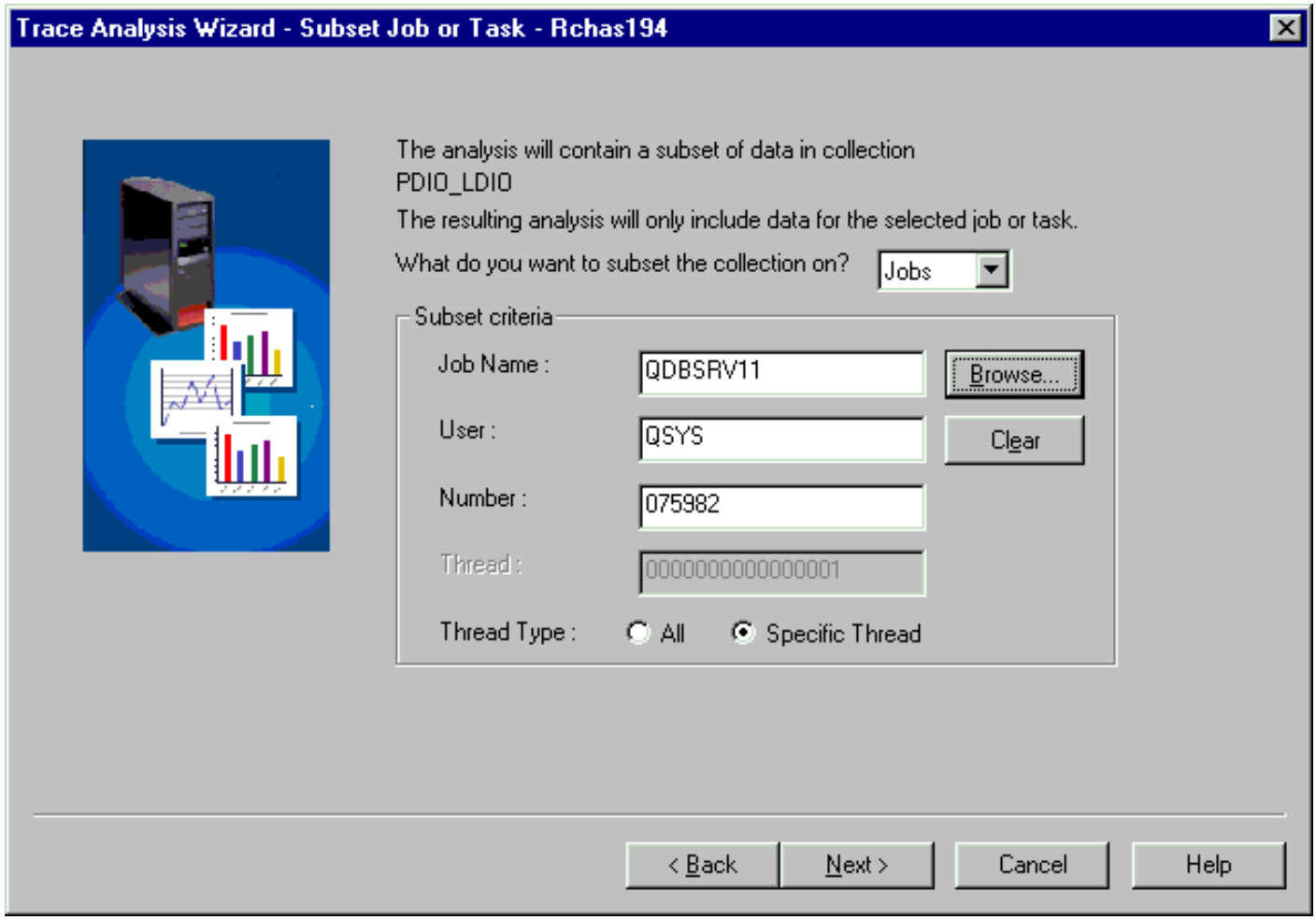
Step 3: Select the type of subsetting desired to apply on the PEX Trace collection. Each type of analysis type has a different set of subsetting available depending on settings applied to the collection when it was created. Check the desired types of subsetting for the analysis and click the Next button to continue the wizard. In this example the collection will be subsetted by a specific job.



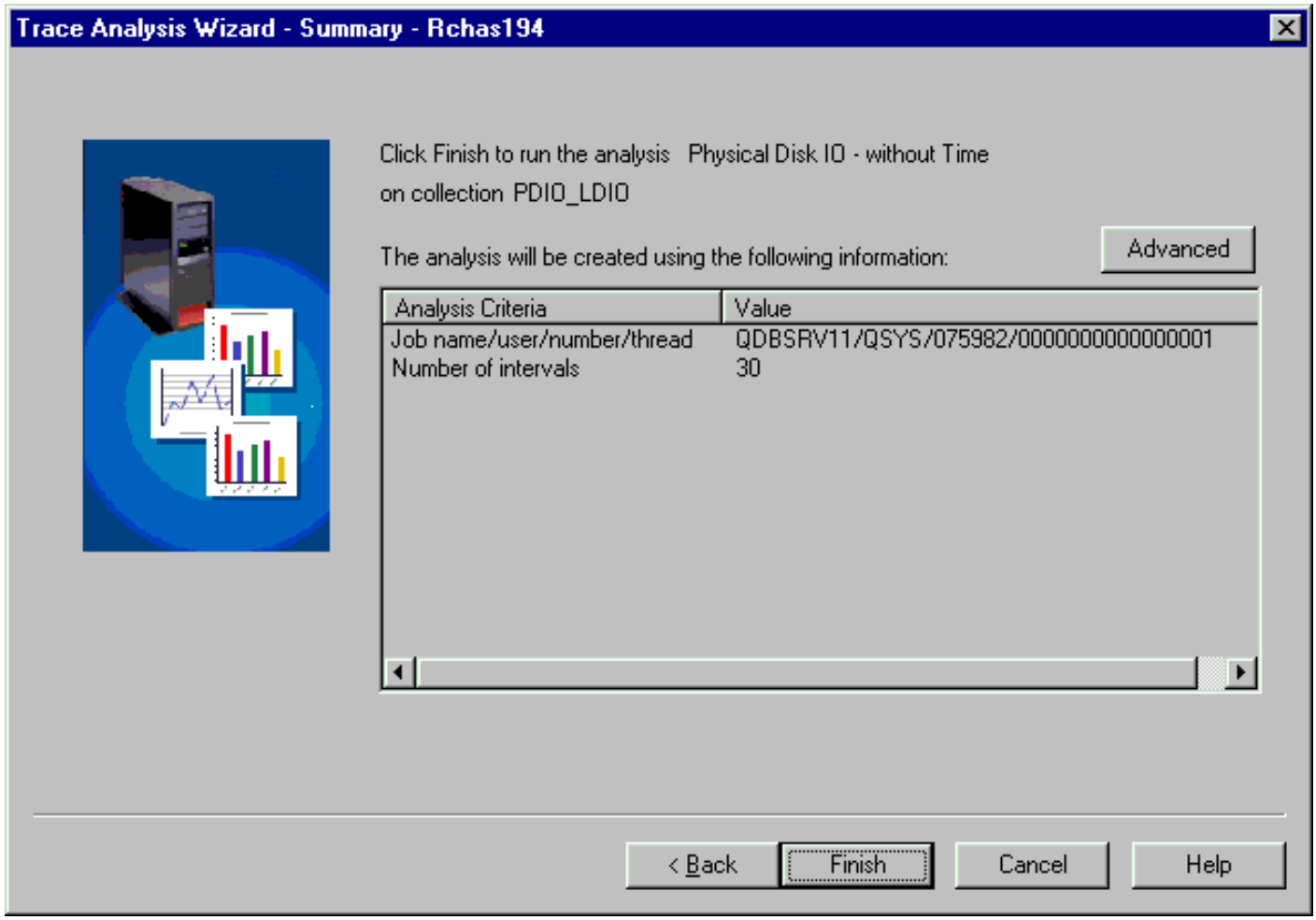
Step 4: Click the Browse... button to select a specific job in the collection. If Task subsetting is desired change the Job/Task drop down list value to Tasks and then click the Browse... button. For this example a list of jobs is displayed.



Step 5: Select a specific job from the list and click the OK button. The selected job's information will be filled into the Analysis Wizard. Click the Next button to continue the wizard.



Step 6: The analysis subset criteria is listed on the summary page. Click the Wizard's Finish button to submit the analysis request to the server for processing.



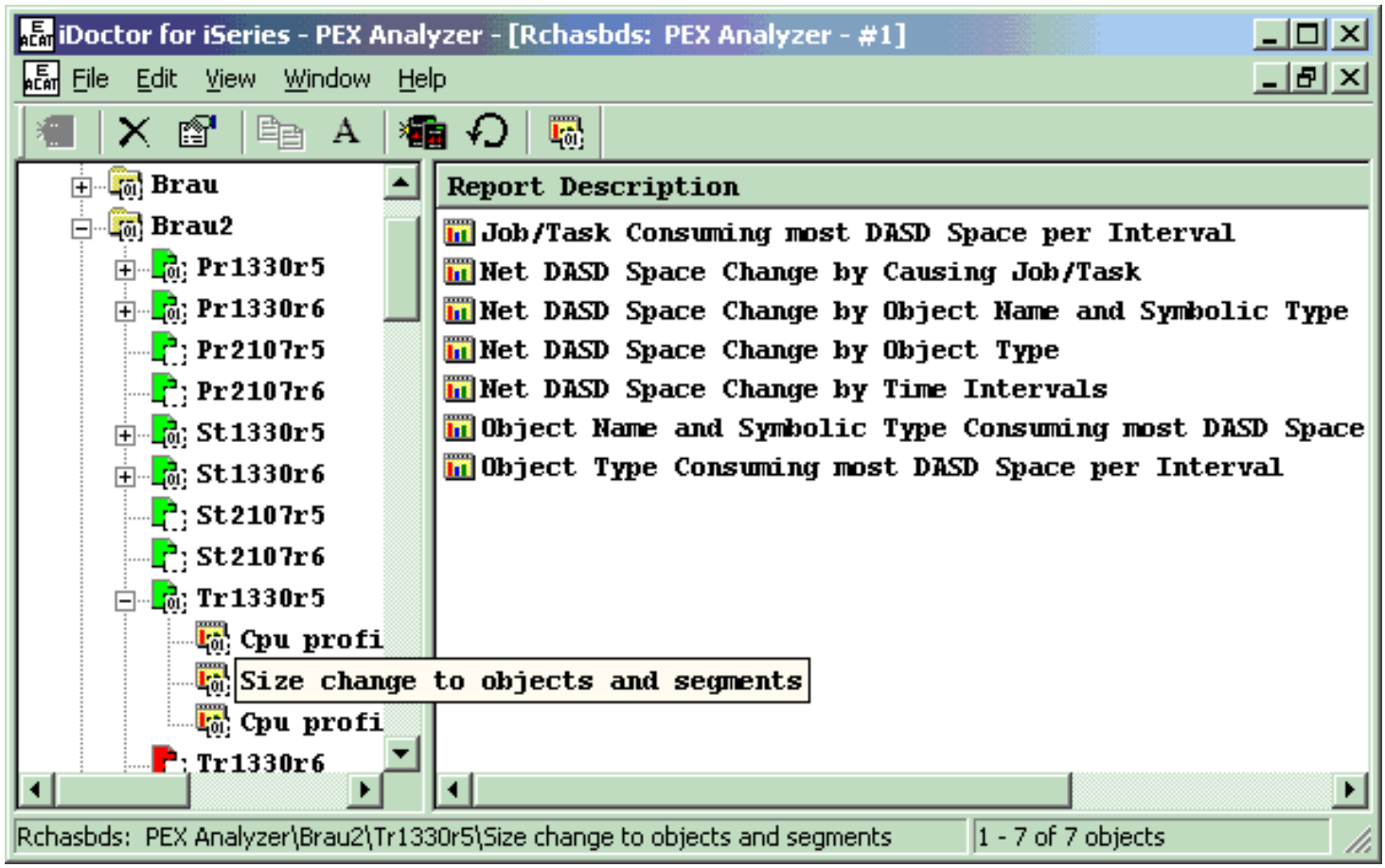
Step 7: At this point the request has been sent to the server to create the analysis. Use the View | Refresh menu (or F5) to refresh the list of analyses periodically until the analysis has been created. The status field will indicate 'In Progress' for the analysis being created. Once the analysis has been created the status field will show 'Complete'.

Step 8: The new analysis contains reports which may be viewed in the Data Viewer by either double-clicking the reports or right-clicking the desired reports within the analysis and selecting the appropriate Open Table or Open Graph menu.

Table of Contents | Previous | Next

4.8 Reports

Every analysis will contain zero or more reports. A report is a view over the PEX collection data in a format that will hopefully be helpful in determing the performance problem. You can open the report files in the Data Viewer as graphs or tables by right-clicking on the report icons.



Report Options

The table below outlines the different types of operations that may be performed on an report.

Menu Item	Description
Open Table(s)	Opens the selected report(s) into table view(s). You have the option to open the report into a new Data Viewer window or into an existing one via a dynamic submenu.
Open Graph(s)	Opens the selected reports into graph view(s). You have the option to open the report into a new Data Viewer window or into an existing one via a dynamic submenu.

Open Table(s) and Graph(s)	Opens the selected report into table and graph views. You have the option to open the report into a new Data Viewer window or into an existing one via a dynamic submenu.

Table of Contents Previous Next

Chapter 5 PEX Analyzer analysis reports

This chapter provides details about the analysis reports provided with PEX Analyzer.

Table of Contents | Previous | Next

5.1 Database file logical disk IO reports

This analysis shows all database Gets, Puts, Updates, Deletes at the file level showing the causing job/thread, file/member name, I/O type details (type of operation and relative record number), block sizes, file format name, timestamp and exception id.

If MIENTRY/MIEXIT events are included in the collection the analysis also shows the causing application program, elapsed time, physical disk I/O counts and CPU time of each LDIO in DB programs.

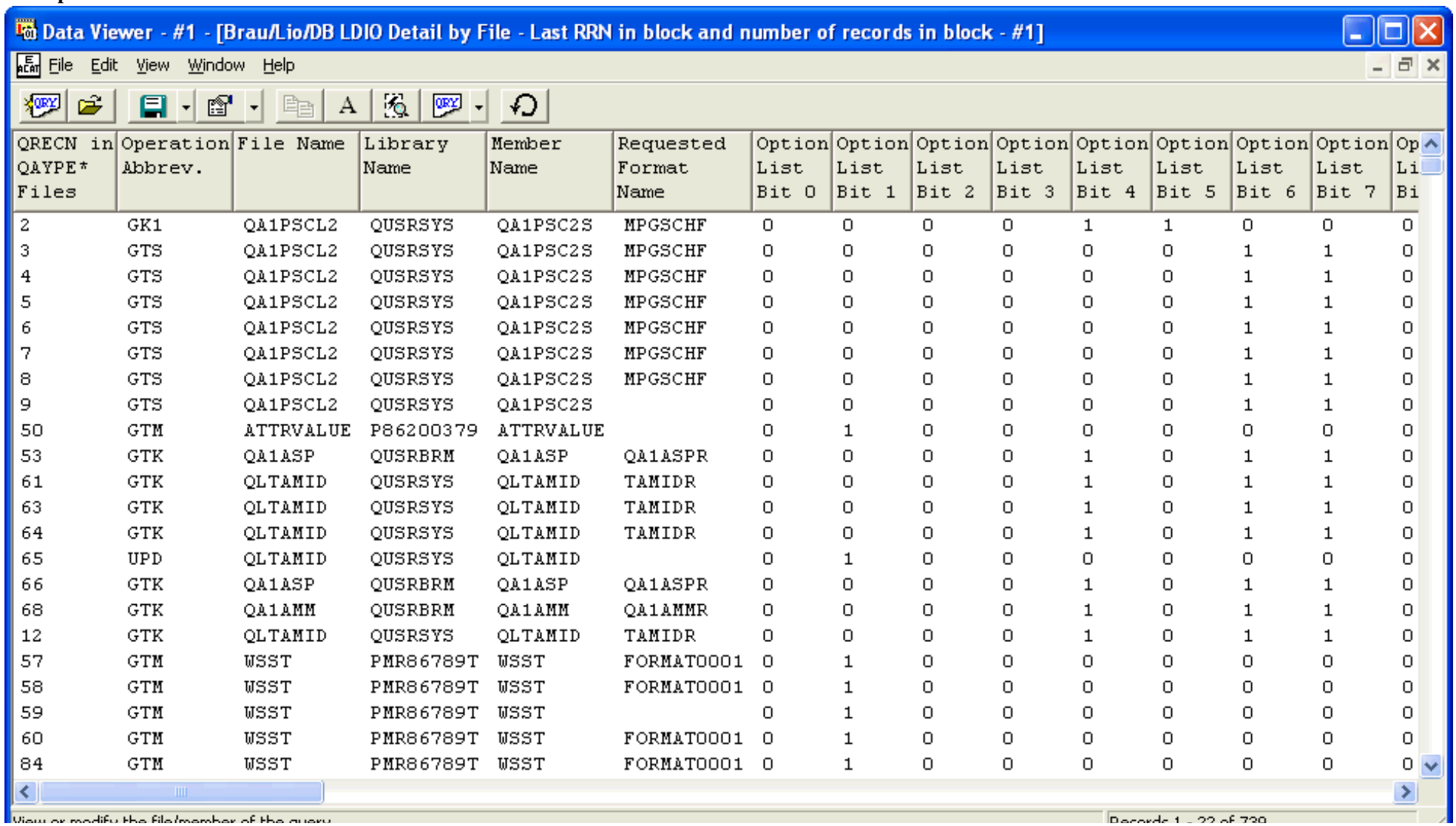
This section describes each of the reports available in this analysis.

Table of Contents | Previous | Next

5.1.1 DB LDIO Detail by File - Last RRN in block and number of records in block

Description: This report displays the LDIO related events captured in detail for each library/file/member occurring within the collection.

Example:



QRECN in QAYPE* Files	Operation Abbrev.	File Name	Library Name	Member Name	Requested Format Name	Option List Bit 0	Option List Bit 1	Option List Bit 2	Option List Bit 3	Option List Bit 4	Option List Bit 5	Option List Bit 6	Option List Bit 7	Op Li Bi
2	GK1	QA1PSCL2	QUSRSYS	QA1PSC2S	MPGSCHF	0	0	0	0	1	1	0	0	0
3	GTS	QA1PSCL2	QUSRSYS	QA1PSC2S	MPGSCHF	0	0	0	0	0	0	1	1	0
4	GTS	QA1PSCL2	QUSRSYS	QA1PSC2S	MPGSCHF	0	0	0	0	0	0	1	1	0
5	GTS	QA1PSCL2	QUSRSYS	QA1PSC2S	MPGSCHF	0	0	0	0	0	0	1	1	0
6	GTS	QA1PSCL2	QUSRSYS	QA1PSC2S	MPGSCHF	0	0	0	0	0	0	1	1	0
7	GTS	QA1PSCL2	QUSRSYS	QA1PSC2S	MPGSCHF	0	0	0	0	0	0	1	1	0
8	GTS	QA1PSCL2	QUSRSYS	QA1PSC2S	MPGSCHF	0	0	0	0	0	0	1	1	0
9	GTS	QA1PSCL2	QUSRSYS	QA1PSC2S		0	0	0	0	0	0	1	1	0
50	GTM	ATTRVALUE	P86200379	ATTRVALUE		0	1	0	0	0	0	0	0	0
53	GTK	QA1ASP	QUSRBRM	QA1ASP	QA1ASPR	0	0	0	0	1	0	1	1	0
61	GTK	QLTAMID	QUSRSYS	QLTAMID	TAMIDR	0	0	0	0	1	0	1	1	0
63	GTK	QLTAMID	QUSRSYS	QLTAMID	TAMIDR	0	0	0	0	1	0	1	1	0
64	GTK	QLTAMID	QUSRSYS	QLTAMID	TAMIDR	0	0	0	0	1	0	1	1	0
65	UPD	QLTAMID	QUSRSYS	QLTAMID		0	1	0	0	0	0	0	0	0
66	GTK	QA1ASP	QUSRBRM	QA1ASP	QA1ASPR	0	0	0	0	1	0	1	1	0
68	GTK	QA1AMM	QUSRBRM	QA1AMM	QA1AMMR	0	0	0	0	1	0	1	1	0
12	GTK	QLTAMID	QUSRSYS	QLTAMID	TAMIDR	0	0	0	0	1	0	1	1	0
57	GTM	WSST	PMR86789T	WSST	FORMAT0001	0	1	0	0	0	0	0	0	0
58	GTM	WSST	PMR86789T	WSST	FORMAT0001	0	1	0	0	0	0	0	0	0
59	GTM	WSST	PMR86789T	WSST		0	1	0	0	0	0	0	0	0
60	GTM	WSST	PMR86789T	WSST	FORMAT0001	0	1	0	0	0	0	0	0	0
84	GTM	WSST	PMR86789T	WSST	FORMAT0001	0	1	0	0	0	0	0	0	0

Table of Contents | Previous | Next

5.1.2 DB File Detail LDIO by File-Library - No Intervals

Description: This report shows the files with the most "logical disk I/O" (LDIO) activity during the collection. For each of these files, it also shows the type of LDIO and what percentage of the total collection's LDIO each file represents.

Each color in a bar graph represents a type of LDIO. Clicking one of these colors brings up additional details about the LDIO, including the full file name and the amount of physical disk I/O (PDIO) that occurred on behalf of this LDIO.

Example:

Data Viewer - #1 - [Brau/Lio/Local DB File Detail LDIO by File-Library - No Intervals - #1]

File Edit View Window Help

File-Library		File-Lib LDIO Total	File-Lib LDIO Rate per Sec	File-Lib LDIO Pct System	LDIO Type	File/LDIO Type Total	File/LDIO Type Rate per Sec
ENUMBLD	QTEMP	166	9.481727	22.46	GTM	2	.114238
ENUMBLD	QTEMP	166	9.481727	22.46	GTS	163	9.310370
ENUMBLD	QTEMP	166	9.481727	22.46	PTM	1	.057119
ENUMBLD2	QTEMP	164	9.367489	22.19	GTM	163	9.310370
ENUMBLD2	QTEMP	164	9.367489	22.19	PTM	1	.057119
QPYRTJWATB	QPYRTJW	162	9.253251	21.92	PUT	162	9.253251
QAPZGRP	QUSRSYS	81	4.626626	10.96	GTK	81	4.626626
QPYRTJWAD	QPYRTJW	32	1.827803	4.33	PUT	32	1.827803
BUCKETBLD	QTEMP	32	1.827803	4.33	GTS	32	1.827803
WSST	PMR86789T	20	1.142377	2.71	GTM	20	1.142377
QADBXRMTNM	QSYS	14	.799664	1.89	GTK	14	.799664
QAPZPTF	QUSRSYS	12	.685426	1.62	GTK	12	.685426
QAPZREQ2	QUSRSYS	10	.571188	1.35	GTK	10	.571188
QASQRESL	QSYS2	8	.456951	1.08	GTK	8	.456951
QA1PSCL2	QUSRSYS	8	.456951	1.08	GK1	1	.057119
QA1PSCL2	QUSRSYS	8	.456951	1.08	GTS	7	.399832
QA52OPACI	BRAU	6	.342713	.81	GTK	1	.057119

Records 1 - 17 of 29

Graph Type: stacked vertical bar
X-axis: File and library name. The entire field is shown by placing the mouse over a bar.
Y-Axis: This value is the number of LDIO ops occurring of each type per second for each file. Each color represents a

different type of LDIO.
Second Y-Axis: This line shows how much each file/library's LDIO ops contributed to the total system LDIOs.

Example:

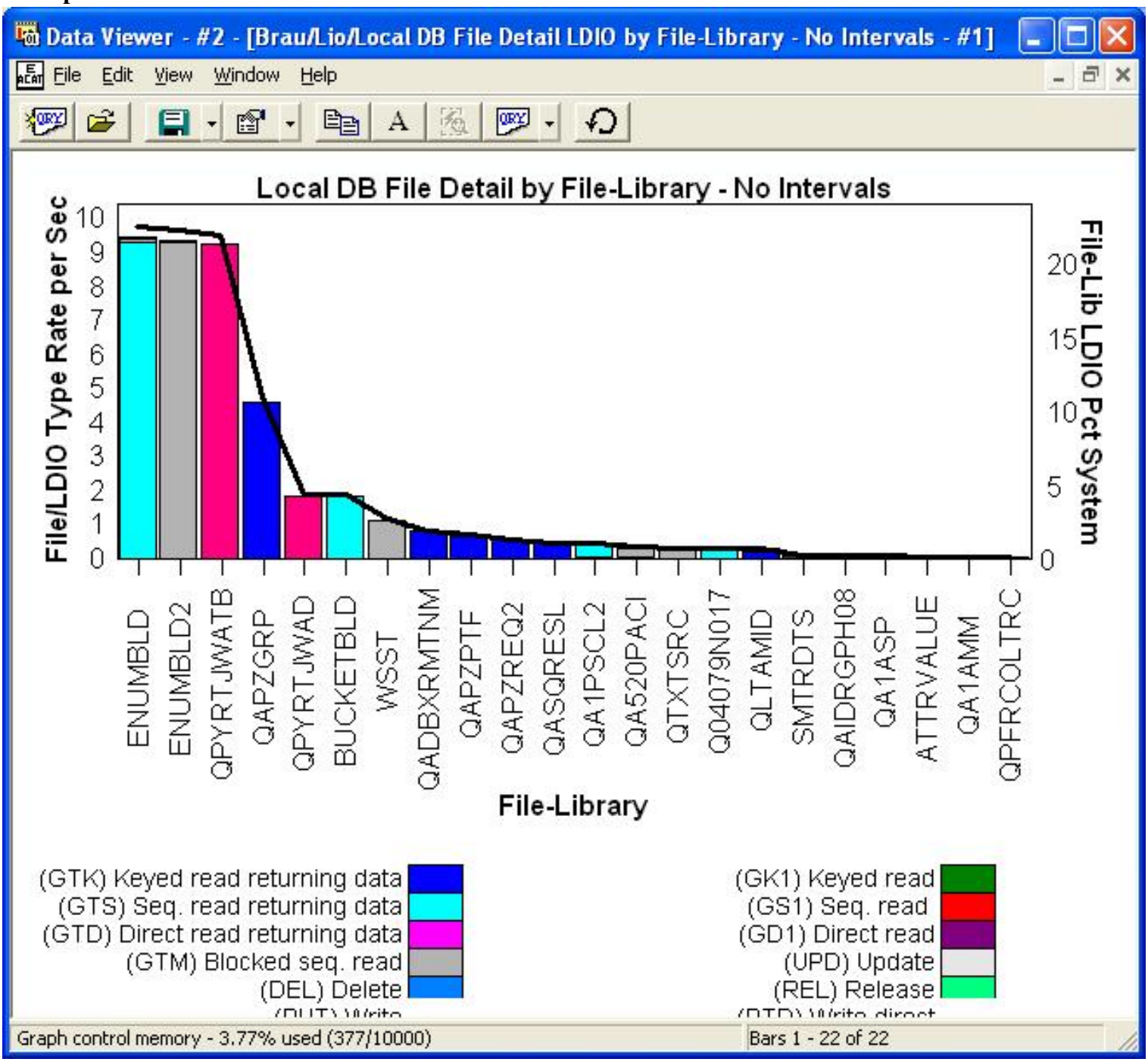
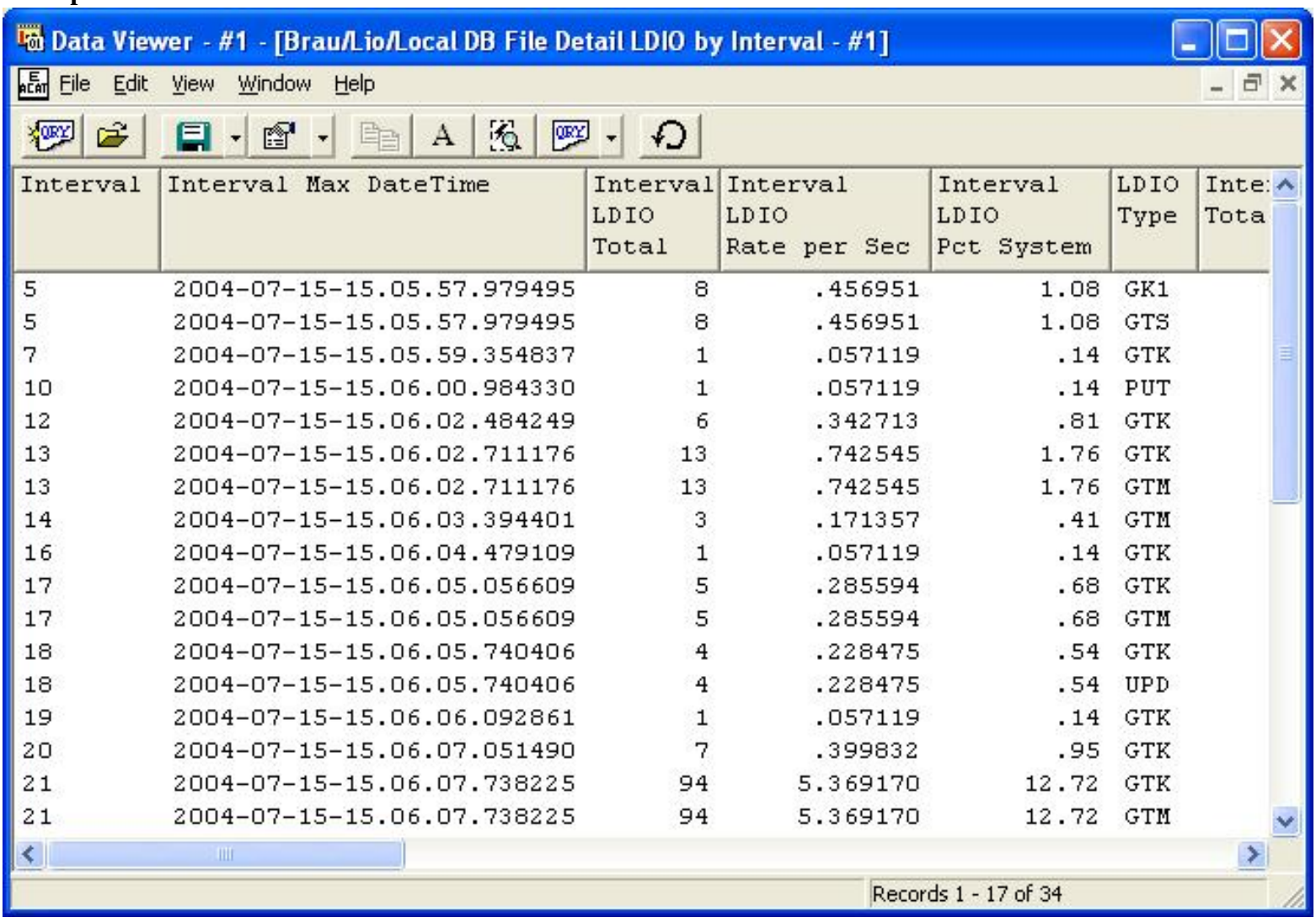


Table of Contents | Previous | Next

5.1.3 DB File Detail LDIO by Interval

Description: This report shows the amount and type of "logical disk I/O" (LDIO) activity for each time interval in the collection. The percentage each interval's LDIO represents of the collection's total LDIO is also shown. Each color in a bar represents a type of LDIO. Clicking one of these colors brings up additional details about the LDIO for the interval selected.

Example:



Interval	Interval Max DateTime	Interval LDIO Total	Interval LDIO Rate per Sec	Interval LDIO Pct System	LDIO Type	Inte: Tota
5	2004-07-15-15.05.57.979495	8	.456951	1.08	GK1	
5	2004-07-15-15.05.57.979495	8	.456951	1.08	GTS	
7	2004-07-15-15.05.59.354837	1	.057119	.14	GTK	
10	2004-07-15-15.06.00.984330	1	.057119	.14	PUT	
12	2004-07-15-15.06.02.484249	6	.342713	.81	GTK	
13	2004-07-15-15.06.02.711176	13	.742545	1.76	GTK	
13	2004-07-15-15.06.02.711176	13	.742545	1.76	GTM	
14	2004-07-15-15.06.03.394401	3	.171357	.41	GTM	
16	2004-07-15-15.06.04.479109	1	.057119	.14	GTK	
17	2004-07-15-15.06.05.056609	5	.285594	.68	GTK	
17	2004-07-15-15.06.05.056609	5	.285594	.68	GTM	
18	2004-07-15-15.06.05.740406	4	.228475	.54	GTK	
18	2004-07-15-15.06.05.740406	4	.228475	.54	UPD	
19	2004-07-15-15.06.06.092861	1	.057119	.14	GTK	
20	2004-07-15-15.06.07.051490	7	.399832	.95	GTK	
21	2004-07-15-15.06.07.738225	94	5.369170	12.72	GTK	
21	2004-07-15-15.06.07.738225	94	5.369170	12.72	GTM	

Graph Type: stacked vertical bar
X-axis: Interval number
Y-Axis: This value is the number of LDIO ops occurring of each type per second within each interval. Each color represents a different type of LDIO.
Second Y-Axis: This line shows how much each interval's total LDIOs for the collection contributed to the total system LDIOs.

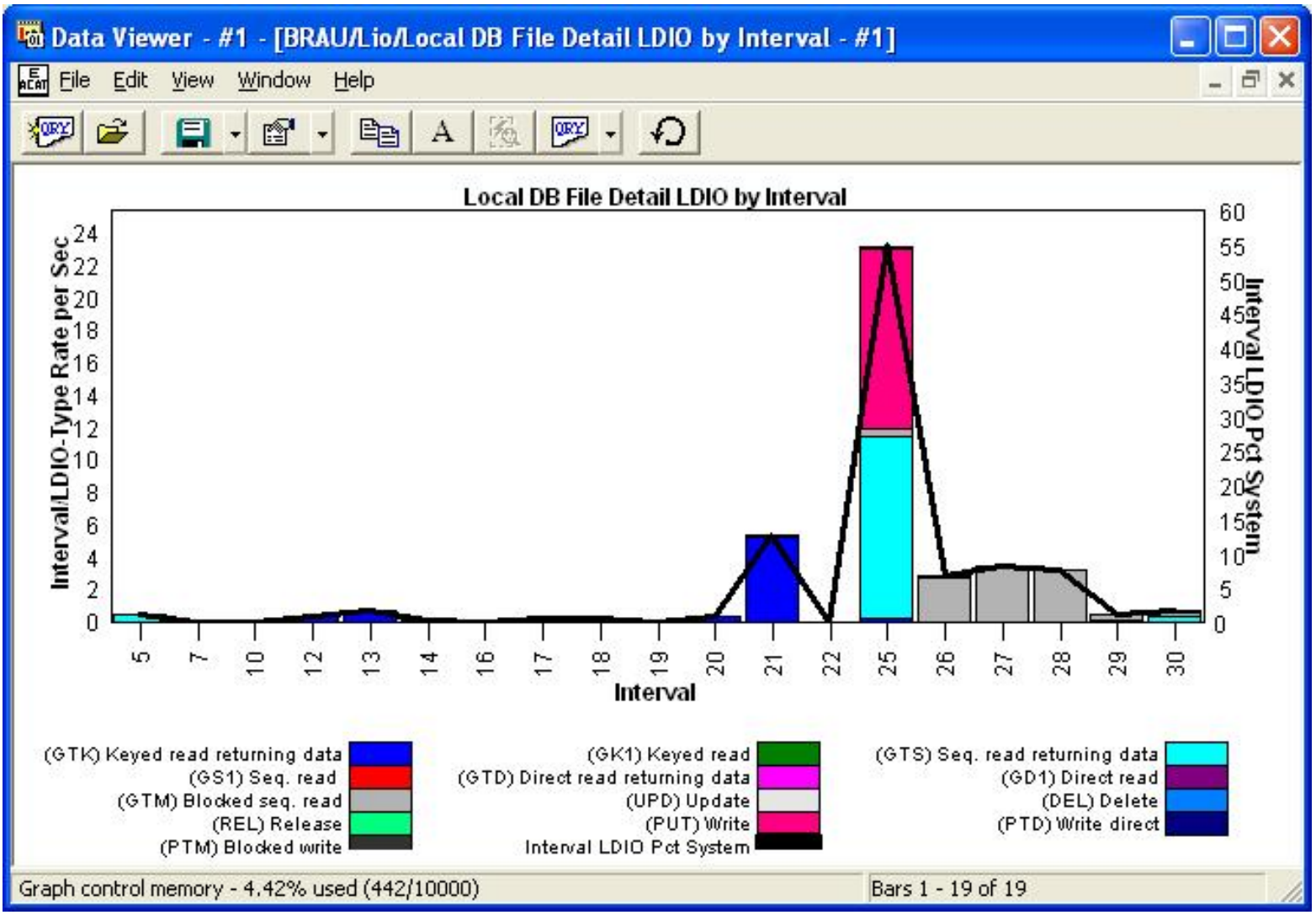

Data Viewer - #1 - [BRAU/Lio/Local DB File Detail LDIO by Interval - #1]
File Edit View Window Help
Local DB File Detail LDIO by Interval
Interval/LDIO-Type Rate per Sec
Interval LDIO Pct System
Interval
5 7 10 12 13 14 16 17 18 19 20 21 22 25 26 27 28 29 30
(GTK) Keyed read returning data
(GS1) Seq. read
(GTM) Blocked seq. read
(REL) Release
(PTM) Blocked write
(GK1) Keyed read
(GTD) Direct read returning data
(UPD) Update
(PUT) Write
Interval LDIO Pct System
(GTS) Seq. read returning data
(GD1) Direct read
(DEL) Delete
(PTD) Write direct
Graph control memory - 4.42% used (442/10000)
Bars 1 - 19 of 19

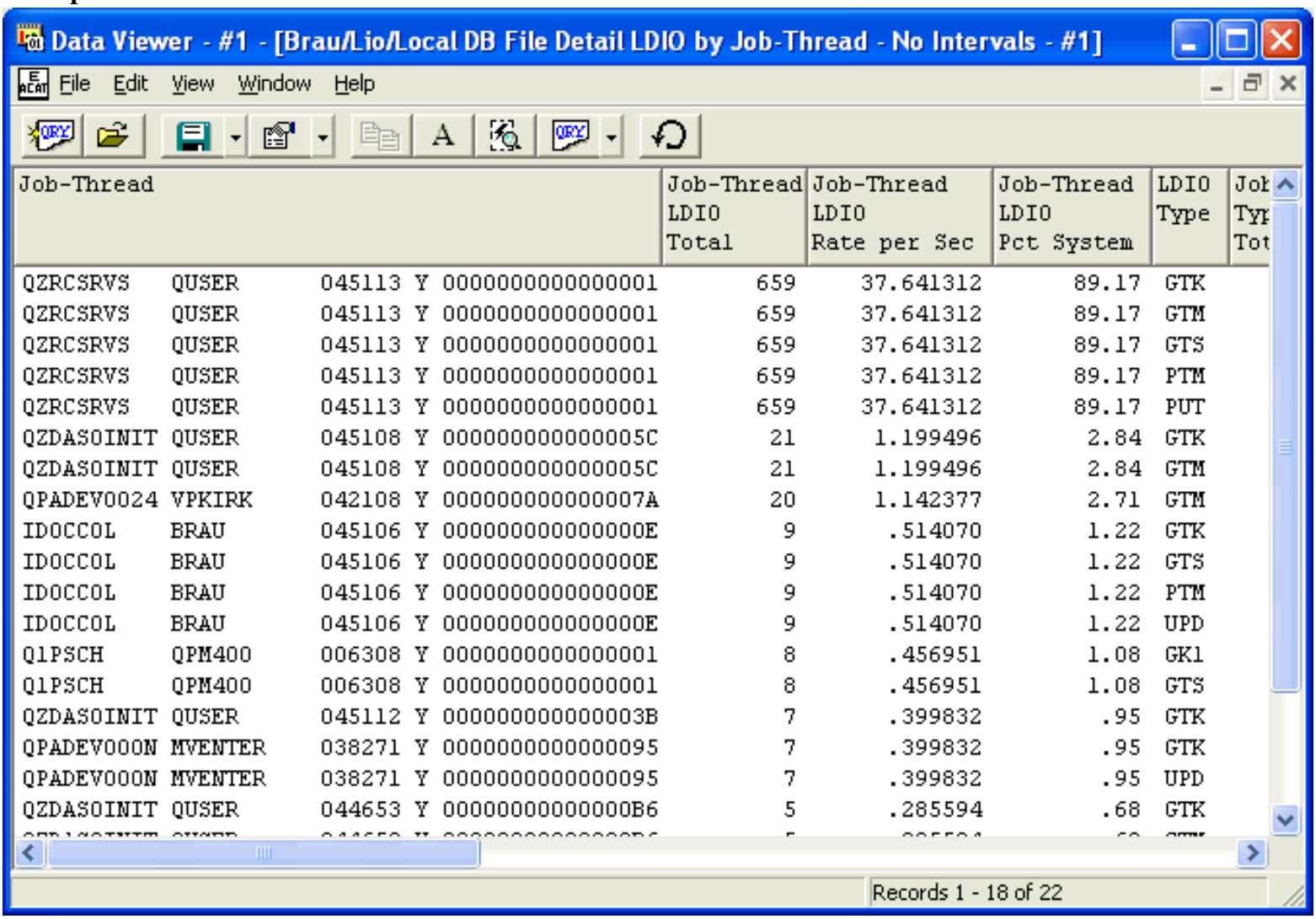
Table of Contents | Previous | Next

5.1.4 DB File Detail LDIO by Job-thread - No Intervals

Description: This graph shows the jobs or threads with the most "logical disk I/O" (LDIO) activity during the collection.

For each of these jobs or threads, it also shows the type of LDIO and what percentage of the total collection's LDIO each job/thread represents. Each color in a bar represents a type of LDIO. Clicking one of these colors brings up additional details about the LDIO for the job-thread selected.

Example:



Graph Type: stacked vertical bar
X-axis: Fully qualified job name and thread id. The entire field is shown by placing the mouse over a bar.
Y-Axis: This value is the number of LDIO ops occurring of each type per second for a job within the collection. Each color represents a different type of LDIO.

Second Y-Axis: This line shows how much each job's total LDIO ops contributed to the total system LDIOs.

Example:

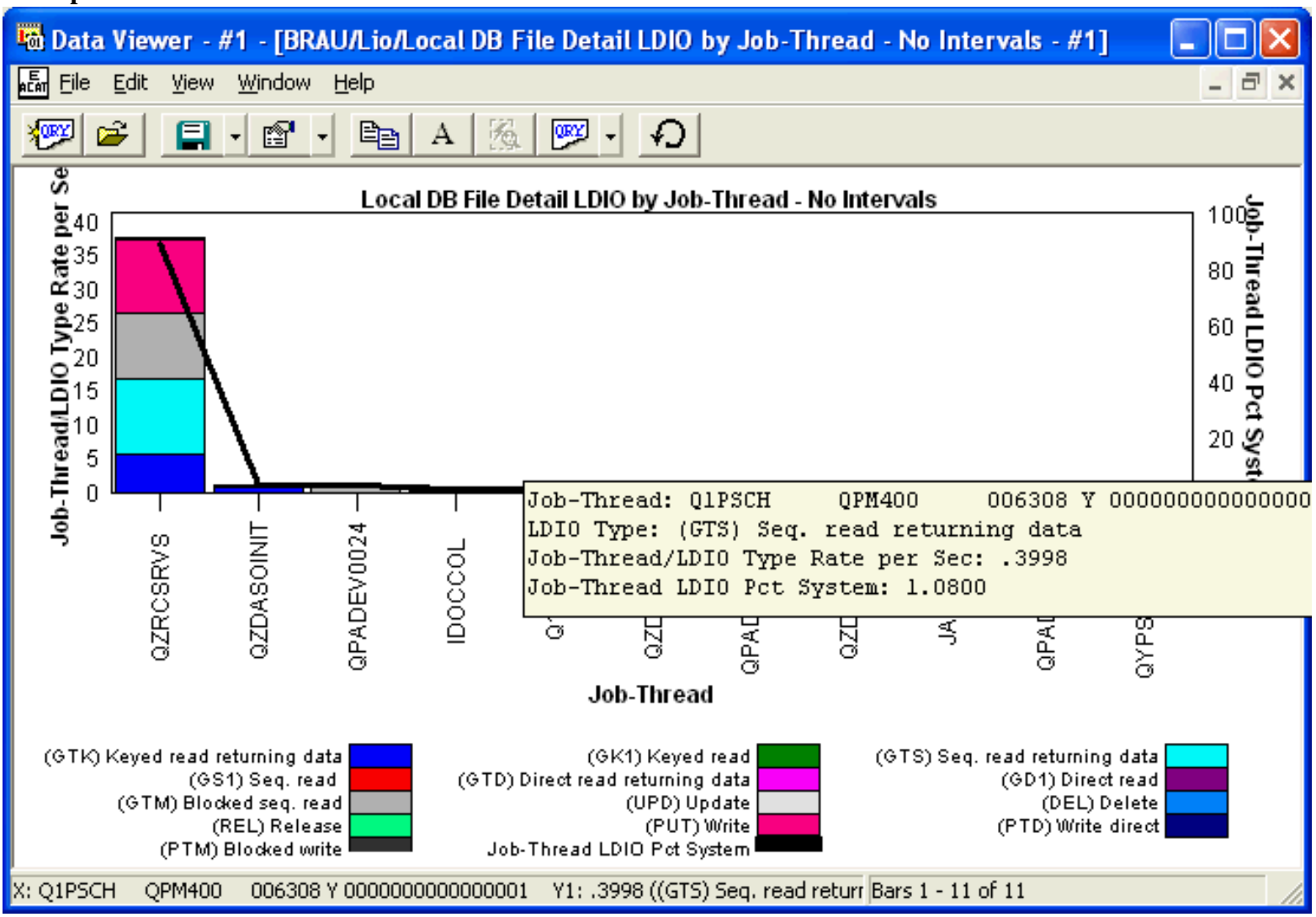


Table of Contents | Previous | Next

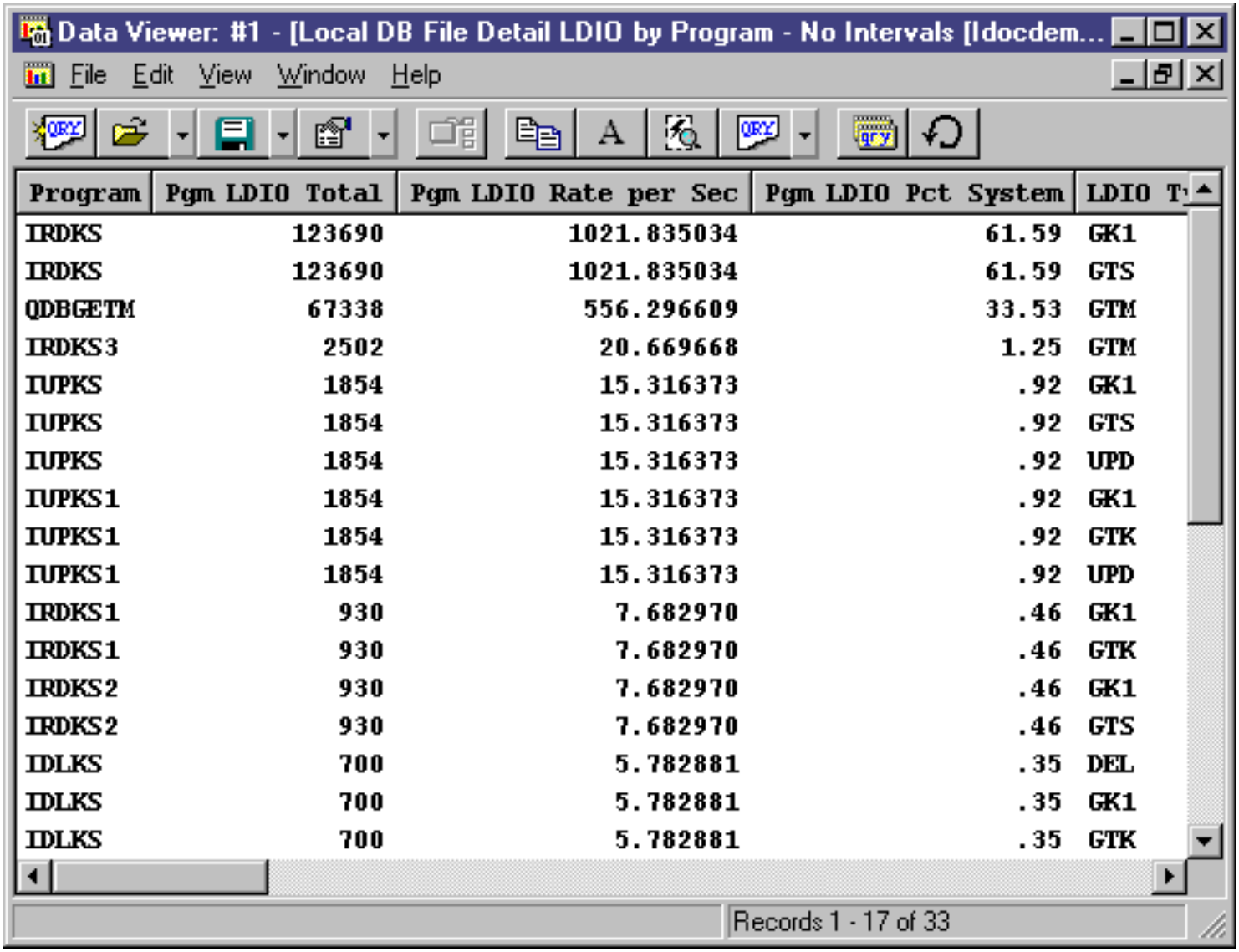
5.1.5 DB File Detail LDIO by Program - No Intervals

Description: This graph shows the programs with the most "logical disk I/O" (LDIO) activity during the collection.

For each of these programs, it also shows the type of LDIO and what percentage of the total collection's LDIO each program represents. Each color in a bar represents a type of LDIO. Clicking one of these colors brings up additional details about the LDIO for the program selected.

Note: In order for program names to be provided, *MIENTRY and *MIEXIT events must be included in the PEX collection.

Example:



Program	Pgm LDIO Total	Pgm LDIO Rate per Sec	Pgm LDIO Pct System	LDIO T
IRDKS	123690	1021.835034	61.59	GK1
IRDKS	123690	1021.835034	61.59	GTS
QDBGETM	67338	556.296609	33.53	GTM
IRDKS3	2502	20.669668	1.25	GTM
IUPKS	1854	15.316373	.92	GK1
IUPKS	1854	15.316373	.92	GTS
IUPKS	1854	15.316373	.92	UPD
IUPKS1	1854	15.316373	.92	GK1
IUPKS1	1854	15.316373	.92	GTK
IUPKS1	1854	15.316373	.92	UPD
IRDKS1	930	7.682970	.46	GK1
IRDKS1	930	7.682970	.46	GTK
IRDKS2	930	7.682970	.46	GK1
IRDKS2	930	7.682970	.46	GTS
IDLKS	700	5.782881	.35	DEL
IDLKS	700	5.782881	.35	GK1
IDLKS	700	5.782881	.35	GTK

Graph Type: stacked vertical bar
X-axis: This axis shows programs in the collection that performed LDIO operations.
Y-Axis: This value is the number of LDIO ops occurring of each type per second. Each color represents a different type of LDIO.
Second Y-Axis: This line shows how much each program's total LDIO contributed to the total system LDIO.

Example:

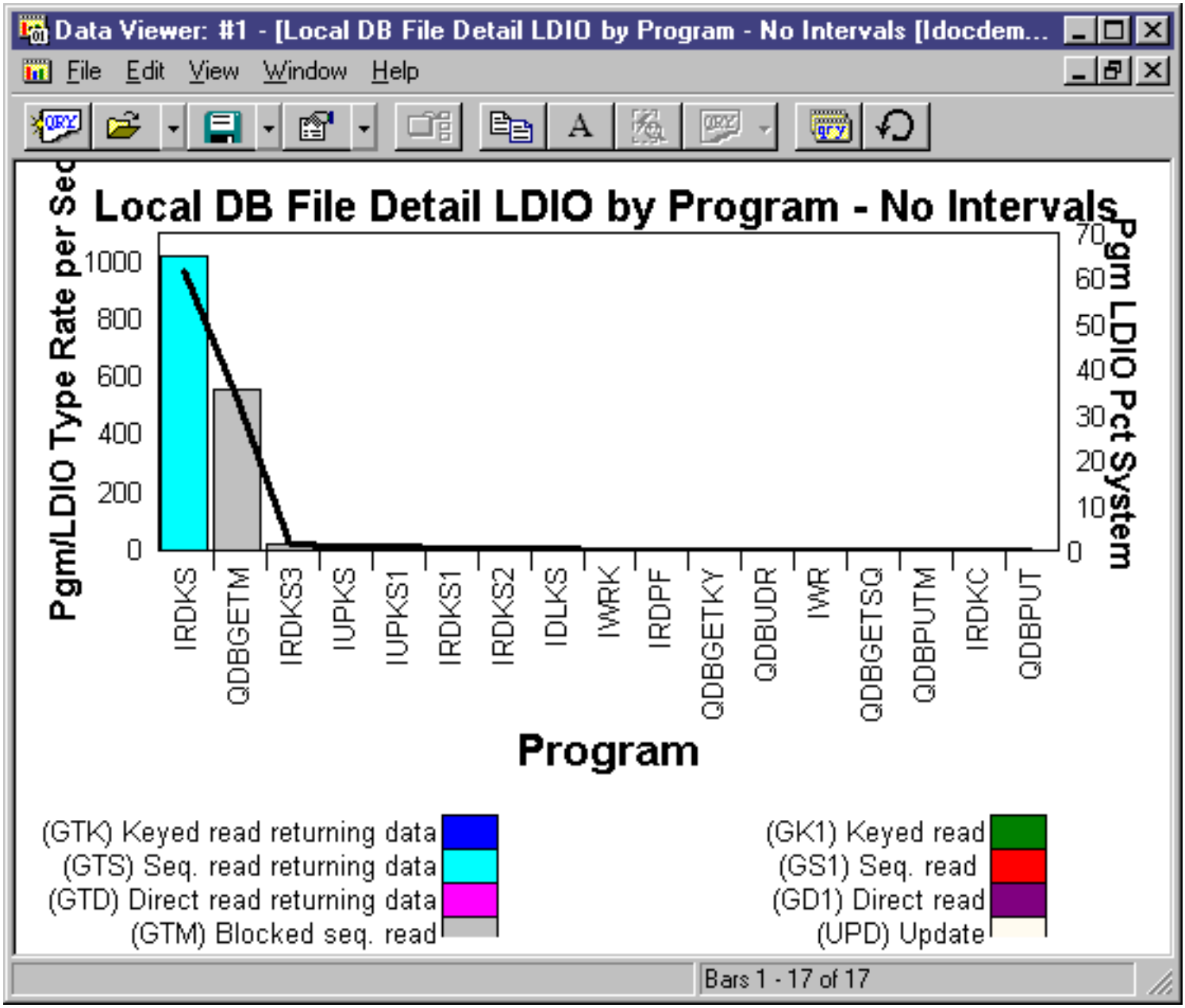
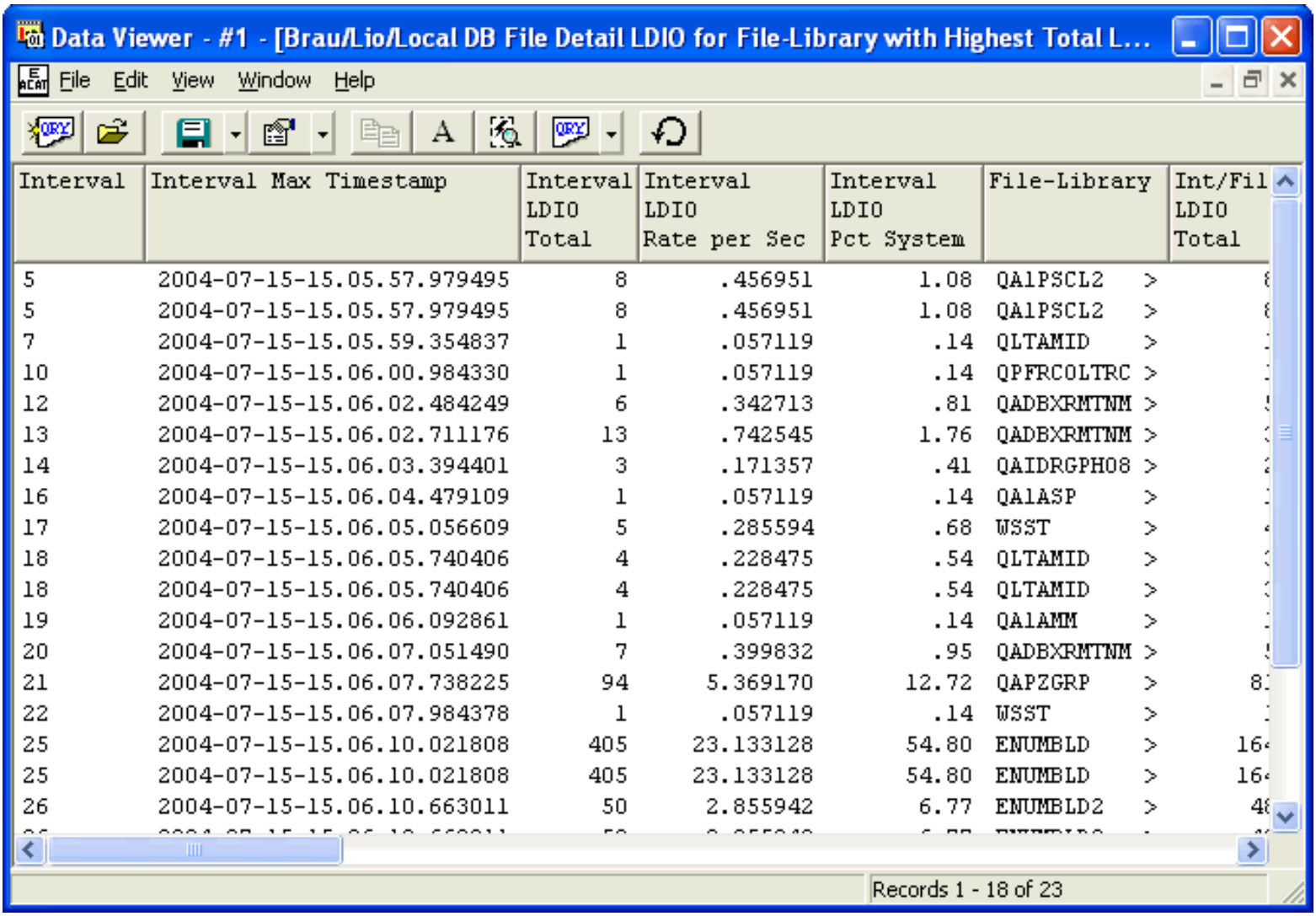

Data Viewer: #1 - [Local DB File Detail LDIO by Program - No Intervals [Idocdem...
File Edit View Window Help
Local DB File Detail LDIO by Program - No Intervals
Pgm/LDIO Type Rate per Sec
Pgm LDIO Pct System
1000
800
600
400
200
0
70
60
50
40
30
20
10
0
IRDKS
QDBGETM
IRDKS3
IUPKS
IUPKS1
IRDKS1
IRDKS2
IDLKS
IWRK
IRDPF
QDBGETKY
QDBUDR
IWR
QDBGETSQ
QDBPUTM
IRDKC
QDBPUT
Program
(GTK) Keyed read returning data
(GTS) Seq. read returning data
(GTD) Direct read returning data
(GTM) Blocked seq. read
(GK1) Keyed read
(GS1) Seq. read
(GD1) Direct read
(UPD) Update
Bars 1 - 17 of 17

Table of Contents | Previous | Next

5.1.6 DB File Detail LDIO for File-Library With Highest Total LDIO per Interval

Description: This report shows the file with the most "logical disk I/O" (LDIO) activity for each time interval in the collection.
It also shows the type(s) of LDIO done to these files and the percentage of the collection's total LDIO. Each file's LDIO activity per interval is also shown.

Each color in a bar represents a type of LDIO. Clicking one of these colors brings up additional details about the LDIO for the file/interval selected.



Interval	Interval Max Timestamp	Interval LDIO Total	Interval LDIO Rate per Sec	Interval LDIO Pct System	File-Library	Int/Fil LDIO Total
5	2004-07-15-15.05.57.979495	8	.456951	1.08	QA1PSCL2 >	8
5	2004-07-15-15.05.57.979495	8	.456951	1.08	QA1PSCL2 >	8
7	2004-07-15-15.05.59.354837	1	.057119	.14	QLTAMID >	1
10	2004-07-15-15.06.00.984330	1	.057119	.14	QPFRCOLTRC >	1
12	2004-07-15-15.06.02.484249	6	.342713	.81	QADBXRMTNM >	5
13	2004-07-15-15.06.02.711176	13	.742545	1.76	QADBXRMTNM >	3
14	2004-07-15-15.06.03.394401	3	.171357	.41	QAIDRGPH08 >	2
16	2004-07-15-15.06.04.479109	1	.057119	.14	QA1ASP >	1
17	2004-07-15-15.06.05.056609	5	.285594	.68	WSST >	4
18	2004-07-15-15.06.05.740406	4	.228475	.54	QLTAMID >	3
18	2004-07-15-15.06.05.740406	4	.228475	.54	QLTAMID >	3
19	2004-07-15-15.06.06.092861	1	.057119	.14	QA1AMM >	1
20	2004-07-15-15.06.07.051490	7	.399832	.95	QADBXRMTNM >	5
21	2004-07-15-15.06.07.738225	94	5.369170	12.72	QAPZGRP >	81
22	2004-07-15-15.06.07.984378	1	.057119	.14	WSST >	1
25	2004-07-15-15.06.10.021808	405	23.133128	54.80	ENUMBLD >	164
25	2004-07-15-15.06.10.021808	405	23.133128	54.80	ENUMBLD >	164
26	2004-07-15-15.06.10.663011	50	2.855942	6.77	ENUMBLD2 >	48

Graph Type: stacked vertical bar
X-axis: Interval number
Y-Axis: This value is the number of LDIO ops occurring of each type per second. Each color represents a different type of LDIO.

Second Y-Axis: This line shows how much each file/library's LDIOs contributed to the total system LDIOs for every interval.

Example:

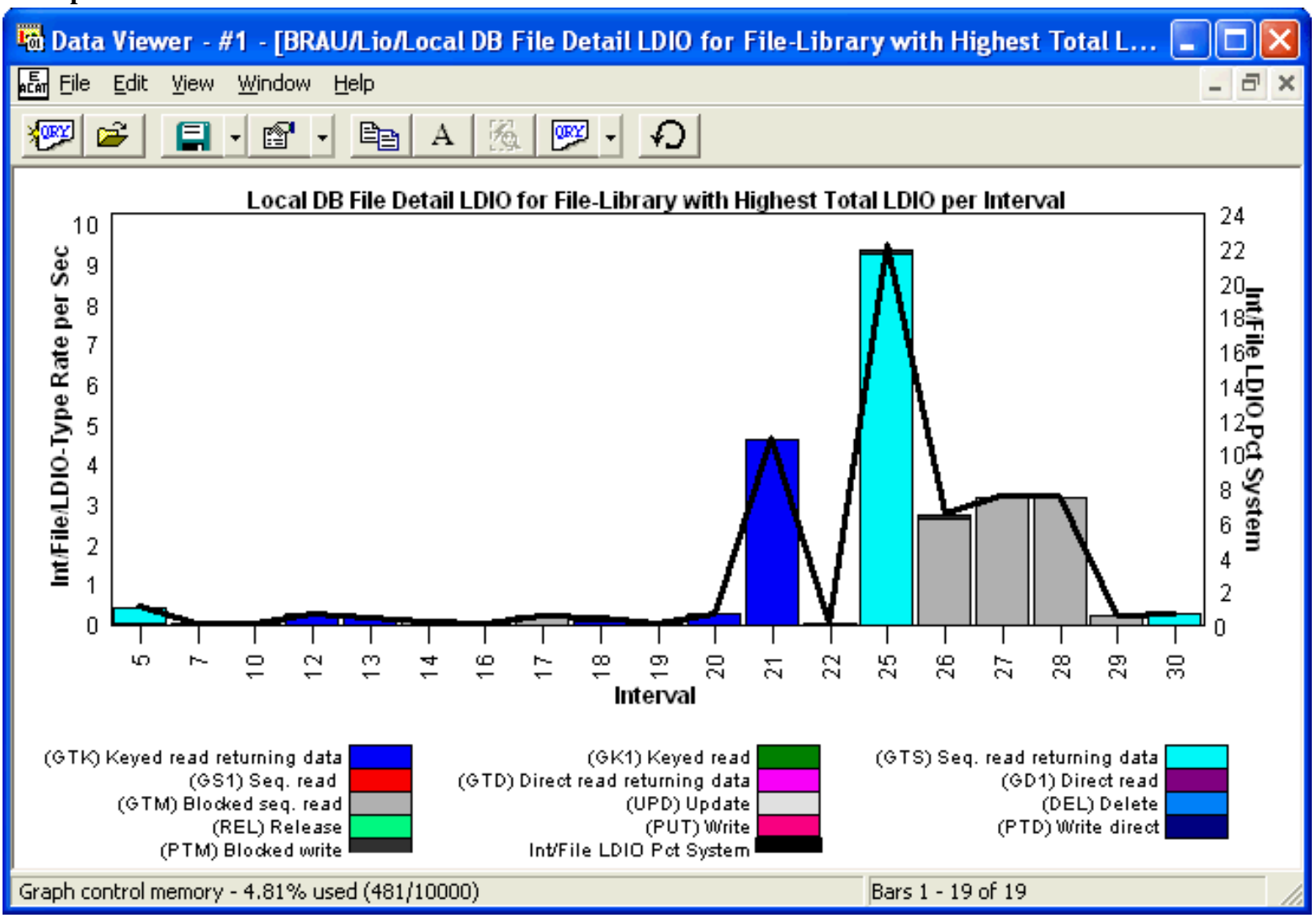


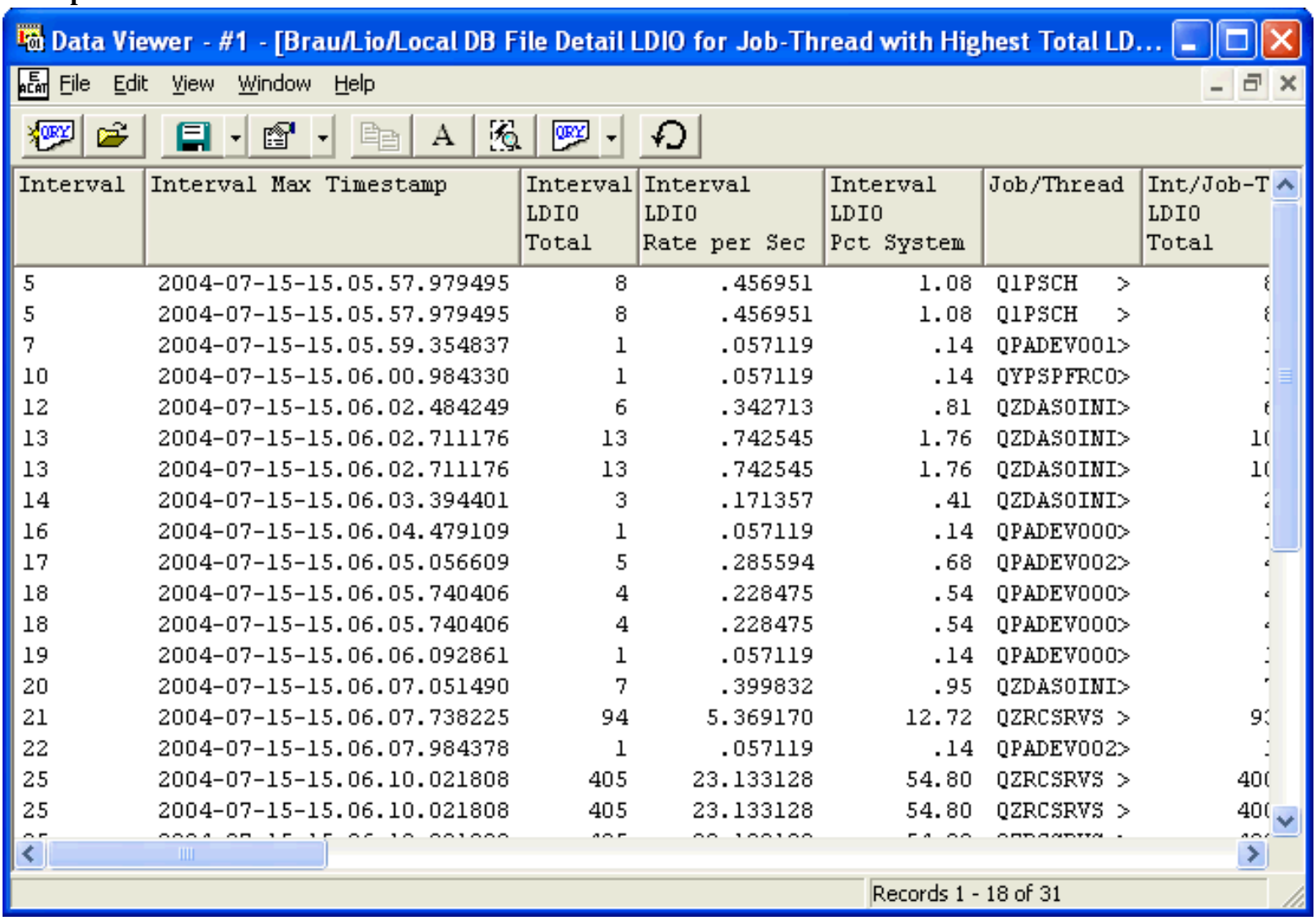
Table of Contents | Previous | Next

5.1.7 DB File Detail LDIO for Job-thread with Highest Total LDIO per Interval

Description: This graph shows the job or thread with the most "logical disk I/O" (LDIO) activity for each time interval in the collection.
It also shows the type(s) of LDIO done by these jobs/threads and the percentage of the collection's total LDIO. Each job/thread's LDIO activity is also shown.

Each color in a bar represents a type of LDIO. Clicking one of these colors brings up additional details about the LDIO for the interval selected.

Example:



Interval	Interval Max Timestamp	Interval LDIO Total	Interval LDIO Rate per Sec	Interval LDIO Pct System	Job/Thread	Int/Job-T LDIO Total
5	2004-07-15-15.05.57.979495	8	.456951	1.08	Q1PSCH >	8
5	2004-07-15-15.05.57.979495	8	.456951	1.08	Q1PSCH >	8
7	2004-07-15-15.05.59.354837	1	.057119	.14	QPADEV001>	1
10	2004-07-15-15.06.00.984330	1	.057119	.14	QYPSPFRCO>	1
12	2004-07-15-15.06.02.484249	6	.342713	.81	QZDASOINI>	6
13	2004-07-15-15.06.02.711176	13	.742545	1.76	QZDASOINI>	1(
13	2004-07-15-15.06.02.711176	13	.742545	1.76	QZDASOINI>	1(
14	2004-07-15-15.06.03.394401	3	.171357	.41	QZDASOINI>	2
16	2004-07-15-15.06.04.479109	1	.057119	.14	QPADEV000>	1
17	2004-07-15-15.06.05.056609	5	.285594	.68	QPADEV002>	4
18	2004-07-15-15.06.05.740406	4	.228475	.54	QPADEV000>	4
18	2004-07-15-15.06.05.740406	4	.228475	.54	QPADEV000>	4
19	2004-07-15-15.06.06.092861	1	.057119	.14	QPADEV000>	1
20	2004-07-15-15.06.07.051490	7	.399832	.95	QZDASOINI>	7
21	2004-07-15-15.06.07.738225	94	5.369170	12.72	QZRCSRVS >	9(
22	2004-07-15-15.06.07.984378	1	.057119	.14	QPADEV002>	1
25	2004-07-15-15.06.10.021808	405	23.133128	54.80	QZRCSRVS >	40(
25	2004-07-15-15.06.10.021808	405	23.133128	54.80	QZRCSRVS >	40(

Graph Type: stacked vertical bar
X-axis: Interval number
Y-Axis: This value is the number of LDIO ops occurring of each type per second for the job/thread having the highest

total LDIO. Each color represents a different type of LDIO.
Second Y-Axis: This line shows how much each job/thread's LDIOs contributed to the total system LDIOs.

Example:

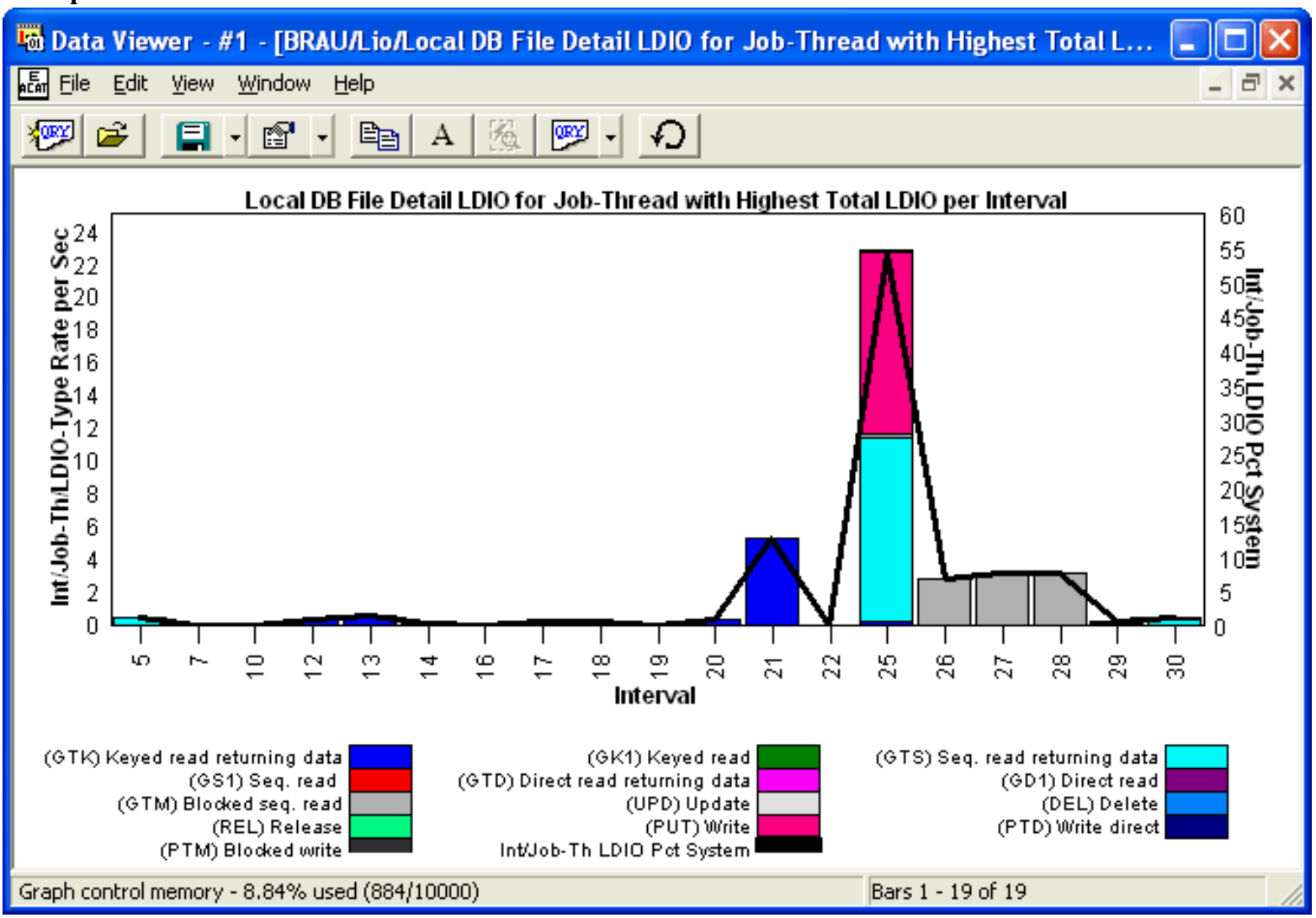


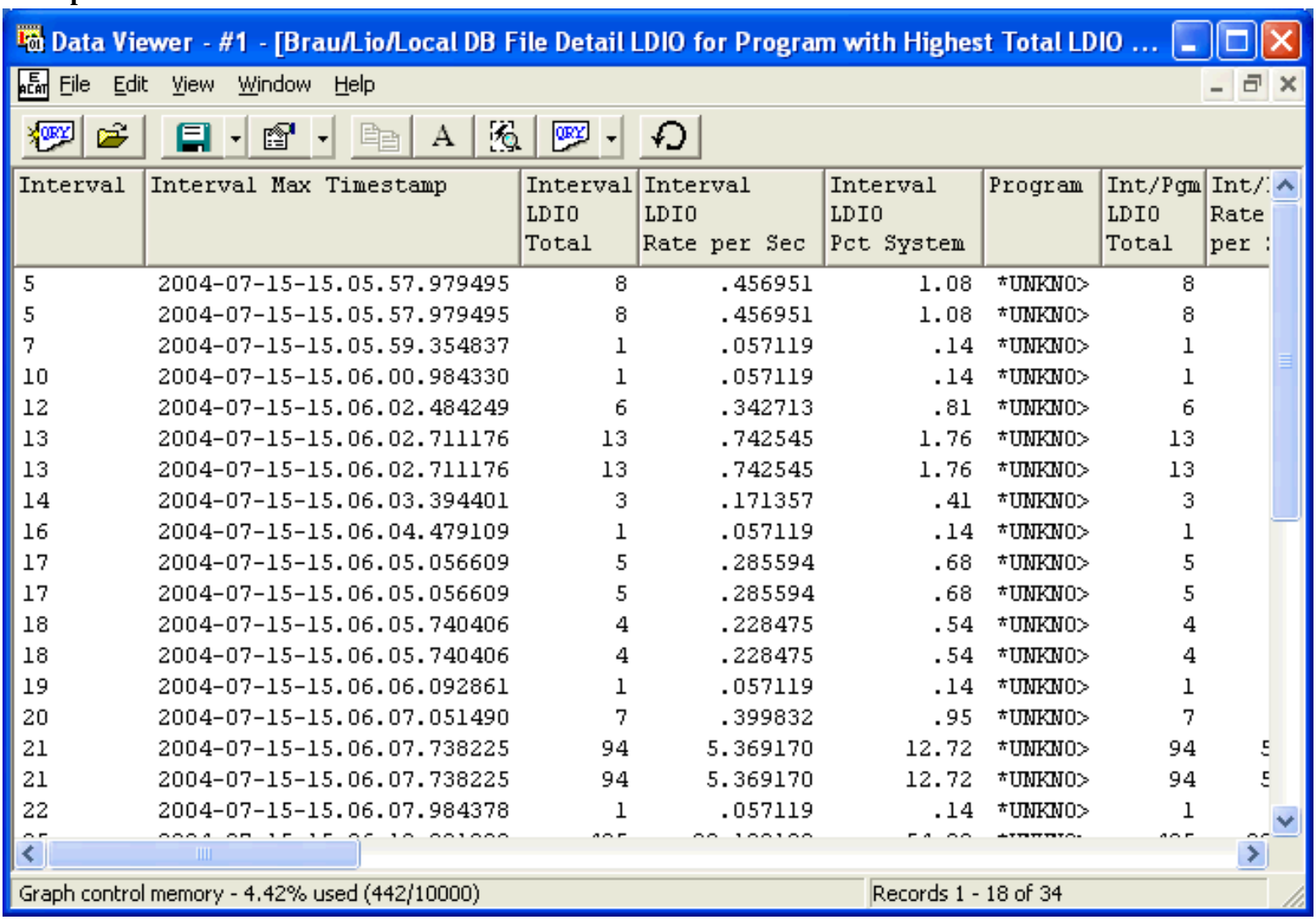
Table of Contents | Previous | Next

5.1.8 DB File Detail LDIO for Program with Highest Total LDIO per Interval

Description: This report shows the program with the most "logical disk I/O" (LDIO) activity for each time interval in the collection. It also shows the type(s) of LDIO done by these programs and the percentage of the collection's total LDIO. Each program's LDIO activity is also shown.

Note: In order for program names to be provided, *MIENTRY and *MIEXIT events must be included in the PEX collection.

Example:



Interval	Interval Max Timestamp	Interval LDIO Total	Interval LDIO Rate per Sec	Interval LDIO Pct System	Program	Int/Pgm LDIO Total	Int/ Rate per
5	2004-07-15-15.05.57.979495	8	.456951	1.08	*UNKNO>	8	
5	2004-07-15-15.05.57.979495	8	.456951	1.08	*UNKNO>	8	
7	2004-07-15-15.05.59.354837	1	.057119	.14	*UNKNO>	1	
10	2004-07-15-15.06.00.984330	1	.057119	.14	*UNKNO>	1	
12	2004-07-15-15.06.02.484249	6	.342713	.81	*UNKNO>	6	
13	2004-07-15-15.06.02.711176	13	.742545	1.76	*UNKNO>	13	
13	2004-07-15-15.06.02.711176	13	.742545	1.76	*UNKNO>	13	
14	2004-07-15-15.06.03.394401	3	.171357	.41	*UNKNO>	3	
16	2004-07-15-15.06.04.479109	1	.057119	.14	*UNKNO>	1	
17	2004-07-15-15.06.05.056609	5	.285594	.68	*UNKNO>	5	
17	2004-07-15-15.06.05.056609	5	.285594	.68	*UNKNO>	5	
18	2004-07-15-15.06.05.740406	4	.228475	.54	*UNKNO>	4	
18	2004-07-15-15.06.05.740406	4	.228475	.54	*UNKNO>	4	
19	2004-07-15-15.06.06.092861	1	.057119	.14	*UNKNO>	1	
20	2004-07-15-15.06.07.051490	7	.399832	.95	*UNKNO>	7	
21	2004-07-15-15.06.07.738225	94	5.369170	12.72	*UNKNO>	94	5
21	2004-07-15-15.06.07.738225	94	5.369170	12.72	*UNKNO>	94	5
22	2004-07-15-15.06.07.984378	1	.057119	.14	*UNKNO>	1	

Graph Type: stacked vertical bar
X-axis: Interval number
Y-Axis: This value is the number of LDIO ops occurring of each type per second. Each color represents a different type of LDIO.

Second Y-Axis: This line shows how much each program's LDIO ops contributed to the total system LDIOs.

Example:

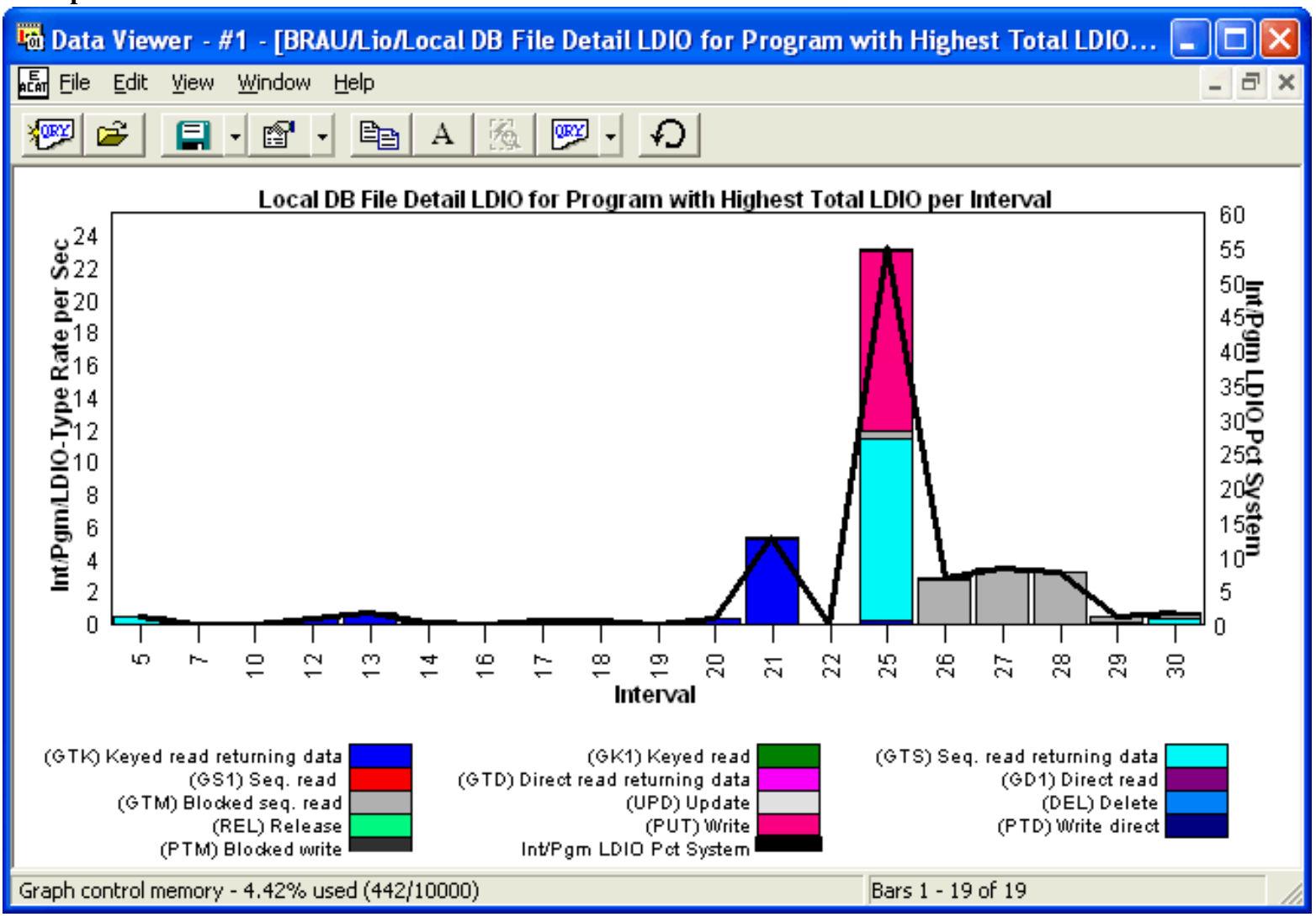
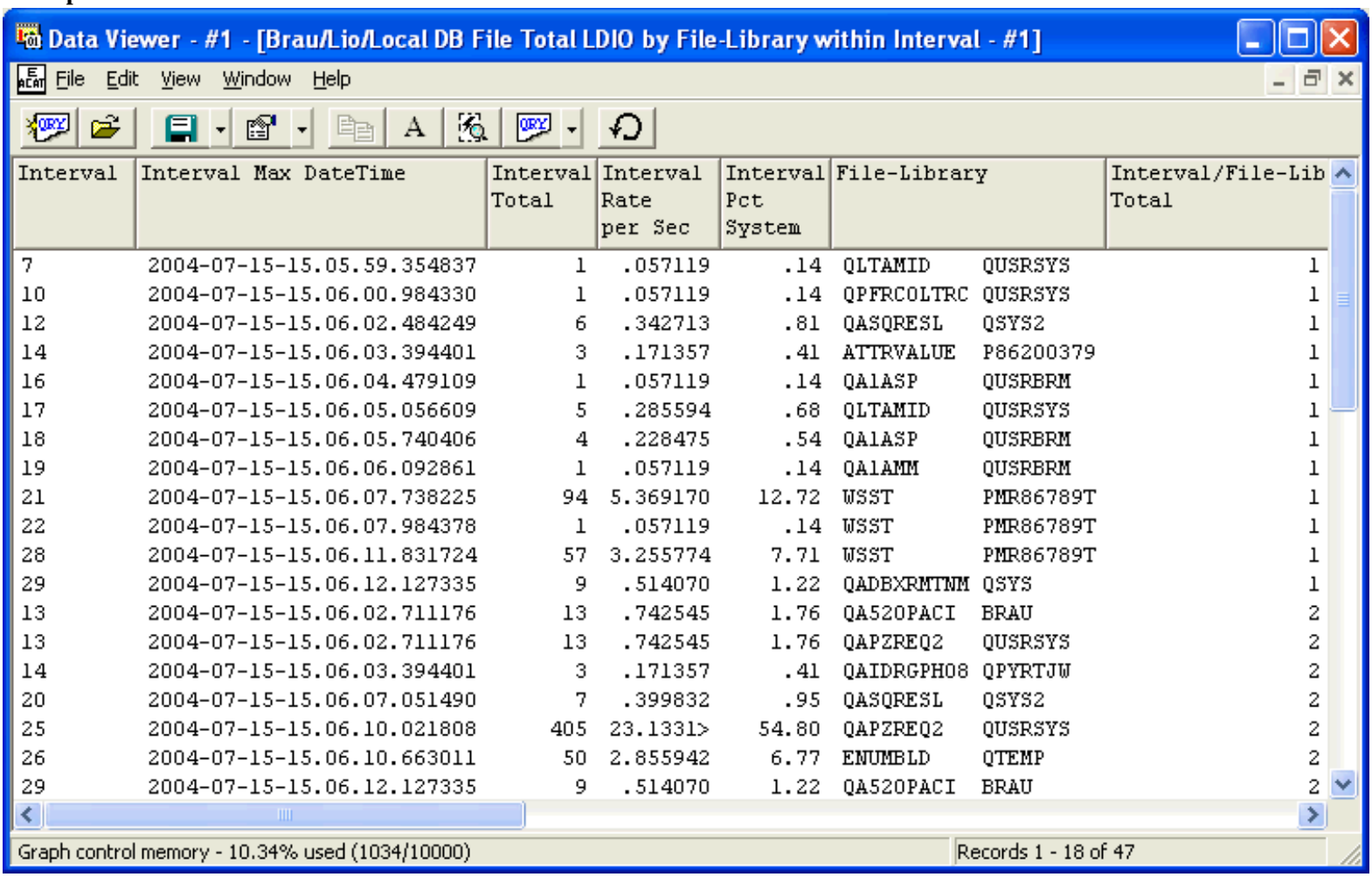


Table of Contents | Previous | Next

5.1.9 DB File Detail LDIO by File-library within Interval

Description: This graph shows the "logical disk I/O" (LDIO) activity for every file by time interval. It shows the percentage of the collection's total LDIO each interval/file's LDIO activity represents.

Example:



Interval	Interval Max DateTime	Interval Total	Interval Rate per Sec	Interval Pct System	File-Library		Interval/File-Lib Total
7	2004-07-15-15.05.59.354837	1	.057119	.14	QLTAMID	QUSRSYS	1
10	2004-07-15-15.06.00.984330	1	.057119	.14	QPFRCOLTRC	QUSRSYS	1
12	2004-07-15-15.06.02.484249	6	.342713	.81	QASQRESL	QSYS2	1
14	2004-07-15-15.06.03.394401	3	.171357	.41	ATTRVALUE	P86200379	1
16	2004-07-15-15.06.04.479109	1	.057119	.14	QA1ASP	QUSRBRM	1
17	2004-07-15-15.06.05.056609	5	.285594	.68	QLTAMID	QUSRSYS	1
18	2004-07-15-15.06.05.740406	4	.228475	.54	QA1ASP	QUSRBRM	1
19	2004-07-15-15.06.06.092861	1	.057119	.14	QA1AMM	QUSRBRM	1
21	2004-07-15-15.06.07.738225	94	5.369170	12.72	WSST	PMR86789T	1
22	2004-07-15-15.06.07.984378	1	.057119	.14	WSST	PMR86789T	1
28	2004-07-15-15.06.11.831724	57	3.255774	7.71	WSST	PMR86789T	1
29	2004-07-15-15.06.12.127335	9	.514070	1.22	QADBXRMTNM	QSYS	1
13	2004-07-15-15.06.02.711176	13	.742545	1.76	QA520PACI	BRAU	2
13	2004-07-15-15.06.02.711176	13	.742545	1.76	QAPZREQ2	QUSRSYS	2
14	2004-07-15-15.06.03.394401	3	.171357	.41	QAIDRGPH08	QPYRTJW	2
20	2004-07-15-15.06.07.051490	7	.399832	.95	QASQRESL	QSYS2	2
25	2004-07-15-15.06.10.021808	405	23.1331>	54.80	QAPZREQ2	QUSRSYS	2
26	2004-07-15-15.06.10.663011	50	2.855942	6.77	ENUMBLD	QTEMP	2
29	2004-07-15-15.06.12.127335	9	.514070	1.22	QA520PACI	BRAU	2

Graph Type: stacked vertical bar
X-axis: Interval number
Y-Axis: This value is the number of LDIO ops occurring per second for a file within an interval. Each color represents a different file.
Second Y-Axis: This line shows how much the collection's LDIOs contributed to the total system LDIOs for each interval.

Example:

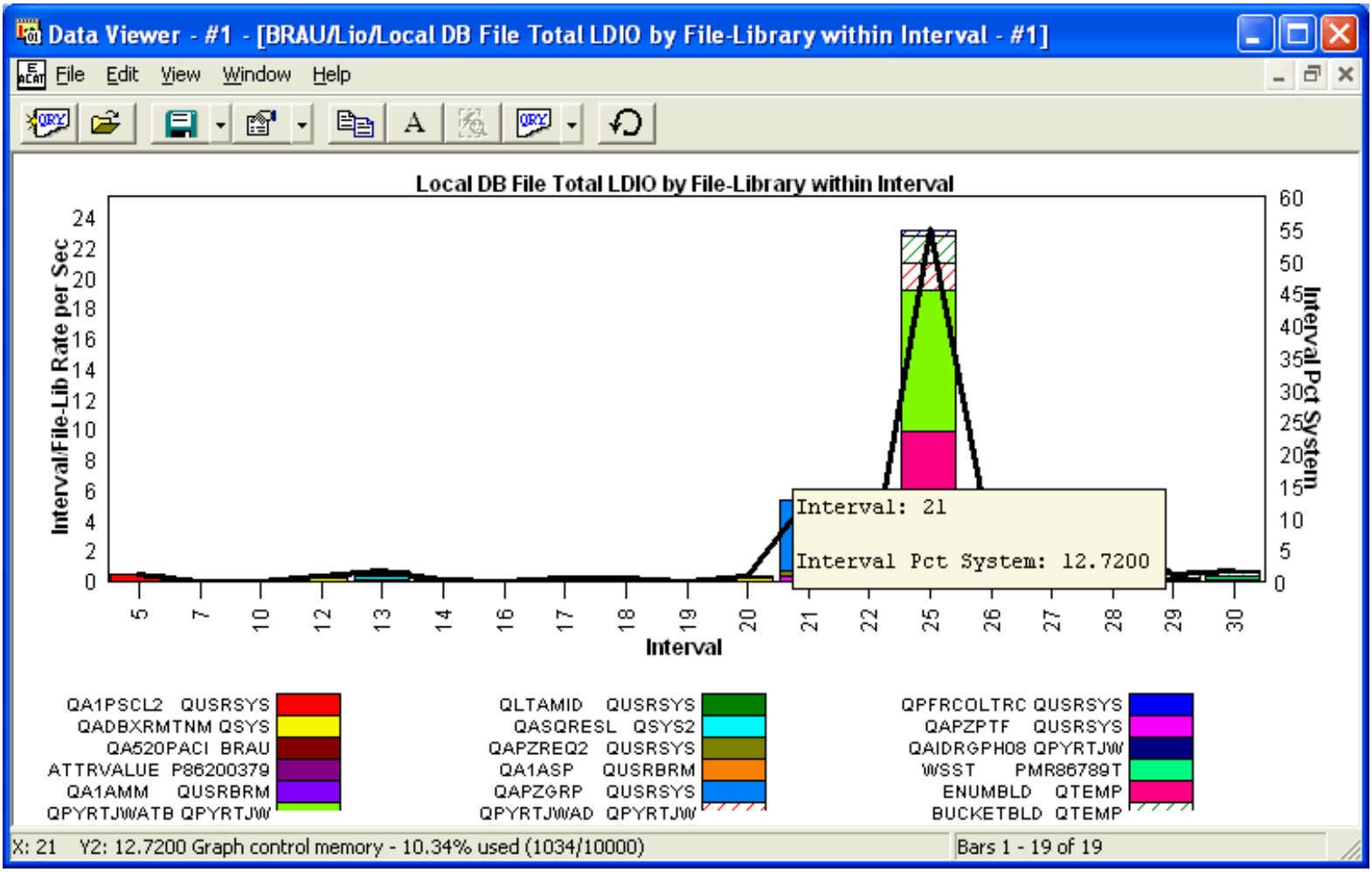

Data Viewer - #1 - [BRAU/Lio/Local DB File Total LDIO by File-Library within Interval - #1]
File Edit View Window Help
Local DB File Total LDIO by File-Library within Interval
Interval/File-Lib Rate per Sec
Interval Pct System
Interval
Interval: 21
Interval Pct System: 12.7200
QA1PSCL2 QUSRSYS
QADBXRMTNM QSYS
QA520PACI BRAU
ATTRVALUE P86200379
QA1AMM QUSRBRM
QPYRTJWATB QPYRTJW
QLTAMID QUSRSYS
QASQRESL QSYS2
QAPZREQ2 QUSRSYS
QA1ASP QUSRBRM
QAPZGRP QUSRSYS
QPYRTJWAD QPYRTJW
QPFRCOLTRC QUSRSYS
QAPZPTF QUSRSYS
QAIDRGPH08 QPYRTJW
WSST PMR86789T
ENUMBLD QTEMP
BUCKETBLD QTEMP
X: 21 Y2: 12.7200 Graph control memory - 10.34% used (1034/10000)
Bars 1 - 19 of 19

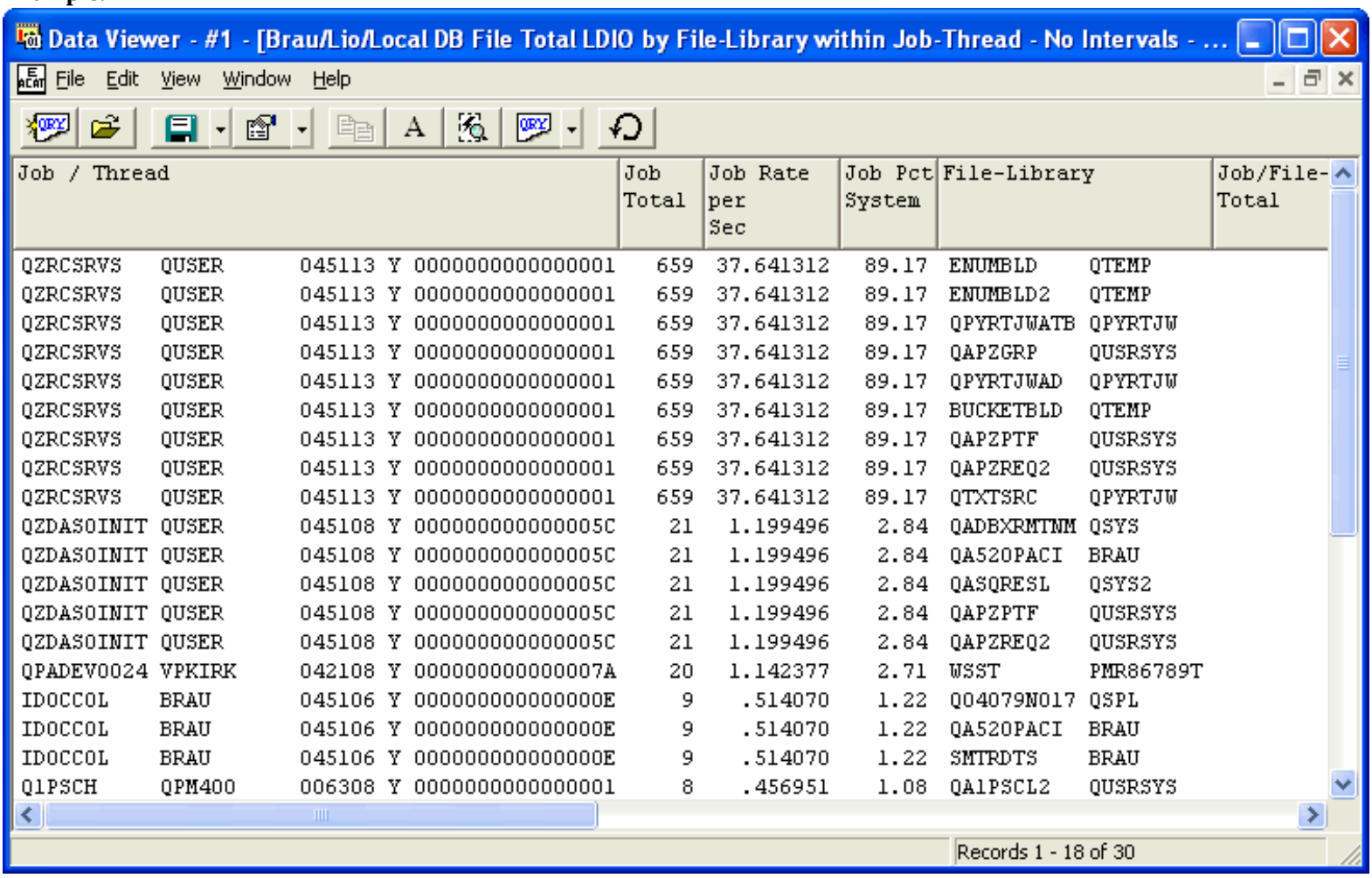
Table of Contents | Previous | Next

5.1.10 DB File Detail LDIO by File-library within Job-thread - No Intervals

Description: This graph shows the job or thread with the most "logical disk I/O" (LDIO) activity for file and time interval in the collection.
It also shows the type(s) of LDIO done by these jobs/threads and the percentage of the collection's total LDIO.

Each color in a bar represents a type of LDIO. Clicking one of these colors brings up additional details about the LDIO for the file and job selected.

Example:



Job / Thread					Job Total	Job Rate per Sec	Job Pct System	File-Library		Job/File-Total
QZRCSRVS	QUSER	045113	Y	0000000000000001	659	37.641312	89.17	ENUMBLD	QTEMP	
QZRCSRVS	QUSER	045113	Y	0000000000000001	659	37.641312	89.17	ENUMBLD2	QTEMP	
QZRCSRVS	QUSER	045113	Y	0000000000000001	659	37.641312	89.17	QPYRTJWATB	QPYRTJW	
QZRCSRVS	QUSER	045113	Y	0000000000000001	659	37.641312	89.17	QAPZGRP	QUSRSYS	
QZRCSRVS	QUSER	045113	Y	0000000000000001	659	37.641312	89.17	QPYRTJWAD	QPYRTJW	
QZRCSRVS	QUSER	045113	Y	0000000000000001	659	37.641312	89.17	BUCKETBLD	QTEMP	
QZRCSRVS	QUSER	045113	Y	0000000000000001	659	37.641312	89.17	QAPZPTF	QUSRSYS	
QZRCSRVS	QUSER	045113	Y	0000000000000001	659	37.641312	89.17	QAPZREQ2	QUSRSYS	
QZRCSRVS	QUSER	045113	Y	0000000000000001	659	37.641312	89.17	QTXTSRC	QPYRTJW	
QZDASOINIT	QUSER	045108	Y	000000000000005C	21	1.199496	2.84	QADBXRMTNM	QSYS	
QZDASOINIT	QUSER	045108	Y	000000000000005C	21	1.199496	2.84	QA520PACI	BRAU	
QZDASOINIT	QUSER	045108	Y	000000000000005C	21	1.199496	2.84	QASQRESL	QSYS2	
QZDASOINIT	QUSER	045108	Y	000000000000005C	21	1.199496	2.84	QAPZPTF	QUSRSYS	
QZDASOINIT	QUSER	045108	Y	000000000000005C	21	1.199496	2.84	QAPZREQ2	QUSRSYS	
QPADEV0024	VPKIRK	042108	Y	000000000000007A	20	1.142377	2.71	WSST	PMR86789T	
IDOCCOL	BRAU	045106	Y	000000000000000E	9	.514070	1.22	Q04079N017	QSPL	
IDOCCOL	BRAU	045106	Y	000000000000000E	9	.514070	1.22	QA520PACI	BRAU	
IDOCCOL	BRAU	045106	Y	000000000000000E	9	.514070	1.22	SMTRDTS	BRAU	
Q1PSCH	QPM400	006308	Y	0000000000000001	8	.456951	1.08	QA1PSCL2	QUSRSYS	

Graph Type: stacked vertical bar
X-axis: Job/thread contributing LDIOs within the collection. The entire field is shown by placing the mouse over a bar.
Y-Axis: This value is the number of LDIO ops occurring of each type per second for a specific file within a job. Each color represents a different file.
Second Y-Axis: This line shows how much each jobs's total LDIO ops contributed to the total system LDIOs.

