MainView *for CICS* PERFORMANCE REPORTER User Guide

Supporting

Version 6.7 of MainView *for CICS*

December 2015
Contacting BMC Software

Several methods are available for contacting BMC Software.

You can access the BMC Software website at [http://www.bmc.com](http://www.bmc.com). From this website, you can obtain information about the company, its products, corporate offices, special events, and career opportunities.

United States and Canada

<table>
<thead>
<tr>
<th>Address</th>
<th>Telephone</th>
<th>Fax</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC SOFTWARE INC</td>
<td>1 713 918 8800</td>
<td>1 713 918 8000</td>
</tr>
<tr>
<td>2101 CITYWEST BLVD</td>
<td>or</td>
<td>or</td>
</tr>
<tr>
<td>HOUSTON TX 77042-2827 USA</td>
<td>1 800 841 2031</td>
<td>1 800 841 2031</td>
</tr>
</tbody>
</table>

Outside United States and Canada

<table>
<thead>
<tr>
<th>Telephone</th>
<th>Fax</th>
</tr>
</thead>
<tbody>
<tr>
<td>+01 713 918 8800</td>
<td>+01 713 918 8000</td>
</tr>
</tbody>
</table>

© Copyright 1987-2015 BMC Software, Inc.

BMC, BMC Software, and the BMC Software logo are the exclusive properties of BMC Software, Inc., are registered with the U.S. Patent and Trademark Office, and may be registered or pending registration in other countries. All other BMC trademarks, service marks, and logos may be registered or pending registration in the U.S. or in other countries. All other trademarks or registered trademarks are the property of their respective owners.

Together is a trademark or registered trademark of Cisco Systems, Inc. and/or its affiliates in the United States and certain other countries.

CICS, DB2, IBM, MVS, VTAM, and WebSphere are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both.

The information included in this documentation is the proprietary and confidential information of BMC Software, Inc., its affiliates, or licensors. Your use of this information is subject to the terms and conditions of the applicable End User License agreement for the product and to the proprietary and restricted rights notices included in the product documentation.

Restricted rights legend

U.S. Government Restricted Rights to Computer Software. UNPUBLISHED—RIGHTS RESERVED UNDER THE COPYRIGHT LAWS OF THE UNITED STATES. Use, duplication, or disclosure of any data and computer software by the U.S. Government is subject to restrictions, as applicable, set forth in FAR Section 52.227-14, DFARS 252.227-7013, DFARS 252.227-7014, DFARS 252.227-7015, and DFARS 252.227-7025, as amended from time to time. Contractor/Manufacturer is BMC SOFTWARE INC, 2101 CITYWEST BLVD, HOUSTON TX 77042-2827, USA. Any contract notices should be sent to this address.
Customer support

Support website
You can obtain technical support from BMC 24 hours a day, 7 days a week at [http://www.bmc.com/support](http://www.bmc.com/support). From this website, you can:

- Read overviews about support services and programs that BMC offers
- Find the most current information about BMC products
- Search a database for problems similar to yours and possible solutions
- Order or download product documentation
- Download products and maintenance
- Report a problem or ask a question
- Subscribe to receive proactive e-mail alerts
- Find worldwide BMC support center locations and contact information, including e-mail addresses, fax numbers, and telephone numbers

Support by telephone or e-mail
In the United States and Canada, if you need technical support and do not have access to the web, call 1 800 537 1813 or send an e-mail message to [customer_support@bmc.com](mailto:customer_support@bmc.com). (In the subject line, enter **SupID:yourSupportContractID**, such as SupID:12345). Outside the United States and Canada, contact your local support center for assistance.

Before contacting BMC
Have the following information available so that Customer Support can begin working on your issue immediately:

- Product information
  - Product name
  - Product version (release number)
  - License number and password (trial or permanent)
- Operating system and environment information
  - Machine type
  - Operating system type, version, and service pack or other maintenance level such as PUT or PTF
  - System hardware configuration
  - Serial numbers
  - Related software (database, application, and communication) including type, version, and service pack or maintenance level
- Sequence of events leading to the problem
- Commands and options that you used
- Messages received (and the time and date that you received them)
  - Product error messages
  - Messages