Example:

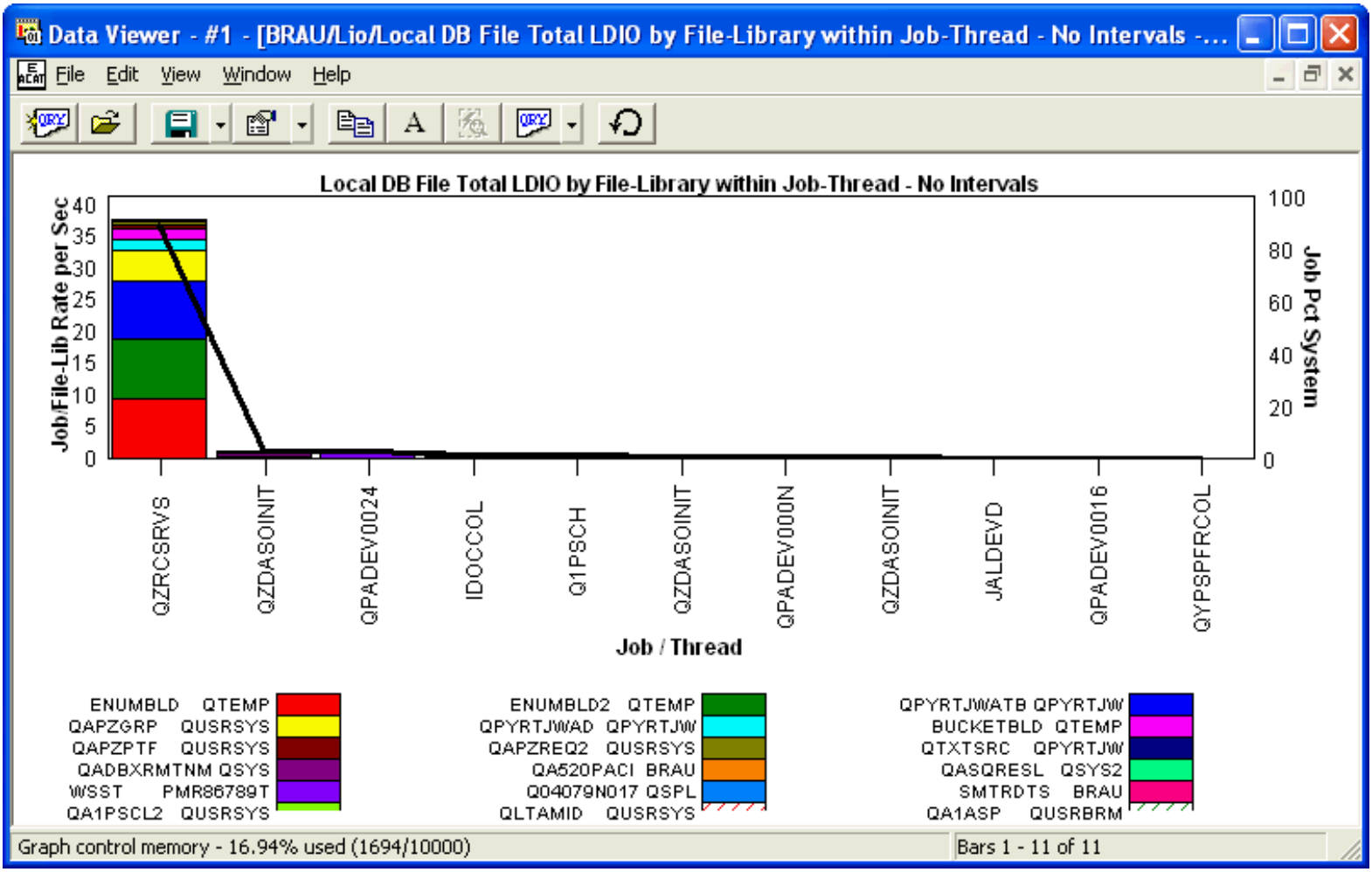

Data Viewer - #1 - [BRAU/Lio/Local DB File Total LDIO by File-Library within Job-Thread - No Intervals -...
File Edit View Window Help
Local DB File Total LDIO by File-Library within Job-Thread - No Intervals
Job/File-Lib Rate per Sec
Job Pct System
0 5 10 15 20 25 30 35 40
0 20 40 60 80 100
QZRCSRVS
QZDASOINIT
QPADEV0024
IDOCCOL
Q1PSCH
QZDASOINIT
QPADEV000N
QZDASOINIT
JALDEVD
QPADEV0016
QYPSPFRCOL
Job / Thread
ENUMBLD QTEMP
QAPZGRP QUSRSYS
QAPZPTF QUSRSYS
QADBXRMTNM QSYS
WSST PMR86789T
QA1PSCL2 QUSRSYS
ENUMBLD2 QTEMP
QPYRTJWAD QPYRTJW
QAPZREQ2 QUSRSYS
QA520PACI BRAU
Q04079N017 QSPL
QLTAMID QUSRSYS
QPYRTJWATB QPYRTJW
BUCKETBLD QTEMP
QTXTSRC QPYRTJW
QASQRESL QSYS2
SMTRDTS BRAU
QA1ASP QUSRBRM
Graph control memory - 16.94% used (1694/10000)
Bars 1 - 11 of 11

Table of Contents | Previous | Next

5.1.11 DB File Detail LDIO by File-library within Program - No Intervals

Description: This report shows the program LDIO activity for each file within the collection.

Note: In order for program names to be provided, *MIENTRY and *MIEXIT events must be included in the PEX collection.

Example:

Data Viewer - #1 - [Brau/Lio/Local DB File Total LDIO by File-Library within Program - No Intervals - #1]

File Edit View Window Help

Program	Pgm Total	Pgm Rate per Sec	Pgm Pct System	File-Library		Pgm/File-Lib Total	Pgm/File-Lib Rate per Sec	Pgm/File-Lib Pct System	Pgm/File-Lib LDIO CPU Microseconds
*UNKNOWN	739	42.210819	100	ENUMBLD	QTEMP	166	9.481727	22.46	
*UNKNOWN	739	42.210819	100	ENUMBLD2	QTEMP	164	9.367489	22.19	
*UNKNOWN	739	42.210819	100	QPYRTJWATB	QPYRTJW	162	9.253251	21.92	
*UNKNOWN	739	42.210819	100	QAPZGRP	QUSRSYS	81	4.626626	10.96	
*UNKNOWN	739	42.210819	100	QPYRTJWAD	QPYRTJW	32	1.827803	4.33	
*UNKNOWN	739	42.210819	100	BUCKETBLD	QTEMP	32	1.827803	4.33	
*UNKNOWN	739	42.210819	100	WSST	PMR86789T	20	1.142377	2.71	
*UNKNOWN	739	42.210819	100	QADBXRMTNM	QSYS	14	.799664	1.89	
*UNKNOWN	739	42.210819	100	QAPZPTF	QUSRSYS	12	.685426	1.62	
*UNKNOWN	739	42.210819	100	QAPZREQ2	QUSRSYS	10	.571188	1.35	
*UNKNOWN	739	42.210819	100	**OTHER***		46	2.627466	6.22	

Records 1 - 11 of 11

Graph Type: stacked vertical bar
X-axis: Program name. The entire field is shown by placing the mouse over a bar.
Y-Axis: This value is the number of LDIO ops occurring per second for a file by program. Each color represents a different file
Second Y-Axis: This line shows how much each program's LDIO ops contributed to the total system LDIOs.

Example:

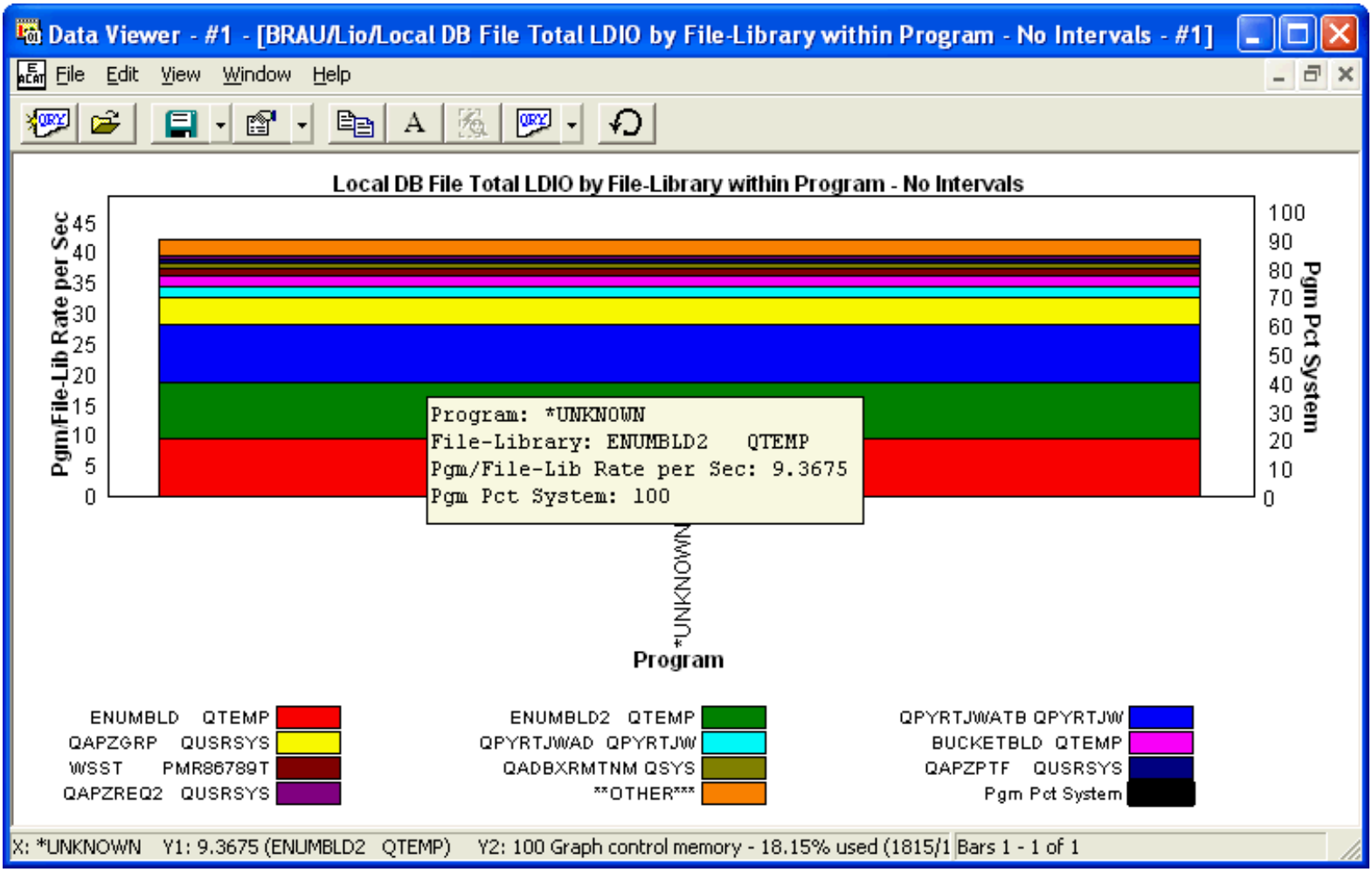
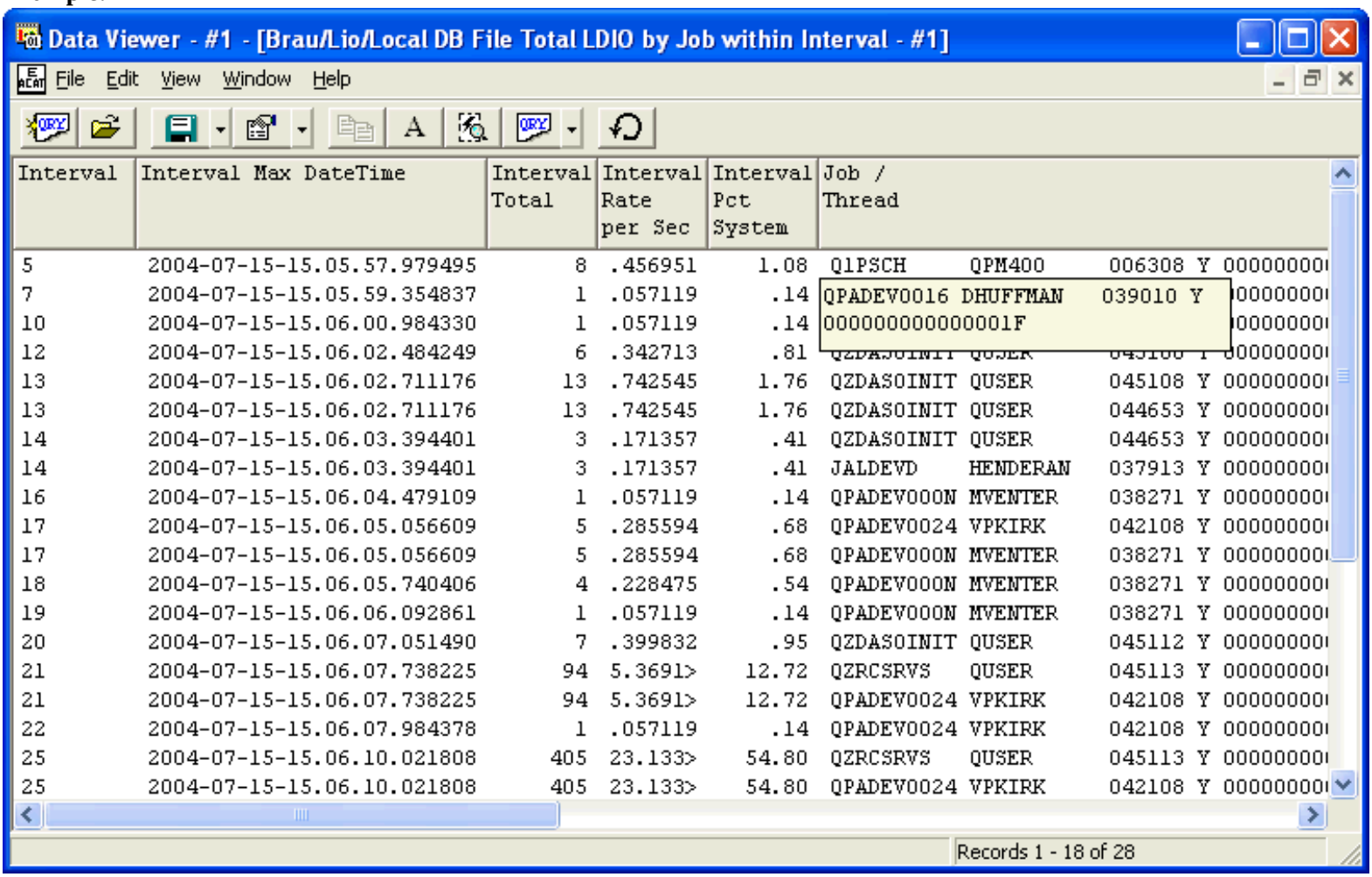

Data Viewer - #1 - [BRAU/Lio/Local DB File Total LDIO by File-Library within Program - No Intervals - #1]
File Edit View Window Help
Local DB File Total LDIO by File-Library within Program - No Intervals
Pgm/File-Lib Rate per Sec
45 40 35 30 25 20 15 10 5 0
Pgm Pct System
100 90 80 70 60 50 40 30 20 10 0
Program: *UNKNOWN
File-Library: ENUMBLD2 QTEMP
Pgm/File-Lib Rate per Sec: 9.3675
Pgm Pct System: 100
*UNKNOWN
Program
ENUMBLD QTEMP
QAPZGRP QUSRSYS
WSST PMR86789T
QAPZREQ2 QUSRSYS
ENUMBLD2 QTEMP
QPYRTJWAD QPYRTJW
QADBXRMTNM QSYS
OTHER*
QPYRTJWATB QPYRTJW
BUCKETBLD QTEMP
QAPZPTF QUSRSYS
Pgm Pct System
X: *UNKNOWN Y1: 9.3675 (ENUMBLD2 QTEMP) Y2: 100 Graph control memory - 18.15% used (1815/1
Bars 1 - 1 of 1

Table of Contents | Previous | Next

5.1.12 DB File Detail LDIO by Job within Interval

Description: This report shows the LDIO activity for each job within the collection each interval.
Each color in the graph repesents a unique job. The graph can be used to see which jobs were the most dominant as far as the LDIO operation rates throughout the collection.

Example:



Interval	Interval Max DateTime	Interval Total	Interval Rate per Sec	Interval Pct System	Job / Thread
5	2004-07-15-15.05.57.979495	8	.456951	1.08	Q1PSCH QPM400 006308 Y 00000000
7	2004-07-15-15.05.59.354837	1	.057119	.14	QPADEV0016 DHUFFMAN 039010 Y 00000000
10	2004-07-15-15.06.00.984330	1	.057119	.14	000000000000001F 00000000
12	2004-07-15-15.06.02.484249	6	.342713	.81	[illegible] 00000000
13	2004-07-15-15.06.02.711176	13	.742545	1.76	QZDASOINIT QUSER 045108 Y 00000000
13	2004-07-15-15.06.02.711176	13	.742545	1.76	QZDASOINIT QUSER 044653 Y 00000000
14	2004-07-15-15.06.03.394401	3	.171357	.41	QZDASOINIT QUSER 044653 Y 00000000
14	2004-07-15-15.06.03.394401	3	.171357	.41	JALDEVD HENDERAN 037913 Y 00000000
16	2004-07-15-15.06.04.479109	1	.057119	.14	QPADEV000N MVENTER 038271 Y 00000000
17	2004-07-15-15.06.05.056609	5	.285594	.68	QPADEV0024 VPKIRK 042108 Y 00000000
17	2004-07-15-15.06.05.056609	5	.285594	.68	QPADEV000N MVENTER 038271 Y 00000000
18	2004-07-15-15.06.05.740406	4	.228475	.54	QPADEV000N MVENTER 038271 Y 00000000
19	2004-07-15-15.06.06.092861	1	.057119	.14	QPADEV000N MVENTER 038271 Y 00000000
20	2004-07-15-15.06.07.051490	7	.399832	.95	QZDASOINIT QUSER 045112 Y 00000000
21	2004-07-15-15.06.07.738225	94	5.3691>	12.72	QZRCSRVS QUSER 045113 Y 00000000
21	2004-07-15-15.06.07.738225	94	5.3691>	12.72	QPADEV0024 VPKIRK 042108 Y 00000000
22	2004-07-15-15.06.07.984378	1	.057119	.14	QPADEV0024 VPKIRK 042108 Y 00000000
25	2004-07-15-15.06.10.021808	405	23.133>	54.80	QZRCSRVS QUSER 045113 Y 00000000
25	2004-07-15-15.06.10.021808	405	23.133>	54.80	QPADEV0024 VPKIRK 042108 Y 00000000

Graph Type: stacked vertical bar
X-axis: Interval number
Y-Axis: This value is the number of LDIO ops occurring per second per job. Each color represents a different job.
Second Y-Axis: This line shows how interval's LDIO ops contributed to the total system LDIOs.

Example:

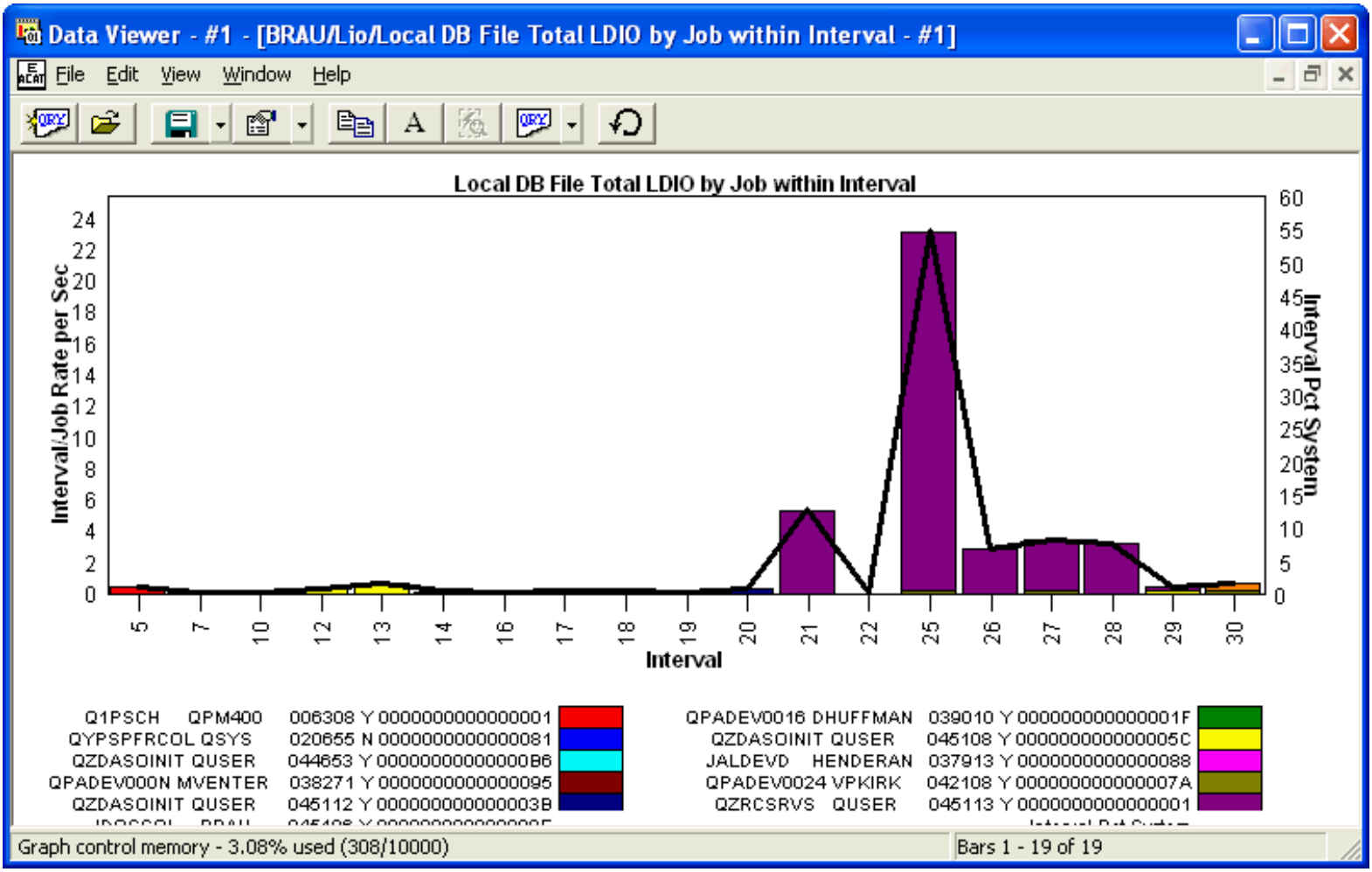
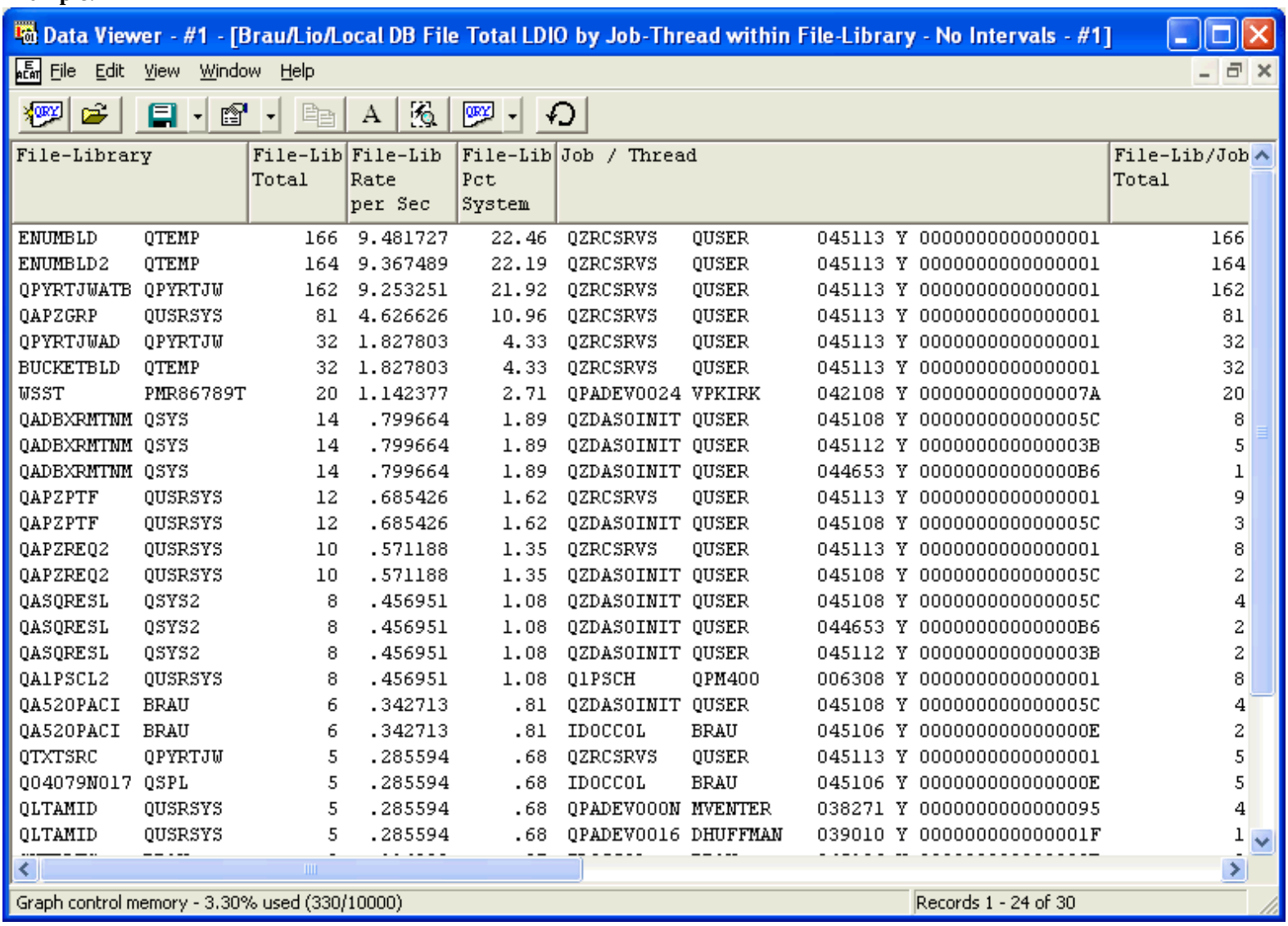

Data Viewer - #1 - [BRAU/Lio/Local DB File Total LDIO by Job within Interval - #1]
File Edit View Window Help
Local DB File Total LDIO by Job within Interval
Interval/Job Rate per Sec
Interval Pct System
Interval
5 7 10 12 13 14 16 17 18 19 20 21 22 25 26 27 28 29 30
Q1PSCH QPM400 006308 Y 0000000000000001
QYPSPFRCOL QSYS 020655 N 0000000000000081
QZDASOINIT QUSER 044653 Y 00000000000000B6
QPADEV000N MVENTER 038271 Y 0000000000000095
QZDASOINIT QUSER 045112 Y 000000000000003B
QPADEV0016 DHUFFMAN 039010 Y 000000000000001F
QZDASOINIT QUSER 045108 Y 000000000000005C
JALDEVD HENDERAN 037913 Y 0000000000000088
QPADEV0024 VPKIRK 042108 Y 000000000000007A
QZRCSRVS QUSER 045113 Y 0000000000000001
Graph control memory - 3.08% used (308/10000)
Bars 1 - 19 of 19

Table of Contents | Previous | Next

5.1.13 DB File Detail LDIO by Job-thread within File-library - No intervals

Description: This report shows the LDIO activity for each job by file within the collection.
Each color in the graph repesents a unique job. The X axis in the graph displays file names with the files/jobs receiving the highest rate of LDIO operations first.

Example:



File-Library		File-Lib Total	File-Lib Rate per Sec	File-Lib Pct System	Job / Thread				File-Lib/Job Total
ENUMBLD	QTEMP	166	9.481727	22.46	QZRCSRVS	QUSER	045113	Y 0000000000000001	166
ENUMBLD2	QTEMP	164	9.367489	22.19	QZRCSRVS	QUSER	045113	Y 0000000000000001	164
QPYRTJWATB	QPYRTJW	162	9.253251	21.92	QZRCSRVS	QUSER	045113	Y 0000000000000001	162
QAPZGRP	QUSRSYS	81	4.626626	10.96	QZRCSRVS	QUSER	045113	Y 0000000000000001	81
QPYRTJWAD	QPYRTJW	32	1.827803	4.33	QZRCSRVS	QUSER	045113	Y 0000000000000001	32
BUCKETBLD	QTEMP	32	1.827803	4.33	QZRCSRVS	QUSER	045113	Y 0000000000000001	32
WSST	PMR86789T	20	1.142377	2.71	QPADEV0024	VPKIRK	042108	Y 000000000000007A	20
QADBXRMTNM	QSYS	14	.799664	1.89	QZDASOINIT	QUSER	045108	Y 000000000000005C	8
QADBXRMTNM	QSYS	14	.799664	1.89	QZDASOINIT	QUSER	045112	Y 000000000000003B	5
QADBXRMTNM	QSYS	14	.799664	1.89	QZDASOINIT	QUSER	044653	Y 00000000000000B6	1
QAPZPTF	QUSRSYS	12	.685426	1.62	QZRCSRVS	QUSER	045113	Y 0000000000000001	9
QAPZPTF	QUSRSYS	12	.685426	1.62	QZDASOINIT	QUSER	045108	Y 000000000000005C	3
QAPZREQ2	QUSRSYS	10	.571188	1.35	QZRCSRVS	QUSER	045113	Y 0000000000000001	8
QAPZREQ2	QUSRSYS	10	.571188	1.35	QZDASOINIT	QUSER	045108	Y 000000000000005C	2
QASQRESL	QSYS2	8	.456951	1.08	QZDASOINIT	QUSER	045108	Y 000000000000005C	4
QASQRESL	QSYS2	8	.456951	1.08	QZDASOINIT	QUSER	044653	Y 00000000000000B6	2
QASQRESL	QSYS2	8	.456951	1.08	QZDASOINIT	QUSER	045112	Y 000000000000003B	2
QA1PSCL2	QUSRSYS	8	.456951	1.08	Q1PSCH	QPM400	006308	Y 0000000000000001	8
QA520PACI	BRAU	6	.342713	.81	QZDASOINIT	QUSER	045108	Y 000000000000005C	4
QA520PACI	BRAU	6	.342713	.81	IDOCCOL	BRAU	045106	Y 000000000000000E	2
QTXTSRC	QPYRTJW	5	.285594	.68	QZRCSRVS	QUSER	045113	Y 0000000000000001	5
Q04079N017	QSPL	5	.285594	.68	IDOCCOL	BRAU	045106	Y 000000000000000E	5
QLTAMID	QUSRSYS	5	.285594	.68	QPADEV000N	MVENTER	038271	Y 0000000000000095	4
QLTAMID	QUSRSYS	5	.285594	.68	QPADEV0016	DHUFFMAN	039010	Y 000000000000001F	1

Graph Type: stacked vertical bar
X-axis: File and library name. The entire field is shown by placing the mouse over a bar.
Y-Axis: This value is the number of LDIO ops occurring per second per file within job. Each color represents a different job.
Second Y-Axis: This line shows how much each file/library's LDIO ops contributed to the total system LDIOs.

Example:

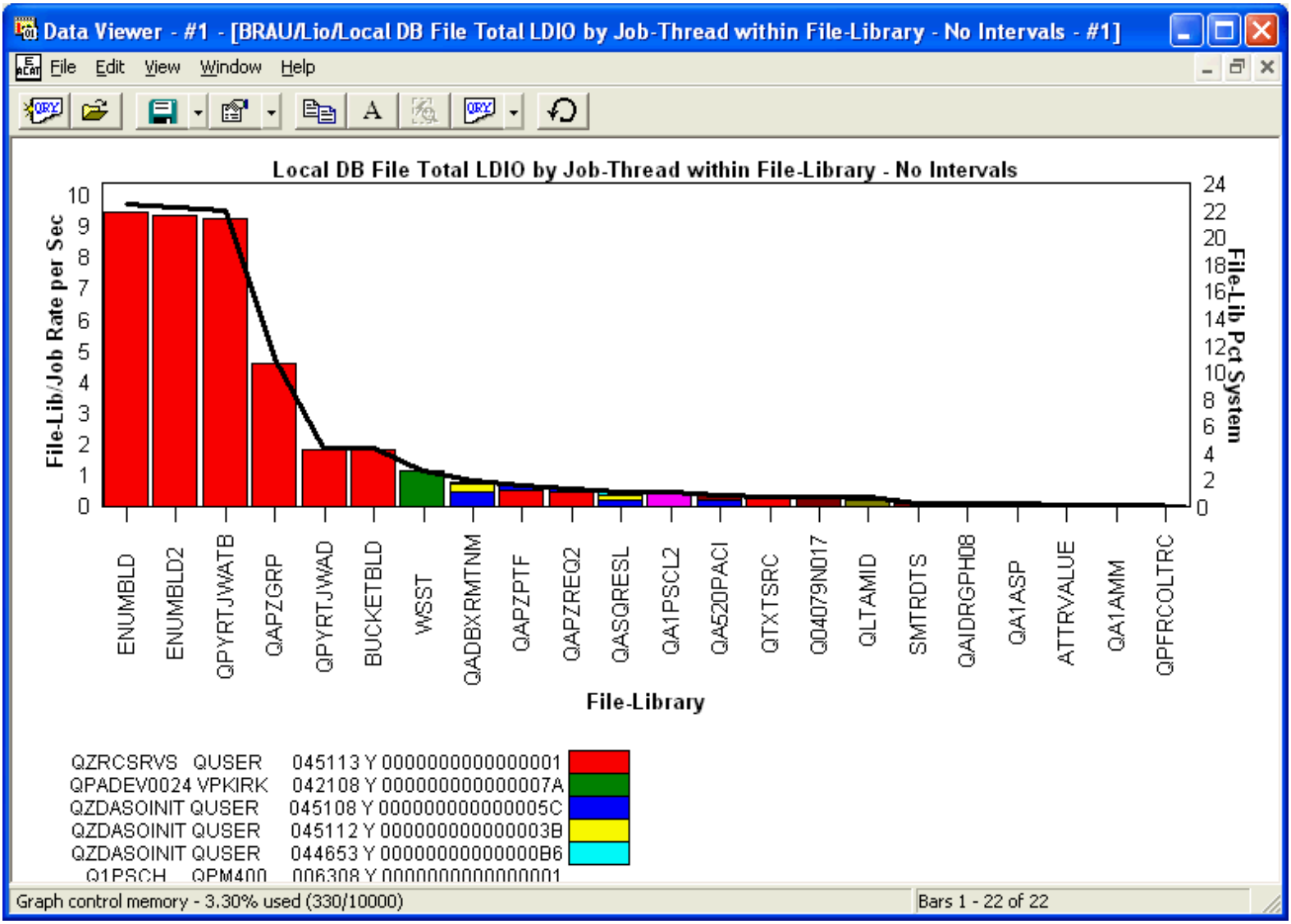

Data Viewer - #1 - [BRAU/Lio/Local DB File Total LDIO by Job-Thread within File-Library - No Intervals - #1]
File Edit View Window Help
Local DB File Total LDIO by Job-Thread within File-Library - No Intervals
File-Lib/Job Rate per Sec
FileLib Pct System
ENUMBLD
ENUMBLD2
QPYRTJWATB
QAPZGRP
QPYRTJWAD
BUCKETBLD
WSST
QADBXRMTNM
QAPZPTF
QAPZREQ2
QASQRESL
QA1PSCL2
QA520PACI
QTXTSRC
Q04079N017
QLTAMID
SMTRDTS
QAIDRGPH08
QA1ASP
ATTRVALUE
QA1AMM
QPFRCOLTRC
File-Library
QZRCSRVS QUSER 045113 Y 0000000000000001
QPADEV0024 VPKIRK 042108 Y 000000000000007A
QZDASOINIT QUSER 045108 Y 000000000000005C
QZDASOINIT QUSER 045112 Y 000000000000003B
QZDASOINIT QUSER 044653 Y 00000000000000B6
Graph control memory - 3.30% used (330/10000)
Bars 1 - 22 of 22

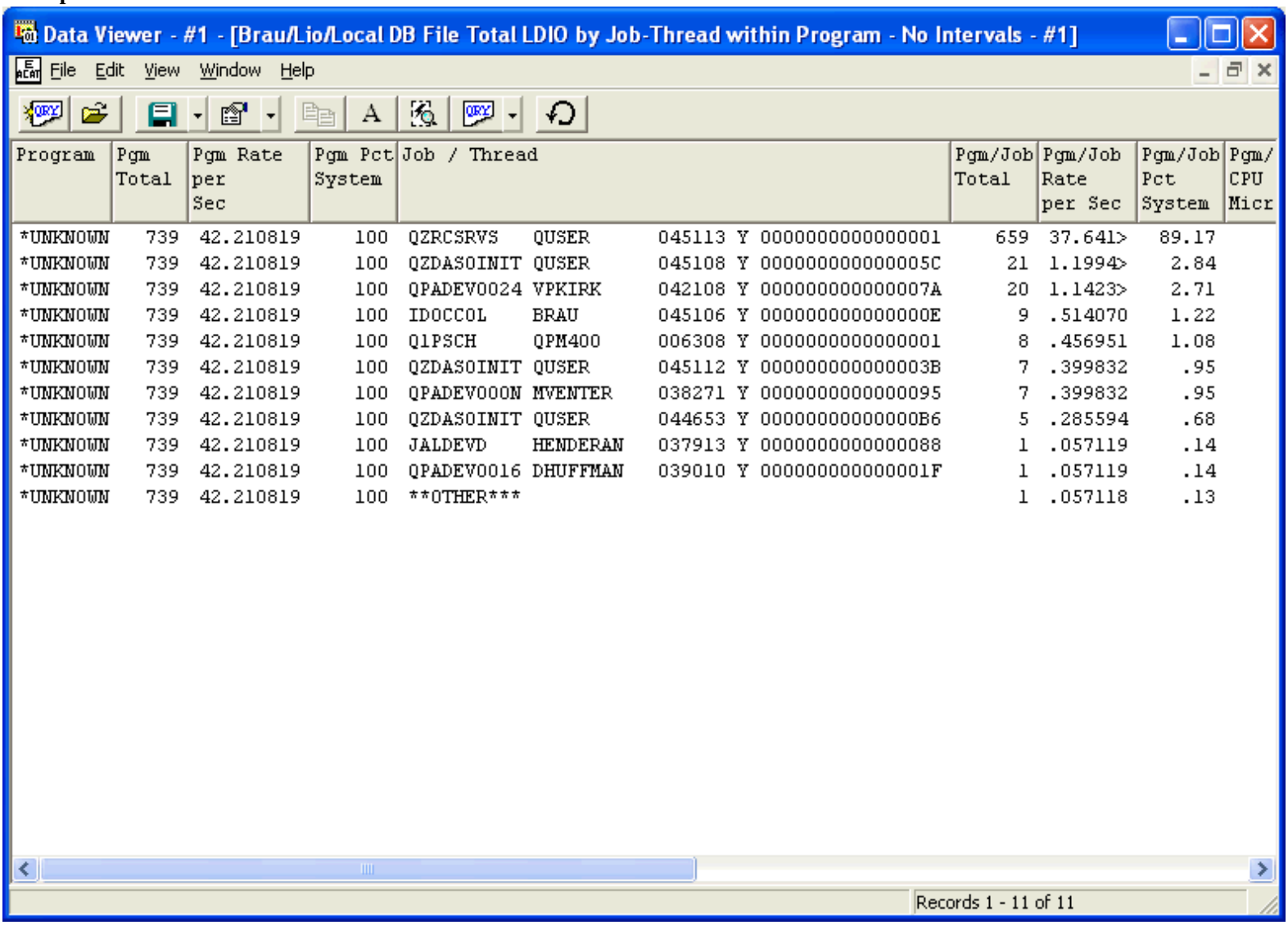
Table of Contents | Previous | Next

5.1.14 DB File Detail LDIO by Job-thread within program - No Intervals

Description: This report shows the program LDIO activity for each job within the collection.

Note: In order for program names to be provided, *MIENTRY and *MIEXIT events must be included in the PEX collection.

Example:



Program	Pgm Total	Pgm Rate per Sec	Pgm Pct System	Job / Thread					Pgm/Job Total	Pgm/Job Rate per Sec	Pgm/Job Pct System	Pgm/ CPU Micr
*UNKNOWN	739	42.210819	100	QZRCSRVS	QUSER	045113	Y	0000000000000001	659	37.641>	89.17	
*UNKNOWN	739	42.210819	100	QZDASOINIT	QUSER	045108	Y	000000000000005C	21	1.1994>	2.84	
*UNKNOWN	739	42.210819	100	QPADEV0024	VPKIRK	042108	Y	000000000000007A	20	1.1423>	2.71	
*UNKNOWN	739	42.210819	100	IDOCCOL	BRAU	045106	Y	000000000000000E	9	.514070	1.22	
*UNKNOWN	739	42.210819	100	Q1PSCH	QPM400	006308	Y	0000000000000001	8	.456951	1.08	
*UNKNOWN	739	42.210819	100	QZDASOINIT	QUSER	045112	Y	000000000000003B	7	.399832	.95	
*UNKNOWN	739	42.210819	100	QPADEV000N	MVENTER	038271	Y	0000000000000095	7	.399832	.95	
*UNKNOWN	739	42.210819	100	QZDASOINIT	QUSER	044653	Y	00000000000000B6	5	.285594	.68	
*UNKNOWN	739	42.210819	100	JALDEVD	HENDERAN	037913	Y	0000000000000088	1	.057119	.14	
*UNKNOWN	739	42.210819	100	QPADEV0016	DHUFFMAN	039010	Y	000000000000001F	1	.057119	.14	
*UNKNOWN	739	42.210819	100	**OTHER***					1	.057118	.13	

Graph Type: stacked vertical bar
X-axis: Program name.
Y-Axis: This value is the number of LDIO ops occurring of each type per second by job within program. Each color represents a different type of job.
Second Y-Axis: This line shows how much program's LDIO ops contributed to the total system LDIOs.

Example:

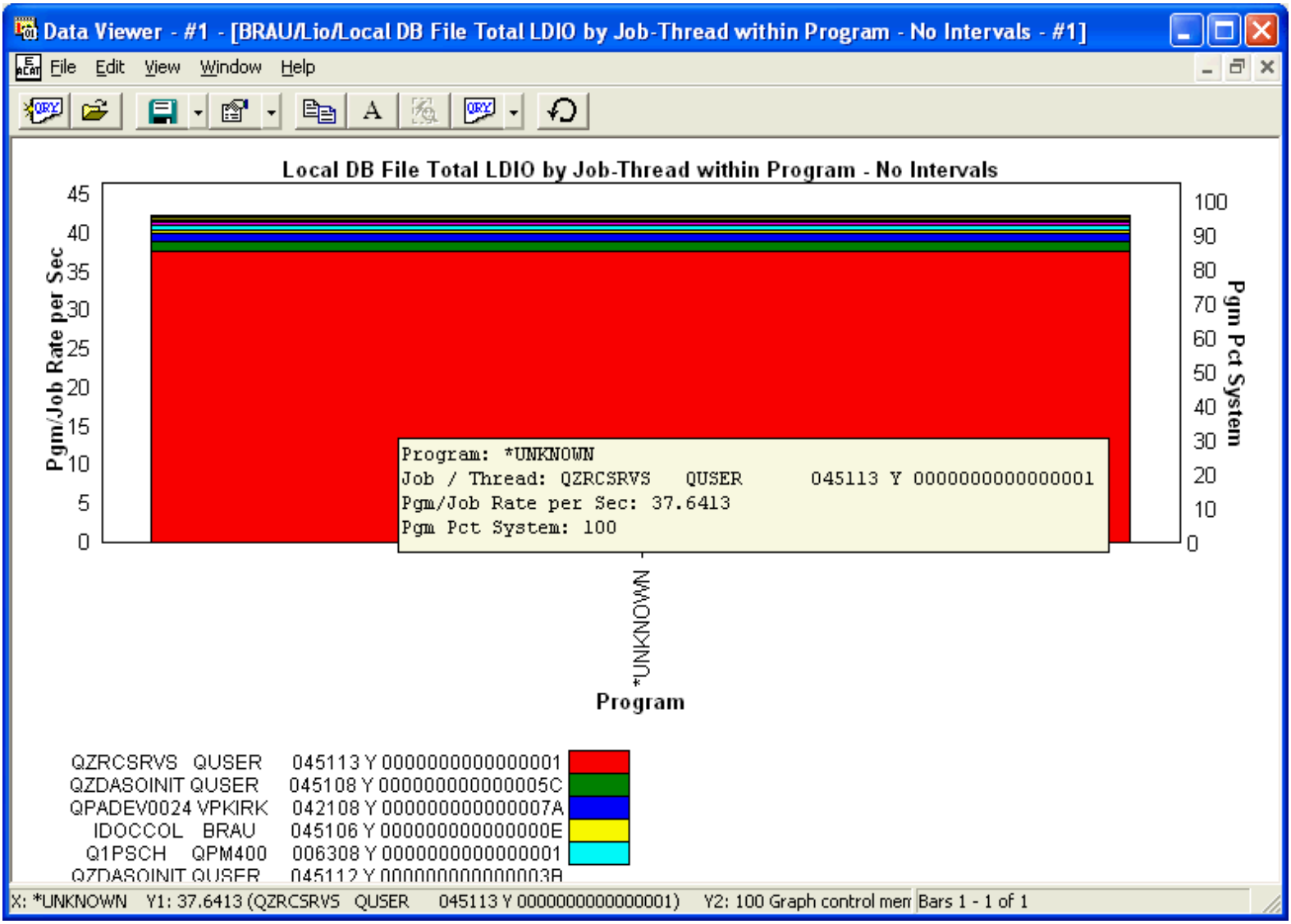

Data Viewer - #1 - [BRAU/Lio/Local DB File Total LDIO by Job-Thread within Program - No Intervals - #1]
File Edit View Window Help
Local DB File Total LDIO by Job-Thread within Program - No Intervals
Pgm/Job Rate per Sec
Pgm Pct System
Program: *UNKNOWN
Job / Thread: QZRCSRVS QUSER 045113 Y 0000000000000001
Pgm/Job Rate per Sec: 37.6413
Pgm Pct System: 100
*UNKNOWN
Program
QZRCSRVS QUSER 045113 Y 0000000000000001
QZDASOINIT QUSER 045108 Y 000000000000005C
QPADEV0024 VPKIRK 042108 Y 000000000000007A
IDOCCOL BRAU 045106 Y 000000000000000E
Q1PSCH QPM400 006308 Y 0000000000000001
X: *UNKNOWN Y1: 37.6413 (QZRCSRVS QUSER 045113 Y 0000000000000001) Y2: 100 Graph control men
Bars 1 - 1 of 1

Table of Contents | Previous | Next

5.1.15 DB File Detail LDIO by Program within File-library - No Intervals

Description: This report shows the program LDIO activity for each file within the collection.

Note: In order for program names to be provided, *MIENTRY and *MIEXIT events must be included in the PEX collection.

Example:

Data Viewer: #1 - [Local DB File Total LDIO by Program within File-Library - No Intervals [Idocdemo/G...

File Edit View Window Help

File-Library		File-Lib Total	File-Lib Rate per Sec	File-Lib Pct System	Program	F
CLIENTL1	PEXHATT1	132796	1060.767853	84.14	IRDKS	
CLIENTL1	PEXHATT1	132796	1060.767853	84.14	IRDKS3	
CLIENTL1	PEXHATT1	132796	1060.767853	84.14	IUPKS	
CLIENTL1	PEXHATT1	132796	1060.767853	84.14	IUPKS1	
CLIENTL1	PEXHATT1	132796	1060.767853	84.14	IRDKS1	
CLIENTL1	PEXHATT1	132796	1060.767853	84.14	IRDKS2	
CLIENTL1	PEXHATT1	132796	1060.767853	84.14	IDLKS	
CLIENTL1	PEXHATT1	132796	1060.767853	84.14	IWRK	
CLIENTL1	PEXHATT1	132796	1060.767853	84.14	IWR	
CLIENTL1	PEXHATT1	132796	1060.767853	84.14	IRDKC	
CLIENT	PEXHATT1	24963	199.403204	15.82	IRDPF	
QATOCTCPIP	QUSRSYS	23	.183723	.01	QDBGETSQ	
QA1PSCL2	QMPGLIB	20	.159759	.01	QDBGETSQ	
QA1PSCL2	QMPGLIB	20	.159759	.01	QDBGETKY	
QAPMMIOP	QMPGDATA	7	.055916	0	QDBPUTM	
QAPMSNA	QMPGDATA	4	.031952	0	QDBPUTM	
QADBXLFN	QSYS	4	.031952	0	QDBGETKY	
QAPMJOBL	QMPGDATA	3	.023964	0	QDBGETM	
DFM_A00002	QDLFM	2	.015976	0	QDBGETM	
QDNDLLDR	QDMT	2	.015976	0	QDBGETKY	
QDNDLLDR	QDMT	2	.015976	0	QDBGETSQ	
QA1ANET	QUSRBRM	2	.015976	0	QDBGETSQ	
QAPMAPPN	QMPGDATA	1	.007988	0	QDBPUTM	
QAPMDISK	QMPGDATA	1	.007988	0	QDBPUTM	
QAPMJOBOS	QMPGDATA	1	.007988	0	QDBPUTM	

Records 1 - 24 of 26

Graph Type: stacked vertical bar

X-axis: File and library name. The entire field is shown by placing the mouse over a bar.
Y-Axis: This value is the number of LDIO ops occurring per second for a file within program. Each color represents a different program.
Second Y-Axis: This line shows how much each file/library's LDIO ops contributed to the total system LDIOs.

Example:

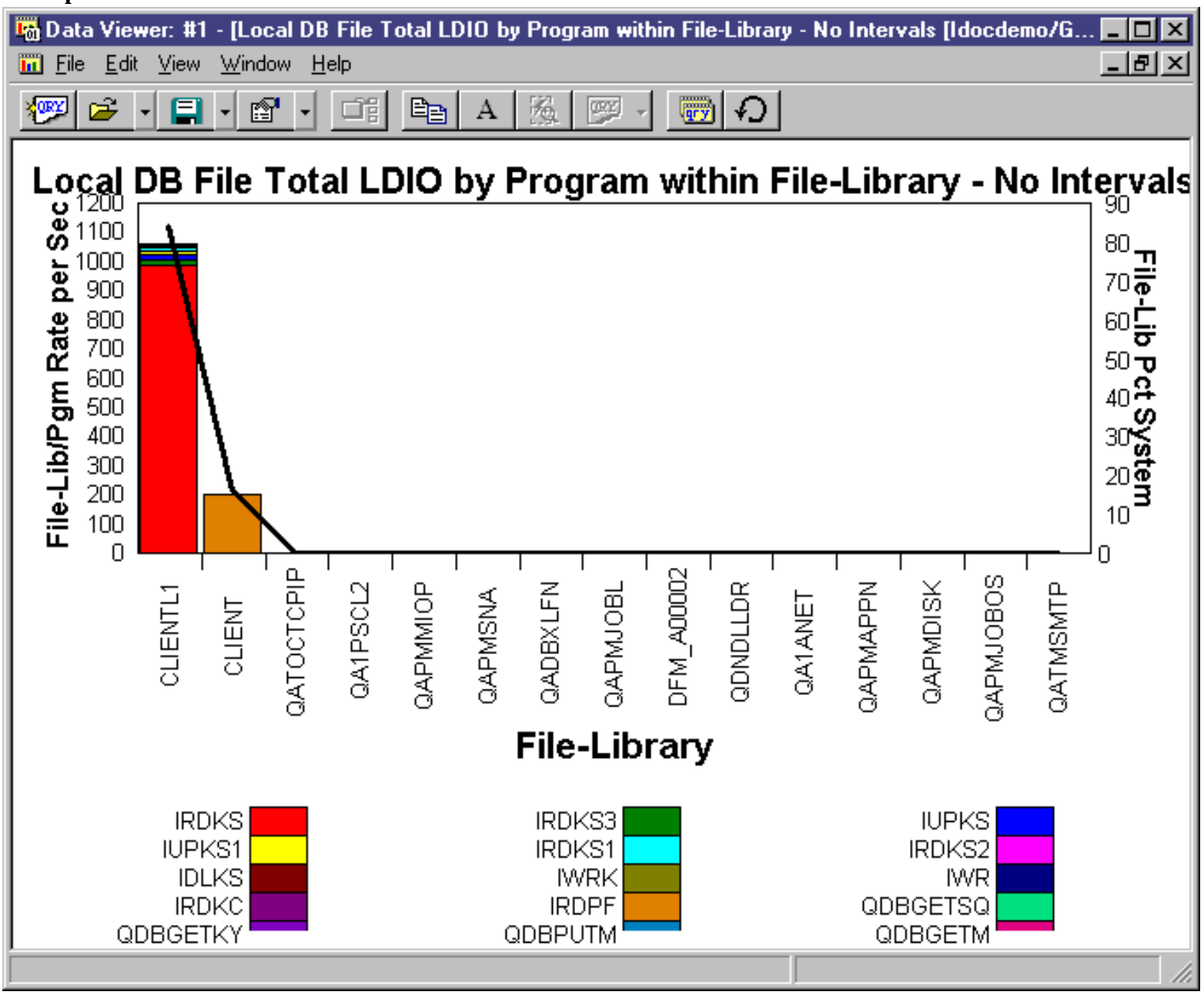


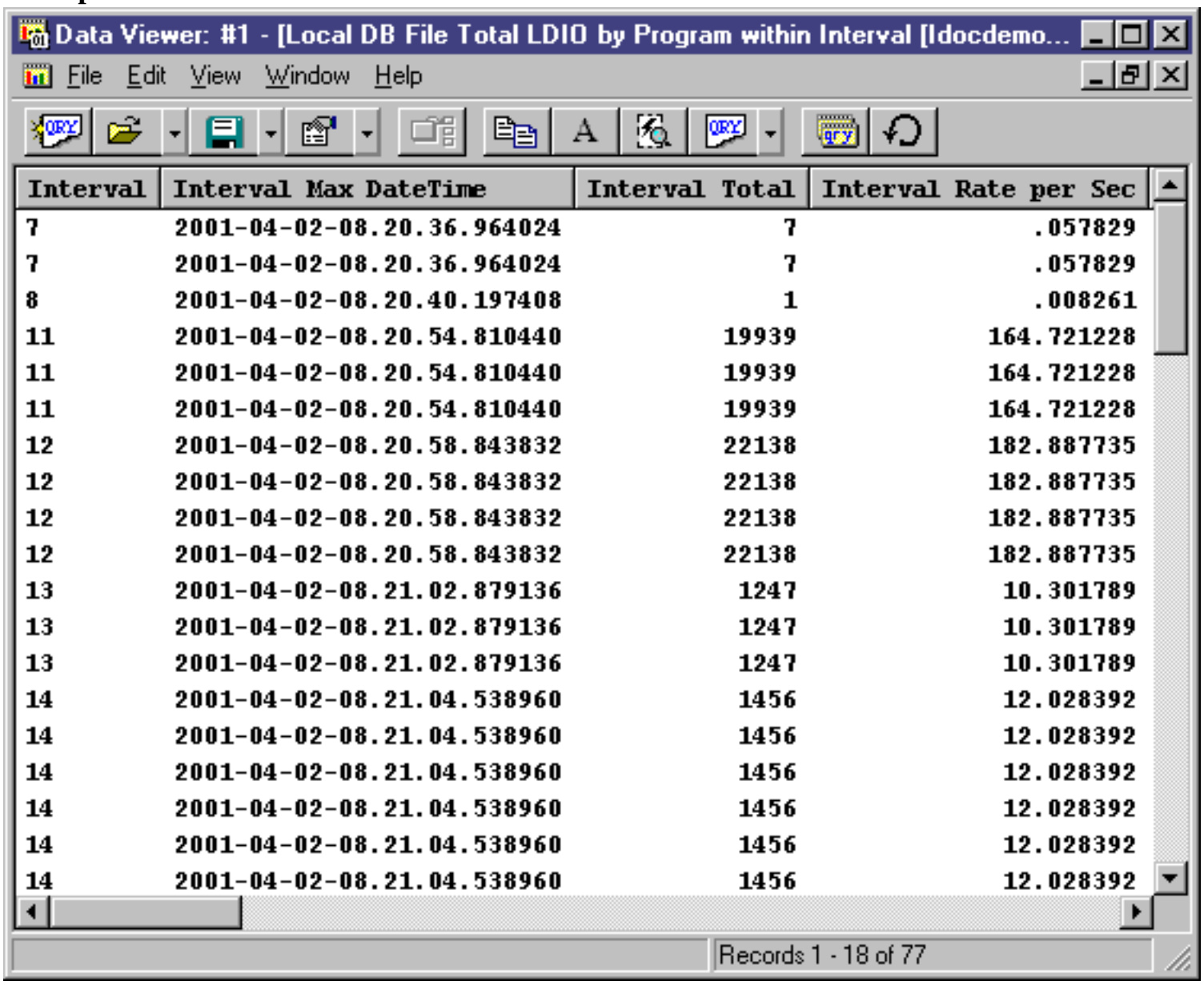
Table of Contents | Previous | Next

5.1.16 DB File Detail LDIO by Program within Interval

Description: This report shows the program LDIO activity for each interval in the collection.

Note: In order for program names to be provided, *MIENTRY and *MIEXIT events must be included in the PEX collection.

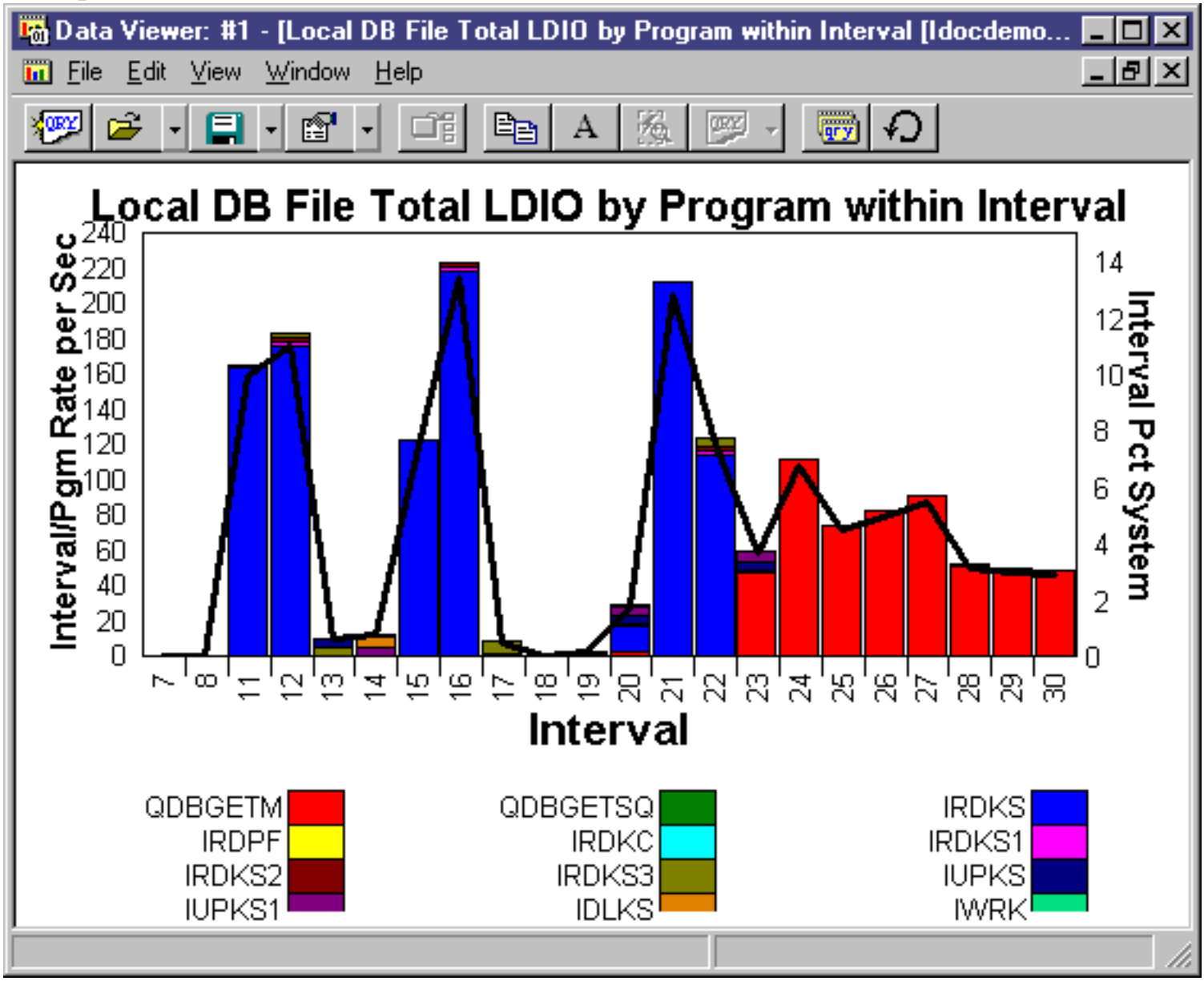
Example:



Interval	Interval Max DateTime	Interval Total	Interval Rate per Sec
7	2001-04-02-08.20.36.964024	7	.057829
7	2001-04-02-08.20.36.964024	7	.057829
8	2001-04-02-08.20.40.197408	1	.008261
11	2001-04-02-08.20.54.810440	19939	164.721228
11	2001-04-02-08.20.54.810440	19939	164.721228
11	2001-04-02-08.20.54.810440	19939	164.721228
12	2001-04-02-08.20.58.843832	22138	182.887735
12	2001-04-02-08.20.58.843832	22138	182.887735
12	2001-04-02-08.20.58.843832	22138	182.887735
12	2001-04-02-08.20.58.843832	22138	182.887735
13	2001-04-02-08.21.02.879136	1247	10.301789
13	2001-04-02-08.21.02.879136	1247	10.301789
13	2001-04-02-08.21.02.879136	1247	10.301789
14	2001-04-02-08.21.04.538960	1456	12.028392
14	2001-04-02-08.21.04.538960	1456	12.028392
14	2001-04-02-08.21.04.538960	1456	12.028392
14	2001-04-02-08.21.04.538960	1456	12.028392
14	2001-04-02-08.21.04.538960	1456	12.028392
14	2001-04-02-08.21.04.538960	1456	12.028392

Graph Type: stacked vertical bar
X-axis: Interval number
Y-Axis: This value is the number of LDIO ops occurring per second by program within interval. Each color represents a different program
Second Y-Axis: This line shows how much each collection interval's total LDIOs contributed to the total system LDIOs.

Example:

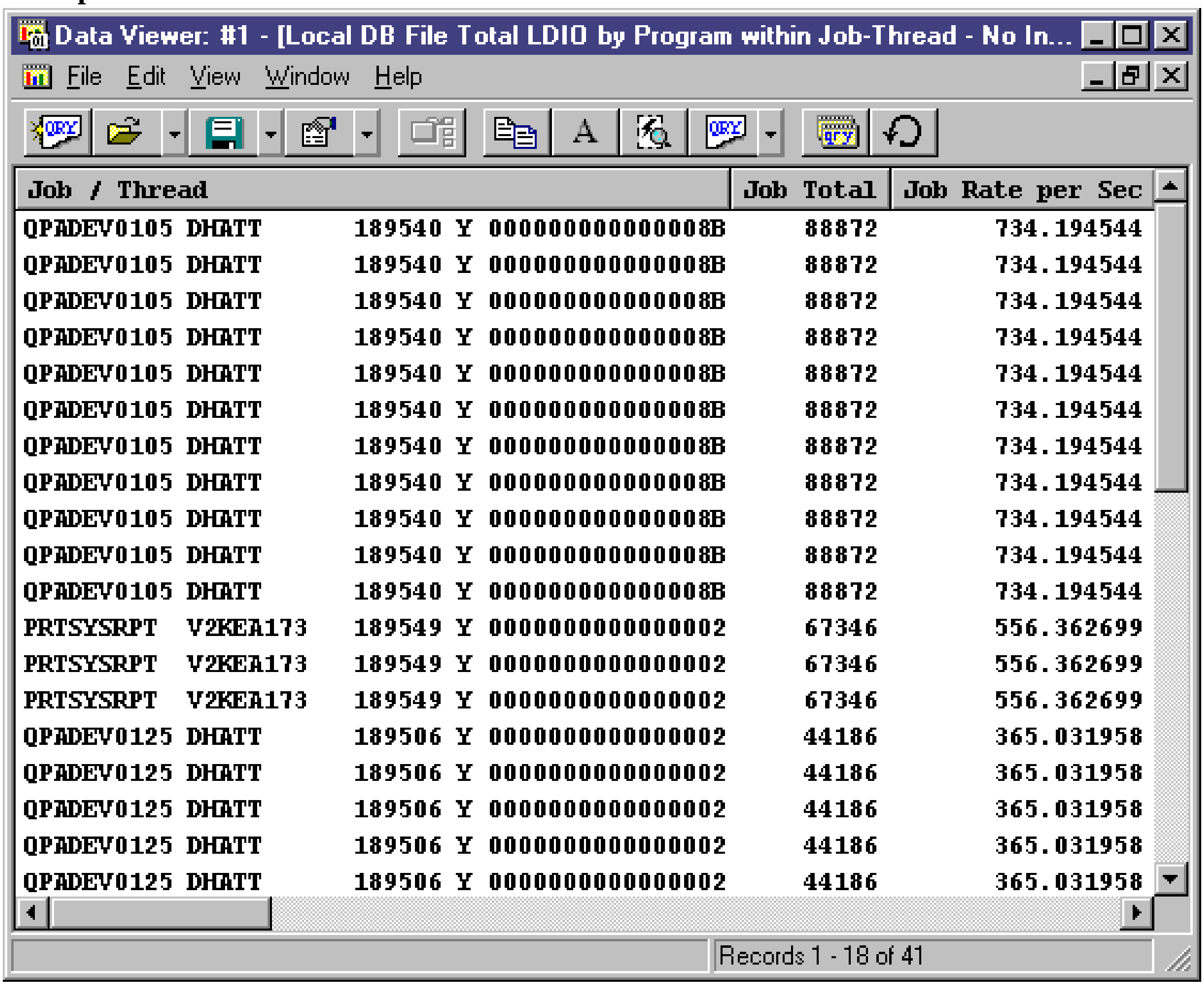


5.1.17 DB File Detail LDIO by Program within Job-thread - No Intervals

Description: This report shows the program LDIO activity for each job in the collection. Each color in the graph represents a unique program.

Note: In order for program names to be provided, *MIENTRY and *MIEXIT events must be included in the PEX collection.

Example:



Job / Thread	Job Total	Job Rate per Sec
QPADEV0105 DHATT 189540 Y 000000000000008B	88872	734.194544
QPADEV0105 DHATT 189540 Y 000000000000008B	88872	734.194544
QPADEV0105 DHATT 189540 Y 000000000000008B	88872	734.194544
QPADEV0105 DHATT 189540 Y 000000000000008B	88872	734.194544
QPADEV0105 DHATT 189540 Y 000000000000008B	88872	734.194544
QPADEV0105 DHATT 189540 Y 000000000000008B	88872	734.194544
QPADEV0105 DHATT 189540 Y 000000000000008B	88872	734.194544
QPADEV0105 DHATT 189540 Y 000000000000008B	88872	734.194544
QPADEV0105 DHATT 189540 Y 000000000000008B	88872	734.194544
QPADEV0105 DHATT 189540 Y 000000000000008B	88872	734.194544
QPADEV0105 DHATT 189540 Y 000000000000008B	88872	734.194544
PRTSYSRPT V2KEA173 189549 Y 0000000000000002	67346	556.362699
PRTSYSRPT V2KEA173 189549 Y 0000000000000002	67346	556.362699
PRTSYSRPT V2KEA173 189549 Y 0000000000000002	67346	556.362699
QPADEV0125 DHATT 189506 Y 0000000000000002	44186	365.031958
QPADEV0125 DHATT 189506 Y 0000000000000002	44186	365.031958
QPADEV0125 DHATT 189506 Y 0000000000000002	44186	365.031958
QPADEV0125 DHATT 189506 Y 0000000000000002	44186	365.031958
QPADEV0125 DHATT 189506 Y 0000000000000002	44186	365.031958

Graph Type: stacked vertical bar
X-axis: Fully qualified job name and thread ID. The entire field is shown by placing the mouse over a bar.
Y-Axis: This value is the number of LDIO ops occurring per second for each program within job.. Each color represents a different program.
Second Y-Axis: This line shows how much each job's LDIO ops contributed to the total system LDIOs.

Example:

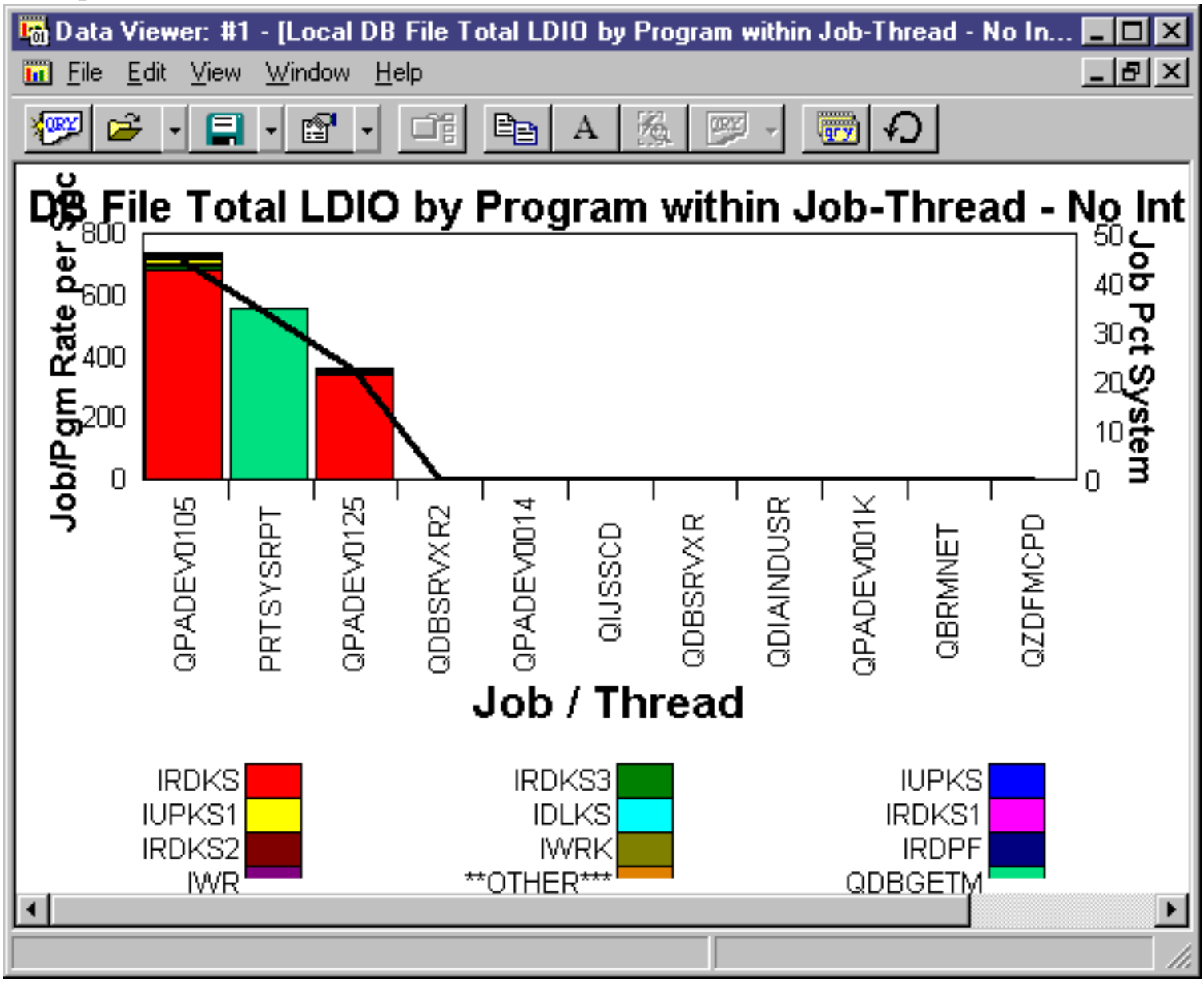


Table of Contents Previous Next

5.2 CPU Profile by job priority reports

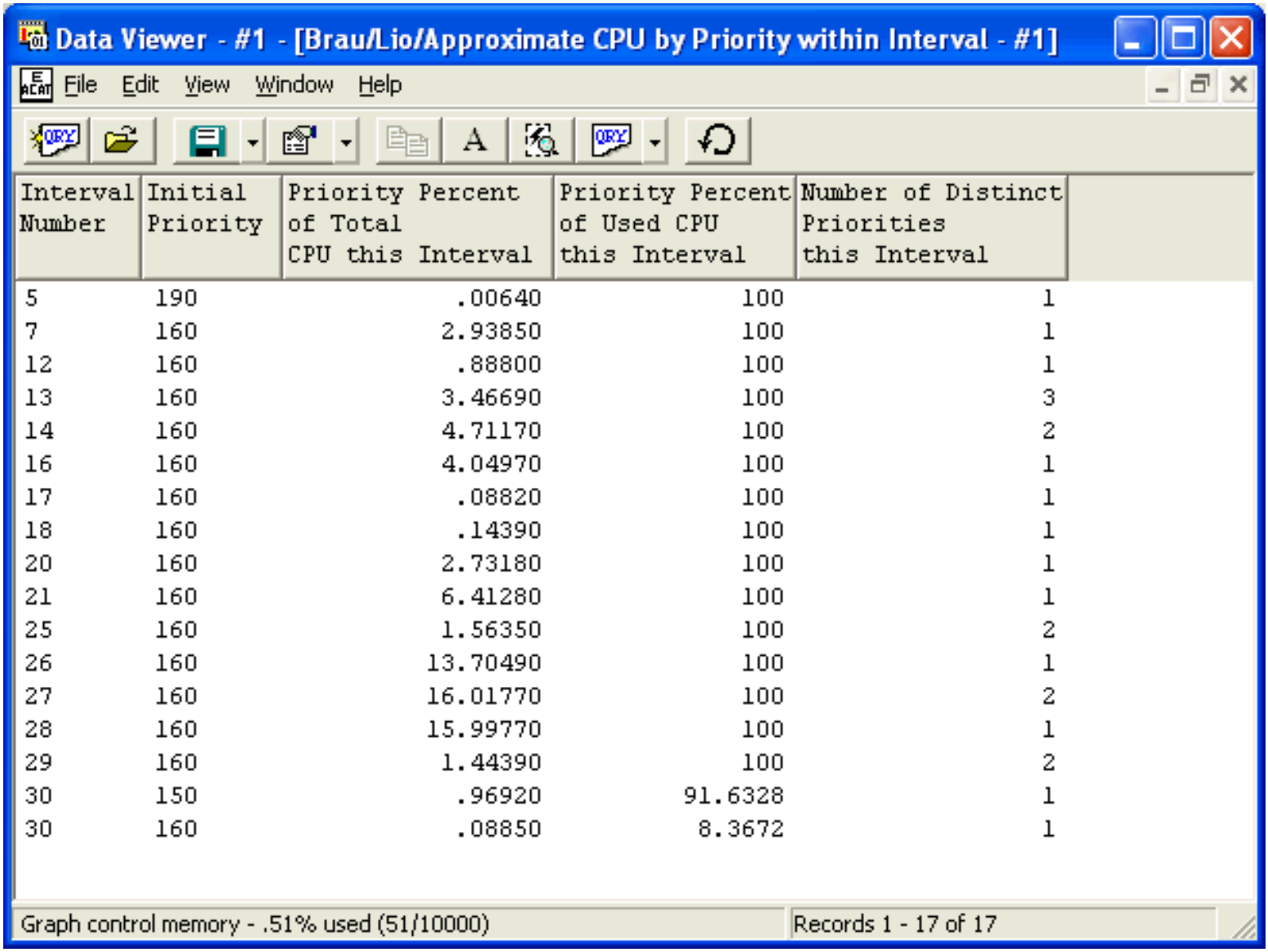
This analysis shows the job/task CPU usage percentage by time interval. Alternately you can view the CPU usage by job priority over time.

Table of Contents | Previous | Next

5.2.1 Approximate CPU by priority within Interval

Description: This report shows the approximate cpu usage by priority within an interval. The user can subset the analysis to any time and divide the subset time into many intervals and observe the CPU usage for each interval of time. Each priority is color coded to take a particular color. The user can get additional information by clicking on any bar in the graph. This information includes interval number, priority, percent of CPU usage, CPU percent usage in this interval, and the number of different job priorities found.

Example:



Interval Number	Initial Priority	Priority Percent of Total CPU this Interval	Priority Percent of Used CPU this Interval	Number of Distinct Priorities this Interval
5	190	.00640	100	1
7	160	2.93850	100	1
12	160	.88800	100	1
13	160	3.46690	100	3
14	160	4.71170	100	2
16	160	4.04970	100	1
17	160	.08820	100	1
18	160	.14390	100	1
20	160	2.73180	100	1
21	160	6.41280	100	1
25	160	1.56350	100	2
26	160	13.70490	100	1
27	160	16.01770	100	2
28	160	15.99770	100	1
29	160	1.44390	100	2
30	150	.96920	91.6328	1
30	160	.08850	8.3672	1

Graph Type: stacked vertical bar
X-axis: Interval number
Y-Axis: This value is the approximate percentage of the total system CPU used by a priority value. Each priority value is represented by a different color.

Example:

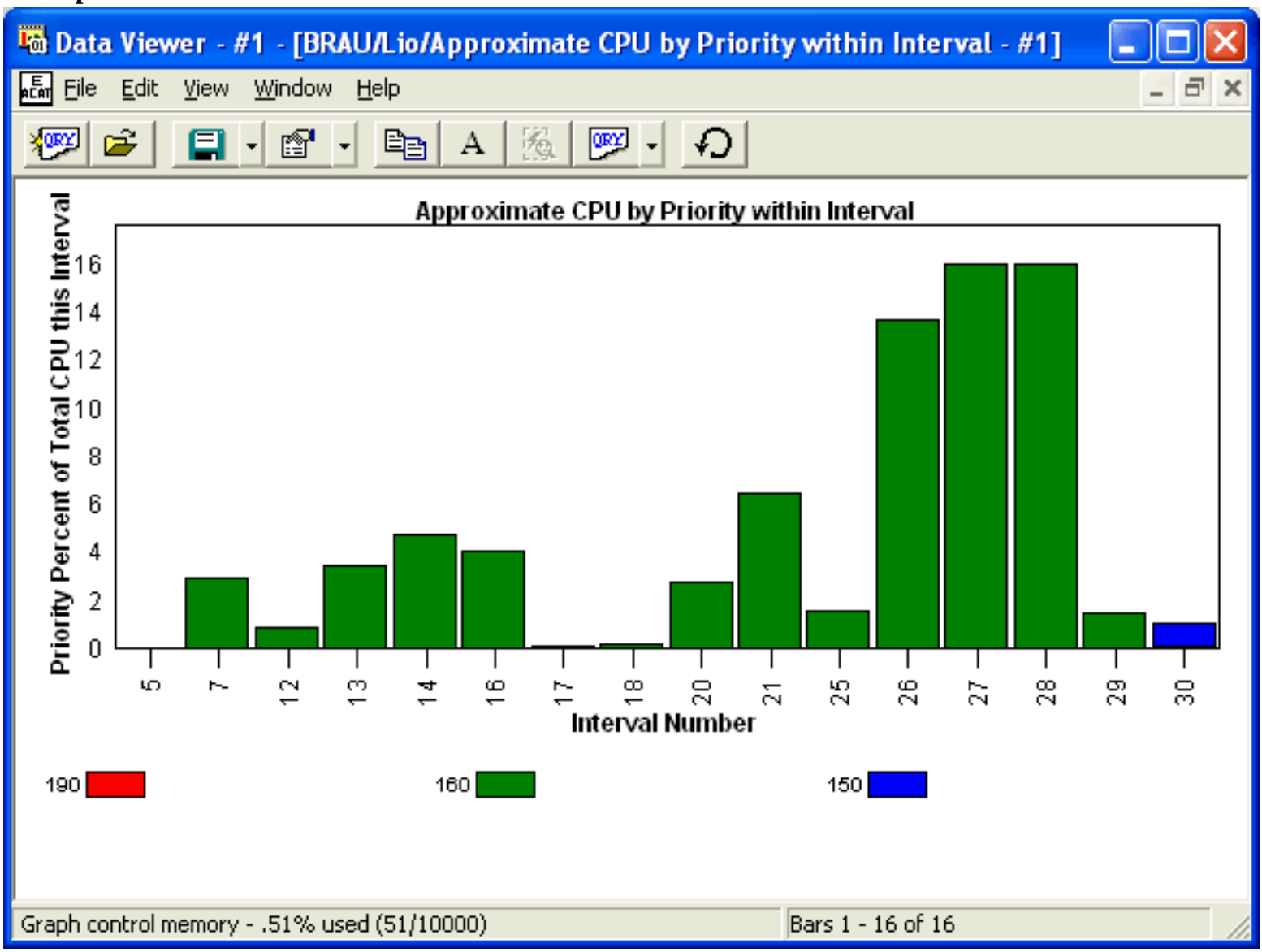
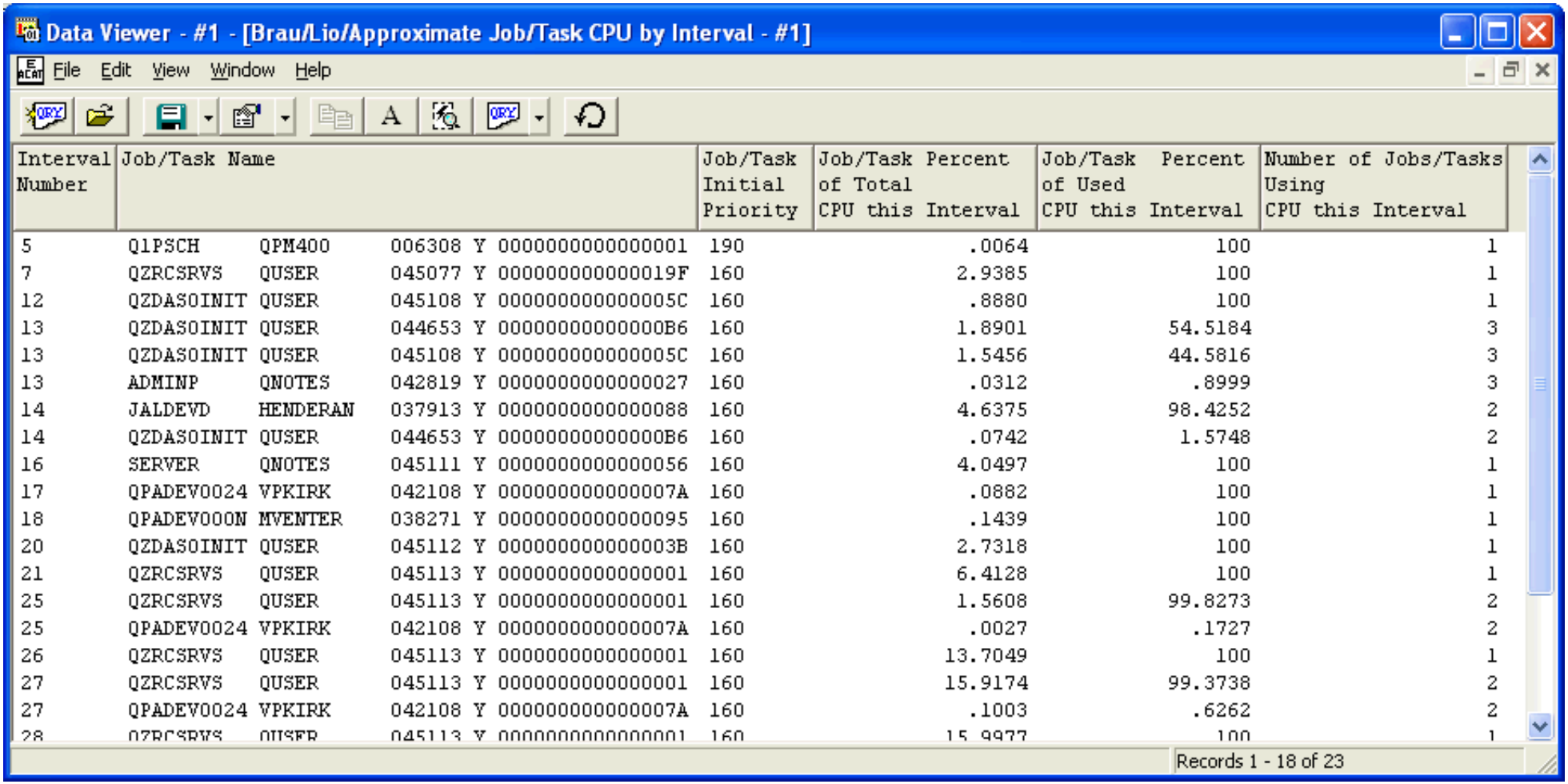


Table of Contents | Previous | Next

5.2.2 Approximate Job/Task CPU by interval

This report displays information about the CPU usage of jobs for each interval within the PEX collection.



Interval Number	Job/Task Name				Job/Task Initial Priority	Job/Task Percent of Total CPU this Interval	Job/Task Percent of Used CPU this Interval	Number of Jobs/Tasks Using CPU this Interval
5	Q1PSCH	QPM400	006308	Y 0000000000000001	190	.0064	100	1
7	QZRCSRVS	QUSER	045077	Y 000000000000019F	160	2.9385	100	1
12	QZDASOINIT	QUSER	045108	Y 000000000000005C	160	.8880	100	1
13	QZDASOINIT	QUSER	044653	Y 00000000000000B6	160	1.8901	54.5184	3
13	QZDASOINIT	QUSER	045108	Y 000000000000005C	160	1.5456	44.5816	3
13	ADMINP	QNOTES	042819	Y 0000000000000027	160	.0312	.8999	3
14	JALDEVD	HENDERAN	037913	Y 0000000000000088	160	4.6375	98.4252	2
14	QZDASOINIT	QUSER	044653	Y 00000000000000B6	160	.0742	1.5748	2
16	SERVER	QNOTES	045111	Y 0000000000000056	160	4.0497	100	1
17	QPADEV0024	VPKIRK	042108	Y 000000000000007A	160	.0882	100	1
18	QPADEV000N	MVENTER	038271	Y 0000000000000095	160	.1439	100	1
20	QZDASOINIT	QUSER	045112	Y 000000000000003B	160	2.7318	100	1
21	QZRCSRVS	QUSER	045113	Y 0000000000000001	160	6.4128	100	1
25	QZRCSRVS	QUSER	045113	Y 0000000000000001	160	1.5608	99.8273	2
25	QPADEV0024	VPKIRK	042108	Y 000000000000007A	160	.0027	.1727	2
26	QZRCSRVS	QUSER	045113	Y 0000000000000001	160	13.7049	100	1
27	QZRCSRVS	QUSER	045113	Y 0000000000000001	160	15.9174	99.3738	2
27	QPADEV0024	VPKIRK	042108	Y 000000000000007A	160	.1003	.6262	2
28	QZRCSRVS	QUSER	045113	Y 0000000000000001	160	15.9977	100	1

Graph Type: stacked vertical bar
X-axis: Interval number
Y-Axis: This value is the approximate percentage of total CPU used by a job each interval. Each color represents a different job.

Example:

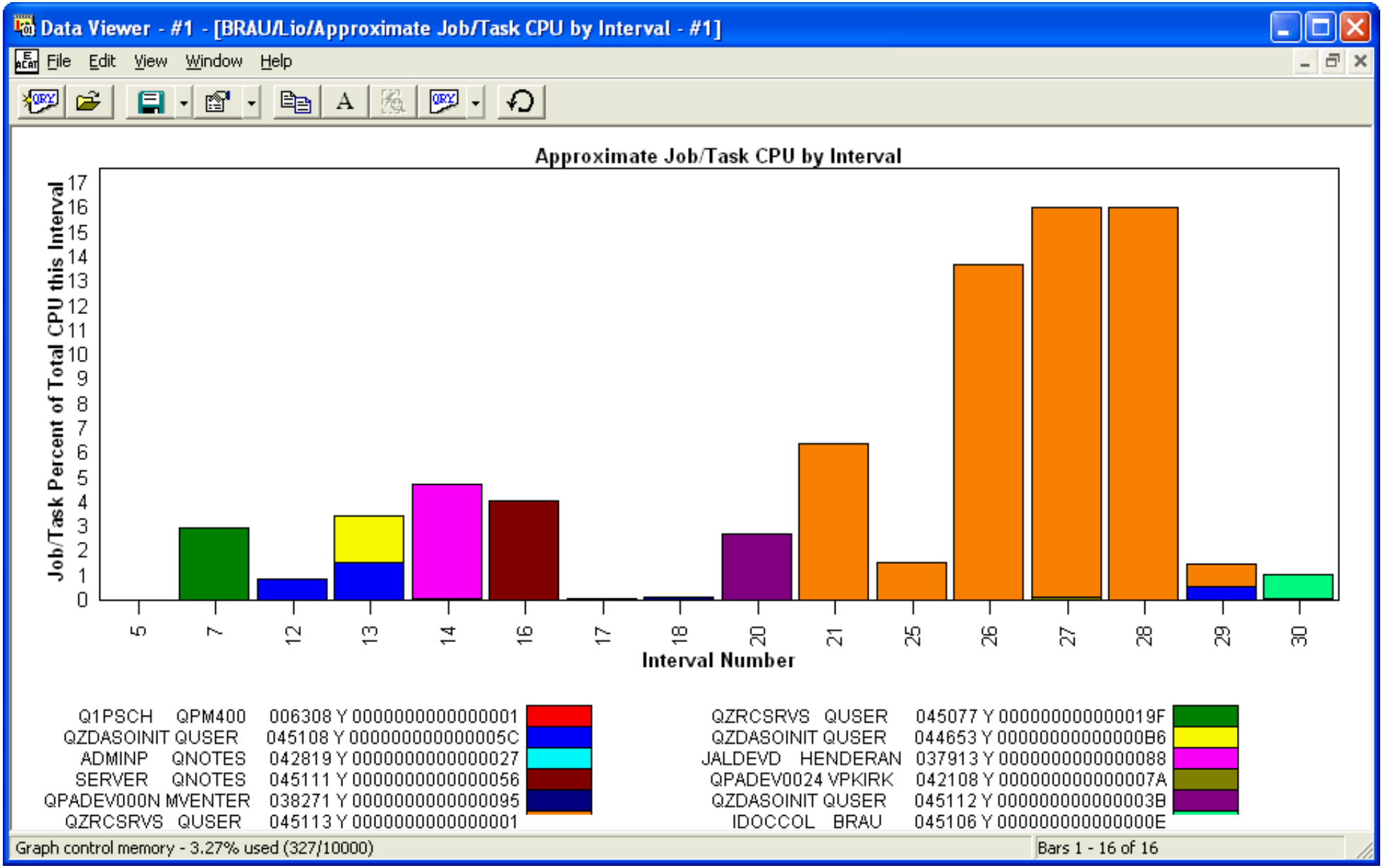

Data Viewer - #1 - [BRAU/Lio/Approximate Job/Task CPU by Interval - #1]
File Edit View Window Help
Approximate Job/Task CPU by Interval
Job/Task Percent of Total CPU this Interval
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
5 7 12 13 14 16 17 18 20 21 25 26 27 28 29 30
Interval Number
Q1PSCH QPM400 006308 Y 0000000000000001
QZDASOINIT QUSER 045108 Y 000000000000005C
ADMINP QNOTES 042819 Y 0000000000000027
SERVER QNOTES 045111 Y 0000000000000056
QPADEV000N MVENTER 038271 Y 0000000000000095
QZRCSRVS QUSER 045113 Y 0000000000000001
QZRCSRVS QUSER 045077 Y 000000000000019F
QZDASOINIT QUSER 044653 Y 00000000000000B6
JALDEVD HENDERAN 037913 Y 0000000000000088
QPADEV0024 VPKIRK 042108 Y 000000000000007A
QZDASOINIT QUSER 045112 Y 000000000000003B
IDOCCOL BRAU 045106 Y 000000000000000E
Graph control memory - 3.27% used (327/10000)
Bars 1 - 16 of 16

Table of Contents | Previous | Next

5.3 CPU Profile Summary (TPROF) reports

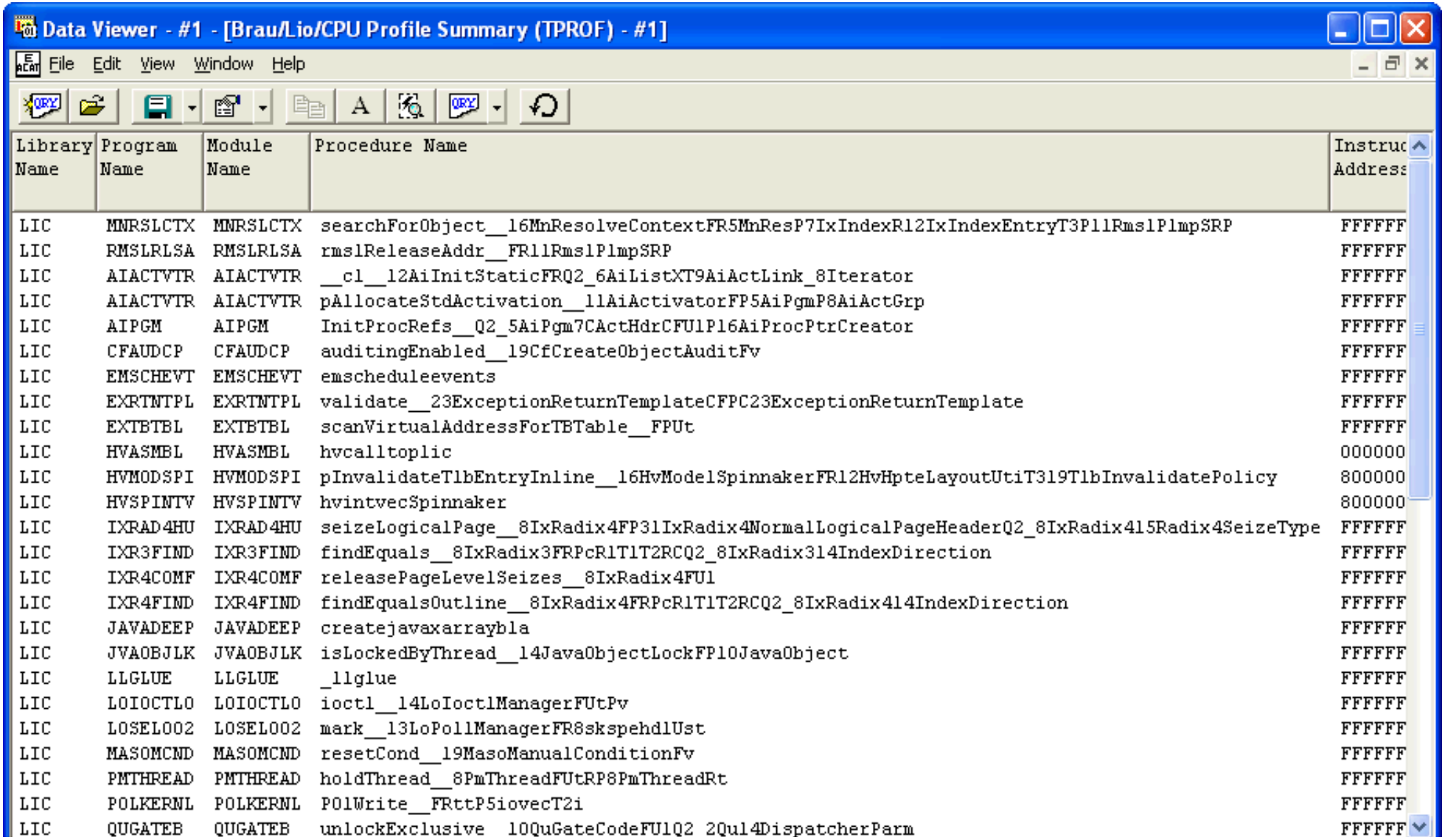
The TPROF analysis shows CPU usage for ALL types of executable code, including SLIC programs or "unhooked" programs, running in the selected job/task(s).

TPROF shows the run time CPU execution "hot spots" and indicates what programs are being used that are consuming the CPU.

5.3.1 CPU Profile Summary (TPROF)

This report shows the procedures/programs and modules that used the most CPU in a PEX trace collection.

PMCO events much be collected in order to produce this analysis report.



Library Name	Program Name	Module Name	Procedure Name	Instruc Address
LIC	MNRSLCTX	MNRSLCTX	searchForObject__16MnResolveContextFR5MnResP7IxIndexR12IxIndexEntryT3P11RmslPlmpSRP	FFFFFF
LIC	RMSLRLSA	RMSLRLSA	rmslReleaseAddr__FR11RmslPlmpSRP	FFFFFF
LIC	AIACTVTR	AIACTVTR	__cl__12AiInitStaticFRQ2_6AiListXT9AiActLink_8Iterator	FFFFFF
LIC	AIACTVTR	AIACTVTR	pAllocateStdActivation__11AiActivatorFP5AiPgmP8AiActGrp	FFFFFF
LIC	AIPGM	AIPGM	InitProcRefs__Q2_5AiPgm7CActHdrCFU1P16AiProcPtrCreator	FFFFFF
LIC	CFAUDCP	CFAUDCP	auditingEnabled__19CfCreateObjectAuditFv	FFFFFF
LIC	EMSCHEVT	EMSCHEVT	emscheduleevents	FFFFFF
LIC	EXRTNTPL	EXRTNTPL	validate__23ExceptionReturnTemplateCFPC23ExceptionReturnTemplate	FFFFFF
LIC	EXTBTBL	EXTBTBL	scanVirtualAddressForTBTable__FPUt	FFFFFF
LIC	HVASMBL	HVASMBL	hvcalltoplic	000000
LIC	HVMODSPI	HVMODSPI	pInvalidateTlbEntryInline__16HvModelSpinnakerFR12HvHpteLayoutUtiT3l9TlbInvalidatePolicy	800000
LIC	HVSPINTV	HVSPINTV	hvintvecSpinnaker	800000
LIC	IXRAD4HU	IXRAD4HU	seizeLogicalPage__8IxRadix4FP31IxRadix4NormalLogicalPageHeaderQ2_8IxRadix4l5Radix4SeizeType	FFFFFF
LIC	IXR3FIND	IXR3FIND	findEquals__8IxRadix3FRPcR1T1T2RCQ2_8IxRadix3l4IndexDirection	FFFFFF
LIC	IXR4COMF	IXR4COMF	releasePageLevelSeizes__8IxRadix4FU1	FFFFFF
LIC	IXR4FIND	IXR4FIND	findEqualsOutline__8IxRadix4FRPcR1T1T2RCQ2_8IxRadix4l4IndexDirection	FFFFFF
LIC	JAVADEEP	JAVADEEP	createjavaxarraybla	FFFFFF
LIC	JVAOBJLK	JVAOBJLK	isLockedByThread__14JavaObjectLockFP10JavaObject	FFFFFF
LIC	LLGLUE	LLGLUE	_llglue	FFFFFF
LIC	LOIOCTLO	LOIOCTLO	ioctl__14LoIoctlManagerFUtPv	FFFFFF
LIC	LOSELOO2	LOSELOO2	mark__13LoPollManagerFR8skspehdlUst	FFFFFF
LIC	MASOMCND	MASOMCND	resetCond__19MasoManualConditionFv	FFFFFF
LIC	PMTHREAD	PMTHREAD	holdThread__8PmThreadFUtRP8PmThreadRt	FFFFFF
LIC	POLKERNL	POLKERNL	POlWrite__FRttP5iovecT2i	FFFFFF
LIC	QUGATEB	QUGATEB	unlockExclusive__10QuGateCodeFU1Q2_2Qul4DispatcherParm	FFFFFF

Records 1 - 24 of 46

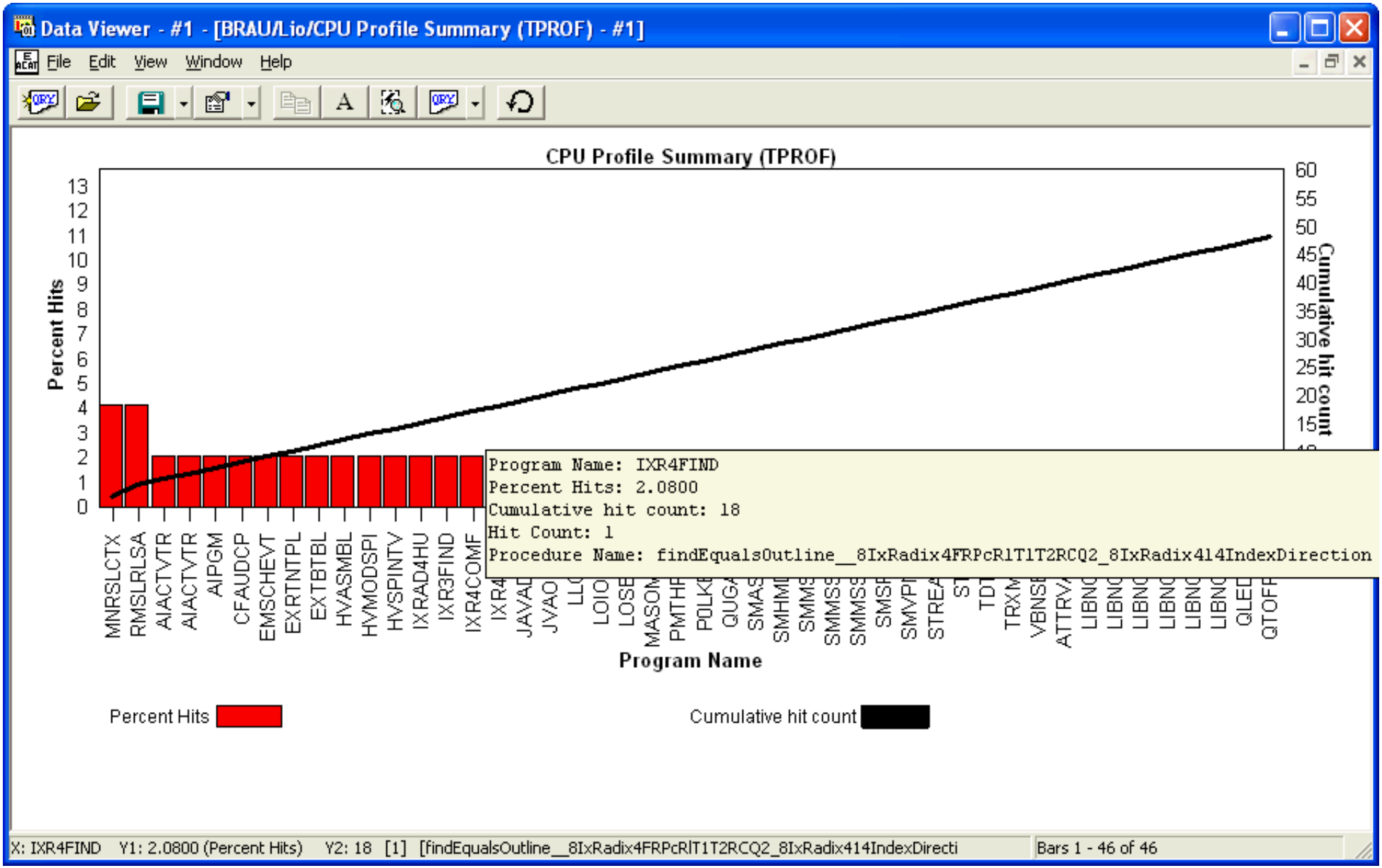


Table of Contents | Previous | Next

5.4 Events count reports

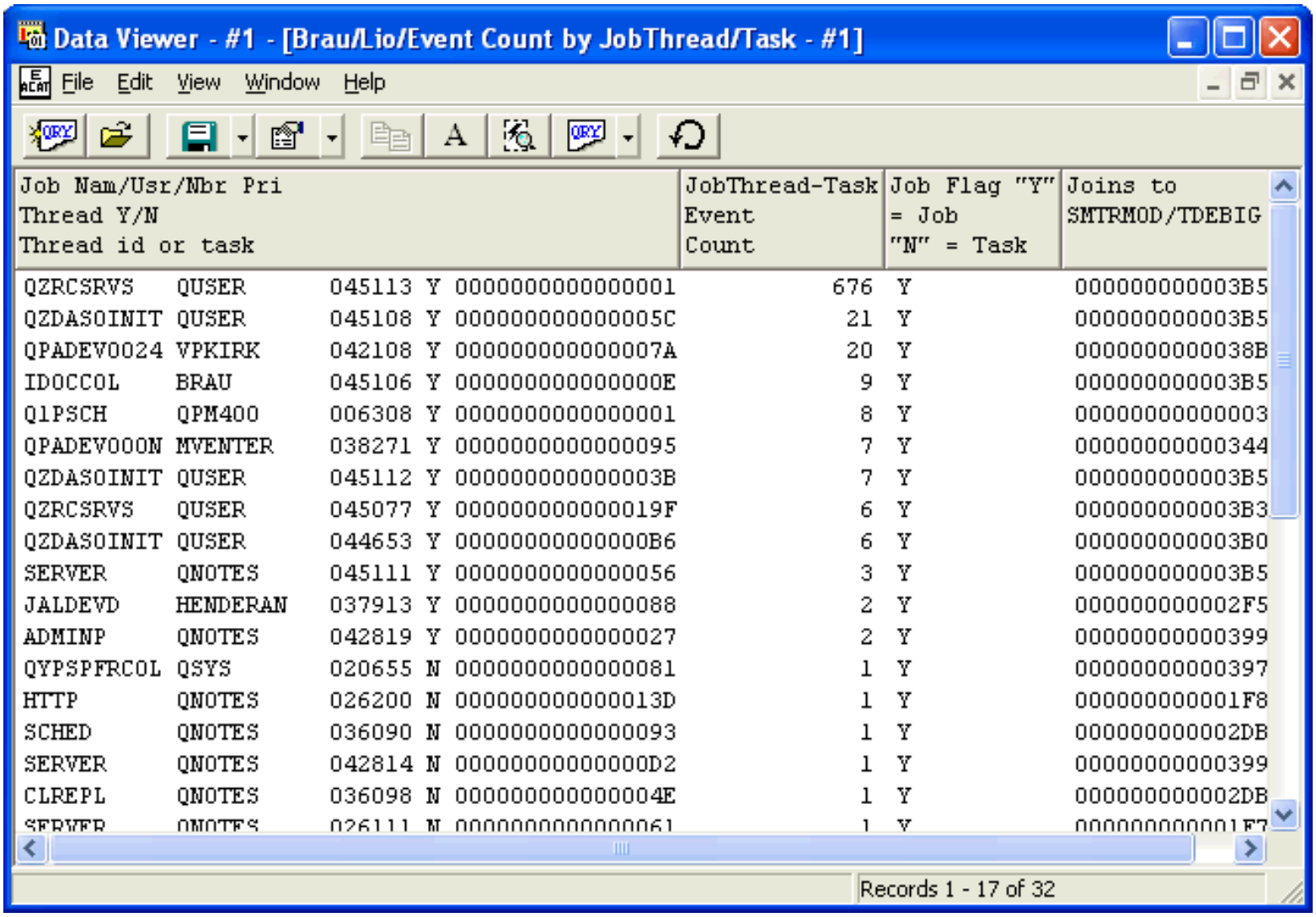
This analysis provides a breakdown of the number of PEX events that were collected per job/task in the PEX collection.

In addition to seeing the total number of PEX events per job/task, you can also see the totals for each event type/subtype combination per job/task.

Table of Contents | Previous | Next

5.4.1 Event Count by JobThread/Task

This report shows how many PEX events were captured for each job in the PEX collection.

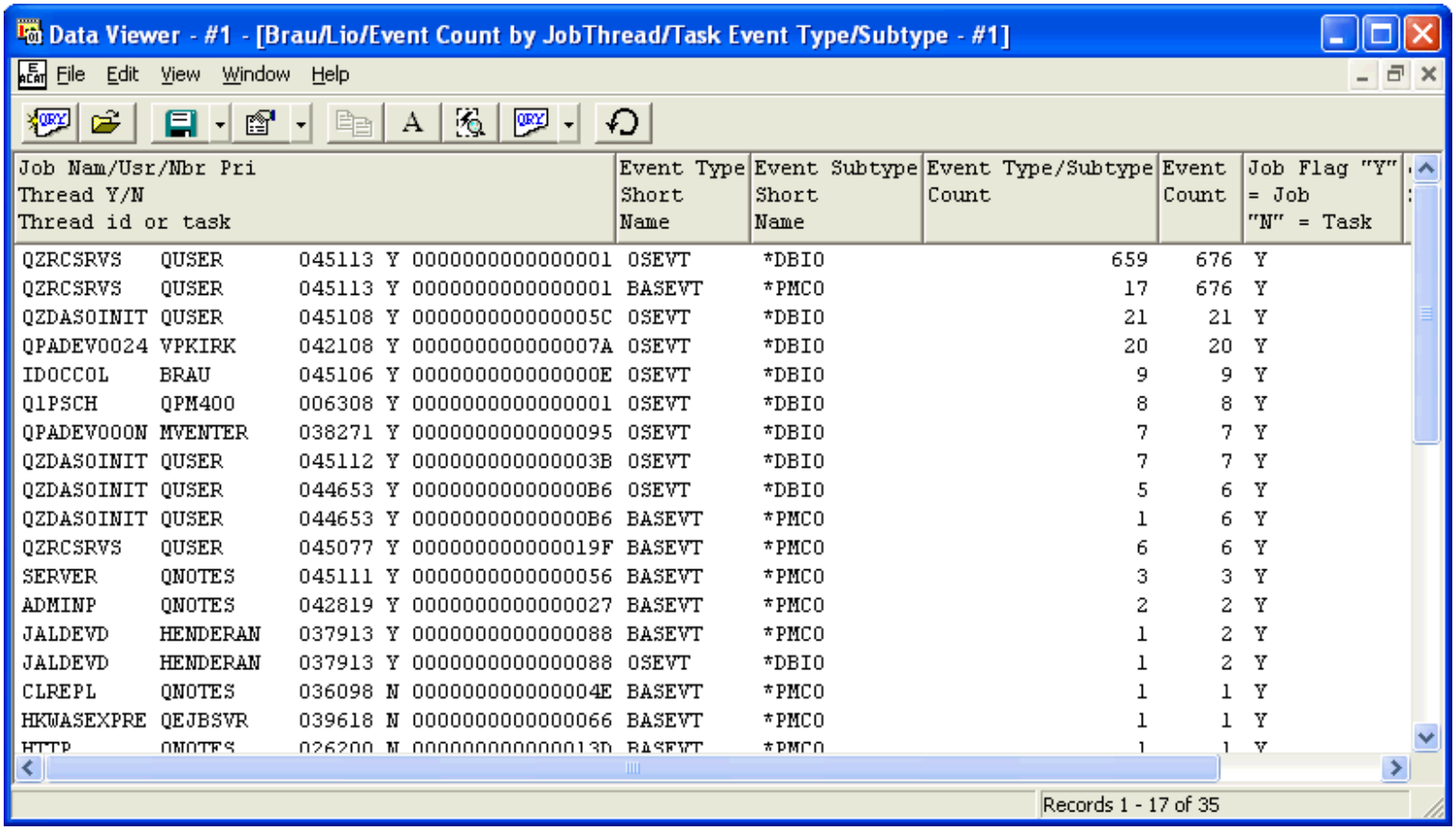


Job Nam/Usr/Nbr Pri Thread Y/N Thread id or task					JobThread-Task Event Count	Job Flag "Y" = Job "N" = Task	Joins to SMTRMOD/TDEBIG
QZRCSRVS	QUSER	045113	Y	0000000000000001	676	Y	0000000000003B5
QZDASOINIT	QUSER	045108	Y	000000000000005C	21	Y	0000000000003B5
QPADEV0024	VPKIRK	042108	Y	000000000000007A	20	Y	00000000000038B
IDOCCOL	BRAU	045106	Y	000000000000000E	9	Y	0000000000003B5
Q1PSCH	QPM400	006308	Y	0000000000000001	8	Y	00000000000003
QPADEV000N	MVENTER	038271	Y	0000000000000095	7	Y	00000000000344
QZDASOINIT	QUSER	045112	Y	000000000000003B	7	Y	0000000000003B5
QZRCSRVS	QUSER	045077	Y	000000000000019F	6	Y	0000000000003B3
QZDASOINIT	QUSER	044653	Y	00000000000000B6	6	Y	0000000000003B0
SERVER	QNOTES	045111	Y	0000000000000056	3	Y	0000000000003B5
JALDEVD	HENDERAN	037913	Y	0000000000000088	2	Y	000000000002F5
ADMINP	QNOTES	042819	Y	0000000000000027	2	Y	00000000000399
QYPSPFRCOL	QSYS	020655	N	0000000000000081	1	Y	00000000000397
HTTP	QNOTES	026200	N	000000000000013D	1	Y	00000000001F8
SCHED	QNOTES	036090	N	0000000000000093	1	Y	000000000002DB
SERVER	QNOTES	042814	N	00000000000000D2	1	Y	00000000000399
CLREPL	QNOTES	036098	N	000000000000004E	1	Y	000000000002DB
SERVER	QNOTES	026111	N	0000000000000061	1	Y	00000000001E7

Records 1 - 17 of 32

5.4.2 Event Count by JobThread/Task Event Type/Subtype

This report shows how many PEX events were captured of each type for each job in the PEX collection.



Job Nam/Usr/Nbr Pri Thread Y/N Thread id or task					Event Type Short Name	Event Subtype Short Name	Event Type/Subtype Count	Event Count	Job Flag "Y" = Job "N" = Task
QZRCSRVS	QUSER	045113	Y	0000000000000001	OSEVT	*DBIO	659	676	Y
QZRCSRVS	QUSER	045113	Y	0000000000000001	BASEVT	*PMCO	17	676	Y
QZDASOINIT	QUSER	045108	Y	000000000000005C	OSEVT	*DBIO	21	21	Y
QPADEV0024	VPKIRK	042108	Y	000000000000007A	OSEVT	*DBIO	20	20	Y
IDOCCOL	BRAU	045106	Y	000000000000000E	OSEVT	*DBIO	9	9	Y
Q1PSCH	QPM400	006308	Y	0000000000000001	OSEVT	*DBIO	8	8	Y
QPADEV000N	MVENTER	038271	Y	0000000000000095	OSEVT	*DBIO	7	7	Y
QZDASOINIT	QUSER	045112	Y	000000000000003B	OSEVT	*DBIO	7	7	Y
QZDASOINIT	QUSER	044653	Y	00000000000000B6	OSEVT	*DBIO	5	6	Y
QZDASOINIT	QUSER	044653	Y	00000000000000B6	BASEVT	*PMCO	1	6	Y
QZRCSRVS	QUSER	045077	Y	000000000000019F	BASEVT	*PMCO	6	6	Y
SERVER	QNOTES	045111	Y	0000000000000056	BASEVT	*PMCO	3	3	Y
ADMINP	QNOTES	042819	Y	0000000000000027	BASEVT	*PMCO	2	2	Y
JALDEVD	HENDERAN	037913	Y	0000000000000088	BASEVT	*PMCO	1	2	Y
JALDEVD	HENDERAN	037913	Y	0000000000000088	OSEVT	*DBIO	1	2	Y
CLREPL	QNOTES	036098	N	000000000000004E	BASEVT	*PMCO	1	1	Y
HKWASEXPRE	QEJBSVR	039618	N	0000000000000066	BASEVT	*PMCO	1	1	Y
HTTP	QNOTES	026200	N	000000000000013D	BASEVT	*PMCO	1	1	Y

Table of Contents | Previous | Next

5.5 Physical disk IO reports

This analysis shows every physical disk IO operation over the course of the collection. The issuing TDE, time of day, object name & type, dasd unit and more are shown in this analysis.

If the MI Entry/Exit events were collected, the jobs program stack is visible by right-clicking on any output file containing the STACK field (with a value > 0) and choosing the 'Display Call Stack' menu.

Table of Contents | Previous | Next

5.5.1 IO Total

This report shows the total number of physical disk IO events that occurred over the data analyzed.

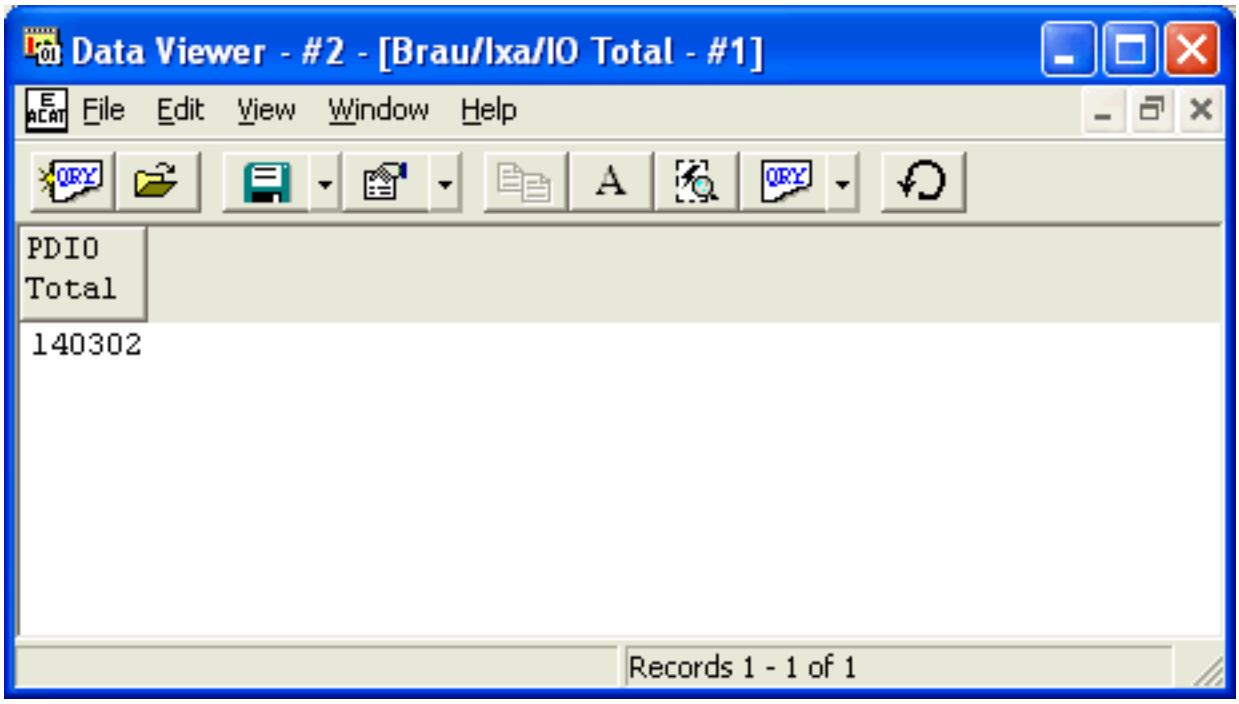
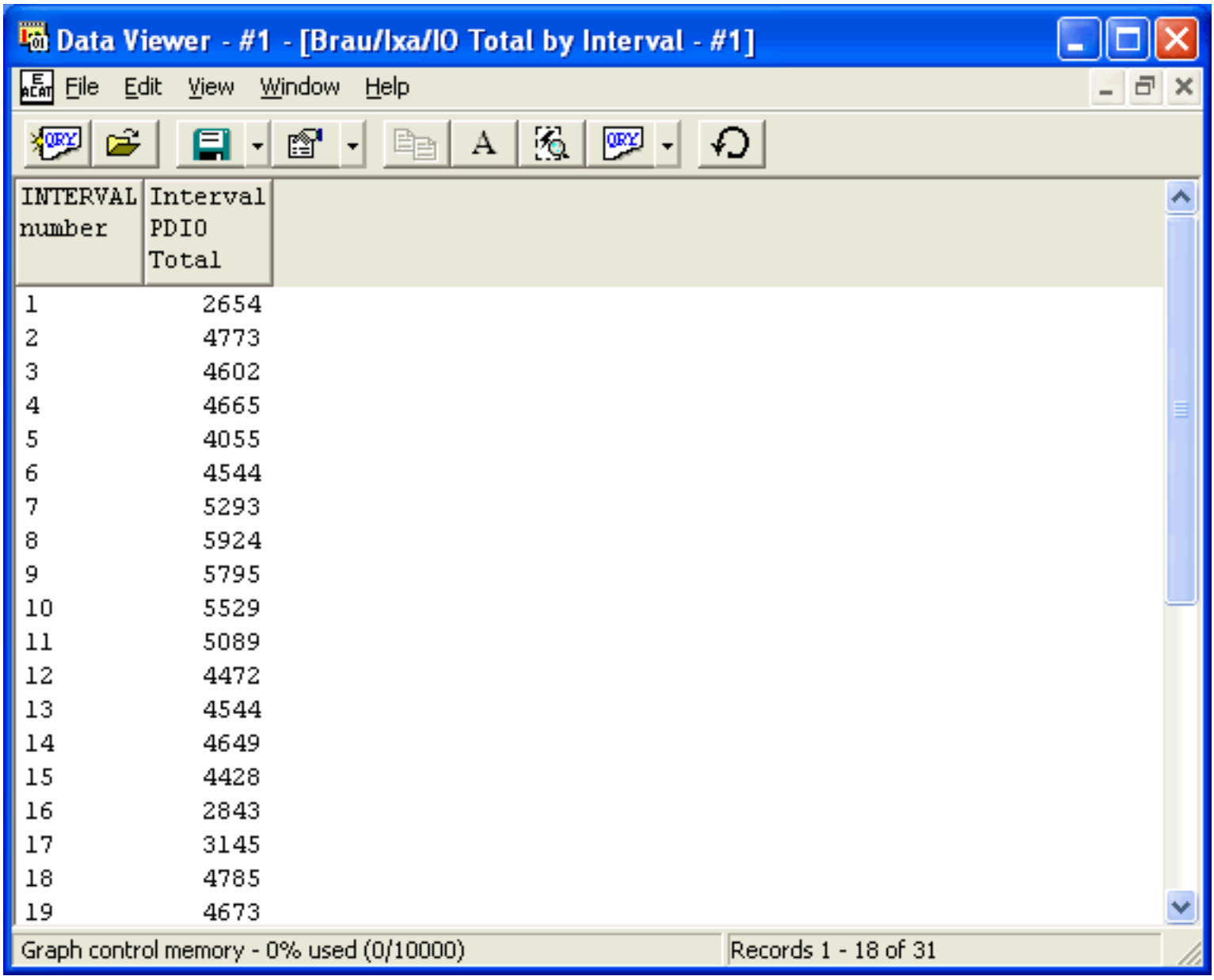


Table of Contents | Previous | Next

5.5.2 IO Total by Interval

This report shows the total number of physical disk IO events that occurred each interval over the data analyzed.

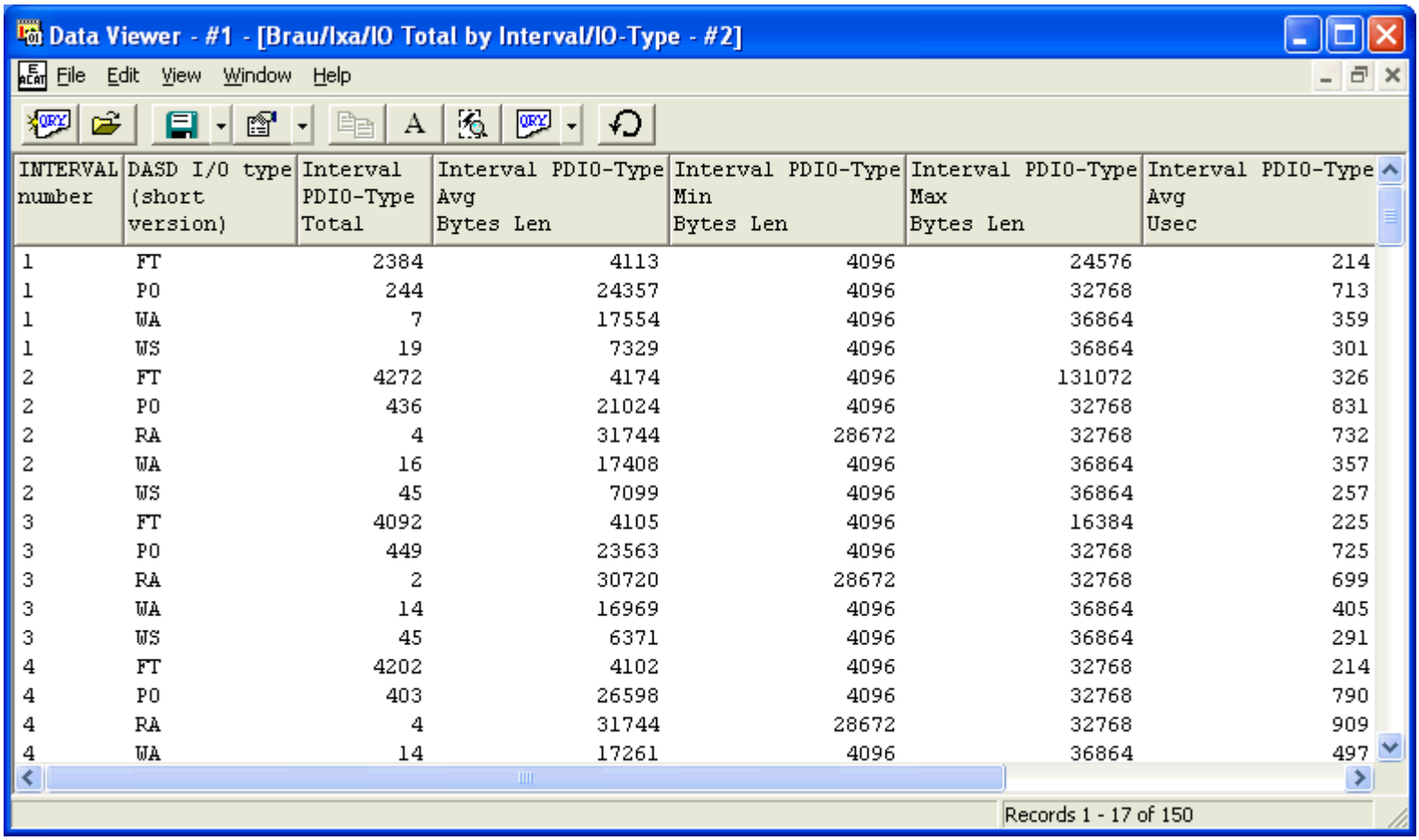


INTERVAL number	Interval PDIO Total
1	2654
2	4773
3	4602
4	4665
5	4055
6	4544
7	5293
8	5924
9	5795
10	5529
11	5089
12	4472
13	4544
14	4649
15	4428
16	2843
17	3145
18	4785
19	4673

Table of Contents | Previous | Next

5.5.3 IO Total by Interval/IO-Type

This report shows a breakdown of the physical disk IO events that occurred over the data analyzed by IO type for each interval.



INTERVAL number	DASD I/O type (short version)	Interval PDIO-Type Total	Interval PDIO-Type Avg Bytes Len	Interval PDIO-Type Min Bytes Len	Interval PDIO-Type Max Bytes Len	Interval PDIO-Type Avg Usec
1	FT	2384	4113	4096	24576	214
1	PO	244	24357	4096	32768	713
1	WA	7	17554	4096	36864	359
1	WS	19	7329	4096	36864	301
2	FT	4272	4174	4096	131072	326
2	PO	436	21024	4096	32768	831
2	RA	4	31744	28672	32768	732
2	WA	16	17408	4096	36864	357
2	WS	45	7099	4096	36864	257
3	FT	4092	4105	4096	16384	225
3	PO	449	23563	4096	32768	725
3	RA	2	30720	28672	32768	699
3	WA	14	16969	4096	36864	405
3	WS	45	6371	4096	36864	291
4	FT	4202	4102	4096	32768	214
4	PO	403	26598	4096	32768	790
4	RA	4	31744	28672	32768	909
4	WA	14	17261	4096	36864	497

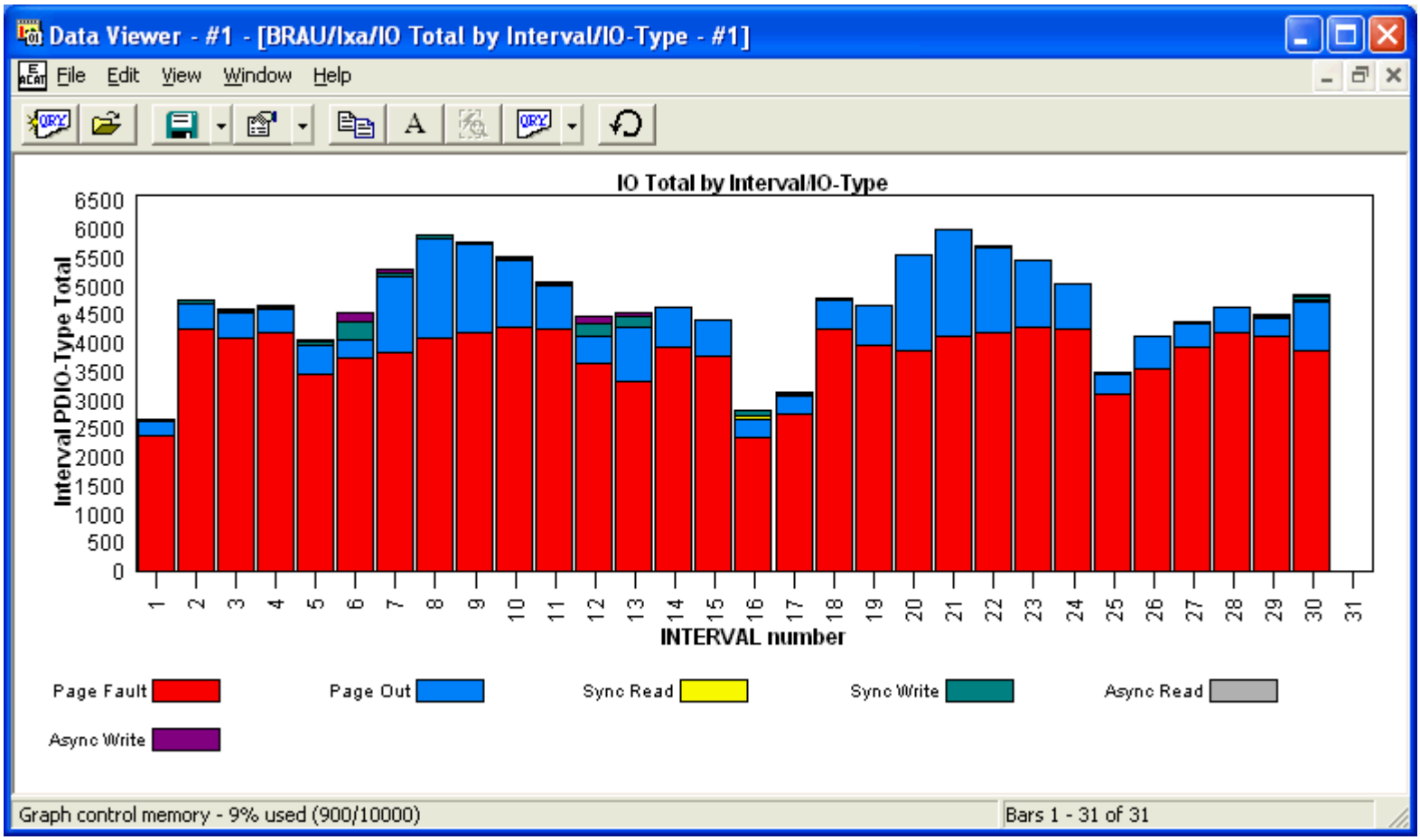
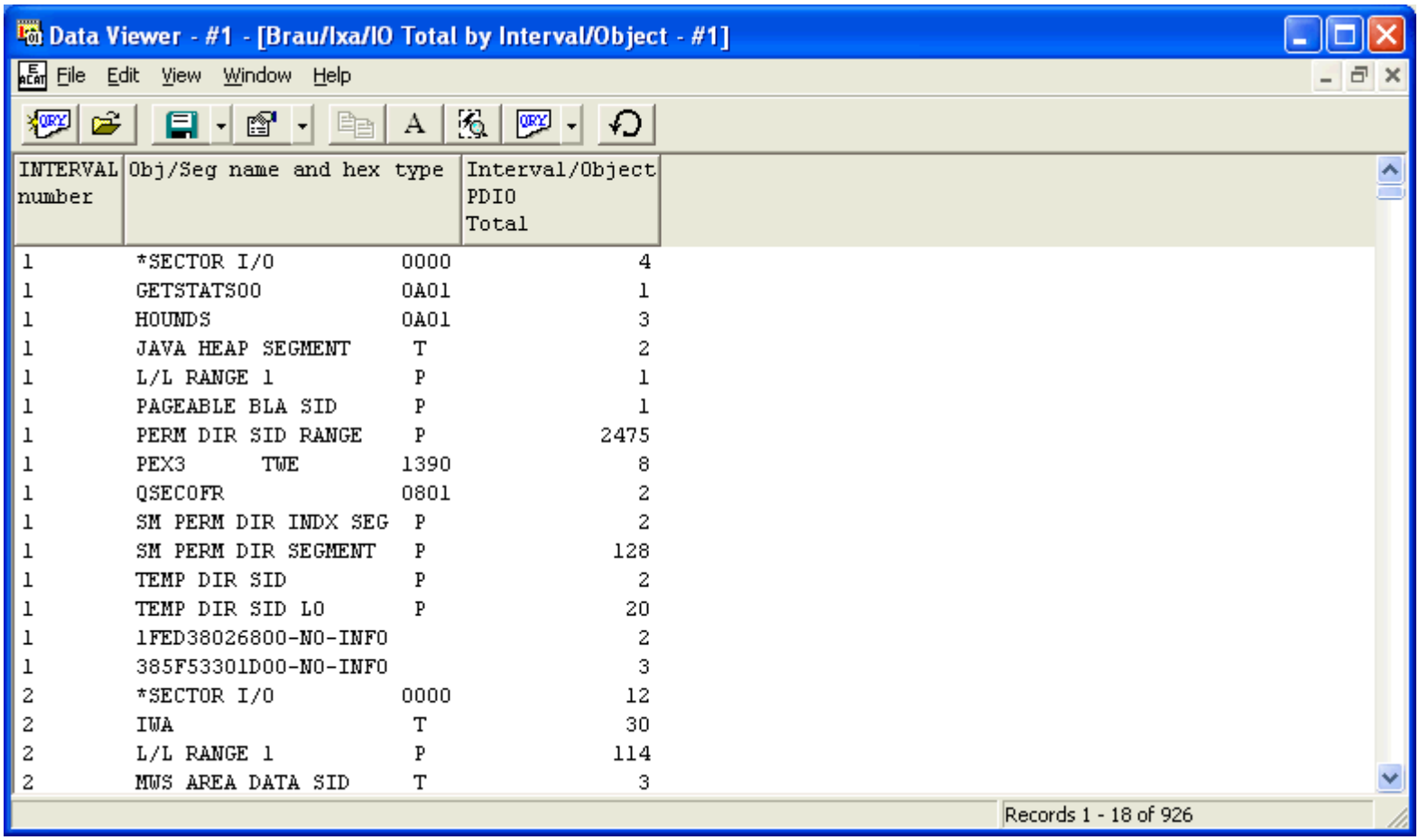

Data Viewer - #1 - [BRAU/Ixa/IO Total by Interval/IO-Type - #1]
File Edit View Window Help
IO Total by Interval/IO-Type
Interval PDIO-Type Total
6500
6000
5500
5000
4500
4000
3500
3000
2500
2000
1500
1000
500
0
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
INTERVAL number
Page Fault
Page Out
Sync Read
Sync Write
Async Read
Async Write
Graph control memory - 9% used (900/10000)
Bars 1 - 31 of 31

Table of Contents | Previous | Next

5.5.4 IO Total by Interval/Object

This report shows a breakdown of the physical disk IO events that occurred over the data analyzed for each object within each interval.



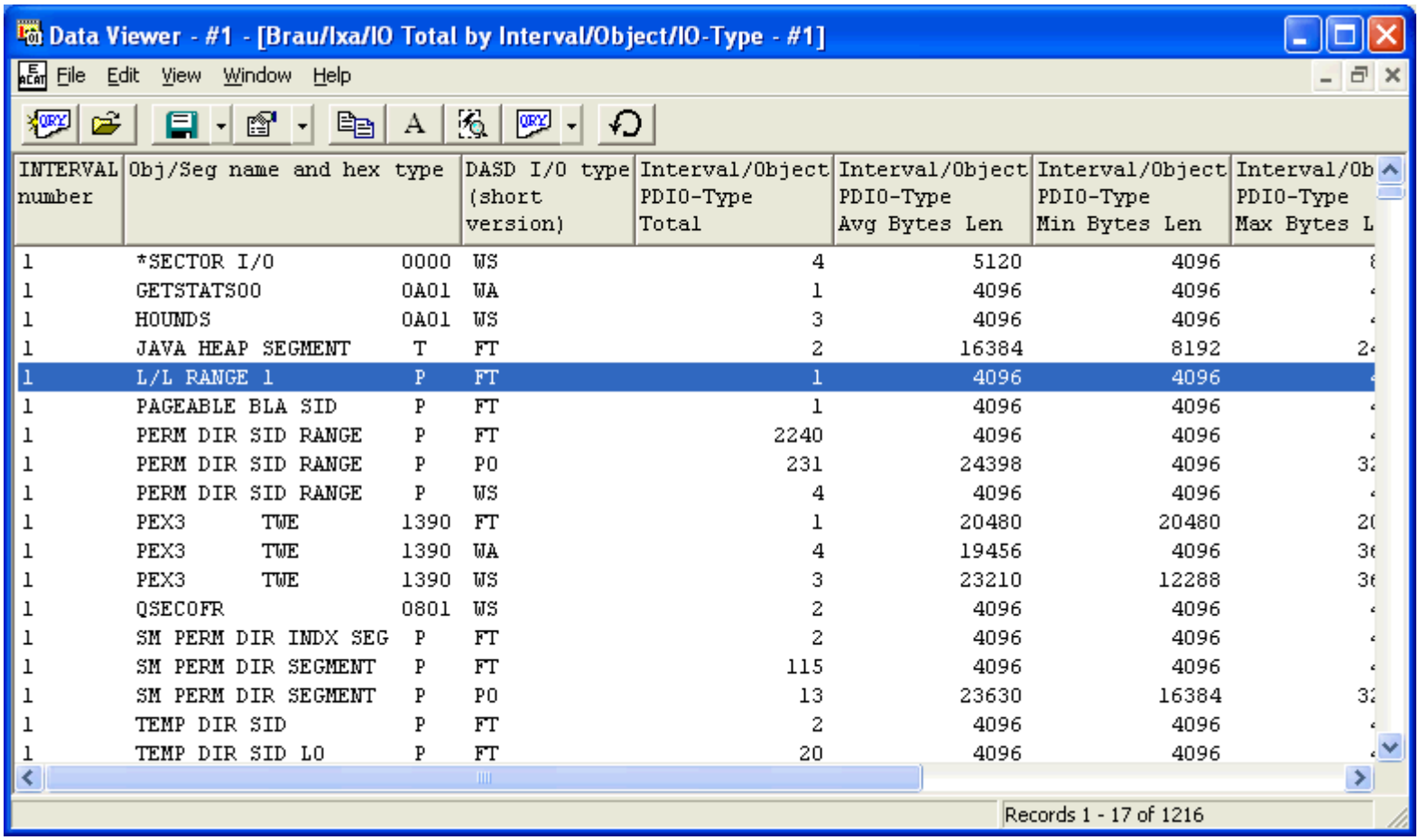
INTERVAL number	Obj/Seg name and hex type		Interval/Object PDIO Total
1	*SECTOR I/O	0000	4
1	GETSTATS00	0A01	1
1	HOUNDS	0A01	3
1	JAVA HEAP SEGMENT	T	2
1	L/L RANGE 1	P	1
1	PAGEABLE BLA SID	P	1
1	PERM DIR SID RANGE	P	2475
1	PEX3 TWE	1390	8
1	QSECOFR	0801	2
1	SM PERM DIR INDX SEG	P	2
1	SM PERM DIR SEGMENT	P	128
1	TEMP DIR SID	P	2
1	TEMP DIR SID L0	P	20
1	1FED38026800-N0-INF0		2
1	385F53301D00-N0-INF0		3
2	*SECTOR I/O	0000	12
2	IWA	T	30
2	L/L RANGE 1	P	114
2	MWS AREA DATA SID	T	3

Records 1 - 18 of 926

Table of Contents | Previous | Next

5.5.5 IO Total by Interval/Object/IO-Type

This report shows a breakdown of the PDIO events that occurred over the data analyzed for each interval, object, and type combination.



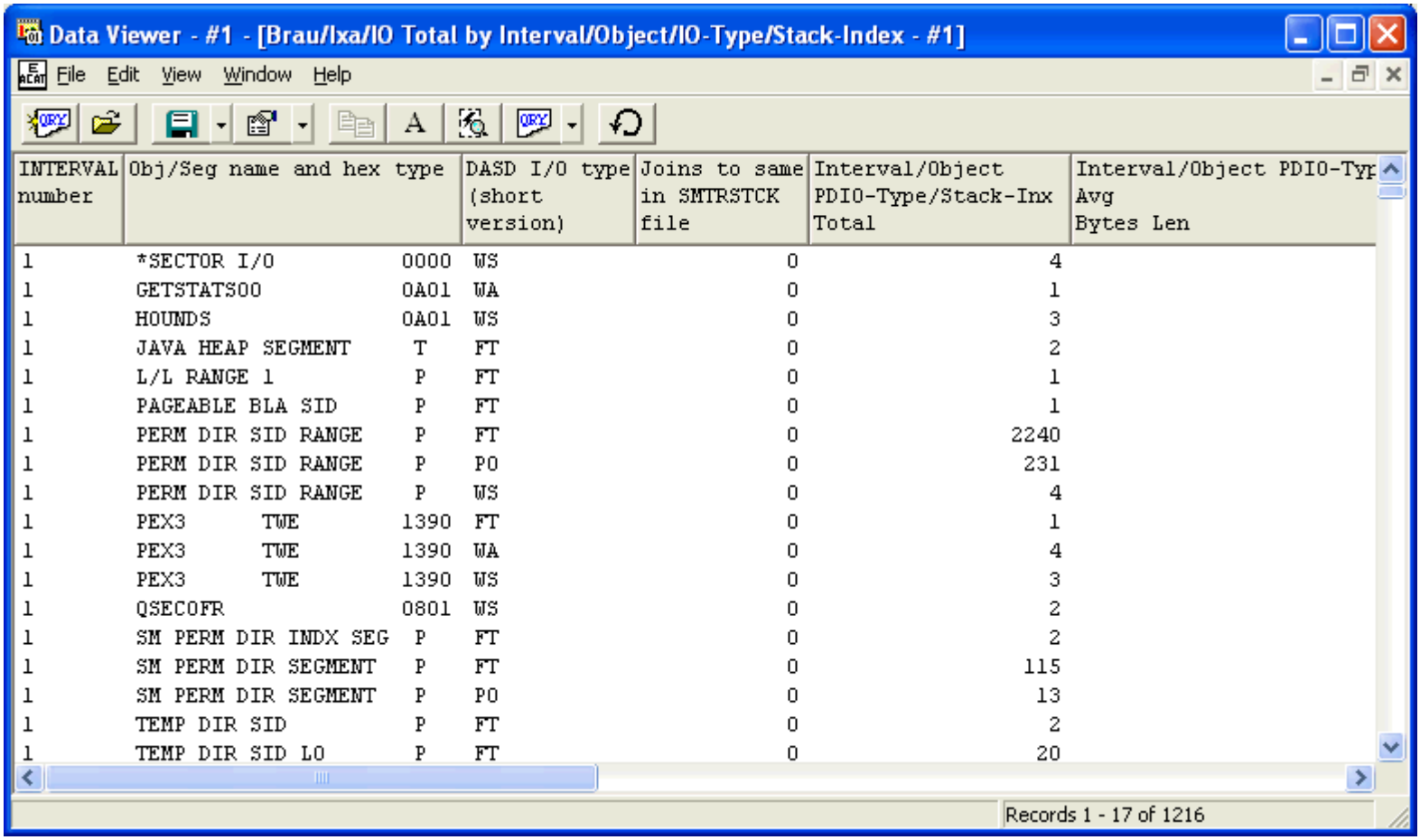
INTERVAL number	Obj/Seg name and hex type		DASD I/O type (short version)	Interval/Object PDIO-Type Total	Interval/Object PDIO-Type Avg Bytes Len	Interval/Object PDIO-Type Min Bytes Len	Interval/Ob PDIO-Type Max Bytes L
1	*SECTOR I/O	0000	WS	4	5120	4096	8
1	GETSTATS00	0A01	WA	1	4096	4096	4
1	HOUNDS	0A01	WS	3	4096	4096	4
1	JAVA HEAP SEGMENT	T	FT	2	16384	8192	2
1	L/L RANGE 1	P	FT	1	4096	4096	4
1	PAGEABLE BLA SID	P	FT	1	4096	4096	4
1	PERM DIR SID RANGE	P	FT	2240	4096	4096	4
1	PERM DIR SID RANGE	P	PO	231	24398	4096	32
1	PERM DIR SID RANGE	P	WS	4	4096	4096	4
1	PEX3 TWE	1390	FT	1	20480	20480	20
1	PEX3 TWE	1390	WA	4	19456	4096	36
1	PEX3 TWE	1390	WS	3	23210	12288	36
1	QSECOFR	0801	WS	2	4096	4096	4
1	SM PERM DIR INDX SEG	P	FT	2	4096	4096	4
1	SM PERM DIR SEGMENT	P	FT	115	4096	4096	4
1	SM PERM DIR SEGMENT	P	PO	13	23630	16384	32
1	TEMP DIR SID	P	FT	2	4096	4096	4
1	TEMP DIR SID LO	P	FT	20	4096	4096	4

Table of Contents | Previous | Next

5.5.6 IO Total by Interval/Object/IO-Type/Stack-Index

This report shows a breakdown of the physical disk IO events that occurred over the data analyzed for each interval, object, type and stack index combination.

If the stack index is non zero, a call stack will appear in the properties found by double-clicking on the desired record.

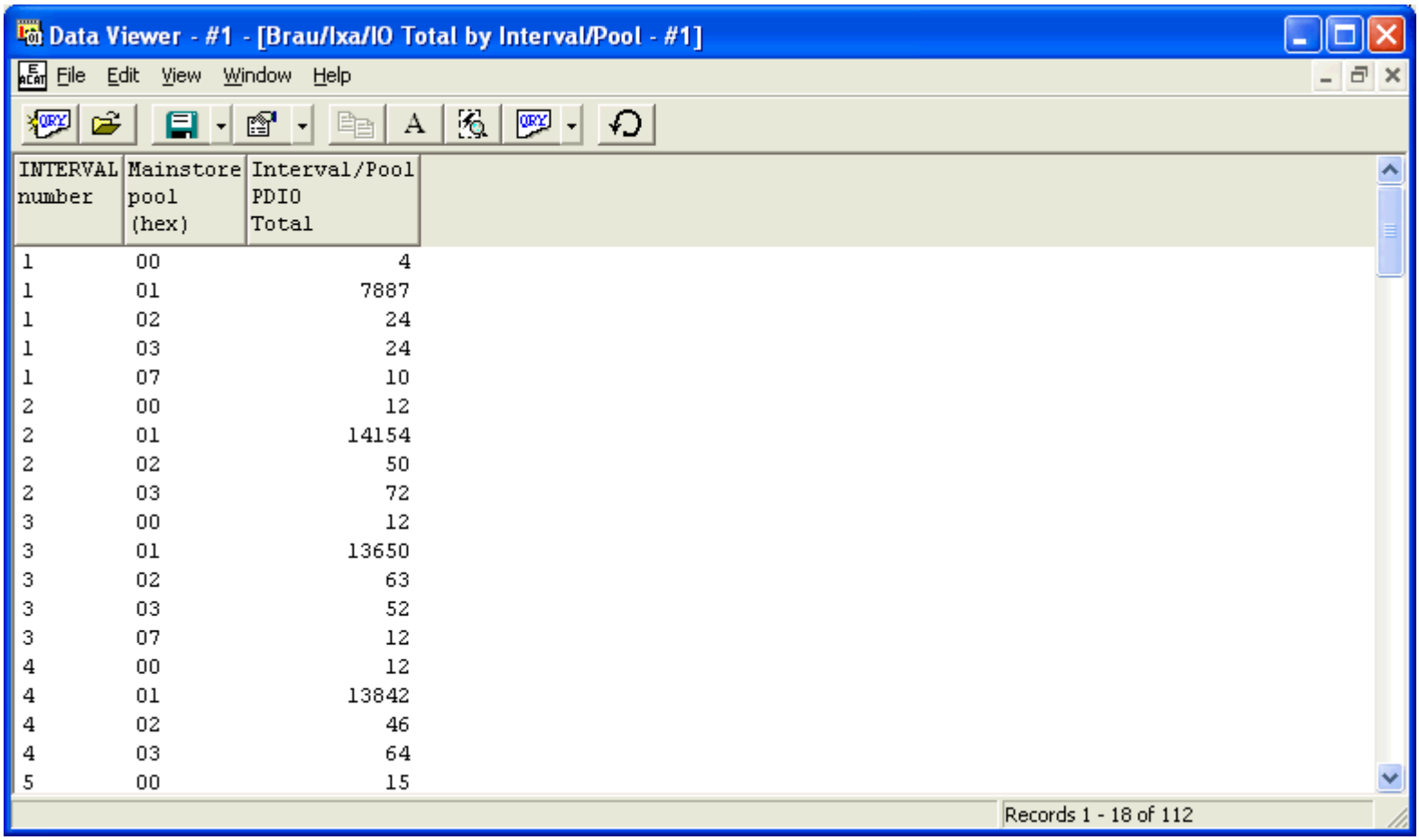


INTERVAL number	Obj/Seg name and hex type	DASD I/O type (short version)	Joins to same in SMTRSTCK file	Interval/Object PDIO-Type/Stack-Inx Total	Interval/Object PDIO-Typ Avg Bytes Len
1	*SECTOR I/O 0000	WS	0	4	
1	GETSTATS00 0A01	WA	0	1	
1	HOUNDS 0A01	WS	0	3	
1	JAVA HEAP SEGMENT T	FT	0	2	
1	L/L RANGE 1 P	FT	0	1	
1	PAGEABLE BLA SID P	FT	0	1	
1	PERM DIR SID RANGE P	FT	0	2240	
1	PERM DIR SID RANGE P	PO	0	231	
1	PERM DIR SID RANGE P	WS	0	4	
1	PEX3 TWE 1390	FT	0	1	
1	PEX3 TWE 1390	WA	0	4	
1	PEX3 TWE 1390	WS	0	3	
1	QSECOFR 0801	WS	0	2	
1	SM PERM DIR INDX SEG P	FT	0	2	
1	SM PERM DIR SEGMENT P	FT	0	115	
1	SM PERM DIR SEGMENT P	PO	0	13	
1	TEMP DIR SID P	FT	0	2	
1	TEMP DIR SID LO P	FT	0	20	

Table of Contents | Previous | Next

5.5.7 IO Total by Interval/Pool

This report shows the total number of physical disk IO events that occurred over the data analyzed for each memory pool within each interval.



INTERVAL number	Mainstore pool (hex)	Interval/Pool PDIO Total
1	00	4
1	01	7887
1	02	24
1	03	24
1	07	10
2	00	12
2	01	14154
2	02	50
2	03	72
3	00	12
3	01	13650
3	02	63
3	03	52
3	07	12
4	00	12
4	01	13842
4	02	46
4	03	64
5	00	15

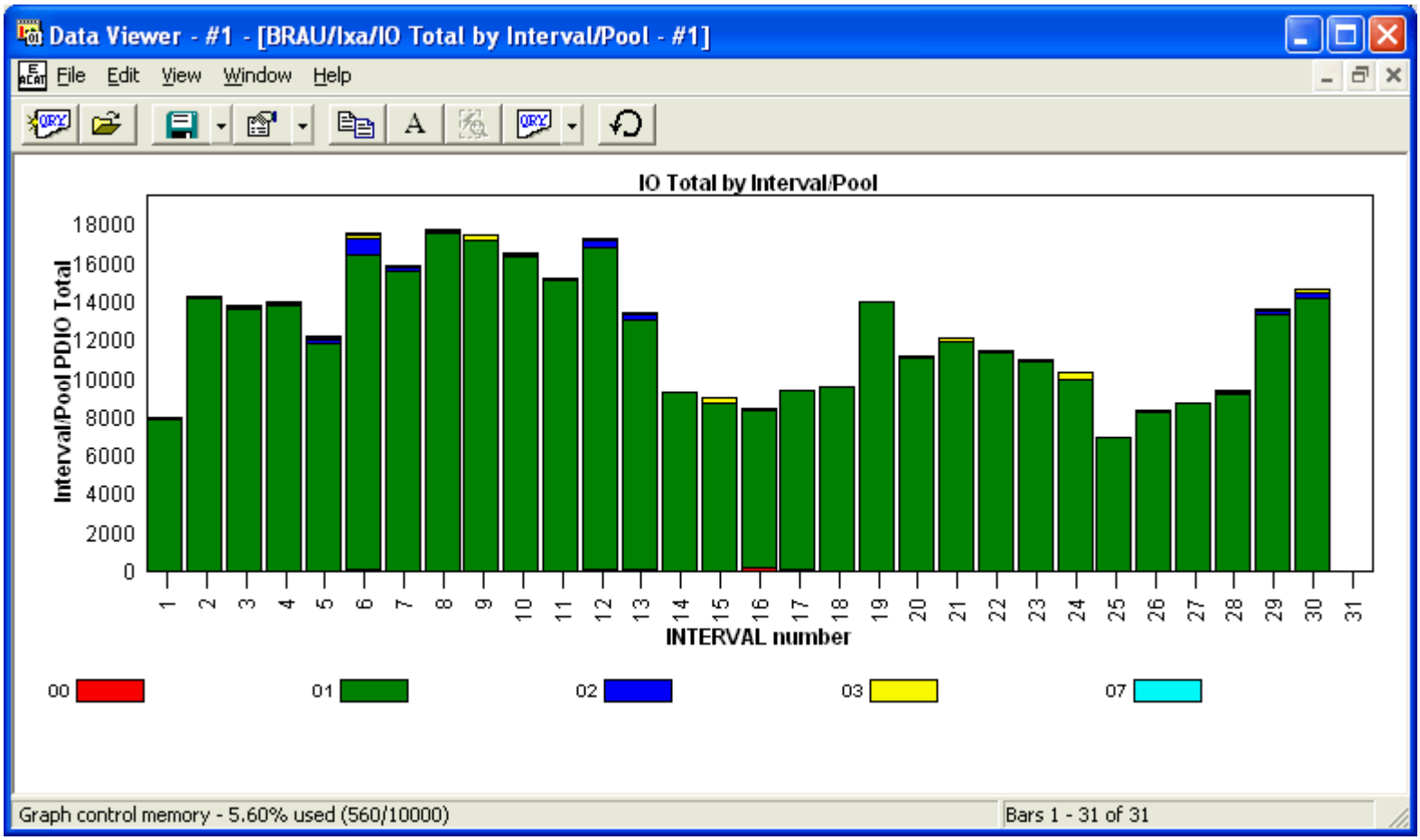
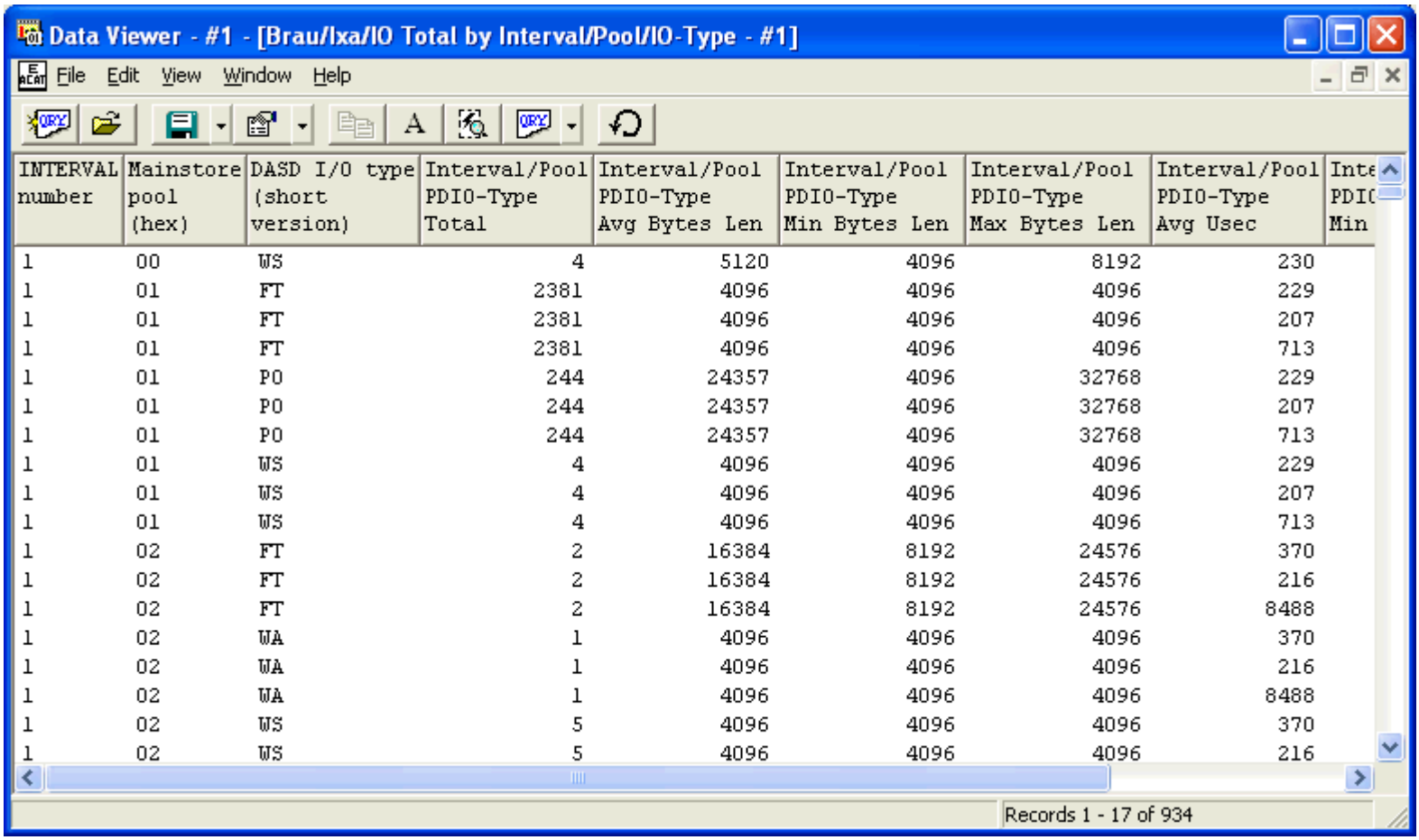

Data Viewer - #1 - [BRAU/Ixa/IO Total by Interval/Pool - #1]
File Edit View Window Help
IO Total by Interval/Pool
Interval/Pool PDIO Total
18000
16000
14000
12000
10000
8000
6000
4000
2000
0
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
INTERVAL number
00
01
02
03
07
Graph control memory - 5.60% used (560/10000)
Bars 1 - 31 of 31

Table of Contents | Previous | Next

5.5.8 IO Total by Interval/Pool/IO-Type

This report shows a breakdown of the physical disk IO events that occurred over the data analyzed for each memory pool, io type, and interval combination.

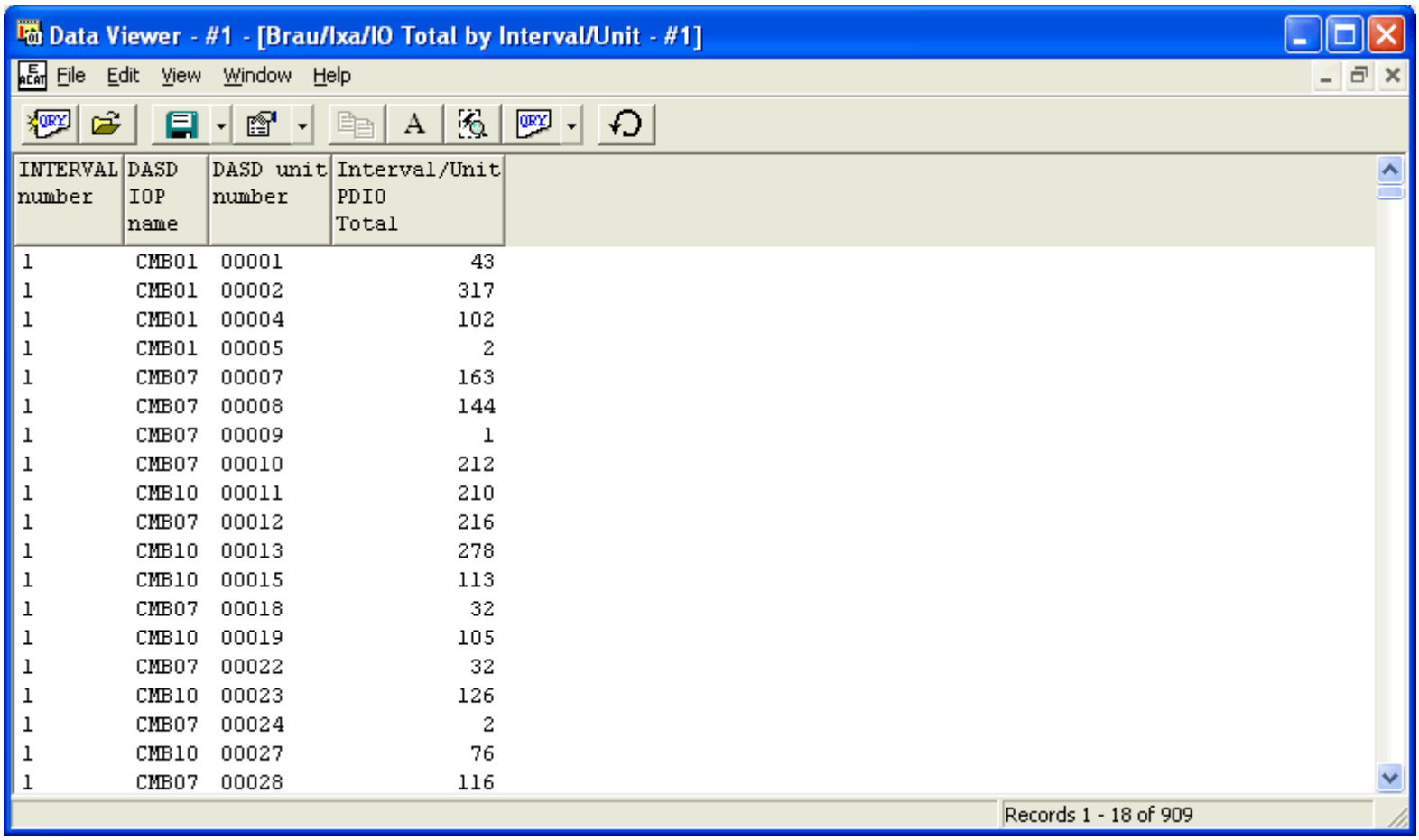


INTERVAL number	Mainstore pool (hex)	DASD I/O type (short version)	Interval/Pool PDIO-Type Total	Interval/Pool PDIO-Type Avg Bytes Len	Interval/Pool PDIO-Type Min Bytes Len	Interval/Pool PDIO-Type Max Bytes Len	Interval/Pool PDIO-Type Avg Usec	Inte PDIO Min
1	00	WS	4	5120	4096	8192	230	
1	01	FT	2381	4096	4096	4096	229	
1	01	FT	2381	4096	4096	4096	207	
1	01	FT	2381	4096	4096	4096	713	
1	01	PO	244	24357	4096	32768	229	
1	01	PO	244	24357	4096	32768	207	
1	01	PO	244	24357	4096	32768	713	
1	01	WS	4	4096	4096	4096	229	
1	01	WS	4	4096	4096	4096	207	
1	01	WS	4	4096	4096	4096	713	
1	02	FT	2	16384	8192	24576	370	
1	02	FT	2	16384	8192	24576	216	
1	02	FT	2	16384	8192	24576	8488	
1	02	WA	1	4096	4096	4096	370	
1	02	WA	1	4096	4096	4096	216	
1	02	WA	1	4096	4096	4096	8488	
1	02	WS	5	4096	4096	4096	370	
1	02	WS	5	4096	4096	4096	216	

Table of Contents | Previous | Next

5.5.9 IO Total by Interval/Unit

This report shows the total number of physical disk IO events that occurred over the data analyzed for each disk unit during each interval.

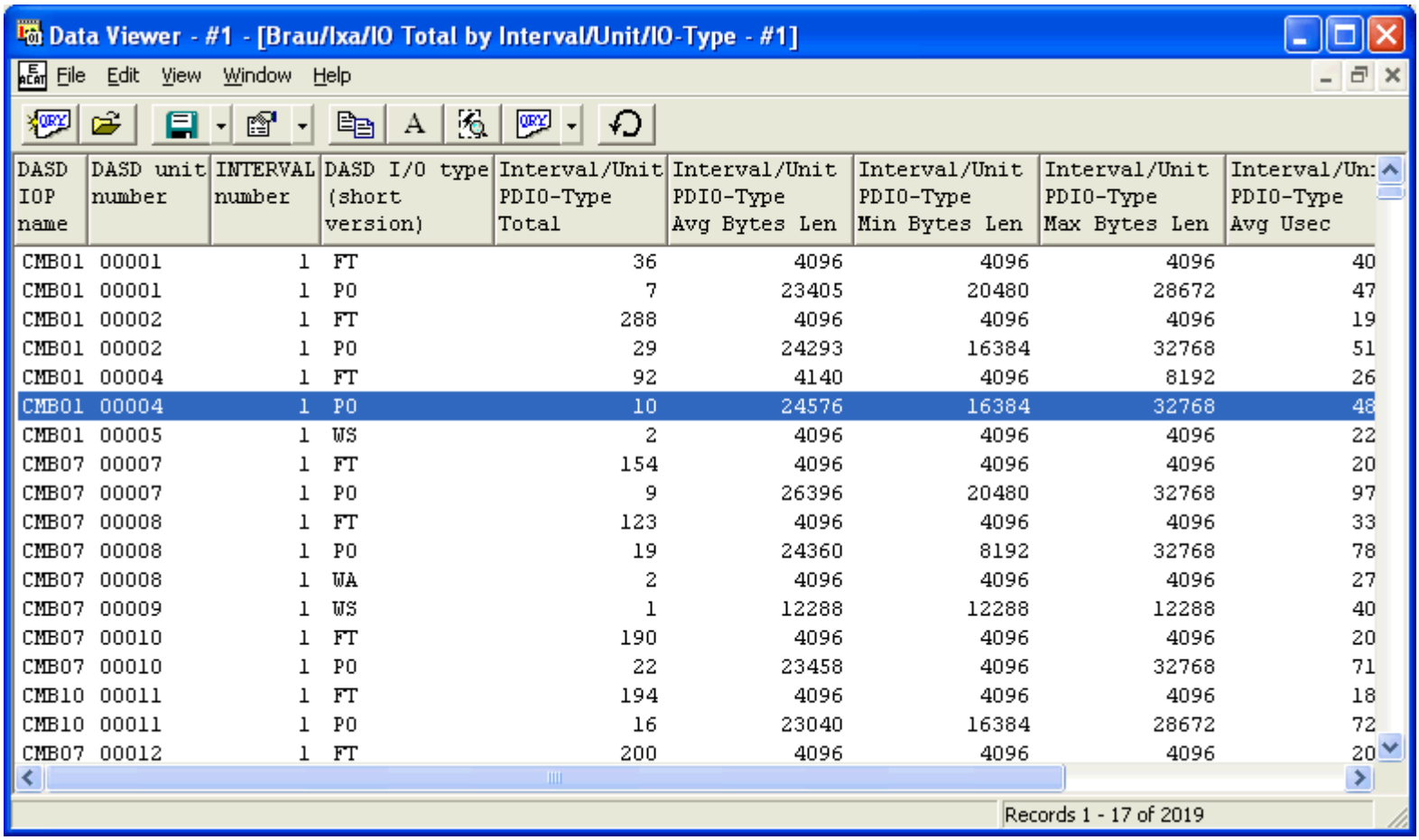


INTERVAL number	DASD IOP name	DASD unit number	Interval/Unit PDIO Total
1	CMB01	00001	43
1	CMB01	00002	317
1	CMB01	00004	102
1	CMB01	00005	2
1	CMB07	00007	163
1	CMB07	00008	144
1	CMB07	00009	1
1	CMB07	00010	212
1	CMB10	00011	210
1	CMB07	00012	216
1	CMB10	00013	278
1	CMB10	00015	113
1	CMB07	00018	32
1	CMB10	00019	105
1	CMB07	00022	32
1	CMB10	00023	126
1	CMB07	00024	2
1	CMB10	00027	76
1	CMB07	00028	116

Table of Contents | Previous | Next

5.5.10 IO Total by Interval/Unit/IO-Type

This report shows a breakdown of the physical disk IO events that occurred over the data analyzed for each disk unit, io type and interval combination.

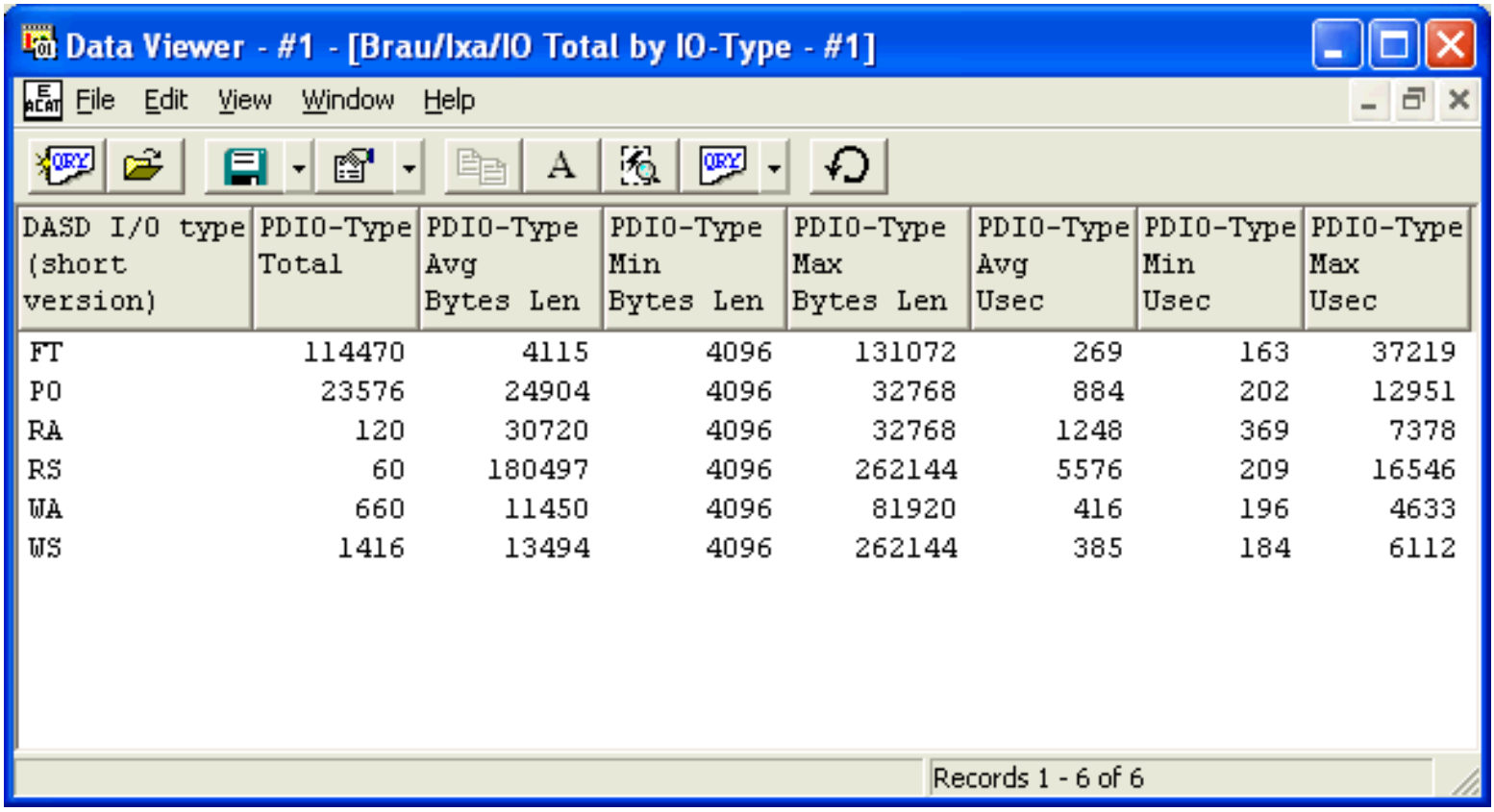


DASD IOP name	DASD unit number	INTERVAL number	DASD I/O type (short version)	Interval/Unit PDIO-Type Total	Interval/Unit PDIO-Type Avg Bytes Len	Interval/Unit PDIO-Type Min Bytes Len	Interval/Unit PDIO-Type Max Bytes Len	Interval/Un: PDIO-Type Avg Usec
CMB01	00001	1	FT	36	4096	4096	4096	40
CMB01	00001	1	PO	7	23405	20480	28672	47
CMB01	00002	1	FT	288	4096	4096	4096	19
CMB01	00002	1	PO	29	24293	16384	32768	51
CMB01	00004	1	FT	92	4140	4096	8192	26
CMB01	00004	1	PO	10	24576	16384	32768	48
CMB01	00005	1	WS	2	4096	4096	4096	22
CMB07	00007	1	FT	154	4096	4096	4096	20
CMB07	00007	1	PO	9	26396	20480	32768	97
CMB07	00008	1	FT	123	4096	4096	4096	33
CMB07	00008	1	PO	19	24360	8192	32768	78
CMB07	00008	1	WA	2	4096	4096	4096	27
CMB07	00009	1	WS	1	12288	12288	12288	40
CMB07	00010	1	FT	190	4096	4096	4096	20
CMB07	00010	1	PO	22	23458	4096	32768	71
CMB10	00011	1	FT	194	4096	4096	4096	18
CMB10	00011	1	PO	16	23040	16384	28672	72
CMB07	00012	1	FT	200	4096	4096	4096	20

Table of Contents | Previous | Next

5.5.11 IO Total by IO-Type

This report shows the total IOs that occurred in the collection of each type as well as the average, min, and max times and sizes.



DASD I/O type (short version)	PDIO-Type Total	PDIO-Type Avg Bytes Len	PDIO-Type Min Bytes Len	PDIO-Type Max Bytes Len	PDIO-Type Avg Usec	PDIO-Type Min Usec	PDIO-Type Max Usec
FT	114470	4115	4096	131072	269	163	37219
PO	23576	24904	4096	32768	884	202	12951
RA	120	30720	4096	32768	1248	369	7378
RS	60	180497	4096	262144	5576	209	16546
WA	660	11450	4096	81920	416	196	4633
WS	1416	13494	4096	262144	385	184	6112

Table of Contents | Previous | Next

5.5.12 IO Total by IOP

This report shows the total number of physical disk IO events that occurred for each DASD IOP.

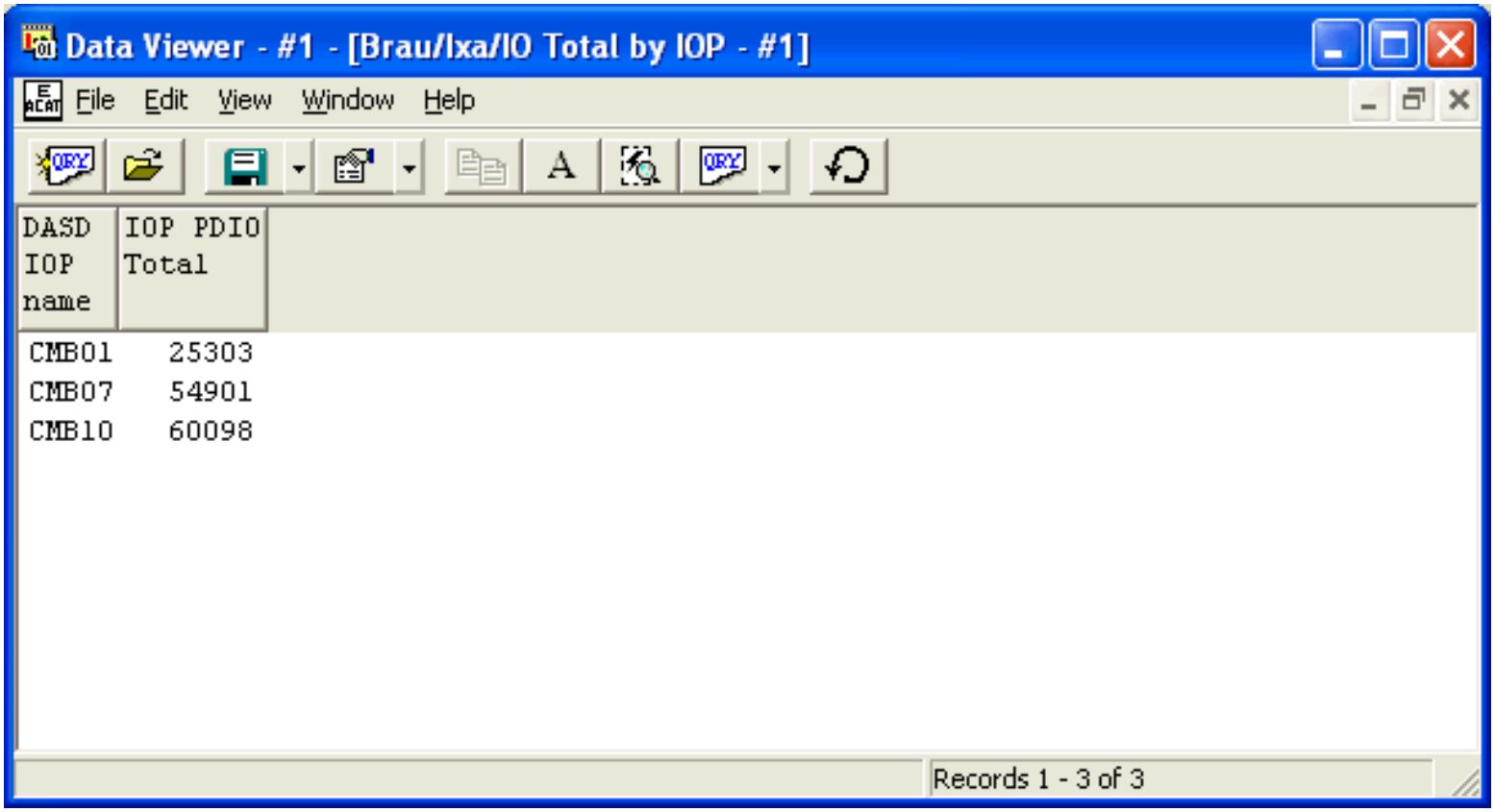
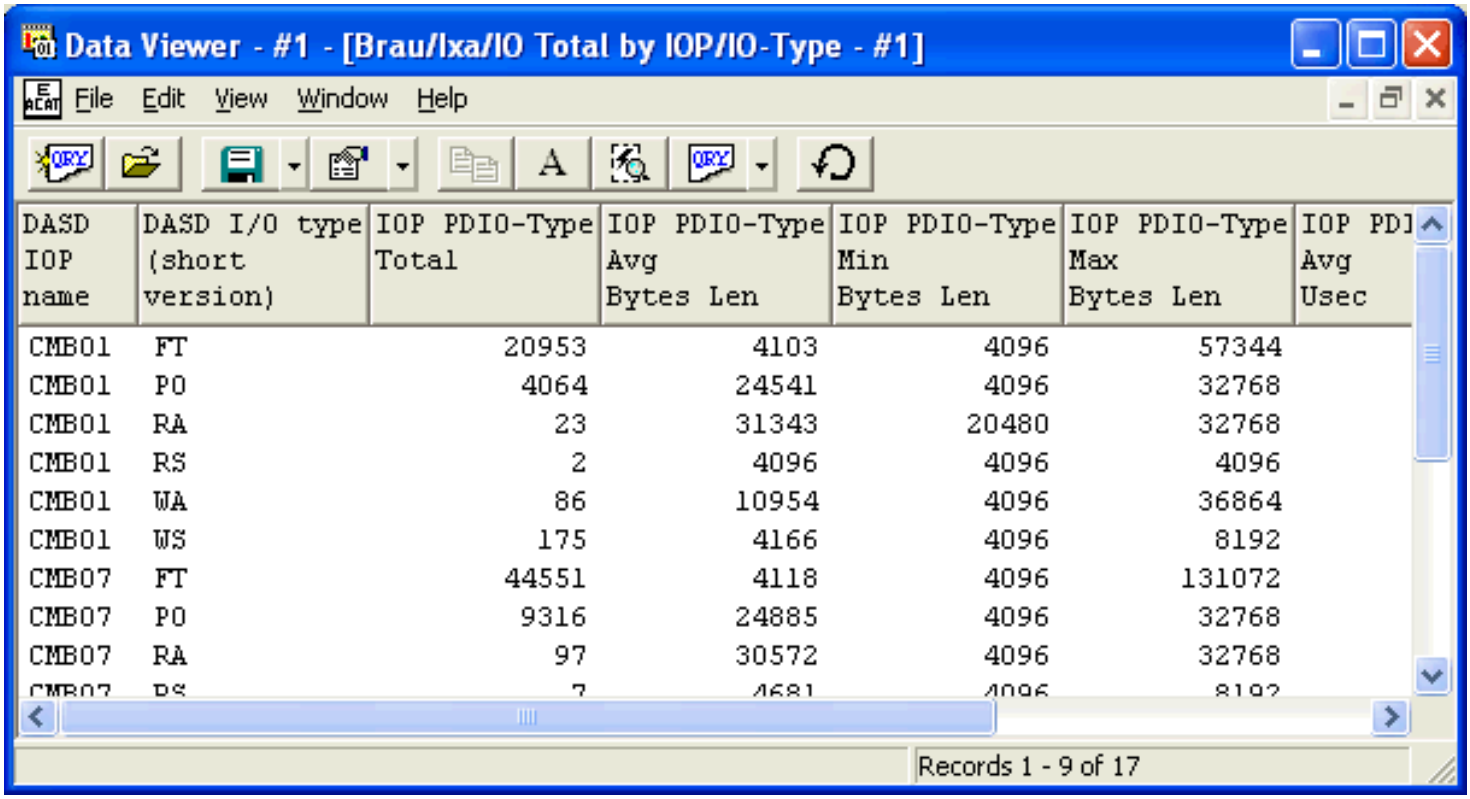


Table of Contents | Previous | Next

5.5.13 IO Total by IOP/IO-Type

This report shows a breakdown of the physical disk IO events that occurred for each DASD IOP by IO type.



DASD IOP name	DASD I/O type (short version)	IOP PDIO-Type Total	IOP PDIO-Type Avg Bytes Len	IOP PDIO-Type Min Bytes Len	IOP PDIO-Type Max Bytes Len	IOP PDI Avg Usec
CMB01	FT	20953	4103	4096	57344	
CMB01	PO	4064	24541	4096	32768	
CMB01	RA	23	31343	20480	32768	
CMB01	RS	2	4096	4096	4096	
CMB01	WA	86	10954	4096	36864	
CMB01	WS	175	4166	4096	8192	
CMB07	FT	44551	4118	4096	131072	
CMB07	PO	9316	24885	4096	32768	
CMB07	RA	97	30572	4096	32768	
CMB07	RS	7	4681	4096	8192	

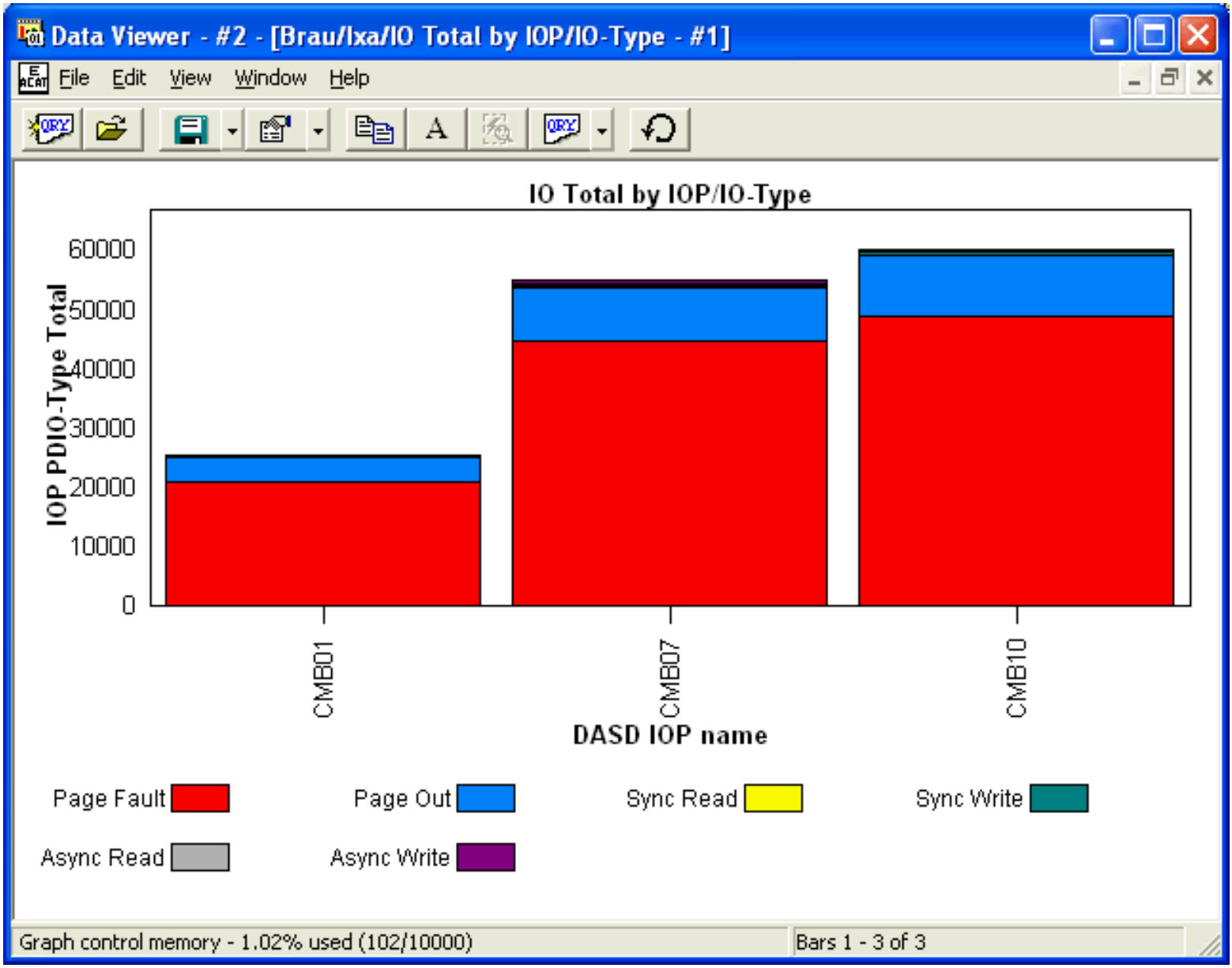
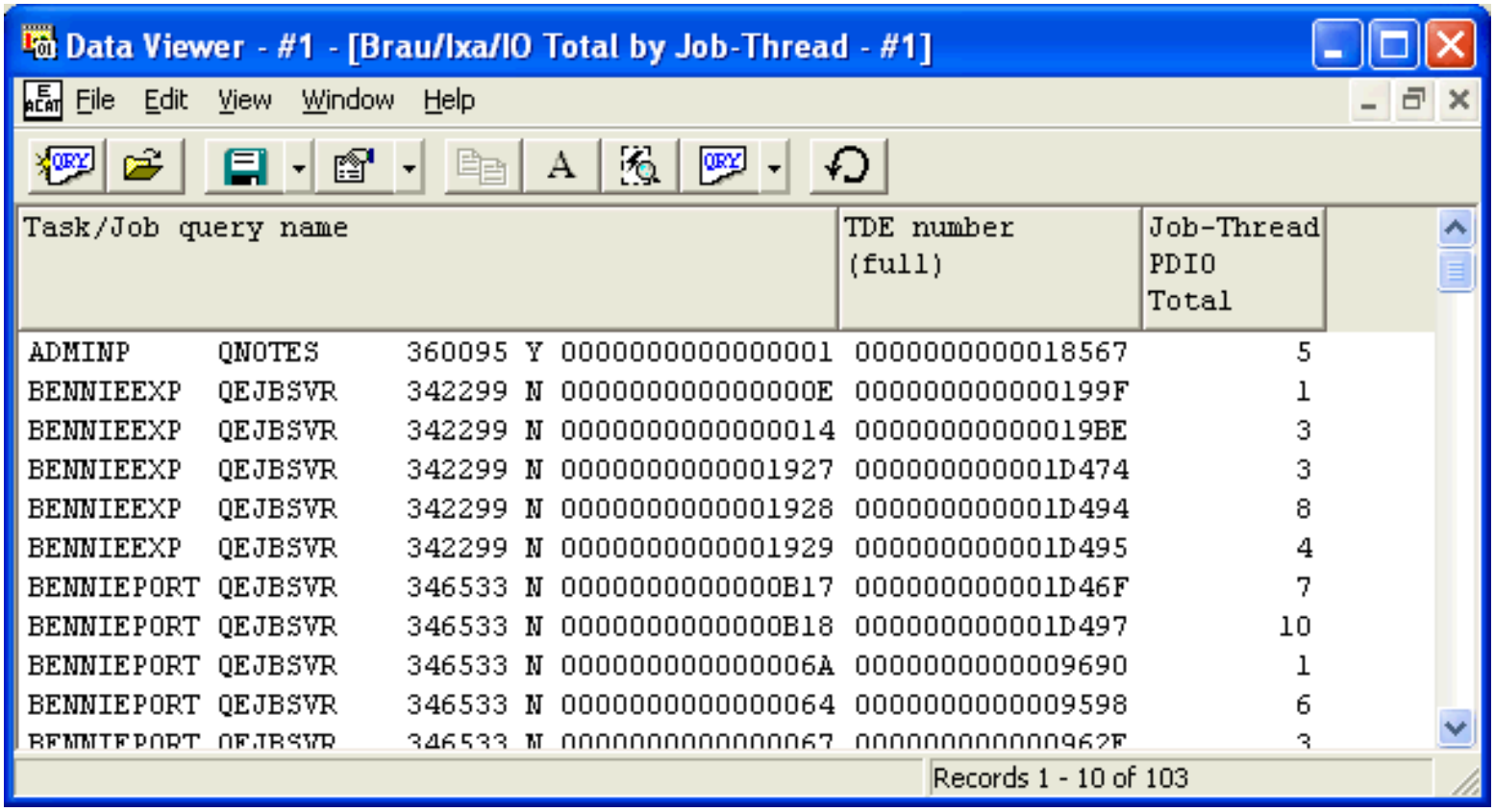

Data Viewer - #2 - [Brau/Ixa/IO Total by IOP/IO-Type - #1]
File Edit View Window Help
IO Total by IOP/IO-Type
IOP PDIO-Type Total
60000
50000
40000
30000
20000
10000
0
CMB01
CMB07
CMB10
DASD IOP name
Page Fault
Page Out
Sync Read
Sync Write
Async Read
Async Write
Graph control memory - 1.02% used (102/10000)
Bars 1 - 3 of 3

Table of Contents | Previous | Next

5.5.14 IO Total by Job-Thread

This report shows the total number of physical disk IO events that occurred for each job/thread within the collection.

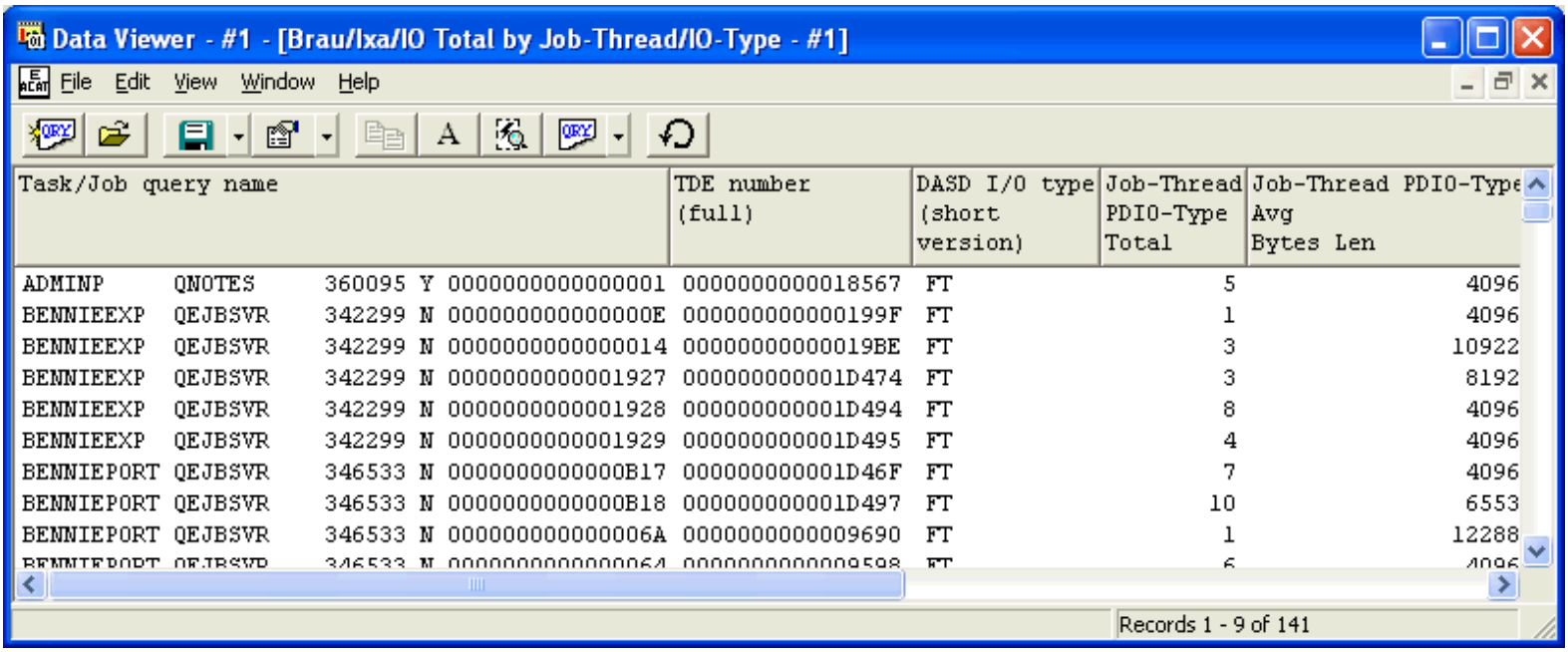


Task/Job query name				TDE number (full)	Job-Thread PDIO Total
ADMINP	QNOTES	360095	Y	0000000000000001	0000000000018567	5
BENNIEEXP	QEJBSVR	342299	N	000000000000000E	000000000000199F	1
BENNIEEXP	QEJBSVR	342299	N	0000000000000014	00000000000019BE	3
BENNIEEXP	QEJBSVR	342299	N	0000000000001927	000000000001D474	3
BENNIEEXP	QEJBSVR	342299	N	0000000000001928	000000000001D494	8
BENNIEEXP	QEJBSVR	342299	N	0000000000001929	000000000001D495	4
BENNIEPORT	QEJBSVR	346533	N	0000000000000B17	000000000001D46F	7
BENNIEPORT	QEJBSVR	346533	N	0000000000000B18	000000000001D497	10
BENNIEPORT	QEJBSVR	346533	N	000000000000006A	0000000000009690	1
BENNIEPORT	QEJBSVR	346533	N	0000000000000064	0000000000009598	6
BENNIEPORT	QEJBSVR	346533	N	0000000000000067	000000000000962F	3

Table of Contents | Previous | Next

5.5.15 IO Total by Job-Thread/IO-Type

This report shows a breakdown of the physical disk IO events that occurred for each IO type within job/thread.



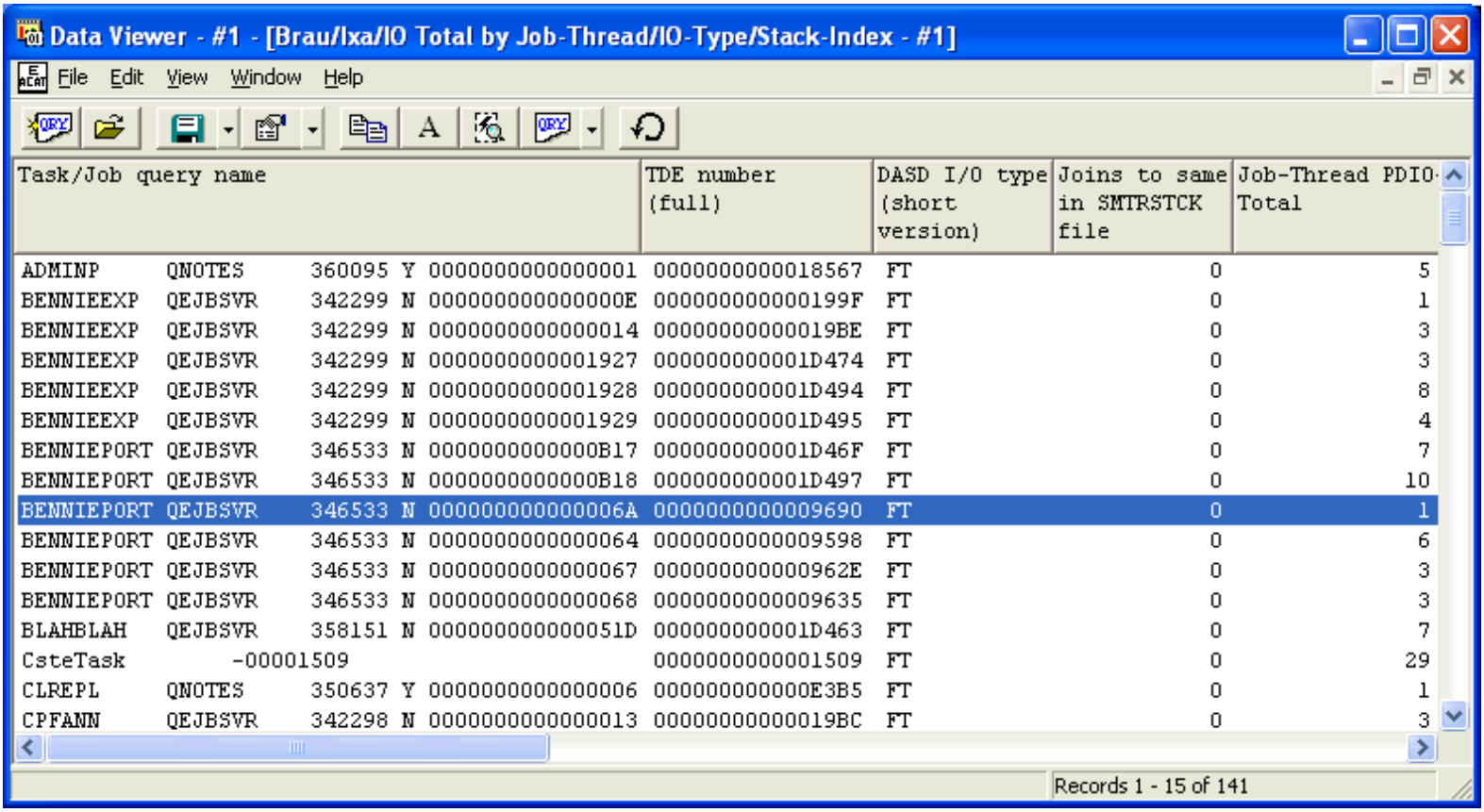
Task/Job query name				TDE number (full)	DASD I/O type (short version)	Job-Thread PDIO-Type Total	Job-Thread PDIO-Type Avg Bytes Len
ADMINP	QNOTES	360095	Y 0000000000000001	0000000000018567	FT	5	4096
BENNIEEXP	QEJBSVR	342299	N 000000000000000E	000000000000199F	FT	1	4096
BENNIEEXP	QEJBSVR	342299	N 0000000000000014	00000000000019BE	FT	3	10922
BENNIEEXP	QEJBSVR	342299	N 0000000000001927	000000000001D474	FT	3	8192
BENNIEEXP	QEJBSVR	342299	N 0000000000001928	000000000001D494	FT	8	4096
BENNIEEXP	QEJBSVR	342299	N 0000000000001929	000000000001D495	FT	4	4096
BENNIEPORT	QEJBSVR	346533	N 0000000000000B17	000000000001D46F	FT	7	4096
BENNIEPORT	QEJBSVR	346533	N 0000000000000B18	000000000001D497	FT	10	6553
BENNIEPORT	QEJBSVR	346533	N 000000000000006A	0000000000009690	FT	1	12288

Table of Contents | Previous | Next

5.5.16 IO Total by Job-Thread/IO-Type/Stack-Index

This report shows a breakdown of the physical disk IO events that occurred for each IO type within job/thread.

If a call stack is available for one of the events for a specific job/IO-type the STCKINDX field will be greater than 0. Double-clicking on a record will show the call stack if it is available.



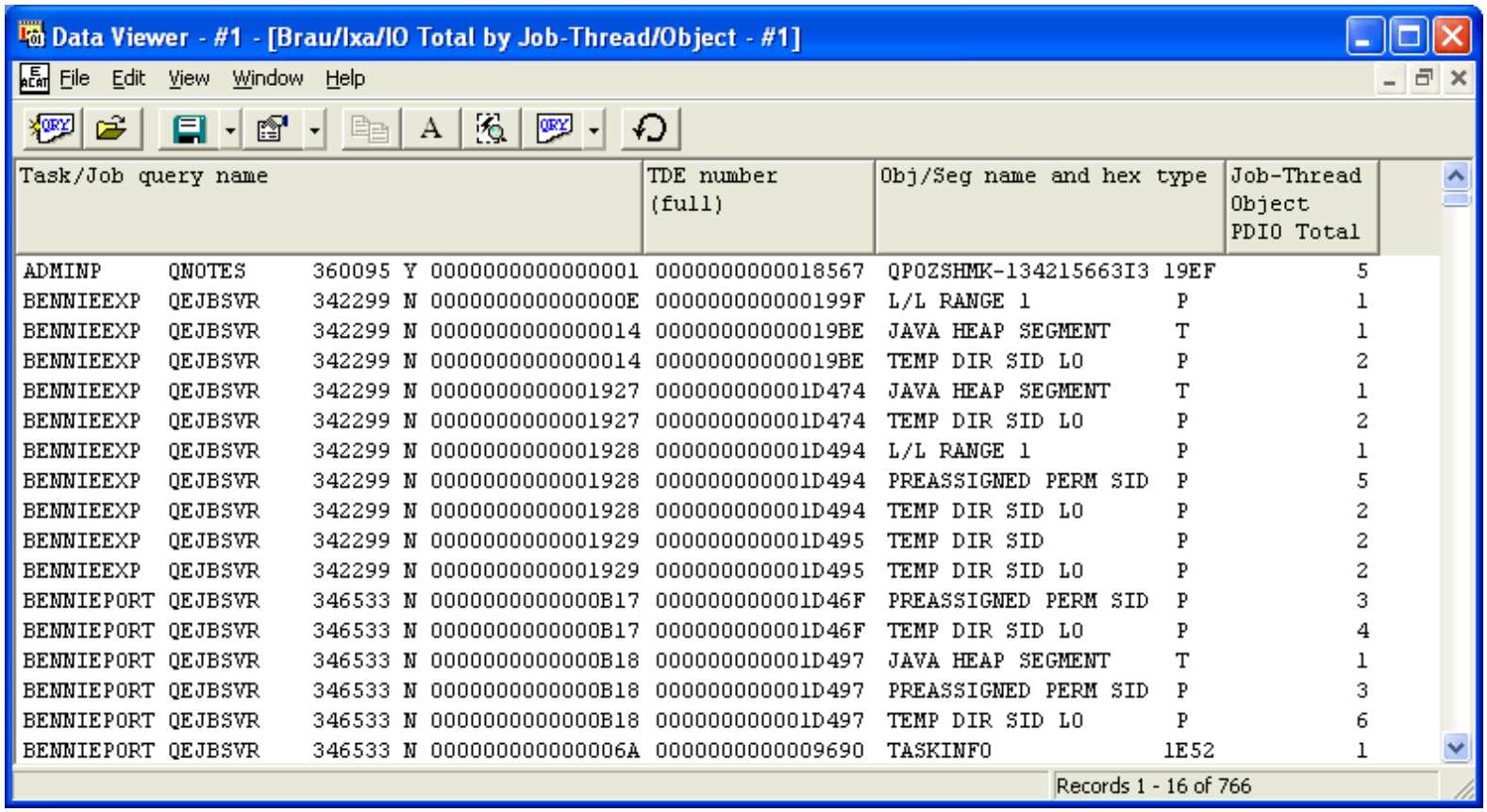
Task/Job query name				TDE number (full)	DASD I/O type (short version)	Joins to same in SMTRSTCK file	Job-Thread PDIO Total
ADMINP	QNOTES	360095	Y	0000000000000001	0000000000018567	FT	0	5
BENNIEEXP	QEJBSVR	342299	N	000000000000000E	000000000000199F	FT	0	1
BENNIEEXP	QEJBSVR	342299	N	0000000000000014	00000000000019BE	FT	0	3
BENNIEEXP	QEJBSVR	342299	N	0000000000001927	000000000001D474	FT	0	3
BENNIEEXP	QEJBSVR	342299	N	0000000000001928	000000000001D494	FT	0	8
BENNIEEXP	QEJBSVR	342299	N	0000000000001929	000000000001D495	FT	0	4
BENNIEPORT	QEJBSVR	346533	N	0000000000000B17	000000000001D46F	FT	0	7
BENNIEPORT	QEJBSVR	346533	N	0000000000000B18	000000000001D497	FT	0	10
BENNIEPORT	QEJBSVR	346533	N	000000000000006A	0000000000009690	FT	0	1
BENNIEPORT	QEJBSVR	346533	N	0000000000000064	0000000000009598	FT	0	6
BENNIEPORT	QEJBSVR	346533	N	0000000000000067	000000000000962E	FT	0	3
BENNIEPORT	QEJBSVR	346533	N	0000000000000068	0000000000009635	FT	0	3
BLAHBLAH	QEJBSVR	358151	N	000000000000051D	000000000001D463	FT	0	7
CsteTask		-00001509			0000000000001509	FT	0	29
CLREPL	QNOTES	350637	Y	0000000000000006	000000000000E3B5	FT	0	1
CPFANN	QEJBSVR	342298	N	0000000000000013	00000000000019BC	FT	0	3

Records 1 - 15 of 141

Table of Contents | Previous | Next

5.5.17 IO Total by Job-Thread/Object

This report shows the total number of physical disk IO events that occurred for each job/thread by object the event was over.



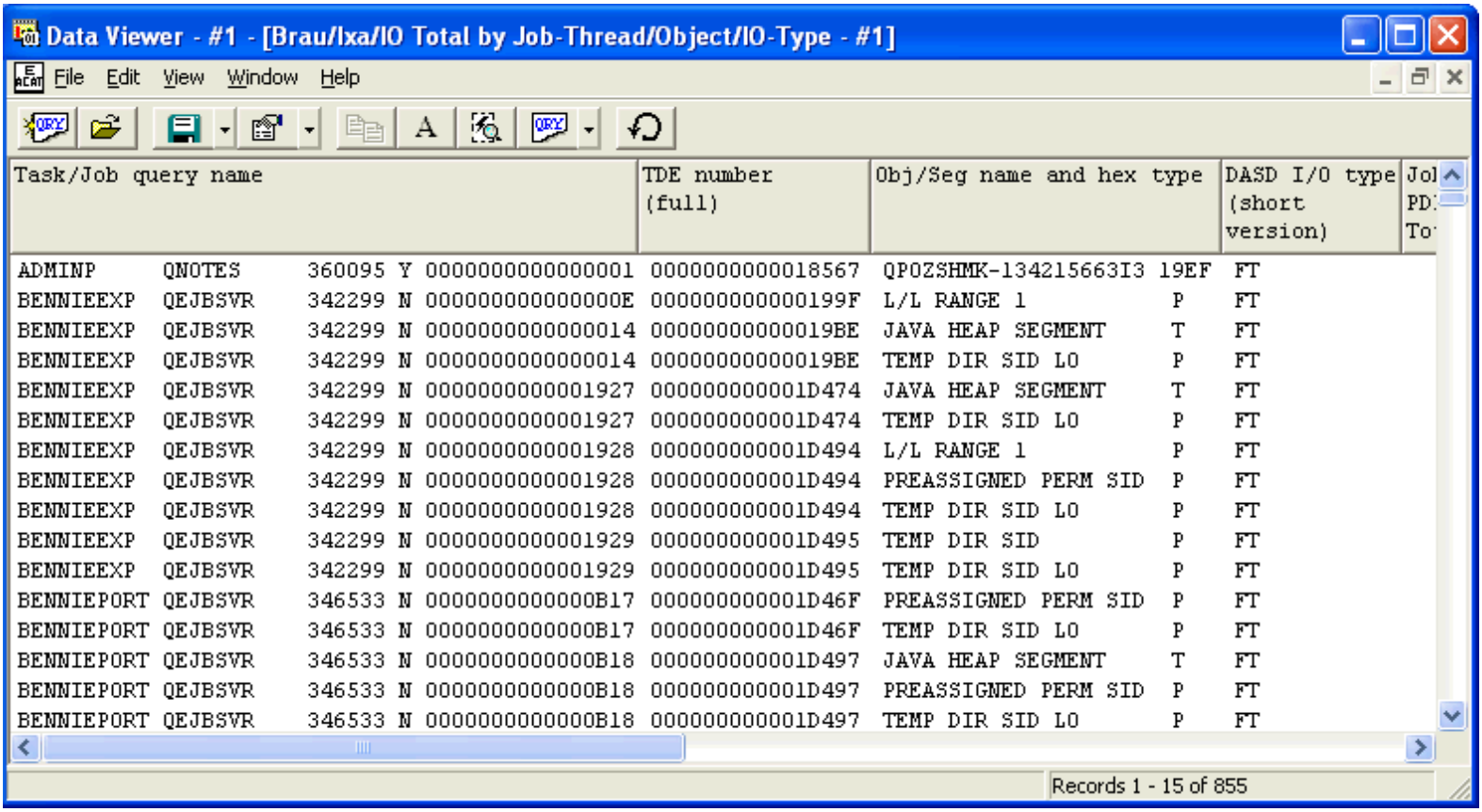
Task/Job query name				TDE number (full)	Obj/Seg name and hex type		Job-Thread Object PDIO Total
ADMINP	QNOTES	360095 Y	0000000000000001	0000000000018567	QP0ZSHMK-13421563I3	19EF	5
BENNIEEXP	QEJBSVR	342299 N	000000000000000E	000000000000199F	L/L RANGE 1	P	1
BENNIEEXP	QEJBSVR	342299 N	0000000000000014	00000000000019BE	JAVA HEAP SEGMENT	T	1
BENNIEEXP	QEJBSVR	342299 N	0000000000000014	00000000000019BE	TEMP DIR SID LO	P	2
BENNIEEXP	QEJBSVR	342299 N	0000000000001927	000000000001D474	JAVA HEAP SEGMENT	T	1
BENNIEEXP	QEJBSVR	342299 N	0000000000001927	000000000001D474	TEMP DIR SID LO	P	2
BENNIEEXP	QEJBSVR	342299 N	0000000000001928	000000000001D494	L/L RANGE 1	P	1
BENNIEEXP	QEJBSVR	342299 N	0000000000001928	000000000001D494	PREASSIGNED PERM SID	P	5
BENNIEEXP	QEJBSVR	342299 N	0000000000001928	000000000001D494	TEMP DIR SID LO	P	2
BENNIEEXP	QEJBSVR	342299 N	0000000000001929	000000000001D495	TEMP DIR SID	P	2
BENNIEEXP	QEJBSVR	342299 N	0000000000001929	000000000001D495	TEMP DIR SID LO	P	2
BENNIEPORT	QEJBSVR	346533 N	0000000000000B17	000000000001D46F	PREASSIGNED PERM SID	P	3
BENNIEPORT	QEJBSVR	346533 N	0000000000000B17	000000000001D46F	TEMP DIR SID LO	P	4
BENNIEPORT	QEJBSVR	346533 N	0000000000000B18	000000000001D497	JAVA HEAP SEGMENT	T	1
BENNIEPORT	QEJBSVR	346533 N	0000000000000B18	000000000001D497	PREASSIGNED PERM SID	P	3
BENNIEPORT	QEJBSVR	346533 N	0000000000000B18	000000000001D497	TEMP DIR SID LO	P	6
BENNIEPORT	QEJBSVR	346533 N	000000000000006A	0000000000009690	TASKINFO	1E52	1

Records 1 - 16 of 766

Table of Contents | Previous | Next

5.5.18 IO Total by Job-Thread/Object/IO-Type

This report shows a breakdown of the physical disk IO events that occurred for each job/thread by object by IO type.



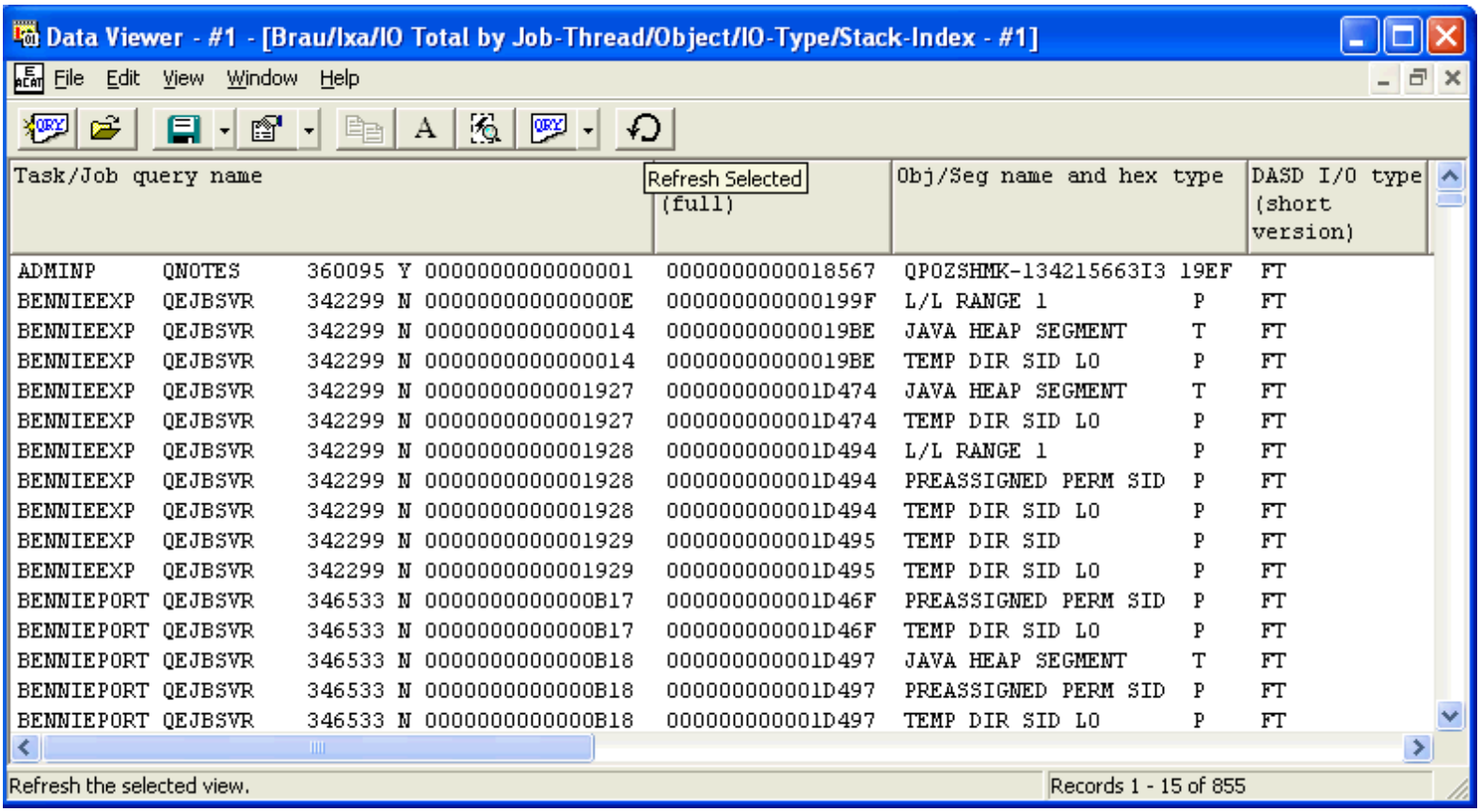
Task/Job query name				TDE number (full)	Obj/Seg name and hex type		DASD I/O type (short version)
ADMINP	QNOTES	360095	Y	0000000000000001	0000000000018567	QP0ZSHMK-134215663I3 19EF	FT
BENNIEEXP	QEJBSVR	342299	N	000000000000000E	000000000000199F	L/L RANGE 1 P	FT
BENNIEEXP	QEJBSVR	342299	N	0000000000000014	00000000000019BE	JAVA HEAP SEGMENT T	FT
BENNIEEXP	QEJBSVR	342299	N	0000000000000014	00000000000019BE	TEMP DIR SID LO P	FT
BENNIEEXP	QEJBSVR	342299	N	0000000000001927	000000000001D474	JAVA HEAP SEGMENT T	FT
BENNIEEXP	QEJBSVR	342299	N	0000000000001927	000000000001D474	TEMP DIR SID LO P	FT
BENNIEEXP	QEJBSVR	342299	N	0000000000001928	000000000001D494	L/L RANGE 1 P	FT
BENNIEEXP	QEJBSVR	342299	N	0000000000001928	000000000001D494	PREASSIGNED PERM SID P	FT
BENNIEEXP	QEJBSVR	342299	N	0000000000001928	000000000001D494	TEMP DIR SID LO P	FT
BENNIEEXP	QEJBSVR	342299	N	0000000000001929	000000000001D495	TEMP DIR SID P	FT
BENNIEEXP	QEJBSVR	342299	N	0000000000001929	000000000001D495	TEMP DIR SID LO P	FT
BENNIEPORT	QEJBSVR	346533	N	0000000000000B17	000000000001D46F	PREASSIGNED PERM SID P	FT
BENNIEPORT	QEJBSVR	346533	N	0000000000000B17	000000000001D46F	TEMP DIR SID LO P	FT
BENNIEPORT	QEJBSVR	346533	N	0000000000000B18	000000000001D497	JAVA HEAP SEGMENT T	FT
BENNIEPORT	QEJBSVR	346533	N	0000000000000B18	000000000001D497	PREASSIGNED PERM SID P	FT
BENNIEPORT	QEJBSVR	346533	N	0000000000000B18	000000000001D497	TEMP DIR SID LO P	FT

Table of Contents | Previous | Next

5.5.19 IO Total by Job-Thread/Object/IO-Type/Stack-Index

This report shows a breakdown of the physical disk IO events that occurred for each job/thread by object by IO type.

If a call stack is available the STCKINDX field will be greater than 0. Double-clicking on a record will show the call stack if it is available.

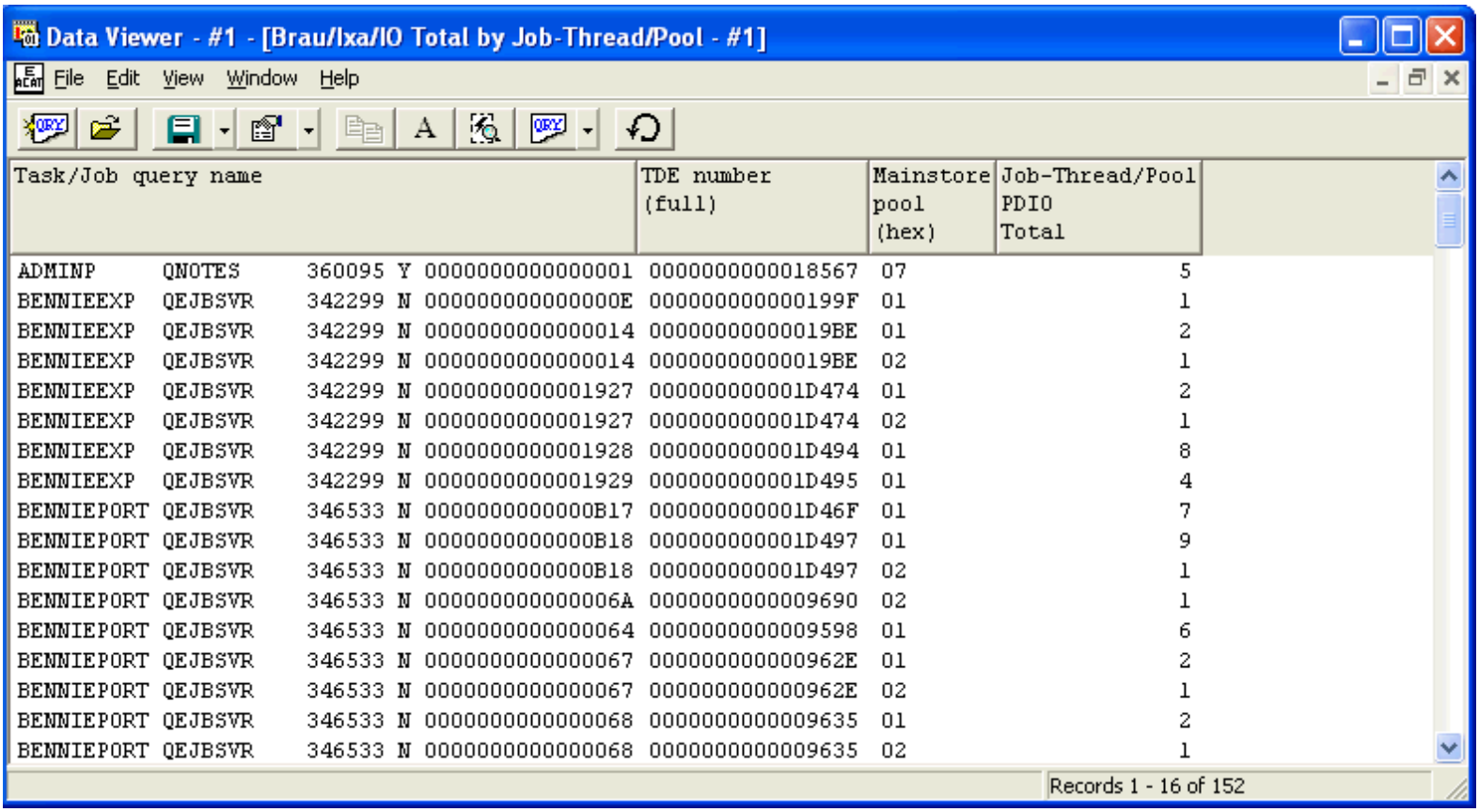


Task/Job query name					Obj/Seg name and hex type		DASD I/O type (short version)
ADMINP	QNOTES	360095	Y	0000000000000001	0000000000018567	QP0ZSHMK-13421563I3 19EF	FT
BENNIEEXP	QEJBSVR	342299	N	000000000000000E	000000000000199F	L/L RANGE 1 P	FT
BENNIEEXP	QEJBSVR	342299	N	0000000000000014	00000000000019BE	JAVA HEAP SEGMENT T	FT
BENNIEEXP	QEJBSVR	342299	N	0000000000000014	00000000000019BE	TEMP DIR SID LO P	FT
BENNIEEXP	QEJBSVR	342299	N	0000000000001927	000000000001D474	JAVA HEAP SEGMENT T	FT
BENNIEEXP	QEJBSVR	342299	N	0000000000001927	000000000001D474	TEMP DIR SID LO P	FT
BENNIEEXP	QEJBSVR	342299	N	0000000000001928	000000000001D494	L/L RANGE 1 P	FT
BENNIEEXP	QEJBSVR	342299	N	0000000000001928	000000000001D494	PREASSIGNED PERM SID P	FT
BENNIEEXP	QEJBSVR	342299	N	0000000000001928	000000000001D494	TEMP DIR SID LO P	FT
BENNIEEXP	QEJBSVR	342299	N	0000000000001929	000000000001D495	TEMP DIR SID P	FT
BENNIEEXP	QEJBSVR	342299	N	0000000000001929	000000000001D495	TEMP DIR SID LO P	FT
BENNIEPORT	QEJBSVR	346533	N	0000000000000B17	000000000001D46F	PREASSIGNED PERM SID P	FT
BENNIEPORT	QEJBSVR	346533	N	0000000000000B17	000000000001D46F	TEMP DIR SID LO P	FT
BENNIEPORT	QEJBSVR	346533	N	0000000000000B18	000000000001D497	JAVA HEAP SEGMENT T	FT
BENNIEPORT	QEJBSVR	346533	N	0000000000000B18	000000000001D497	PREASSIGNED PERM SID P	FT
BENNIEPORT	QEJBSVR	346533	N	0000000000000B18	000000000001D497	TEMP DIR SID LO P	FT

Table of Contents | Previous | Next

5.5.20 IO Total by Job-Thread/Pool

This report shows the total number of physical disk IO events that occurred for each job/thread by mainstorage pool. The pool ID is listed in HEX format within this report.



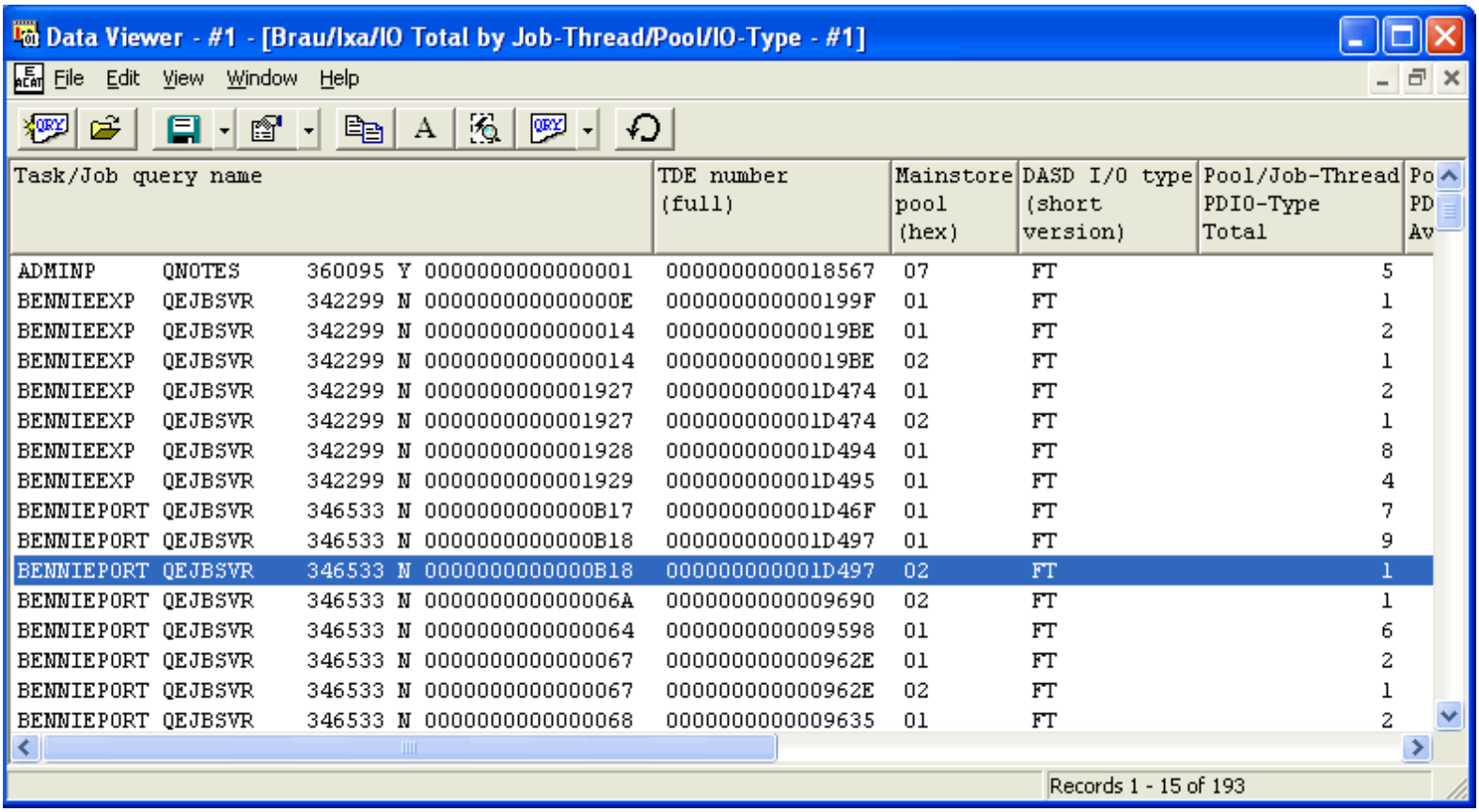
Task/Job query name				TDE number (full)	Mainstore pool (hex)	Job-Thread/Pool PDIO Total
ADMINP	QNOTES	360095	Y	0000000000000001	0000000000018567	07	5
BENNIEEXP	QEJBSVR	342299	N	000000000000000E	000000000000199F	01	1
BENNIEEXP	QEJBSVR	342299	N	0000000000000014	00000000000019BE	01	2
BENNIEEXP	QEJBSVR	342299	N	0000000000000014	00000000000019BE	02	1
BENNIEEXP	QEJBSVR	342299	N	0000000000001927	000000000001D474	01	2
BENNIEEXP	QEJBSVR	342299	N	0000000000001927	000000000001D474	02	1
BENNIEEXP	QEJBSVR	342299	N	0000000000001928	000000000001D494	01	8
BENNIEEXP	QEJBSVR	342299	N	0000000000001929	000000000001D495	01	4
BENNIEPORT	QEJBSVR	346533	N	0000000000000B17	000000000001D46F	01	7
BENNIEPORT	QEJBSVR	346533	N	0000000000000B18	000000000001D497	01	9
BENNIEPORT	QEJBSVR	346533	N	0000000000000B18	000000000001D497	02	1
BENNIEPORT	QEJBSVR	346533	N	000000000000006A	0000000000009690	02	1
BENNIEPORT	QEJBSVR	346533	N	0000000000000064	0000000000009598	01	6
BENNIEPORT	QEJBSVR	346533	N	0000000000000067	000000000000962E	01	2
BENNIEPORT	QEJBSVR	346533	N	0000000000000067	000000000000962E	02	1
BENNIEPORT	QEJBSVR	346533	N	0000000000000068	0000000000009635	01	2
BENNIEPORT	QEJBSVR	346533	N	0000000000000068	0000000000009635	02	1

Records 1 - 16 of 152

Table of Contents | Previous | Next

5.5.21 IO Total by Job-Thread/Pool/IO-Type

This report shows a breakdown of the physical disk IO events that occurred for each job/thread by IO type within each mainstorage pool. The pool ID is listed in HEX format within this report.



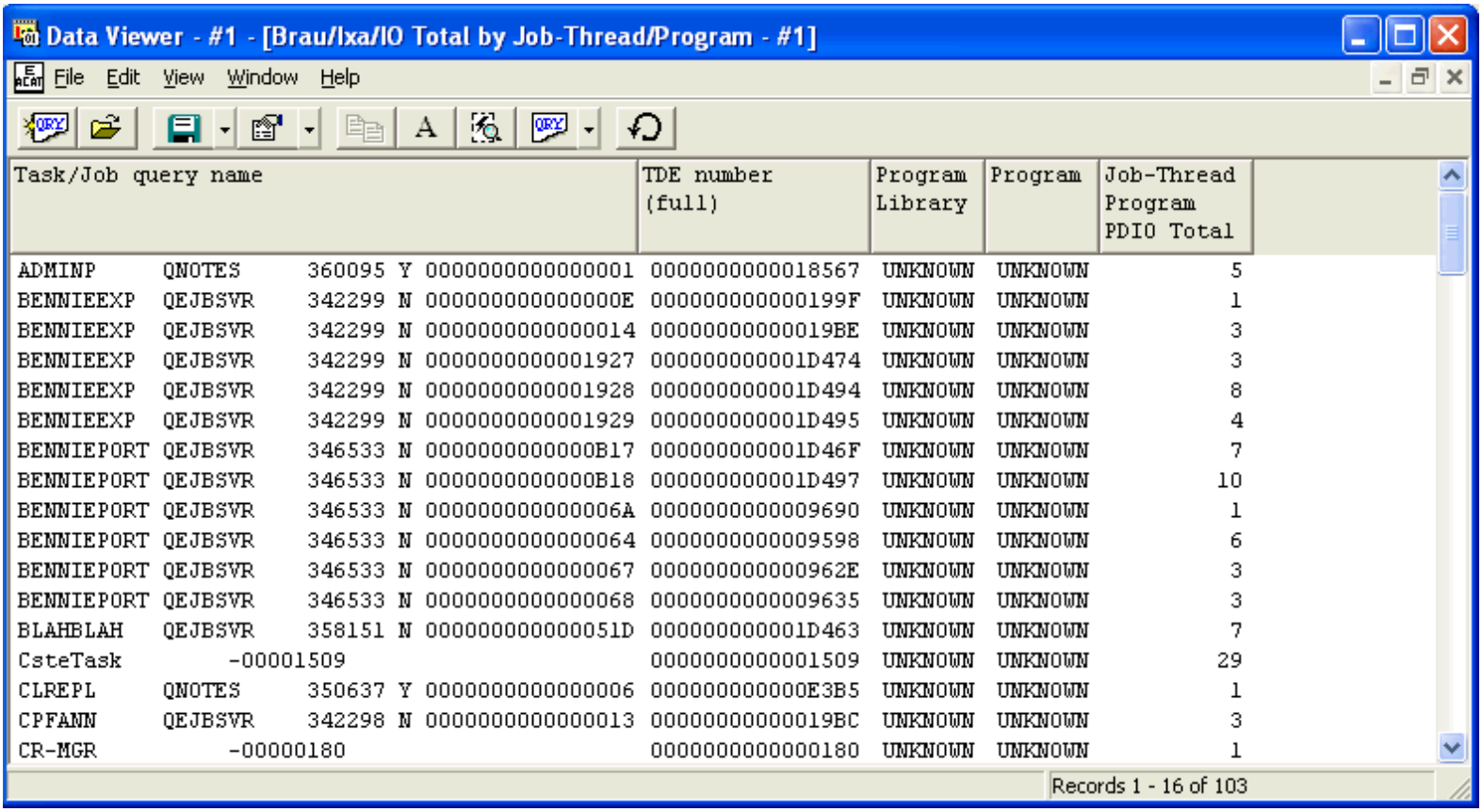
Task/Job query name				TDE number (full)	Mainstore pool (hex)	DASD I/O type (short version)	Pool/Job-Thread PDIO-Type Total	Po PD Av
ADMINP	QNOTES	360095	Y 0000000000000001	0000000000018567	07	FT	5	
BENNIEEXP	QEJBSVR	342299	N 000000000000000E	000000000000199F	01	FT	1	
BENNIEEXP	QEJBSVR	342299	N 0000000000000014	00000000000019BE	01	FT	2	
BENNIEEXP	QEJBSVR	342299	N 0000000000000014	00000000000019BE	02	FT	1	
BENNIEEXP	QEJBSVR	342299	N 0000000000001927	000000000001D474	01	FT	2	
BENNIEEXP	QEJBSVR	342299	N 0000000000001927	000000000001D474	02	FT	1	
BENNIEEXP	QEJBSVR	342299	N 0000000000001928	000000000001D494	01	FT	8	
BENNIEEXP	QEJBSVR	342299	N 0000000000001929	000000000001D495	01	FT	4	
BENNIEPORT	QEJBSVR	346533	N 0000000000000B17	000000000001D46F	01	FT	7	
BENNIEPORT	QEJBSVR	346533	N 0000000000000B18	000000000001D497	01	FT	9	
BENNIEPORT	QEJBSVR	346533	N 0000000000000B18	000000000001D497	02	FT	1	
BENNIEPORT	QEJBSVR	346533	N 000000000000006A	0000000000009690	02	FT	1	
BENNIEPORT	QEJBSVR	346533	N 0000000000000064	0000000000009598	01	FT	6	
BENNIEPORT	QEJBSVR	346533	N 0000000000000067	000000000000962E	01	FT	2	
BENNIEPORT	QEJBSVR	346533	N 0000000000000067	000000000000962E	02	FT	1	
BENNIEPORT	QEJBSVR	346533	N 0000000000000068	0000000000009635	01	FT	2	

Table of Contents | Previous | Next

5.5.22 IO Total by Job-Thread/Program

This report shows the total physical disk IO events that occurred for each program within each job/thread.

Note: In order for program names to be provided, *MIENTRY and *MIEXIT events must be included in the PEX collection.



Task/Job query name				TDE number (full)	Program Library	Program	Job-Thread Program PDIO Total
ADMINP	QNOTES	360095	Y 0000000000000001	0000000000018567	UNKNOWN	UNKNOWN	5
BENNIEEXP	QEJBSVR	342299	N 000000000000000E	000000000000199F	UNKNOWN	UNKNOWN	1
BENNIEEXP	QEJBSVR	342299	N 0000000000000014	00000000000019BE	UNKNOWN	UNKNOWN	3
BENNIEEXP	QEJBSVR	342299	N 0000000000001927	000000000001D474	UNKNOWN	UNKNOWN	3
BENNIEEXP	QEJBSVR	342299	N 0000000000001928	000000000001D494	UNKNOWN	UNKNOWN	8
BENNIEEXP	QEJBSVR	342299	N 0000000000001929	000000000001D495	UNKNOWN	UNKNOWN	4
BENNIEPORT	QEJBSVR	346533	N 0000000000000B17	000000000001D46F	UNKNOWN	UNKNOWN	7
BENNIEPORT	QEJBSVR	346533	N 0000000000000B18	000000000001D497	UNKNOWN	UNKNOWN	10
BENNIEPORT	QEJBSVR	346533	N 000000000000006A	0000000000009690	UNKNOWN	UNKNOWN	1
BENNIEPORT	QEJBSVR	346533	N 0000000000000064	0000000000009598	UNKNOWN	UNKNOWN	6
BENNIEPORT	QEJBSVR	346533	N 0000000000000067	000000000000962E	UNKNOWN	UNKNOWN	3
BENNIEPORT	QEJBSVR	346533	N 0000000000000068	0000000000009635	UNKNOWN	UNKNOWN	3
BLAHBLAH	QEJBSVR	358151	N 000000000000051D	000000000001D463	UNKNOWN	UNKNOWN	7
CsteTask	-00001509			0000000000001509	UNKNOWN	UNKNOWN	29
CLREPL	QNOTES	350637	Y 0000000000000006	000000000000E3B5	UNKNOWN	UNKNOWN	1
CPFANN	QEJBSVR	342298	N 0000000000000013	00000000000019BC	UNKNOWN	UNKNOWN	3
CR-MGR	-00000180			0000000000000180	UNKNOWN	UNKNOWN	1

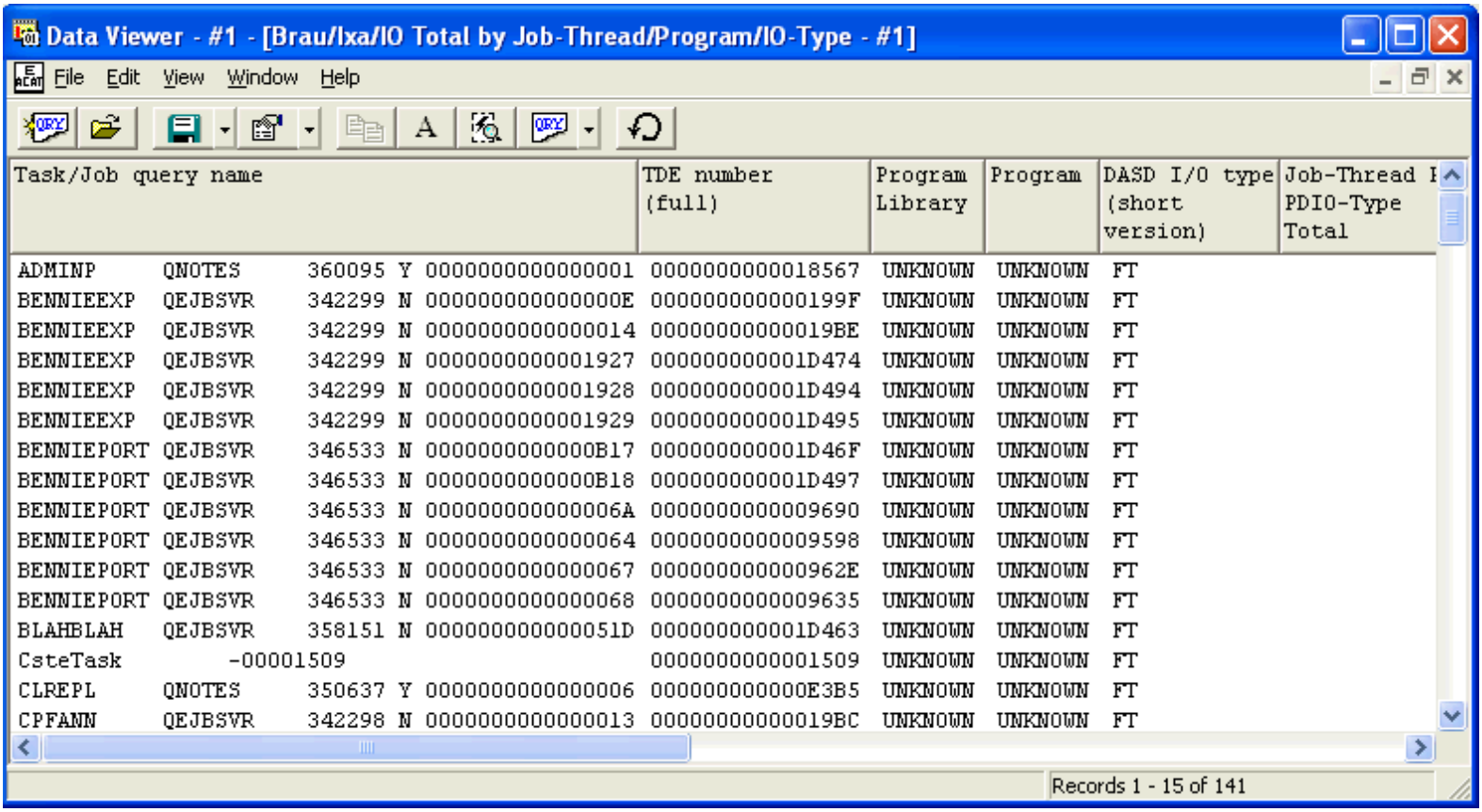
Records 1 - 16 of 103

Table of Contents | Previous | Next

5.5.23 IO Total by Job-Thread/Program/IO-Type

This report shows a breakdown of physical disk IO events that occurred for each IO type by program within each job/thread.

Note: In order for program names to be provided, *MIENTRY and *MIEXIT events must be included in the PEX collection.



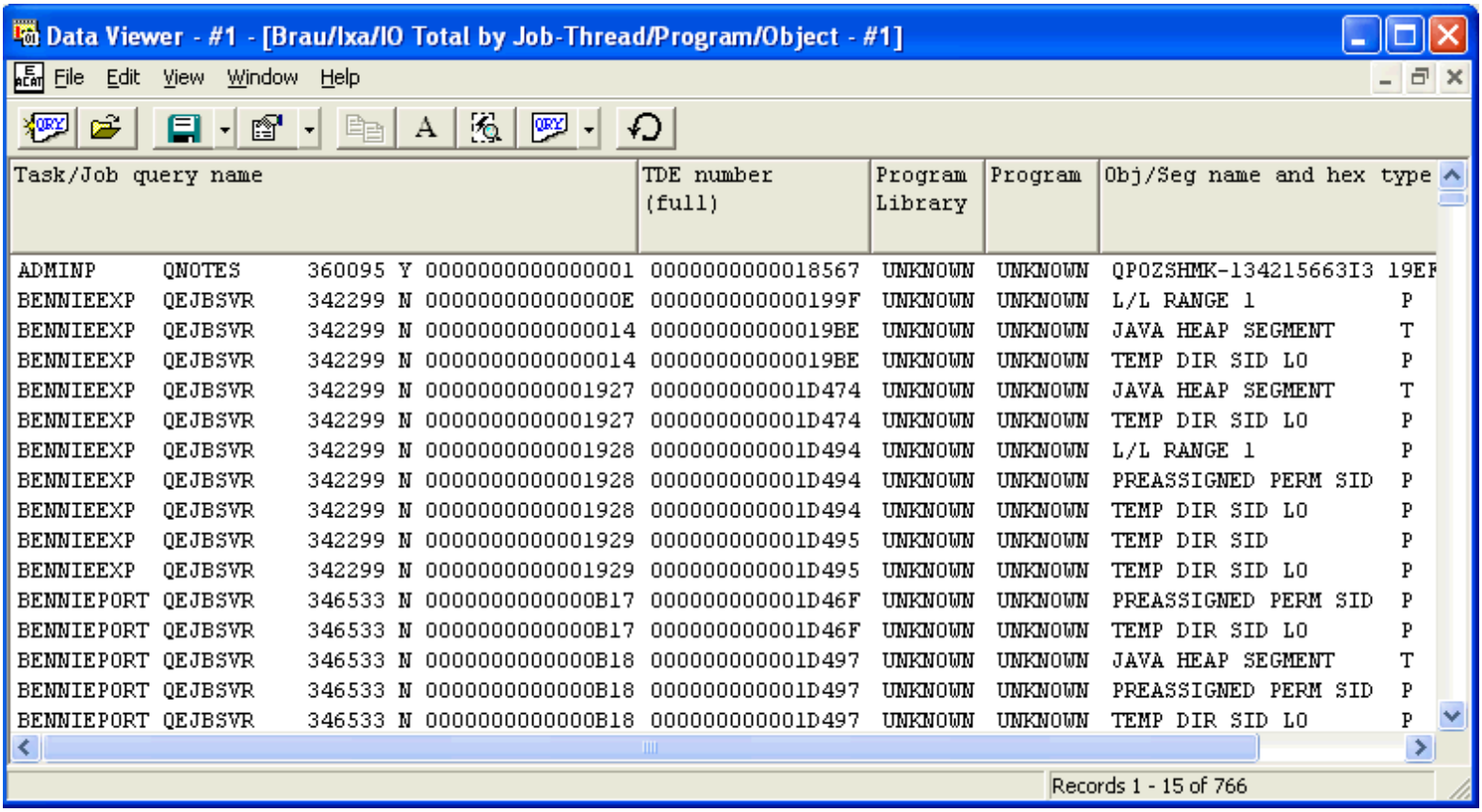
Task/Job query name				TDE number (full)	Program Library	Program	DASD I/O type (short version)	Job-Thread I PDIO-Type Total
ADMINP	QNOTES	360095	Y	0000000000000001	0000000000018567	UNKNOWN	UNKNOWN	FT
BENNIEEXP	QEJBSVR	342299	N	000000000000000E	000000000000199F	UNKNOWN	UNKNOWN	FT
BENNIEEXP	QEJBSVR	342299	N	0000000000000014	00000000000019BE	UNKNOWN	UNKNOWN	FT
BENNIEEXP	QEJBSVR	342299	N	0000000000001927	000000000001D474	UNKNOWN	UNKNOWN	FT
BENNIEEXP	QEJBSVR	342299	N	0000000000001928	000000000001D494	UNKNOWN	UNKNOWN	FT
BENNIEEXP	QEJBSVR	342299	N	0000000000001929	000000000001D495	UNKNOWN	UNKNOWN	FT
BENNIEPORT	QEJBSVR	346533	N	0000000000000B17	000000000001D46F	UNKNOWN	UNKNOWN	FT
BENNIEPORT	QEJBSVR	346533	N	0000000000000B18	000000000001D497	UNKNOWN	UNKNOWN	FT
BENNIEPORT	QEJBSVR	346533	N	000000000000006A	0000000000009690	UNKNOWN	UNKNOWN	FT
BENNIEPORT	QEJBSVR	346533	N	0000000000000064	0000000000009598	UNKNOWN	UNKNOWN	FT
BENNIEPORT	QEJBSVR	346533	N	0000000000000067	000000000000962E	UNKNOWN	UNKNOWN	FT
BENNIEPORT	QEJBSVR	346533	N	0000000000000068	0000000000009635	UNKNOWN	UNKNOWN	FT
BLAHBLAH	QEJBSVR	358151	N	000000000000051D	000000000001D463	UNKNOWN	UNKNOWN	FT
CsteTask	-00001509				0000000000001509	UNKNOWN	UNKNOWN	FT
CLREPL	QNOTES	350637	Y	0000000000000006	000000000000E3B5	UNKNOWN	UNKNOWN	FT
CPFANN	QEJBSVR	342298	N	0000000000000013	00000000000019BC	UNKNOWN	UNKNOWN	FT

Table of Contents | Previous | Next

5.5.24 IO Total by Job-Thread/Program/Object

This report shows the total physical disk IO events that occurred for each object within program within each job/thread.

Note: In order for program names to be provided, *MIENTRY and *MIEXIT events must be included in the PEX collection.



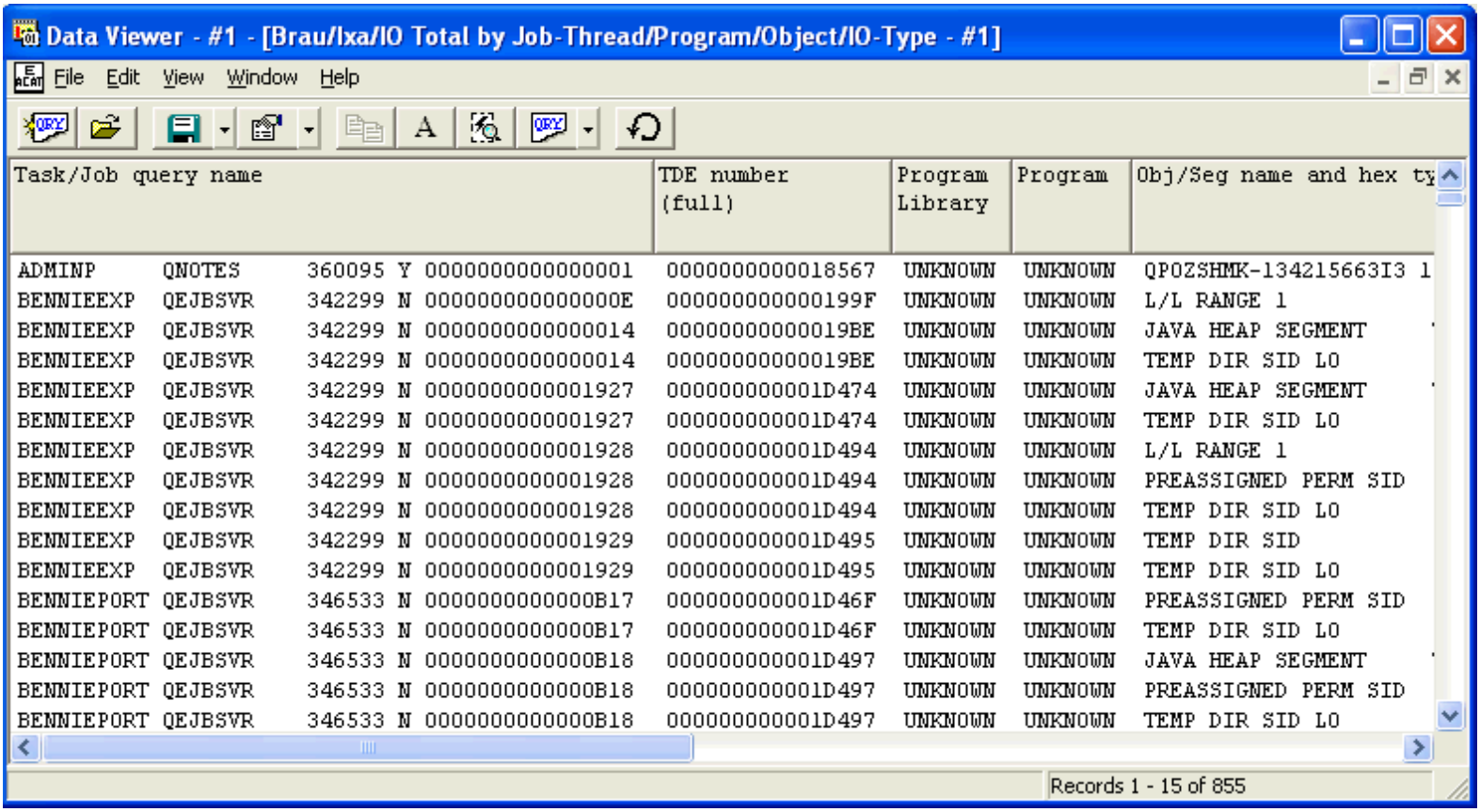
Task/Job query name				TDE number (full)	Program Library	Program	Obj/Seg name and hex type	
ADMINP	QNOTES	360095	Y 0000000000000001	0000000000018567	UNKNOWN	UNKNOWN	QPOZSHMK-134215663I3	19EF
BENNIEEXP	QEJBSVR	342299	N 000000000000000E	000000000000199F	UNKNOWN	UNKNOWN	L/L RANGE 1	P
BENNIEEXP	QEJBSVR	342299	N 0000000000000014	00000000000019BE	UNKNOWN	UNKNOWN	JAVA HEAP SEGMENT	T
BENNIEEXP	QEJBSVR	342299	N 0000000000000014	00000000000019BE	UNKNOWN	UNKNOWN	TEMP DIR SID LO	P
BENNIEEXP	QEJBSVR	342299	N 0000000000001927	000000000001D474	UNKNOWN	UNKNOWN	JAVA HEAP SEGMENT	T
BENNIEEXP	QEJBSVR	342299	N 0000000000001927	000000000001D474	UNKNOWN	UNKNOWN	TEMP DIR SID LO	P
BENNIEEXP	QEJBSVR	342299	N 0000000000001928	000000000001D494	UNKNOWN	UNKNOWN	L/L RANGE 1	P
BENNIEEXP	QEJBSVR	342299	N 0000000000001928	000000000001D494	UNKNOWN	UNKNOWN	PREASSIGNED PERM SID	P
BENNIEEXP	QEJBSVR	342299	N 0000000000001928	000000000001D494	UNKNOWN	UNKNOWN	TEMP DIR SID LO	P
BENNIEEXP	QEJBSVR	342299	N 0000000000001929	000000000001D495	UNKNOWN	UNKNOWN	TEMP DIR SID	P
BENNIEEXP	QEJBSVR	342299	N 0000000000001929	000000000001D495	UNKNOWN	UNKNOWN	TEMP DIR SID LO	P
BENNIEPORT	QEJBSVR	346533	N 0000000000000B17	000000000001D46F	UNKNOWN	UNKNOWN	PREASSIGNED PERM SID	P
BENNIEPORT	QEJBSVR	346533	N 0000000000000B17	000000000001D46F	UNKNOWN	UNKNOWN	TEMP DIR SID LO	P
BENNIEPORT	QEJBSVR	346533	N 0000000000000B18	000000000001D497	UNKNOWN	UNKNOWN	JAVA HEAP SEGMENT	T
BENNIEPORT	QEJBSVR	346533	N 0000000000000B18	000000000001D497	UNKNOWN	UNKNOWN	PREASSIGNED PERM SID	P
BENNIEPORT	QEJBSVR	346533	N 0000000000000B18	000000000001D497	UNKNOWN	UNKNOWN	TEMP DIR SID LO	P

Table of Contents | Previous | Next

5.5.25 IO Total by Job-Thread/Program/Object/IO-Type

This report shows a breakdown of physical disk IO events that occurred for each IO type by object within program within each job/thread.

Note: In order for program names to be provided, *MIENTRY and *MIEXIT events must be included in the PEX collection.



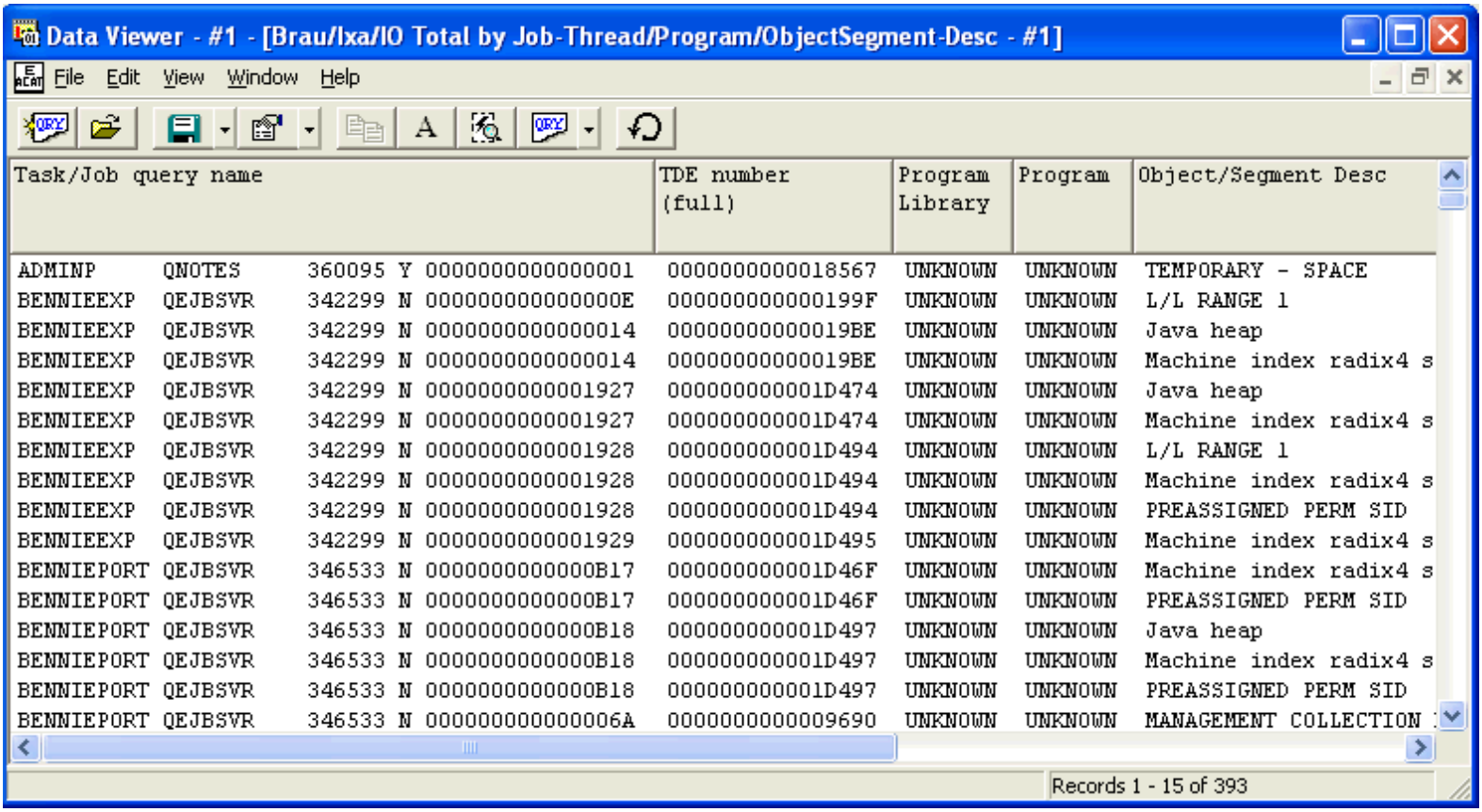
Task/Job query name				TDE number (full)	Program Library	Program	Obj/Seg name and hex ty
ADMINP	QNOTES	360095 Y	0000000000000001	0000000000018567	UNKNOWN	UNKNOWN	QPOZSHMK-13421566313 1
BENNIEEXP	QEJBSVR	342299 N	000000000000000E	000000000000199F	UNKNOWN	UNKNOWN	L/L RANGE 1
BENNIEEXP	QEJBSVR	342299 N	0000000000000014	00000000000019BE	UNKNOWN	UNKNOWN	JAVA HEAP SEGMENT
BENNIEEXP	QEJBSVR	342299 N	0000000000000014	00000000000019BE	UNKNOWN	UNKNOWN	TEMP DIR SID LO
BENNIEEXP	QEJBSVR	342299 N	0000000000001927	000000000001D474	UNKNOWN	UNKNOWN	JAVA HEAP SEGMENT
BENNIEEXP	QEJBSVR	342299 N	0000000000001927	000000000001D474	UNKNOWN	UNKNOWN	TEMP DIR SID LO
BENNIEEXP	QEJBSVR	342299 N	0000000000001928	000000000001D494	UNKNOWN	UNKNOWN	L/L RANGE 1
BENNIEEXP	QEJBSVR	342299 N	0000000000001928	000000000001D494	UNKNOWN	UNKNOWN	PREASSIGNED PERM SID
BENNIEEXP	QEJBSVR	342299 N	0000000000001928	000000000001D494	UNKNOWN	UNKNOWN	TEMP DIR SID LO
BENNIEEXP	QEJBSVR	342299 N	0000000000001929	000000000001D495	UNKNOWN	UNKNOWN	TEMP DIR SID
BENNIEEXP	QEJBSVR	342299 N	0000000000001929	000000000001D495	UNKNOWN	UNKNOWN	TEMP DIR SID LO
BENNIEPORT	QEJBSVR	346533 N	0000000000000B17	000000000001D46F	UNKNOWN	UNKNOWN	PREASSIGNED PERM SID
BENNIEPORT	QEJBSVR	346533 N	0000000000000B17	000000000001D46F	UNKNOWN	UNKNOWN	TEMP DIR SID LO
BENNIEPORT	QEJBSVR	346533 N	0000000000000B18	000000000001D497	UNKNOWN	UNKNOWN	JAVA HEAP SEGMENT
BENNIEPORT	QEJBSVR	346533 N	0000000000000B18	000000000001D497	UNKNOWN	UNKNOWN	PREASSIGNED PERM SID
BENNIEPORT	QEJBSVR	346533 N	0000000000000B18	000000000001D497	UNKNOWN	UNKNOWN	TEMP DIR SID LO

Table of Contents | Previous | Next

5.5.26 IO Total by Job-Thread/Program/ObjectSegment-Desc

This report shows the total physical disk IO events that occurred for each object type within program within each job/thread. The object type descriptions are listed in this report.

Note: In order for program names to be provided, *MIENTRY and *MIEXIT events must be included in the PEX collection.



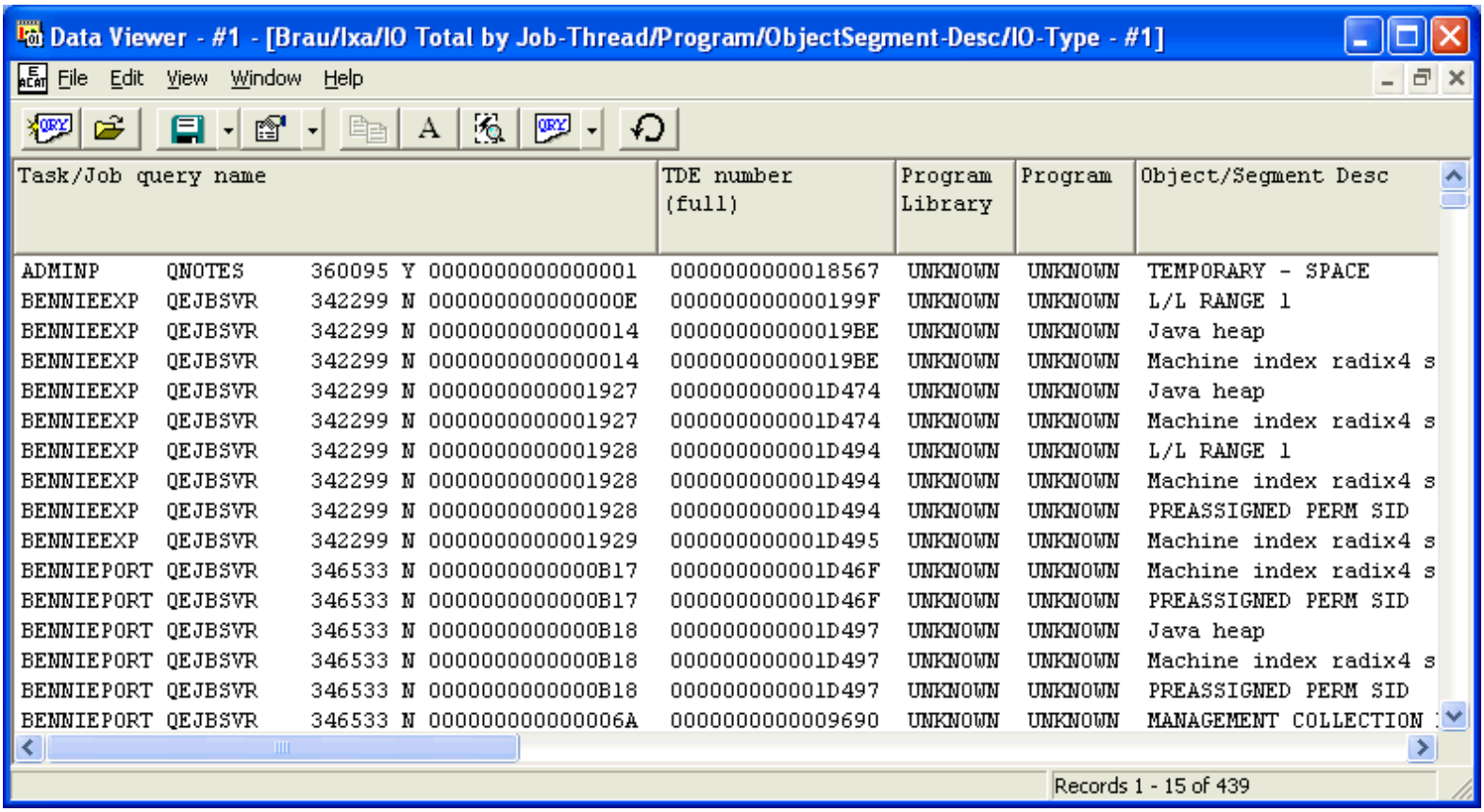
Task/Job query name				TDE number (full)	Program Library	Program	Object/Segment Desc
ADMINP	QNOTES	360095	Y 0000000000000001	0000000000018567	UNKNOWN	UNKNOWN	TEMPORARY - SPACE
BENNIEEXP	QEJBSVR	342299	N 000000000000000E	000000000000199F	UNKNOWN	UNKNOWN	L/L RANGE 1
BENNIEEXP	QEJBSVR	342299	N 0000000000000014	00000000000019BE	UNKNOWN	UNKNOWN	Java heap
BENNIEEXP	QEJBSVR	342299	N 0000000000000014	00000000000019BE	UNKNOWN	UNKNOWN	Machine index radix4 s
BENNIEEXP	QEJBSVR	342299	N 0000000000001927	000000000001D474	UNKNOWN	UNKNOWN	Java heap
BENNIEEXP	QEJBSVR	342299	N 0000000000001927	000000000001D474	UNKNOWN	UNKNOWN	Machine index radix4 s
BENNIEEXP	QEJBSVR	342299	N 0000000000001928	000000000001D494	UNKNOWN	UNKNOWN	L/L RANGE 1
BENNIEEXP	QEJBSVR	342299	N 0000000000001928	000000000001D494	UNKNOWN	UNKNOWN	Machine index radix4 s
BENNIEEXP	QEJBSVR	342299	N 0000000000001928	000000000001D494	UNKNOWN	UNKNOWN	PREASSIGNED PERM SID
BENNIEEXP	QEJBSVR	342299	N 0000000000001929	000000000001D495	UNKNOWN	UNKNOWN	Machine index radix4 s
BENNIEPORT	QEJBSVR	346533	N 0000000000000B17	000000000001D46F	UNKNOWN	UNKNOWN	Machine index radix4 s
BENNIEPORT	QEJBSVR	346533	N 0000000000000B17	000000000001D46F	UNKNOWN	UNKNOWN	PREASSIGNED PERM SID
BENNIEPORT	QEJBSVR	346533	N 0000000000000B18	000000000001D497	UNKNOWN	UNKNOWN	Java heap
BENNIEPORT	QEJBSVR	346533	N 0000000000000B18	000000000001D497	UNKNOWN	UNKNOWN	Machine index radix4 s
BENNIEPORT	QEJBSVR	346533	N 0000000000000B18	000000000001D497	UNKNOWN	UNKNOWN	PREASSIGNED PERM SID
BENNIEPORT	QEJBSVR	346533	N 000000000000006A	0000000000009690	UNKNOWN	UNKNOWN	MANAGEMENT COLLECTION

Table of Contents | Previous | Next

5.5.27 IO Total by Job-Thread/Program/ObjectSegment-Desc/IO-Type

This report shows a breakdown of the physical disk IO events that occurred for each IO type within object type within program within each job/thread. The object type descriptions are listed in this report.

Note: In order for program names to be provided, *MIENTRY and *MIEXIT events must be included in the PEX collection.

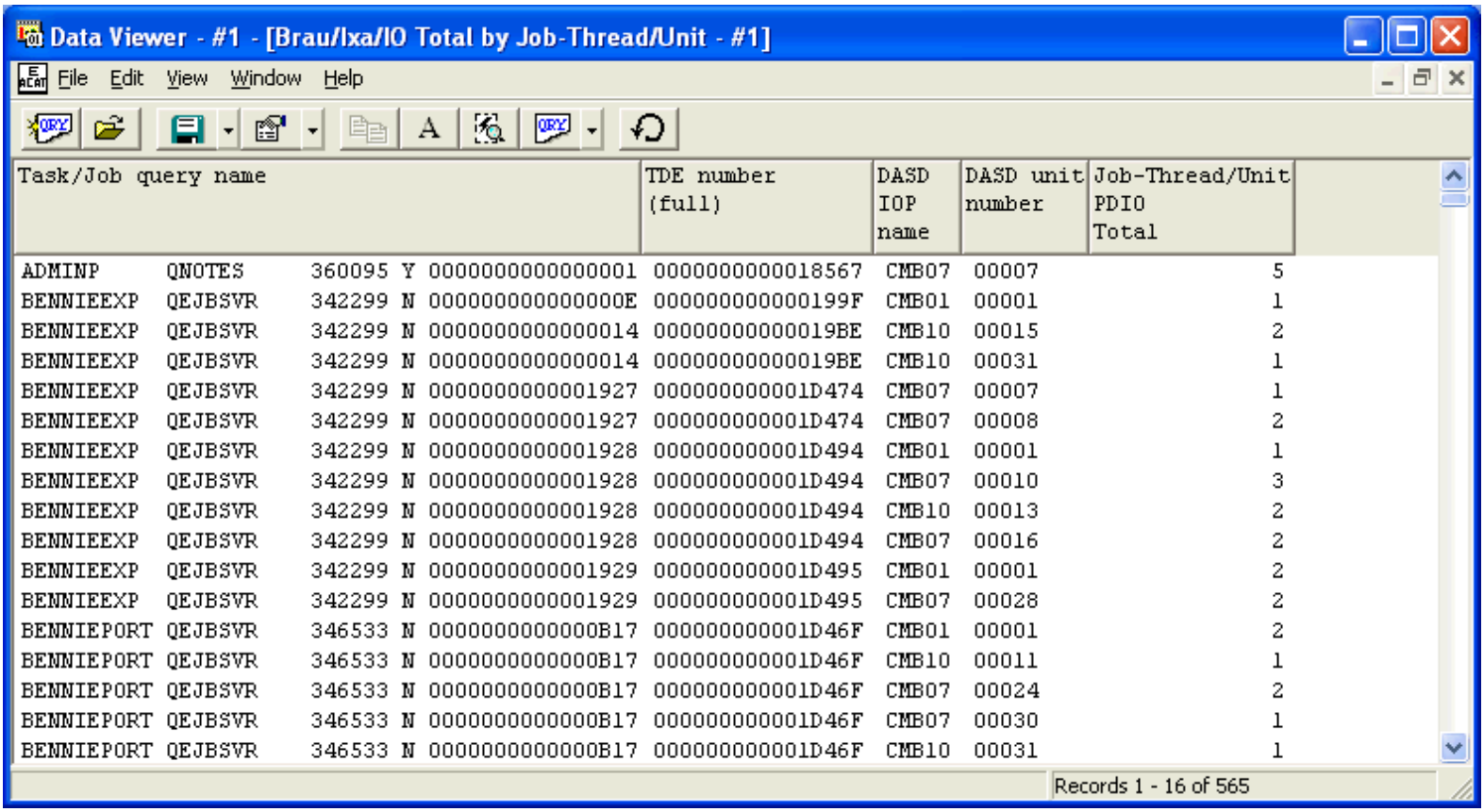


Task/Job query name				TDE number (full)	Program Library	Program	Object/Segment Desc
ADMINP	QNOTES	360095 Y	0000000000000001	0000000000018567	UNKNOWN	UNKNOWN	TEMPORARY - SPACE
BENNIEEXP	QEJBSVR	342299 N	000000000000000E	000000000000199F	UNKNOWN	UNKNOWN	L/L RANGE 1
BENNIEEXP	QEJBSVR	342299 N	0000000000000014	00000000000019BE	UNKNOWN	UNKNOWN	Java heap
BENNIEEXP	QEJBSVR	342299 N	0000000000000014	00000000000019BE	UNKNOWN	UNKNOWN	Machine index radix4 s
BENNIEEXP	QEJBSVR	342299 N	0000000000001927	000000000001D474	UNKNOWN	UNKNOWN	Java heap
BENNIEEXP	QEJBSVR	342299 N	0000000000001927	000000000001D474	UNKNOWN	UNKNOWN	Machine index radix4 s
BENNIEEXP	QEJBSVR	342299 N	0000000000001928	000000000001D494	UNKNOWN	UNKNOWN	L/L RANGE 1
BENNIEEXP	QEJBSVR	342299 N	0000000000001928	000000000001D494	UNKNOWN	UNKNOWN	Machine index radix4 s
BENNIEEXP	QEJBSVR	342299 N	0000000000001928	000000000001D494	UNKNOWN	UNKNOWN	PREASSIGNED PERM SID
BENNIEEXP	QEJBSVR	342299 N	0000000000001929	000000000001D495	UNKNOWN	UNKNOWN	Machine index radix4 s
BENNIEPORT	QEJBSVR	346533 N	0000000000000B17	000000000001D46F	UNKNOWN	UNKNOWN	Machine index radix4 s
BENNIEPORT	QEJBSVR	346533 N	0000000000000B17	000000000001D46F	UNKNOWN	UNKNOWN	PREASSIGNED PERM SID
BENNIEPORT	QEJBSVR	346533 N	0000000000000B18	000000000001D497	UNKNOWN	UNKNOWN	Java heap
BENNIEPORT	QEJBSVR	346533 N	0000000000000B18	000000000001D497	UNKNOWN	UNKNOWN	Machine index radix4 s
BENNIEPORT	QEJBSVR	346533 N	0000000000000B18	000000000001D497	UNKNOWN	UNKNOWN	PREASSIGNED PERM SID
BENNIEPORT	QEJBSVR	346533 N	000000000000006A	0000000000009690	UNKNOWN	UNKNOWN	MANAGEMENT COLLECTION

Table of Contents | Previous | Next

5.5.28 IO Total by Job-Thread/Unit

This report shows the total number of physical disk IO events that occurred for each disk unit within job/thread.

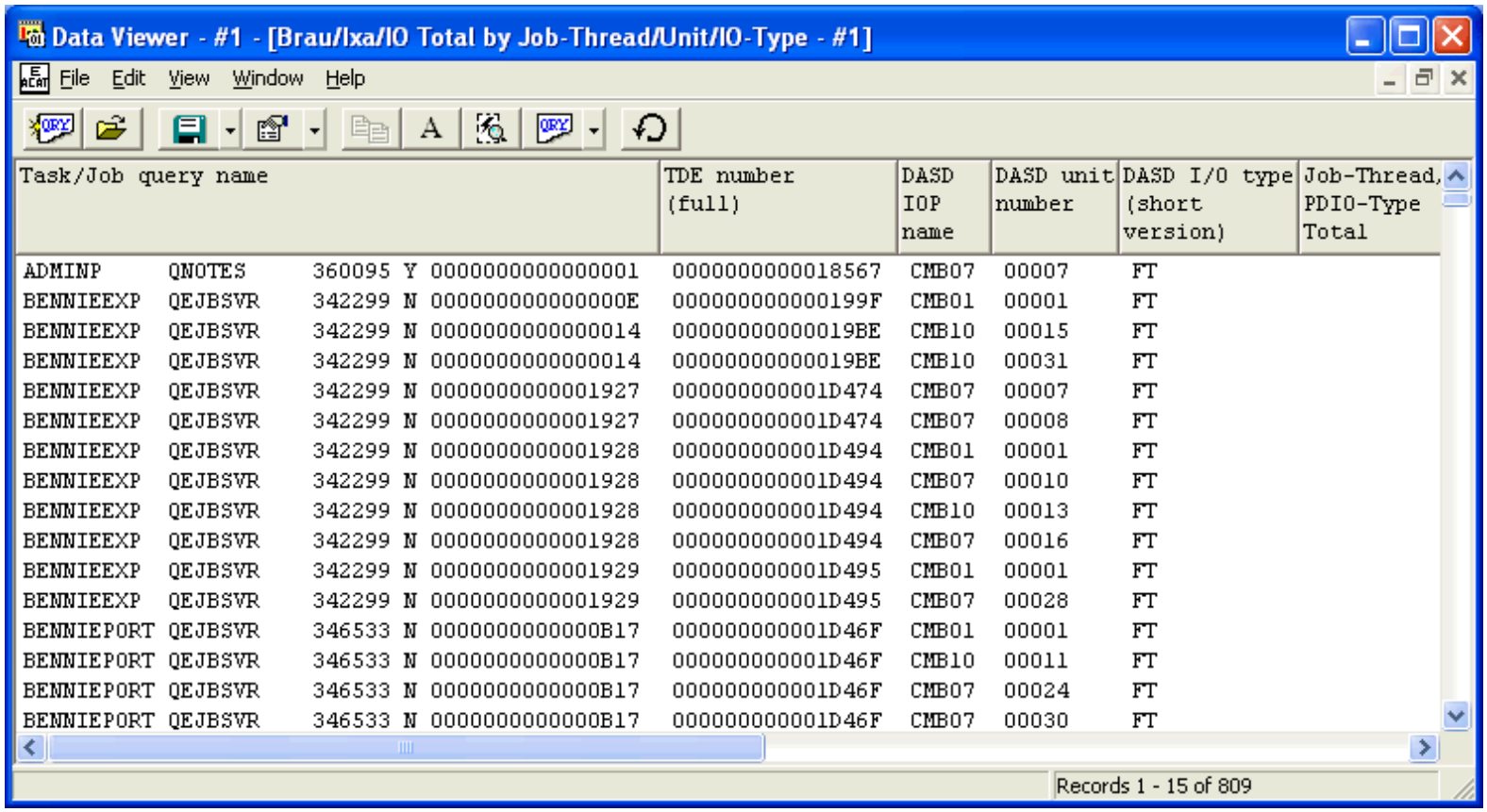


Task/Job query name				TDE number (full)	DASD IOP name	DASD unit number	Job-Thread/Unit PDIO Total
ADMINP	QNOTES	360095	Y	0000000000000001	0000000000018567	CMB07	00007	5
BENNIEEXP	QEJBSVR	342299	N	000000000000000E	000000000000199F	CMB01	00001	1
BENNIEEXP	QEJBSVR	342299	N	0000000000000014	00000000000019BE	CMB10	00015	2
BENNIEEXP	QEJBSVR	342299	N	0000000000000014	00000000000019BE	CMB10	00031	1
BENNIEEXP	QEJBSVR	342299	N	0000000000001927	000000000001D474	CMB07	00007	1
BENNIEEXP	QEJBSVR	342299	N	0000000000001927	000000000001D474	CMB07	00008	2
BENNIEEXP	QEJBSVR	342299	N	0000000000001928	000000000001D494	CMB01	00001	1
BENNIEEXP	QEJBSVR	342299	N	0000000000001928	000000000001D494	CMB07	00010	3
BENNIEEXP	QEJBSVR	342299	N	0000000000001928	000000000001D494	CMB10	00013	2
BENNIEEXP	QEJBSVR	342299	N	0000000000001928	000000000001D494	CMB07	00016	2
BENNIEEXP	QEJBSVR	342299	N	0000000000001929	000000000001D495	CMB01	00001	2
BENNIEEXP	QEJBSVR	342299	N	0000000000001929	000000000001D495	CMB07	00028	2
BENNIEPORT	QEJBSVR	346533	N	0000000000000B17	000000000001D46F	CMB01	00001	2
BENNIEPORT	QEJBSVR	346533	N	0000000000000B17	000000000001D46F	CMB10	00011	1
BENNIEPORT	QEJBSVR	346533	N	0000000000000B17	000000000001D46F	CMB07	00024	2
BENNIEPORT	QEJBSVR	346533	N	0000000000000B17	000000000001D46F	CMB07	00030	1
BENNIEPORT	QEJBSVR	346533	N	0000000000000B17	000000000001D46F	CMB10	00031	1

Table of Contents | Previous | Next

5.5.29 IO Total by Job-Thread/Unit/IO-Type

This report shows a breakdown of physical disk IO events that occurred for each IO type within DASD unit within job.

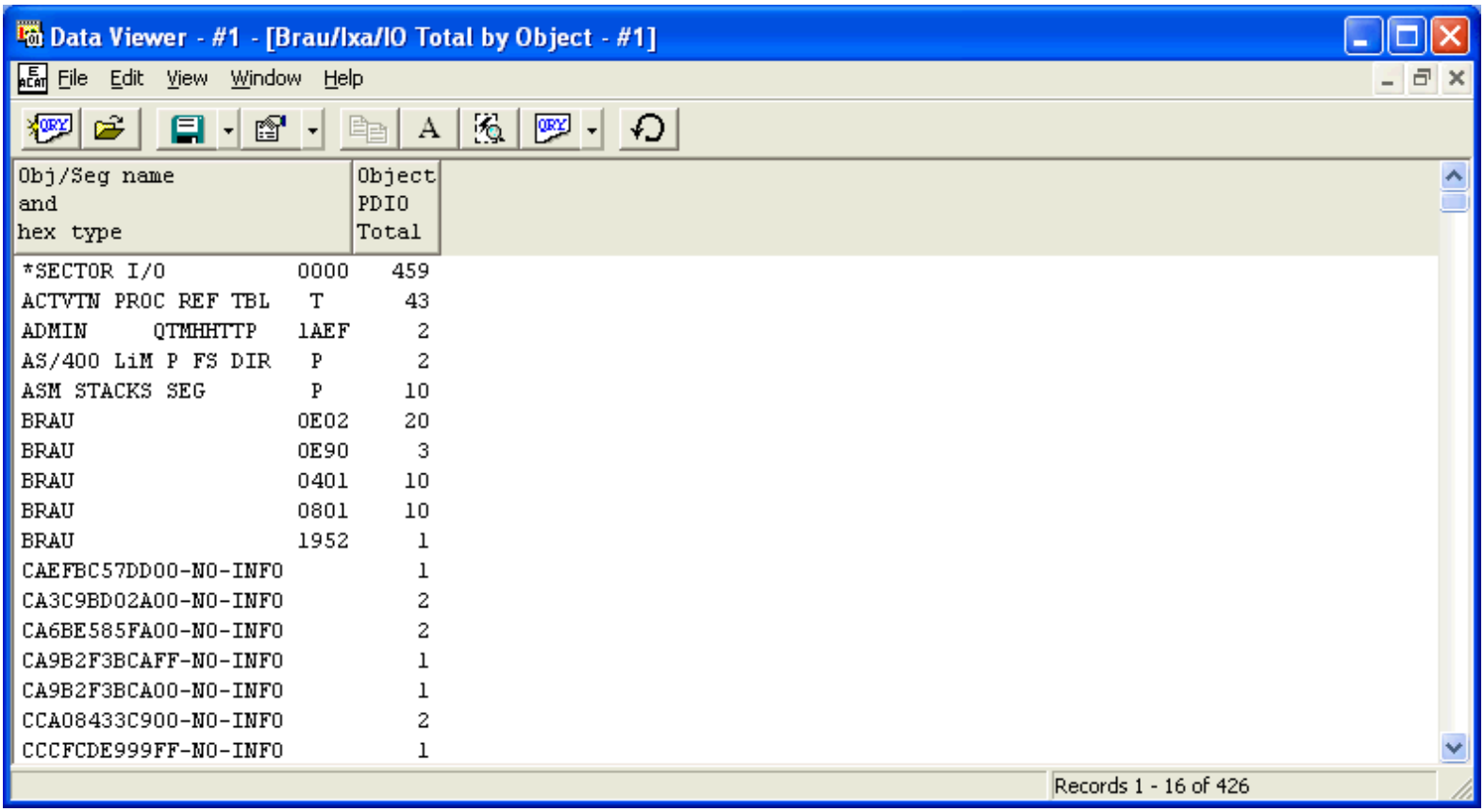


Task/Job query name				TDE number (full)	DASD IOP name	DASD unit number	DASD I/O type (short version)	Job-Thread, PDIO-Type Total
ADMINP	QNOTES	360095	Y 0000000000000001	0000000000018567	CMB07	00007	FT	
BENNIEEXP	QEJBSVR	342299	N 000000000000000E	000000000000199F	CMB01	00001	FT	
BENNIEEXP	QEJBSVR	342299	N 0000000000000014	00000000000019BE	CMB10	00015	FT	
BENNIEEXP	QEJBSVR	342299	N 0000000000000014	00000000000019BE	CMB10	00031	FT	
BENNIEEXP	QEJBSVR	342299	N 0000000000001927	000000000001D474	CMB07	00007	FT	
BENNIEEXP	QEJBSVR	342299	N 0000000000001927	000000000001D474	CMB07	00008	FT	
BENNIEEXP	QEJBSVR	342299	N 0000000000001928	000000000001D494	CMB01	00001	FT	
BENNIEEXP	QEJBSVR	342299	N 0000000000001928	000000000001D494	CMB07	00010	FT	
BENNIEEXP	QEJBSVR	342299	N 0000000000001928	000000000001D494	CMB10	00013	FT	
BENNIEEXP	QEJBSVR	342299	N 0000000000001928	000000000001D494	CMB07	00016	FT	
BENNIEEXP	QEJBSVR	342299	N 0000000000001929	000000000001D495	CMB01	00001	FT	
BENNIEEXP	QEJBSVR	342299	N 0000000000001929	000000000001D495	CMB07	00028	FT	
BENNIEPORT	QEJBSVR	346533	N 0000000000000B17	000000000001D46F	CMB01	00001	FT	
BENNIEPORT	QEJBSVR	346533	N 0000000000000B17	000000000001D46F	CMB10	00011	FT	
BENNIEPORT	QEJBSVR	346533	N 0000000000000B17	000000000001D46F	CMB07	00024	FT	
BENNIEPORT	QEJBSVR	346533	N 0000000000000B17	000000000001D46F	CMB07	00030	FT	

Table of Contents | Previous | Next

5.5.30 IO Total by Object

This report shows the total number of physical disk IO events that occurred for each object within the collection.

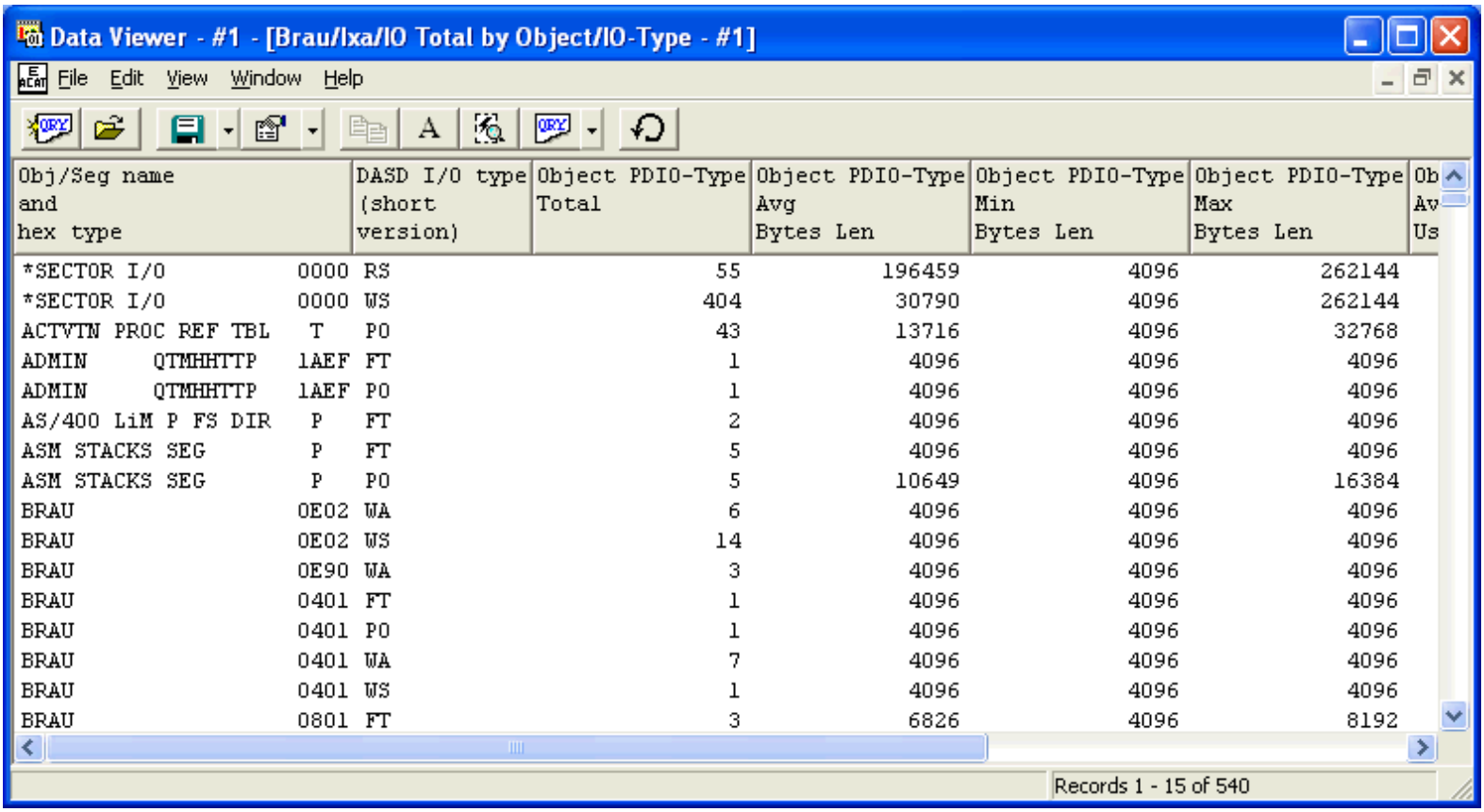


Obj/Seg name and hex type		Object PDIO Total
*SECTOR I/O	0000	459
ACTVTN PROC REF TBL	T	43
ADMIN QTMHHTTP	1AEF	2
AS/400 LiM P FS DIR	P	2
ASM STACKS SEG	P	10
BRAU	0E02	20
BRAU	0E90	3
BRAU	0401	10
BRAU	0801	10
BRAU	1952	1
CAEFBC57DD00-N0-INF0		1
CA3C9BD02A00-N0-INF0		2
CA6BE585FA00-N0-INF0		2
CA9B2F3BCAFF-N0-INF0		1
CA9B2F3BCA00-N0-INF0		1
CCA08433C900-N0-INF0		2
CCCFCDE999FF-N0-INF0		1

Table of Contents | Previous | Next

5.5.31 IO Total by Object/IO-Type

This report shows a breakdown of physical disk IO events that occurred by IO type within object.



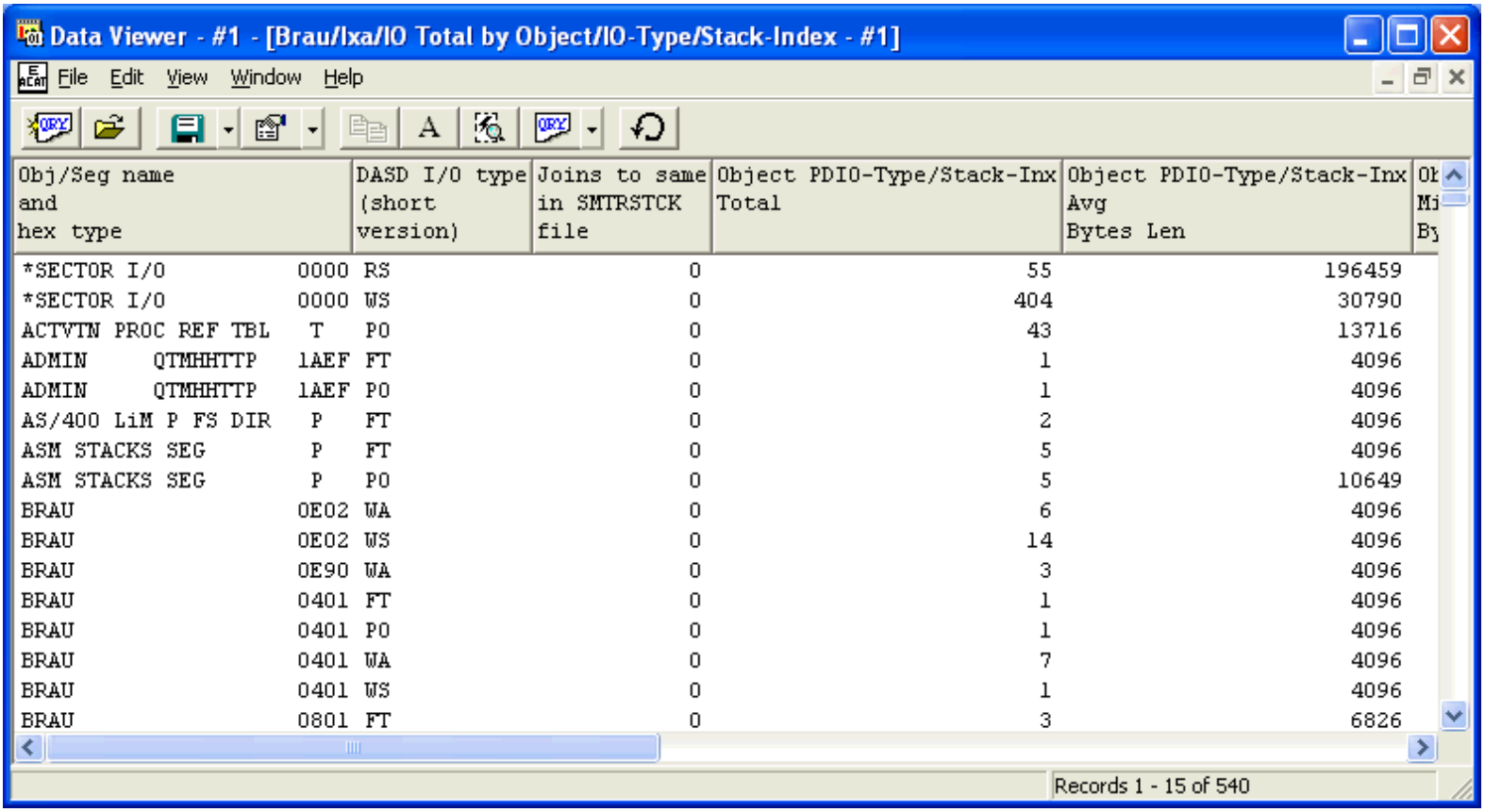
Obj/Seg name and hex type	DASD I/O type (short version)	Object PDIO-Type Total	Object PDIO-Type Avg Bytes Len	Object PDIO-Type Min Bytes Len	Object PDIO-Type Max Bytes Len	Ob Av Us
*SECTOR I/O 0000	RS	55	196459	4096	262144	
*SECTOR I/O 0000	WS	404	30790	4096	262144	
ACTVTN PROC REF TBL T	PO	43	13716	4096	32768	
ADMIN QTMHHTTP 1AEF	FT	1	4096	4096	4096	
ADMIN QTMHHTTP 1AEF	PO	1	4096	4096	4096	
AS/400 LiM P FS DIR P	FT	2	4096	4096	4096	
ASM STACKS SEG P	FT	5	4096	4096	4096	
ASM STACKS SEG P	PO	5	10649	4096	16384	
BRAU 0E02	WA	6	4096	4096	4096	
BRAU 0E02	WS	14	4096	4096	4096	
BRAU 0E90	WA	3	4096	4096	4096	
BRAU 0401	FT	1	4096	4096	4096	
BRAU 0401	PO	1	4096	4096	4096	
BRAU 0401	WA	7	4096	4096	4096	
BRAU 0401	WS	1	4096	4096	4096	
BRAU 0801	FT	3	6826	4096	8192	

Table of Contents | Previous | Next

5.5.32 IO Total by Object/IO-Type/Stack-Index

This report shows a breakdown of physical disk IO events that occurred by IO type within object.

If a call stack is available the STCKINDX field will be greater than 0. Double-clicking on a record will show the call stack if it is available.



Obj/Seg name and hex type	DASD I/O type (short version)	Joins to same in SMTRSTCK file	Object PDIO-Type/Stack-Inx Total	Object PDIO-Type/Stack-Inx Avg Bytes Len	Ob Mi By
*SECTOR I/O 0000	RS	0	55	196459	
*SECTOR I/O 0000	WS	0	404	30790	
ACTVTN PROC REF TBL T	PO	0	43	13716	
ADMIN QTMHHTTP 1AEF	FT	0	1	4096	
ADMIN QTMHHTTP 1AEF	PO	0	1	4096	
AS/400 LiM P FS DIR P	FT	0	2	4096	
ASM STACKS SEG P	FT	0	5	4096	
ASM STACKS SEG P	PO	0	5	10649	
BRAU 0E02	WA	0	6	4096	
BRAU 0E02	WS	0	14	4096	
BRAU 0E90	WA	0	3	4096	
BRAU 0401	FT	0	1	4096	
BRAU 0401	PO	0	1	4096	
BRAU 0401	WA	0	7	4096	
BRAU 0401	WS	0	1	4096	
BRAU 0801	FT	0	3	6826	

Table of Contents | Previous | Next

5.5.33 IO Total by Object/Job-Thread

This report shows the total number of physical disk IO events that occurred by jobs within each object.

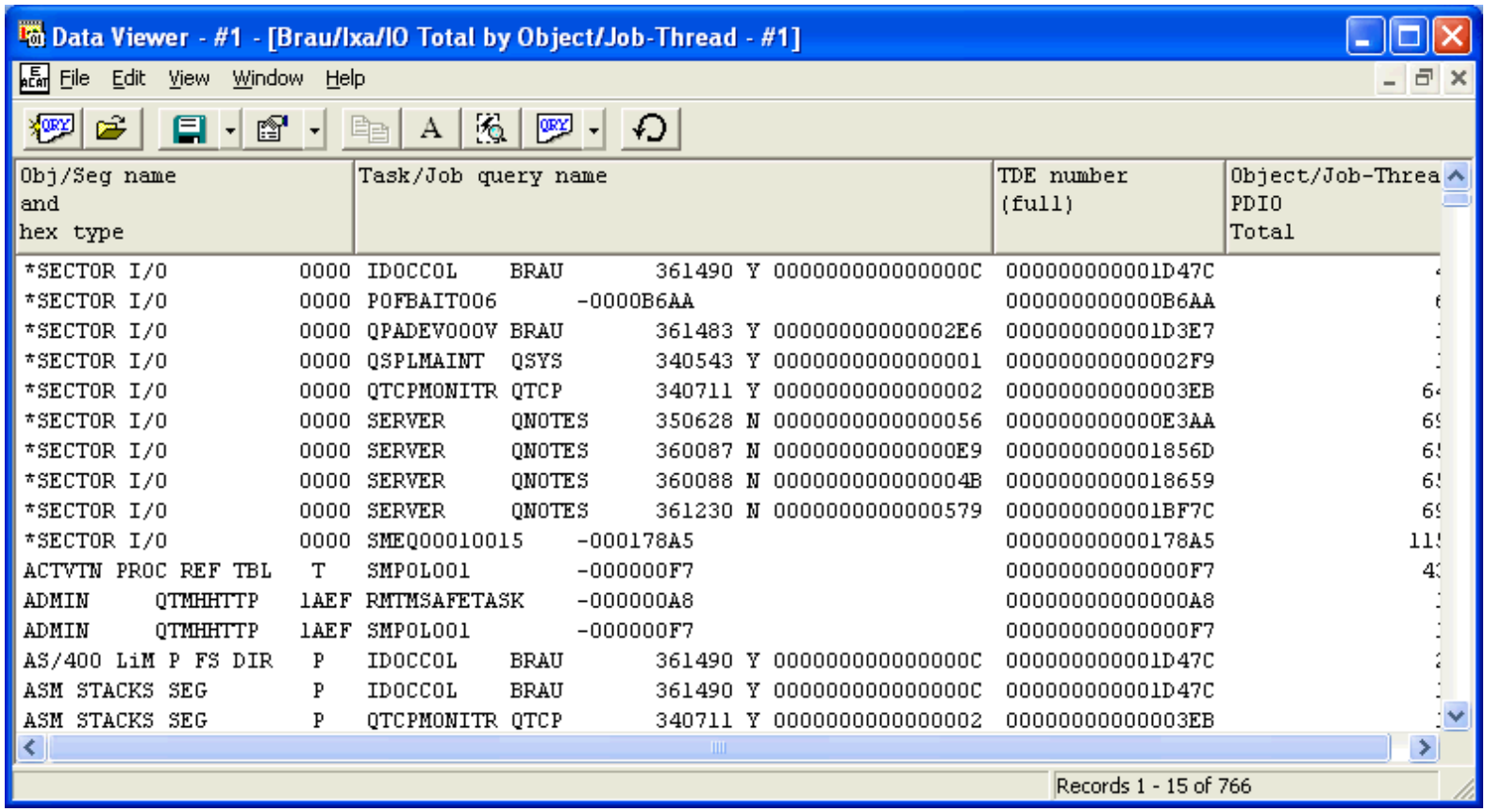
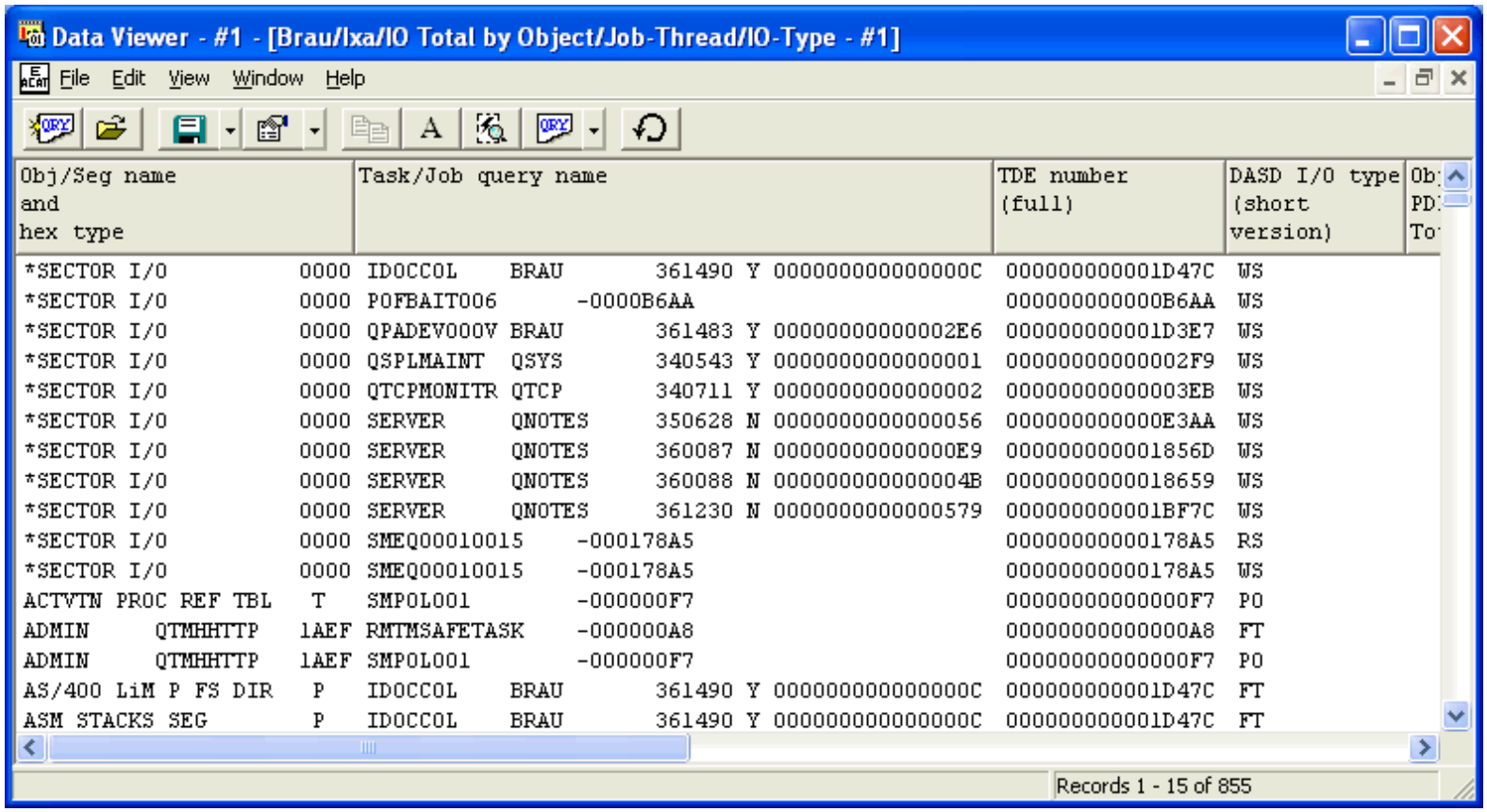


Table of Contents | Previous | Next

5.5.34 IO Total by Object/Job-Thread/IO-Type

This report shows a breakdown of physical disk IO events that occurred by for each IO type for jobs within each object.



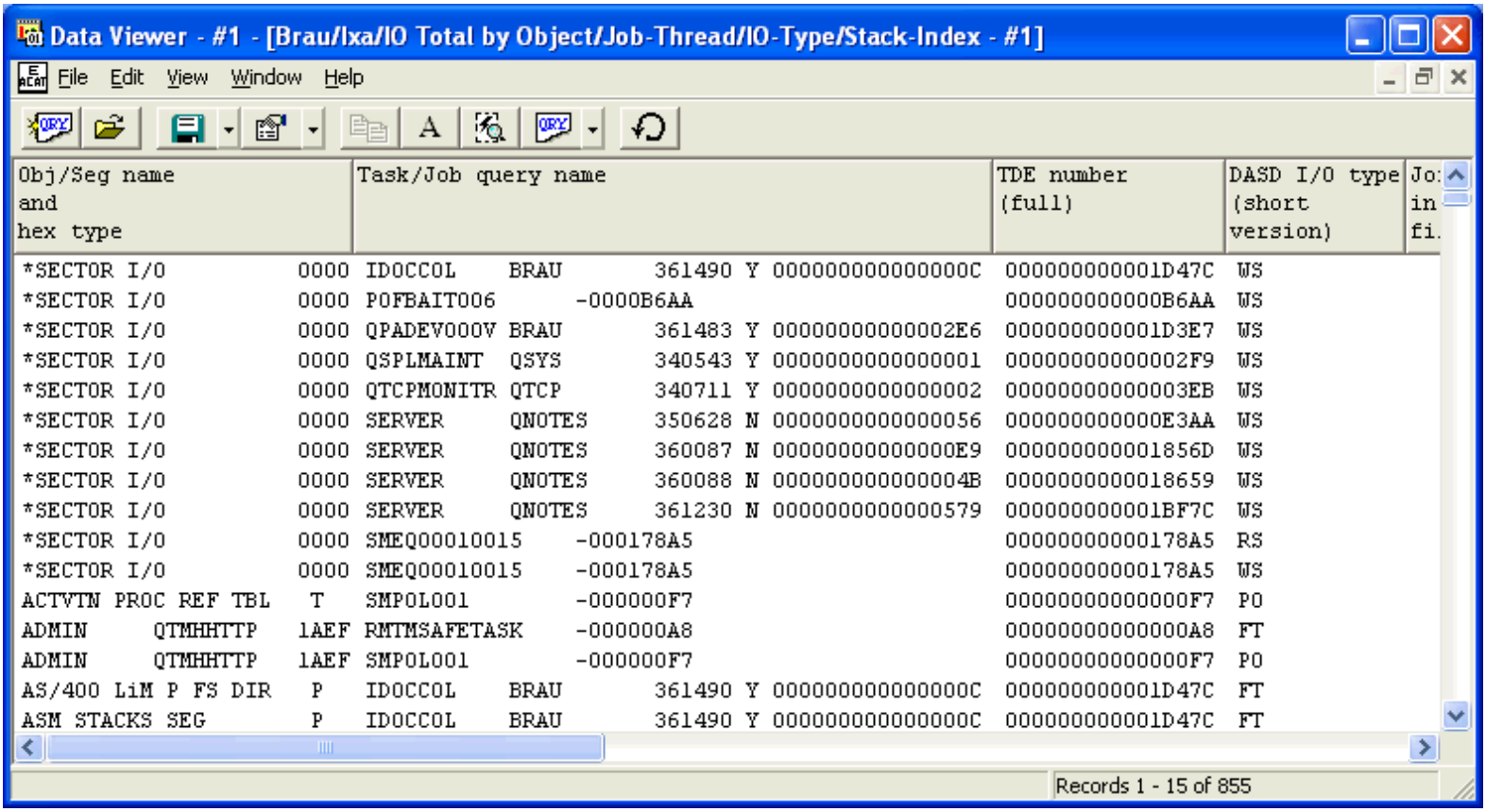
Obj/Seg name and hex type	Task/Job query name	TDE number (full)	DASD I/O type (short version)	Ob PD To
*SECTOR I/O 0000	IDOCCOL BRAU 361490 Y	000000000000000C	000000000001D47C	WS
*SECTOR I/O 0000	POFBAIT006 -0000B6AA		000000000000B6AA	WS
*SECTOR I/O 0000	QPADEV000V BRAU 361483 Y	00000000000002E6	000000000001D3E7	WS
*SECTOR I/O 0000	QSPLMAINT QSYS 340543 Y	0000000000000001	00000000000002F9	WS
*SECTOR I/O 0000	QTCPMONITR QTCP 340711 Y	0000000000000002	00000000000003EB	WS
*SECTOR I/O 0000	SERVER QNOTES 350628 N	0000000000000056	000000000000E3AA	WS
*SECTOR I/O 0000	SERVER QNOTES 360087 N	00000000000000E9	000000000001856D	WS
*SECTOR I/O 0000	SERVER QNOTES 360088 N	000000000000004B	0000000000018659	WS
*SECTOR I/O 0000	SERVER QNOTES 361230 N	0000000000000579	000000000001BF7C	WS
*SECTOR I/O 0000	SMEQ000010015 -000178A5		00000000000178A5	RS
*SECTOR I/O 0000	SMEQ000010015 -000178A5		00000000000178A5	WS
ACTVTN PROC REF TBL T	SMPOL001 -000000F7		00000000000000F7	PO
ADMIN QTMHHTTP lAEF	RMTMSAFETASK -000000A8		00000000000000A8	FT
ADMIN QTMHHTTP lAEF	SMPOL001 -000000F7		00000000000000F7	PO
AS/400 LiM P FS DIR P	IDOCCOL BRAU 361490 Y	000000000000000C	000000000001D47C	FT
ASM STACKS SEG P	IDOCCOL BRAU 361490 Y	000000000000000C	000000000001D47C	FT

Table of Contents | Previous | Next

5.5.35 IO Total by Object/Job-Thread/IO-Type/Stack-Index

This report shows a breakdown of physical disk IO events that occurred by for each IO type for jobs within each object.

If a call stack is available the STCKINDX field will be greater than 0. Double-clicking on a record will show the call stack if it is available.

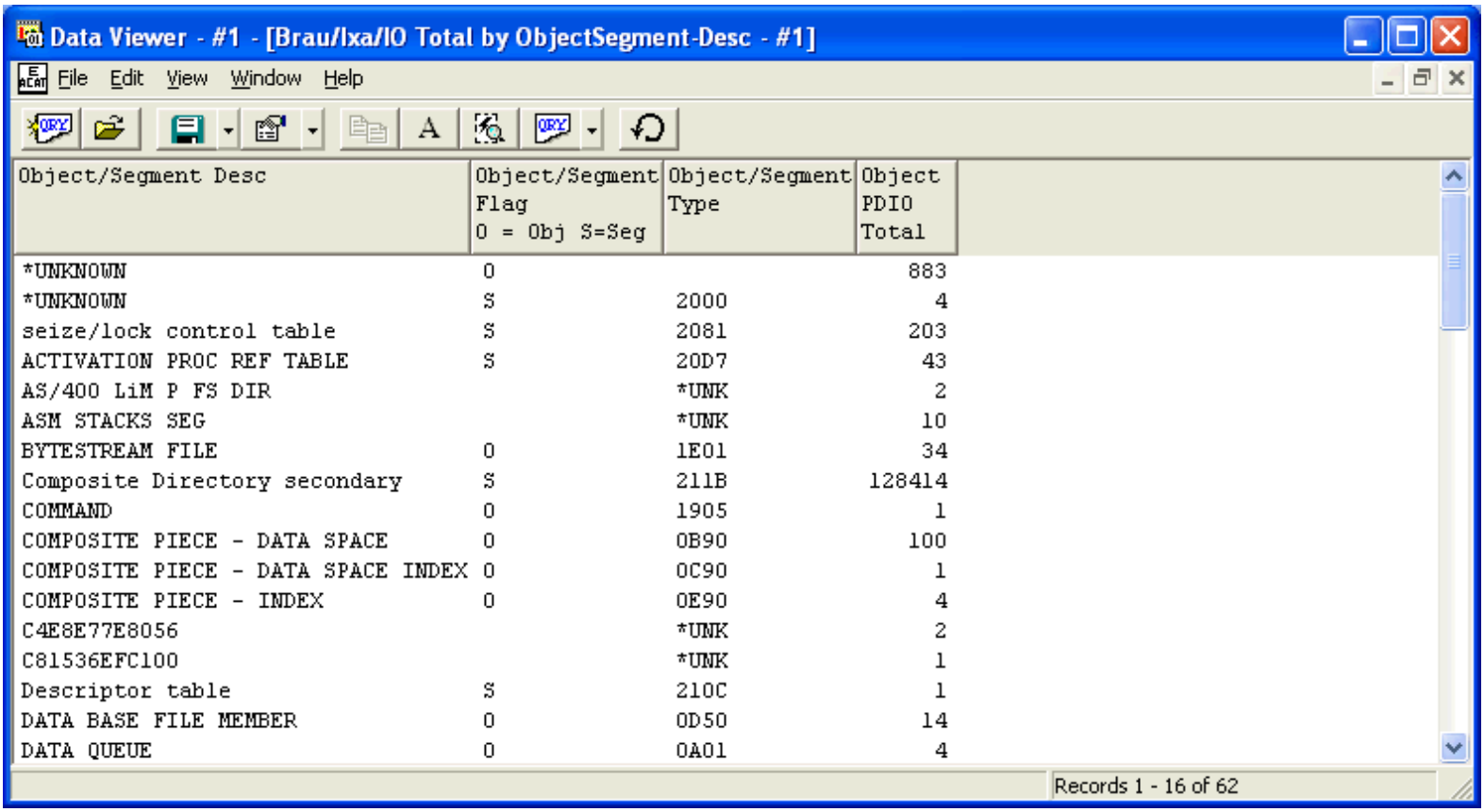


Obj/Seg name and hex type	Task/Job query name	TDE number (full)	DASD I/O type (short version)	Jo: in fi.
*SECTOR I/O 0000	IDOCCOL BRAU 361490 Y 000000000000000C	000000000001D47C	WS	
*SECTOR I/O 0000	POFBAIT006 -0000B6AA	000000000000B6AA	WS	
*SECTOR I/O 0000	QPADEV000V BRAU 361483 Y 00000000000002E6	000000000001D3E7	WS	
*SECTOR I/O 0000	QSPLMAINT QSYS 340543 Y 0000000000000001	00000000000002F9	WS	
*SECTOR I/O 0000	QTCPMONITR QTCP 340711 Y 0000000000000002	00000000000003EB	WS	
*SECTOR I/O 0000	SERVER QNOTES 350628 N 0000000000000056	000000000000E3AA	WS	
*SECTOR I/O 0000	SERVER QNOTES 360087 N 00000000000000E9	000000000001856D	WS	
*SECTOR I/O 0000	SERVER QNOTES 360088 N 000000000000004B	0000000000018659	WS	
*SECTOR I/O 0000	SERVER QNOTES 361230 N 0000000000000579	000000000001BF7C	WS	
*SECTOR I/O 0000	SMEQ000010015 -000178A5	00000000000178A5	RS	
*SECTOR I/O 0000	SMEQ000010015 -000178A5	00000000000178A5	WS	
ACTVTN PROC REF TBL T	SMPOL001 -000000F7	00000000000000F7	PO	
ADMIN QTMHHTTP 1AEF	RMTMSAFETASK -000000A8	00000000000000A8	FT	
ADMIN QTMHHTTP 1AEF	SMPOL001 -000000F7	00000000000000F7	PO	
AS/400 LiM P FS DIR P	IDOCCOL BRAU 361490 Y 000000000000000C	000000000001D47C	FT	
ASM STACKS SEG P	IDOCCOL BRAU 361490 Y 000000000000000C	000000000001D47C	FT	

Table of Contents | Previous | Next

5.5.36 IO Total by ObjectSegment-Desc

This report shows the total number of physical disk IO events that occurred for each object type.



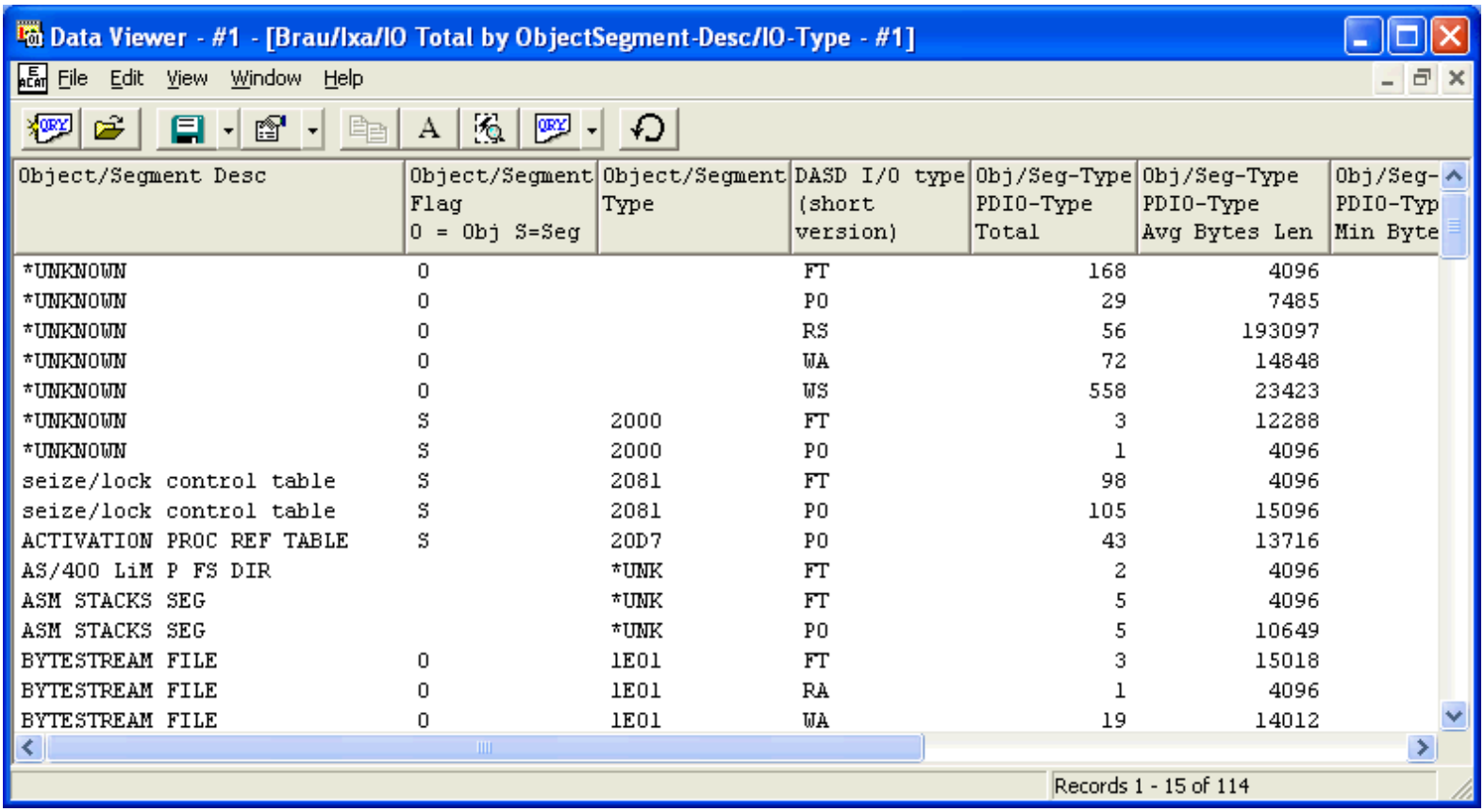
Object/Segment Desc	Object/Segment Flag 0 = Obj S=Seg	Object/Segment Type	Object PDIO Total
*UNKNOWN	0		883
*UNKNOWN	S	2000	4
seize/lock control table	S	2081	203
ACTIVATION PROC REF TABLE	S	20D7	43
AS/400 LiM P FS DIR		*UNK	2
ASM STACKS SEG		*UNK	10
BYTESTREAM FILE	0	1E01	34
Composite Directory secondary	S	211B	128414
COMMAND	0	1905	1
COMPOSITE PIECE - DATA SPACE	0	0B90	100
COMPOSITE PIECE - DATA SPACE INDEX	0	0C90	1
COMPOSITE PIECE - INDEX	0	0E90	4
C4E8E77E8056		*UNK	2
C81536EFC100		*UNK	1
Descriptor table	S	210C	1
DATA BASE FILE MEMBER	0	0D50	14
DATA QUEUE	0	0A01	4

Records 1 - 16 of 62

Table of Contents | Previous | Next

5.5.37 IO Total by ObjectSegment-Desc/IO-Type

This report shows a breakdown of physical disk IO events that occurred for each IO type within object type.



Object/Segment Desc	Object/Segment Flag 0 = Obj S=Seg	Object/Segment Type	DASD I/O type (short version)	Obj/Seg-Type PDIO-Type Total	Obj/Seg-Type PDIO-Type Avg Bytes Len	Obj/Seg- PDIO-Typ Min Byte
*UNKNOWN	0		FT	168	4096	
*UNKNOWN	0		PO	29	7485	
*UNKNOWN	0		RS	56	193097	
*UNKNOWN	0		WA	72	14848	
*UNKNOWN	0		WS	558	23423	
*UNKNOWN	S	2000	FT	3	12288	
*UNKNOWN	S	2000	PO	1	4096	
seize/lock control table	S	2081	FT	98	4096	
seize/lock control table	S	2081	PO	105	15096	
ACTIVATION PROC REF TABLE	S	20D7	PO	43	13716	
AS/400 LiM P FS DIR		*UNK	FT	2	4096	
ASM STACKS SEG		*UNK	FT	5	4096	
ASM STACKS SEG		*UNK	PO	5	10649	
BYTESTREAM FILE	0	1E01	FT	3	15018	
BYTESTREAM FILE	0	1E01	RA	1	4096	
BYTESTREAM FILE	0	1E01	WA	19	14012	

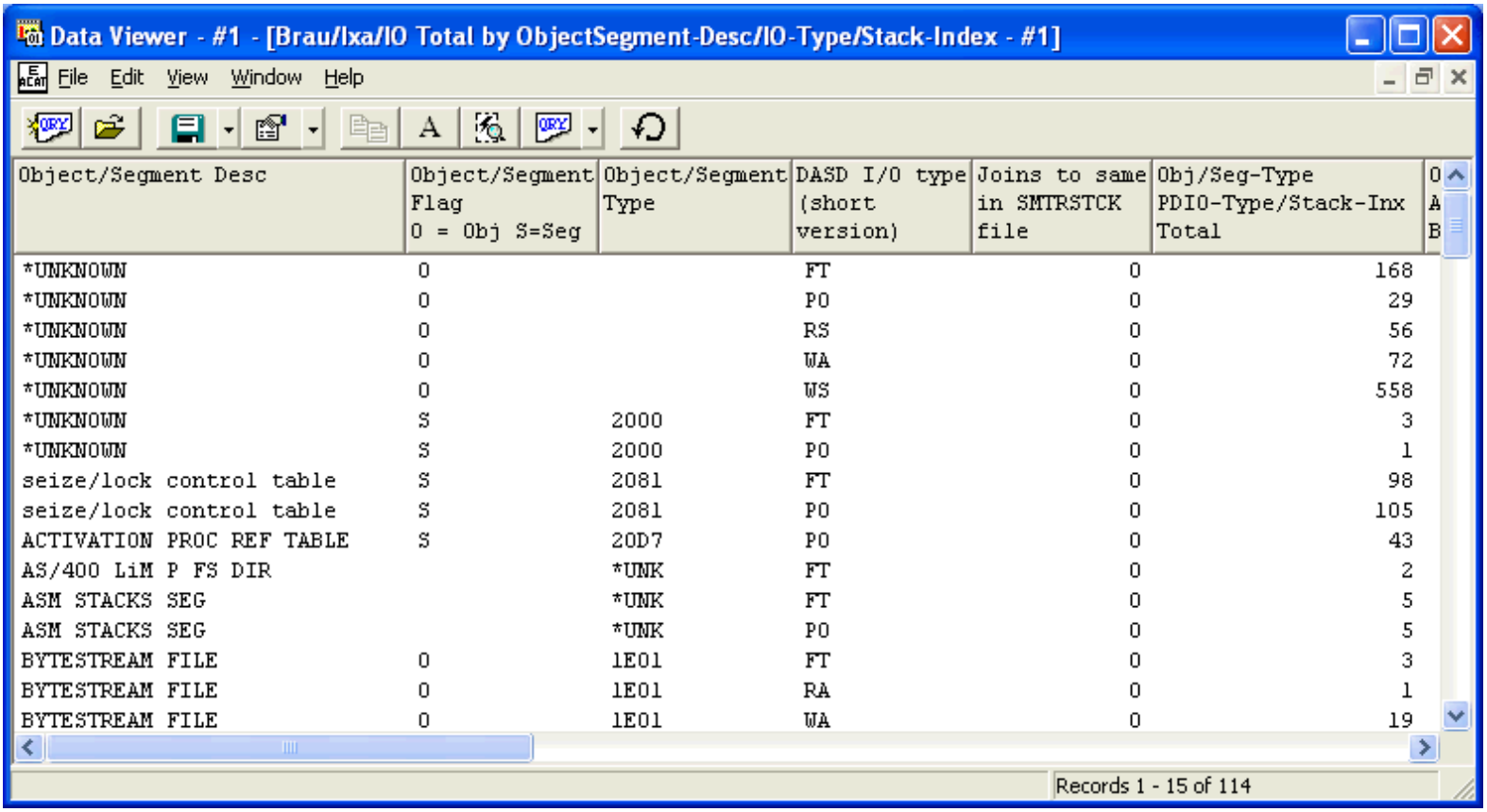
Records 1 - 15 of 114

Table of Contents | Previous | Next

5.5.38 IO Total by ObjectSegment-Desc/IO-Type/Stack-Index

This report shows a breakdown of physical disk IO events that occurred for each IO type within object type.

If a call stack is available the STCKINDX field will be greater than 0. Double-clicking on a record will show the call stack if it is available.

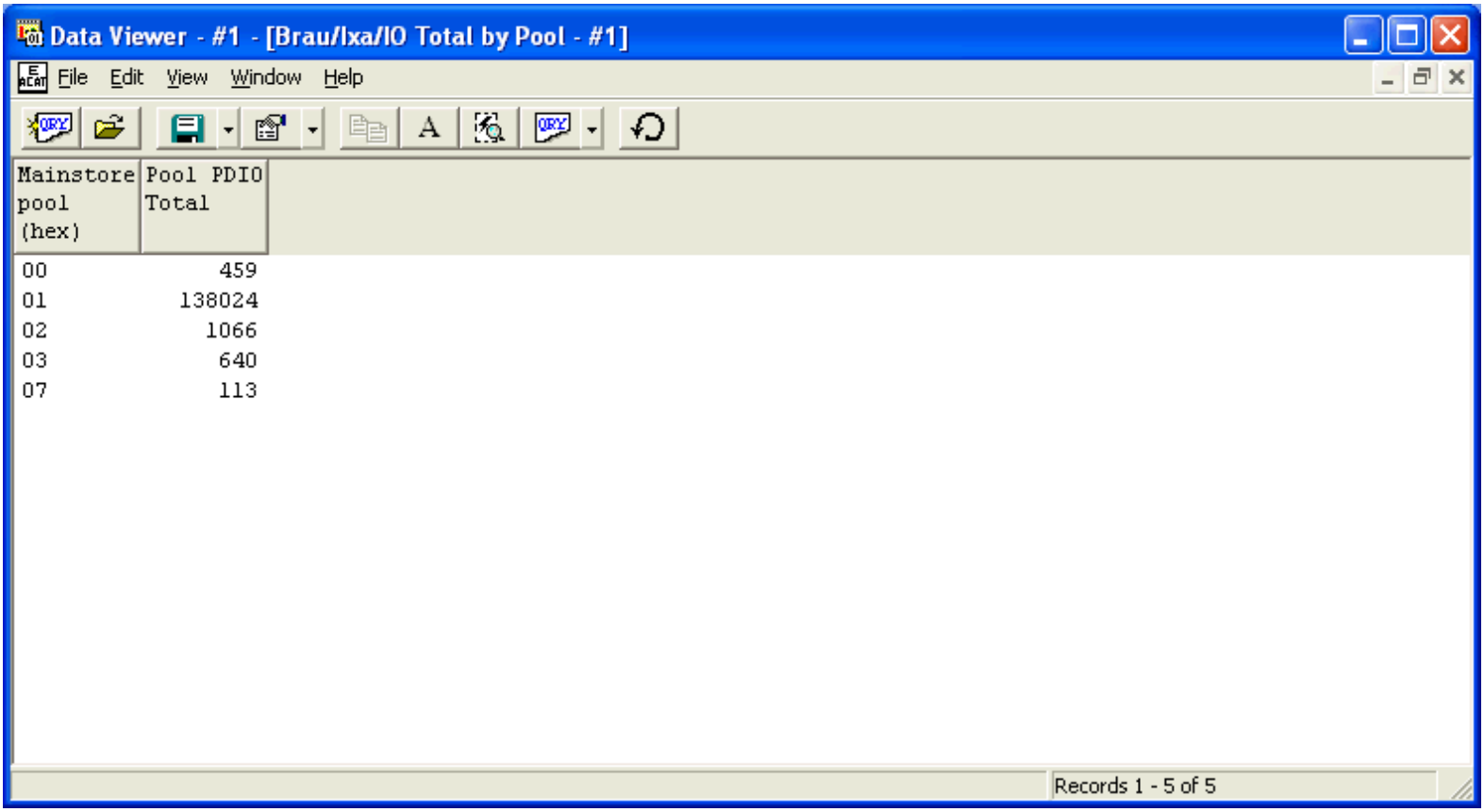


Object/Segment Desc	Object/Segment Flag 0 = Obj S=Seg	Object/Segment Type	DASD I/O type (short version)	Joins to same in SMTRSTCK file	Obj/Seg-Type PDIO-Type/Stack-Inx Total	O A B
*UNKNOWN	0		FT	0	168	
*UNKNOWN	0		PO	0	29	
*UNKNOWN	0		RS	0	56	
*UNKNOWN	0		WA	0	72	
*UNKNOWN	0		WS	0	558	
*UNKNOWN	S	2000	FT	0	3	
*UNKNOWN	S	2000	PO	0	1	
seize/lock control table	S	2081	FT	0	98	
seize/lock control table	S	2081	PO	0	105	
ACTIVATION PROC REF TABLE	S	20D7	PO	0	43	
AS/400 LiM P FS DIR		*UNK	FT	0	2	
ASM STACKS SEG		*UNK	FT	0	5	
ASM STACKS SEG		*UNK	PO	0	5	
BYTESTREAM FILE	0	1E01	FT	0	3	
BYTESTREAM FILE	0	1E01	RA	0	1	
BYTESTREAM FILE	0	1E01	WA	0	19	

Table of Contents | Previous | Next

5.5.39 IO Total by Pool

This report shows the total number of physical disk IO events that occurred for each mainstorage pool within the collection. The pool is listed as a hex number.



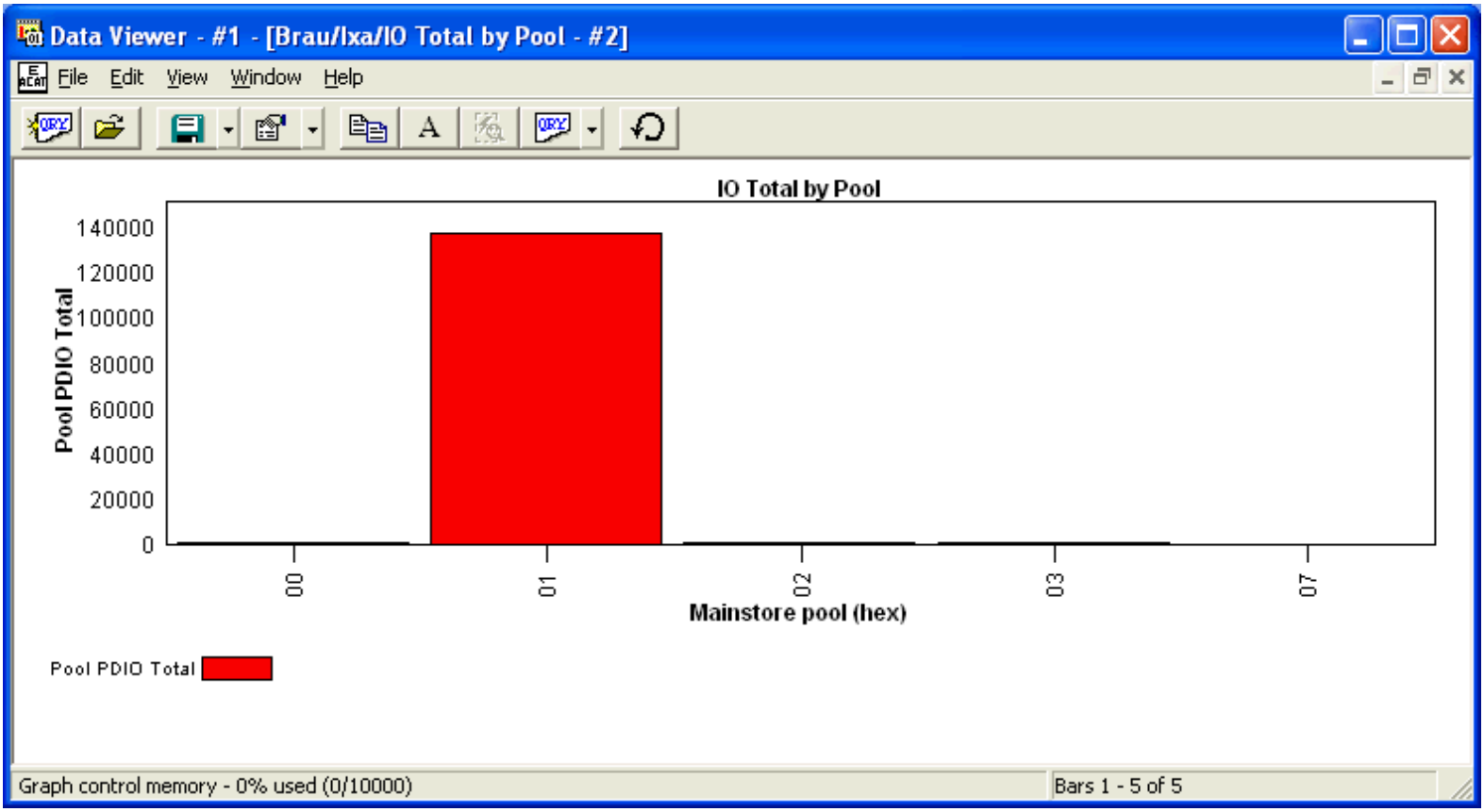
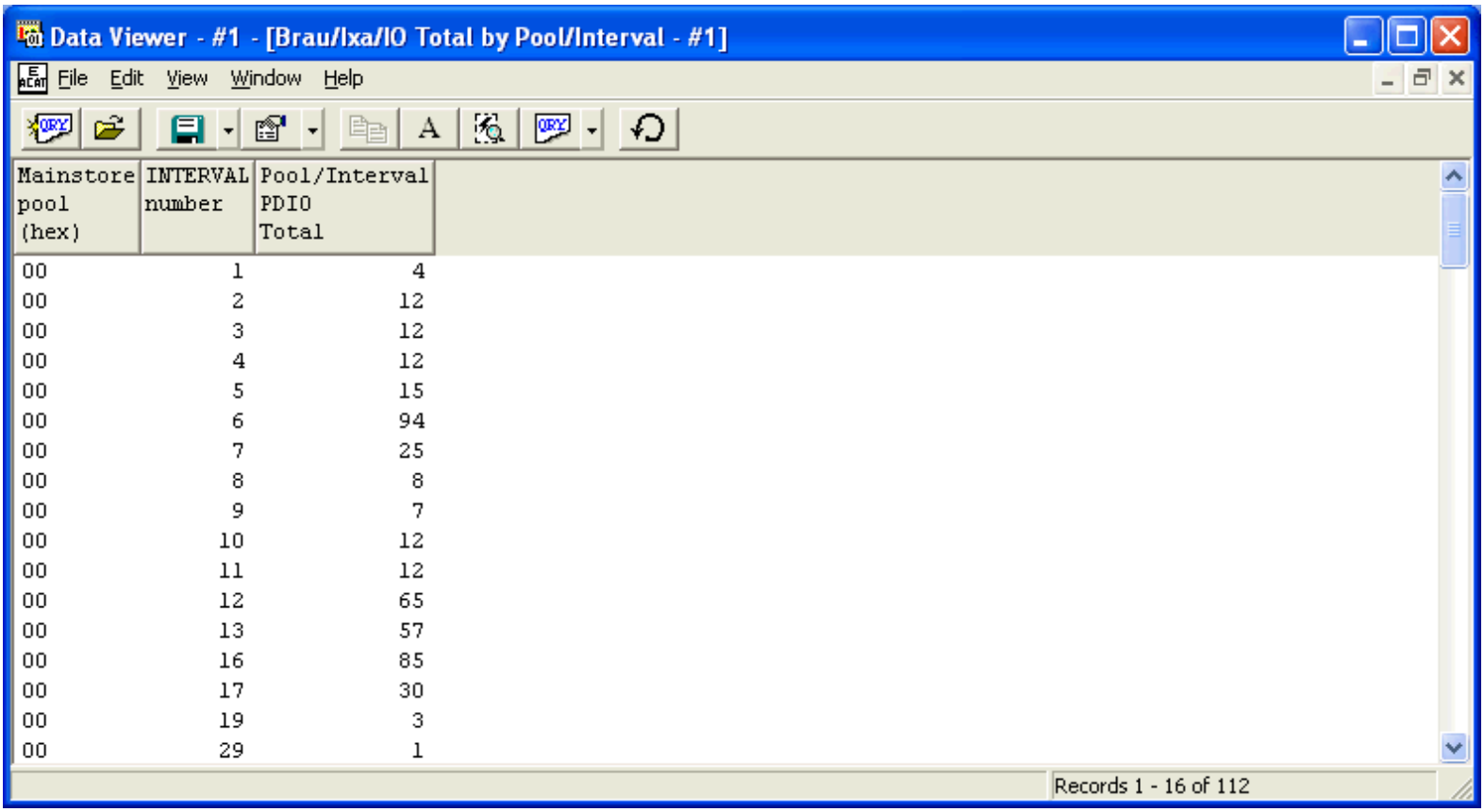

Data Viewer - #1 - [Brau/Ixa/IO Total by Pool - #2]
File Edit View Window Help
IO Total by Pool
Pool PDIO Total
140000
120000
100000
80000
60000
40000
20000
0
00
01
02
03
07
Mainstore pool (hex)
Pool PDIO Total
Graph control memory - 0% used (0/10000)
Bars 1 - 5 of 5

Table of Contents | Previous | Next

5.5.40 IO Total by Pool/Interval

This report shows the total number of physical disk IO events that occurred for each interval within mainstorage pools.

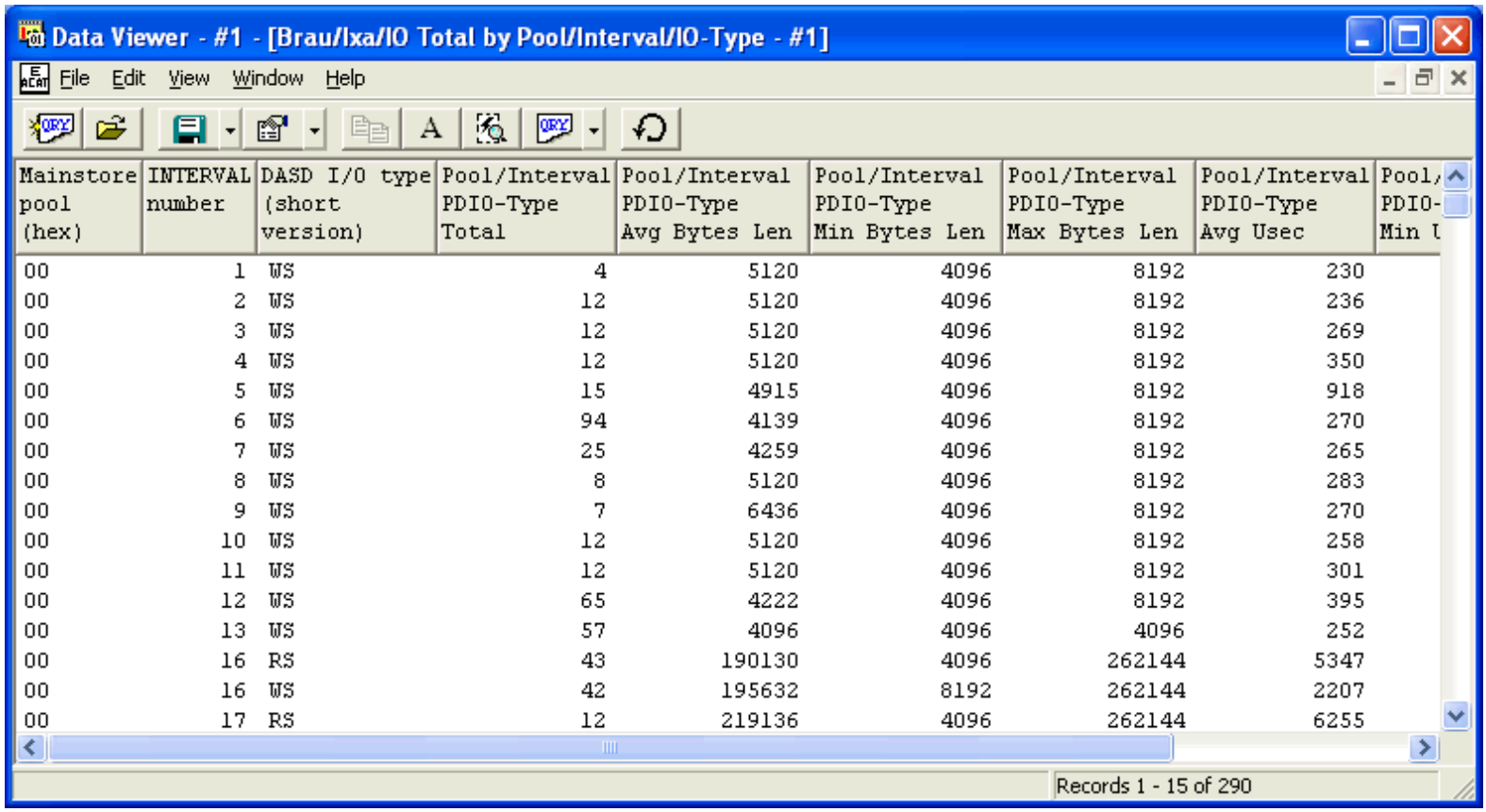


Mainstore pool (hex)	INTERVAL number	Pool/Interval PDIO Total
00	1	4
00	2	12
00	3	12
00	4	12
00	5	15
00	6	94
00	7	25
00	8	8
00	9	7
00	10	12
00	11	12
00	12	65
00	13	57
00	16	85
00	17	30
00	19	3
00	29	1

Table of Contents | Previous | Next

5.5.41 IO Total by Pool/Interval/IO-Type

This report shows a breakdown of physical disk IO events that occurred for each IO type by interval within mainstorage pools.

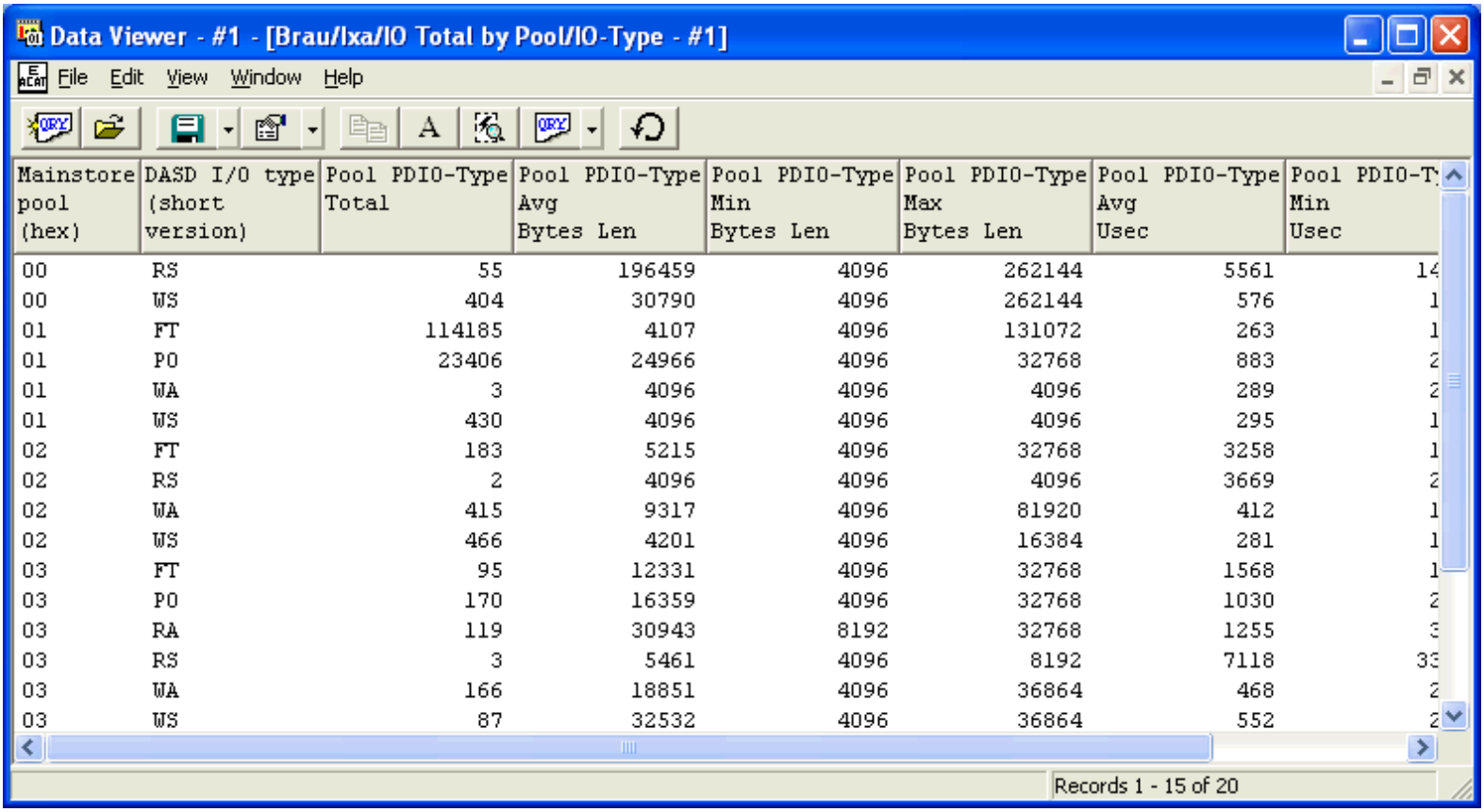


Mainstore pool (hex)	INTERVAL number	DASD I/O type (short version)	Pool/Interval PDIO-Type Total	Pool/Interval PDIO-Type Avg Bytes Len	Pool/Interval PDIO-Type Min Bytes Len	Pool/Interval PDIO-Type Max Bytes Len	Pool/Interval PDIO-Type Avg Usec	Pool/ PDIO- Min
00	1	WS	4	5120	4096	8192	230	
00	2	WS	12	5120	4096	8192	236	
00	3	WS	12	5120	4096	8192	269	
00	4	WS	12	5120	4096	8192	350	
00	5	WS	15	4915	4096	8192	918	
00	6	WS	94	4139	4096	8192	270	
00	7	WS	25	4259	4096	8192	265	
00	8	WS	8	5120	4096	8192	283	
00	9	WS	7	6436	4096	8192	270	
00	10	WS	12	5120	4096	8192	258	
00	11	WS	12	5120	4096	8192	301	
00	12	WS	65	4222	4096	8192	395	
00	13	WS	57	4096	4096	4096	252	
00	16	RS	43	190130	4096	262144	5347	
00	16	WS	42	195632	8192	262144	2207	
00	17	RS	12	219136	4096	262144	6255	

Table of Contents | Previous | Next

5.5.42 IO Total by Pool/IO-Type

This report shows a breakdown of physical disk IO events that occurred for each IO type within mainstorage pool. The pool is listed in hex format.

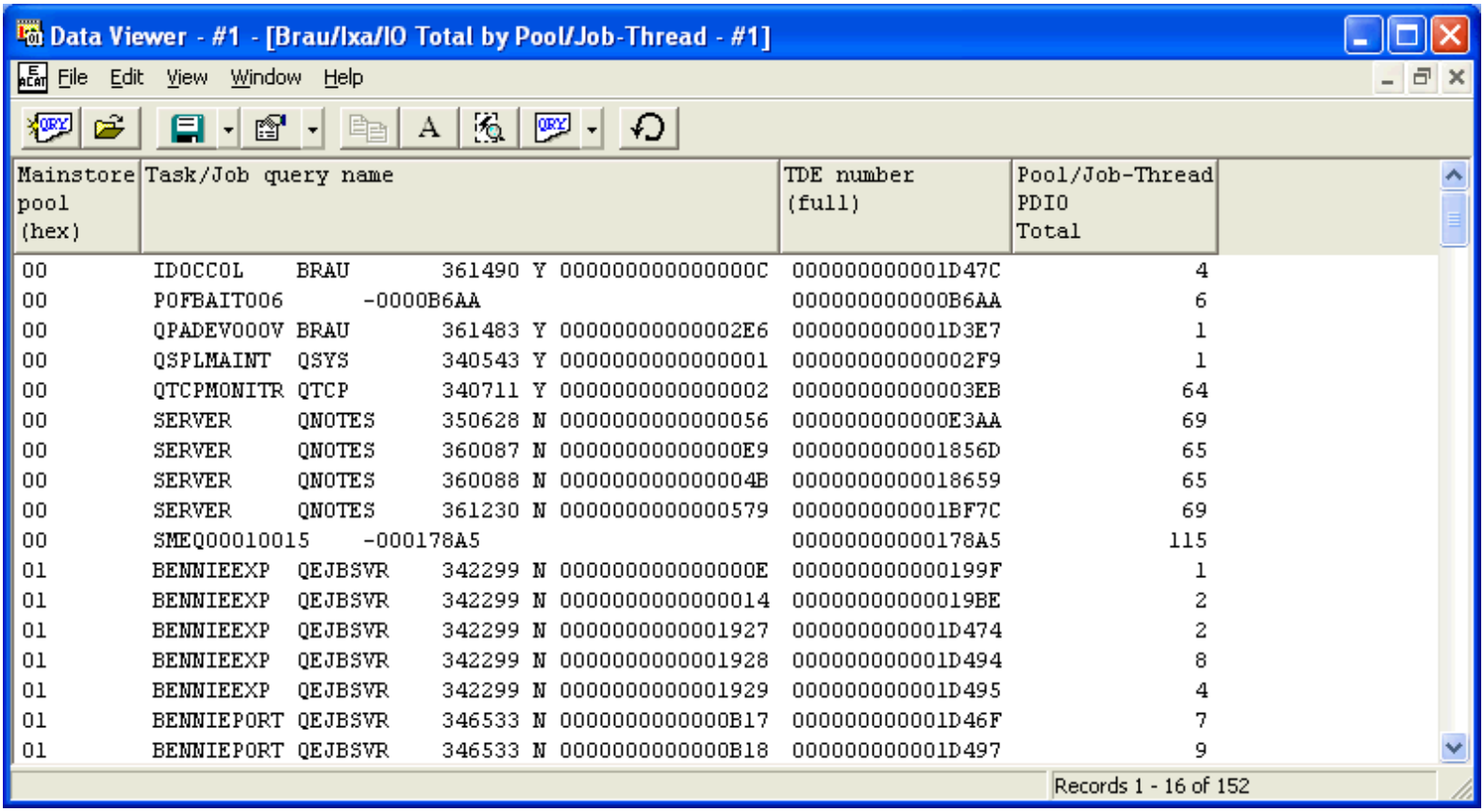


Mainstore pool (hex)	DASD I/O type (short version)	Pool PDIO-Type Total	Pool PDIO-Type Avg Bytes Len	Pool PDIO-Type Min Bytes Len	Pool PDIO-Type Max Bytes Len	Pool PDIO-Type Avg Usec	Pool PDIO-Ty Min Usec
00	RS	55	196459	4096	262144	5561	14
00	WS	404	30790	4096	262144	576	1
01	FT	114185	4107	4096	131072	263	1
01	PO	23406	24966	4096	32768	883	2
01	WA	3	4096	4096	4096	289	2
01	WS	430	4096	4096	4096	295	1
02	FT	183	5215	4096	32768	3258	1
02	RS	2	4096	4096	4096	3669	2
02	WA	415	9317	4096	81920	412	1
02	WS	466	4201	4096	16384	281	1
03	FT	95	12331	4096	32768	1568	1
03	PO	170	16359	4096	32768	1030	2
03	RA	119	30943	8192	32768	1255	3
03	RS	3	5461	4096	8192	7118	33
03	WA	166	18851	4096	36864	468	2
03	WS	87	32532	4096	36864	552	2

Table of Contents | Previous | Next

5.5.43 IO Total by Pool/Job-Thread

This report shows the total number of physical disk IO events that occurred for each job within mainstorage pool. The pool id is listed in hex format.



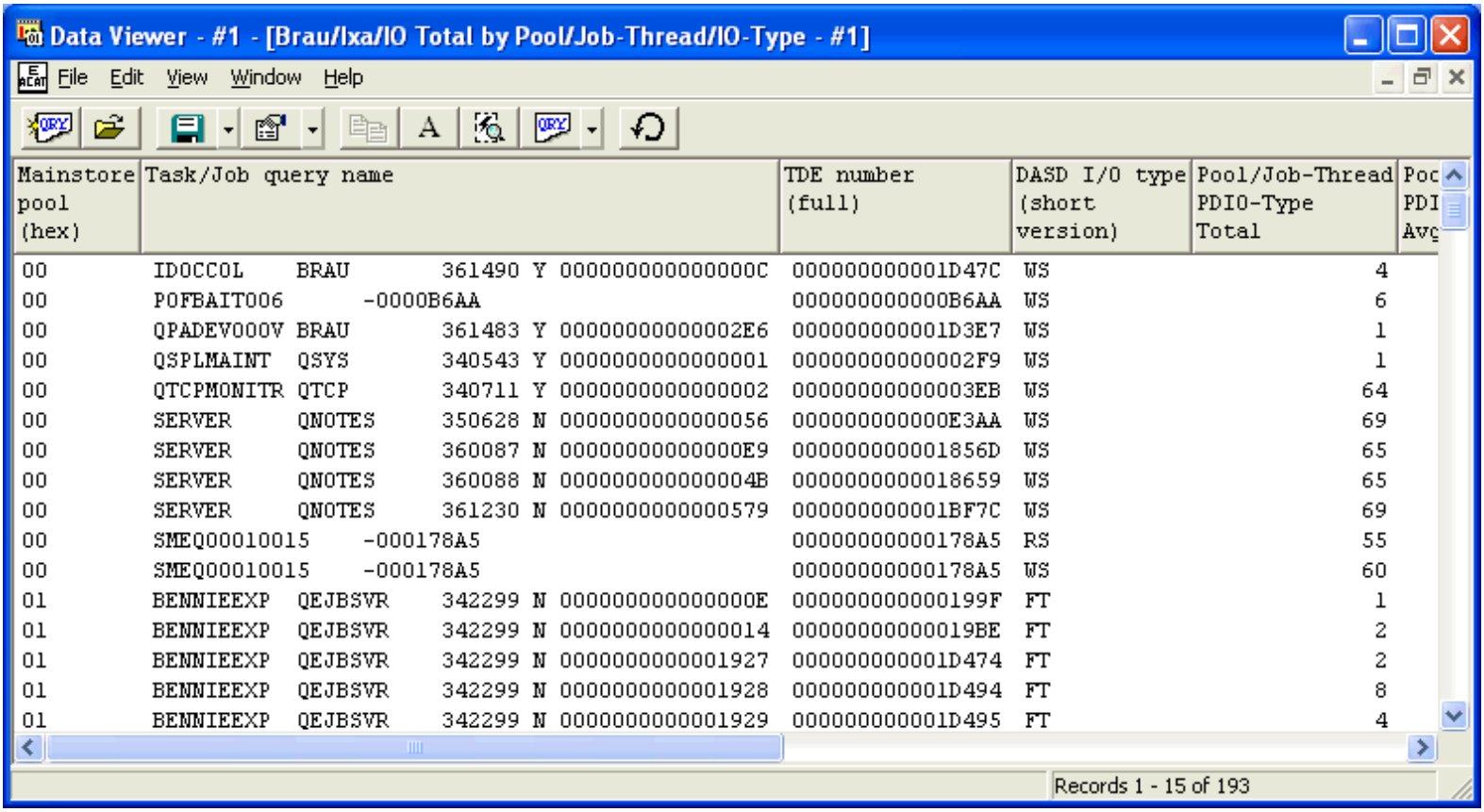
Mainstore pool (hex)	Task/Job query name				TDE number (full)	Pool/Job-Thread PDIO Total
00	IDOCCOL	BRAU	361490	Y 000000000000000C	000000000001D47C	4
00	POFBAIT006	-0000B6AA			000000000000B6AA	6
00	QPADEV000V	BRAU	361483	Y 00000000000002E6	000000000001D3E7	1
00	QSPLMAINT	QSYS	340543	Y 0000000000000001	00000000000002F9	1
00	QTCPMONITR	QTCP	340711	Y 0000000000000002	00000000000003EB	64
00	SERVER	QNOTES	350628	N 0000000000000056	000000000000E3AA	69
00	SERVER	QNOTES	360087	N 00000000000000E9	000000000001856D	65
00	SERVER	QNOTES	360088	N 000000000000004B	0000000000018659	65
00	SERVER	QNOTES	361230	N 0000000000000579	000000000001BF7C	69
00	SMEQ000010015	-000178A5			00000000000178A5	115
01	BENNIEEXP	QEJBSVR	342299	N 000000000000000E	000000000000199F	1
01	BENNIEEXP	QEJBSVR	342299	N 0000000000000014	00000000000019BE	2
01	BENNIEEXP	QEJBSVR	342299	N 0000000000001927	000000000001D474	2
01	BENNIEEXP	QEJBSVR	342299	N 0000000000001928	000000000001D494	8
01	BENNIEEXP	QEJBSVR	342299	N 0000000000001929	000000000001D495	4
01	BENNIEPORT	QEJBSVR	346533	N 0000000000000B17	000000000001D46F	7
01	BENNIEPORT	QEJBSVR	346533	N 0000000000000B18	000000000001D497	9

Records 1 - 16 of 152

Table of Contents | Previous | Next

5.5.44 IO Total by Pool/Job-Thread/IO-Type

This report shows a breakdown of physical disk IO events that occurred for each IO type by job within mainstorage pool. Pool id is listed in hex format.

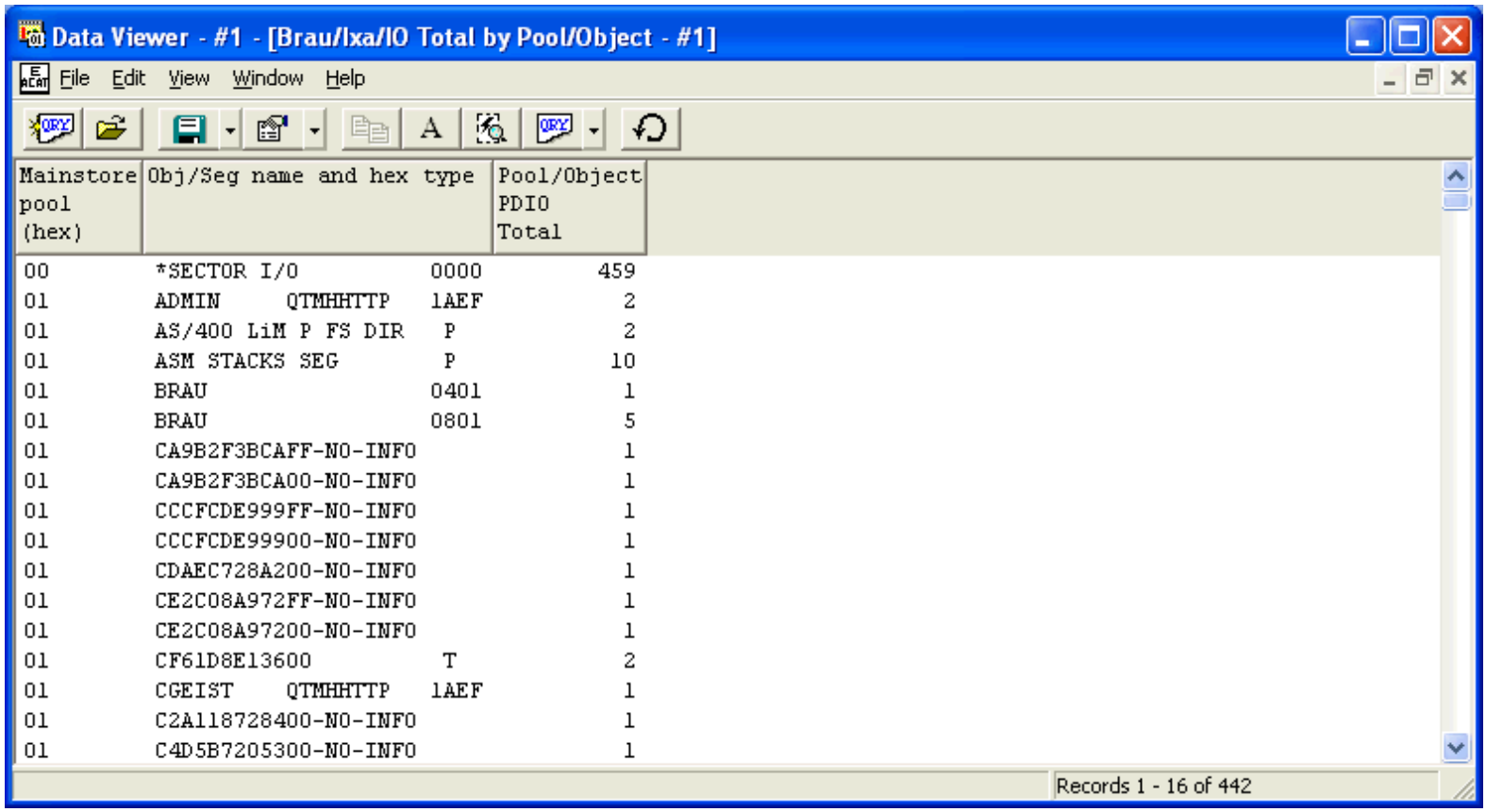


Mainstore pool (hex)	Task/Job query name				TDE number (full)	DASD I/O type (short version)	Pool/Job-Thread PDIO-Type Total	Poc PDI Avg
00	IDOCCOL	BRAU	361490	Y 000000000000000C	000000000001D47C	WS	4	
00	POFBAIT006	-0000B6AA			000000000000B6AA	WS	6	
00	QPADEV000V	BRAU	361483	Y 00000000000002E6	000000000001D3E7	WS	1	
00	QSPLMAINT	QSYS	340543	Y 0000000000000001	00000000000002F9	WS	1	
00	QTCPMONITR	QTCP	340711	Y 0000000000000002	00000000000003EB	WS	64	
00	SERVER	QNOTES	350628	N 0000000000000056	000000000000E3AA	WS	69	
00	SERVER	QNOTES	360087	N 00000000000000E9	000000000001856D	WS	65	
00	SERVER	QNOTES	360088	N 000000000000004B	0000000000018659	WS	65	
00	SERVER	QNOTES	361230	N 0000000000000579	000000000001BF7C	WS	69	
00	SMEQ000010015	-000178A5			00000000000178A5	RS	55	
00	SMEQ000010015	-000178A5			00000000000178A5	WS	60	
01	BENNIEEXP	QEJBSVR	342299	N 000000000000000E	0000000000000199F	FT	1	
01	BENNIEEXP	QEJBSVR	342299	N 0000000000000014	00000000000019BE	FT	2	
01	BENNIEEXP	QEJBSVR	342299	N 0000000000001927	000000000001D474	FT	2	
01	BENNIEEXP	QEJBSVR	342299	N 0000000000001928	000000000001D494	FT	8	
01	BENNIEEXP	QEJBSVR	342299	N 0000000000001929	000000000001D495	FT	4	

Table of Contents | Previous | Next

5.5.45 IO Total by Pool/Object

This report shows the total number of physical disk IO events that occurred by object within mainstorage pool

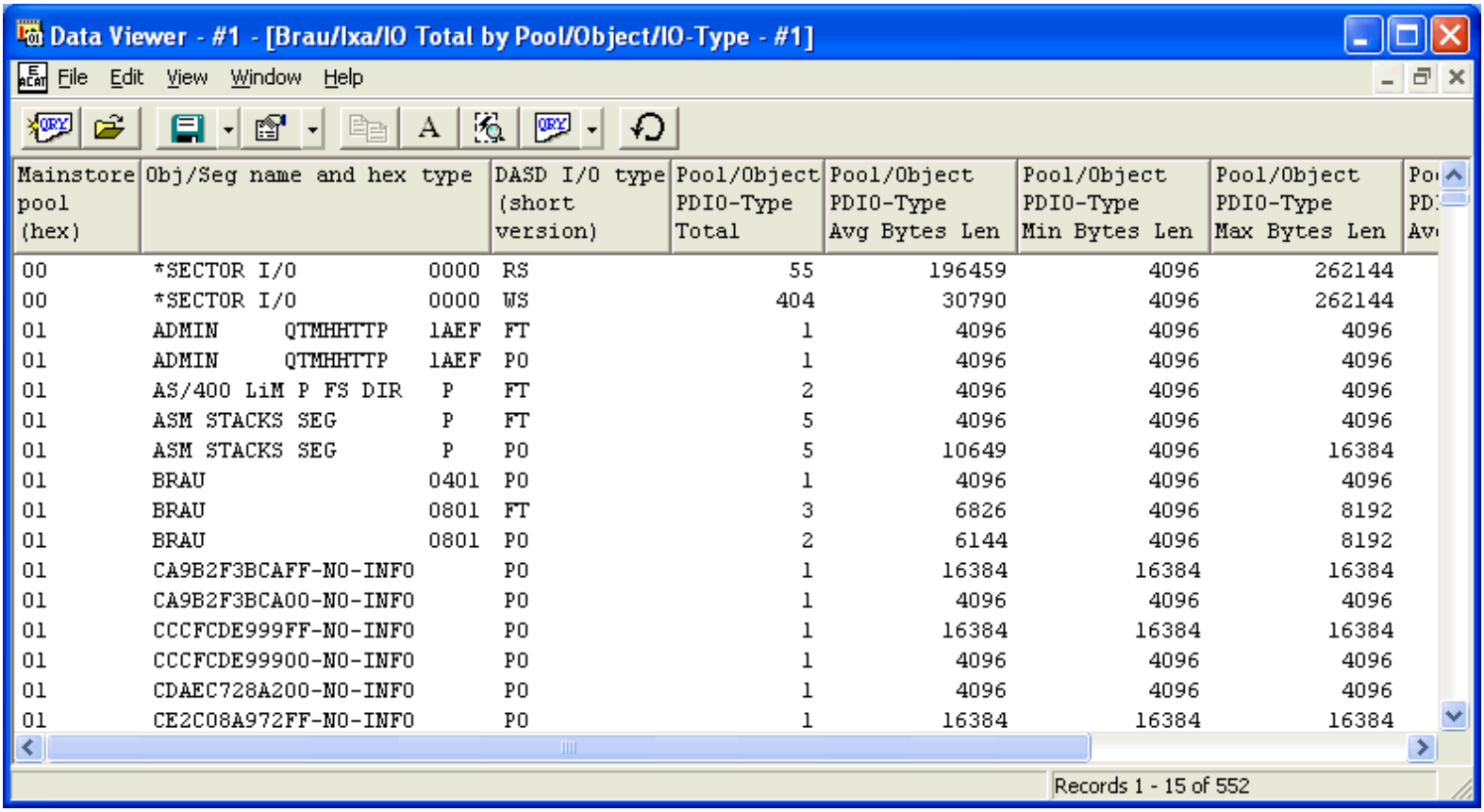


Mainstore pool (hex)	Obj/Seg name and hex type	Pool/Object PDIO Total
00	*SECTOR I/O 0000	459
01	ADMIN QTMHHTTP 1AEF	2
01	AS/400 LiM P FS DIR P	2
01	ASM STACKS SEG P	10
01	BRAU 0401	1
01	BRAU 0801	5
01	CA9B2F3BCAFF-N0-INF0	1
01	CA9B2F3BCA00-N0-INF0	1
01	CCCFCDE999FF-N0-INF0	1
01	CCCFCDE99900-N0-INF0	1
01	CDAEC728A200-N0-INF0	1
01	CE2C08A972FF-N0-INF0	1
01	CE2C08A97200-N0-INF0	1
01	CF61D8E13600 T	2
01	CGEIST QTMHHTTP 1AEF	1
01	C2A118728400-N0-INF0	1
01	C4D5B7205300-N0-INF0	1

Table of Contents | Previous | Next

5.5.46 IO Total by Pool/Object/IO-Type

This report shows a breakdown of physical disk IO events that occurred by IO type within object within mainstorage pool. Pool id is listed in hex format.



Mainstore pool (hex)	Obj/Seg name and hex type		DASD I/O type (short version)	Pool/Object PDIO-Type Total	Pool/Object PDIO-Type Avg Bytes Len	Pool/Object PDIO-Type Min Bytes Len	Pool/Object PDIO-Type Max Bytes Len	Po PD Av
00	*SECTOR I/O	0000	RS	55	196459	4096	262144	
00	*SECTOR I/O	0000	WS	404	30790	4096	262144	
01	ADMIN QTMHHTTP	1AEF	FT	1	4096	4096	4096	
01	ADMIN QTMHHTTP	1AEF	PO	1	4096	4096	4096	
01	AS/400 LiM P FS DIR	P	FT	2	4096	4096	4096	
01	ASM STACKS SEG	P	FT	5	4096	4096	4096	
01	ASM STACKS SEG	P	PO	5	10649	4096	16384	
01	BRAU	0401	PO	1	4096	4096	4096	
01	BRAU	0801	FT	3	6826	4096	8192	
01	BRAU	0801	PO	2	6144	4096	8192	
01	CA9B2F3BCAFF-NO-INFO		PO	1	16384	16384	16384	
01	CA9B2F3BCA00-NO-INFO		PO	1	4096	4096	4096	
01	CCCFCDE999FF-NO-INFO		PO	1	16384	16384	16384	
01	CCCFCDE99900-NO-INFO		PO	1	4096	4096	4096	
01	CDAEC728A200-NO-INFO		PO	1	4096	4096	4096	
01	CE2C08A972FF-NO-INFO		PO	1	16384	16384	16384	

Table of Contents | Previous | Next

5.5.47 IO Total by Program

This report shows the total number of physical disk IO events that occurred for each program in the collection.

Note: In order for program names to be provided, *MIENTRY and *MIEXIT events must be included in the PEX collection.

Table of Contents | Previous | Next

5.5.48 IO Total by Program/IO-Type

This report shows a breakdown of physical disk IO events that occurred for each IO type within program in the collection.

Note: In order for program names to be provided, *MIENTRY and *MIEXIT events must be included in the PEX collection.

Table of Contents | Previous | Next

5.5.49 IO Total by Program/Object

This report shows the total number of physical disk IO events that occurred for each IO type within object within program in the collection.

Note: In order for program names to be provided, *MIENTRY and *MIEXIT events must be included in the PEX collection.

Table of Contents | Previous | Next

5.5.50 IO Total by Program/Object/IO-Type

This report shows the total number of physical disk IO events that occurred over the data analyzed.

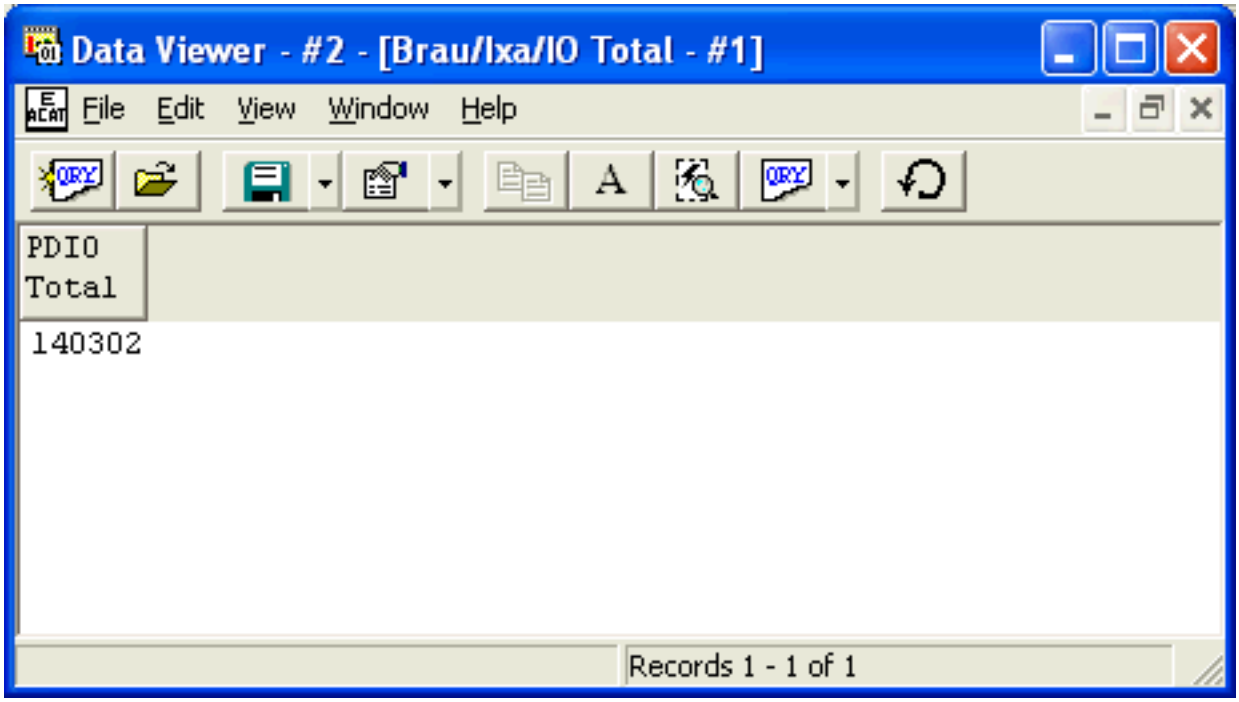
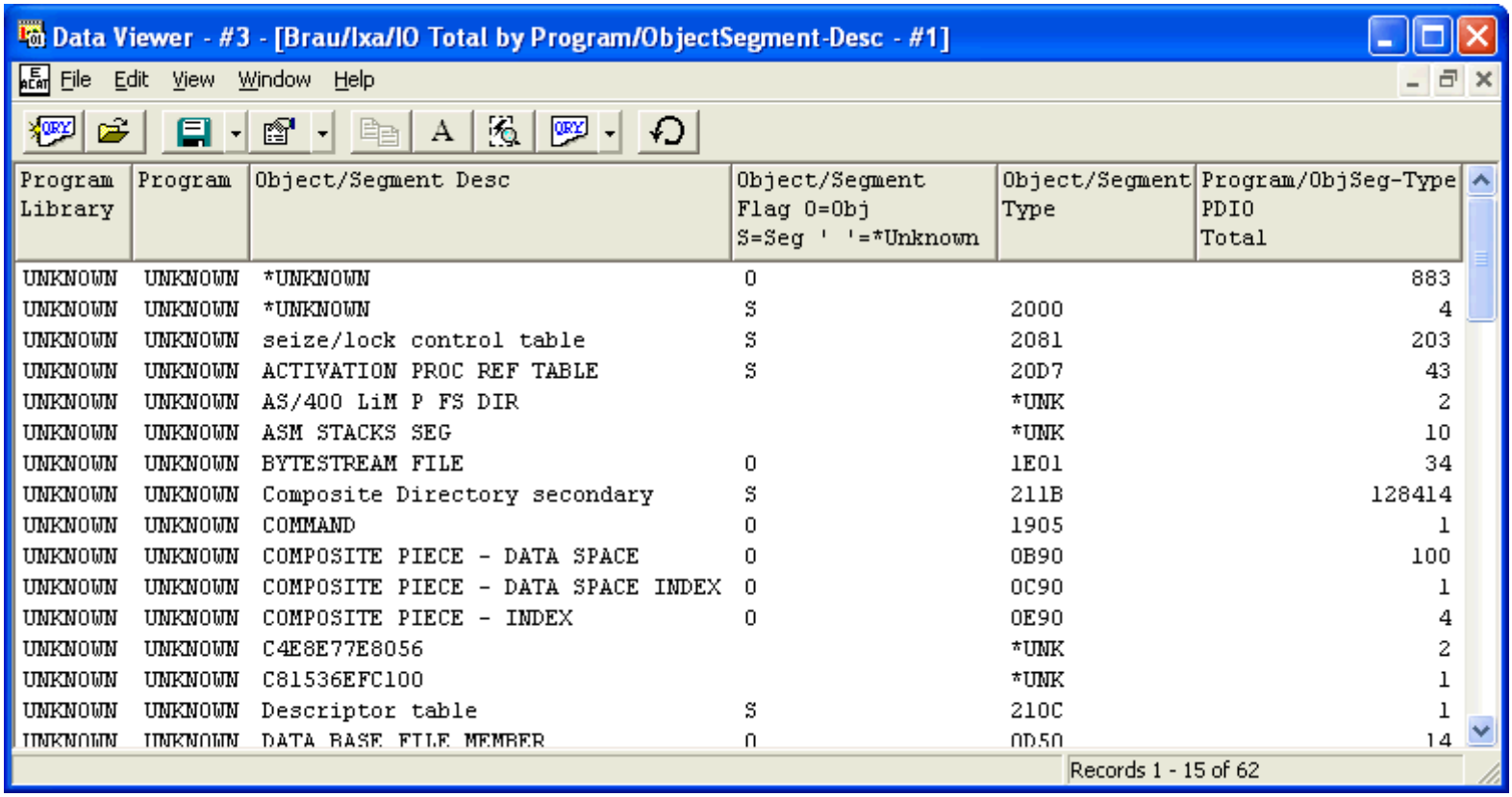


Table of Contents | Previous | Next

5.5.51 IO Total by Program/ObjectSegment-Desc

This report shows the total number of physical disk IO events that occurred for each object type by program.

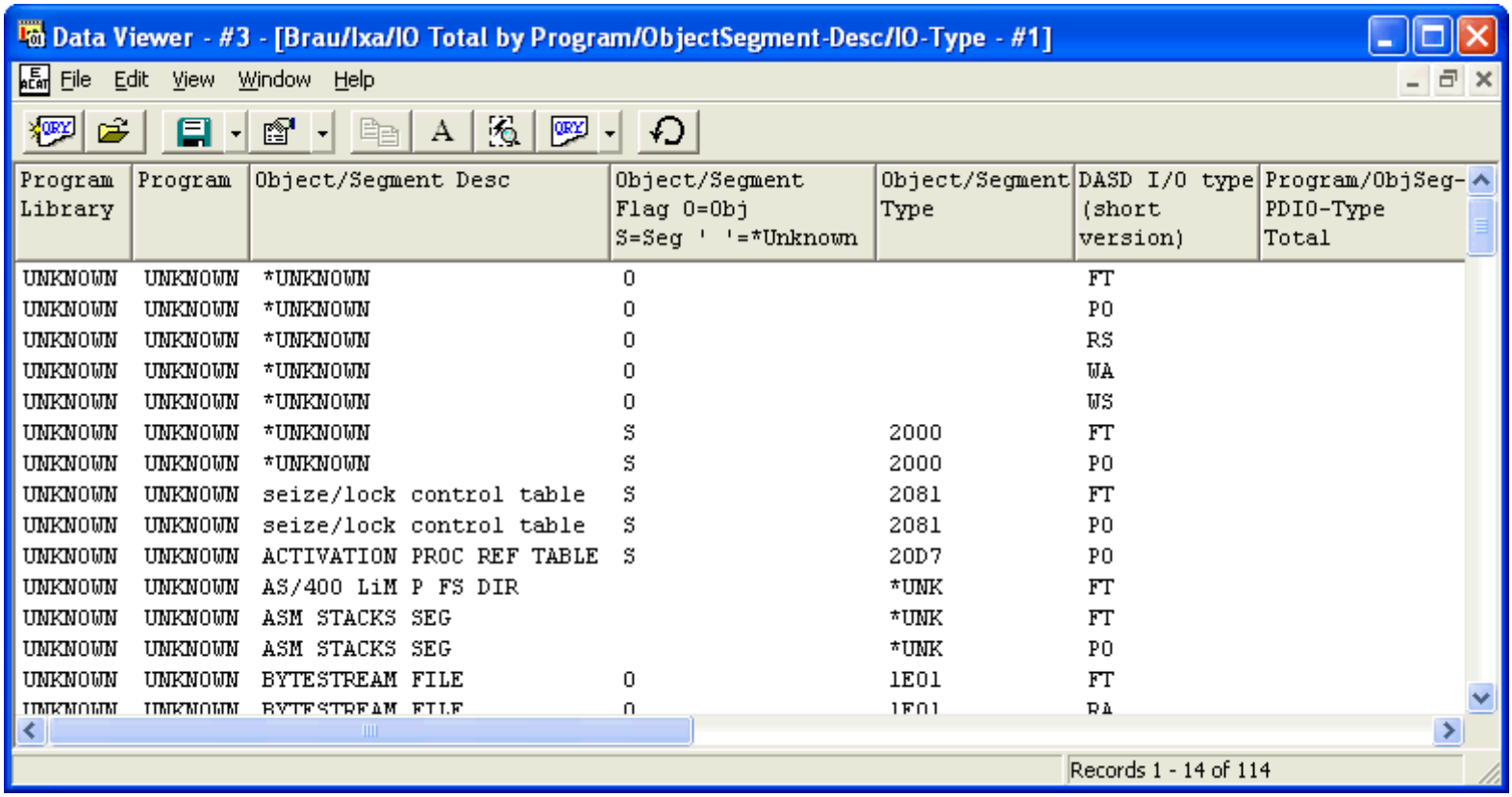


Program Library	Program	Object/Segment Desc	Object/Segment Flag 0=Obj S=Seg ' '=*Unknown	Object/Segment Type	Program/ObjSeg-Type PDIO Total
UNKNOWN	UNKNOWN	*UNKNOWN	0		883
UNKNOWN	UNKNOWN	*UNKNOWN	S	2000	4
UNKNOWN	UNKNOWN	seize/lock control table	S	2081	203
UNKNOWN	UNKNOWN	ACTIVATION PROC REF TABLE	S	20D7	43
UNKNOWN	UNKNOWN	AS/400 LiM P FS DIR		*UNK	2
UNKNOWN	UNKNOWN	ASM STACKS SEG		*UNK	10
UNKNOWN	UNKNOWN	BYTESTREAM FILE	0	1E01	34
UNKNOWN	UNKNOWN	Composite Directory secondary	S	211B	128414
UNKNOWN	UNKNOWN	COMMAND	0	1905	1
UNKNOWN	UNKNOWN	COMPOSITE PIECE - DATA SPACE	0	0B90	100
UNKNOWN	UNKNOWN	COMPOSITE PIECE - DATA SPACE INDEX	0	0C90	1
UNKNOWN	UNKNOWN	COMPOSITE PIECE - INDEX	0	0E90	4
UNKNOWN	UNKNOWN	C4E8E77E8056		*UNK	2
UNKNOWN	UNKNOWN	C81536EFC100		*UNK	1
UNKNOWN	UNKNOWN	Descriptor table	S	210C	1
UNKNOWN	UNKNOWN	DATA BASE FILE MEMBER	0	0D50	14

Table of Contents | Previous | Next

5.5.52 IO Total by Program/ObjectSegment-Desc/IO-Type

This report shows the total number of physical disk IO events that occurred for each IO type by object type within program.

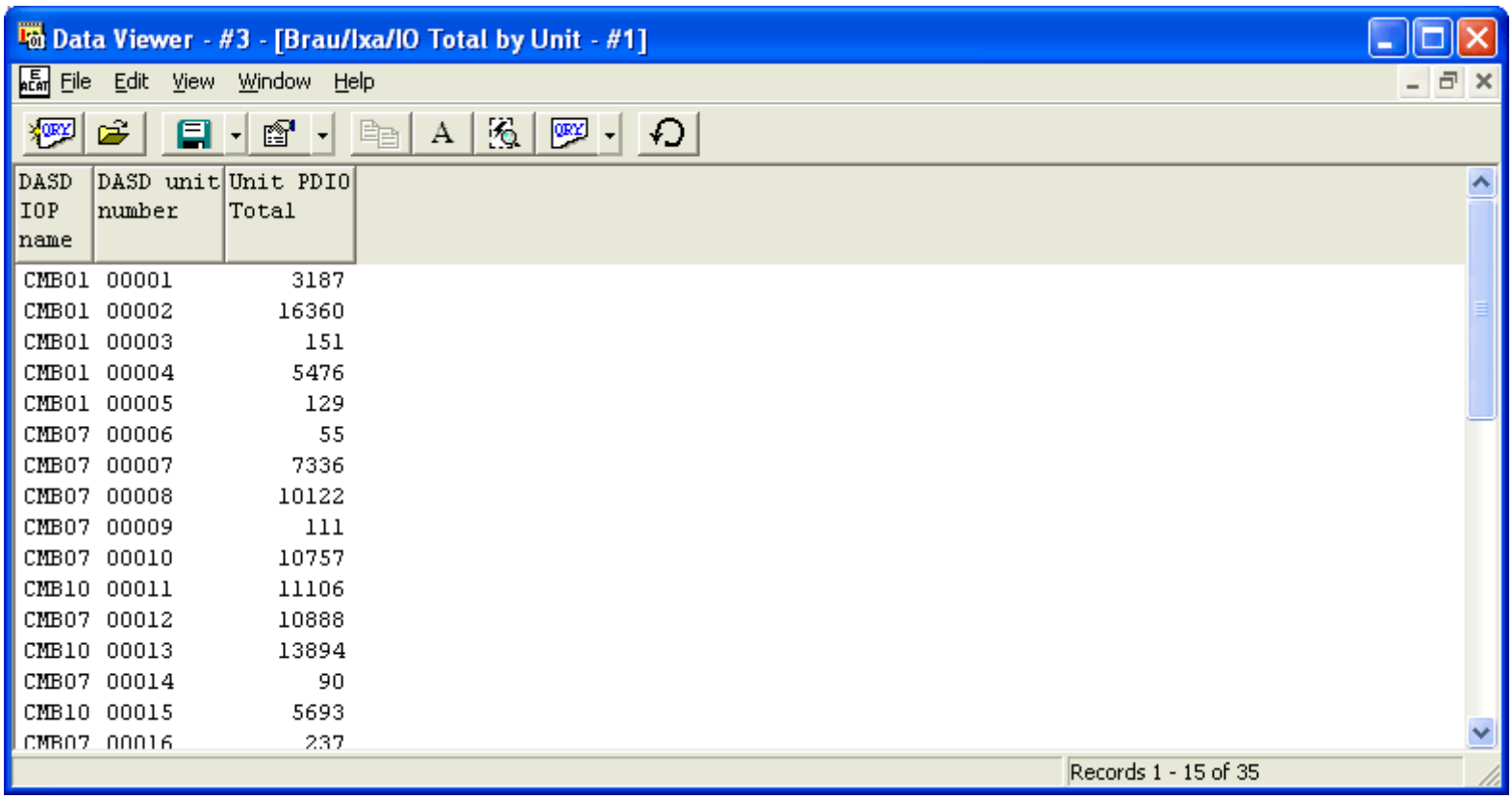


Program Library	Program	Object/Segment Desc	Object/Segment Flag 0=Obj S=Seg ' '=*Unknown	Object/Segment Type	DASD I/O type (short version)	Program/ObjSeg-PDIO-Type Total
UNKNOWN	UNKNOWN	*UNKNOWN	0		FT	
UNKNOWN	UNKNOWN	*UNKNOWN	0		PO	
UNKNOWN	UNKNOWN	*UNKNOWN	0		RS	
UNKNOWN	UNKNOWN	*UNKNOWN	0		WA	
UNKNOWN	UNKNOWN	*UNKNOWN	0		WS	
UNKNOWN	UNKNOWN	*UNKNOWN	S	2000	FT	
UNKNOWN	UNKNOWN	*UNKNOWN	S	2000	PO	
UNKNOWN	UNKNOWN	seize/lock control table	S	2081	FT	
UNKNOWN	UNKNOWN	seize/lock control table	S	2081	PO	
UNKNOWN	UNKNOWN	ACTIVATION PROC REF TABLE	S	20D7	PO	
UNKNOWN	UNKNOWN	AS/400 LiM P FS DIR		*UNK	FT	
UNKNOWN	UNKNOWN	ASM STACKS SEG		*UNK	FT	
UNKNOWN	UNKNOWN	ASM STACKS SEG		*UNK	PO	
UNKNOWN	UNKNOWN	BYTESTREAM FILE	0	1E01	FT	
UNKNOWN	UNKNOWN	BYTESTREAM FILE	0	1E01	RA	

Table of Contents | Previous | Next

5.5.53 IO Total by Unit

This report shows the total number of physical disk IO events that occurred for each disk unit.

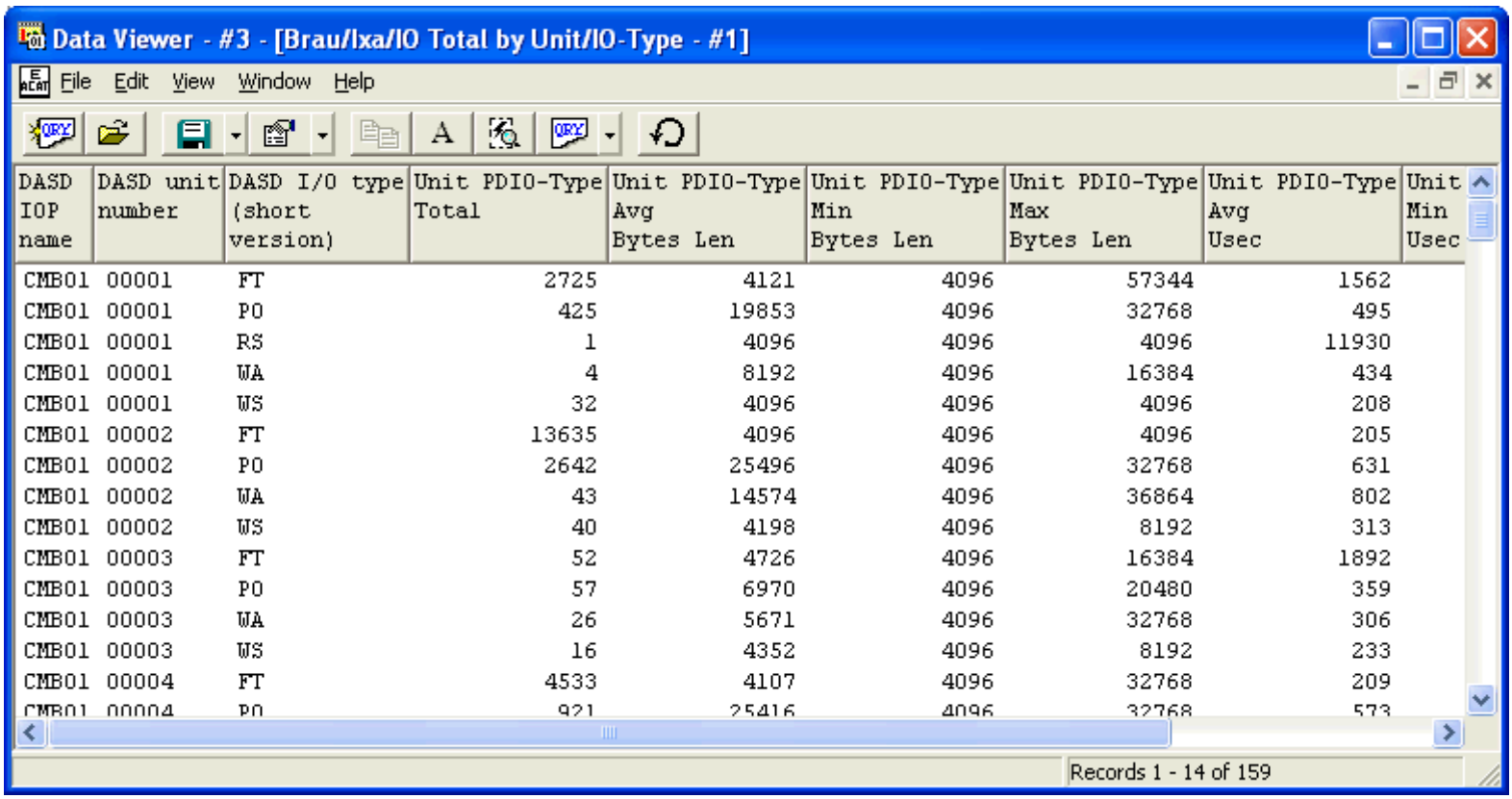


DASD IOP name	DASD unit number	Unit PDIO Total
CMB01	00001	3187
CMB01	00002	16360
CMB01	00003	151
CMB01	00004	5476
CMB01	00005	129
CMB07	00006	55
CMB07	00007	7336
CMB07	00008	10122
CMB07	00009	111
CMB07	00010	10757
CMB10	00011	11106
CMB07	00012	10888
CMB10	00013	13894
CMB07	00014	90
CMB10	00015	5693
CMB07	00016	237

Table of Contents | Previous | Next

5.5.54 IO Total by Unit/IO-Type

This report shows a breakdown of physical disk IO events that occurred by each IO type by disk unit.

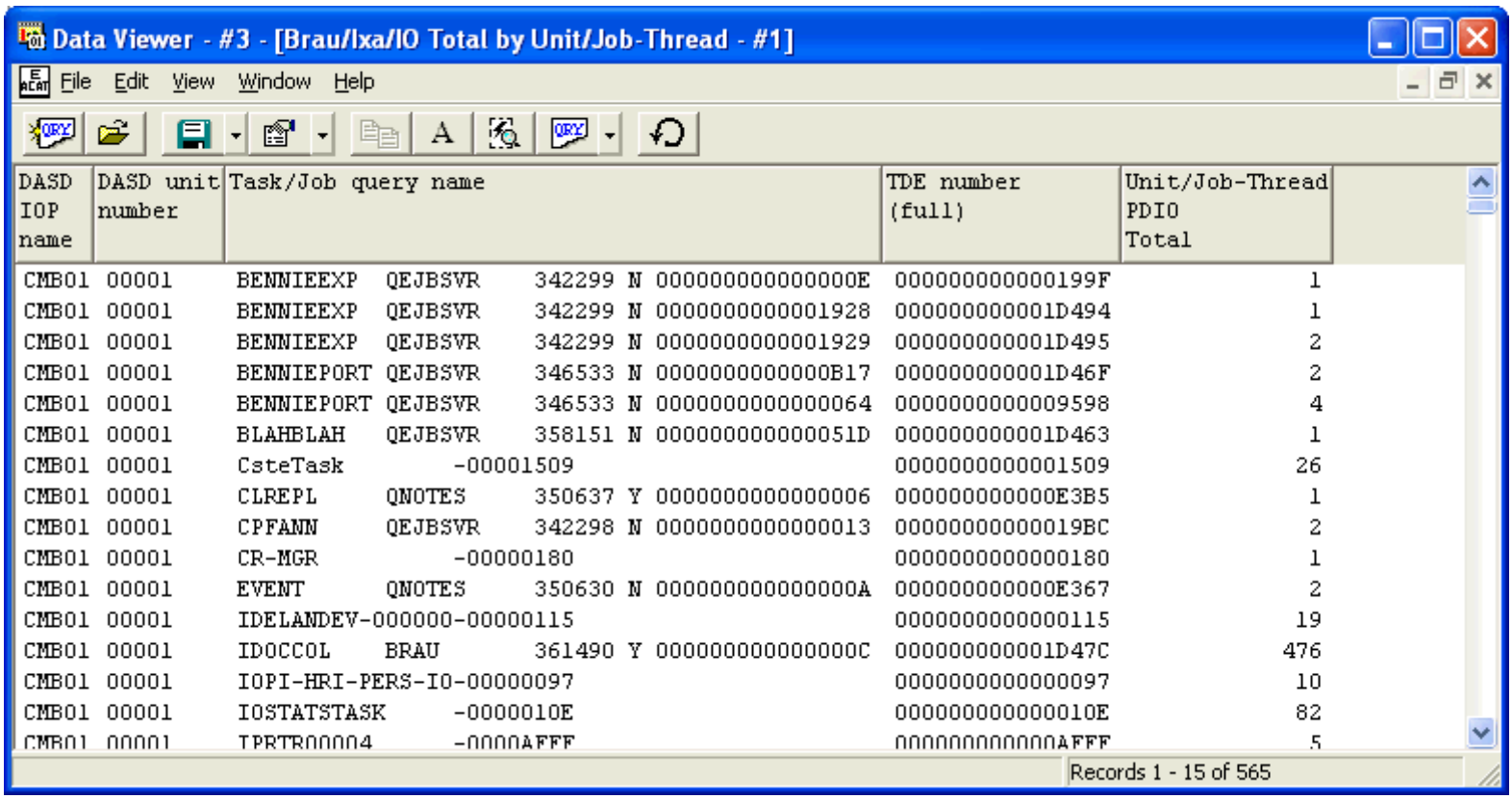


DASD IOP name	DASD unit number	DASD I/O type (short version)	Unit PDIO-Type Total	Unit PDIO-Type Avg Bytes Len	Unit PDIO-Type Min Bytes Len	Unit PDIO-Type Max Bytes Len	Unit PDIO-Type Avg Usec	Unit Min Usec
CMB01	00001	FT	2725	4121	4096	57344	1562	
CMB01	00001	PO	425	19853	4096	32768	495	
CMB01	00001	RS	1	4096	4096	4096	11930	
CMB01	00001	WA	4	8192	4096	16384	434	
CMB01	00001	WS	32	4096	4096	4096	208	
CMB01	00002	FT	13635	4096	4096	4096	205	
CMB01	00002	PO	2642	25496	4096	32768	631	
CMB01	00002	WA	43	14574	4096	36864	802	
CMB01	00002	WS	40	4198	4096	8192	313	
CMB01	00003	FT	52	4726	4096	16384	1892	
CMB01	00003	PO	57	6970	4096	20480	359	
CMB01	00003	WA	26	5671	4096	32768	306	
CMB01	00003	WS	16	4352	4096	8192	233	
CMB01	00004	FT	4533	4107	4096	32768	209	
CMB01	00004	PO	921	25416	4096	32768	573	

Table of Contents | Previous | Next

5.5.55 IO Total by Unit/Job-Thread

This report shows the total number of physical disk IO events that occurred for each disk unit by job/thread.



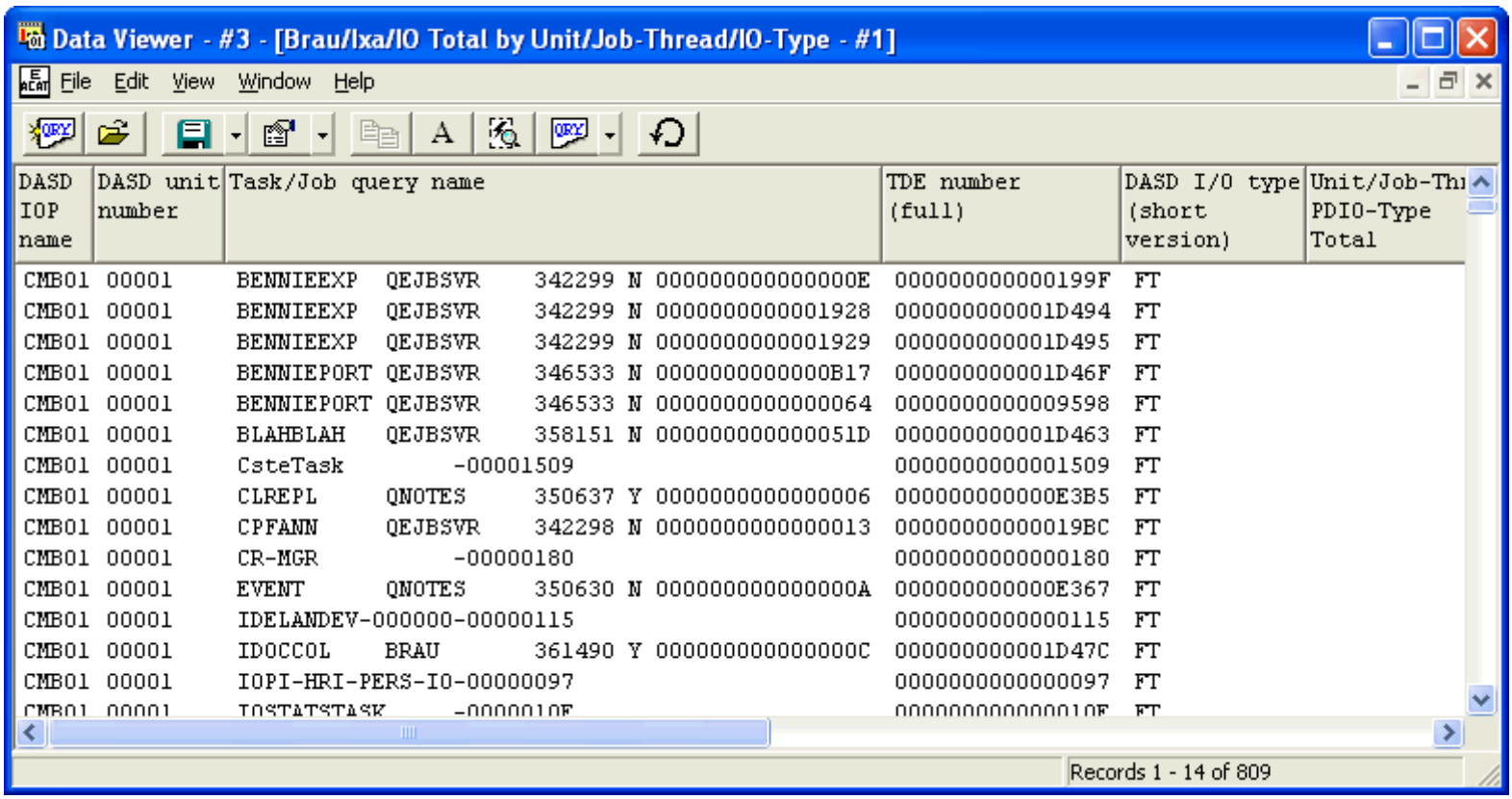
DASD IOP name	DASD unit number	Task/Job query name					TDE number (full)	Unit/Job-Thread PDIO Total
CMB01	00001	BENNIEEXP	QEJBSVR	342299	N	000000000000000E	000000000000199F	1
CMB01	00001	BENNIEEXP	QEJBSVR	342299	N	0000000000001928	0000000000001D494	1
CMB01	00001	BENNIEEXP	QEJBSVR	342299	N	0000000000001929	0000000000001D495	2
CMB01	00001	BENNIEPORT	QEJBSVR	346533	N	0000000000000B17	0000000000001D46F	2
CMB01	00001	BENNIEPORT	QEJBSVR	346533	N	0000000000000064	0000000000009598	4
CMB01	00001	BLAHBLAH	QEJBSVR	358151	N	000000000000051D	0000000000001D463	1
CMB01	00001	CsteTask	-00001509				0000000000001509	26
CMB01	00001	CLREPL	QNOTES	350637	Y	0000000000000006	000000000000E3B5	1
CMB01	00001	CPFANN	QEJBSVR	342298	N	0000000000000013	00000000000019BC	2
CMB01	00001	CR-MGR	-00000180				0000000000000180	1
CMB01	00001	EVENT	QNOTES	350630	N	000000000000000A	000000000000E367	2
CMB01	00001	IDELANDEV-000000-00000115					0000000000000115	19
CMB01	00001	IDOCCOL	BRAU	361490	Y	000000000000000C	0000000000001D47C	476
CMB01	00001	IOPI-HRI-PERS-IO-00000097					0000000000000097	10
CMB01	00001	IOSTATSTASK	-0000010E				000000000000010E	82
CMB01	00001	IPRTR00004	-0000AFFF				000000000000AFFF	5

Records 1 - 15 of 565

Table of Contents | Previous | Next

5.5.56 IO Total by Unit/Job-Thread/IO-Type

This report shows the total number of physical disk IO events that occurred for each IO type by disk unit by job/thread.

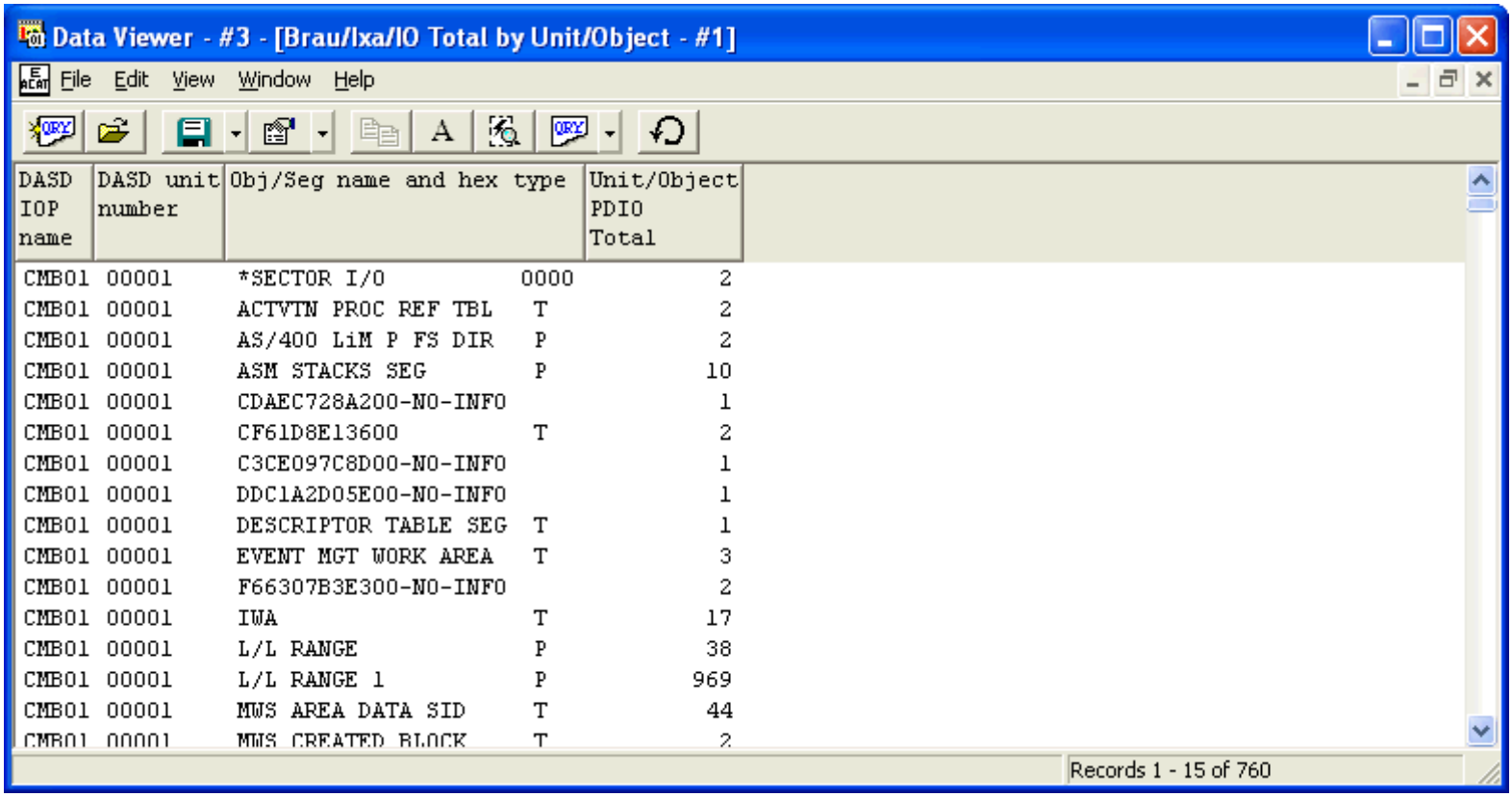


DASD IOP name	DASD unit number	Task/Job query name				TDE number (full)	DASD I/O type (short version)	Unit/Job-Thr PDIO-Type Total
CMB01	00001	BENNIEEXP	QEJBSVR	342299	N 000000000000000E	0000000000001 99F	FT	
CMB01	00001	BENNIEEXP	QEJBSVR	342299	N 0000000000001928	0000000000001D494	FT	
CMB01	00001	BENNIEEXP	QEJBSVR	342299	N 0000000000001929	0000000000001D495	FT	
CMB01	00001	BENNIEPORT	QEJBSVR	346533	N 0000000000000B17	0000000000001D46F	FT	
CMB01	00001	BENNIEPORT	QEJBSVR	346533	N 0000000000000064	0000000000009598	FT	
CMB01	00001	BLAHBLAH	QEJBSVR	358151	N 000000000000051D	0000000000001D463	FT	
CMB01	00001	CsteTask		-00001509		0000000000001509	FT	
CMB01	00001	CLREPL	QNOTES	350637	Y 0000000000000006	0000000000E3B5	FT	
CMB01	00001	CPFANN	QEJBSVR	342298	N 0000000000000013	00000000000019BC	FT	
CMB01	00001	CR-MGR		-00000180		0000000000000180	FT	
CMB01	00001	EVENT	QNOTES	350630	N 000000000000000A	0000000000E367	FT	
CMB01	00001	IDELANDEV-000000-00000115				0000000000000115	FT	
CMB01	00001	IDOCCOL	BRAU	361490	Y 000000000000000C	0000000000001D47C	FT	
CMB01	00001	IOPI-HRI-PERS-IO-00000097				0000000000000097	FT	
CMB01	00001	IOSTATSTASK		-0000010F		000000000000010F	FT	

Table of Contents | Previous | Next

5.5.57 IO Total by Unit/Object

This report shows the total number of physical disk IO events that occurred for each object within disk unit.

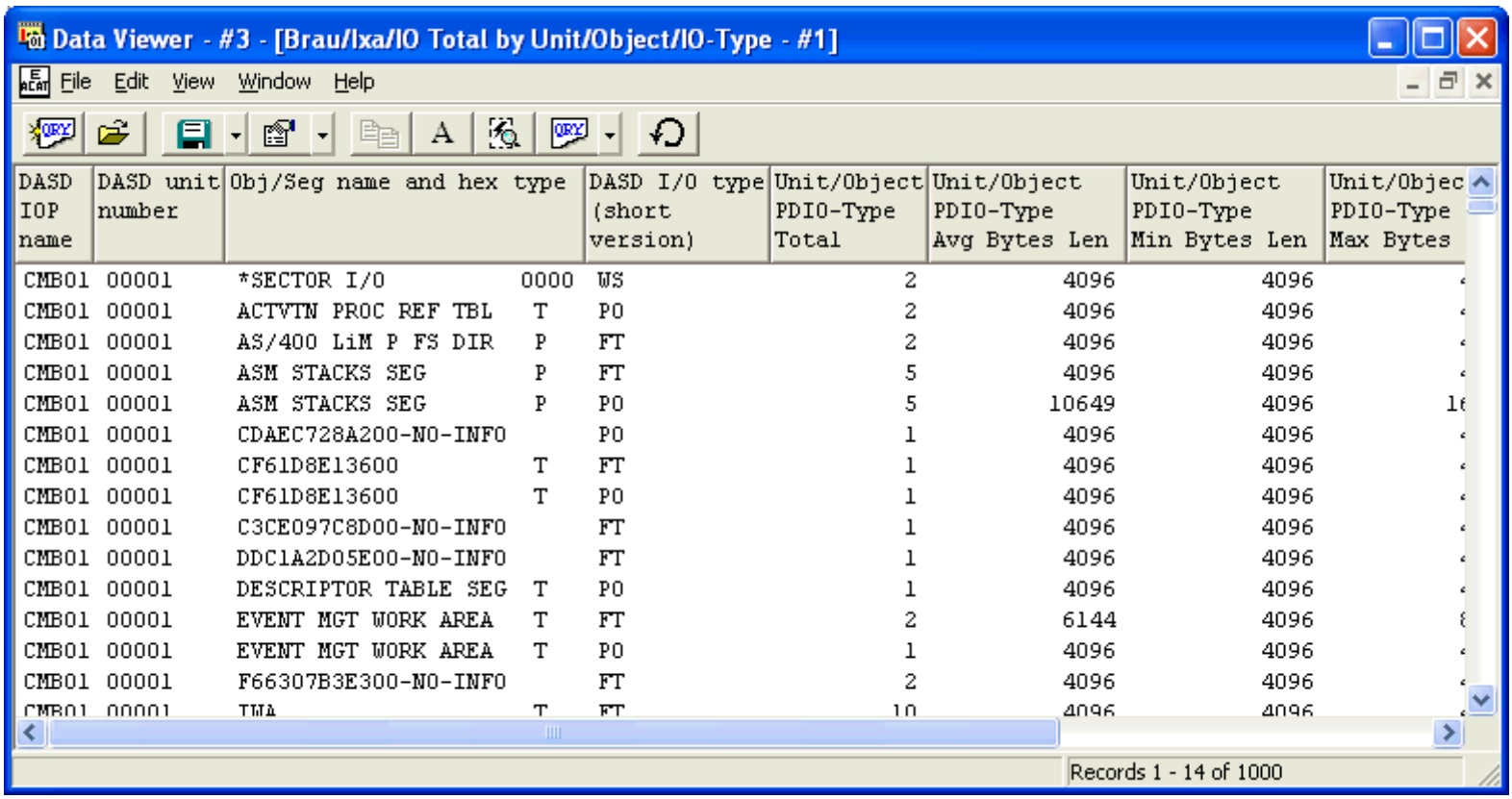


DASD IOP name	DASD unit number	Obj/Seg name and hex type		Unit/Object PDIO Total
CMB01	00001	*SECTOR I/O	0000	2
CMB01	00001	ACTVTN PROC REF TBL	T	2
CMB01	00001	AS/400 LiM P FS DIR	P	2
CMB01	00001	ASM STACKS SEG	P	10
CMB01	00001	CDAEC728A200-N0-INF0		1
CMB01	00001	CF61D8E13600	T	2
CMB01	00001	C3CE097C8D00-N0-INF0		1
CMB01	00001	DDC1A2D05E00-N0-INF0		1
CMB01	00001	DESCRIPTOR TABLE SEG	T	1
CMB01	00001	EVENT MGT WORK AREA	T	3
CMB01	00001	F66307B3E300-N0-INF0		2
CMB01	00001	IWA	T	17
CMB01	00001	L/L RANGE	P	38
CMB01	00001	L/L RANGE 1	P	969
CMB01	00001	MWS AREA DATA SID	T	44
CMB01	00001	MWS CREATED BLOCK	T	2

Table of Contents | Previous | Next

5.5.58 IO Total by Unit/Object/IO-Type

This report shows a breakdown of physical disk IO events for each IO type by object within disk unit.



DASD IOP name	DASD unit number	Obj/Seg name and hex type		DASD I/O type (short version)	Unit/Object PDIO-Type Total	Unit/Object PDIO-Type Avg Bytes Len	Unit/Object PDIO-Type Min Bytes Len	Unit/Objec PDIO-Type Max Bytes
CMB01	00001	*SECTOR I/O	0000	WS	2	4096	4096	4
CMB01	00001	ACTVTN PROC REF TBL	T	PO	2	4096	4096	4
CMB01	00001	AS/400 LiM P FS DIR	P	FT	2	4096	4096	4
CMB01	00001	ASM STACKS SEG	P	FT	5	4096	4096	4
CMB01	00001	ASM STACKS SEG	P	PO	5	10649	4096	16
CMB01	00001	CDAEC728A200-NO-INFO		PO	1	4096	4096	4
CMB01	00001	CF61D8E13600	T	FT	1	4096	4096	4
CMB01	00001	CF61D8E13600	T	PO	1	4096	4096	4
CMB01	00001	C3CE097C8D00-NO-INFO		FT	1	4096	4096	4
CMB01	00001	DDC1A2D05E00-NO-INFO		FT	1	4096	4096	4
CMB01	00001	DESCRIPTOR TABLE SEG	T	PO	1	4096	4096	4
CMB01	00001	EVENT MGT WORK AREA	T	FT	2	6144	4096	8
CMB01	00001	EVENT MGT WORK AREA	T	PO	1	4096	4096	4
CMB01	00001	F66307B3E300-NO-INFO		FT	2	4096	4096	4
CMB01	00001	IWA	T	FT	10	4096	4096	4

Records 1 - 14 of 1000

Table of Contents | Previous | Next

5.6 Size change to objects and segments reports

This analysis shows every dasd space create/destroy or size change of external objects/segments. It covers both permanent and temporary objects.
The analysis detail includes the object name and object type, causing job/task and length (size) of the create/destroy.

Table of Contents | Previous | Next

5.6.1 Job/Task Consuming most DASD Space per Interval

Description: This report shows the job within each interval of the collection that consumed the most DASD space. The delta bytes consumed by this job is provided.

Example:

Data Viewer - #1 - [Mccargar/Temptest/Job/Task Consuming most DASD Space per Interval - #1]

File Edit View Window Help

Interval	Job/Task Thread					Total TEMP Storage Bytes Size Change	Total PERM Storage Bytes Size Change	Total Bytes Size Change	Interval Max Timestamp
2	RGBLOM3	QEJBSVR	585154	N	000000000000001E	528384	0	528384	2004-02-06-1
3	QYPSJSVR	QYPSJSVR	583939	N	0000000000000A8F	-65536	0	-65536	2004-02-06-1
4	QPADEV0001	CKUHLMAN	584002	Y	0000000000000004	585728	0	585728	2004-02-06-1
5	QPADEV0001	CKUHLMAN	584002	Y	0000000000000004	2764800	0	2764800	2004-02-06-1
6	QPADEV0001	CKUHLMAN	584002	Y	0000000000000004	118784	0	118784	2004-02-06-1
7	SERVER1	QEJBSVR	584748	N	0000000000000155	786432	0	786432	2004-02-06-1
12	QPADEV0001	CKUHLMAN	584002	Y	0000000000000004	32768	0	32768	2004-02-06-1
13	QPADEV0001	CKUHLMAN	584002	Y	0000000000000004	-32768	4096	-28672	2004-02-06-1
14	PAULSSVR	QEJBSVR	585119	N	000000000000016B	131072	0	131072	2004-02-06-1
15	QPADEV0001	CKUHLMAN	584002	Y	0000000000000004	8192	0	8192	2004-02-06-1
18	QPADEV0001	CKUHLMAN	584002	Y	0000000000000004	53248	0	53248	2004-02-06-1
19	QYPSJSVR	QYPSJSVR	583939	N	0000000000000A92	65536	0	65536	2004-02-06-1
22	QYPSJSVR	QYPSJSVR	583939	N	0000000000000A93	65536	0	65536	2004-02-06-1
25	QYPSJSVR	QYPSJSVR	583939	N	0000000000000A94	65536	0	65536	2004-02-06-1
27	ALLSTATE	QEJBSVR	585808	N	000000000000015E	163840	0	163840	2004-02-06-1
29	QYPSJSVR	QYPSJSVR	583939	N	0000000000000A93	-65536	0	-65536	2004-02-06-1
30	IDOCCOL	CKUHLMAN	587354	Y	0000000000000005	20480	24576	45056	2004-02-06-1

Records 1 - 17 of 17

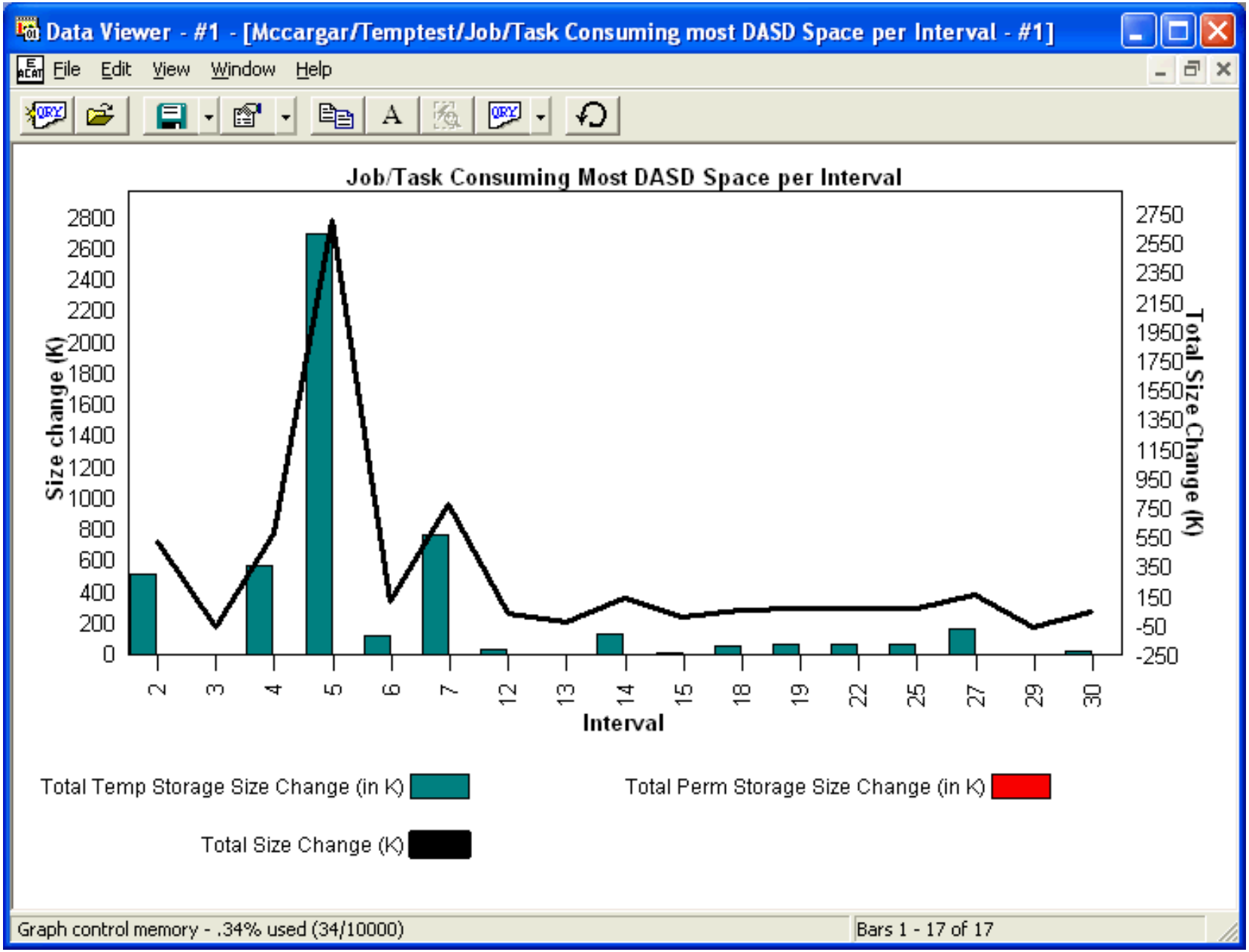
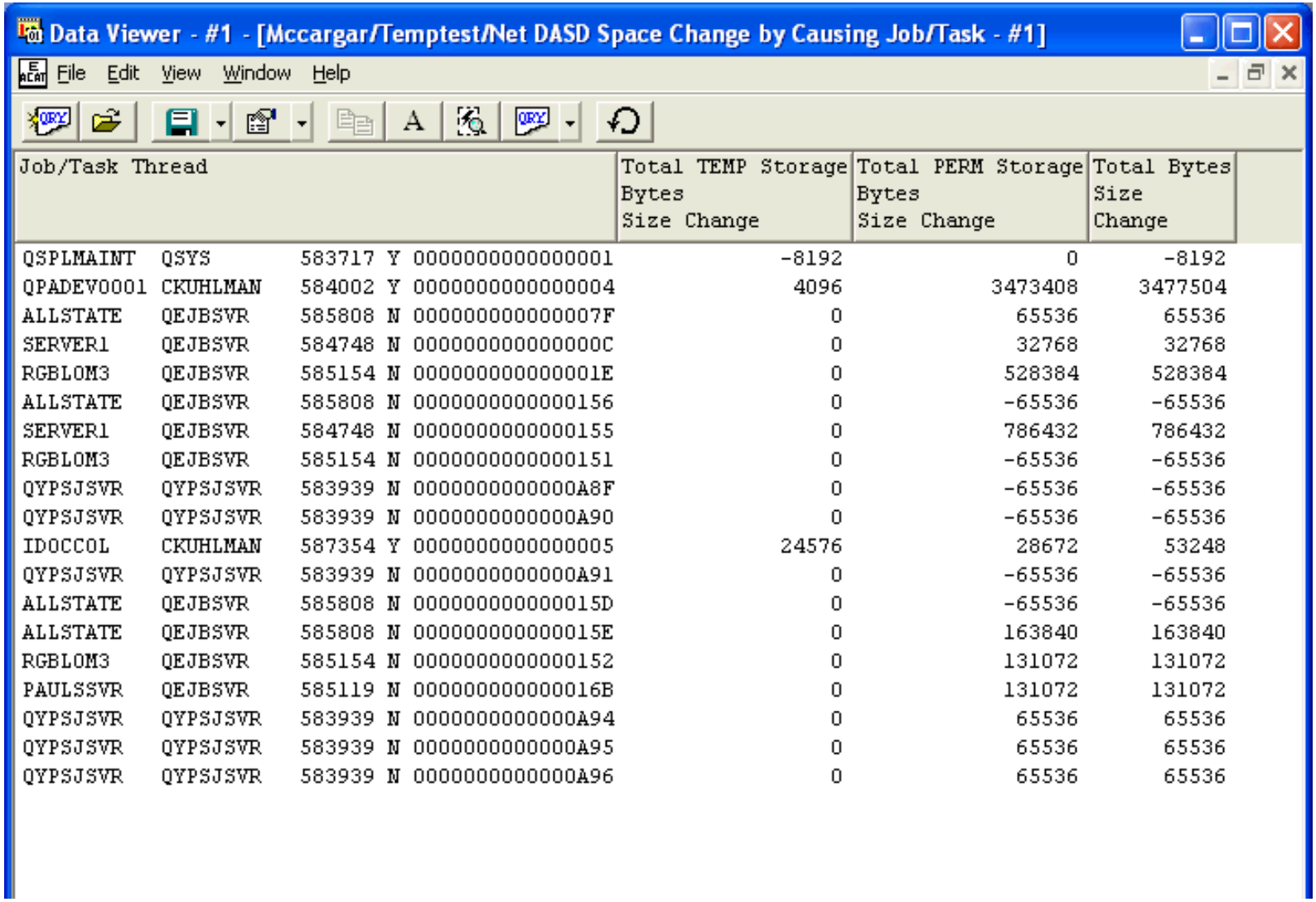

Data Viewer - #1 - [Mccargar/Temptest/Job/Task Consuming most DASD Space per Interval - #1]
File Edit View Window Help
Job/Task Consuming Most DASD Space per Interval
Size change (K)
Total Size Change (K)
Interval
2 3 4 5 6 7 12 13 14 15 18 19 22 25 27 29 30
Total Temp Storage Size Change (in K)
Total Perm Storage Size Change (in K)
Total Size Change (K)
Graph control memory - .34% used (34/10000)
Bars 1 - 17 of 17

Table of Contents | Previous | Next

5.6.2 Net DASD Space Change by Causing Job/Task

Description: The report shows the temporary and permanent Dasd space change for each job/task. In the graph the user can click on any bar (temporary or permanent space change) and get information on the job/task(fully qualified job name or task name) causing it, and space change. The graph's right y-axis provides the total of the temporary and permanent Dasd space change.

Example:



Job/Task Thread					Total TEMP Storage Bytes Size Change	Total PERM Storage Bytes Size Change	Total Bytes Size Change
QSPLMAINT	QSYS	583717	Y	0000000000000001	-8192	0	-8192
QPADEV0001	CKUHLMAN	584002	Y	0000000000000004	4096	3473408	3477504
ALLSTATE	QEJBSVR	585808	N	000000000000007F	0	65536	65536
SERVER1	QEJBSVR	584748	N	000000000000000C	0	32768	32768
RGBLOM3	QEJBSVR	585154	N	000000000000001E	0	528384	528384
ALLSTATE	QEJBSVR	585808	N	0000000000000156	0	-65536	-65536
SERVER1	QEJBSVR	584748	N	0000000000000155	0	786432	786432
RGBLOM3	QEJBSVR	585154	N	0000000000000151	0	-65536	-65536
QYPSJSVR	QYPSJSVR	583939	N	0000000000000A8F	0	-65536	-65536
QYPSJSVR	QYPSJSVR	583939	N	0000000000000A90	0	-65536	-65536
IDOCCOL	CKUHLMAN	587354	Y	0000000000000005	24576	28672	53248
QYPSJSVR	QYPSJSVR	583939	N	0000000000000A91	0	-65536	-65536
ALLSTATE	QEJBSVR	585808	N	000000000000015D	0	-65536	-65536
ALLSTATE	QEJBSVR	585808	N	000000000000015E	0	163840	163840
RGBLOM3	QEJBSVR	585154	N	0000000000000152	0	131072	131072
PAULSSVR	QEJBSVR	585119	N	000000000000016B	0	131072	131072
QYPSJSVR	QYPSJSVR	583939	N	0000000000000A94	0	65536	65536
QYPSJSVR	QYPSJSVR	583939	N	0000000000000A95	0	65536	65536
QYPSJSVR	QYPSJSVR	583939	N	0000000000000A96	0	65536	65536

Records 1 - 19 of 19

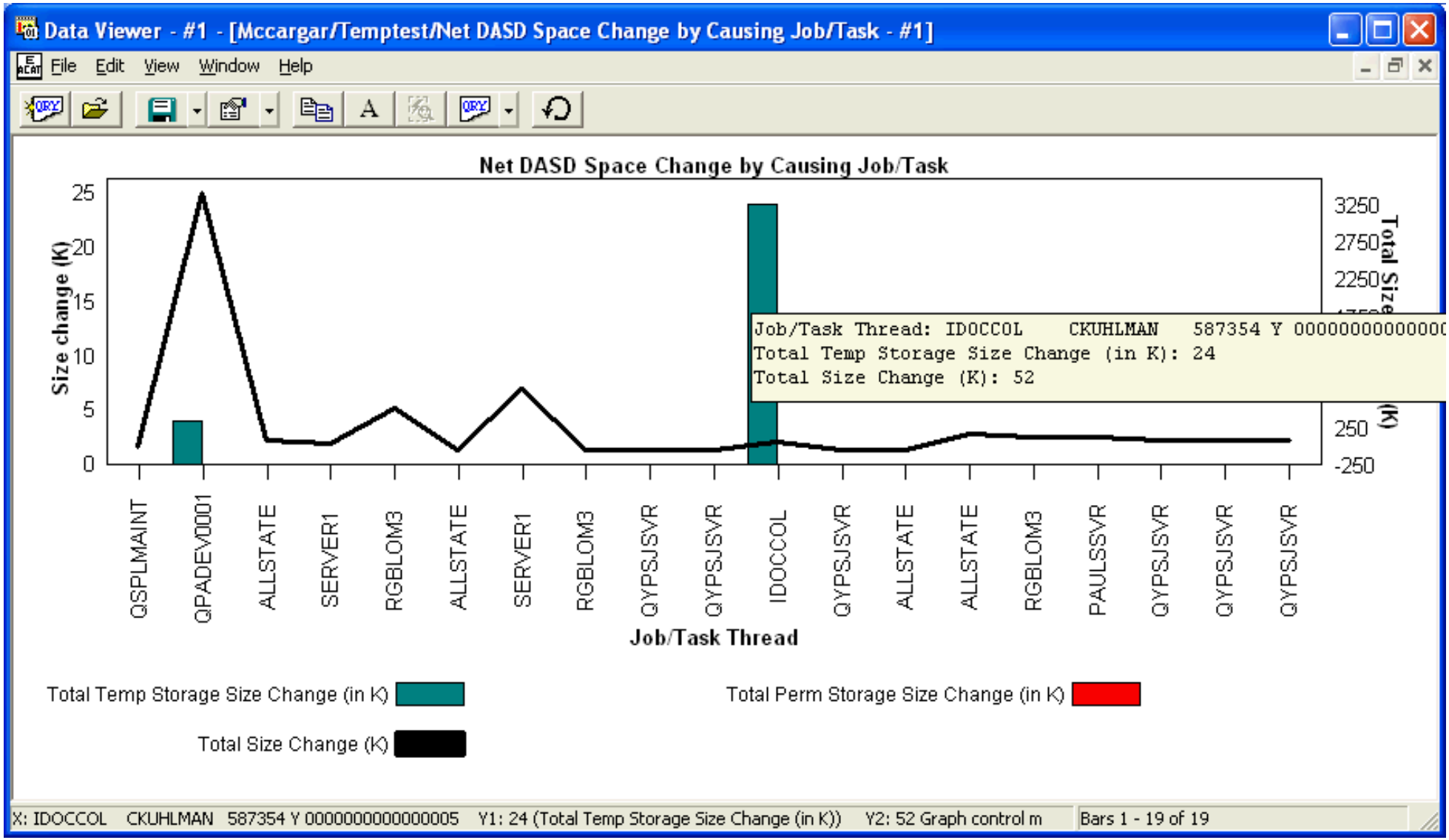


Table of Contents | Previous | Next

5.6.3 Net DASD Space Change by Object Name and Symbolic Type

Description:

Example:

Data Viewer - #1 - [Mccargar/Temptest/Net DASD Space Change by Object Name and Symbolic Type - #1]

File Edit View Window Help

Object/Segment Name/Type	Total TEMP Storage Bytes Size Change	Total PERM Storage Bytes Size Change	Total Bytes Size Change
QPADEV0001CKUHLMAN 1AEF	2453504	0	2453504
JAVA HEAP SEGMENT T	917504	0	917504
MWS AREA DATA SID T	524288	0	524288
ACTVTN PROC REF TBL T	131072	0	131072
MWS CREATED BLOCK T	114688	0	114688
QAPMJOBOS Q318063008 0B90	86016	0	86016
QCLIST 19EF	77824	0	77824
QSQCURSORS 19EF	73728	0	73728
SQLDA TEMP SPACE0000 19EF	69632	0	69632
ALLSTATE QEJBSVR 1AEF	65536	0	65536
PAULSSVR QEJBSVR 1AEF	65536	0	65536
RGBLOM3 QEJBSVR 1AEF	65536	0	65536
9673A3000000 T	65536	0	65536
XQSQPAS 0000 19EF	49152	0	49152
LCKLIST 19EF	36864	0	36864
QQSPC0000007 19EF	36864	0	36864
QSQDT00001 19EF	36864	0	36864
TRIGGER SPACE 0000 19EF	36864	0	36864
QQSPC0000005 19EF	28672	0	28672
QQSPC0000006 19EF	28672	0	28672
SQL.WORK.SPACE.XTRT 19EF	28672	0	28672
CKUHLMAN 0EC4	24576	0	24576
QQSPC0000002 19EF	24576	0	24576

Records 1 - 22 of 46

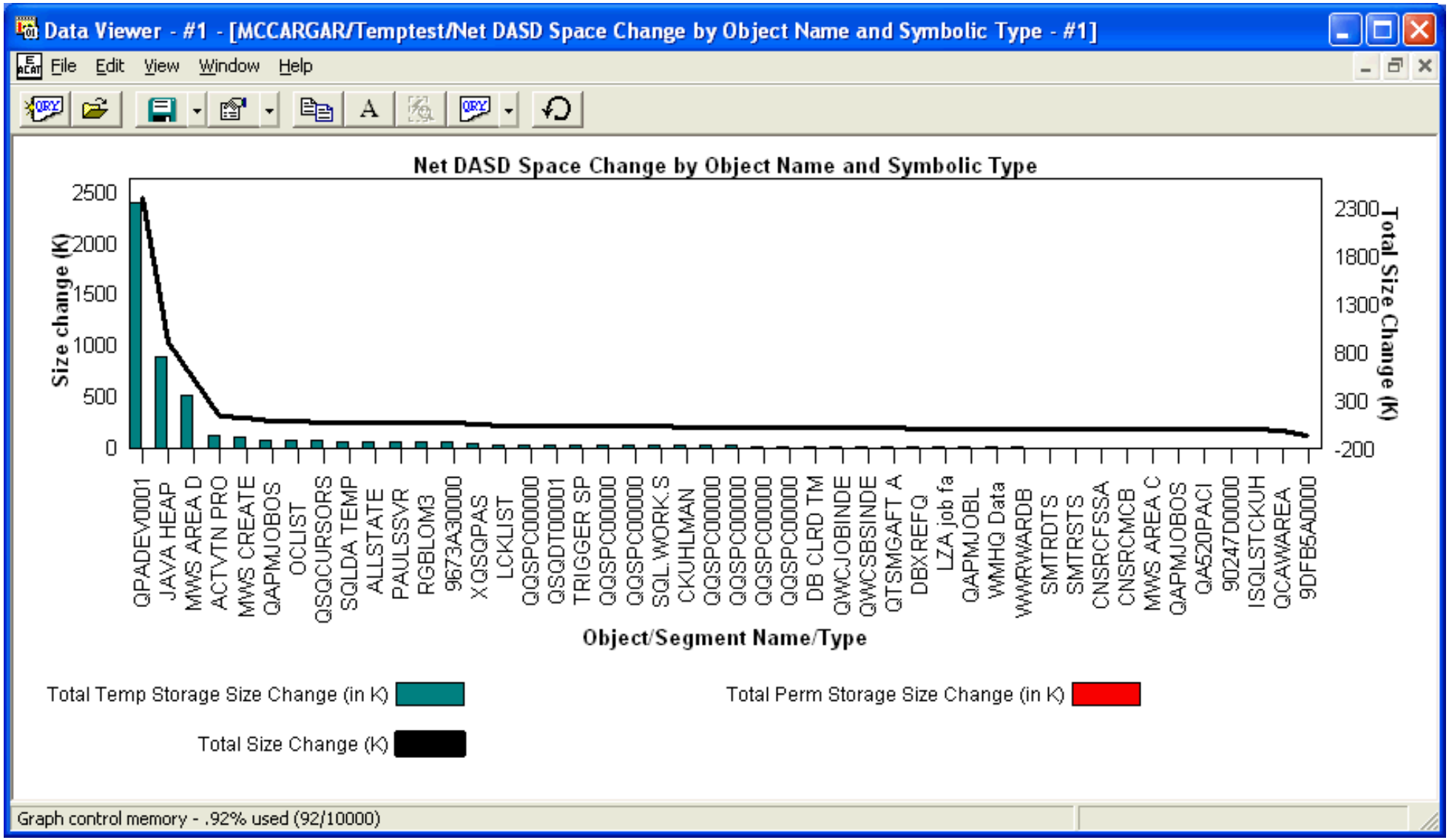
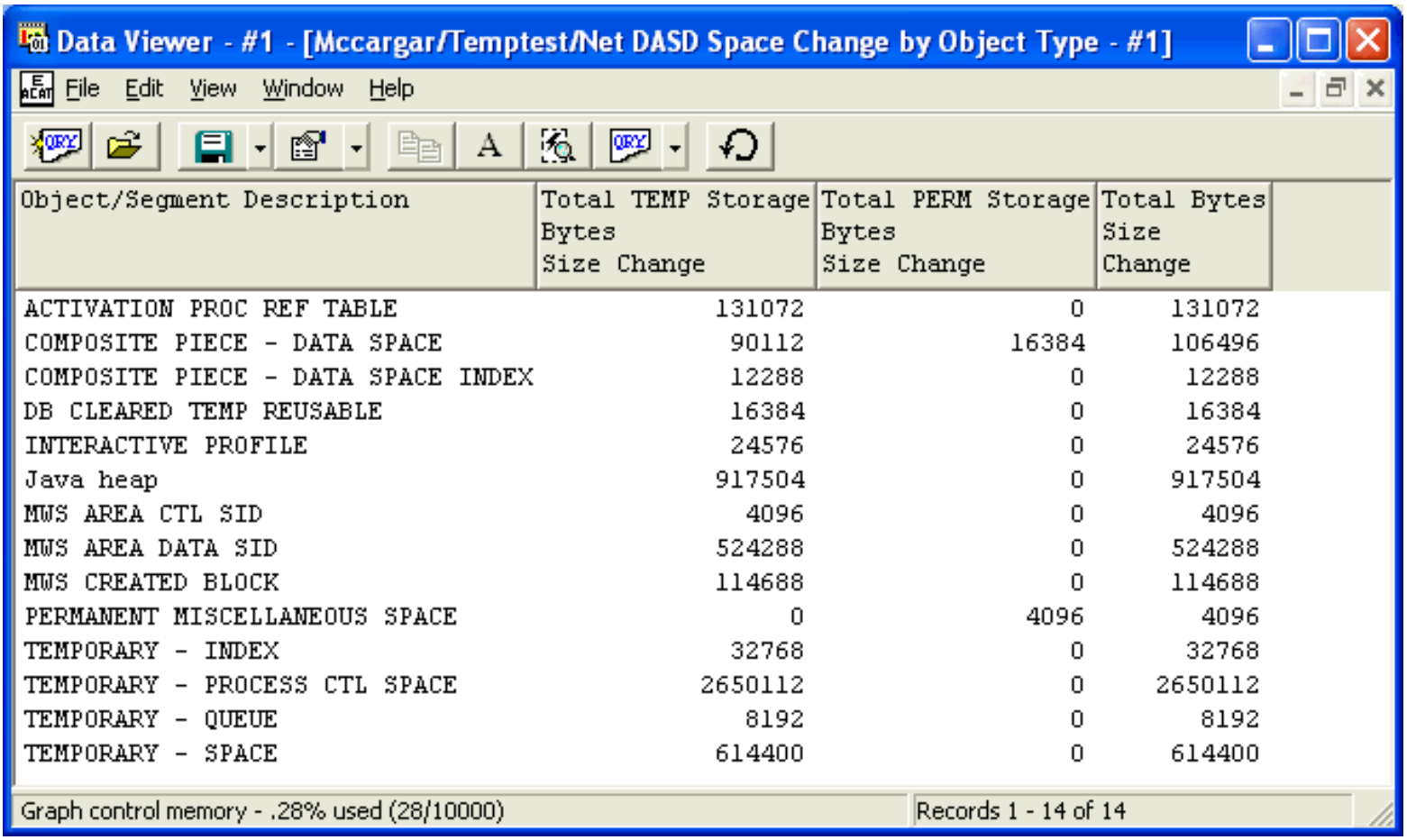


Table of Contents | Previous | Next

5.6.4 Net DASD Space Change by Object Type

Description:

Example:



Object/Segment Description	Total TEMP Storage Bytes Size Change	Total PERM Storage Bytes Size Change	Total Bytes Size Change
ACTIVATION PROC REF TABLE	131072	0	131072
COMPOSITE PIECE - DATA SPACE	90112	16384	106496
COMPOSITE PIECE - DATA SPACE INDEX	12288	0	12288
DB CLEARED TEMP REUSABLE	16384	0	16384
INTERACTIVE PROFILE	24576	0	24576
Java heap	917504	0	917504
MWS AREA CTL SID	4096	0	4096
MWS AREA DATA SID	524288	0	524288
MWS CREATED BLOCK	114688	0	114688
PERMANENT MISCELLANEOUS SPACE	0	4096	4096
TEMPORARY - INDEX	32768	0	32768
TEMPORARY - PROCESS CTL SPACE	2650112	0	2650112
TEMPORARY - QUEUE	8192	0	8192
TEMPORARY - SPACE	614400	0	614400

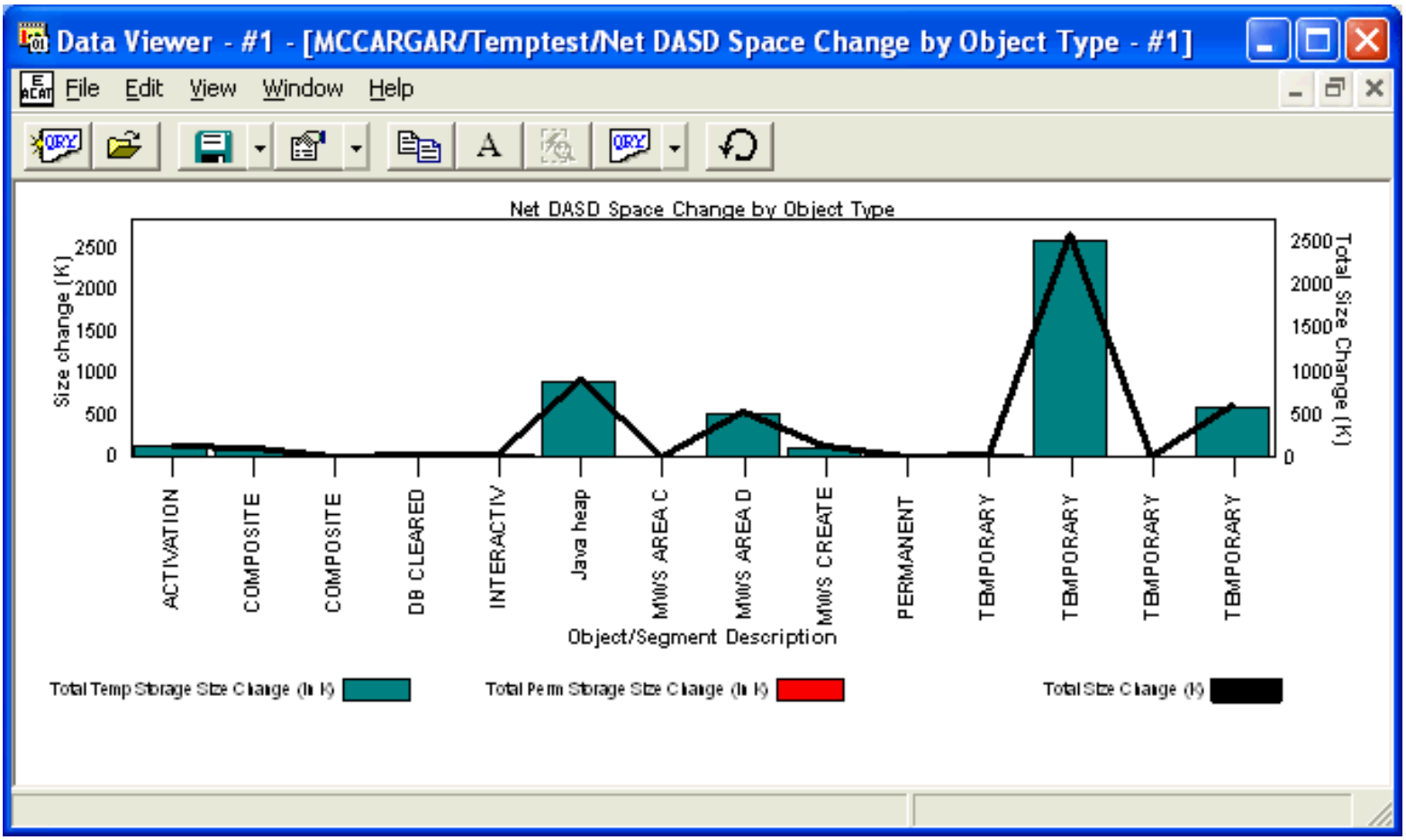
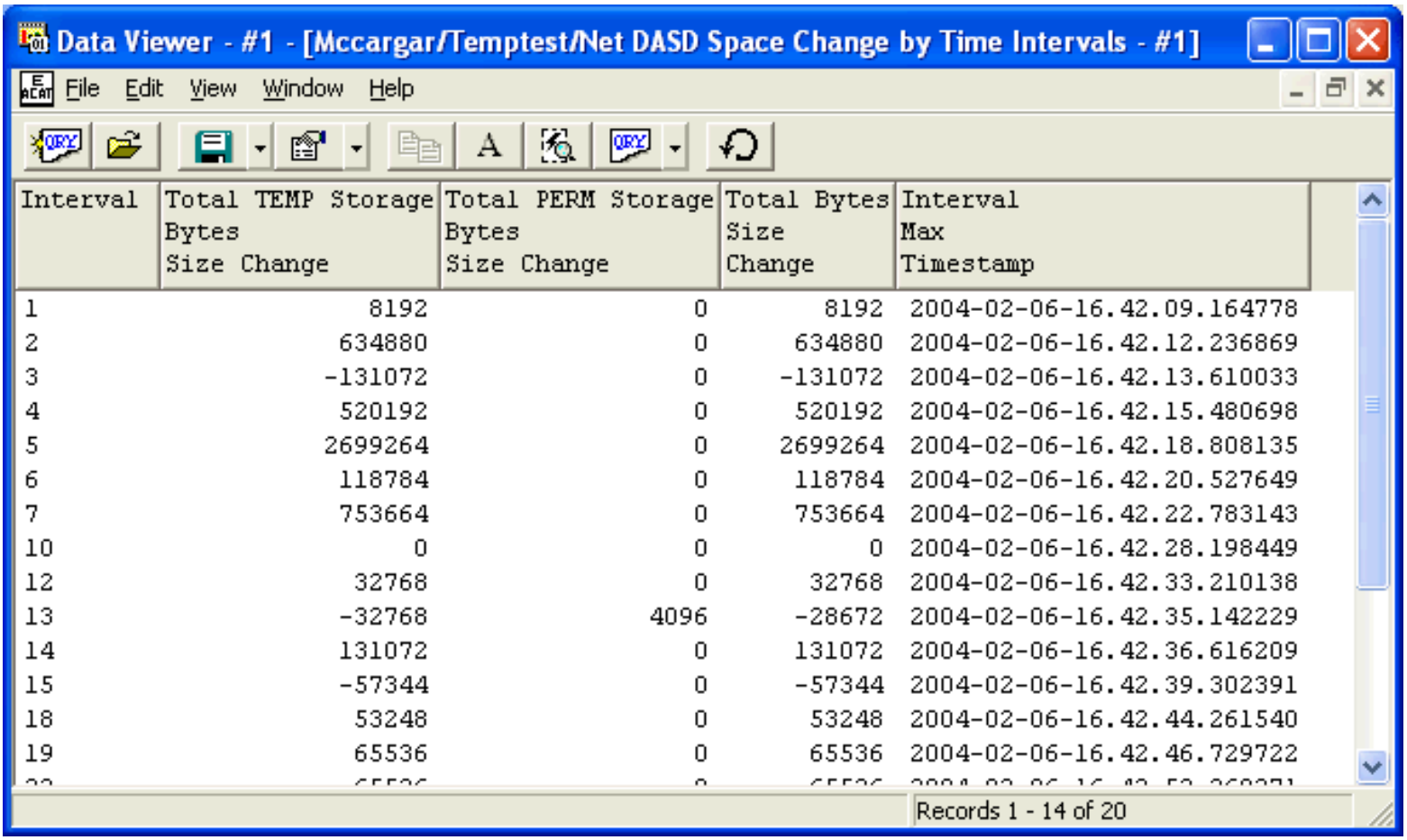

Data Viewer - #1 - [MCCARGAR/Temptest/Net DASD Space Change by Object Type - #1]
File Edit View Window Help
Net DASD Space Change by Object Type
Size change (K)
2500
2000
1500
1000
500
0
Total Size Change (K)
ACTIVATION
COMPOSITE
COMPOSITE
DB CLEARED
INTERACTIV
Java heap
MWS AREA C
MWS AREA D
MWS CREATE
PERMANENT
TEMPORARY
TEMPORARY
TEMPORARY
TEMPORARY
Object/Segment Description
Total Temp Storage Size Change (in K)
Total Perm Storage Size Change (in K)
Total Size Change (K)

Table of Contents | Previous | Next

5.6.5 Net DASD Space Change by Time Intervals

Description:

Example:



Interval	Total TEMP Storage Bytes Size Change	Total PERM Storage Bytes Size Change	Total Bytes Size Change	Interval Max Timestamp
1	8192	0	8192	2004-02-06-16.42.09.164778
2	634880	0	634880	2004-02-06-16.42.12.236869
3	-131072	0	-131072	2004-02-06-16.42.13.610033
4	520192	0	520192	2004-02-06-16.42.15.480698
5	2699264	0	2699264	2004-02-06-16.42.18.808135
6	118784	0	118784	2004-02-06-16.42.20.527649
7	753664	0	753664	2004-02-06-16.42.22.783143
10	0	0	0	2004-02-06-16.42.28.198449
12	32768	0	32768	2004-02-06-16.42.33.210138
13	-32768	4096	-28672	2004-02-06-16.42.35.142229
14	131072	0	131072	2004-02-06-16.42.36.616209
15	-57344	0	-57344	2004-02-06-16.42.39.302391
18	53248	0	53248	2004-02-06-16.42.44.261540
19	65536	0	65536	2004-02-06-16.42.46.729722

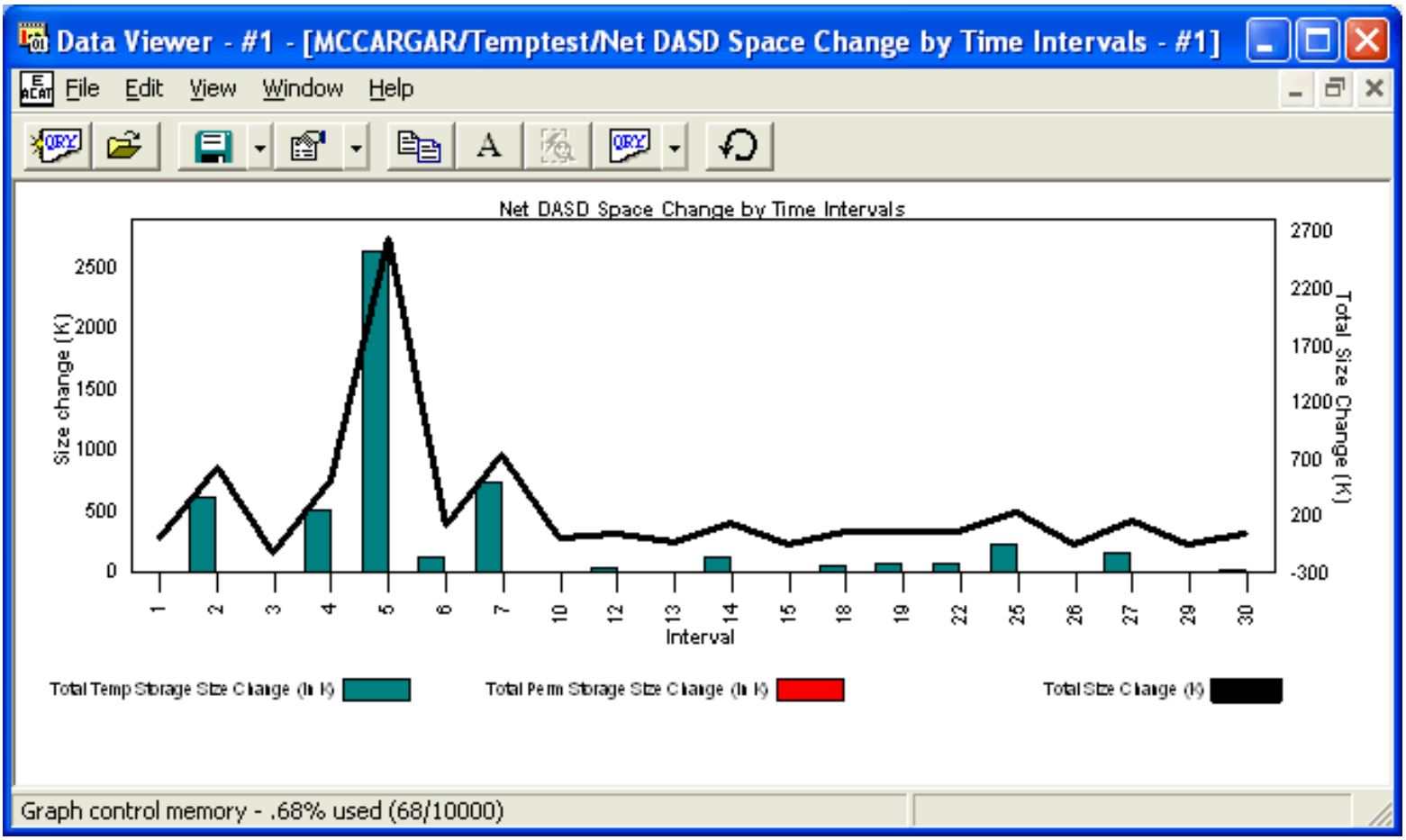
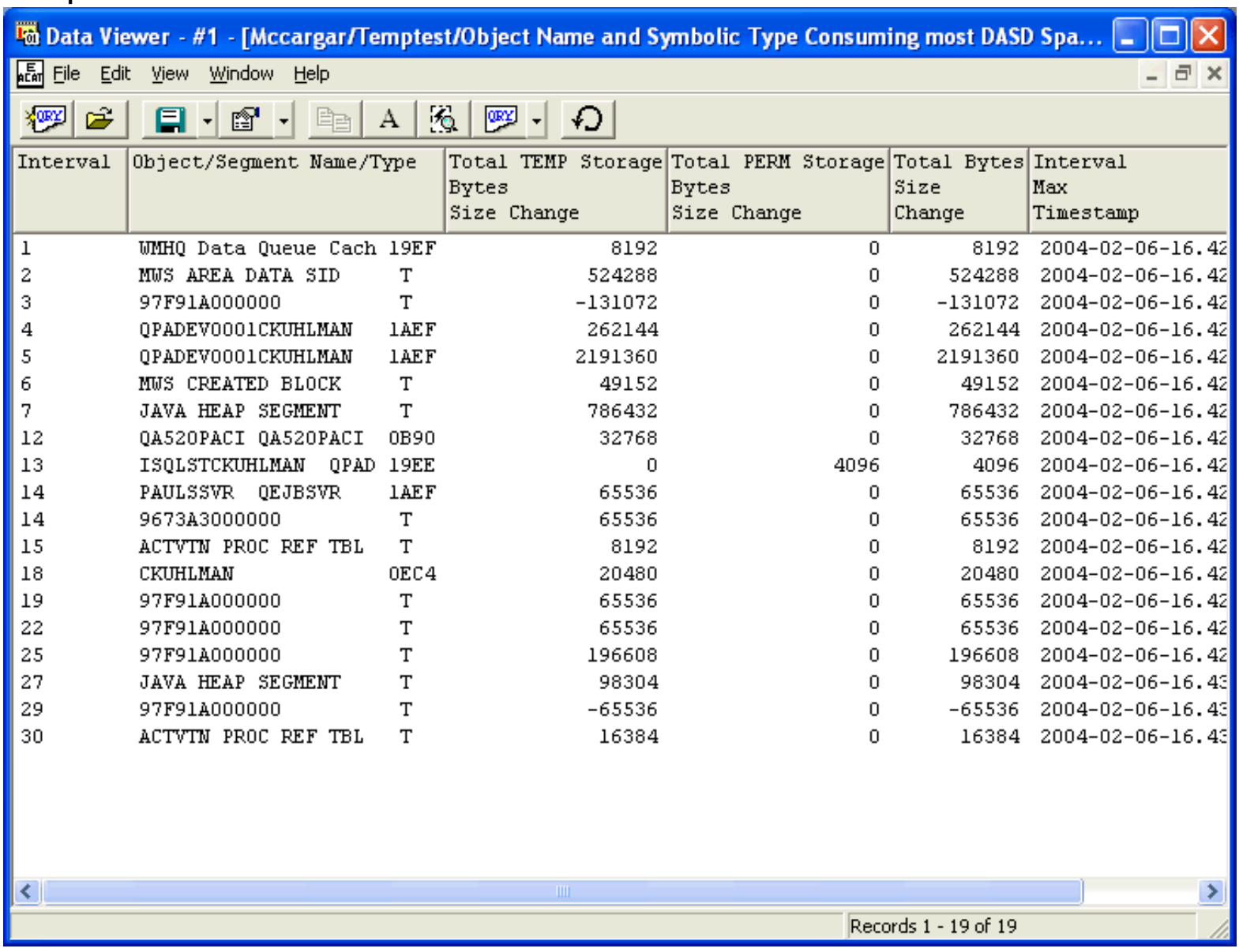

Data Viewer - #1 - [MCCARGAR/Temptest/Net DASD Space Change by Time Intervals - #1]
File Edit View Window Help
Net DASD Space Change by Time Intervals
Size change (K)
Total Size Change (K)
Interval
Total Temp Storage Size Change (In K)
Total Perm Storage Size Change (In K)
Total Size Change (K)
Graph control memory - .68% used (68/10000)

Table of Contents | Previous | Next

5.6.6 Object Name and Symbolic Type Consuming most DASD Space per Interval

Description:

Example:



Interval	Object/Segment Name/Type	Total TEMP Storage Bytes Size Change	Total PERM Storage Bytes Size Change	Total Bytes Size Change	Interval Max Timestamp
1	WMHQ Data Queue Cach 19EF	8192	0	8192	2004-02-06-16.42
2	MWS AREA DATA SID T	524288	0	524288	2004-02-06-16.42
3	97F91A000000 T	-131072	0	-131072	2004-02-06-16.42
4	QPADEV0001CKUHLMAN 1AEF	262144	0	262144	2004-02-06-16.42
5	QPADEV0001CKUHLMAN 1AEF	2191360	0	2191360	2004-02-06-16.42
6	MWS CREATED BLOCK T	49152	0	49152	2004-02-06-16.42
7	JAVA HEAP SEGMENT T	786432	0	786432	2004-02-06-16.42
12	QA520PACI QA520PACI 0B90	32768	0	32768	2004-02-06-16.42
13	ISQLSTCKUHLMAN QPAD 19EE	0	4096	4096	2004-02-06-16.42
14	PAULSSVR QEJBSVR 1AEF	65536	0	65536	2004-02-06-16.42
14	9673A3000000 T	65536	0	65536	2004-02-06-16.42
15	ACTVTN PROC REF TBL T	8192	0	8192	2004-02-06-16.42
18	CKUHLMAN 0EC4	20480	0	20480	2004-02-06-16.42
19	97F91A000000 T	65536	0	65536	2004-02-06-16.42
22	97F91A000000 T	65536	0	65536	2004-02-06-16.42
25	97F91A000000 T	196608	0	196608	2004-02-06-16.42
27	JAVA HEAP SEGMENT T	98304	0	98304	2004-02-06-16.43
29	97F91A000000 T	-65536	0	-65536	2004-02-06-16.43
30	ACTVTN PROC REF TBL T	16384	0	16384	2004-02-06-16.43

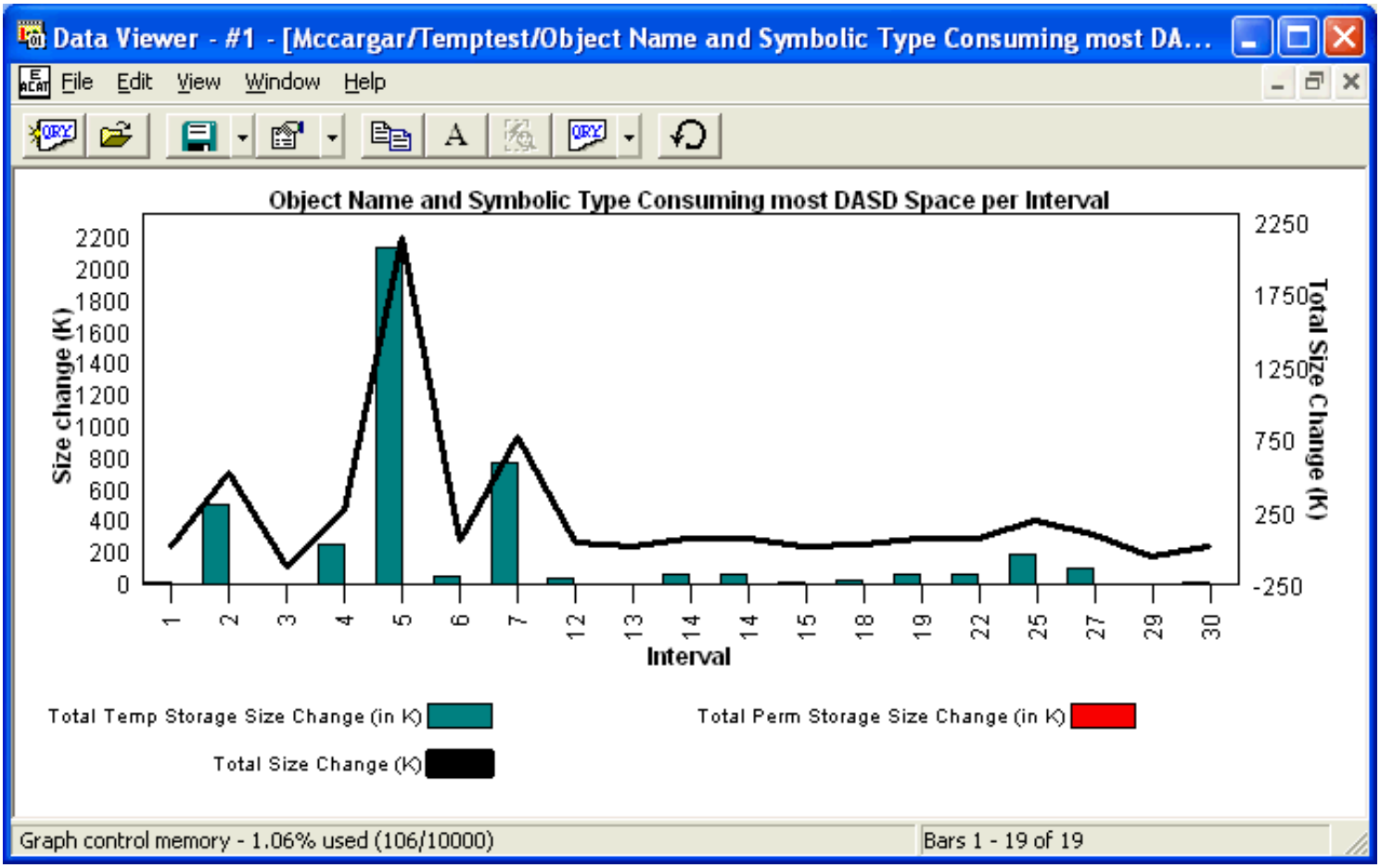
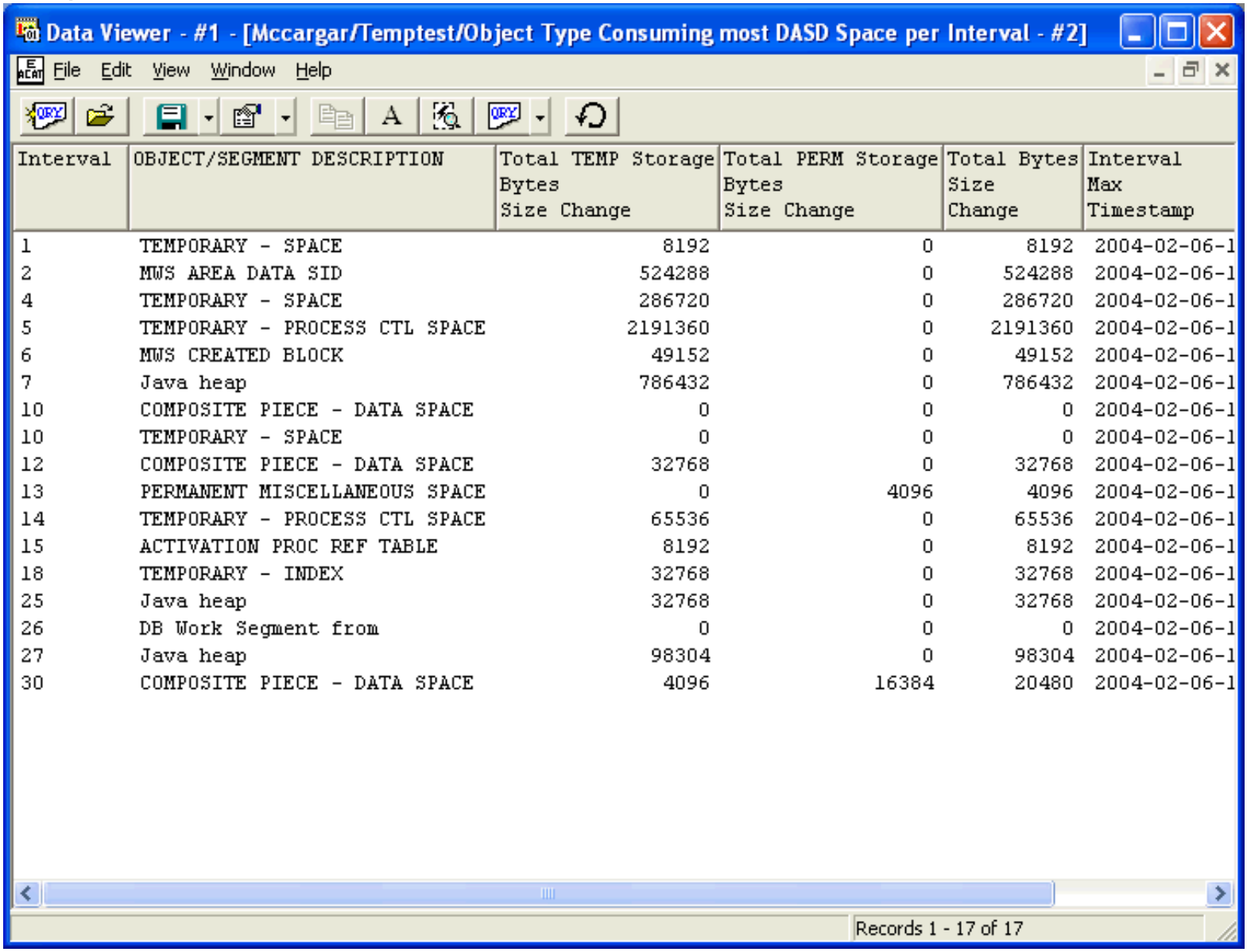

Data Viewer - #1 - [Mccargar/Temptest/Object Name and Symbolic Type Consuming most DA...
File Edit View Window Help
Object Name and Symbolic Type Consuming most DASD Space per Interval
Size change (K)
2200
2000
1800
1600
1400
1200
1000
800
600
400
200
0
2250
1750
1250
750
250
-250
Total Size Change (K)
1 2 3 4 5 6 7 12 13 14 14 15 18 19 22 25 27 29 30
Interval
Total Temp Storage Size Change (in K)
Total Perm Storage Size Change (in K)
Total Size Change (K)
Graph control memory - 1.06% used (106/10000)
Bars 1 - 19 of 19

Table of Contents | Previous | Next

5.6.7 Object Type Consuming most DASD Space per Interval

Description: This report enables the user to know the Dasd space changes for each interval of an analysis and the amount of change and the object/segment type causing it. Within the graphs the user can click on any bar (temporary or permanent space change) and get information on the interval, space change, the object/segment type causing it, and the date-time stamp.

Example:



Interval	OBJECT/SEGMENT DESCRIPTION	Total TEMP Storage Bytes Size Change	Total PERM Storage Bytes Size Change	Total Bytes Size Change	Interval Max Timestamp
1	TEMPORARY - SPACE	8192	0	8192	2004-02-06-1
2	MWS AREA DATA SID	524288	0	524288	2004-02-06-1
4	TEMPORARY - SPACE	286720	0	286720	2004-02-06-1
5	TEMPORARY - PROCESS CTL SPACE	2191360	0	2191360	2004-02-06-1
6	MWS CREATED BLOCK	49152	0	49152	2004-02-06-1
7	Java heap	786432	0	786432	2004-02-06-1
10	COMPOSITE PIECE - DATA SPACE	0	0	0	2004-02-06-1
10	TEMPORARY - SPACE	0	0	0	2004-02-06-1
12	COMPOSITE PIECE - DATA SPACE	32768	0	32768	2004-02-06-1
13	PERMANENT MISCELLANEOUS SPACE	0	4096	4096	2004-02-06-1
14	TEMPORARY - PROCESS CTL SPACE	65536	0	65536	2004-02-06-1
15	ACTIVATION PROC REF TABLE	8192	0	8192	2004-02-06-1
18	TEMPORARY - INDEX	32768	0	32768	2004-02-06-1
25	Java heap	32768	0	32768	2004-02-06-1
26	DB Work Segment from	0	0	0	2004-02-06-1
27	Java heap	98304	0	98304	2004-02-06-1
30	COMPOSITE PIECE - DATA SPACE	4096	16384	20480	2004-02-06-1

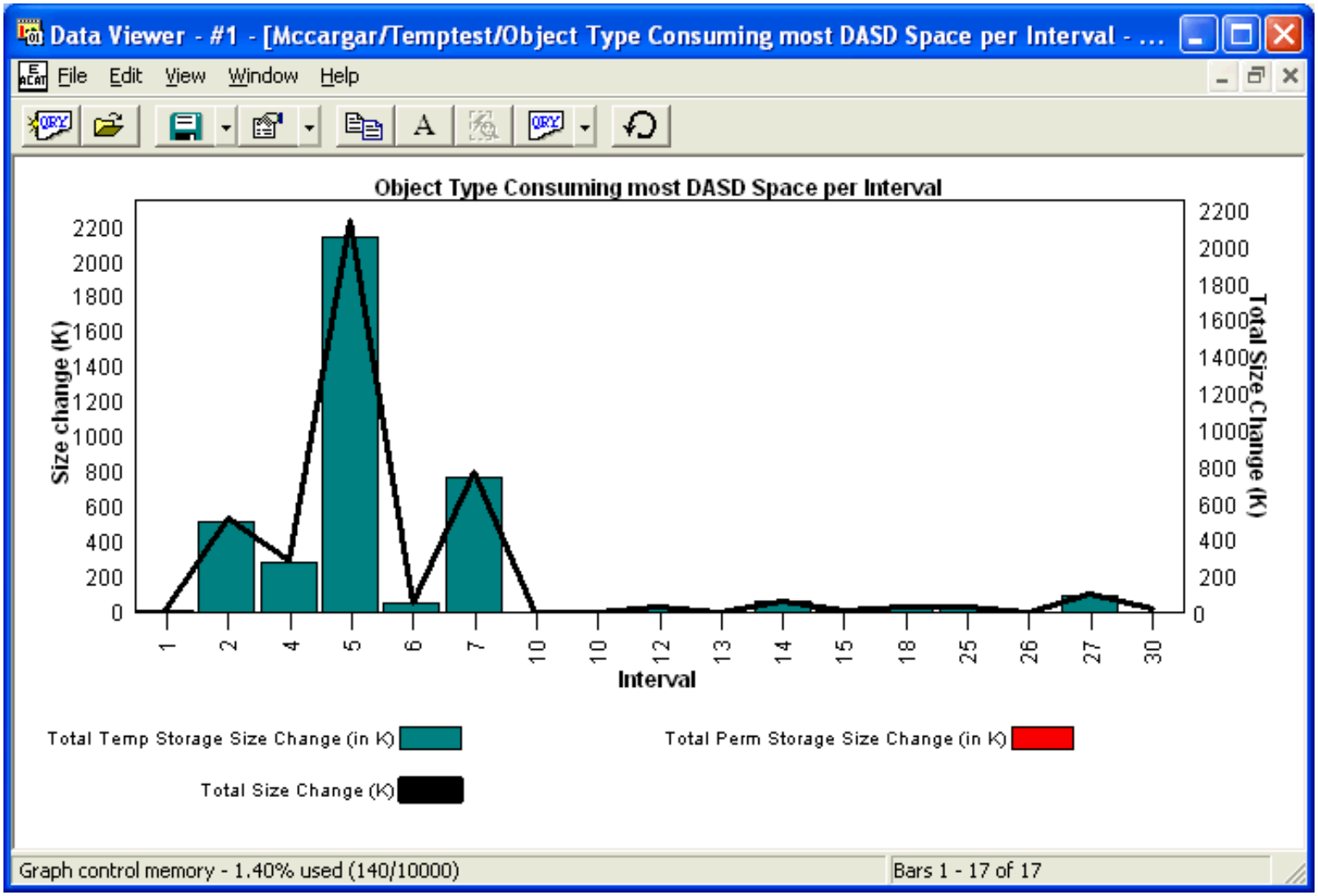

Data Viewer - #1 - [Mccargar/Temptest/Object Type Consuming most DASD Space per Interval - ...
File Edit View Window Help
Object Type Consuming most DASD Space per Interval
Size change (K)
Total Size Change (K)
Interval
2200
2000
1800
1600
1400
1200
1000
800
600
400
200
0
1 2 4 5 6 7 10 10 12 13 14 15 18 25 26 27 30
Total Temp Storage Size Change (in K)
Total Perm Storage Size Change (in K)
Total Size Change (K)
Graph control memory - 1.40% used (140/10000)
Bars 1 - 17 of 17

Table of Contents Previous Next

5.7 Database file full opens/closes reports

This analysis traces all full opens and closes showing the causing job/thread, and file/member name.

If MIENTRY and MIEXIT events are included in the collection, the analysis will also include the causing program name, elapsed time, and cpu time of each open/close. This analysis currently only supports local files (not DDM).

Table of Contents | Previous | Next

5.7.1 Local DB File Full Closes by File-Library within Interval

Description:

Example:

Table of Contents | Previous | Next

5.7.2 Local DB File Full Closes by File-Library within Job-Thread - No Intervals

Description:

Example:

Table of Contents | Previous | Next

5.7.3 Local DB File Full Closes by File-Library within Program - No Intervals

Description:

Example:

Table of Contents | Previous | Next

5.7.4 Local DB File Full Closes by Job within Interval

Description:

Example:

Table of Contents | Previous | Next

5.7.5 Local DB File Full Closes by Job-Thread within File-Library - No Intervals

Description:

Example:

Table of Contents | Previous | Next

5.7.6 Local DB File Full Closes by Job-Thread within Program - No Intervals

Description:

Example:

Table of Contents | Previous | Next

5.7.7 Local DB File Full Closes by Program within File-Library - No Intervals

Description:

Example:

Table of Contents | Previous | Next

5.7.8 Local DB File Full Closes by Program within Interval

Description:

Example:

Table of Contents | Previous | Next

5.7.9 Local DB File Full Closes Program within Job-Thread - No Intervals

Description:

Example:

Table of Contents Previous Next

5.7.10 Local DB File Full Opens by File-Library within Interval

Description:

Example:

Table of Contents | Previous | Next

5.7.11 Local DB File Full Opens by File-Library within Job-Thread - No Intervals

Description:

Example:

Table of Contents | Previous | Next

5.7.12 Local DB File Full Opens by File-Library within Program - No Intervals

Description:

Example:

Table of Contents | Previous | Next

5.7.13 Local DB File Full Opens by Job within Interval

Description:

Example:

Table of Contents | Previous | Next

5.7.14 Local DB File Full Opens by Job-Thread within File-Library - No Intervals

Description:

Example:

Table of Contents | Previous | Next

5.7.15 Local DB File Full Opens by Job-Thread within Program - No Intervals

Description:

Example:

Table of Contents | Previous | Next

5.7.16 Local DB File Full Opens by Program within File-Library - No Intervals

Description:

Example:

Table of Contents | Previous | Next

5.7.17 Local DB File Full Opens by Program within Interval

Description:

Example:

Table of Contents | Previous | Next

5.7.18 Local DB File Full Opens by Program within Job-Thread - No Intervals

Description:

Example:

Table of Contents Previous Next

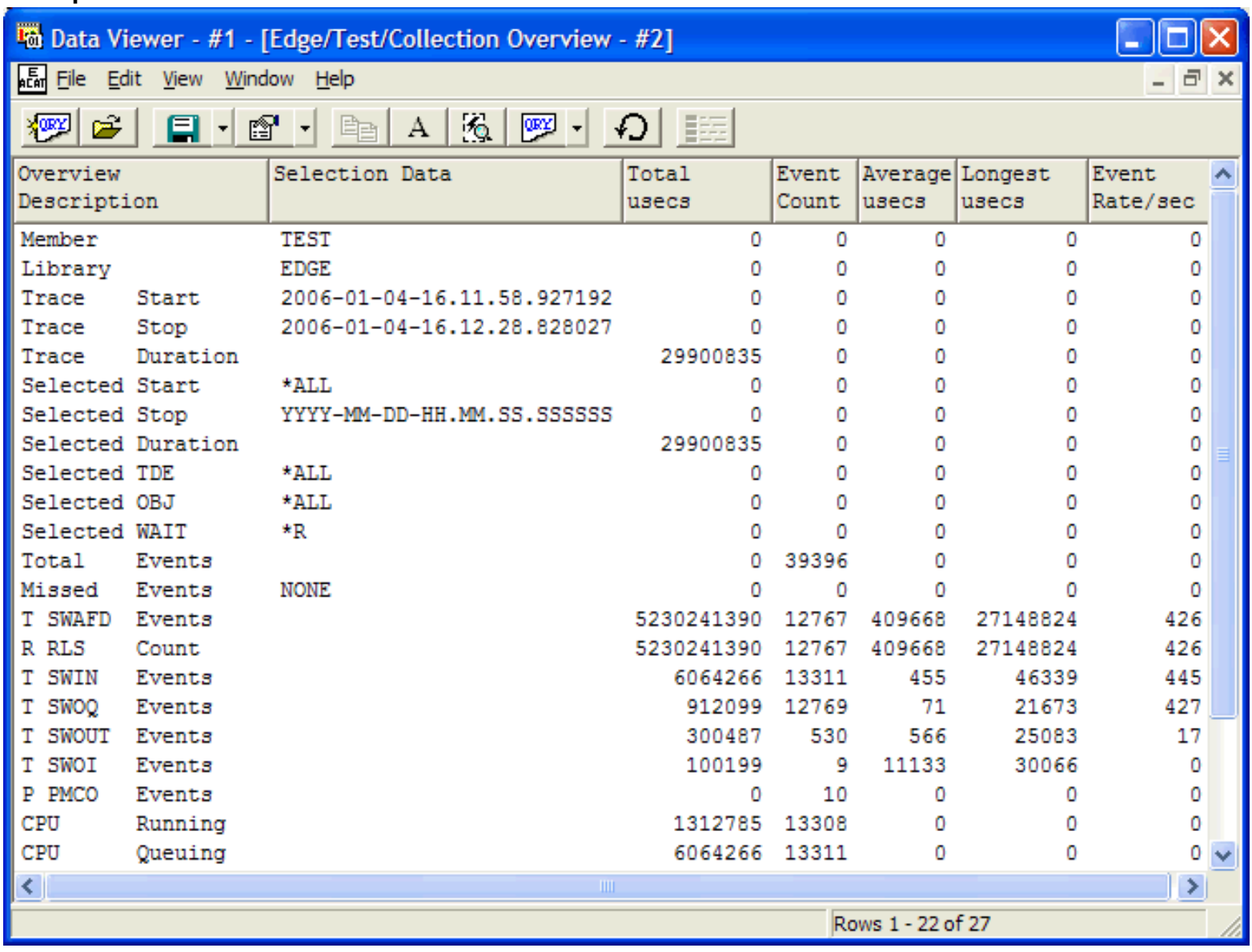
5.8 Task switch data analysis reports

Table of Contents | Previous | Next

5.8.1 Collection Overview

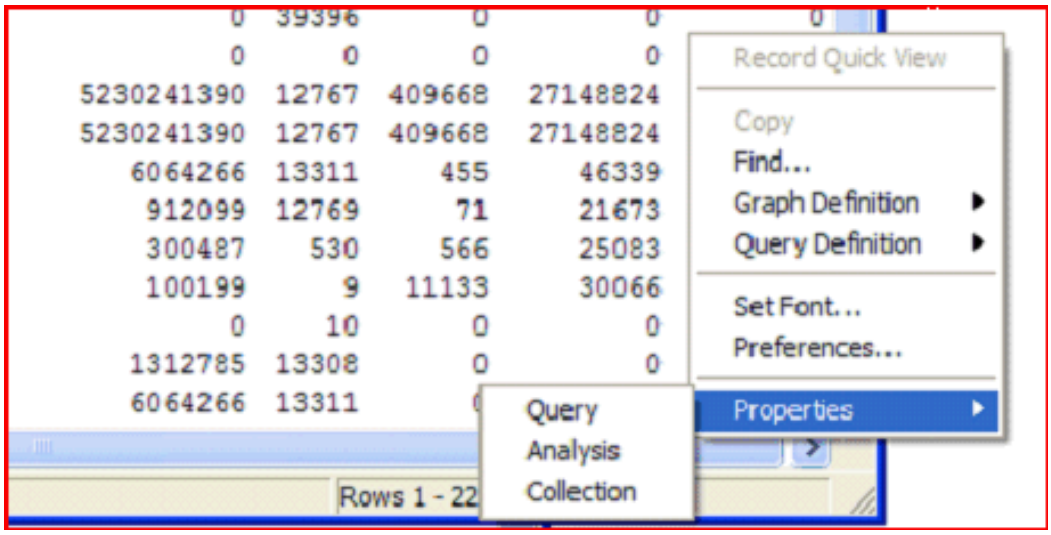
Description: The Collection Overview shows some totals, averages, rates and maximum values for certain categories of records as generated and recorded to new summary and detail files created by the Task Switch Analysis Program. It can be used to quickly assess the size and scope of the collection and set some expectations for further drill down efforts.

Example:



Overview Description		Selection Data	Total usecs	Event Count	Average usecs	Longest usecs	Event Rate/sec
Member		TEST	0	0	0	0	0
Library		EDGE	0	0	0	0	0
Trace	Start	2006-01-04-16.11.58.927192	0	0	0	0	0
Trace	Stop	2006-01-04-16.12.28.828027	0	0	0	0	0
Trace	Duration		29900835	0	0	0	0
Selected	Start	*ALL	0	0	0	0	0
Selected	Stop	YYYY-MM-DD-HH.MM.SS.SSSSSS	0	0	0	0	0
Selected	Duration		29900835	0	0	0	0
Selected	TDE	*ALL	0	0	0	0	0
Selected	OBJ	*ALL	0	0	0	0	0
Selected	WAIT	*R	0	0	0	0	0
Total	Events		0	39396	0	0	0
Missed	Events	NONE	0	0	0	0	0
T SWAFD	Events		5230241390	12767	409668	27148824	426
R RLS	Count		5230241390	12767	409668	27148824	426
T SWIN	Events		6064266	13311	455	46339	445
T SWOQ	Events		912099	12769	71	21673	427
T SWOUT	Events		300487	530	566	25083	17
T SWOI	Events		100199	9	11133	30066	0
P PMCO	Events		0	10	0	0	0
CPU	Running		1312785	13308	0	0	0
CPU	Queuing		6064266	13311	0	0	0

Note that further details about the collection, the files created by the Task Switch Analysis Program, and the query and files used to display the results in the data viewer can also be viewed from the Collection Overview display (as well as any of the Task Switch Analysis displays) by right clicking and selecting the Properties popup menu option.



The submenu options:

Query - This option shows the construct of the query and the underlying DB files and fields used for the current view.

Analysis - This option shows the parameters used in the analysis program and all the files created by the analysis.

Collection - This option shows the properties of the collection. See Chapter 4, PEX Definitions, Properties for a description of all the information presented within the tabs for the Collection option.

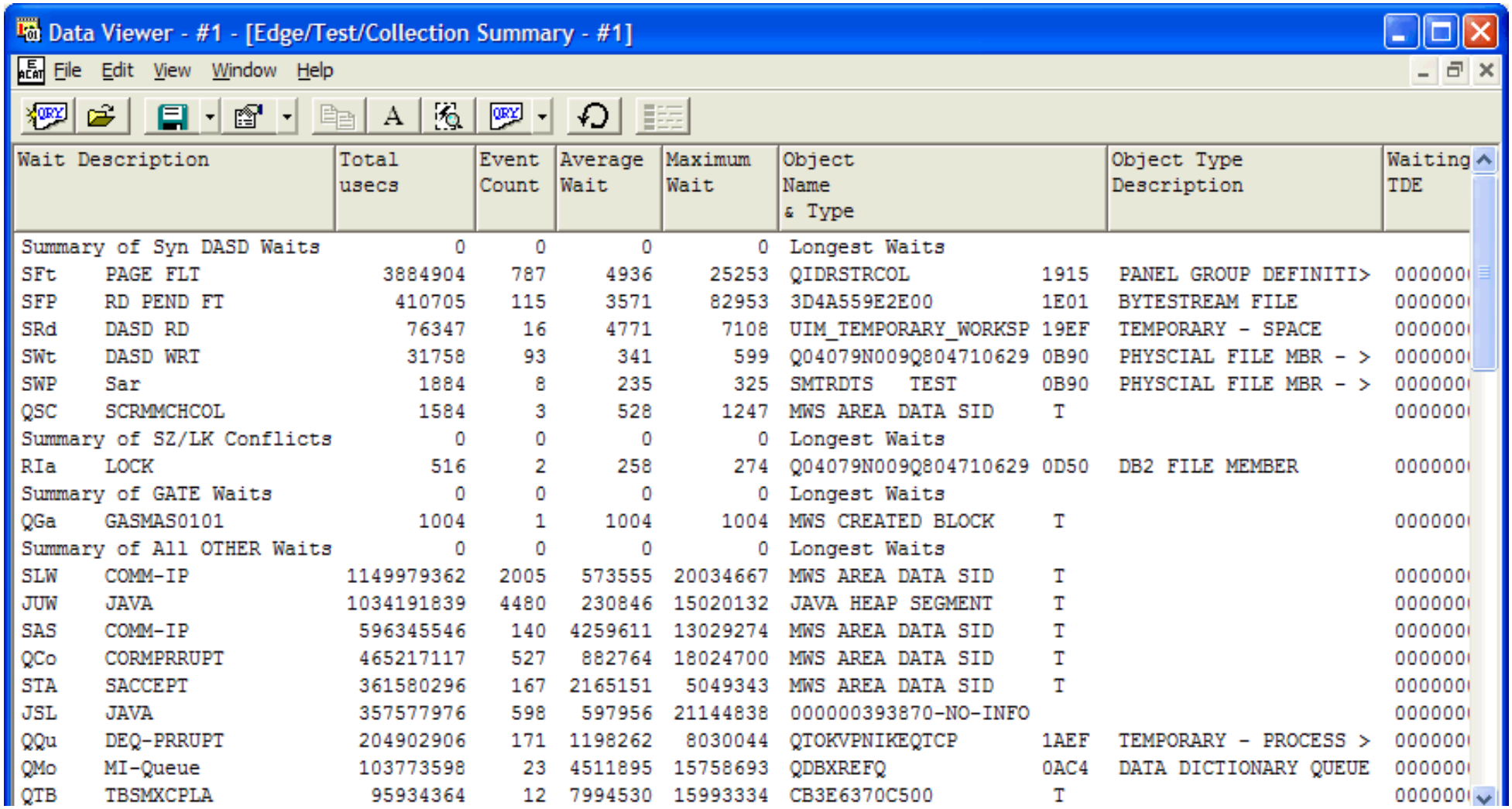
Note that the query can be modified through the Query Definition option.

Table of Contents | Previous | Next

5.8.2 Collection Summary

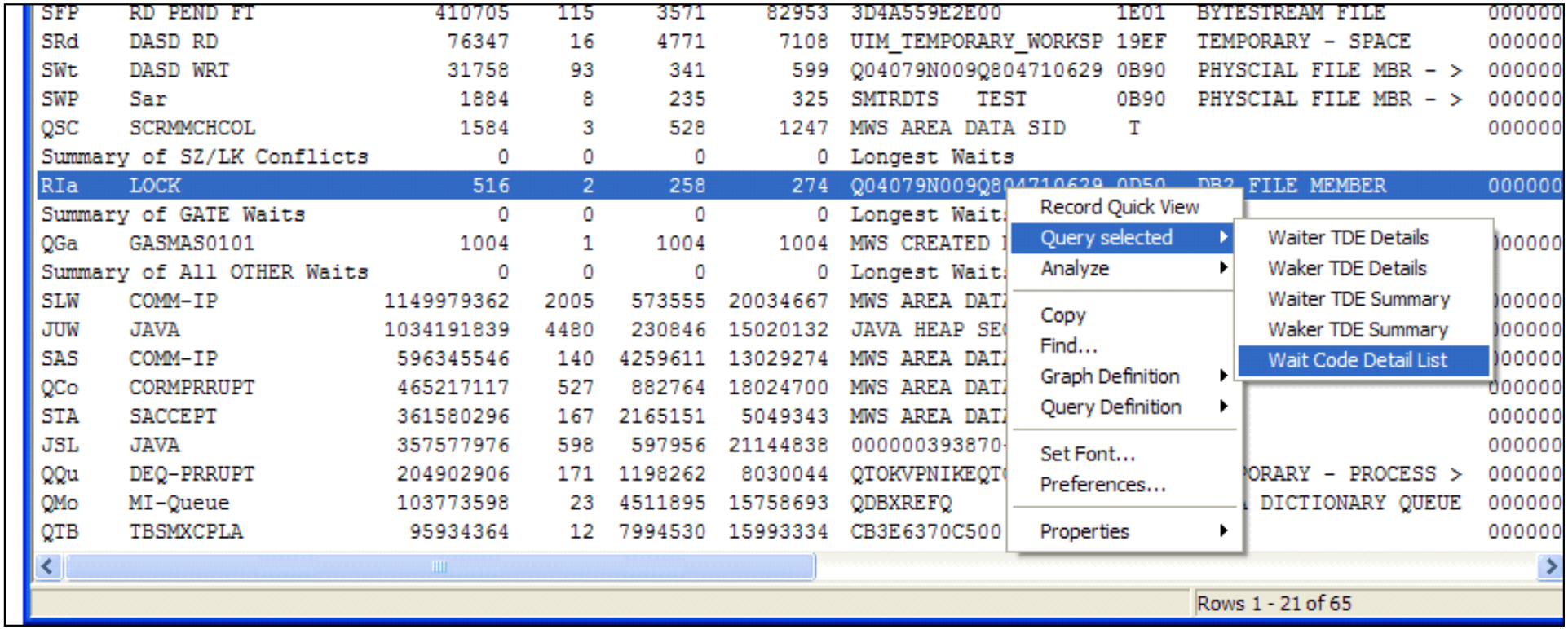
Description: The Collection Summary view is a good starting point if one is looking for the largest concentration of time spent in various categories of wait. The longest wait event for a category is also shown along with the waiter, the object name and the holder (when known). This display can be used to begin further detailed analysis (examine the investigative steps below which drill into the lock events shown in the example).

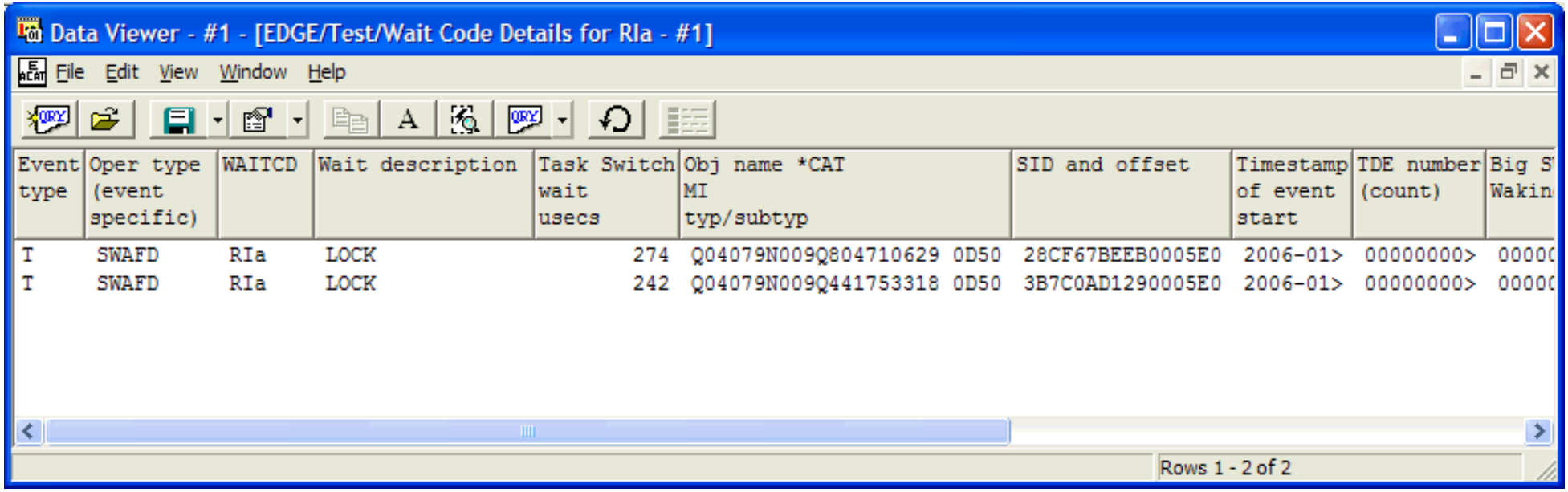
Example:



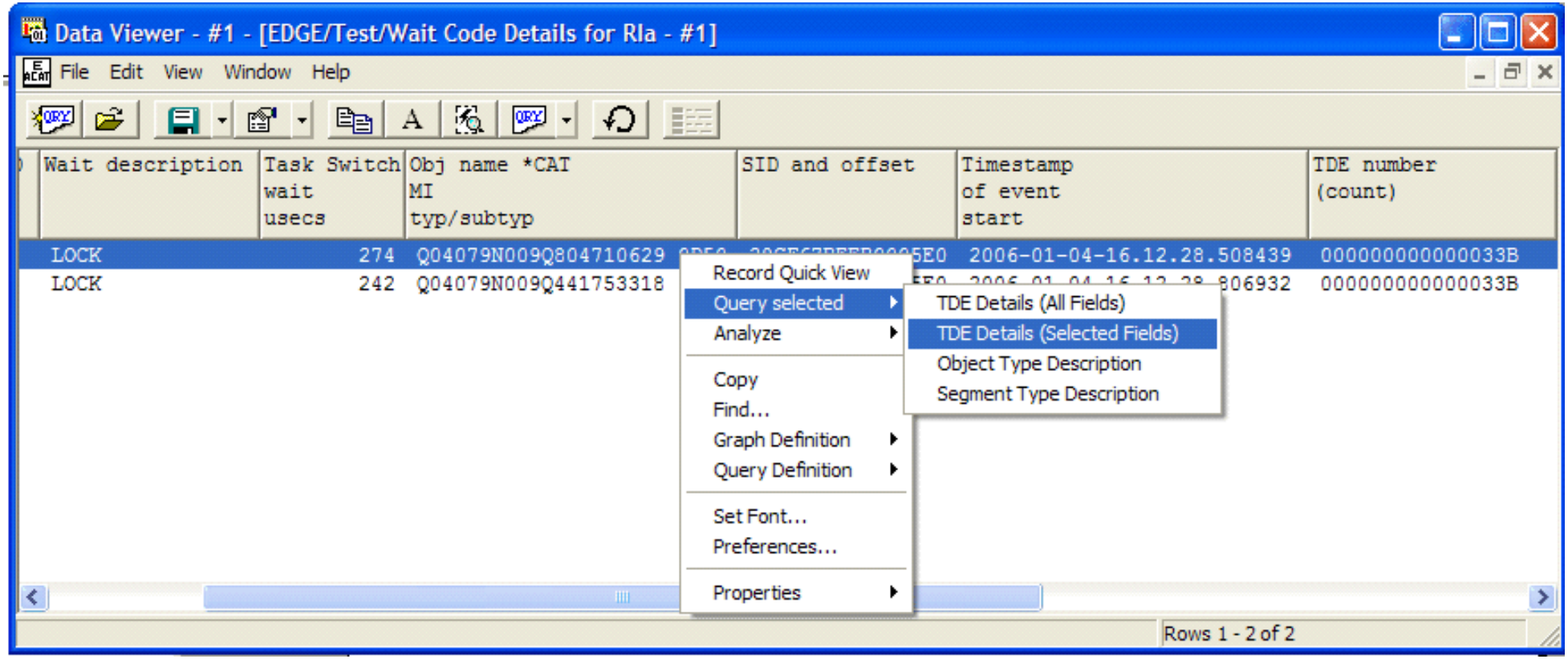
Wait Description	Total usecs	Event Count	Average Wait	Maximum Wait	Object Name & Type		Object Type Description	Waiting TDE
Summary of Syn DASD Waits	0	0	0	0	Longest Waits			
SFt PAGE FLT	3884904	787	4936	25253	QIDRSTRCOL	1915	PANEL GROUP DEFINITI>	000000
SFP RD PEND FT	410705	115	3571	82953	3D4A559E2E00	1E01	BYTESTREAM FILE	000000
SRd DASD RD	76347	16	4771	7108	UIM_TEMPORARY_WORKSP	19EF	TEMPORARY - SPACE	000000
SWt DASD WRT	31758	93	341	599	Q04079N009Q804710629	0B90	PHYSCIAL FILE MBR - >	000000
SWP Sar	1884	8	235	325	SMTRDTS TEST	0B90	PHYSCIAL FILE MBR - >	000000
QSC SCRMMCHCOL	1584	3	528	1247	MWS AREA DATA SID	T		000000
Summary of SZ/LK Conflicts	0	0	0	0	Longest Waits			
RIa LOCK	516	2	258	274	Q04079N009Q804710629	0D50	DB2 FILE MEMBER	000000
Summary of GATE Waits	0	0	0	0	Longest Waits			
QGa GASMAS0101	1004	1	1004	1004	MWS CREATED BLOCK	T		000000
Summary of All OTHER Waits	0	0	0	0	Longest Waits			
SLW COMM-IP	1149979362	2005	573555	20034667	MWS AREA DATA SID	T		000000
JUW JAVA	1034191839	4480	230846	15020132	JAVA HEAP SEGMENT	T		000000
SAS COMM-IP	596345546	140	4259611	13029274	MWS AREA DATA SID	T		000000
QCo CORMPRRUPT	465217117	527	882764	18024700	MWS AREA DATA SID	T		000000
STA SACCEPT	361580296	167	2165151	5049343	MWS AREA DATA SID	T		000000
JSL JAVA	357577976	598	597956	21144838	000000393870-NO-INFO			000000
QQu DEQ-PRRUPT	204902906	171	1198262	8030044	QTOKVPNIKEQTCP	1AEF	TEMPORARY - PROCESS >	000000
QMo MI-Queue	103773598	23	4511895	15758693	QDBXREFQ	0AC4	DATA DICTIONARY QUEUE	000000
QTB TBSMXCPLA	95934364	12	7994530	15993334	CB3E6370C500	T		000000

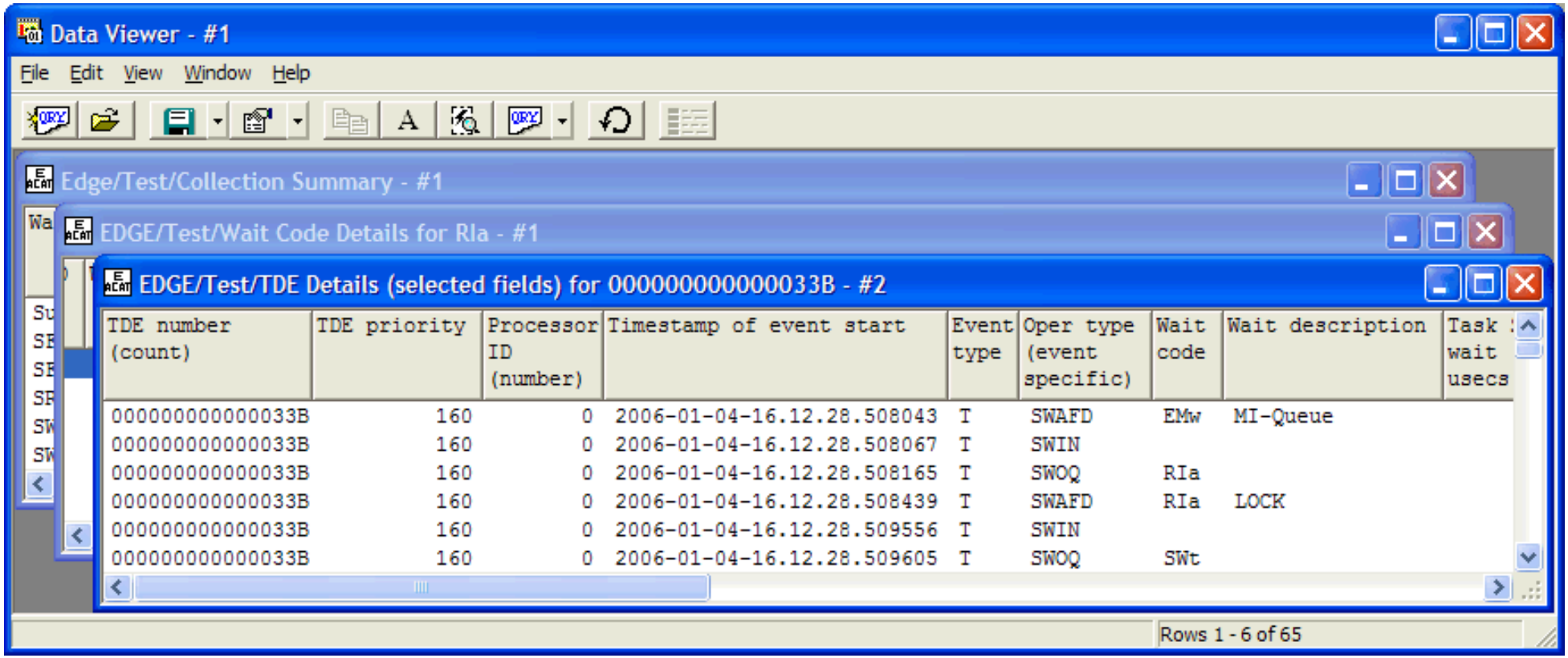
The example below shows selecting a wait category, right clicking to get to the popup menu, and selecting the option to display all the wait code details (trace events) for that category. The resulting display that follows and can be sorted and filtered even further by the user.





Even the Wait Code Details data can be used as an starting point to show all the trace events collected for a TDE (job). Just select an event record, right click to get a popup menu and pick the detail format that suits your needs from the submenu. The last display shows the trace event details for the selected TDE and the cascade of views that transpired to get to that level of detail.


Data Viewer - #1 - [EDGE/Test/Wait Code Details for Rla - #1]
File Edit View Window Help
Wait description
Task Switch wait usecs
Obj name *CAT MI typ/subtyp
SID and offset
Timestamp of event start
TDE number (count)
LOCK 274 Q04079N009Q804710629 2006-01-04-16.12.28.508439 000000000000033B
LOCK 242 Q04079N009Q441753318 000000000000033B
Record Quick View
Query selected
Analyze
Copy
Find...
Graph Definition
Query Definition
Set Font...
Preferences...
Properties
TDE Details (All Fields)
TDE Details (Selected Fields)
Object Type Description
Segment Type Description
Rows 1 - 2 of 2



TDE number (count)	TDE priority	Processor ID (number)	Timestamp of event start	Event type	Oper type (event specific)	Wait code	Wait description	Task wait usecs
000000000000033B	160	0	2006-01-04-16.12.28.508043	T	SWAFD	EMw	MI-Queue	
000000000000033B	160	0	2006-01-04-16.12.28.508067	T	SWIN			
000000000000033B	160	0	2006-01-04-16.12.28.508165	T	SWOQ	RIa		
000000000000033B	160	0	2006-01-04-16.12.28.508439	T	SWAFD	RIa	LOCK	
000000000000033B	160	0	2006-01-04-16.12.28.509556	T	SWIN			
000000000000033B	160	0	2006-01-04-16.12.28.509605	T	SWOQ	SWt		

Table of Contents Previous Next

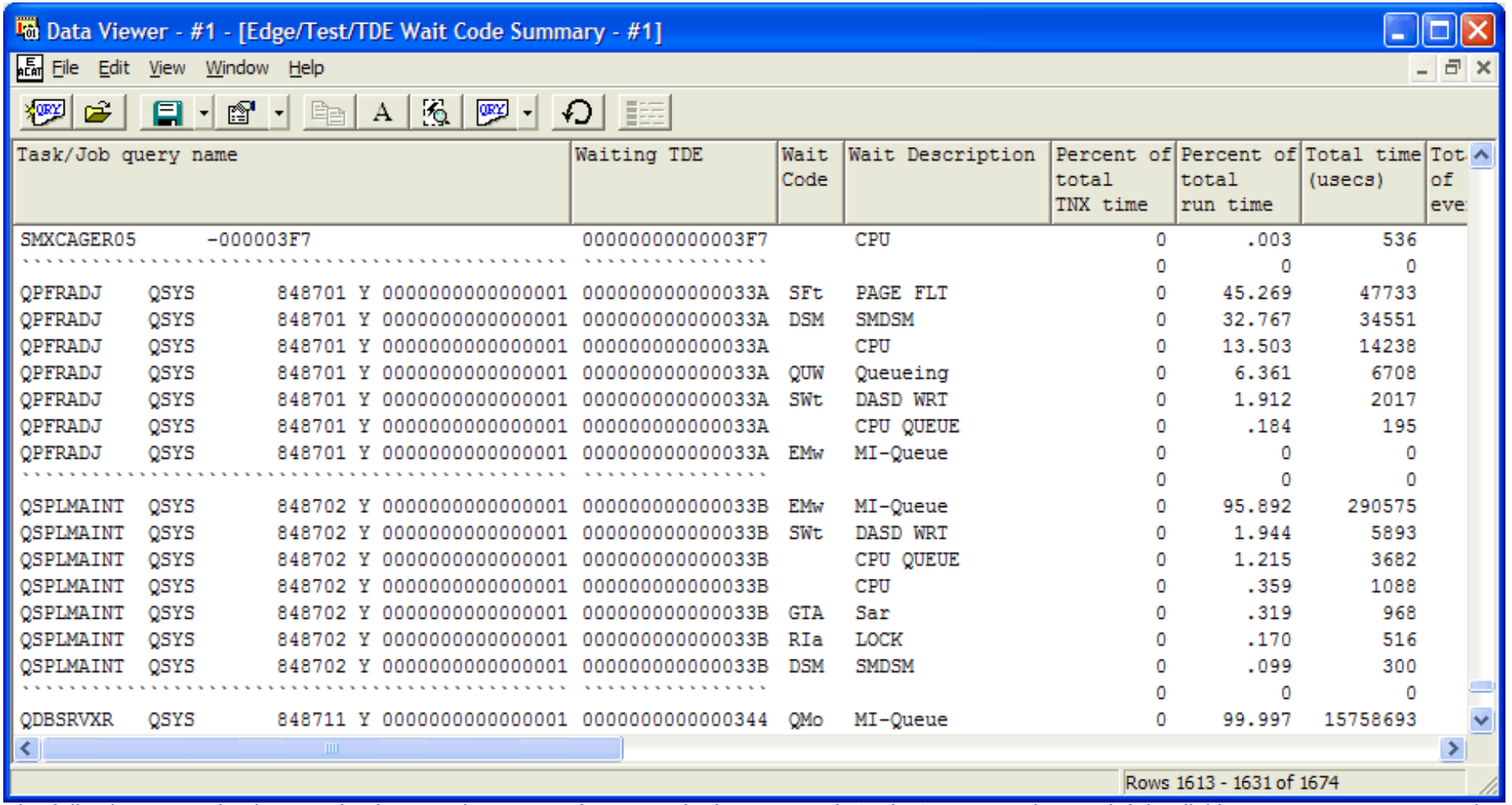
5.8.3 TDE Wait Code Summary

Description: The TDE Wait Code Summary shows a summarized list by wait code category for all TDEs included in the collection. By scrolling to the right you can see the counts for each wait code category as well as the average and longest event.

The list can be sorted and filtered through the Query Definition popup menu option.

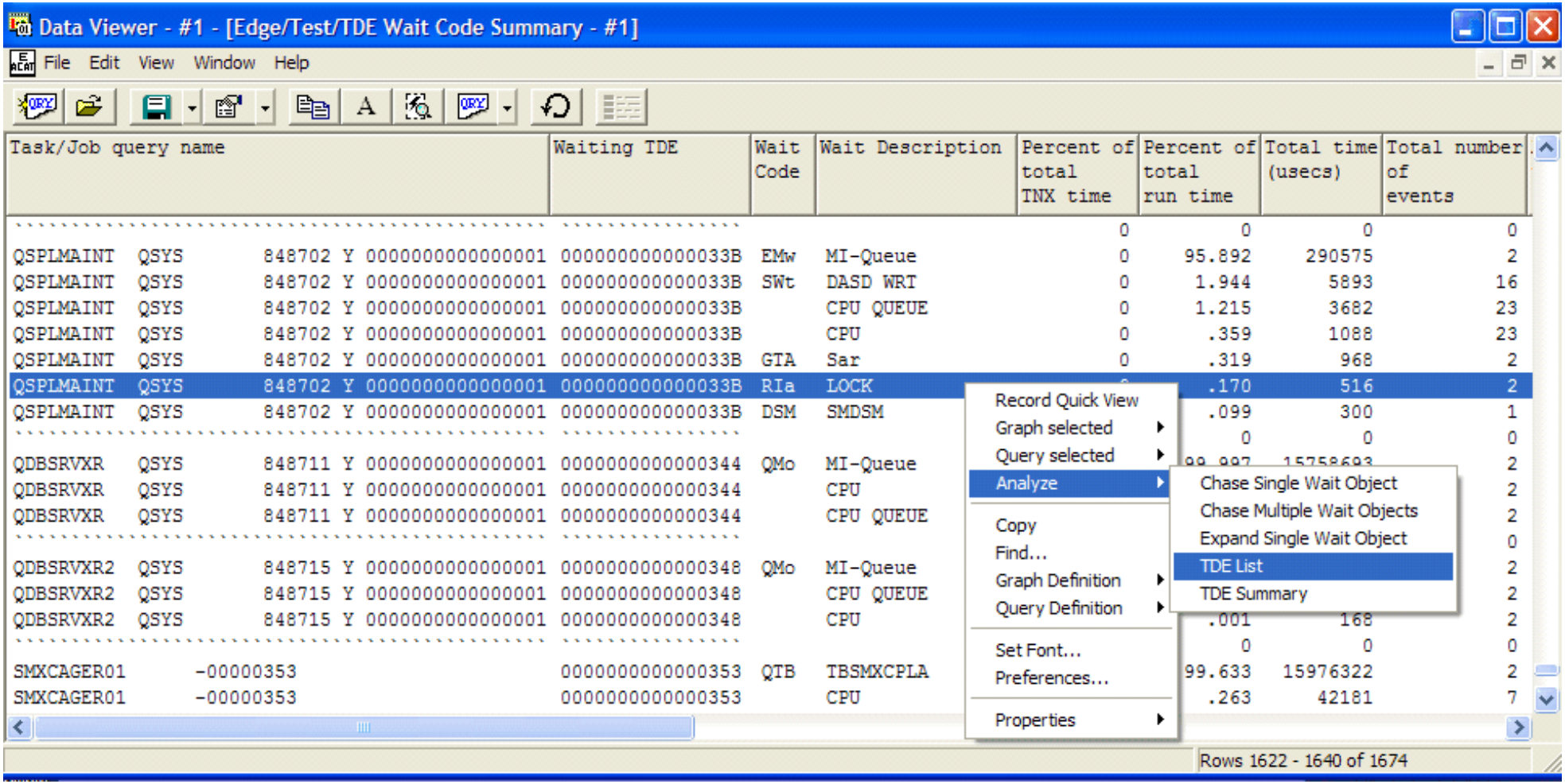
This list also can be used as a starting point to begin drilling down for more details through either the Query selected or Analyze options of the popup menu. Query selected options will produce results that are not saved but that can be sorted and filtered through user modification. Analysis options will produce a report, much like a printed report, that is also saved as a Child analysis report entry, and as such, can be reviewed without having to rerun any analysis programs or queries.

Example:



Task/Job query name				Waiting TDE	Wait Code	Wait Description	Percent of total TNX time	Percent of total run time	Total time (usecs)	Tot of eve
SMXCAGER05 -000003F7				00000000000003F7		CPU	0	.003	536	
							0	0	0	
QPFRADJ	QSYS	848701	Y 0000000000000001	000000000000033A	SFt	PAGE FLT	0	45.269	47733	
QPFRADJ	QSYS	848701	Y 0000000000000001	000000000000033A	DSM	SMDSM	0	32.767	34551	
QPFRADJ	QSYS	848701	Y 0000000000000001	000000000000033A		CPU	0	13.503	14238	
QPFRADJ	QSYS	848701	Y 0000000000000001	000000000000033A	QUW	Queueing	0	6.361	6708	
QPFRADJ	QSYS	848701	Y 0000000000000001	000000000000033A	SWt	DASD WRT	0	1.912	2017	
QPFRADJ	QSYS	848701	Y 0000000000000001	000000000000033A		CPU QUEUE	0	.184	195	
QPFRADJ	QSYS	848701	Y 0000000000000001	000000000000033A	EMw	MI-Queue	0	0	0	
							0	0	0	
QSPLMAINT	QSYS	848702	Y 0000000000000001	000000000000033B	EMw	MI-Queue	0	95.892	290575	
QSPLMAINT	QSYS	848702	Y 0000000000000001	000000000000033B	SWt	DASD WRT	0	1.944	5893	
QSPLMAINT	QSYS	848702	Y 0000000000000001	000000000000033B		CPU QUEUE	0	1.215	3682	
QSPLMAINT	QSYS	848702	Y 0000000000000001	000000000000033B		CPU	0	.359	1088	
QSPLMAINT	QSYS	848702	Y 0000000000000001	000000000000033B	GTA	Sar	0	.319	968	
QSPLMAINT	QSYS	848702	Y 0000000000000001	000000000000033B	RIa	LOCK	0	.170	516	
QSPLMAINT	QSYS	848702	Y 0000000000000001	000000000000033B	DSM	SMDSM	0	.099	300	
							0	0	0	
QDBSRVXR	QSYS	848711	Y 0000000000000001	0000000000000344	QMo	MI-Queue	0	99.997	15758693	

The following example shows selecting a wait category for a TDE in the TDE Wait Code Summary view and right clicking to get popup menus that allows the selection of Analyze and create a TDE List options. The results will be both displayed and saved to a file that can be reviewed through the Child analysis reports.


Data Viewer - #1 - [Edge/Test/TDE Wait Code Summary - #1]
File Edit View Window Help
Task/Job query name
Waiting TDE
Wait Code
Wait Description
Percent of total TNX time
Percent of total run time
Total time (usecs)
Total number of events
QSPLMAINT QSYS 848702 Y 0000000000000001 000000000000033B EMw MI-Queue 0 95.892 290575 2
QSPLMAINT QSYS 848702 Y 0000000000000001 000000000000033B SWt DASD WRT 0 1.944 5893 16
QSPLMAINT QSYS 848702 Y 0000000000000001 000000000000033B CPU QUEUE 0 1.215 3682 23
QSPLMAINT QSYS 848702 Y 0000000000000001 000000000000033B CPU 0 .359 1088 23
QSPLMAINT QSYS 848702 Y 0000000000000001 000000000000033B GTA Sar 0 .319 968 2
QSPLMAINT QSYS 848702 Y 0000000000000001 000000000000033B RIa LOCK .170 516 2
QSPLMAINT QSYS 848702 Y 0000000000000001 000000000000033B DSM SMDSM .099 300 1
QDBSRVXR QSYS 848711 Y 0000000000000001 0000000000000344 QMo MI-Queue 2
QDBSRVXR QSYS 848711 Y 0000000000000001 0000000000000344 CPU 2
QDBSRVXR QSYS 848711 Y 0000000000000001 0000000000000344 CPU QUEUE 2
QDBSRVXR2 QSYS 848715 Y 0000000000000001 0000000000000348 QMo MI-Queue 2
QDBSRVXR2 QSYS 848715 Y 0000000000000001 0000000000000348 CPU QUEUE 2
QDBSRVXR2 QSYS 848715 Y 0000000000000001 0000000000000348 CPU .001 168 2
SMXCAGER01 -00000353 0000000000000353 QTB TBSMXCPLA 99.633 15976322 2
SMXCAGER01 -00000353 0000000000000353 CPU .263 42181 7
Record Quick View
Graph selected
Query selected
Analyze
Copy
Find...
Graph Definition
Query Definition
Set Font...
Preferences...
Properties
Chase Single Wait Object
Chase Multiple Wait Objects
Expand Single Wait Object
TDE List
TDE Summary
Rows 1622 - 1640 of 1674

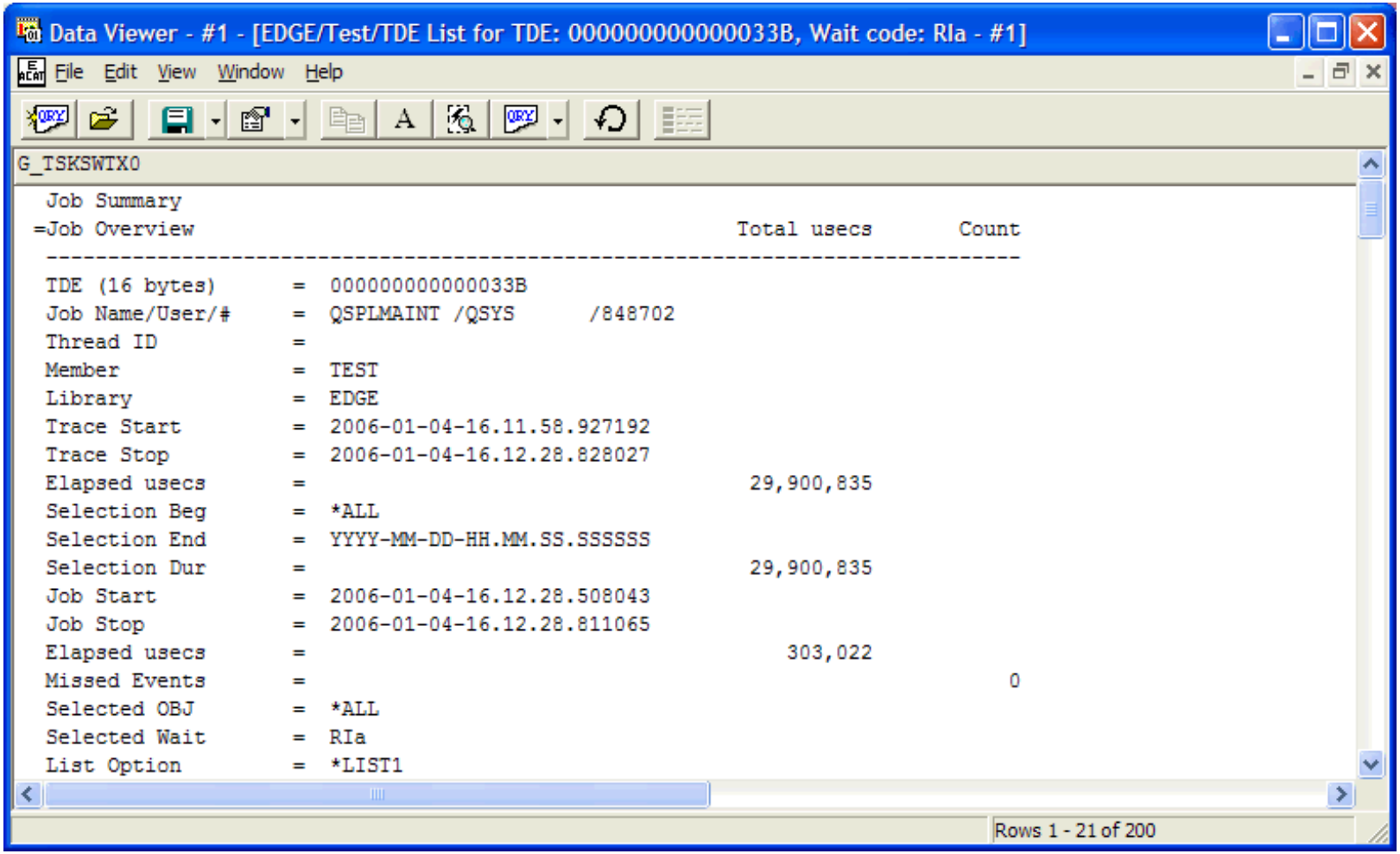

Data Viewer - #1 - [EDGE/Test/TDE List for TDE: 000000000000033B, Wait code: RIa - #1]
File Edit View Window Help
G_TSKSWTX0
Job Summary
=Job Overview Total usecs Count
TDE (16 bytes) = 000000000000033B
Job Name/User/# = QSPLMAINT /QSYS /848702
Thread ID =
Member = TEST
Library = EDGE
Trace Start = 2006-01-04-16.11.58.927192
Trace Stop = 2006-01-04-16.12.28.828027
Elapsed usecs = 29,900,835
Selection Beg = *ALL
Selection End = YYYY-MM-DD-HH.MM.SS.SSSSSS
Selection Dur = 29,900,835
Job Start = 2006-01-04-16.12.28.508043
Job Stop = 2006-01-04-16.12.28.811065
Elapsed usecs = 303,022
Missed Events = 0
Selected OBJ = *ALL
Selected Wait = RIa
List Option = *LIST1
Rows 1 - 21 of 200

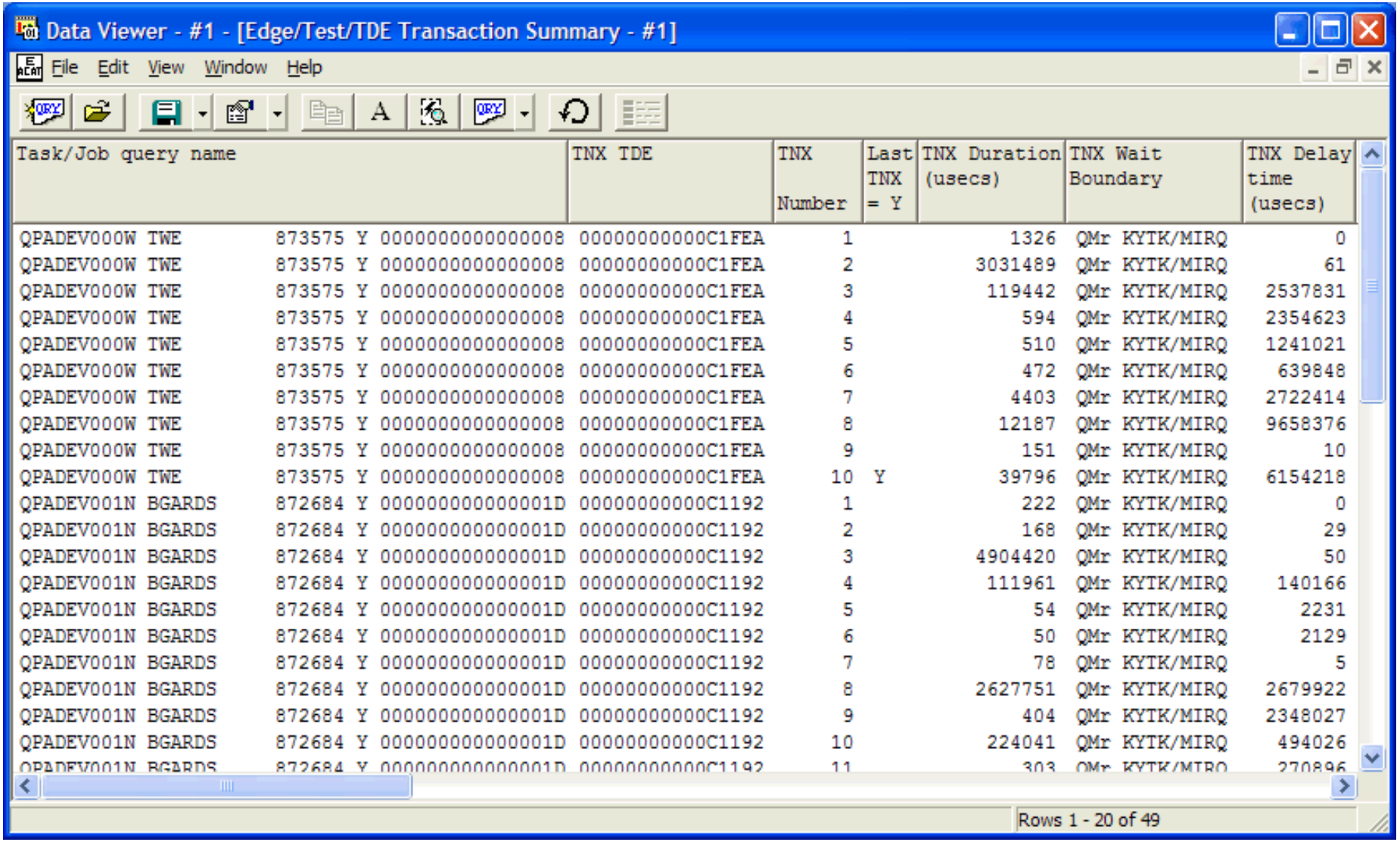
Table of Contents Previous Next

5.8.4 TDE Transaction Summary

Description: The Transaction Summary list shows transactions for interactive jobs where a wait on a QMr wait code event is determined to be a transaction boundary, that is, a transaction has completed and the job is waiting for more work requests to arrive. The transactions are given a sequential reference number for each TDE.

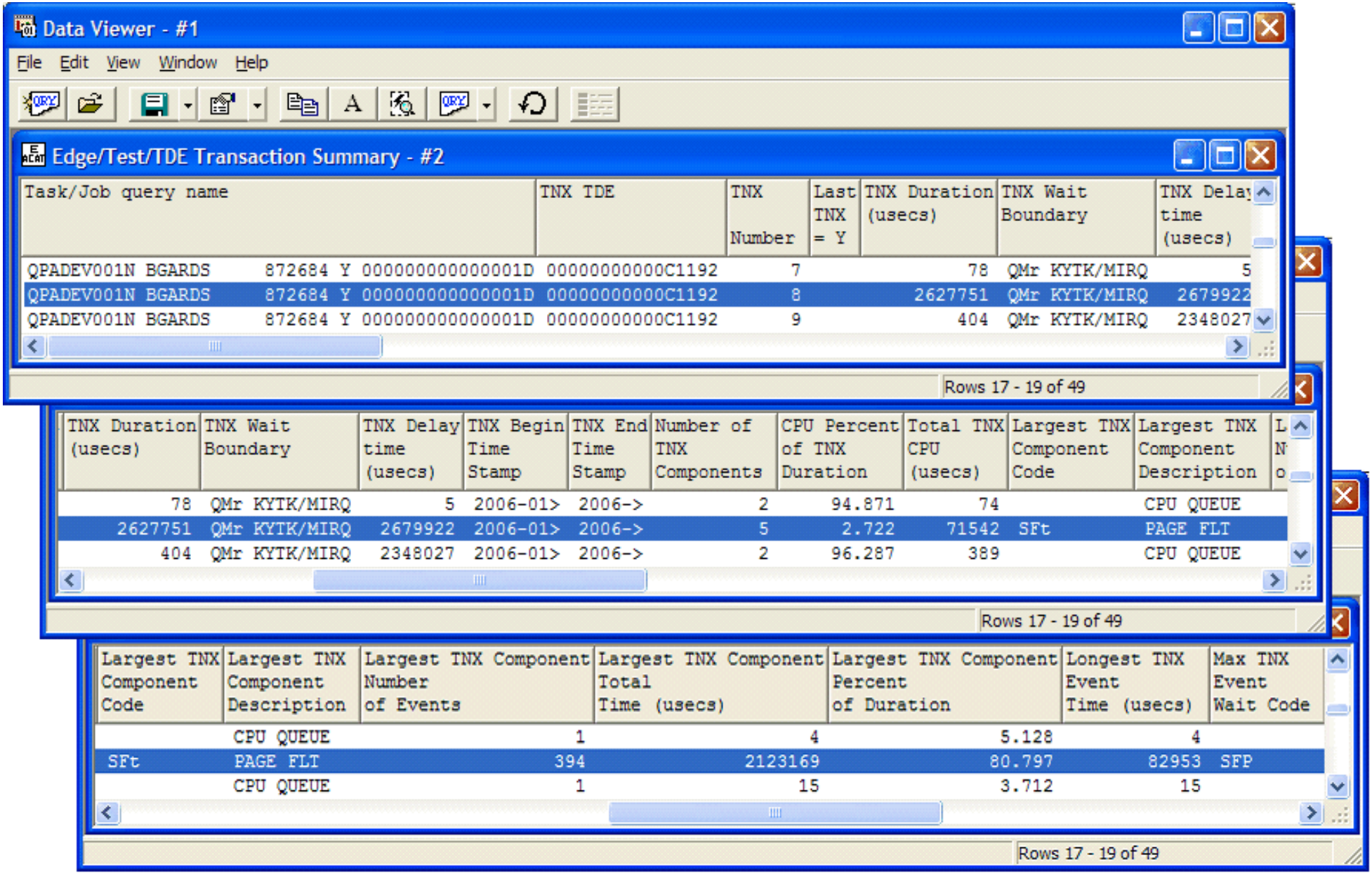
Each transaction is made up of run/wait components. Scrolling to the right will show the CPU used component along with the largest wait component for the transactions total duration (see the second example which shows a series of displays with a selected transaction highlighted).

Example:

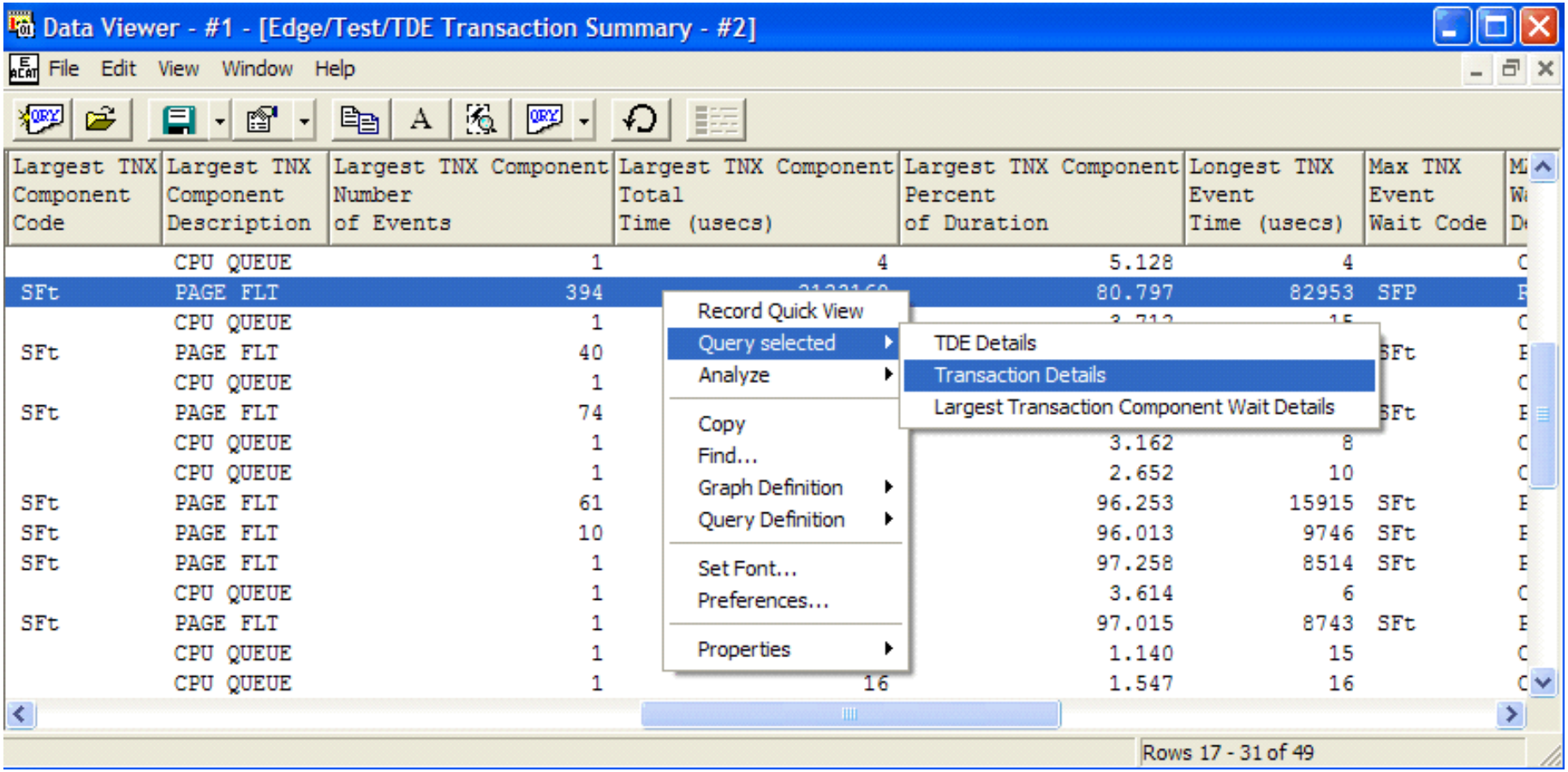


Task/Job query name	TNX TDE	TNX Number	Last TNX = Y	TNX Duration (usecs)	TNX Wait Boundary	TNX Delay time (usecs)
QPADEV000W TWE 873575 Y 0000000000000008	00000000000C1FEA	1		1326	QMr KYTK/MIRQ	0
QPADEV000W TWE 873575 Y 0000000000000008	00000000000C1FEA	2		3031489	QMr KYTK/MIRQ	61
QPADEV000W TWE 873575 Y 0000000000000008	00000000000C1FEA	3		119442	QMr KYTK/MIRQ	2537831
QPADEV000W TWE 873575 Y 0000000000000008	00000000000C1FEA	4		594	QMr KYTK/MIRQ	2354623
QPADEV000W TWE 873575 Y 0000000000000008	00000000000C1FEA	5		510	QMr KYTK/MIRQ	1241021
QPADEV000W TWE 873575 Y 0000000000000008	00000000000C1FEA	6		472	QMr KYTK/MIRQ	639848
QPADEV000W TWE 873575 Y 0000000000000008	00000000000C1FEA	7		4403	QMr KYTK/MIRQ	2722414
QPADEV000W TWE 873575 Y 0000000000000008	00000000000C1FEA	8		12187	QMr KYTK/MIRQ	9658376
QPADEV000W TWE 873575 Y 0000000000000008	00000000000C1FEA	9		151	QMr KYTK/MIRQ	10
QPADEV000W TWE 873575 Y 0000000000000008	00000000000C1FEA	10	Y	39796	QMr KYTK/MIRQ	6154218
QPADEV001N BGARDS 872684 Y 000000000000001D	00000000000C1192	1		222	QMr KYTK/MIRQ	0
QPADEV001N BGARDS 872684 Y 000000000000001D	00000000000C1192	2		168	QMr KYTK/MIRQ	29
QPADEV001N BGARDS 872684 Y 000000000000001D	00000000000C1192	3		4904420	QMr KYTK/MIRQ	50
QPADEV001N BGARDS 872684 Y 000000000000001D	00000000000C1192	4		111961	QMr KYTK/MIRQ	140166
QPADEV001N BGARDS 872684 Y 000000000000001D	00000000000C1192	5		54	QMr KYTK/MIRQ	2231
QPADEV001N BGARDS 872684 Y 000000000000001D	00000000000C1192	6		50	QMr KYTK/MIRQ	2129
QPADEV001N BGARDS 872684 Y 000000000000001D	00000000000C1192	7		78	QMr KYTK/MIRQ	5
QPADEV001N BGARDS 872684 Y 000000000000001D	00000000000C1192	8		2627751	QMr KYTK/MIRQ	2679922
QPADEV001N BGARDS 872684 Y 000000000000001D	00000000000C1192	9		404	QMr KYTK/MIRQ	2348027
QPADEV001N BGARDS 872684 Y 000000000000001D	00000000000C1192	10		224041	QMr KYTK/MIRQ	494026
QPADEV001N BGARDS 872684 Y 000000000000001D	00000000000C1192	11		303	QMr KYTK/MIRQ	270896

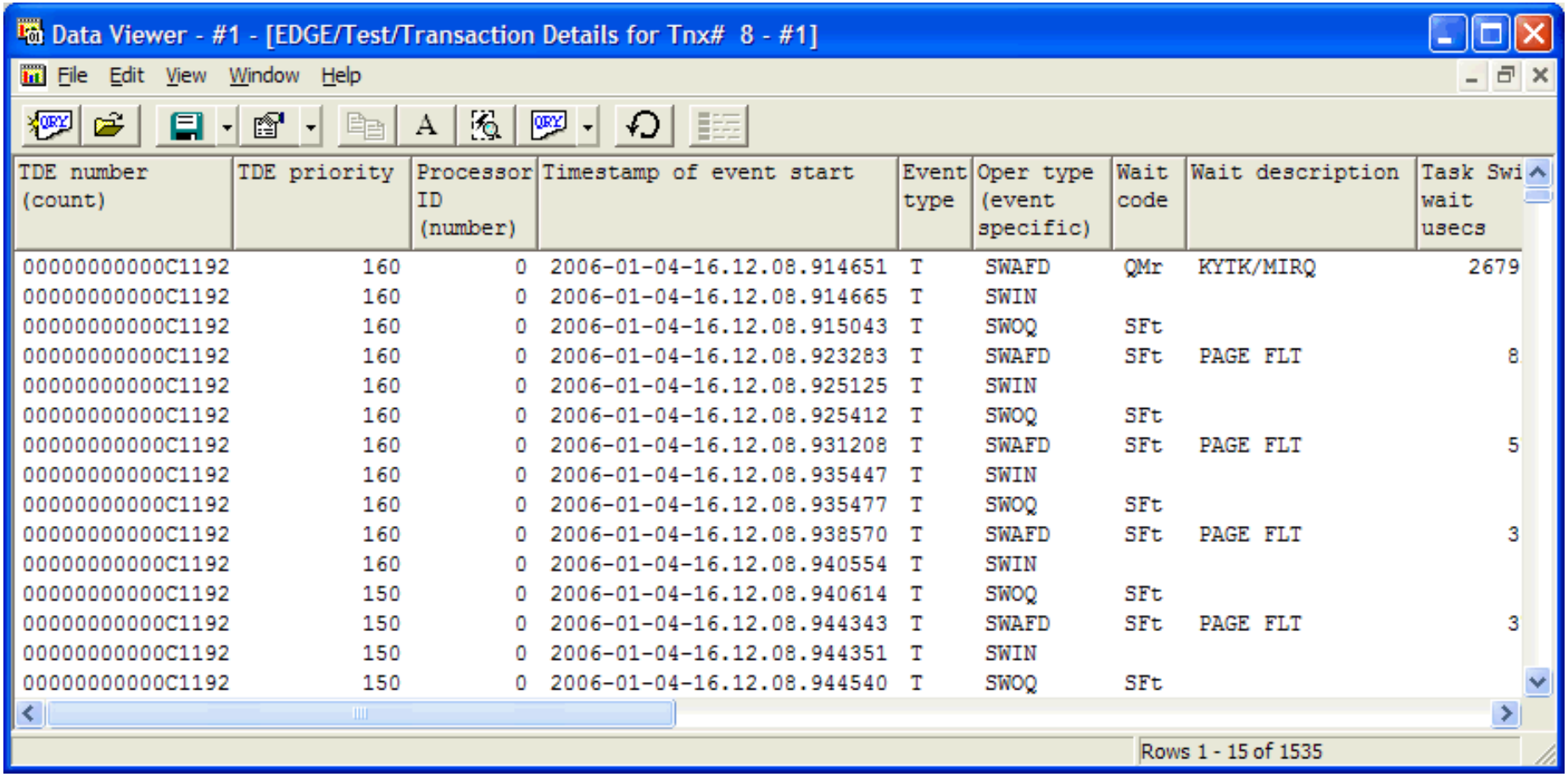
Rows 1 - 20 of 49



A selected transaction can be examined in further detail by using the right click popup menu and taking the Query selected, Transaction Details options as shown below.



Each of the 394 SFt page faults that comprise the transaction's SFt wait component can now be reviewed with regard to duration, object name and type, request size and more. The sequence of events is known and the CPU used and the time spent in CPU queuing as well.



TDE number (count)	TDE priority	Processor ID (number)	Timestamp of event start	Event type	Oper type (event specific)	Wait code	Wait description	Task Swi wait usecs
00000000000C1192	160	0	2006-01-04-16.12.08.914651	T	SWAFD	QMr	KYTK/MIRQ	2679
00000000000C1192	160	0	2006-01-04-16.12.08.914665	T	SWIN			
00000000000C1192	160	0	2006-01-04-16.12.08.915043	T	SWOQ	SFt		
00000000000C1192	160	0	2006-01-04-16.12.08.923283	T	SWAFD	SFt	PAGE FLT	8
00000000000C1192	160	0	2006-01-04-16.12.08.925125	T	SWIN			
00000000000C1192	160	0	2006-01-04-16.12.08.925412	T	SWOQ	SFt		
00000000000C1192	160	0	2006-01-04-16.12.08.931208	T	SWAFD	SFt	PAGE FLT	5
00000000000C1192	160	0	2006-01-04-16.12.08.935447	T	SWIN			
00000000000C1192	160	0	2006-01-04-16.12.08.935477	T	SWOQ	SFt		
00000000000C1192	160	0	2006-01-04-16.12.08.938570	T	SWAFD	SFt	PAGE FLT	3
00000000000C1192	160	0	2006-01-04-16.12.08.940554	T	SWIN			
00000000000C1192	150	0	2006-01-04-16.12.08.940614	T	SWOQ	SFt		
00000000000C1192	150	0	2006-01-04-16.12.08.944343	T	SWAFD	SFt	PAGE FLT	3
00000000000C1192	150	0	2006-01-04-16.12.08.944351	T	SWIN			
00000000000C1192	150	0	2006-01-04-16.12.08.944540	T	SWOQ	SFt		

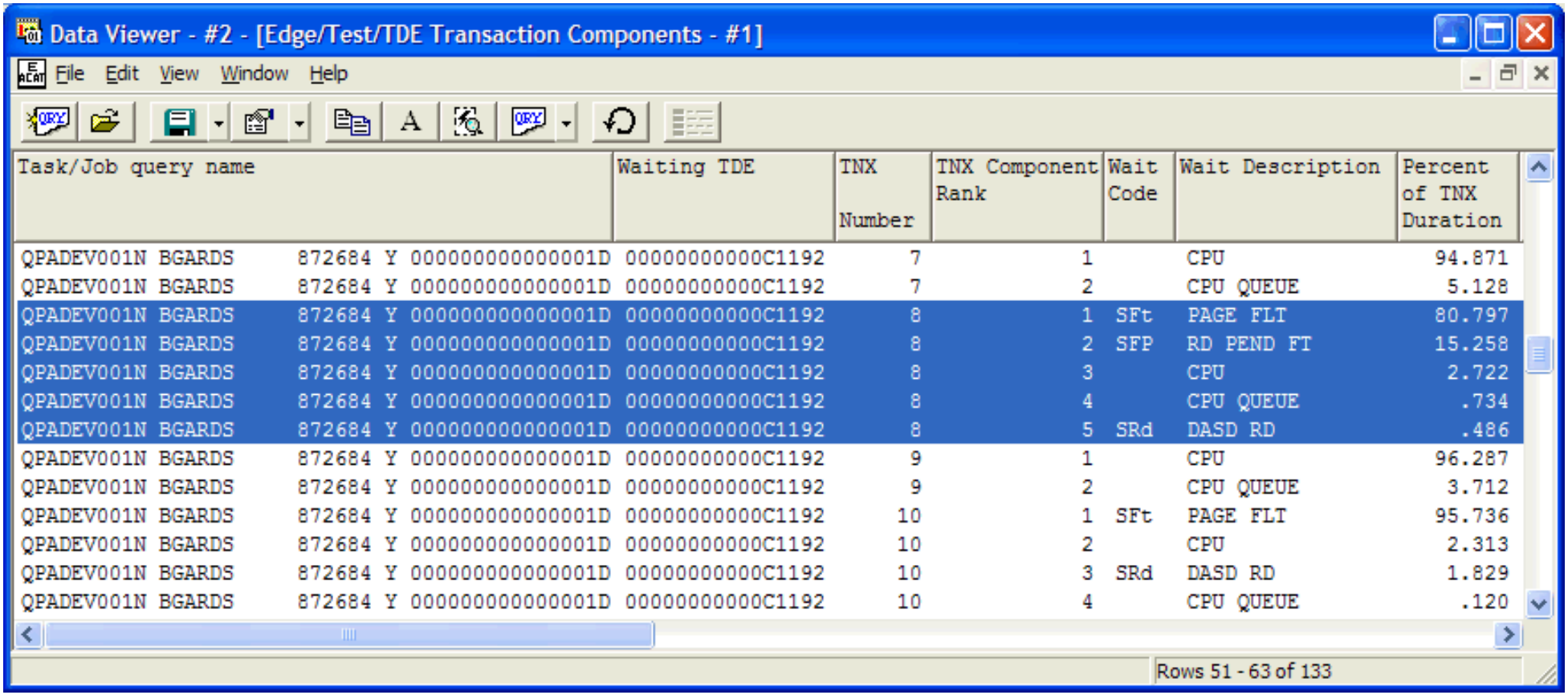
Rows 1 - 15 of 1535

Table of Contents | Previous | Next

5.8.5 TDE Transaction Components

Description: This shows the run/wait components of a transaction in a summarized form per transaction. The example is showing the components of transaction 8 (see example in 5.8.4 TDE Transaction Summary). Further details can viewed by right clicking for popup menu options on a selected transaction component.

Example:



Task/Job query name				Waiting TDE	TNX Number	TNX Component Rank	Wait Code	Wait Description	Percent of TNX Duration
QPADEV001N BGARDS	872684	Y	000000000000001D	00000000000C1192	7	1		CPU	94.871
QPADEV001N BGARDS	872684	Y	000000000000001D	00000000000C1192	7	2		CPU QUEUE	5.128
QPADEV001N BGARDS	872684	Y	000000000000001D	00000000000C1192	8	1	SFt	PAGE FLT	80.797
QPADEV001N BGARDS	872684	Y	000000000000001D	00000000000C1192	8	2	SFP	RD PEND FT	15.258
QPADEV001N BGARDS	872684	Y	000000000000001D	00000000000C1192	8	3		CPU	2.722
QPADEV001N BGARDS	872684	Y	000000000000001D	00000000000C1192	8	4		CPU QUEUE	.734
QPADEV001N BGARDS	872684	Y	000000000000001D	00000000000C1192	8	5	SRd	DASD RD	.486
QPADEV001N BGARDS	872684	Y	000000000000001D	00000000000C1192	9	1		CPU	96.287
QPADEV001N BGARDS	872684	Y	000000000000001D	00000000000C1192	9	2		CPU QUEUE	3.712
QPADEV001N BGARDS	872684	Y	000000000000001D	00000000000C1192	10	1	SFt	PAGE FLT	95.736
QPADEV001N BGARDS	872684	Y	000000000000001D	00000000000C1192	10	2		CPU	2.313
QPADEV001N BGARDS	872684	Y	000000000000001D	00000000000C1192	10	3	SRd	DASD RD	1.829
QPADEV001N BGARDS	872684	Y	000000000000001D	00000000000C1192	10	4		CPU QUEUE	.120

Rows 51 - 63 of 133

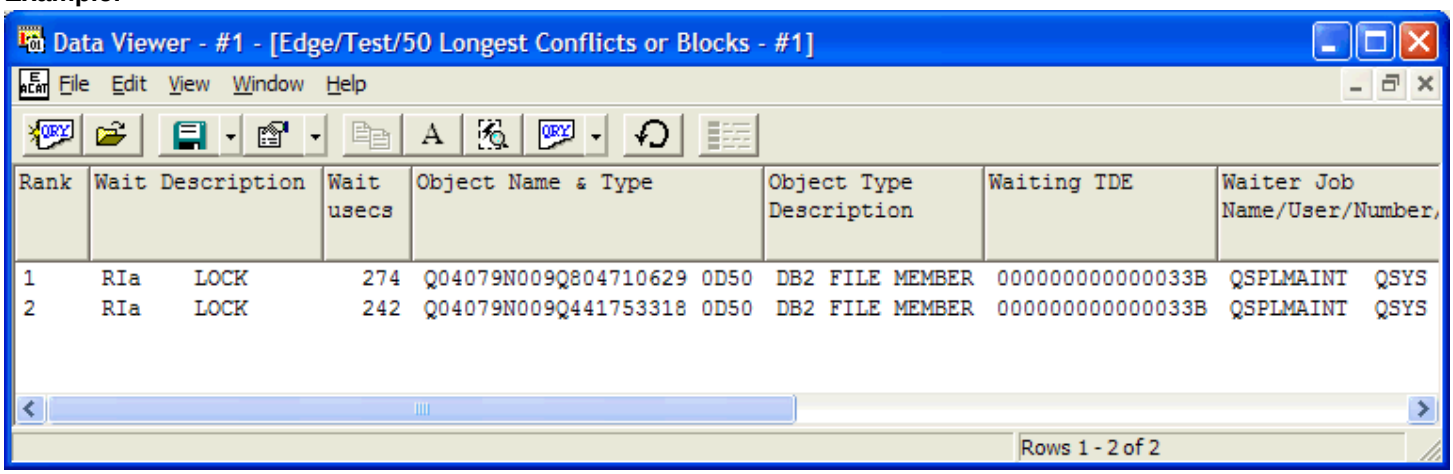
Table of Contents | Previous | Next

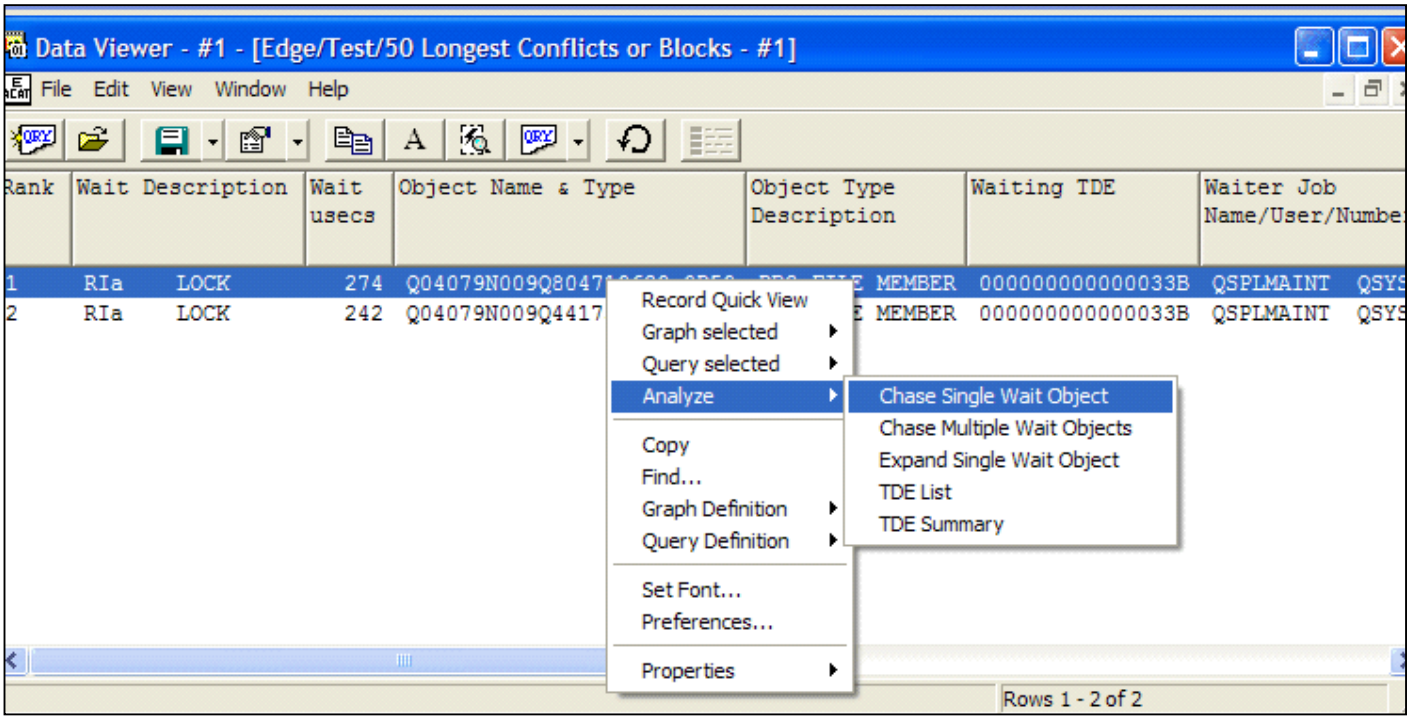
5.8.6 50 Longest Conflicts or Blocks

Description: The 50 longest seize or lock conflicts found in the entire collection are listed. This could be a starting point to chase down the wakers (leading to the holders or real culprits) of the seize or lock waiters (victims) for the worst/longest conflicts.

The second display shows selecting an event, right clicking for popup menu options and performing Analyze and Chase Single Wait Object options to zero in on the waker/holder TDE.

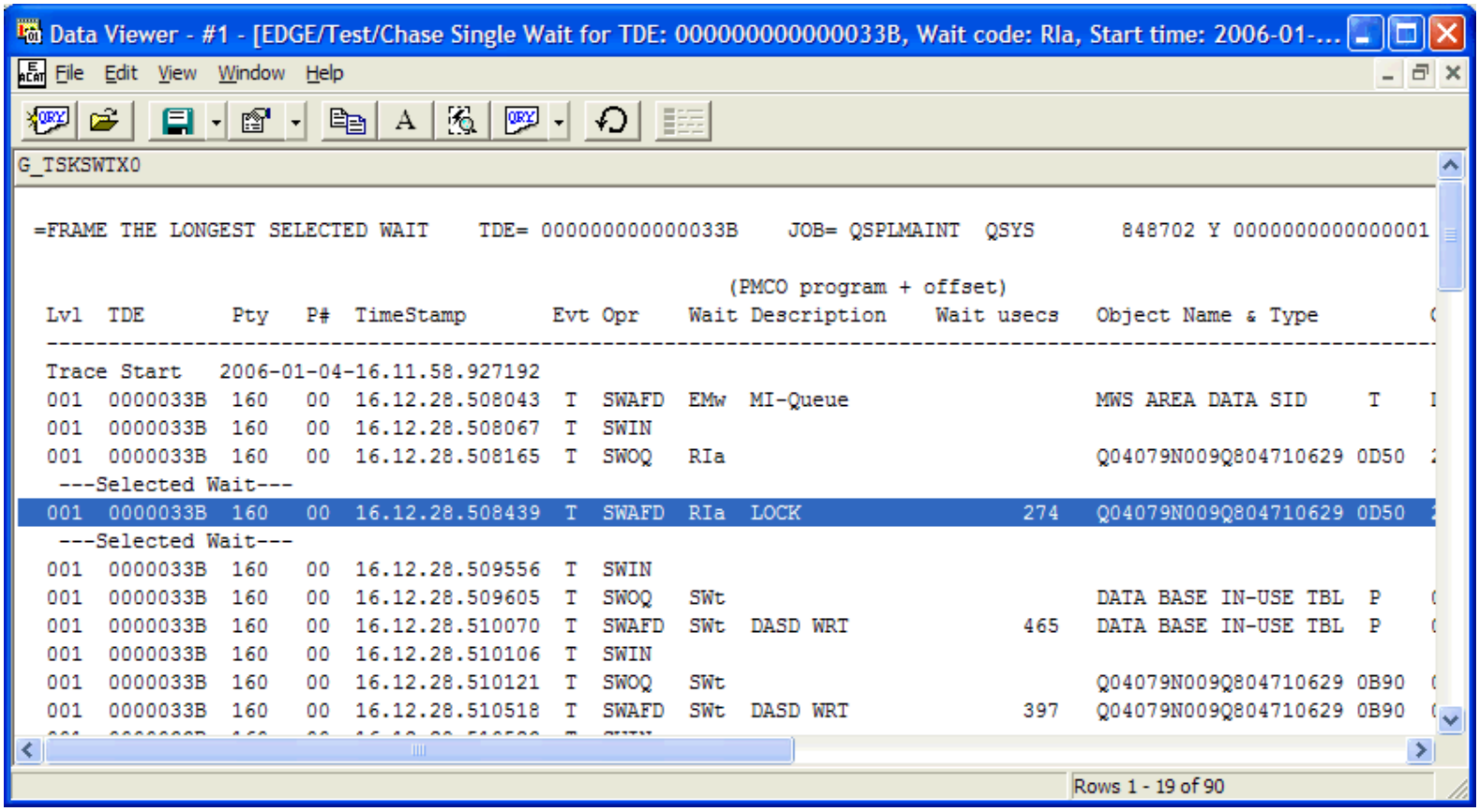
Example:





The results of the chase are shown below. The GUI report is similar to the printed report that is created using the iDoctor PEX green screen command GTSKSWTX. Because this report is the result of an Analysis option, it is also saved as a file member and can be viewed through the Child analysis report menu.

The highlighted field below shows the event that was used to perform the chase. This was done to frame the selected event (put it into perspective relative to events preceding and following).



By scrolling further down into the report you will come to a section that unravels the reason for the wait. The chase is in descending time sequence, working back in time from the release of the object waited on by the waker level by level. In the example below, which is only 2 levels deep, we find the TDE that released the object at level 2. In this case the level 2 TDE was the waker and the holder when the level one TDE wanted to obtain a lock on the object.

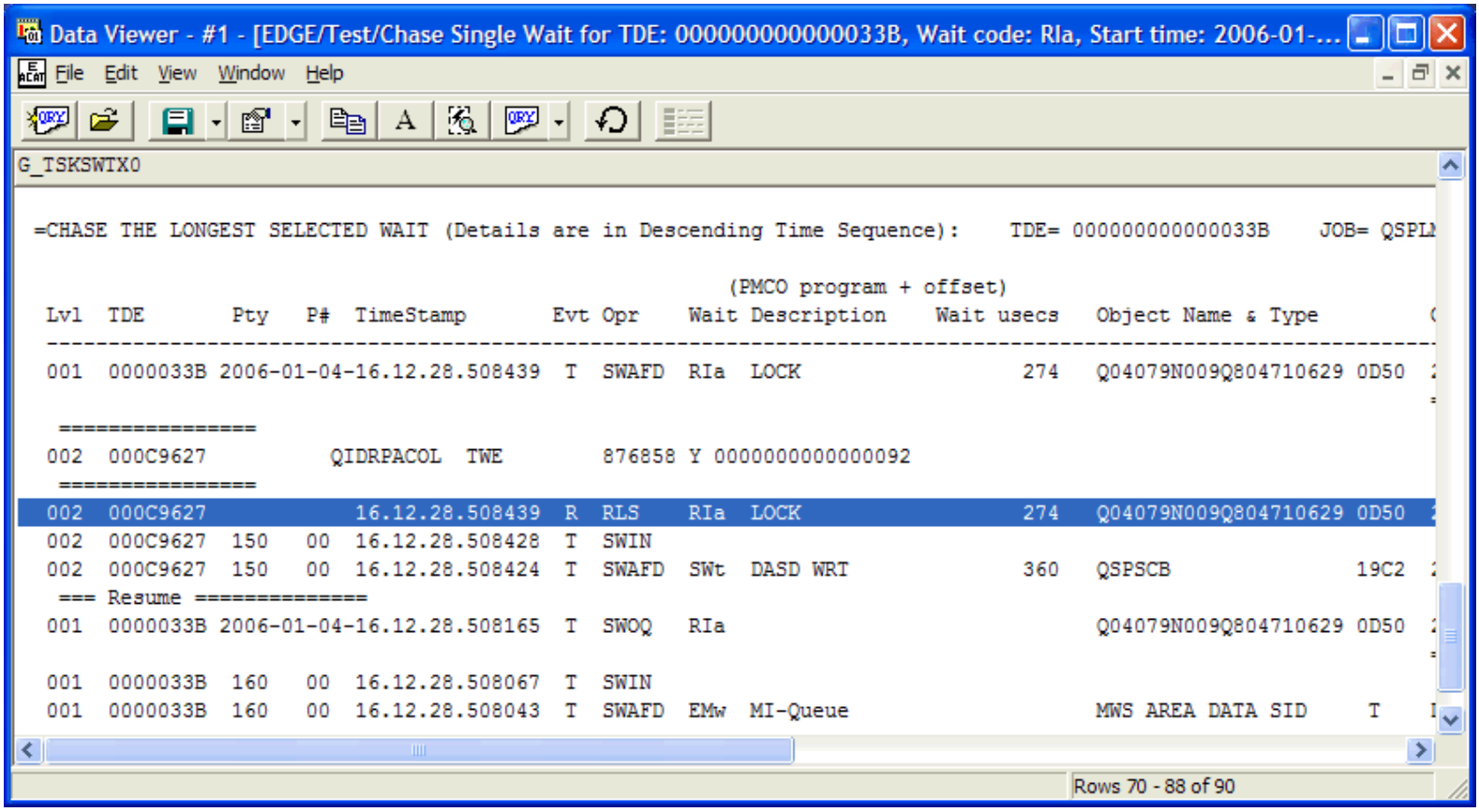
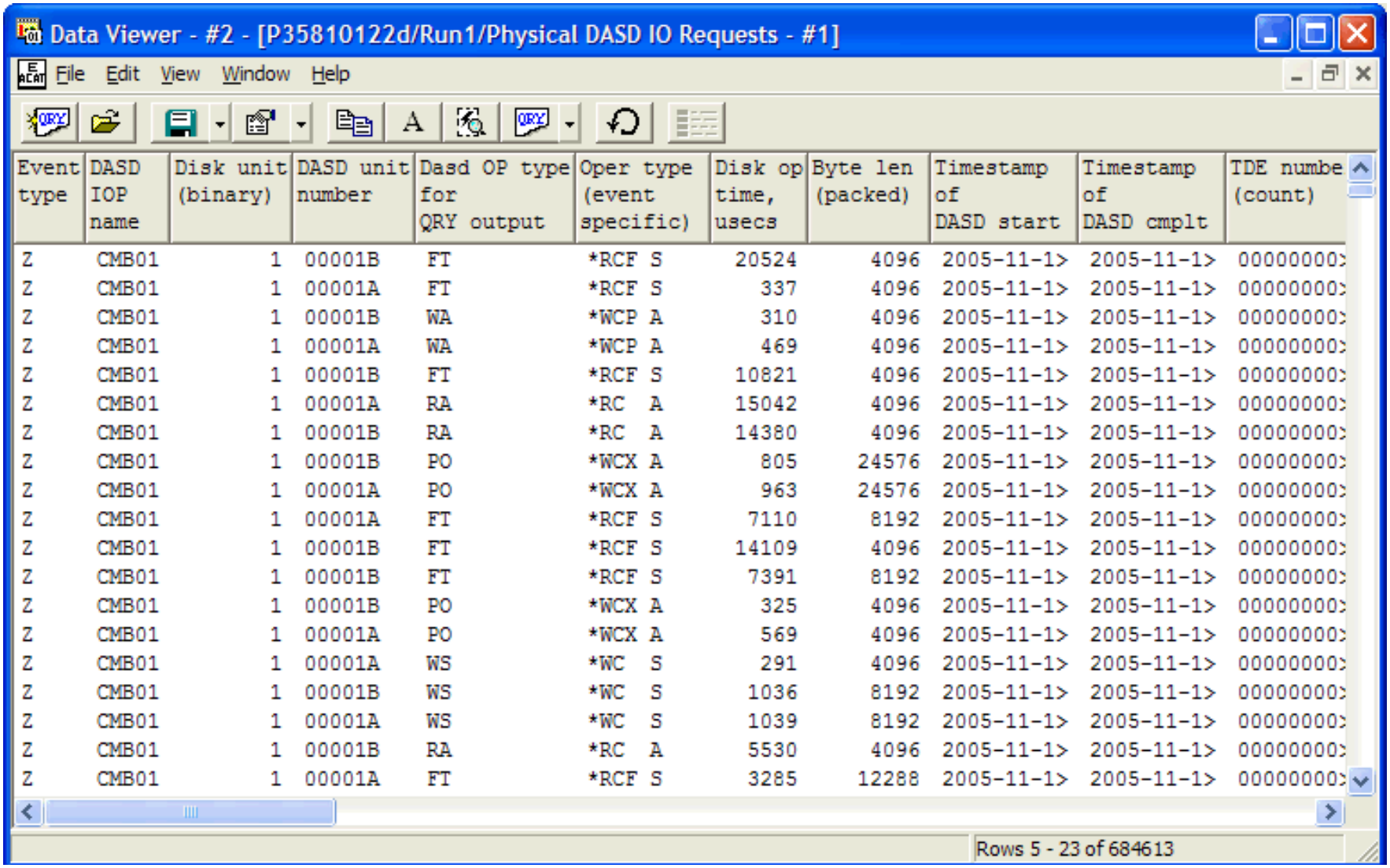


Table of Contents | Previous | Next

5.8.7 Physical DASD IO Requests

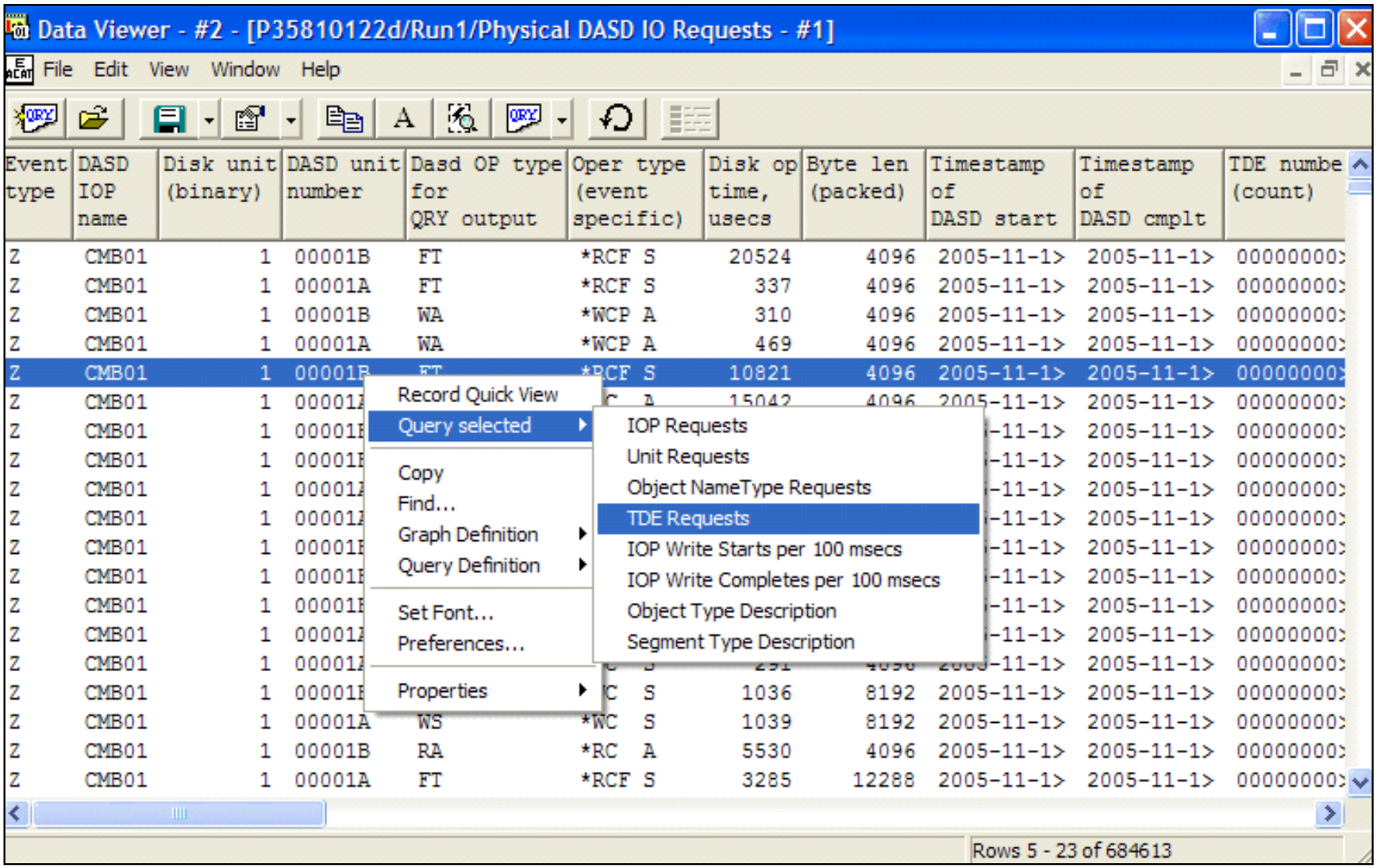
Description: This list is available when DASD op starts and stops are collected as a PEX trace event option. The object name and type, the DASD unit and IOP, the size of the op and the timestamp are known and can facilitate a very thorough, detailed analysis.

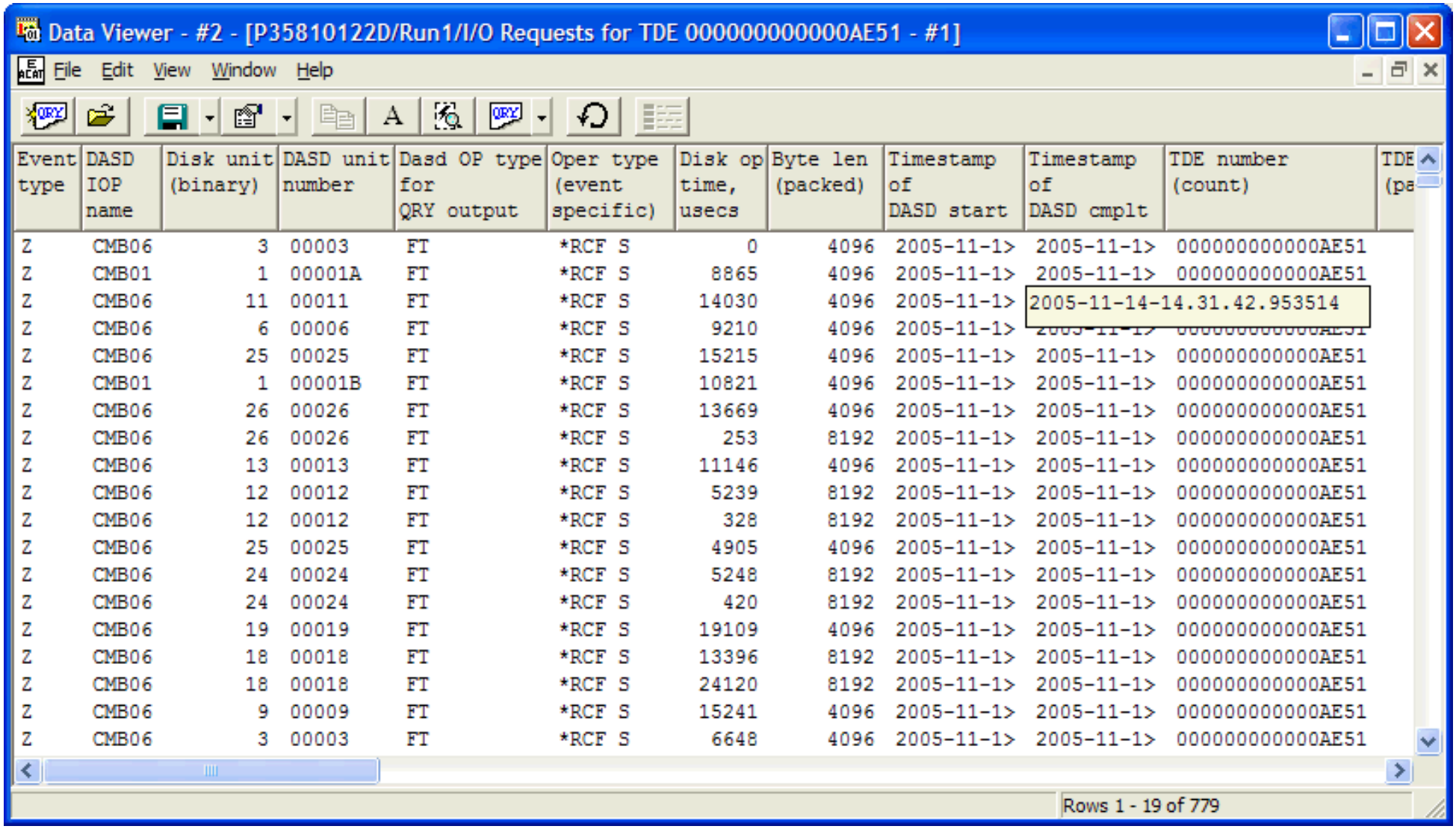
Example:



Event type	DASD IOP name	Disk unit (binary)	DASD unit number	Dasd OP type for QRY output	Oper type (event specific)	Disk op time, usecs	Byte len (packed)	Timestamp of DASD start	Timestamp of DASD cmplt	TDE numbe (count)
Z	CMB01	1	00001B	FT	*RCF S	20524	4096	2005-11-1>	2005-11-1>	00000000>
Z	CMB01	1	00001A	FT	*RCF S	337	4096	2005-11-1>	2005-11-1>	00000000>
Z	CMB01	1	00001B	WA	*WCP A	310	4096	2005-11-1>	2005-11-1>	00000000>
Z	CMB01	1	00001A	WA	*WCP A	469	4096	2005-11-1>	2005-11-1>	00000000>
Z	CMB01	1	00001B	FT	*RCF S	10821	4096	2005-11-1>	2005-11-1>	00000000>
Z	CMB01	1	00001A	RA	*RC A	15042	4096	2005-11-1>	2005-11-1>	00000000>
Z	CMB01	1	00001B	RA	*RC A	14380	4096	2005-11-1>	2005-11-1>	00000000>
Z	CMB01	1	00001B	PO	*WCX A	805	24576	2005-11-1>	2005-11-1>	00000000>
Z	CMB01	1	00001A	PO	*WCX A	963	24576	2005-11-1>	2005-11-1>	00000000>
Z	CMB01	1	00001A	FT	*RCF S	7110	8192	2005-11-1>	2005-11-1>	00000000>
Z	CMB01	1	00001B	FT	*RCF S	14109	4096	2005-11-1>	2005-11-1>	00000000>
Z	CMB01	1	00001B	FT	*RCF S	7391	8192	2005-11-1>	2005-11-1>	00000000>
Z	CMB01	1	00001B	PO	*WCX A	325	4096	2005-11-1>	2005-11-1>	00000000>
Z	CMB01	1	00001A	PO	*WCX A	569	4096	2005-11-1>	2005-11-1>	00000000>
Z	CMB01	1	00001A	WS	*WC S	291	4096	2005-11-1>	2005-11-1>	00000000>
Z	CMB01	1	00001B	WS	*WC S	1036	8192	2005-11-1>	2005-11-1>	00000000>
Z	CMB01	1	00001A	WS	*WC S	1039	8192	2005-11-1>	2005-11-1>	00000000>
Z	CMB01	1	00001B	RA	*RC A	5530	4096	2005-11-1>	2005-11-1>	00000000>
Z	CMB01	1	00001A	FT	*RCF S	3285	12288	2005-11-1>	2005-11-1>	00000000>

The following example shows selecting an event, right clicking to select a popup menu options Query selected, TDE Requests. The results in the next display show all the DASD ops requested for the TDE in the selected event.





Event type	DASD IOP name	Disk unit (binary)	DASD unit number	Dasd OP type for QRY output	Oper type (event specific)	Disk op time, usecs	Byte len (packed)	Timestamp of DASD start	Timestamp of DASD cmplt	TDE number (count)	TDE (pa
Z	CMB06	3	00003	FT	*RCF S	0	4096	2005-11-1>	2005-11-1>	000000000000AE51	
Z	CMB01	1	00001A	FT	*RCF S	8865	4096	2005-11-1>	2005-11-1>	000000000000AE51	
Z	CMB06	11	00011	FT	*RCF S	14030	4096	2005-11-1>	2005-11-14-14.31.42.953514		
Z	CMB06	6	00006	FT	*RCF S	9210	4096	2005-11-1>			
Z	CMB06	25	00025	FT	*RCF S	15215	4096	2005-11-1>	2005-11-1>	000000000000AE51	
Z	CMB01	1	00001B	FT	*RCF S	10821	4096	2005-11-1>	2005-11-1>	000000000000AE51	
Z	CMB06	26	00026	FT	*RCF S	13669	4096	2005-11-1>	2005-11-1>	000000000000AE51	
Z	CMB06	26	00026	FT	*RCF S	253	8192	2005-11-1>	2005-11-1>	000000000000AE51	
Z	CMB06	13	00013	FT	*RCF S	11146	4096	2005-11-1>	2005-11-1>	000000000000AE51	
Z	CMB06	12	00012	FT	*RCF S	5239	8192	2005-11-1>	2005-11-1>	000000000000AE51	
Z	CMB06	12	00012	FT	*RCF S	328	8192	2005-11-1>	2005-11-1>	000000000000AE51	
Z	CMB06	25	00025	FT	*RCF S	4905	4096	2005-11-1>	2005-11-1>	000000000000AE51	
Z	CMB06	24	00024	FT	*RCF S	5248	8192	2005-11-1>	2005-11-1>	000000000000AE51	
Z	CMB06	24	00024	FT	*RCF S	420	8192	2005-11-1>	2005-11-1>	000000000000AE51	
Z	CMB06	19	00019	FT	*RCF S	19109	4096	2005-11-1>	2005-11-1>	000000000000AE51	
Z	CMB06	18	00018	FT	*RCF S	13396	8192	2005-11-1>	2005-11-1>	000000000000AE51	
Z	CMB06	18	00018	FT	*RCF S	24120	8192	2005-11-1>	2005-11-1>	000000000000AE51	
Z	CMB06	9	00009	FT	*RCF S	15241	4096	2005-11-1>	2005-11-1>	000000000000AE51	
Z	CMB06	3	00003	FT	*RCF S	6648	4096	2005-11-1>	2005-11-1>	000000000000AE51	

Table of Contents Previous Next

5.8.8 TDE One Second Intervals

Description: The TDE One Second Intervals view shows a one second summarization of CPU used and certain categories of waits. The one second intervals are sequential starting from the beginning of the collection.

The view is used to quickly determine an active time range or a time range of interest, one that would yield events for certain types of waits. The interval or intervals can be selected and used to frame the time range for drilling into TDE details. The data is also used to display a Job Watcher like run/wait signature graph.

Example:

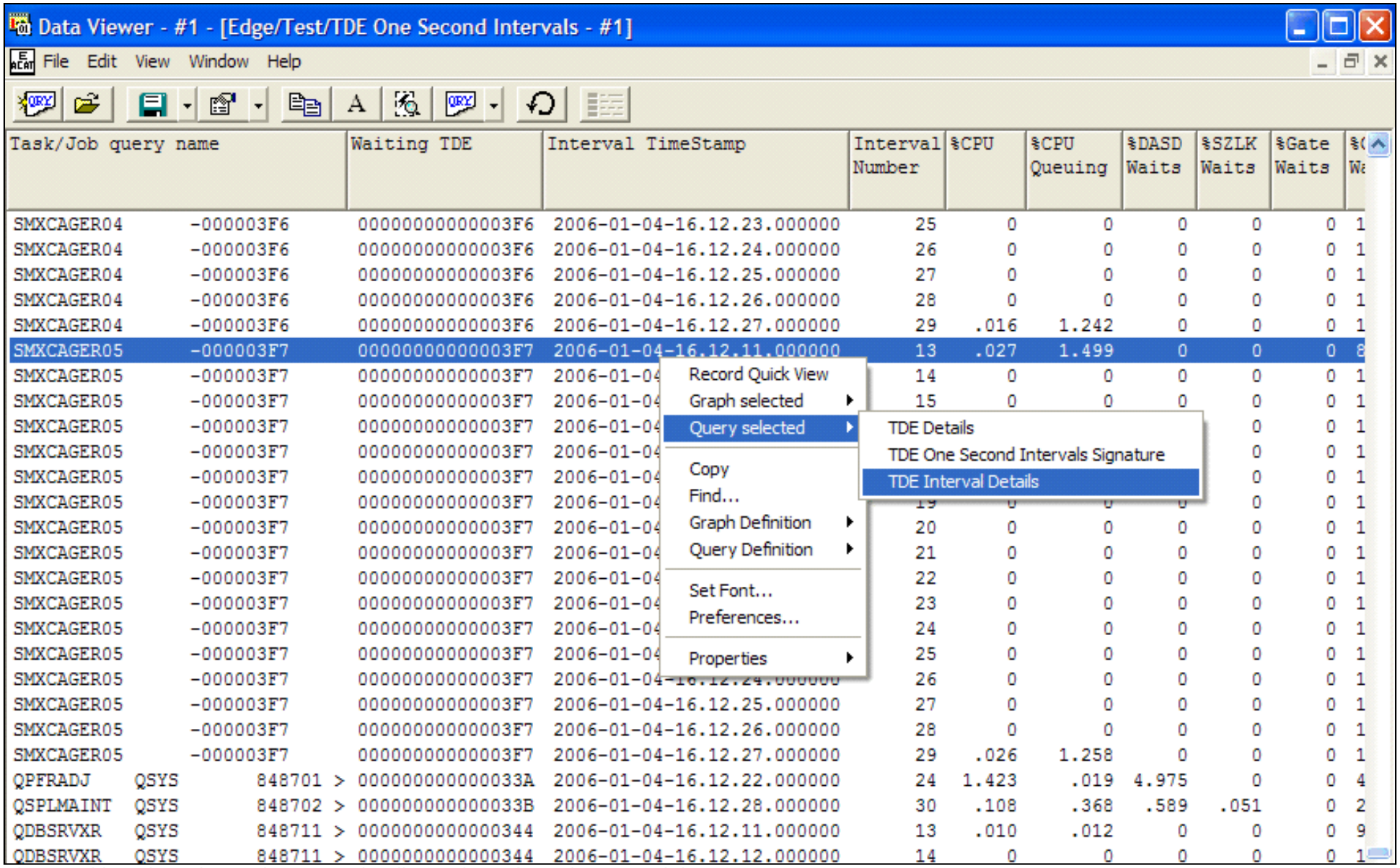
Data Viewer - #1 - [Edge/Test/TDE One Second Intervals - #1]

File Edit View Window Help

Task/Job query name	Waiting TDE	Interval TimeStamp	Interval Number	%CPU	%CPU Queuing	%DASD Waits	%SZLK Waits	%Gate Waits	%(W
SMXCAGER04 -000003F6	00000000000003F6	2006-01-04-16.12.23.000000	25	0	0	0	0	0	1
SMXCAGER04 -000003F6	00000000000003F6	2006-01-04-16.12.24.000000	26	0	0	0	0	0	1
SMXCAGER04 -000003F6	00000000000003F6	2006-01-04-16.12.25.000000	27	0	0	0	0	0	1
SMXCAGER04 -000003F6	00000000000003F6	2006-01-04-16.12.26.000000	28	0	0	0	0	0	1
SMXCAGER04 -000003F6	00000000000003F6	2006-01-04-16.12.27.000000	29	.016	1.242	0	0	0	1
SMXCAGER05 -000003F7	00000000000003F7	2006-01-04-16.12.11.000000	13	.027	1.499	0	0	0	8
SMXCAGER05 -000003F7	00000000000003F7	2006-01-04-16.12.12.000000	14	0	0	0	0	0	1
SMXCAGER05 -000003F7	00000000000003F7	2006-01-04-16.12.13.000000	15	0	0	0	0	0	1
SMXCAGER05 -000003F7	00000000000003F7	2006-01-04-16.12.14.000000	16	0	0	0	0	0	1
SMXCAGER05 -000003F7	00000000000003F7	2006-01-04-16.12.15.000000	17	0	0	0	0	0	1
SMXCAGER05 -000003F7	00000000000003F7	2006-01-04-16.12.16.000000	18	0	0	0	0	0	1
SMXCAGER05 -000003F7	00000000000003F7	2006-01-04-16.12.17.000000	19	0	0	0	0	0	1
SMXCAGER05 -000003F7	00000000000003F7	2006-01-04-16.12.18.000000	20	0	0	0	0	0	1
SMXCAGER05 -000003F7	00000000000003F7	2006-01-04-16.12.19.000000	21	0	0	0	0	0	1
SMXCAGER05 -000003F7	00000000000003F7	2006-01-04-16.12.20.000000	22	0	0	0	0	0	1
SMXCAGER05 -000003F7	00000000000003F7	2006-01-04-16.12.21.000000	23	0	0	0	0	0	1
SMXCAGER05 -000003F7	00000000000003F7	2006-01-04-16.12.22.000000	24	0	0	0	0	0	1
SMXCAGER05 -000003F7	00000000000003F7	2006-01-04-16.12.23.000000	25	0	0	0	0	0	1
SMXCAGER05 -000003F7	00000000000003F7	2006-01-04-16.12.24.000000	26	0	0	0	0	0	1
SMXCAGER05 -000003F7	00000000000003F7	2006-01-04-16.12.25.000000	27	0	0	0	0	0	1
SMXCAGER05 -000003F7	00000000000003F7	2006-01-04-16.12.26.000000	28	0	0	0	0	0	1
SMXCAGER05 -000003F7	00000000000003F7	2006-01-04-16.12.27.000000	29	.026	1.258	0	0	0	1
QPFRADJ QSYS 848701 >	000000000000033A	2006-01-04-16.12.22.000000	24	1.423	.019	4.975	0	0	4
QSPLMAINT QSYS 848702 >	000000000000033B	2006-01-04-16.12.28.000000	30	.108	.368	.589	.051	0	2
QDBSRVXR QSYS 848711 >	0000000000000344	2006-01-04-16.12.11.000000	13	.010	.012	0	0	0	9
QDBSRVXR QSYS 848711 >	0000000000000344	2006-01-04-16.12.12.000000	14	0	0	0	0	0	1
QDBSRVXR QSYS 848711 >	0000000000000344	2006-01-04-16.12.13.000000	15	0	0	0	0	0	1
QDBSRVXR QSYS 848711 >	0000000000000344	2006-01-04-16.12.14.000000	16	0	0	0	0	0	1

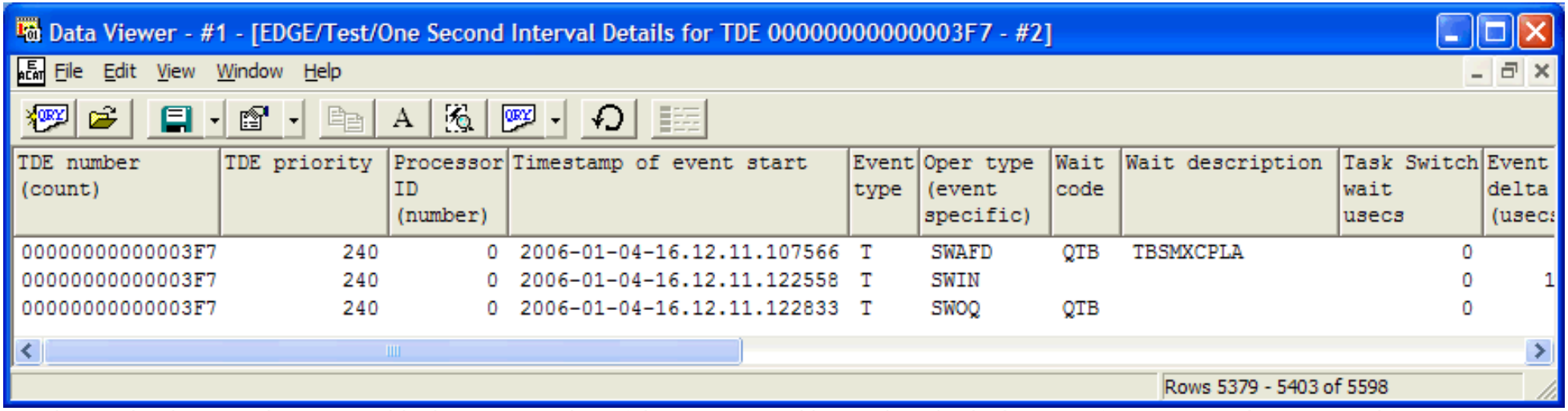
Rows 5379 - 5405 of 5598

The example below shows selecting a time interval for a TDE where we would like to see all the event details.

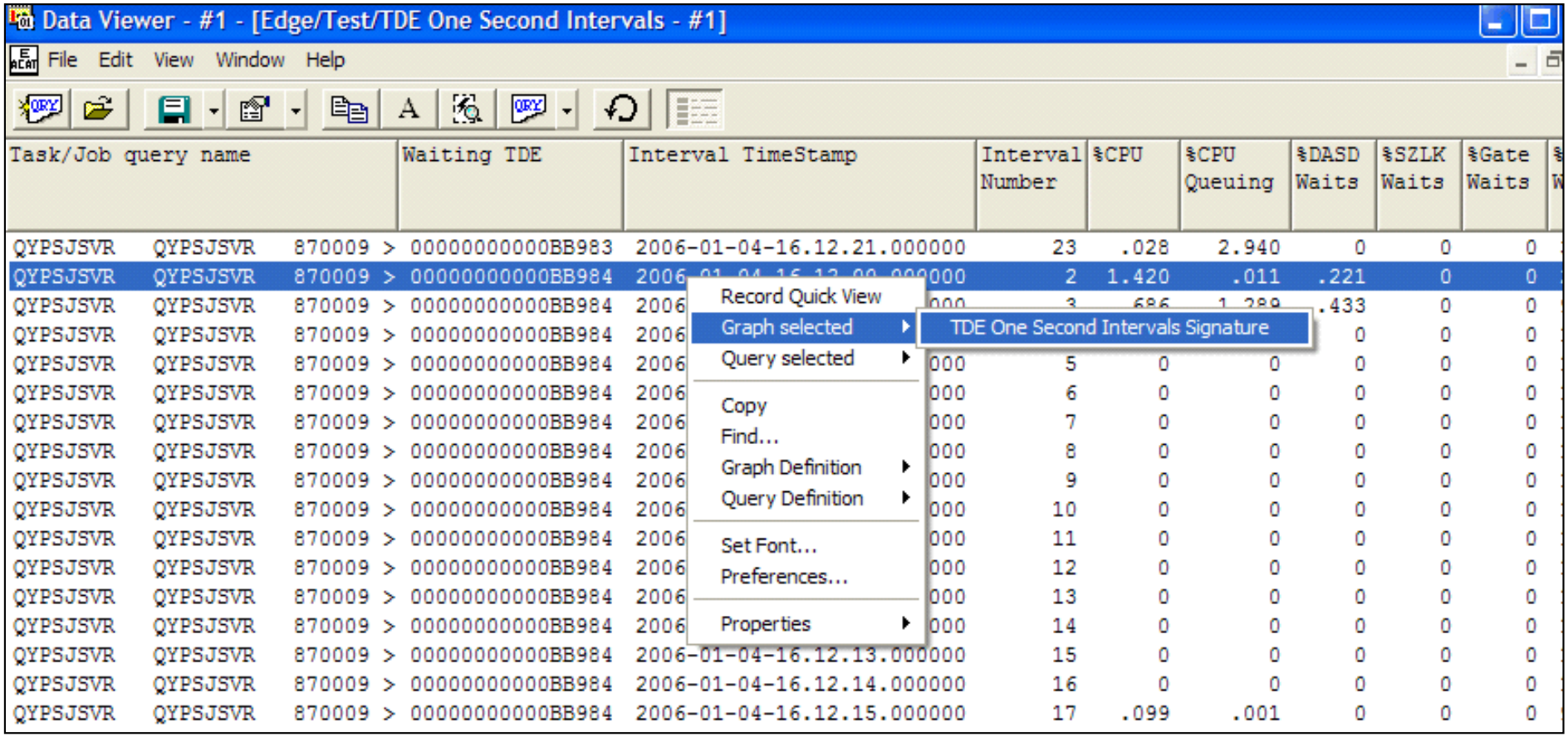


Task/Job query name	Waiting TDE	Interval TimeStamp	Interval Number	%CPU	%CPU Queuing	%DASD Waits	%SZLK Waits	%Gate Waits	%W
SMXCAGER04 -000003F6	00000000000003F6	2006-01-04-16.12.23.000000	25	0	0	0	0	0	1
SMXCAGER04 -000003F6	00000000000003F6	2006-01-04-16.12.24.000000	26	0	0	0	0	0	1
SMXCAGER04 -000003F6	00000000000003F6	2006-01-04-16.12.25.000000	27	0	0	0	0	0	1
SMXCAGER04 -000003F6	00000000000003F6	2006-01-04-16.12.26.000000	28	0	0	0	0	0	1
SMXCAGER04 -000003F6	00000000000003F6	2006-01-04-16.12.27.000000	29	.016	1.242	0	0	0	1
SMXCAGER05 -000003F7	00000000000003F7	2006-01-04-16.12.11.000000	13	.027	1.499	0	0	0	8
SMXCAGER05 -000003F7	00000000000003F7	2006-01-04	14	0	0	0	0	0	1
SMXCAGER05 -000003F7	00000000000003F7	2006-01-04	15	0	0	0	0	0	1
SMXCAGER05 -000003F7	00000000000003F7	2006-01-04					0	0	1
SMXCAGER05 -000003F7	00000000000003F7	2006-01-04					0	0	1
SMXCAGER05 -000003F7	00000000000003F7	2006-01-04					0	0	1
SMXCAGER05 -000003F7	00000000000003F7	2006-01-04					0	0	1
SMXCAGER05 -000003F7	00000000000003F7	2006-01-04	20	0	0	0	0	0	1
SMXCAGER05 -000003F7	00000000000003F7	2006-01-04	21	0	0	0	0	0	1
SMXCAGER05 -000003F7	00000000000003F7	2006-01-04	22	0	0	0	0	0	1
SMXCAGER05 -000003F7	00000000000003F7	2006-01-04	23	0	0	0	0	0	1
SMXCAGER05 -000003F7	00000000000003F7	2006-01-04	24	0	0	0	0	0	1
SMXCAGER05 -000003F7	00000000000003F7	2006-01-04	25	0	0	0	0	0	1
SMXCAGER05 -000003F7	00000000000003F7	2006-01-04	26	0	0	0	0	0	1
SMXCAGER05 -000003F7	00000000000003F7	2006-01-04-16.12.25.000000	27	0	0	0	0	0	1
SMXCAGER05 -000003F7	00000000000003F7	2006-01-04-16.12.26.000000	28	0	0	0	0	0	1
SMXCAGER05 -000003F7	00000000000003F7	2006-01-04-16.12.27.000000	29	.026	1.258	0	0	0	1
QPFRADJ QSYS 848701 >	000000000000033A	2006-01-04-16.12.22.000000	24	1.423	.019	4.975	0	0	4
QSPLMAINT QSYS 848702 >	000000000000033B	2006-01-04-16.12.28.000000	30	.108	.368	.589	.051	0	2
QDBSRVXR QSYS 848711 >	0000000000000344	2006-01-04-16.12.11.000000	13	.010	.012	0	0	0	9
QDBSRVXR QSYS 848711 >	0000000000000344	2006-01-04-16.12.12.000000	14	0	0	0	0	0	1

The resulting view shows the event details for the selected TDE and time frame.



Another option is to produce a graphical representation of the one second interval run/wait signature for a TDE. Select a record for a TDE of interest and right click for popup menu options Graph selected, TDE One Second Intervals Signature.



The resulting graph shows when the TDE was using CPU and waiting for various reasons summarized over one second intervals of time.

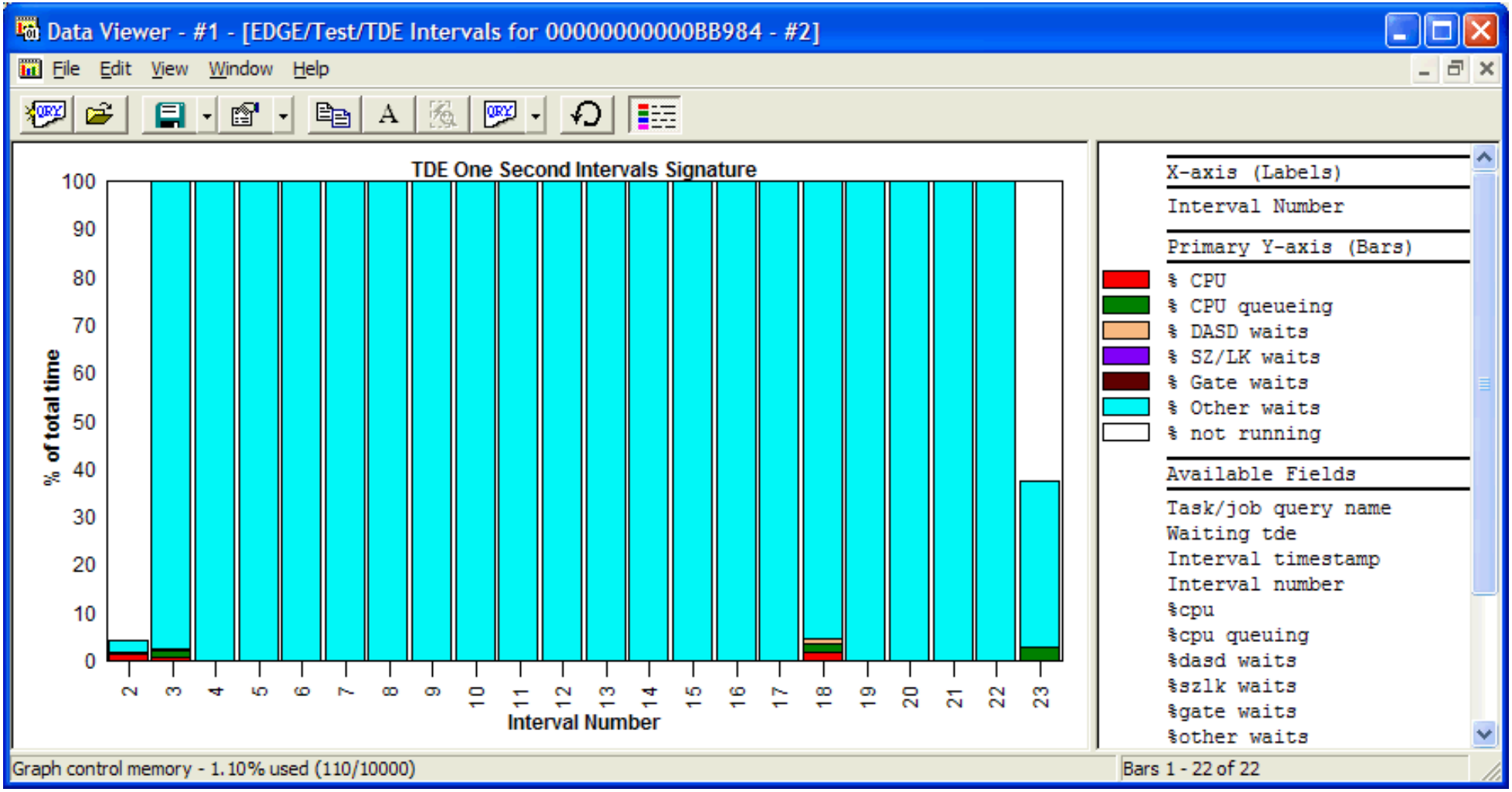


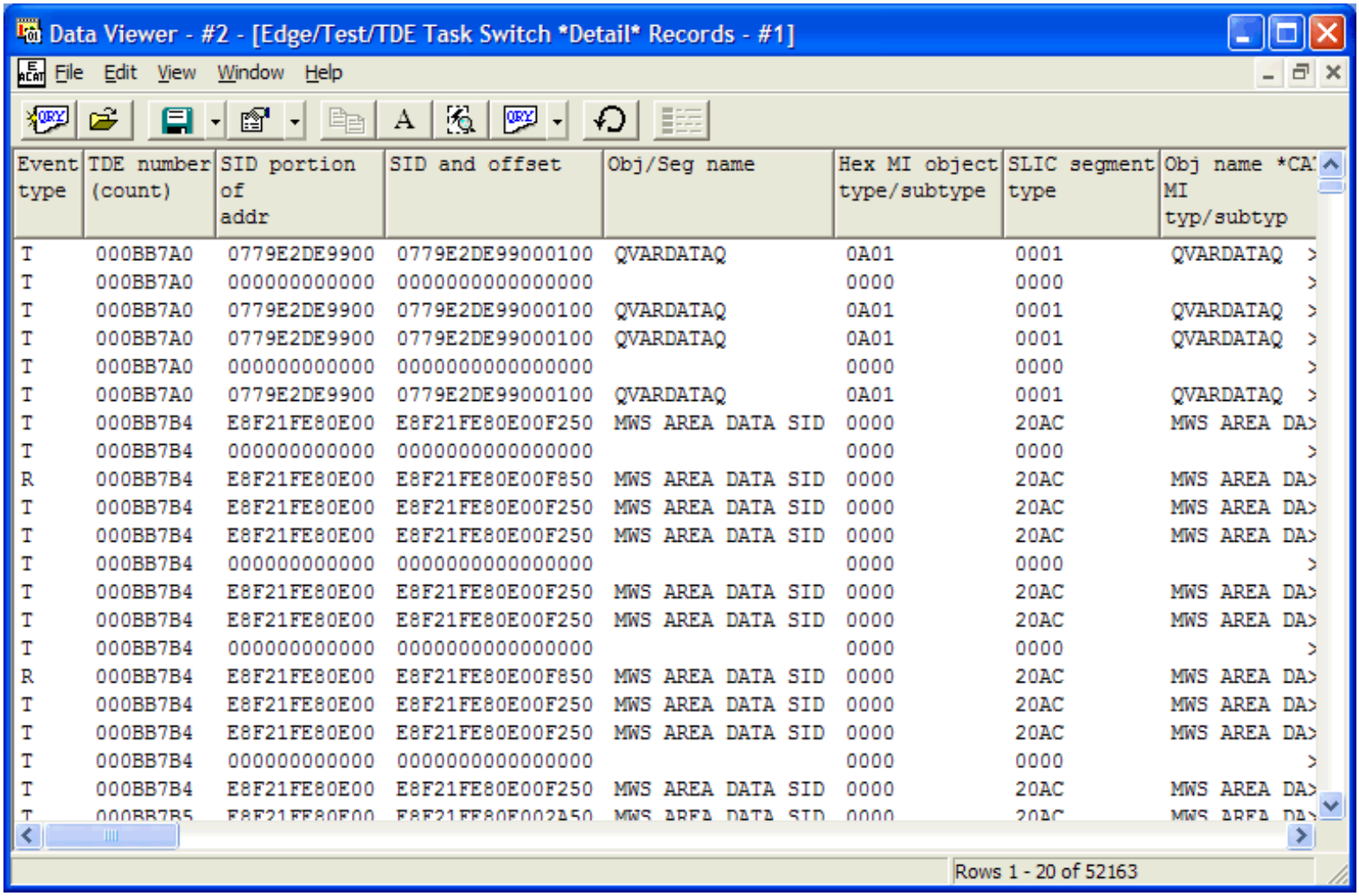
Table of Contents | Previous | Next

5.8.9 TDE Task Switch *Detail* Records

Description: This menu option takes you directly into the trace event details file for the collection.

In order to use this information effectively, you will need to use the Query Definition option on the right click popup menu to setup filters and sorts (see the second display below).

Example:



Event type	TDE number (count)	SID portion of addr	SID and offset	Obj/Seg name	Hex MI object type/subtype	SLIC segment type	Obj name *CA MI typ/subtyp
T	000BB7A0	0779E2DE9900	0779E2DE99000100	QVARDATAQ	0A01	0001	QVARDATAQ >
T	000BB7A0	000000000000	0000000000000000		0000	0000	>
T	000BB7A0	0779E2DE9900	0779E2DE99000100	QVARDATAQ	0A01	0001	QVARDATAQ >
T	000BB7A0	0779E2DE9900	0779E2DE99000100	QVARDATAQ	0A01	0001	QVARDATAQ >
T	000BB7A0	000000000000	0000000000000000		0000	0000	>
T	000BB7A0	0779E2DE9900	0779E2DE99000100	QVARDATAQ	0A01	0001	QVARDATAQ >
T	000BB7B4	E8F21FE80E00	E8F21FE80E00F250	MWS AREA DATA SID	0000	20AC	MWS AREA DA>
T	000BB7B4	000000000000	0000000000000000		0000	0000	>
R	000BB7B4	E8F21FE80E00	E8F21FE80E00F850	MWS AREA DATA SID	0000	20AC	MWS AREA DA>
T	000BB7B4	E8F21FE80E00	E8F21FE80E00F250	MWS AREA DATA SID	0000	20AC	MWS AREA DA>
T	000BB7B4	E8F21FE80E00	E8F21FE80E00F250	MWS AREA DATA SID	0000	20AC	MWS AREA DA>
T	000BB7B4	000000000000	0000000000000000		0000	0000	>
T	000BB7B4	E8F21FE80E00	E8F21FE80E00F250	MWS AREA DATA SID	0000	20AC	MWS AREA DA>
T	000BB7B4	E8F21FE80E00	E8F21FE80E00F250	MWS AREA DATA SID	0000	20AC	MWS AREA DA>
T	000BB7B4	000000000000	0000000000000000		0000	0000	>
R	000BB7B4	E8F21FE80E00	E8F21FE80E00F850	MWS AREA DATA SID	0000	20AC	MWS AREA DA>
T	000BB7B4	E8F21FE80E00	E8F21FE80E00F250	MWS AREA DATA SID	0000	20AC	MWS AREA DA>
T	000BB7B4	E8F21FE80E00	E8F21FE80E00F250	MWS AREA DATA SID	0000	20AC	MWS AREA DA>
T	000BB7B4	000000000000	0000000000000000		0000	0000	>
T	000BB7B4	E8F21FE80E00	E8F21FE80E00F250	MWS AREA DATA SID	0000	20AC	MWS AREA DA>
T	000BB7B5	E8F21FE80E00	E8F21FE80E002A50	MWS AREA DATA SID	0000	20AC	MWS AREA DA>

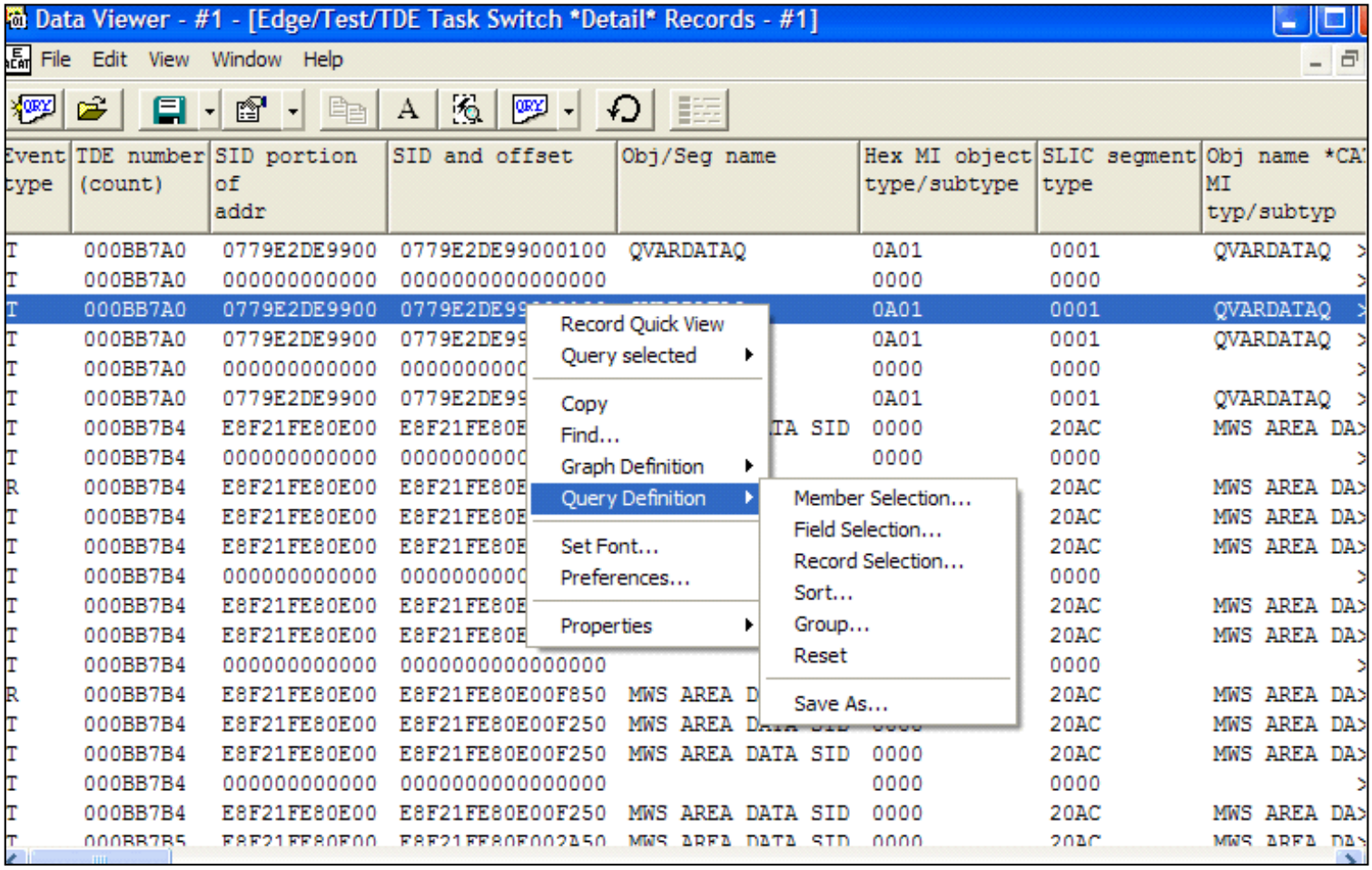

Data Viewer - #1 - [Edge/Test/TDE Task Switch *Detail* Records - #1]
File Edit View Window Help
Event type
TDE number (count)
SID portion of addr
SID and offset
Obj/Seg name
Hex MI object type/subtype
SLIC segment type
Obj name *CA MI typ/subtyp
T 000BB7A0 0779E2DE9900 0779E2DE99000100 QVARDATAQ 0A01 0001 QVARDATAQ
T 000BB7A0 000000000000 0000000000000000 0000 0000
T 000BB7A0 0779E2DE9900 0779E2DE99 0A01 0001 QVARDATAQ
T 000BB7A0 0779E2DE9900 0779E2DE99 0A01 0001 QVARDATAQ
T 000BB7A0 000000000000 0000000000 0000 0000
T 000BB7A0 0779E2DE9900 0779E2DE99 0A01 0001 QVARDATAQ
T 000BB7B4 E8F21FE80E00 E8F21FE80E TA SID 0000 20AC MWS AREA DA
T 000BB7B4 000000000000 0000000000 0000 0000
R 000BB7B4 E8F21FE80E00 E8F21FE80E 20AC MWS AREA DA
T 000BB7B4 E8F21FE80E00 E8F21FE80E 20AC MWS AREA DA
T 000BB7B4 E8F21FE80E00 E8F21FE80E 20AC MWS AREA DA
T 000BB7B4 000000000000 0000000000 0000
T 000BB7B4 E8F21FE80E00 E8F21FE80E 20AC MWS AREA DA
T 000BB7B4 E8F21FE80E00 E8F21FE80E 20AC MWS AREA DA
T 000BB7B4 000000000000 0000000000000000 0000
R 000BB7B4 E8F21FE80E00 E8F21FE80E00F850 MWS AREA D 20AC MWS AREA DA
T 000BB7B4 E8F21FE80E00 E8F21FE80E00F250 MWS AREA 20AC MWS AREA DA
T 000BB7B4 E8F21FE80E00 E8F21FE80E00F250 MWS AREA DATA SID 0000 20AC MWS AREA DA
T 000BB7B4 000000000000 0000000000000000 0000 0000
T 000BB7B4 E8F21FE80E00 E8F21FE80E00F250 MWS AREA DATA SID 0000 20AC MWS AREA DA
Record Quick View
Query selected
Copy
Find...
Graph Definition
Query Definition
Set Font...
Preferences...
Properties
Member Selection...
Field Selection...
Record Selection...
Sort...
Group...
Reset
Save As...

Table of Contents | Previous | Next

5.9 Data area reports

This analysis shows all data area changes and retrieves. It indicates the causing job/thread, data area and library names, data area type, local or DDM, and time of day each operation occurred.

If the program events *MIENTRY and *MIEXIT are collected, then the PEX Analysis shows the calling program name, elapsed time, CPU, and disk I/O during each data area operation.

Table of Contents | Previous | Next

5.9.1 Data area usage by job

Description:
This report displays the jobs that experienced data area usage within the collection.

Example:

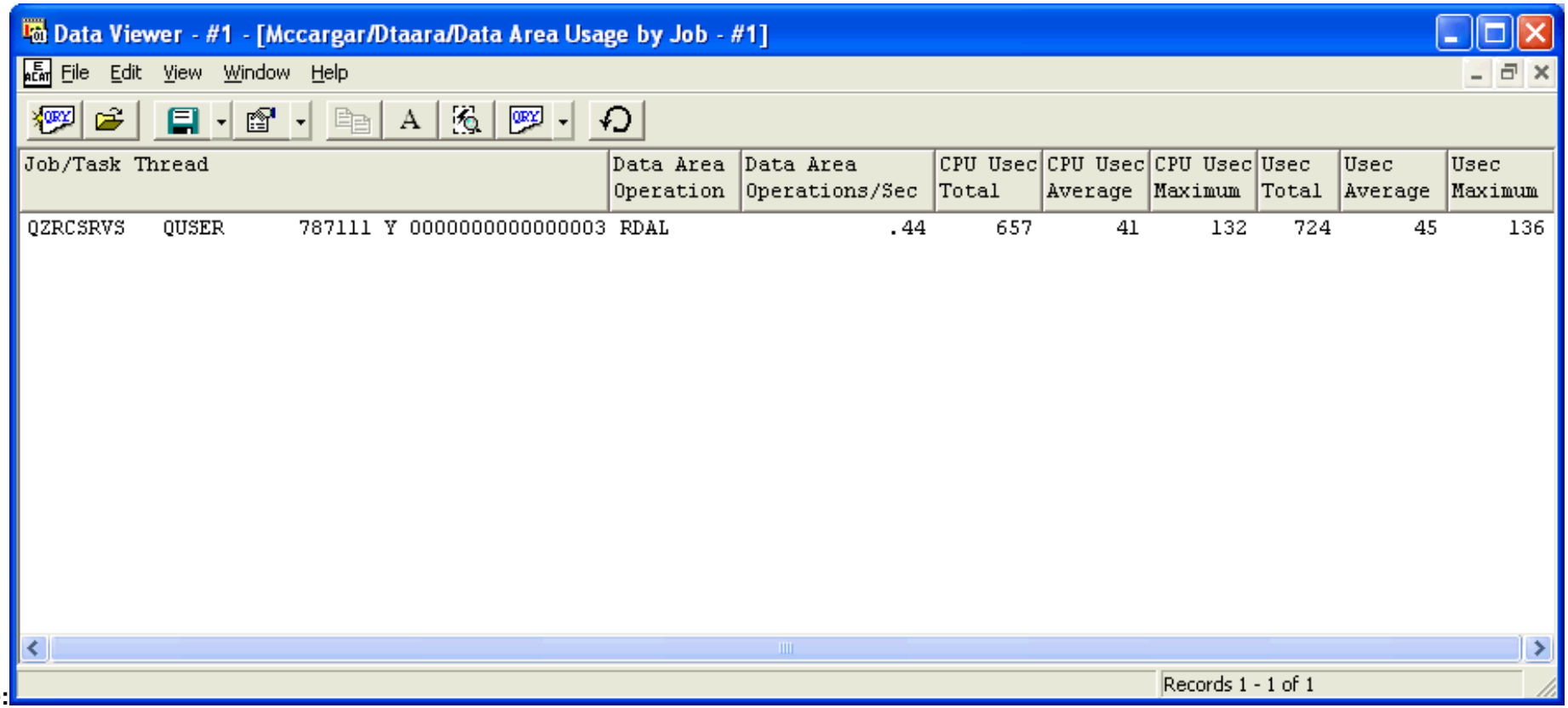
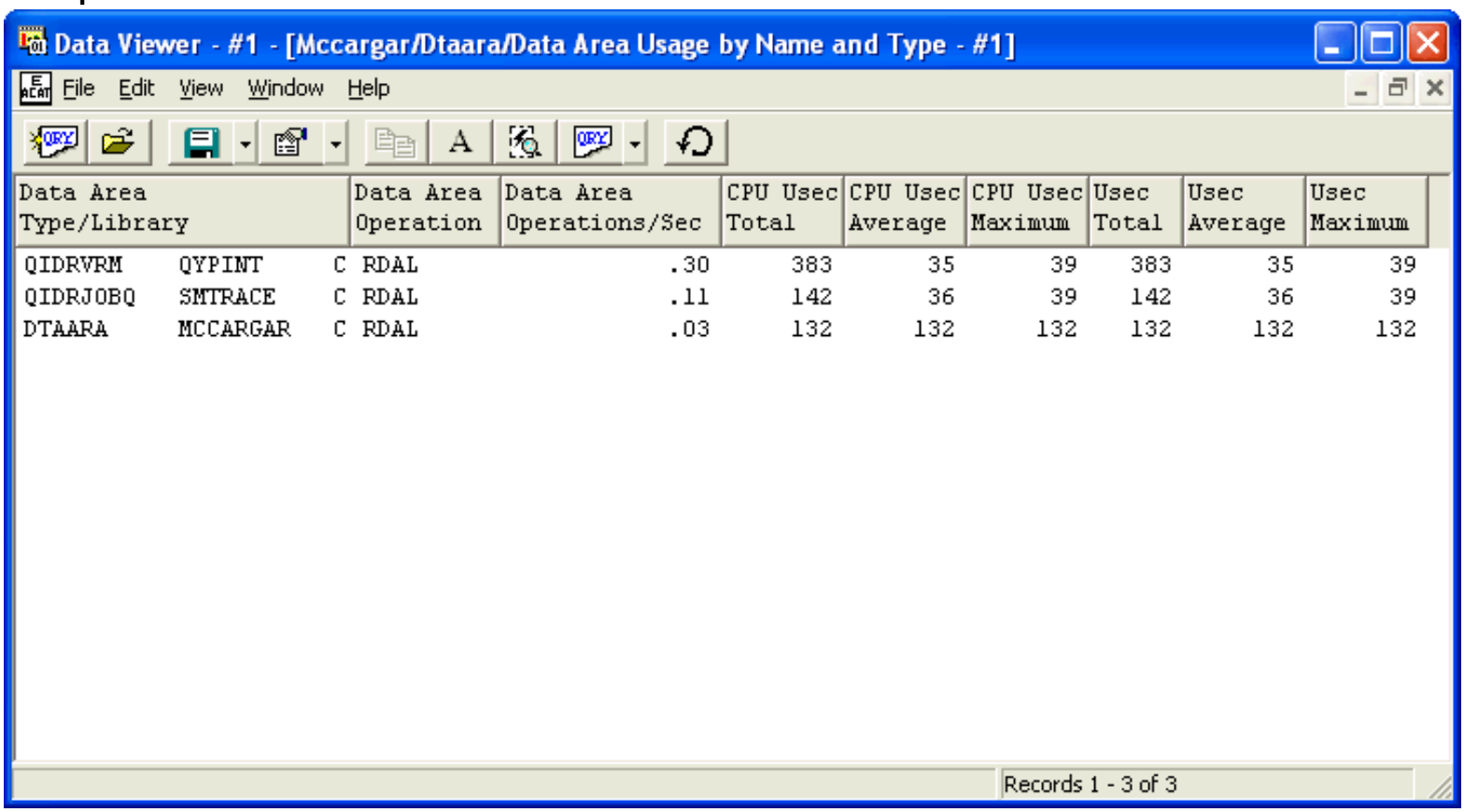


Table of Contents | Previous | Next

5.9.2 Data area usage by name and type

Description:
This report provides the data areas that were operated on as well as statistics for each data area during the collection.

Example:



Data Area Type/Library			Data Area Operation	Data Area Operations/Sec	CPU Usec Total	CPU Usec Average	CPU Usec Maximum	Usec Total	Usec Average	Usec Maximum
QIDRVRM	QYPINT	C	RDAL	.30	383	35	39	383	35	39
QIDRJOBQ	SMTRACE	C	RDAL	.11	142	36	39	142	36	39
DTAARA	MCCARGAR	C	RDAL	.03	132	132	132	132	132	132

Table of Contents | Previous | Next

5.9.3 Data area usage by program

Description:
This report provides the program names that were operated on as well as statistics for each program during the collection.

Example:

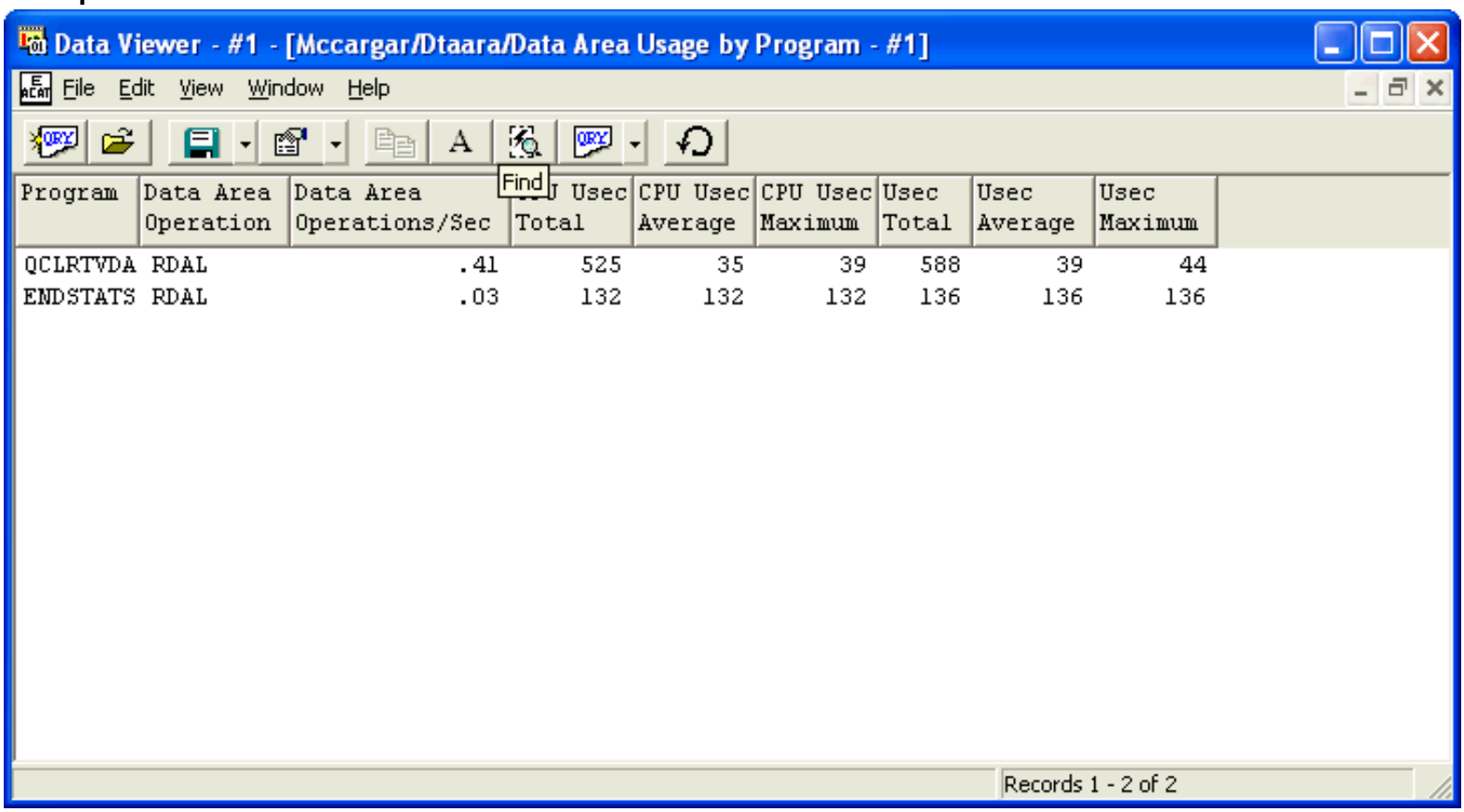


Table of Contents | Previous | Next

5.9.4 Data area usage by time interval

Description:

Example:

Data Viewer - #1 - [Mccargar/Dtaara/Data Area Usage by Time Interval - #1]

File Edit View Window Help

Interval	Interval Max DateTime	Data Area Operation	Data Area Operation Total	CPU Usec Total	CPU Usec Average	CPU Usec Maximum	Usec Total	Usec Average
6	2005-02-09-09.03.55.337707	RDAL	1	35	35	35	40	40
9	2005-02-09-09.03.58.500178	RDAL	1	30	30	30	35	35
10	2005-02-09-09.03.59.943917	RDAL	1	32	32	32	35	35
11	2005-02-09-09.04.01.399640	RDAL	1	37	37	37	41	41
13	2005-02-09-09.04.04.128098	RDAL	2	74	37	39	83	42
15	2005-02-09-09.04.06.736209	RDAL	1	34	34	34	38	38
17	2005-02-09-09.04.08.555118	RDAL	1	33	33	33	38	38
18	2005-02-09-09.04.10.284066	RDAL	1	37	37	37	40	40
20	2005-02-09-09.04.12.467319	RDAL	1	34	34	34	39	39
21	2005-02-09-09.04.12.996833	RDAL	1	33	33	33	37	37
22	2005-02-09-09.04.15.249478	RDAL	1	39	39	39	44	44
23	2005-02-09-09.04.15.972596	RDAL	1	38	38	38	41	41
25	2005-02-09-09.04.18.774776	RDAL	1	35	35	35	40	40
27	2005-02-09-09.04.20.534626	RDAL	1	34	34	34	37	37

Records 1 - 13 of 15

Table of Contents | Previous | Next

5.10 Data queue activity reports

This analysis shows all data queue sends and receives. It indicates the causing job/thread, data queue and library names, and time of day each operation occurred.

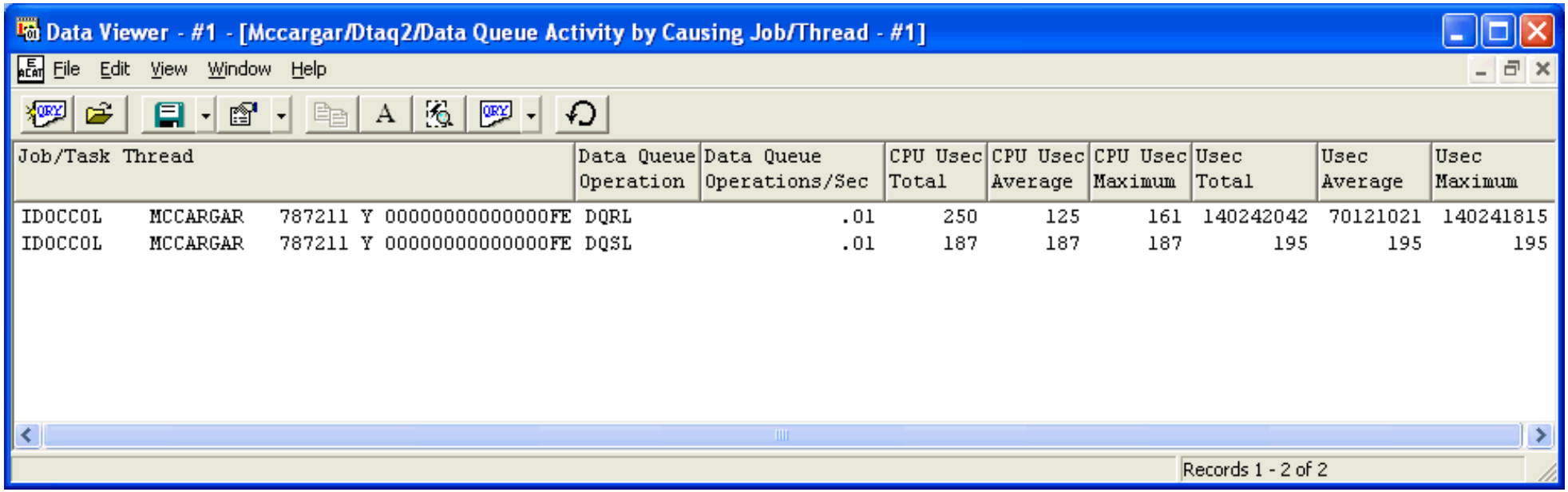
If the program events *MIENTRY and *MIEXIT are collected, then the analysis shows the calling program name, elapsed time, CPU, and disk I/O during each data queue operation.

Table of Contents | Previous | Next

5.10.1 Data queue activity by causing job/thread

Description:

Example:



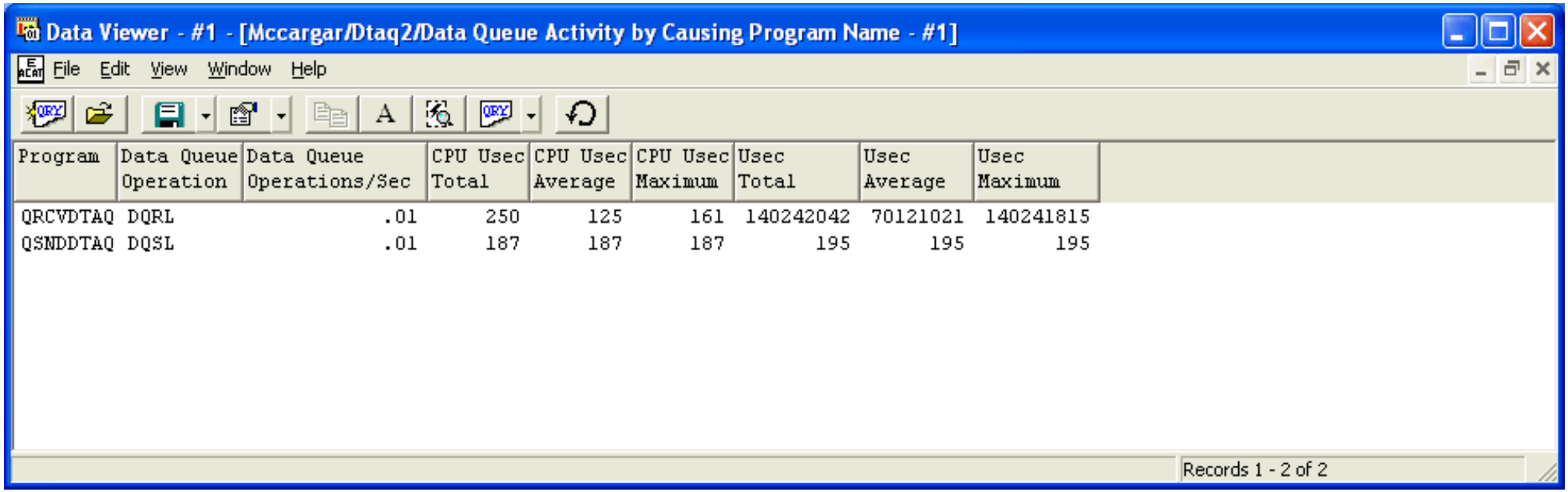
Job/Task	Thread				Data Queue Operation	Data Queue Operations/Sec	CPU Usec Total	CPU Usec Average	CPU Usec Maximum	Usec Total	Usec Average	Usec Maximum
IDOCCOL	MCCARGAR	787211	Y	00000000000000FE	DQRL	.01	250	125	161	140242042	70121021	140241815
IDOCCOL	MCCARGAR	787211	Y	00000000000000FE	DQSL	.01	187	187	187	195	195	195

Table of Contents | Previous | Next

5.10.2 Data queue activity by causing program name

Description:

Example:



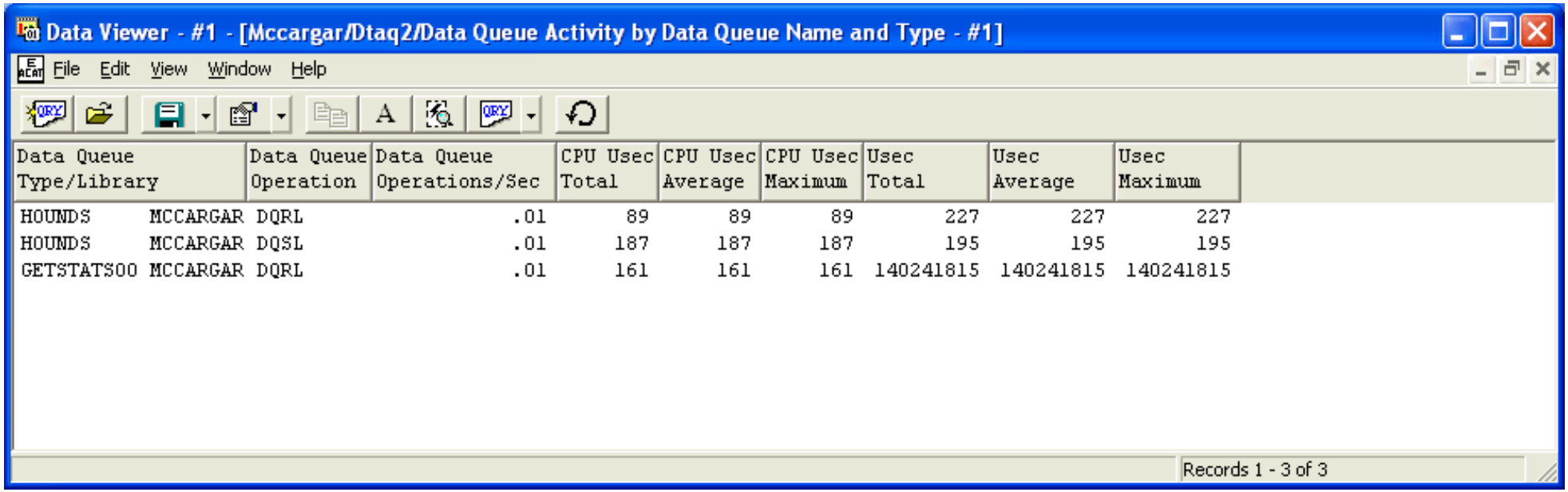
Program	Data Queue Operation	Data Queue Operations/Sec	CPU Usec Total	CPU Usec Average	CPU Usec Maximum	Usec Total	Usec Average	Usec Maximum
QRCVDTAQ	DQRL	.01	250	125	161	140242042	70121021	140241815
QSNDDTAQ	DQSL	.01	187	187	187	195	195	195

Table of Contents | Previous | Next

5.10.3 Data queue activity by data queue name and type

Description:

Example:



Data Queue Type/Library		Data Queue Operation	Data Queue Operations/Sec	CPU Usec Total	CPU Usec Average	CPU Usec Maximum	Usec Total	Usec Average	Usec Maximum
HOUNDS	MCCARGAR	DQRL	.01	89	89	89	227	227	227
HOUNDS	MCCARGAR	DQSL	.01	187	187	187	195	195	195
GETSTATS00	MCCARGAR	DQRL	.01	161	161	161	140241815	140241815	140241815

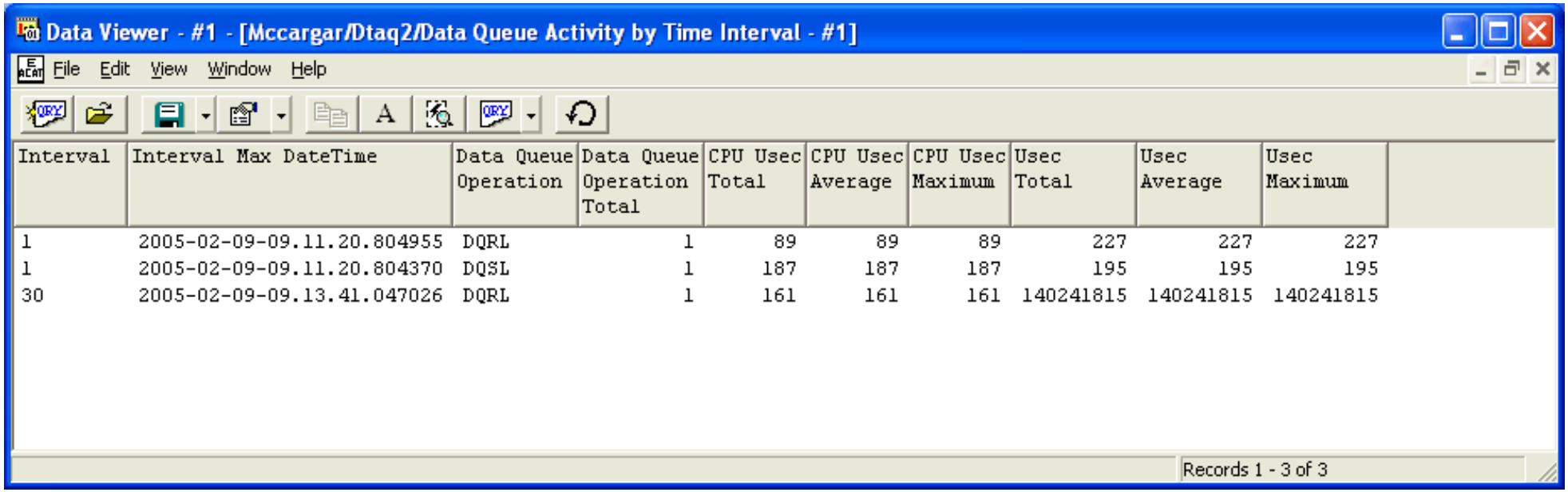
Records 1 - 3 of 3

Table of Contents | Previous | Next

5.10.4 Data queue activity by time interval

Description:

Example:



Interval	Interval Max DateTime	Data Queue Operation	Data Queue Operation Total	CPU Usec Total	CPU Usec Average	CPU Usec Maximum	Usec Total	Usec Average	Usec Maximum
1	2005-02-09-09.11.20.804955	DQRL	1	89	89	89	227	227	227
1	2005-02-09-09.11.20.804370	DQSL	1	187	187	187	195	195	195
30	2005-02-09-09.13.41.047026	DQRL	1	161	161	161	140241815	140241815	140241815

Table of Contents Previous Next

Chapter 6 PTDV

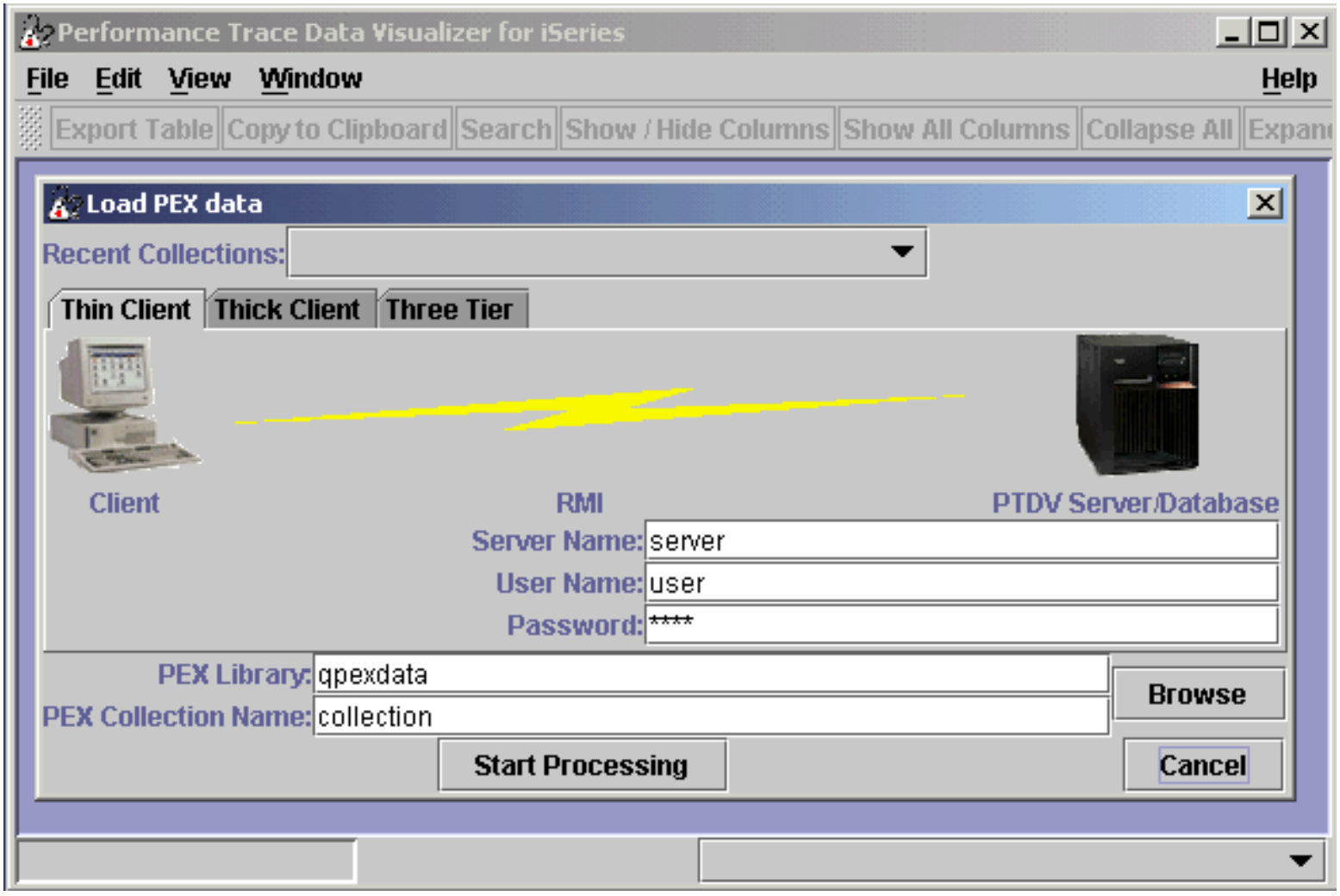
This chapter provides an overview of the modes of operation and interfaces for the iDoctor for iSeries PTDV component.

Table of Contents | Previous | Next

6.1 Starting PTDV

Once PTDV has been installed using its installer, PTDV can be launched by going through Start->iDoctor for iSeries->PTDV.

After starting PTDV, you will see a Load Dialog where you can enter information about the server where the performance data exists. Depending on the run mode, you should enter the system name, user id, and password for the system containing the library with the PEX data. If you are running in Three Tier mode, you will also need to enter the name of another system, user id, and password for the system where you want to do the processing.



Sometimes after starting PTDV to process a collection you can't remember exactly what the library and/or collection name is. After filling in the system, user name, and password, you can click on the Browse button, which will allow you to look at the set of PEX libraries and the data they contain. From this list you can select a collection and the library and collection name will be filled into the load dialog box

automatically.

Table of Contents | Previous | Next

6.2 Modes of Operation

There are three basic modes of operation when using PTDV.

Thin Client

This is the default mode for PTDV. In this mode, there is a PTDV server which runs on your iSeries, and does most of the data processing. The PTDV client acts mostly as a presentation layer on your client system. This mode is generally the fastest method of running the PTDV, as all of the database access is done on the local system. Using this mode also allows you to process the largest number of events (currently the largest trace processed has been about 10 million events), as the data is kept on the iSeries, which can generally handle large amounts of data more effectively than a PC. The primary disadvantage to this mode is that a connection to your iSeries must be active the entire time you are using PTDV, and some processing time on the server will be used. Therefore, this method is not ideal for viewing data on production systems. In addition, the PTDV server must be installed on the iSeries, which is sometimes not practical/desirable.

Thick Client

This mode is very similar to the way that the original version of PTDV ran. In this case, the data processing code and the presentation are both done on your client system, and the iSeries is used only to access the database. This mode tends to be somewhat slower than thin-client mode, and can require more network bandwidth. In addition, the amount of data which can be processed is limited to the amount of memory on your PC -- most systems will run out of memory if more than 300,000 events (or sometimes even less) are processed. However, the thick client mode can be useful when dealing with small collections (of only a few thousand events), since it does not require any PTDV code to be installed on the iSeries. Since it does not require a connection to the server after the events are initially processed, this mode can also be useful when it is not acceptable to do processing on the server over an extended period of time.

Three Tier

This mode is a mix between thin client and thick client. In this case, one iSeries holds the PEX data. Another iSeries runs the PTDV server, and your client system runs the PTDV client. This results in a mix of the advantages and disadvantages above. Processing can sometimes take even longer than in the Thick Client mode. However, the Three Tier mode can process as many events as the Thin Client mode.

In addition, the connection to the first iSeries (database server) is closed after the initial processing (as in Thick Client), so this mode can work well for retrieving data from production systems. In general, the Thin Client mode is preferable unless it is necessary to do only minimal processing on the system with the database. Even in this case, it is generally better to end PEX with the *FILE option, and then move the database over to another iSeries which can be used for the processing.

Run Modes Quick Reference Chart

Run Mode	Speed	Number of Events Possible	iSeries Setup, JV1 LPP Required?	Active Connection to iSeries Required?
Thin Client	Fastest	Most (~10 millions)	Yes	At all times
Thick Client	Medium	Least (low 100s of thousands)	No	Only during event processing
Three Tier	Slowest	Most (low millions)	Yes*	At all times*

Table of Contents | Previous | Next

6.3 Processing Modes for Trace Collections

In some cases, a TRACE collection could be very large and can take a very long time to process. For some collection types, valuable summaries may be available which can be processed and displayed in a much shorter period of time. For these types of collections, the user is provided with the option of **Full Trace Processing**, which provides significant detail about the collection, but may take longer to process, or **Quick View Processing**, where the user is presented with a list of the available views that can be processed and displayed for the current collection.

Full Trace Processing

As the name implies, Full Trace Processing consists of processing the collection data in detail. Each event is processed sequentially in timestamp order, and corresponding event pairs are matched and any significant deltas for the associated performance data are computed. For example, each procedure and method have an entry and exit event. The timestamp and cycle counts are record at the point of the entry and exit. As a result, the inline and cumulative elapsed time and cycle counts can be computed for each procedure and method found in the collection. Other event pairs include object lock and unlock, thread notify and wait, task switch out and task switch in, heap create and delete, and object create and delete.

This type of processing provides the most detailed information about the collection, but also requires the most processing time and storage on your client. Except in cases where your trace collection contains fewer than 300,000 events, you should only use Full Trace Processing in Thin Client or Three Tier mode. In Thick Client mode, you will most likely get a memory error on your client. Refer to Modes of Operation for more information on these processing modes.

Quick View Processing

For some types of collections, a one or more "Quick Views" may be available to choose from. A Quick View is a summarized view of the collection data, which usually can be processed in a much shorter amount of time that if processing is done through Full Trace Processing. In many cases, the Quick View might provide sufficient information for analysis.

One example of the benefit of Quick View processing is with a Trace collection containing object create events. With these types of events, it is possible to generate a collection with several million

events in a short period of time. With the Quick View for this collection, you can quickly generate a table containing a summary of the object creates and deletes, grouped by class name, along with create and delete sizes.

With Quick View processing, the frames and panes generated are limited to the selected set of views depending on the type of events in your collection. Refer to the section on Panes for more information on the types of collections that have Quick Views.

Table of Contents | Previous | Next

6.4 Navigating through Frames and Panes

One of the unique features of PTDV is the way in which the user navigates through the performance data. Data is organized into a set of tables or trees, which are then grouped together into frames. The contents of the frames and panes depends on the types of events in the collection being processed. As you navigate through the data and find areas of interest, you can get to more detailed information by selecting a row from one of the panes and clicking on it.

The Main Frame is the starting point for processing. The panes available on the Main Frame give you views of the data for the entire collection. After selecting a pane, you can click on a row in the table or tree to get to its corresponding frame, which will then display more details on the item you are interested in.

5.2.1 Frames

A frame is a set of one or more tabbed panes, and is usually associated with an entity in the collection. The data in the frame may be scoped to a level other than the entire collection; it might represent data for only a single job or thread.

The following frames are available:

5.2.1.1 Main Trace Frame

For a trace collection, the initial panes are:

- **Collection Information Pane**
  The collection information is displayed in an Informational Pane, which contains details on the current collection.

- **Event Counts Pane**
  This pane contains information about the events that appear in the current trace collection. It provides the event counts organized by event type and subtype.

- **Job/Thread Pane**
  The Job/Thread panedisplays the jobs and threads in the collection, along with performance data for each job and thread.

Once processing has been completed the following panes are displayed:

- **Procedure Summary Pane**
  Information about the procedures used within the collection is displayed in a Procedure Summary Pane. This data represents procedure information over the entire collection.

- **Object Summary Pane**
  A summary of the objects used in this collection are displayed as an Object Summary Pane.

- **Java Transform and JIT compile Pane**
  A summary of the JIT compile start and end events that occurred during the collection. This appears only if there are Java transform events in the collection.

5.2.1.2. Job Frame

The Job Frame contains information about a specific Job within the collection. The Job Frame consists of the following panes:

- **Job Information Pane**
  This pane is an Informational Pane, providing details on the job associated with this frame.

- **Threads Pane**
  A table containing the threads associated with the current job is the Threads Pane.

- **Procedure Summary Pane**
  The Procedure Summary Panefor this job contains performance data collected for the procedures within the current job.

- **Object Summary Pane**
  The Object Summary Panefor this job includes information on the objects used by the current job, organized by class name (object type).

- **Garbage Collection Sweep Pane**
  This pane is optional depending on whether the collection contains garbage collection

sweep events.Garbage Collection Sweep Pane

The WebSphere Summary Pane summarizes information about the WebSphere events that have occurred in this collection within the current job. This pane will appear only if there are WebSphere events in the current collection.

5.2.1.3. Thread Frame

The Thread Frame contains information about a specific Thread or Task within the collection. The Thread Frame consists of the following panes:

- **Procedure Summary Pane**
  The Procedure Summary Paneprovides summary information about the procedures/methods appearing within the current thread or task.

- **Call Trace pane**
  The Call Trace Panefor a Thread Frame displays the call flow for all procedures/methods within the current thread.

- **Object Summary Pane**
  The Object Summary Panefor a Thread contains the summary information about the objects appearing within the thread associated with the current frame. Data in this table is grouped by class name (object type).

5.2.1.4. Procedure Summary Frame

The procedure summary frame contains several panes with detailed information about a specific procedure within a given collection level. The information in the panes for a given frame might contain data for the entire collection, or for a specific job or thread. This frame consists of the following panes:

- **Procedure Information Pane**
  The Procedure Information Pane is an Informational Panewhich provides summary information about the procedure within the current collection level.

- **Call site pane**
  The call site pane is a Procedure Call Pane. It contains all of the individual call sites (i.e., single calls of the procedure) for the current procedure within the frame's collection level.

- **Cumulative callers pane**
  This table is a Procedure Summary Panewhich contains the summarized information about the callers of the current procedure within the frame's collection level.

- **Cumulative callees pane**
  This table is also a Procedure Summary Panewhich contains the summarized information about the callees (procedures that are called by the current procedure and its descendants) of the current procedure.

- **Inline objects pane**
  This table is an Object Summary Pane, containing the summarized information about the java objects that are associated with the java events that have occurred inline within the current procedure.

- **Cumulative objects pane**
  The cumulative objects table is also a Object Summary Pane, containing information about the objects that are associated with the java events that occur by the current procedure or procedures that are called cumulatively by the current procedure.

5.2.1.5. Procedure Call Frame

The procedure frame contains information about a single procedure call within the collection. It consists of the following panes:

- **Information pane**
  This pane is an Informational Paneproviding details about the procedure represented by this frame.

- **Call ancestors pane**
  This pane provides a view of the call stack at the point of the procedure call associated with the current frame. This table is a Procedure Call Pane.

- **Call subtree pane**
  This pane contains the tree representation of the calls that occur beneath the current procedure. It is actually a subtree of the entire call tree, with the current procedure as its root. It is a form of a Call Trace Pane.

- **Cumulative callees pane**
  This table is a Procedure Summary Panecontaining the summarized information about the callees (procedures that are called by the current procedure and its descendants) of

the current procedure.

- **Inline objects pane**
  This table is an Object Summary Panecontaining the summarized information about the java objects that are associated with java events within the current procedure. The data is in this table is organized by object type.

- **Cumulative objects pane**
  This table is also an Object Summary Panecontaining the summarized information about the java objects that are associated with java events within the cumulative callees of the current procedure. The data is in this table is organized by object type.

5.2.1.6. Object Group Frame

The Object Group Frame contains information about objects of a single class, containing data over a specific collection level. The Object Group Frame consists of the following panes:

- **Object Group Information Pane**
  This pane is an Informational Pane, providing some summary information about the objects within the group for the current frame.

- **Associated methods**
  This table is an Object Summary Pane, containing information about the objects for the class for the current object group organized by method name. The data within each row represents the objects that have associated events within a specified method; for example, a row of the table contains information about the create and all locks for objects within a given class for a specific method.

- **Object instances**
  This table is an Object Instances Pane, containing the set of individual objects within the current object group. In some cases, this table can be very large, and is usually represented as a dynamic data pane.

5.2.1.7. Object Info Frame

The Object Info Frame contains information about a single object instance. In most cases, this frame will contain all information for this object over the entire collection. The Object Info Frame consists of the following panes:

- **Create Call Ancestors Pane**

This pane provides a view of the call stack at the point of the object create event for the current object. This table is a Procedure Call Pane.The table for this pane is only generated when there are object create events and java method entry and exit events in the collection.

- **Object Locks Pane**
  This table contains the list of all locks that have occurred on the object associated with the current frame. This data is displayed in an Object Locks Pane. The information in this table is only generated when there are object lock events in the collection.

- **Notify/Wait Events Pane**
  This table is a Object Notify/Wait Pane, containing information about thread notify or thread wait events that have occurred on the current object. The information in this table is only generated when there are thread notify or thread wait events in the collection.

5.2.1.8. Object Lock Frame

The Object Lock Frame contains information about a single object lock, which consists of a grouping of want, get, and release lock events on a single object. It consists of the following panes:

- **Lock Information Pane**
  This is an Informational Paneproviding information about the lock associated with the current frame.
- **Concurrent Locks in Thread**
  This is an Object Lock Pane, containing the locks that occur within the current thread during the hold time for the current lock.
- **Concurrent Locks in Job**
  This is also an Object Lock Pane, containing the locks that occur within the current job during the hold time for the current lock.
- **Locks During Hold Pane**
  This is a Object Lock Pane, containing all the locks that occur on the current object during the hold time of the current lock.
- **Methods During Hold Pane**
  This table is a Call Trace Pane, displaying in tree form the set of methods that are called while the current lock is being held.

5.2.1.9. Event Summary Frame

The Event Summary Frame summarizes information about a specific event that is included in the collection. It contains the following panes:

- **WebSphere Event Information Pane**
  This is an Informational Paneproviding detailed information about the current WebSphere event.

- **Call ancestors pane**
  This pane provides a view of the call stack when the WebSphere event occurred. This table is a Procedure Call Pane.

5.2.1.10. WebSphere Event Frame

The WebSphere Event Frame contains detailed information about an instance of a WebSphere event. It contains the following panes:

- **WebSphere Event Information Pane**
  This is an Informational Paneproviding detailed information about the current WebSphere event.

- **Call ancestors pane**
  This pane provides a view of the call stack when the WebSphere event occurred. This table is a Procedure Call Pane.

5.2.1.11. WebSphere Event Summary Frame

The WebSphere Event Summary Frame contains summary information about a single WebSphere event within a specific collection level. FOr example, it might contain the set of WebSphere events that occurred in the entire collection, or for a specific job or thread. It contains the following panes:

- **Individual WebSphere Events Pane**
  This is an WebSphere Event Paneproviding information about the individual WebSphere events for the type associated with the current frame.

- **Cumulative Callers**
  This is an Procedure Summary Pane, containing counts and other details about the procedures that called the current WebSphere event.

5.2. Panes

A frame consists of one or more panes, which can be selected by clicking on its tab. A pane can contain a table or tree of data. A tree is used in those cases where the data is best represented in a parent/child relationship, such as a call trace. For both tables and trees, the data is organized by rows and columns, and in many cases, clicking on a row of a table or tree will generate a new frame containing more detailed information about that row.

5.2.1. Informational Pane

In most frames, the first pane contains a description for that frame. On the main frame, the informational pane contains information about the collection itself. Each frame type has a specific format.

5.2.2. Event counts pane

This pane contains information about the events that appear in the current trace collection. It provides the event counts organized by event type and subtype.

5.2.3. Job/Thread Pane

This pane appears in the main frame for most collection types. It contains information about the jobs and threads (tasks) in the collection currently being processed. It displays information in a tree format, showing the relationships between jobs and threads.

This table contains performance data for each job and thread in the collection, such as cycle counts, instruction counts, total time, total events, and other information depending on the events found in the collection.

Double-clicking on a job or thread row in this table will generate and show the corresponding Job Frame or Thread Frame. These frames provide performance information scoped to the selected Job or Thread.

5.2.4. Threads Pane

The Threads Pane is a table containing information about the threads (tasks) associated with the job for the current frame. Each row of the table corresponds to a thread, and contains performance data, such as cycles, instructions, time, create and delete timestamps, as well as status flags about the thread.

This table is generated and available for all collections. The availability of some of the data is dependent on the events in the collection, such as thread create and delete timestamps.

Double-clicking on a row of this table will generate and display the Thread Frame associated with the selected thread.

5.2.5. Procedure Summary Pane

This pane contains summary information about the procedures within the collection. This data can be scoped to a specific collection level, such as a job, thread, or the entire collection.

The rows in this table correspond to a specific procedure. Each row of the table contains the procedure name along with the associated performance data, such as times called, time spent inline for the procedure, time spent cumulatively for the procedure (i.e., including time for all procedures called between entry and exit of the procedure), cycles, etc. Other data is available, depending on the events in the collection, such as object create information, object lock information, or WebSphere data.

Information for this table will be generated only if there are program events in the collection being processed. This includes Java entry and exit events, MI program entry and exit events, and Procedure entry and exit events.

Double-clicking on a row in this table will bring up the corresponding Procedure Summary Frame.

5.2.6. Procedure Call Pane

The procedure call pane is a table containing individual procedure call instances (i.e., a single call of a procedure). Each row of the table contains performance data about a single call, such as time, cycles, and other event counts for events from the collection.

This table can only be generated if there are program events in the current collection.

Double-clicking on a row of this pane will generate and display the Procedure Call Frame for the selected procedure call.

5.2.7. Call Trace Pane

This pane contains a tree representing the call flow based on the entry/exit event information in the collection. It displays the parent/child call relationships for a subset of the collection. Each row corresponds to a procedure call within the tree, and contains performance data about the specific procedure call. Other information may also be available depending on the types of events that are present in the collection, such as object events or WebSphere events.

Double-clicking on a procedure row of this table will generate and display the Procedure Call Frame

associated with the selected procedure call.

5.2.8. Object Summary Pane

The Object Summary Pane is a table containing information on objects that are associated with the java events occurring in the current collection. The collection may contain object create, object lock, thread notify, or class load events, all of which will have a class associated with them. The data in this table represents the objects that occur in the events of the collection, grouped by class name (i.e., object type). On an associated method pane, the data may be further grouped by method name.

The data in this table is generated only when there are java object events in the collection, such as object create, object lock, or thread notify.

Double-clicking on a row of this pane will generate and display the Object Group Frame for the selected class name.

5.2.9. Object Instances Pane

This table contains a set of individual object instances. It will contain information about the create and delete timestamps, the creating procedure, and summary information about the locks for the object. The data in this table will depend on the types of object events found in the collection.

Double-clicking on a row in this table will generate and display the corresponding Object Info Frame for the selected object.

5.2.10. Object Lock Pane

This pane contains a table of locks. A "lock" is considered to be a set of want, get, and/or release events operating on a single object, representing the actual lock/unlock behavior for an object. The data in this table provides the timestamps for these events, the hold and wait times, as well as the name of the method, thread, and job locking the object.

Double-clicking on a procedure row of this table will generate and display the Object Lock Frame for the selected lock.

5.2.11. Object Notify/Wait Pane

This pane contains a table of wait and notify events that have occurred for a specific object. One of these events will occur whenever the java.lang.Thread.notify(), notifyAll(), or wait() methods have been invoked. The table contains information about the notify or notifyAll, the wait time, the method that invoked the notify or wait event, and links up the corresponding wait and notify events.

5.2.12. Garbage Collector Sweep Pane

This table contains information about the garbage collector sweep events that have occurred in the collection. This data is similar to what is output to the screen when using the *verbose option on the java command.

This table will be generated only if there are garbage collection sweep events in the trace collection being processed.

5.2.13. WebSphere Summary Pane

The WebSphere Summary Pane is a table containing summary information for all the WebSphere events that have occurred within a specific collection level.

This pane can only be generated when there are WebSphere events in the collection.

Double-clicking on a row of this table will generate and display the WebSphere Event Summary Frame associated with the selected WebSphere event.

5.2.14. WebSphere Event Pane

The WebSphere Event Pane is a table containing information on the individual WebSphere events of the same type as the current WebSphere Event Frame. It contains detailed event information based on the data in each WebSphere event.

This pane can only be generated when there are WebSphere events in the collection.

Double-clicking on a row of this table will generate and display the WebSphere Event Frame associated with the selected individual WebSphere event.

5.2.15. Java Transform/JIT Compile Pane

The Java Transform/JIT compile pane displays information about the JIT compile events that have occurred in the collection. This provides information about the methods that were compiled by the JIT, when the compile started, and the length of the compile.

Table of Contents | Previous | Next

6.4.1 Frames

One of the unique features of PTDV is the way in which the user navigates through the performance data. Data is organized into a set of tables or trees, which are then grouped together into frames. The contents of the frames and panes depends on the types of events in the collection being processed. As you navigate through the data and find areas of interest, you can get to more detailed information by selecting a row from one of the panes and clicking on it.

The Main Frame is the starting point for processing. The panes available on the Main Frame give you views of the data for the entire collection. After selecting a pane, you can click on a row in the table or tree to get to its corresponding frame, which will then display more details on the item you are interested in.

5.2.1 Frames

A frame is a set of one or more tabbed panes, and is usually associated with an entity in the collection. The data in the frame may be scoped to a level other than the entire collection; it might represent data for only a single job or thread.

The following frames are available:

5.2.1.1 Main Trace Frame

For a trace collection, the initial panes are:

- **Collection Information Pane**
  The collection information is displayed in an Informational Pane, which contains details on the current collection.

- **Event Counts Pane**
  This pane contains information about the events that appear in the current trace collection. It provides the event counts organized by event type and subtype.

- **Job/Thread Pane**
  The Job/Thread panedisplays the jobs and threads in the collection, along with

performance data for each job and thread.

Once processing has been completed the following panes are displayed:

- **Procedure Summary Pane**
  Information about the procedures used within the collection is displayed in a Procedure Summary Pane. This data represents procedure information over the entire collection.

- **Object Summary Pane**
  A summary of the objects used in this collection are displayed as an Object Summary Pane.

- **Java Transform and JIT compile Pane**
  A summary of the JIT compile start and end events that occurred during the collection. This appears only if there are Java transform events in the collection.

5.2.1.2. Job Frame

The Job Frame contains information about a specific Job within the collection. The Job Frame consists of the following panes:

- **Job Information Pane**
  This pane is an Informational Pane, providing details on the job associated with this frame.

- **Threads Pane**
  A table containing the threads associated with the current job is the Threads Pane.

- **Procedure Summary Pane**
  The Procedure Summary Panefor this job contains performance data collected for the procedures within the current job.

- **Object Summary Pane**
  The Object Summary Panefor this job includes information on the objects used by the current job, organized by class name (object type).

- **Garbage Collection Sweep Pane**
  This pane is optional depending on whether the collection contains garbage collection sweep events.Garbage Collection Sweep Pane

The WebSphere Summary Pane summarizes information about the WebSphere events that have occurred in this collection within the current job. This pane will appear only if there are WebSphere events in the current collection.

5.2.1.3. Thread Frame

The Thread Frame contains information about a specific Thread or Task within the collection. The Thread Frame consists of the following panes:

- **Procedure Summary Pane**
  The Procedure Summary Paneprovides summary information about the procedures/methods appearing within the current thread or task.

- **Call Trace pane**
  The Call Trace Panefor a Thread Frame displays the call flow for all procedures/methods within the current thread.

- **Object Summary Pane**
  The Object Summary Panefor a Thread contains the summary information about the objects appearing within the thread associated with the current frame. Data in this table is grouped by class name (object type).

5.2.1.4. Procedure Summary Frame

The procedure summary frame contains several panes with detailed information about a specific procedure within a given collection level. The information in the panes for a given frame might contain data for the entire collection, or for a specific job or thread. This frame consists of the following panes:

- **Procedure Information Pane**
  The Procedure Information Pane is an Informational Panewhich provides summary information about the procedure within the current collection level.

- **Call site pane**
  The call site pane is a Procedure Call Pane. It contains all of the individual call sites (i.e., single calls of the procedure) for the current procedure within the frame's collection level.

- **Cumulative callers pane**

This table is a Procedure Summary Panewhich contains the summarized information about the callers of the current procedure within the frame's collection level.

- **Cumulative callees pane**
  This table is also a Procedure Summary Panewhich contains the summarized information about the callees (procedures that are called by the current procedure and its descendants) of the current procedure.

- **Inline objects pane**
  This table is an Object Summary Pane, containing the summarized information about the java objects that are associated with the java events that have occurred inline within the current procedure.

- **Cumulative objects pane**
  The cumulative objects table is also a Object Summary Pane, containing information about the objects that are associated with the java events that occur by the current procedure or procedures that are called cumulatively by the current procedure.

5.2.1.5. Procedure Call Frame

The procedure frame contains information about a single procedure call within the collection. It consists of the following panes:

- **Information pane**
  This pane is an Informational Paneproviding details about the procedure represented by this frame.

- **Call ancestors pane**
  This pane provides a view of the call stack at the point of the procedure call associated with the current frame. This table is a Procedure Call Pane.

- **Call subtree pane**
  This pane contains the tree representation of the calls that occur beneath the current procedure. It is actually a subtree of the entire call tree, with the current procedure as its root. It is a form of a Call Trace Pane.

- **Cumulative callees pane**
  This table is a Procedure Summary Panecontaining the summarized information about the callees (procedures that are called by the current procedure and its descendants) of the current procedure.

- **Inline objects pane**
  This table is an Object Summary Panecontaining the summarized information about the java objects that are associated with java events within the current procedure. The data is in this table is organized by object type.

- **Cumulative objects pane**
  This table is also an Object Summary Panecontaining the summarized information about the java objects that are associated with java events within the cumulative callees of the current procedure. The data is in this table is organized by object type.

5.2.1.6. Object Group Frame

The Object Group Frame contains information about objects of a single class, containing data over a specific collection level. The Object Group Frame consists of the following panes:

- **Object Group Information Pane**
  This pane is an Informational Pane, providing some summary information about the objects within the group for the current frame.

- **Associated methods**
  This table is an Object Summary Pane, containing information about the objects for the class for the current object group organized by method name. The data within each row represents the objects that have associated events within a specified method; for example, a row of the table contains information about the create and all locks for objects within a given class for a specific method.

- **Object instances**
  This table is an Object Instances Pane, containing the set of individual objects within the current object group. In some cases, this table can be very large, and is usually represented as a dynamic data pane.

5.2.1.7. Object Info Frame

The Object Info Frame contains information about a single object instance. In most cases, this frame will contain all information for this object over the entire collection. The Object Info Frame consists of the following panes:

- **Create Call Ancestors Pane**
  This pane provides a view of the call stack at the point of the object create event for the

current object. This table is a Procedure Call Pane.The table for this pane is only generated when there are object create events and java method entry and exit events in the collection.

- **Object Locks Pane**
  This table contains the list of all locks that have occurred on the object associated with the current frame. This data is displayed in an Object Locks Pane. The information in this table is only generated when there are object lock events in the collection.

- **Notify/Wait Events Pane**
  This table is a Object Notify/Wait Pane, containing information about thread notify or thread wait events that have occurred on the current object. The information in this table is only generated when there are thread notify or thread wait events in the collection.

5.2.1.8. Object Lock Frame

The Object Lock Frame contains information about a single object lock, which consists of a grouping of want, get, and release lock events on a single object. It consists of the following panes:

- **Lock Information Pane**
  This is an Informational Paneproviding information about the lock associated with the current frame.
- **Concurrent Locks in Thread**
  This is an Object Lock Pane, containing the locks that occur within the current thread during the hold time for the current lock.
- **Concurrent Locks in Job**
  This is also an Object Lock Pane, containing the locks that occur within the current job during the hold time for the current lock.
- **Locks During Hold Pane**
  This is a Object Lock Pane, containing all the locks that occur on the current object during the hold time of the current lock.
- **Methods During Hold Pane**
  This table is a Call Trace Pane, displaying in tree form the set of methods that are called while the current lock is being held.

5.2.1.9. Event Summary Frame

The Event Summary Frame summarizes information about a specific event that is included in the collection. It contains the following panes:

- **WebSphere Event Information Pane**
  This is an Informational Paneproviding detailed information about the current WebSphere event.

- **Call ancestors pane**
  This pane provides a view of the call stack when the WebSphere event occurred. This table is a Procedure Call Pane.

5.2.1.10. WebSphere Event Frame

The WebSphere Event Frame contains detailed information about an instance of a WebSphere event. It contains the following panes:

- **WebSphere Event Information Pane**
  This is an Informational Paneproviding detailed information about the current WebSphere event.

- **Call ancestors pane**
  This pane provides a view of the call stack when the WebSphere event occurred. This table is a Procedure Call Pane.

5.2.1.11. WebSphere Event Summary Frame

The WebSphere Event Summary Frame contains summary information about a single WebSphere event within a specific collection level. FOr example, it might contain the set of WebSphere events that occurred in the entire collection, or for a specific job or thread. It contains the following panes:

- **Individual WebSphere Events Pane**
  This is an WebSphere Event Paneproviding information about the individual WebSphere events for the type associated with the current frame.

- **Cumulative Callers**
  This is an Procedure Summary Pane, containing counts and other details about the procedures that called the current WebSphere event.

5.2. Panes

A frame consists of one or more panes, which can be selected by clicking on its tab. A pane can contain a table or tree of data. A tree is used in those cases where the data is best represented in a parent/child relationship, such as a call trace. For both tables and trees, the data is organized by rows and columns, and in many cases, clicking on a row of a table or tree will generate a new frame containing more detailed information about that row.

5.2.1. Informational Pane

In most frames, the first pane contains a description for that frame. On the main frame, the informational pane contains information about the collection itself. Each frame type has a specific format.

5.2.2. Event counts pane

This pane contains information about the events that appear in the current trace collection. It provides the event counts organized by event type and subtype.

5.2.3. Job/Thread Pane

This pane appears in the main frame for most collection types. It contains information about the jobs and threads (tasks) in the collection currently being processed. It displays information in a tree format, showing the relationships between jobs and threads.

This table contains performance data for each job and thread in the collection, such as cycle counts, instruction counts, total time, total events, and other information depending on the events found in the collection.

Double-clicking on a job or thread row in this table will generate and show the corresponding Job Frame or Thread Frame. These frames provide performance information scoped to the selected Job or Thread.

5.2.4. Threads Pane

The Threads Pane is a table containing information about the threads (tasks) associated with the job for the current frame. Each row of the table corresponds to a thread, and contains performance data, such as cycles, instructions, time, create and delete timestamps, as well as status flags about the thread.

This table is generated and available for all collections. The availability of some of the data is dependent on the events in the collection, such as thread create and delete timestamps.

Double-clicking on a row of this table will generate and display the Thread Frame associated with

the selected thread.

5.2.5. Procedure Summary Pane

This pane contains summary information about the procedures within the collection. This data can be scoped to a specific collection level, such as a job, thread, or the entire collection.

The rows in this table correspond to a specific procedure. Each row of the table contains the procedure name along with the associated performance data, such as times called, time spent inline for the procedure, time spent cumulatively for the procedure (i.e., including time for all procedures called between entry and exit of the procedure), cycles, etc. Other data is available, depending on the events in the collection, such as object create information, object lock information, or WebSphere data.

Information for this table will be generated only if there are program events in the collection being processed. This includes Java entry and exit events, MI program entry and exit events, and Procedure entry and exit events.

Double-clicking on a row in this table will bring up the corresponding Procedure Summary Frame.

5.2.6. Procedure Call Pane

The procedure call pane is a table containing individual procedure call instances (i.e., a single call of a procedure). Each row of the table contains performance data about a single call, such as time, cycles, and other event counts for events from the collection.

This table can only be generated if there are program events in the current collection.

Double-clicking on a row of this pane will generate and display the Procedure Call Frame for the selected procedure call.

5.2.7. Call Trace Pane

This pane contains a tree representing the call flow based on the entry/exit event information in the collection. It displays the parent/child call relationships for a subset of the collection. Each row corresponds to a procedure call within the tree, and contains performance data about the specific procedure call. Other information may also be available depending on the types of events that are present in the collection, such as object events or WebSphere events.

Double-clicking on a procedure row of this table will generate and display the Procedure Call Frame associated with the selected procedure call.

5.2.8. Object Summary Pane

The Object Summary Pane is a table containing information on objects that are associated with the java events occurring in the current collection. The collection may contain object create, object lock, thread notify, or class load events, all of which will have a class associated with them. The data in this table represents the objects that occur in the events of the collection, grouped by class name (i.e., object type). On an associated method pane, the data may be further grouped by method name.

The data in this table is generated only when there are java object events in the collection, such as object create, object lock, or thread notify.

Double-clicking on a row of this pane will generate and display the Object Group Frame for the selected class name.

5.2.9. Object Instances Pane

This table contains a set of individual object instances. It will contain information about the create and delete timestamps, the creating procedure, and summary information about the locks for the object. The data in this table will depend on the types of object events found in the collection.

Double-clicking on a row in this table will generate and display the corresponding Object Info Frame for the selected object.

5.2.10. Object Lock Pane

This pane contains a table of locks. A "lock" is considered to be a set of want, get, and/or release events operating on a single object, representing the actual lock/unlock behavior for an object. The data in this table provides the timestamps for these events, the hold and wait times, as well as the name of the method, thread, and job locking the object.

Double-clicking on a procedure row of this table will generate and display the Object Lock Frame for the selected lock.

5.2.11. Object Notify/Wait Pane

This pane contains a table of wait and notify events that have occurred for a specific object. One of these events will occur whenever the java.lang.Thread.notify(), notifyAll(), or wait() methods have been invoked. The table contains information about the notify or notifyAll, the wait time, the method that invoked the notify or wait event, and links up the corresponding wait and notify events.

5.2.12. Garbage Collector Sweep Pane

This table contains information about the garbage collector sweep events that have occurred in the collection. This data is similar to what is output to the screen when using the *verbose option on the java command.

This table will be generated only if there are garbage collection sweep events in the trace collection being processed.

5.2.13. WebSphere Summary Pane

The WebSphere Summary Pane is a table containing summary information for all the WebSphere events that have occurred within a specific collection level.

This pane can only be generated when there are WebSphere events in the collection.

Double-clicking on a row of this table will generate and display the WebSphere Event Summary Frame associated with the selected WebSphere event.

5.2.14. WebSphere Event Pane

The WebSphere Event Pane is a table containing information on the individual WebSphere events of the same type as the current WebSphere Event Frame. It contains detailed event information based on the data in each WebSphere event.

This pane can only be generated when there are WebSphere events in the collection.

Double-clicking on a row of this table will generate and display the WebSphere Event Frame associated with the selected individual WebSphere event.

5.2.15. Java Transform/JIT Compile Pane

The Java Transform/JIT compile pane displays information about the JIT compile events that have occurred in the collection. This provides information about the methods that were compiled by the JIT, when the compile started, and the length of the compile.

Table of Contents | Previous | Next

6.4.2 Panes

A frame consists of one or more panes, which can be selected by clicking on its tab. Once the tab of a pane has been selected, the table or tree contained in that pane is displayed. A tree is used in those cases where the data is best represented in a parent/child relationship, such as a call trace. For both tables and trees, the data is organized by rows and columns, and in many cases, clicking on a row of a table or tree will generate a new frame containing more detailed information about that row.

5.2.1. Informational Pane

In most frames, the first pane contains a description for that frame. On the main frame, the informational pane contains information about the collection itself. Each frame type has a specific format.

5.2.2. Event counts pane

This pane contains information about the events that appear in the current trace collection. It provides the event counts organized by event type and subtype.

5.2.3. Job/Thread Pane

This pane appears in the main frame for most collection types. It contains information about the jobs and threads (tasks) in the collection currently being processed. It displays information in a tree format, showing the relationships between jobs and threads.

This table contains performance data for each job and thread in the collection, such as cycle counts, instruction counts, total time, total events, and other information depending on the events found in the collection.

Double-clicking on a job or thread row in this table will generate and show the corresponding Job Frame or Thread Frame. These frames provide performance information scoped to the selected Job or Thread.

The following panes are available when running in **Full Trace Processing** mode:

5.2.4. Threads Pane

The Threads Pane is a table containing information about the threads (tasks) associated with the job for the current frame. Each row of the table corresponds to a thread, and contains performance data, such as cycles, instructions, time, create and delete timestamps, as well as status flags about the thread.

This table is generated and available for all collections. The availability of some of the data is dependent on the events in the collection, such as thread create and delete timestamps.

Double-clicking on a row of this table will generate and display the Thread Frame associated with the selected thread.

5.2.5. Procedure Summary Pane

This pane contains summary information about the procedures within the collection. This data can be scoped to a specific collection level, such as a job, thread, or the entire collection.

The rows in this table correspond to a specific procedure. Each row of the table contains the procedure name along with the associated performance data, such as times called, time spent inline for the procedure, time spent cumulatively for the procedure (i.e., including time for all procedures called between entry and exit of the procedure), cycles, etc. Other data is available, depending on the events in the collection, such as object create information, object lock information, or WebSphere data.

Information for this table will be generated only if there are program events in the collection being processed. This includes Java entry and exit events, MI program entry and exit events, and Procedure entry and exit events.

Double-clicking on a row in this table will bring up the corresponding Procedure Summary Frame.

5.2.6. Procedure Call Pane

The procedure call pane is a table containing individual procedure call instances (i.e., a single call of a procedure). Each row of the table contains performance data about a single call, such as time, cycles, and other event counts for events from the collection.

This table can only be generated if there are program events in the current collection.

Double-clicking on a row of this pane will generate and display the Procedure Call Frame for the selected procedure call.

5.2.7. Call Trace Pane

This pane contains a tree representing the call flow based on the entry/exit event information in the collection. It displays the parent/child call relationships for a subset of the collection. Each row corresponds to a procedure call within the tree, and contains performance data about the specific procedure call. Other information may also be available depending on the types of events that are present in the collection, such as object events or WebSphere events.

Double-clicking on a procedure row of this table will generate and display the Procedure Call Frame associated with the selected procedure call.

5.2.8. Object Summary Pane

The Object Summary Pane is a table containing information on objects that are associated with the java events occurring in the current collection. The collection may contain object create, object lock, thread notify, or class load events, all of which will have a class associated with them. The data in this table represents the objects that occur in the events of the collection, grouped by class name (i.e., object type). On an associated method pane, the data may be further grouped by method name.

The data in this table is generated only when there are java object events in the collection, such as object create, object lock, or thread notify.

Double-clicking on a row of this pane will generate and display the Object Group Frame for the selected class name.

5.2.9. Object Instances Pane

This table contains a set of individual object instances. It will contain information about the create and delete timestamps, the creating procedure, and summary information about the locks for the object. The data in this table will depend on the types of object events found in the collection.

Double-clicking on a row in this table will generate and display the corresponding Object Info Frame for the selected object.

5.2.10. Object Lock Pane

This pane contains a table of locks. A "lock" is considered to be a set of want, get, and/or release events operating on a single object, representing the actual lock/unlock behavior for an object. The data in this table provides the timestamps for these events, the hold and wait times, as well as the name of the method, thread, and job locking the object.

Double-clicking on a procedure row of this table will generate and display the Object Lock Frame for the selected lock.

5.2.11. Object Notify/Wait Pane

This pane contains a table of wait and notify events that have occurred for a specific object. One of these events will occur whenever the java.lang.Thread.notify(), notifyAll(), or wait() methods have been invoked. The table contains information about the notify or notifyAll, the wait time, the method that invoked the notify or wait event, and links up the corresponding wait and notify events.

5.2.12. Garbage Collector Sweep Pane

This table contains information about the garbage collector sweep events that have occurred in the collection. This data is similar to what is output to the screen when using the *verbose option on the java command.

This table will be generated only if there are garbage collection sweep events in the trace collection being processed.

5.2.13. WebSphere Summary Pane

The WebSphere Summary Pane is a table containing summary information for all the WebSphere events that have occurred within a specific collection level.

This pane can only be generated when there are WebSphere events in the collection.

Double-clicking on a row of this table will generate and display the WebSphere Event Summary Frame associated with the selected WebSphere event.

5.2.14. WebSphere Event Pane

The WebSphere Event Pane is a table containing information on the individual WebSphere events of the same type as the current WebSphere Event Frame. It contains detailed event information based on the data in each WebSphere event.

This pane can only be generated when there are WebSphere events in the collection.

Double-clicking on a row of this table will generate and display the WebSphere Event Frame associated with the selected individual WebSphere event.

5.2.15. Java Transform/JIT Compile Pane

The Java Transform/JIT compile pane displays information about the JIT compile events that have occurred in the collection. This provides information about the methods that were compiled by the JIT, when the compile started, and the length of the compile. This pane is produced in Full Trace

Processing mode when there are Java transform events in the collection, or if the corresponding view was selected in Quick View mode.

The following panes are available in **Quick View Processing** mode:

5.2.16 Object Create Summary Pane

This pane is produced when there are Java object create events in the collection and the user has selected this view, instead of Full Trace Processing. This table provides information about the number and size of objects created for a particular class, along with the call stack information from the point of the create. In V5R3, this can be 5 or 16 levels of call stack information.

5.2.17 Object Lock Summary Pane

This pane is produced when there are Java object lock and unlock events in the collection, and the user has selected this view in Quick View mode instead of Full Trace Processing. This table provides information about the number of locks and unlocks for a particular class, along with the call stack at the point of the lock and unlock, which can be 5 or 16 levels in V5R3. Wait times are not provided in this view; Full Trace Processing is needed to get that view.

5.2.18 Java Exception Summary Pane

This pane is produced when there are Java exception events in the collection, and the user has selected this view in Quick View mode instead of Full Trace Processing. This table provides information about the number of java exceptions in the collection for a given exception class, along with the call stack at the point of the exception throw and catch.

Table of Contents | Previous | Next

6.5 Glossary of terms

Pane:
A pane is a single set of data, and can be displayed by selecting the tab associated with it. It can be either a simple informational pane, table, or tree table. Tables and tree tables contain rows and columns of data about a subset of data from the collection. The data is usually associated with a specific collection level and represents a given subset, such as object locks, or callers of a procedure.

Frame:
A frame is a window that is displayed corresponding to a given entity in the collection. Each frame has a collection level associated with it, such as the full collection, a job, or thread in the collection, or a specific item from the collection, such as an object or a procedure. A frame consists of one or more tabs, and clicking on a tab will display the pane associated with that tab.

Full Trace Processing:

See the section on Full Trace Processing.

Quick View Processing:

See the section on Quick View Processing.

Dynamic Data Pane:
When a table contains more than the default maximum number of rows, the table becomes a dynamic data pane. A dynamic data pane allows the user to view a subset of the rows in a table, based on the ordering of the currently selected sort column. In some cases the table may contain 10,000 or more rows, and the dynamic data pane prevents an excessive number of rows from being brought from server to client. Using dynamic data panes correctly, you can visualize the same information much more quickly, and in some cases, much more effectively.

A dynamic data pane can be identified by the slider bar, "chunk" buttons, and update button at the top of the table. The slider bar identifies the maximum number of rows available for the table, as well as the current number of rows being displayed for this table. Each table has a default sort column, and the initially dispalyed pane shows those columns based on that column. Alternatively, you can click "Next Chunk" or "Previous Chunk" to move the window forward or backwards one full chunk.

Adjusting the number of rows:

- Moving the slider bar: The number of rows can be adjusted by moving the start or end of the slider bar and clicking on the update button. This will bring over the selected number of rows, with the ordering based on the current sort column.
- Clicking on the "Next chunk" or "Previous chunk" button: This will display the next set of rows in the table. The ordering is based on the current sort column, and the number of rows remains the same as the current number, unless the slider bar is adjusted.
- Warning: Showing too many rows can result in out of memory errors on the client.

Changing the sort column:

- The row ordering is based on the current sort column. Clicking on a column header will cause the rows to be resorted based on the new column. Be aware, that sorting will be done only on the currently displayed rows. If you want to resort all rows in the table, and display the top or bottom set of rows based on that column, you must click the Update button.

By default, no data sorting takes place, but if you click on any column header, the data will be re-sorted by that column. In order to get the "first x rows" as sorted by that column, always be sure to click "update" after choosing a different column to sort by. If you don't, only the small chunk of data currently displayed will be sorted. For example, if you had sorted by timestamp, and asked to see the first 100 rows, then clicked on the column header reading "procedure name", it would all of a sudden appear that you were seeing the first 100 procedures, sorted by procedure name. This, however, is **NOT** the case. What you are seeing is the first 100 rows to exist in this table (according to timestamp), then sorted based on their procedure name. If you wanted to actually see the first 100 rows according to procedure name, click on "Update".

Copy or Export Data
Each pane contains data which can be copied to a file or spreadsheet application. The data that is currently shown by the table or tree is copied in a format so that most spreadsheet applications can recognize the column separator. This can be done by the Export button, or by the Copy to clipboard button, both found at the top toolbar for each frame.

Table
A table is a set of data found within a pane which consists of rows and columns of data.

For each table, there is a default number of rows that can be viewed in one window. When the number of rows in the table exceeds that number, the table becomes a dynamic data pane, and only the default number is shown.

Tree table

A tree is a set of rows and columns of data, but differs from a table in that the rows have a parent/child relationship. Children for a row can be displayed by clicking on the "twistee" icon found at the left side of the row.

If a row in the tree has more than the maximum number of children, those children will be grouped into smaller sets of data, and identified with a label "Children 1 - 255". This prevents the display of an excessive number of rows in the tree, to prevent memory errors.

Column attributes and functions
Each contains a pane of data which is displayed as a set of columns. Each column has a column name, and all values in a column must be of the same type.

Columns have the following set of available functions:

- Sorting
- Showing/Hiding
- Moving
- Converting to percent

Row attributes and functions
All panes consists of a set of rows of data. In most cases, the first column contains the identifying name, with subsequent values containing data from the collection associated with that name.

In many cases, a row of a table or tree can be associated with another subset of data within the collection for a specific collection level. Double-clicking on a row will cause a new frame to be displayed, containing more detailed information about the subset of data corresponding to that row. For example, when viewing the Job/Thread table pane, clicking on a job row in that table will cause the corresponding JobFrame to be displayed. When viewing the CallTrace pane for a particular thread in the collection, clicking on a procedure row will cause the corresponding ProcedureFrame to be generated and displayed.

For some rows, right-clicking on the row will bring up a popup that allows you to do select an associated frame.

Sorting columns
In a table, most columns are sortable, and can be sorted in either ascending or descending order by clicking on the column header. If the current table is a dynamic data pane, only the currently viewed rows will be sorted. The Update button is used to indicate tha the entire table must be resorted before displaying the top/bottom number of rows based on the most recently selected sort column.

Data in trees can also be sorted. Sorting for trees is done at the parent level, that is, all siblings for a parent node are sorted based on the sort column.

Showing/hiding columns
When a table or tree is initially displayed, not all available columns are usually displayed. Some columns are shown by default, with others hidden, even though there are values available for those columns. Clicking on the show/hide button at the top toolbar will bring up a window that allows you to add or remove columns from the current table.

It is also possible to hide a column by right-clicking on a column.

Converting column values to percent
Many columns can be converted from its initial value to a percent value by right-clicking on that column. If a column is convertible in this way, it is identified by a blue font and with a bar representing its percent value.

Collection level
Each data subset usually has a collection level associated with it. The collection level refers to the scope of the data, and can be one of the following:

- **the entire collection:** totals are computed over the entire collection
- **a job:** totals are computed only within a specific job
- **a task, or thread:** totals include only those values from within a specific thread or task
- **a procedure summary:** totals are computed based on all calls of a certain procedure, scoped to the entire collection, job, or thread
- **a procedure call:** totals include values only from a specific individual procedure or method call

Table of Contents | Previous | Next

6.6 Common problems and possible solutions

Where to look for information when something doesn't work.

1. No matter what mode you are running in, you can look at the command prompt window that gets generated on your client when PTDV is started. This might provide some exception information or other error information.
2. If you are running in Thin Client or Three Tier, you can go to the server where the PTDV server job is running, and do WRKSPLF on the userid used to process data. There should be QPRINT file corresponding to the PTDV server job. Look in this file for any exceptions.

Some common problems:

- ***Unable to browse libraries on the server***
  If the browsing function does not work at all, it is usually due to a problem with the connection for the iSeries Toolbox for Java. Be sure that the iSeries server has this product installed. If it is installed, you might need to start the host servers by issuing the command STRHOSTSVR SERVER(*ALL)
- ***Unable to connect in Thin Client or Three Tier***
  If you are able to browse libraries on the server, but unable to make the client/server connection there are a variety of problems that could cause this.. Go to the server and find the QPRINT file for your server job, if it exists.
    - If there is no QPRINT file or if it just contains a single line saying that the Java command ended with exit code 1, then for some reason the PTDVSERVER job was unable to start. If you are running on a new development driver, it is possible that something is wrong with Java or the Java environment on that driver. If you just installed a new version of PTDV, there could be a problem with the new version or its installation.
    - If there is a QPRINT file which contains error messages or other Java exceptions, then you need to contact the PTDV support person. If the messages are SQL related, and this is a new driver, then it is possible that the error is to due to SQL problems. If possible try to run on another system to see if the problem is due to PTDV or to a system/driver problem.
    - If there is a QPRINT file containing RMI log messages for the wrong IP address, or that the host name is unknown, then there is a problem in the

way that the IP addresses are set up on the server.. If you go to your server, and ping the name that it is known as on the network, but it is unable to ping itself, then PTDV will not be able to run in Thin Client or Three Tier mode.

- ***Unable to export the current table***
  The button on the menu bar labelled "Export table" will not work correctly if you are running Java 1.2 on your client. You need to run with Java 1.3 or later releases in order for this to work. If you don't want to move to Java 1.3, you can copy your data by using the Copy to Clipboard button on the menu bar and then pasting the values into a spreadsheet tool.
- ***Data appears outside the borders of the tables***
  This is another problem that occurs when using Java 1.2 on the client. Moving to Java 1.3 will correct this problem.
- ***PTDV will not run in Thin Client or Three Tier mode due to incompatible client and server jars.***
  When a new version of PTDV has been downloaded onto your client or server, the tool should be able to automatically detect the situation and update the jar that needs updating. However in some rare cases, this update might not work correctly. If that happens, and PTDV attempts to run, you could get an incompatible class error displayed in the command prompt window on your client. To correct this problem, go to the server and delete the existing ptdv.jar file, which can be found at /QIBM/ProdData/iDoctor/PTDV/ptdv.jar, and try running again. You should be prompted with a dialog box asking if you want to update your server. Answer 'OK' and continue running.
- ***PTDV is unable to process a collection that was collected and saved as a *MTGCOL object and expanded into database files using CRTPEXDTA.***
  When PEX collection data is saved as a *MTGCOL object, it must be expanded into database files using the CRTPEXDTA command on the same release where the data was collected. PTDV is unable to process a collection which was saved as a *MTGCOL object on one release and then restored using CRTPEXDTA on another release.

Table of Contents Previous Next

Chapter 8 Heap Analyzer

This chapter provides an overview of the interfaces within the iDoctor for iSeries - Heap Analyzer component.

Table of Contents | Previous | Next

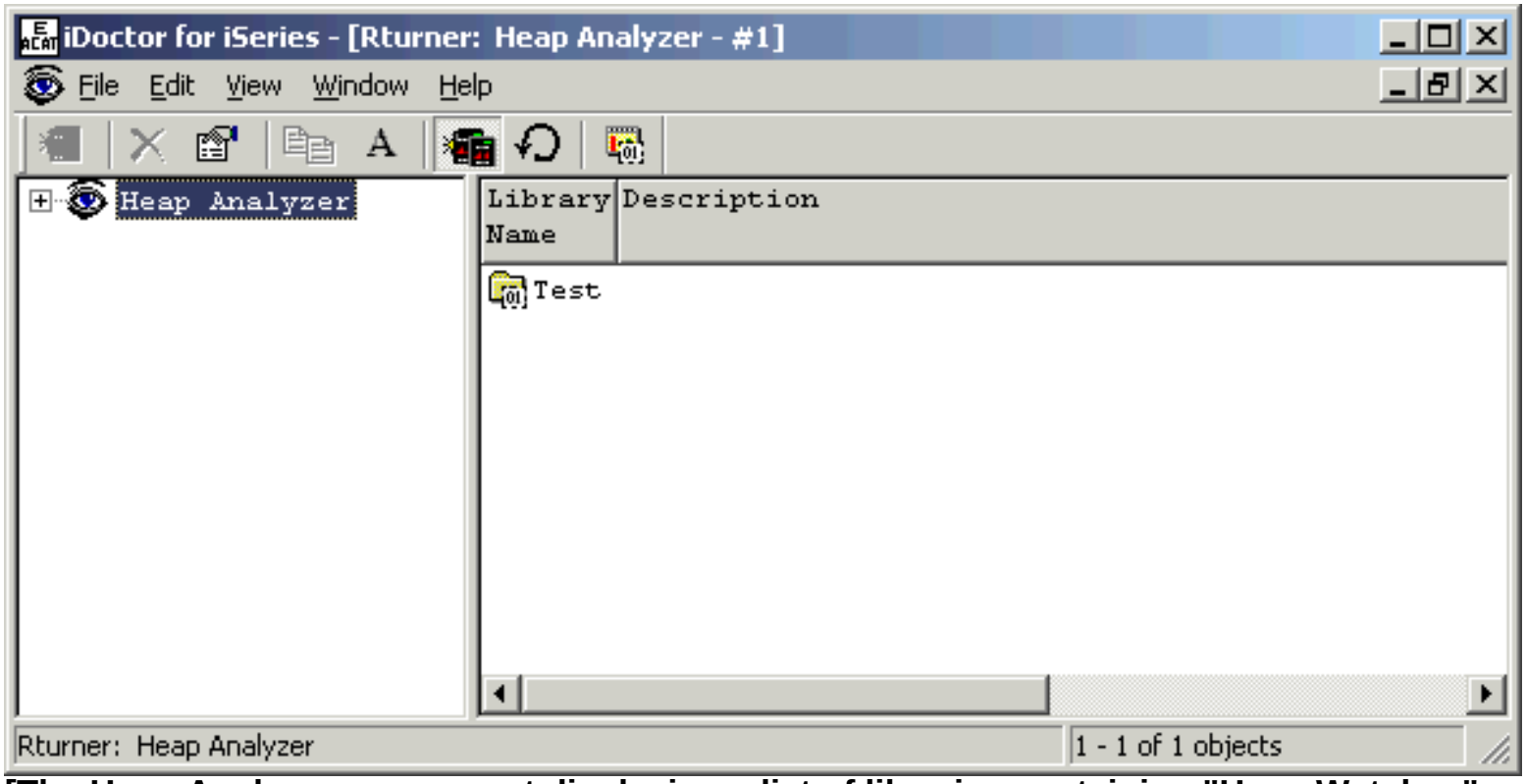
8.1 Heap Analyzer Basics

The Heap Analyzer component allows a Java™ application developer on the iSeries to diagnose Java performance issues such as heap growth issues in the Java Virtual Machine.

Starting Heap Analyzer

Heap Analyzer is a component of the iDoctor for iSeries suite of tools. iDoctor for iSeries can be started using the Start menu: Start->Programs->iDoctor for iSeries. Once the iDoctor for iSeries application appears, the Heap Analyzer component is started from the Connection List View by right-clicking on a system name and choosing the Heap Analyzer menu.

After starting Heap Analyzer you will see a window similar to the following:



[The Heap Analyzer component displaying a list of libraries containing "Heap Watches" on a system.]

The 'Heap Analyzer' folder contains a list of library folders, each representing a library on the iSeries that contains Job Watcher database files (Heap Watches). The list displays each library's name and description.

Heap Analyzer Objects

There are various types of objects within the tree/list views of Heap Analyzer in the following order: **Libraries**, **Heap Watches**, and **reports.** Each of these will covered in more detail in the next sections.

Heap Analyzer Menu Options

The following menu options are available by right-clicking on the 'Heap Analyzer' folder in the tree/list view above.

Menu Item	Description
Explore	Displays the contents of the Heap Analyzer folder (list of libraries on the system containing Heap Analyzer data) in the right pane of the tree/list window.
Start Heap Watch...	This menu will open the Start Heap Watch Wizard where you can define and run a Heap Watch.
Open New Data Viewer	Opens a new Data Viewer window. This window is used to display tables and graphs on the system. You can open Heap Analyzer reports into this window or you can also open any other type of physical file and view as a graph or table.
Properties	Use this menu to display Heap Analyzer version information.

Table of Contents | Previous | Next

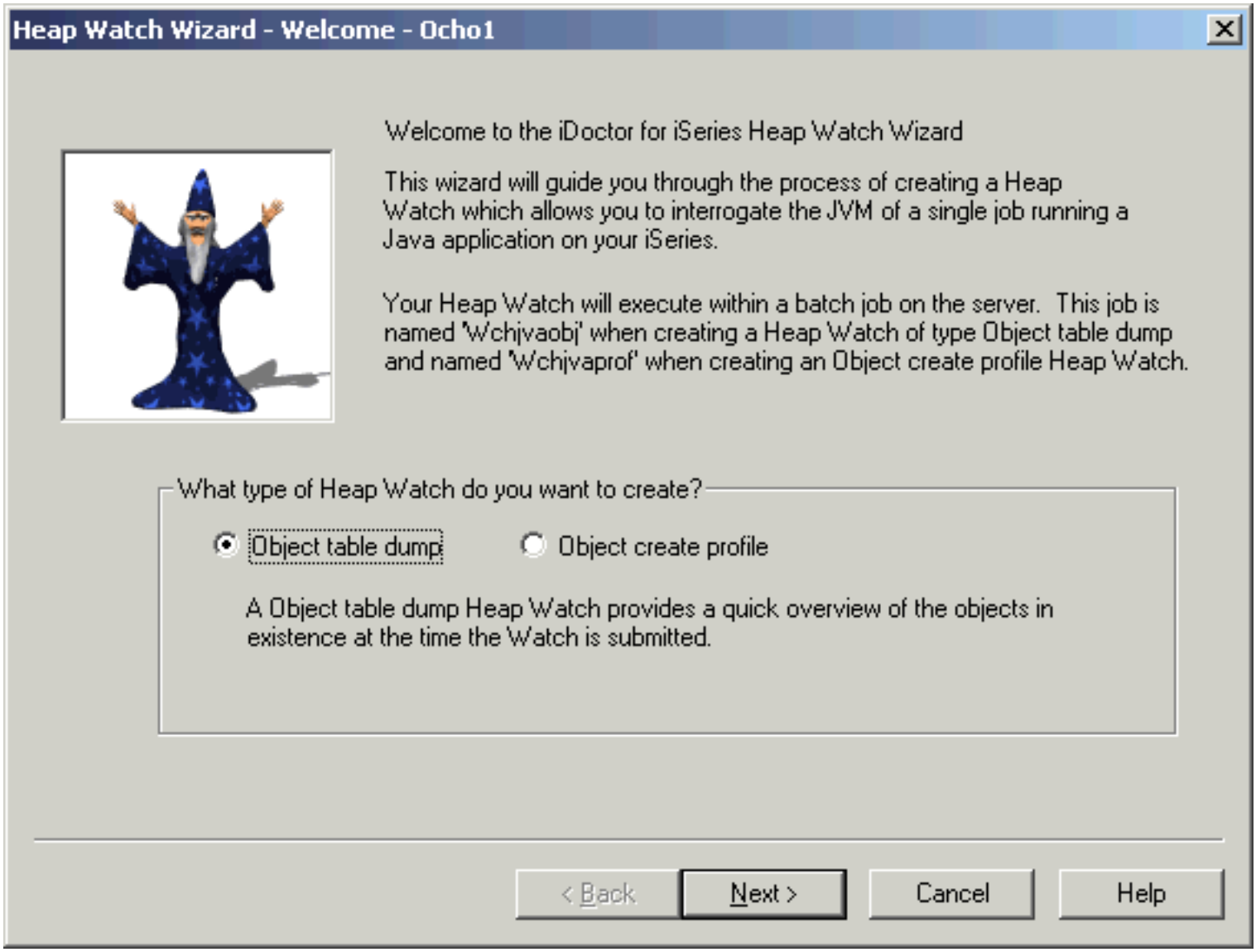
8.2 Start Heap Watch Wizard

Heap Analyzer provides the capability to collect detailed information about any currently running job on the system that is running Java. There are two modes of collection. The first is the called the 'object table snapshot' and the second type is the 'object create profile'. The 'object table snapshot' is a one shot look at the JVM for a particular job showing the breakdown of number of objects/(Java classes) of each type, and how much memory is being consumed by these objects.

The 'object create profile' mode lets you sample information from the JVM as fast as possible until a limit condition is reached. The limit may be provided as a time value (seconds), disk space (megabytes) or number of samples.

To create a new Heap Watch use the Heap Watch Wizard. The Wizard is accessible via the Start Heap Watch menu on the Heap Analyzer or library folder icons.

An example of the Wizard is shown below:



[The Start Heap Watch Wizard]

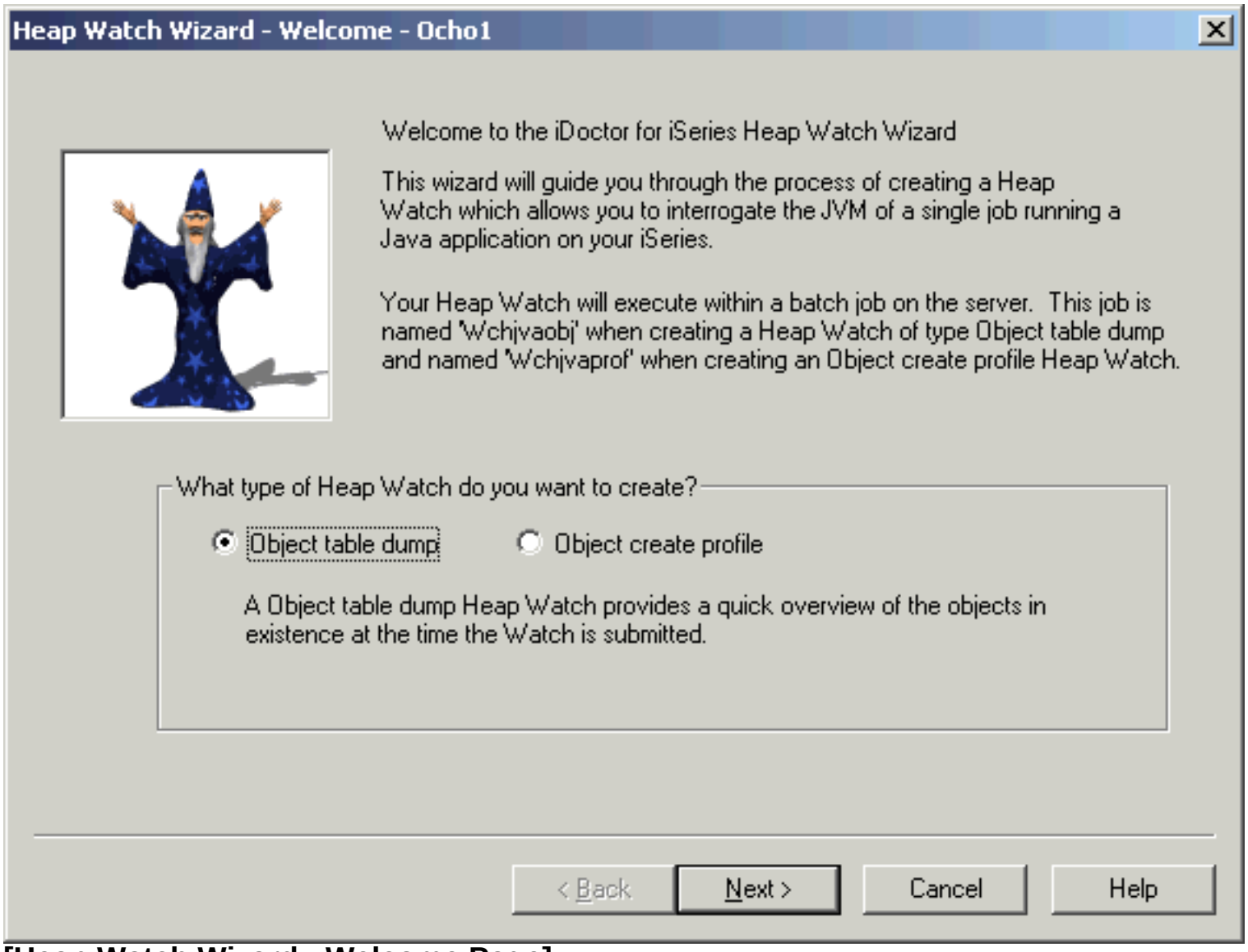
The Heap Watch Wizard guides you step by step through the process of creating the Heap Watch. Each page is covered in detailed within the next sections.

Table of Contents | Previous | Next

8.2.1 Welcome

The Welcome page in the Heap Watch Wizard introduces the user to the wizard and offers information about what the wizard will do. The page also explains the two types of Heap Watches and within what job name the Watches will run in.

On this page you must specify the type of Heap Watch to create.



[Heap Watch Wizard - Welcome Page]

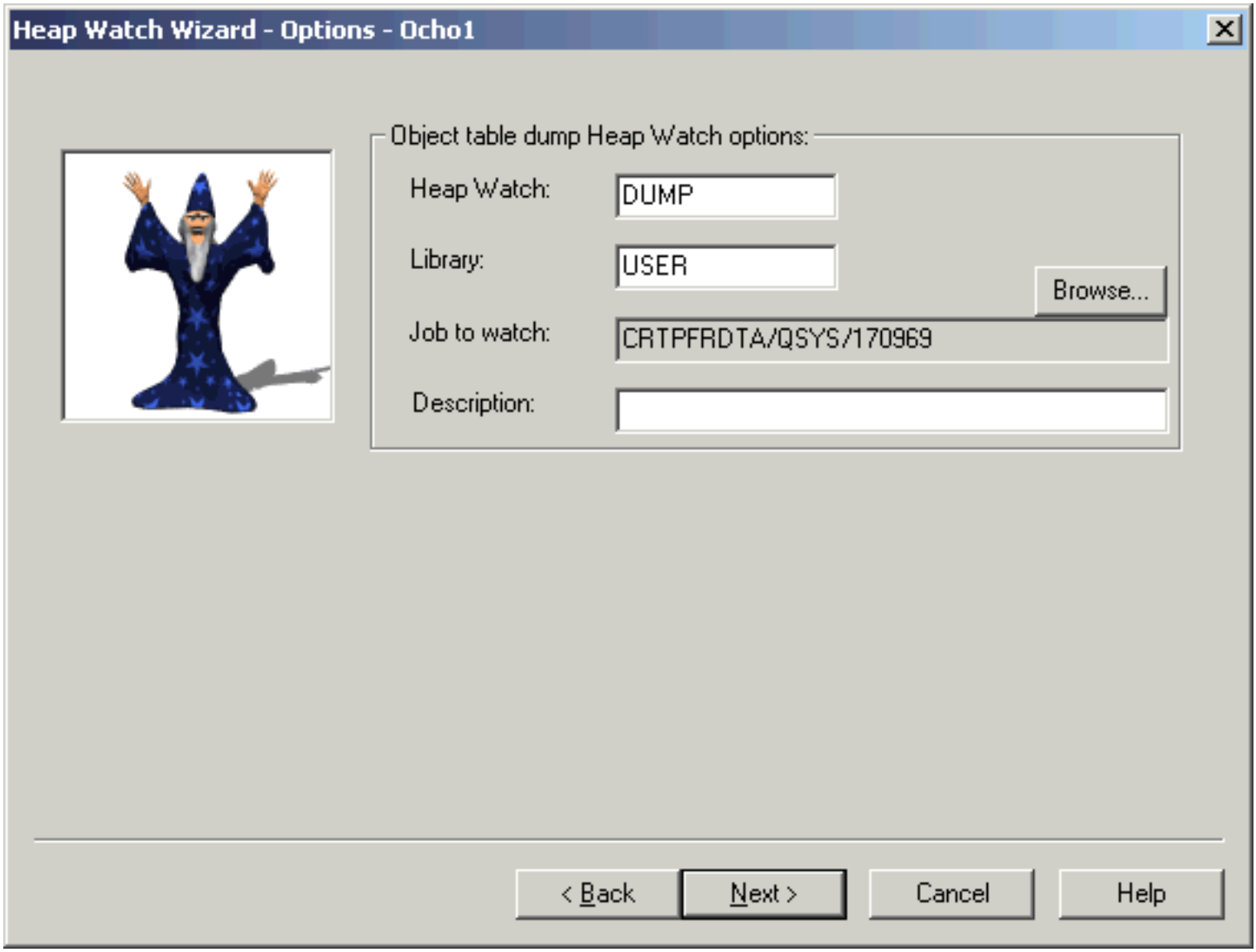
Table of Contents | Previous | Next

8.2.2 'Object table snapshot' Options page

Note: This page does not apply to 'object create profile' Heap Watches.

The following parameters are defined on this window:

Field	Description
Heap Watch	The up to 10 character member name that uniquely identifies the Heap Watch in the library. Files QPYRTJVMH1 and QPYRTJVMH2 will have contain this member after the Heap Watch is started.
Library	Name of the library to create the Heap Watch into.
Job to watch	Name of the job to watch. You can only watch one job/user/number per instance.
Description	50 byte description shown in the client when viewing a list of active/completed Heap Watches.



[Heap Watch Wizard - Object table snapshot Heap Watch options]

Table of Contents | Previous | Next

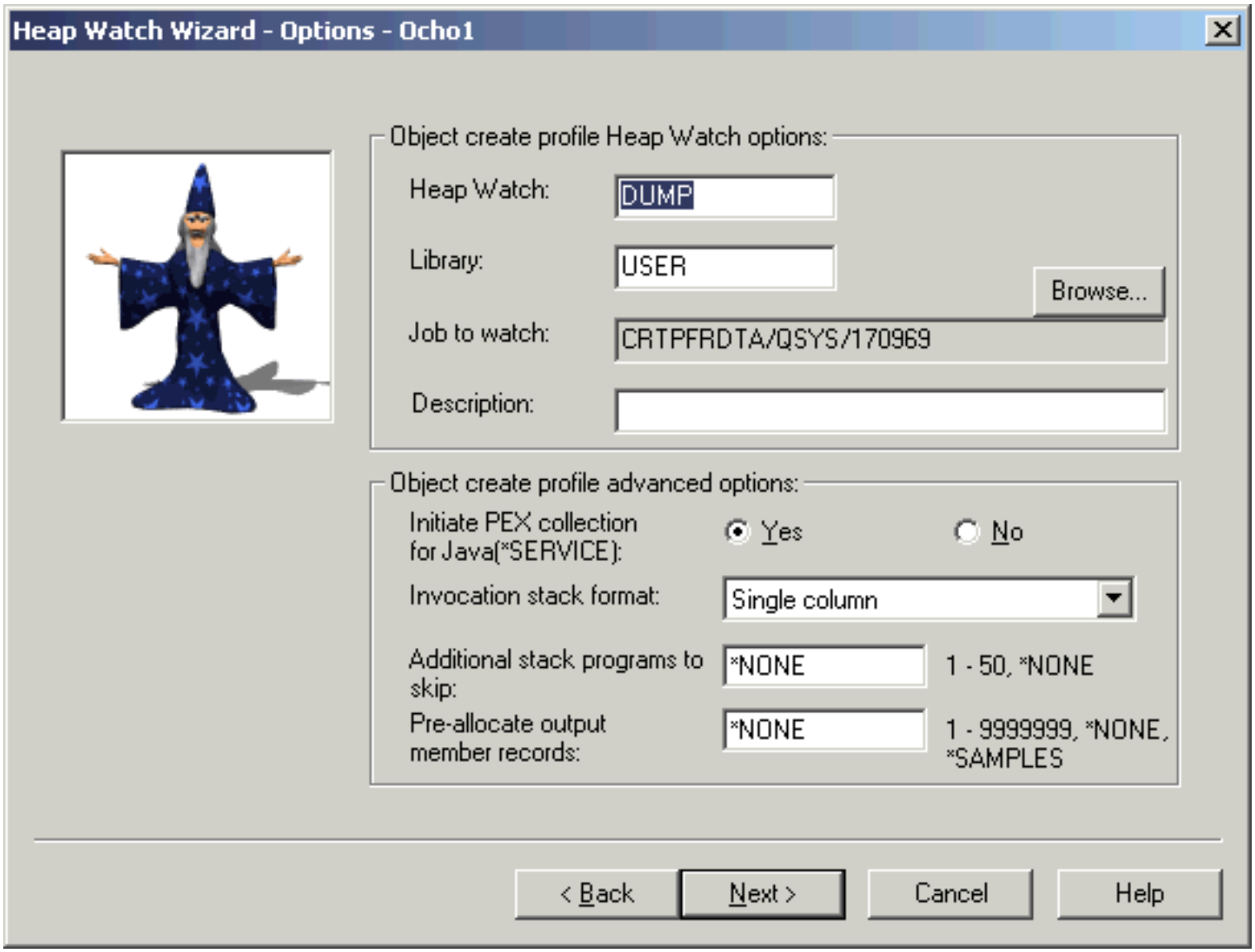
8.2.3 'Object create profile' Options page

Note: This page does not apply to 'object table snapshot' Heap Watches.

The following parameters are defined on this window:

Field	Description
Heap Watch	The up to 10 character member name that uniquely identifies the Heap Watch in the library. Files QPYRTJVMH1 and QPYRTJVMH2 will have contain this member after the Heap Watch is started.
Library	Name of the library to create the Heap Watch into.
Job to watch	Name of the job to watch. You can only watch one job/user/number per instance.
Description	50 byte description shown in the client when viewing a list of active/completed Heap Watches.
Initiate PEX collection for Java (*SERVICE)	Indicates whether a PEX collection should be started over the Job to watch during execution of the Heap Watch. The PEX collection will not collect data, but will turn a flag on in the job to make information retrieval for the profile possible. The PEX collection will be deleted automatically when the Heap Watch completes.
Invocation stack format	Determines if the call stack should be harvested into a single column in the database , or into multiple columns. Depending on this selection determines which file the stack information is written to. This will either be QPYRTJVMF2 (multiple columns) or QPYRTJVMF3 (single column).
Additional stack programs to skip	Only 5 stack entries are harvested due to space constraints. Because there may be many many entries on the call stack, you may want to use this option to throw away everything except each nth entry on the call stack. Note: You can only have one stack skip value setting used for any JVM instance at a time. This means that if two users are watching the JVM at the same time with different stack skip values, the actual value will likely toggle between the two because at the beginning of each interval the stack skip value is sent down to the JVM.

Pre-allocate output member records	Indicates how many records of data to preallocate before starting the Heap Watch. This can be used to remove a delay during sampling in allocating the space. However using this it is possible to allocate much more data than is actually needed.



[Heap Watch Wizard - Object create profile Heap Watch options]

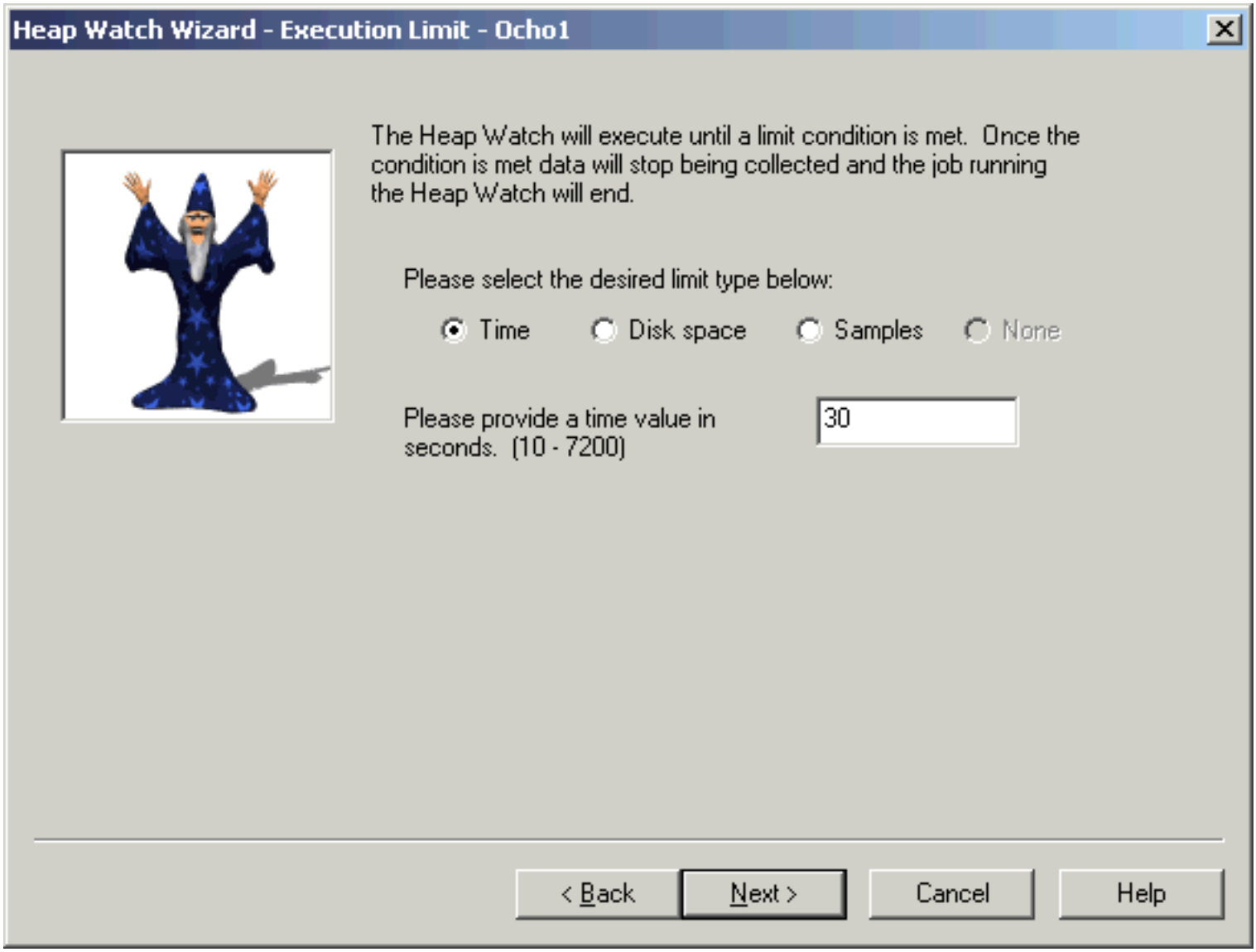
Table of Contents | Previous | Next

8.2.4 Execution limits page

This page lets you define the execution limit. This limit indicates when the Heap Watch will stop executing.

The following parameters are defined on this window:

Field	Description
Limit type	The type of limit that will end execution. The possible limit types is different depending on if you are creating a object table snapshot Heap Watch or a object create profile Heap Watch. A object table snapshot can have a time limit (seconds) or none. Because dumping the JVM can take a very long time depending on how many objects exist within it, you can use the time limit so the job will stop executing if it takes longer than n seconds. If the dump does not complete you will only get partial data though. If you don't want a time limit when running this type of Heap Watch select 'none' for the limit type. A object create profile can have a limit type of time, disk space (megabytes) or samples. When an object create profile runs it will create records as fast as possible, providing information about the objects created during each 'interval'. It's important to choose the desired limit type here because you may end up with many GBs of undesired data if not careful. Example: Setting a time limit of 60 seconds could easily generate 1 GB or more of data. It may be better to set a disk space limit or a limit to the number of samples if disk space is a concern.
Limit value	The value for the execution limit. Its meaning is totally dependent on the limit type. If limit type is 'time' then this value is in seconds. If limit type is 'disk space' then this value is in megabytes. If limit type is 'samples' then this value is the number of samples If limit type is 'none' then there is no limit value and this field is greyed out.

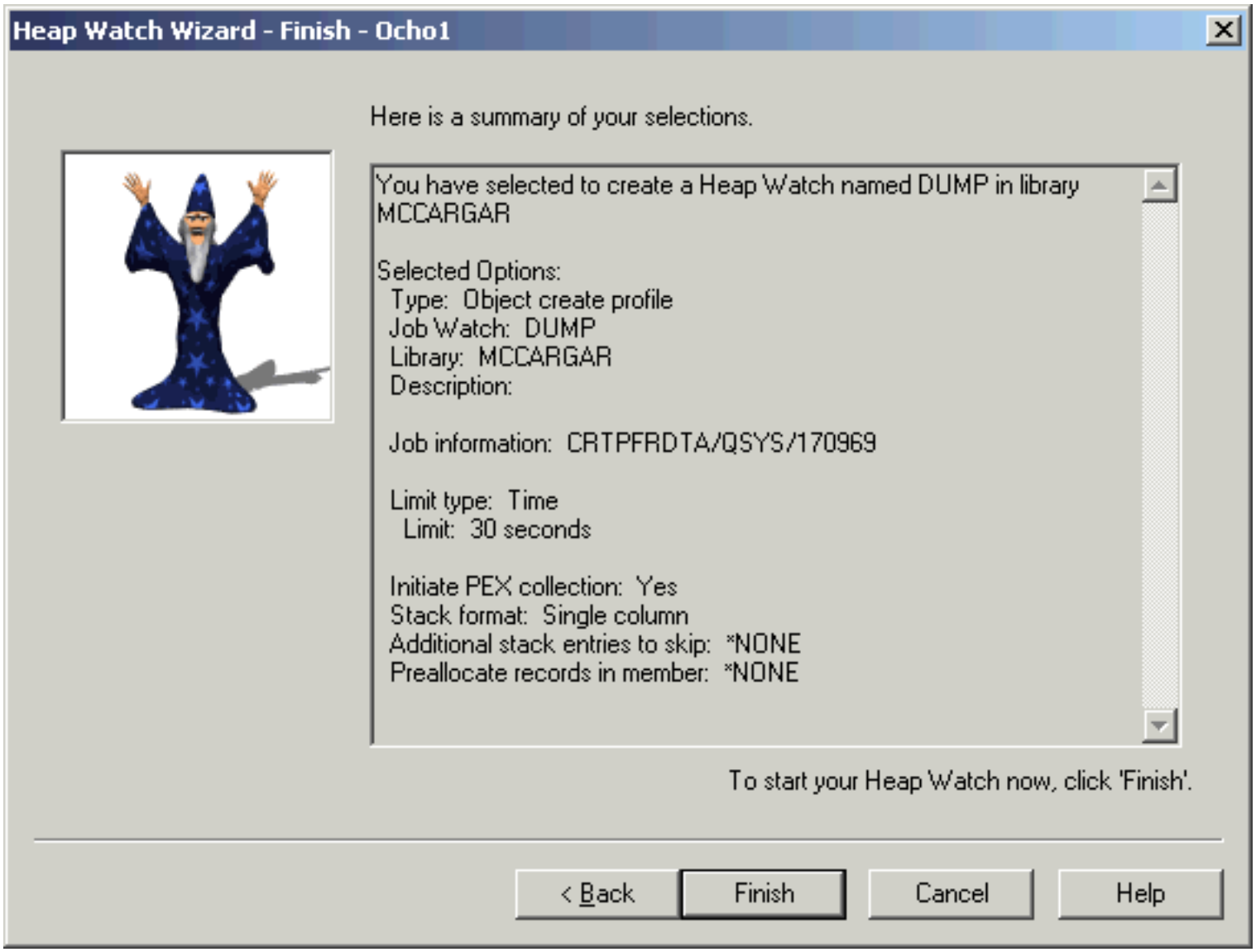


[Heap Watch Wizard - Execution limits page]

Table of Contents | Previous | Next

8.2.5 Summary page

This page simply summarizes your selections. Clicking the 'Finish' button will start the Heap Watch and close the wizard.



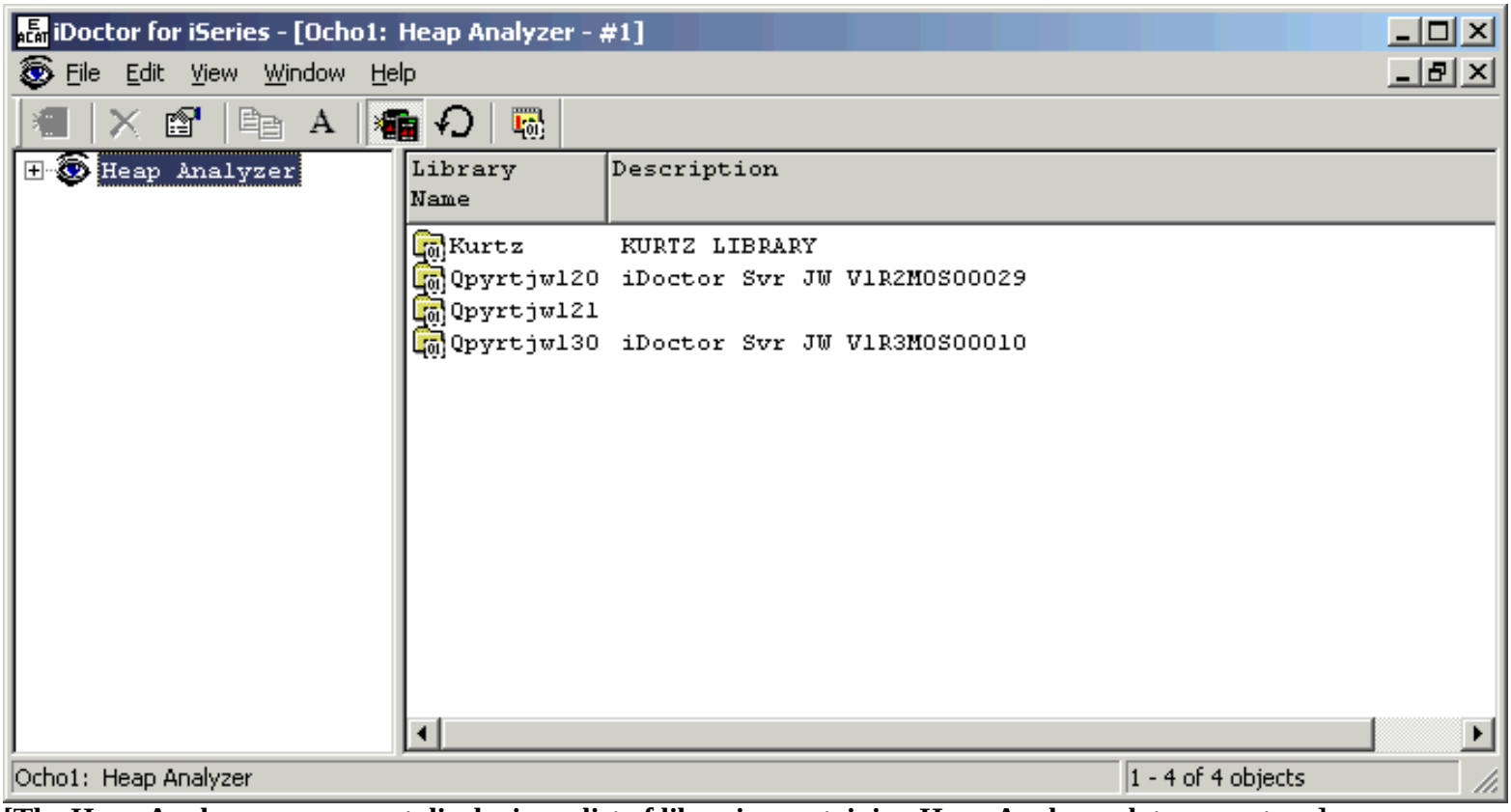
[Heap Watch Wizard - Summary page]

Table of Contents | Previous | Next

8.3 Libraries

The 'Heap Analyzer' folder contains a list of library folders, each representing a library on the iSeries that contains Heap Analyzer database files. The list displays each library's name and description.

By clicking on a library in the tree you will see its contents (a list of Heap Watches).



[The Heap Analyzer component displaying a list of libraries containing Heap Analyzer data on system.]

Library Menu Options

The following menu options are available by right-clicking on a library in the tree/list view above.

Menu Item	Description
Explore	Displays the contents of the library (list of Heap Watches within the library) in the right pane of the tree/list window.
Select fields...	Brings up a window that lets you modify what fields are shown when displaying the contents of library. The contents of a library are Heap Watch objects so this menu lets you hide/display/reorder fields that are relevant to a Heap Watch.

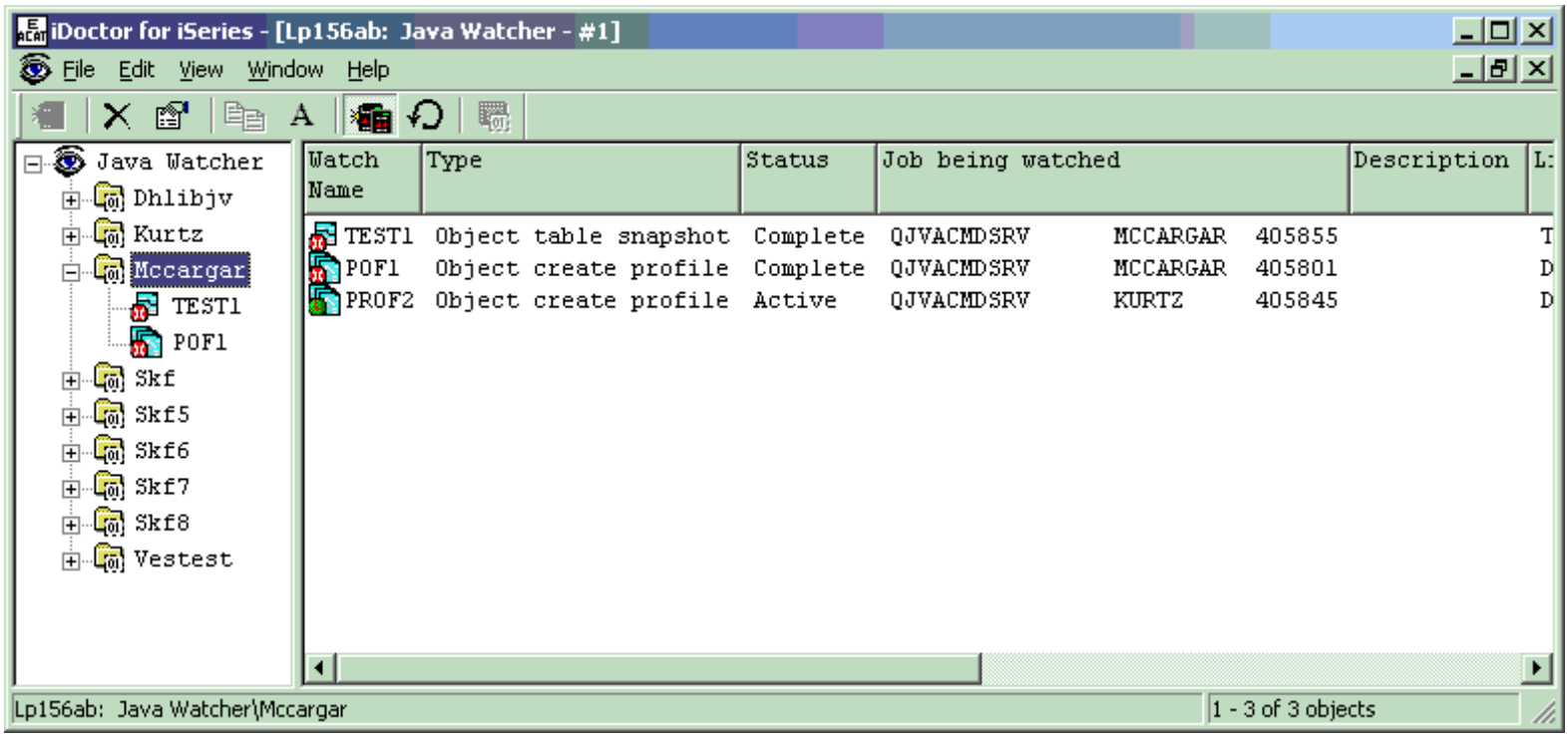
Start Heap Watch...	This menu will open the Heap Watch Wizard so you can create a Heap Watch in the selected library.
Copy...	Allows you to copy the library's contents into a new library or into an existing one.
Save...	This option lets you save the library's contents into a save file.
Clear	This option clears a library (deletes all objects in the library).
Delete	Deletes the library.
Rename	Renames the library.
Properties	Use this menu to display the library property pages. Basic information similar to that provided by DSPOBJD is available through these property pages.

Table of Contents | Previous | Next

8.4 Heap Watches

Moving down the tree within each Library folder are zero or more Heap Watches that have been created (or are being created) within the current library. The icons showing a green clock indicate active Heap Watches and the icons with red stop signs indicate Heap Watches which have completed.

There are two types of Heap Watches: 'object table snapshot' and 'object create profile'. The type of watch will be shown in the 2nd column in the list by default (see example below).



[Heap Analyzer displaying the list of Heap Watches within library 'Mccargar']

Heap Watch Status

Each Watch has a status field indicating whether or not it is currently running. You can also tell the status by the color of the icon: Green = active, Red = not active.

Heap Watch Fields

The list of Heap Watches displays the watch name, watch type, status, description as well as several additional fields.

Each Heap Watch in the list has a set of fields available which can be optionally reordered and displayed. To change the current field selections for the Heap Watch list, use the Select fields... menu from the library folder. A listing of the available fields and a short description is provided in the table below:

Field	Description
Watch name	Name of the Heap Watch. This name matches the member name used in the output files named QPYRTJVM* that exist on the system.
Type	The type of Heap Watch. Possible values are 'object table snapshot' and 'object create profile'.

Status	The status field indicates the status of the job on the system running the Heap Watch (if active) or if not active the status indicates whether or not the watch can be analyzed and/or has data available.
Job being watched	The name of the job this Heap Watch is watching.
Description	50 byte description of the Heap Watch specified at creation time.
Limit type	Execution limit type. Indicates when the Watch will stop executing. Possible values are: time, disk space, or samples.
Limit value	The value of the execution limit.
Job running the watch	The name of the job running the Heap Watch. This will typically be named WCHJVAOBJ for an object table snapshot Heap Watch and WCHJVAPROF for an object create profile Heap Watch. The job running the watch will be an interactive job if the user created the watch using the green screen command WCHJVA.
Start time	The time the Heap Watch started.
End time	The time the Heap Watch ended (stopped collecting data).
Stack format	Determines if the call stack should be harvested into a single column in the database , or into multiple columns. Depending on this selection determines which file the stack information is written to. This will either be QPYRTJVMF2 (multiple columns) or QPYRTJVMF3 (single column).
Additoinal stack entries to skip	Indicates how many entries are skipped if any per every saved call stack entry. Example: A value of 3 means, skip 3 entries, save/write 1 entry, skip 3 entries, save/write 1 entry, etc.

Heap Watch Menu Options

The table below outlines the different types of operations that may be performed on a Heap Watch.

Menu Item	**Description**
Explore	Displays the contents of the Heap Watch (its reports) in the right pane of the tree/list window.
Record Quick View	Displays the fields for a single Heap Watch in a list view vertically for easier viewing. Not available from the tree side, only the list side.
Compare...	Runs a comparison analysis over two object table snapshots Heap Watches.
Delete...	Delete a Heap Watch. Select multiple Heap Watches in order to delete more than one watch at a time.

Table of Contents | Previous | Next

8.4.1 Object table snapshot comparison

Heap Analyzer features an analysis over two object table snapshot Heap Watches. This analysis identifies heap differences between snapshot A and snapshot B. **Note:** This documentation refers to snapshot/watch/collection interchangeably.

The analysis outputs consist three reports:

1. Shows the object count and object/heap size differences between snapshot A and snapshot B.
2. Displays information about snapshot A.
3. Displays information about snapshot B.

By running the analysis from the client the command QPYRTJW/HCOMPARE is executed under the covers. This command creates the output files J_DIFF, J_OBJ1, J_OBJ2 in library QTEMP within the current job (in this case the QZDASOINIT job that the client is using). The member name is randomly generated based on the current timestamp.

After the command finishes executing the reports are shown in the Data Viewer. Once opened the reports may be queried but any queries created cannot be saved. You cannot copy the data or save it via the client. If you wish to save these comparisons, you must use the HCOMPARE command and copy the files from QTEMP within a green screen session.

To run this analysis use the 'Compare...' menu found by right-clicking on a Heap Watch of type 'object table snapshot.'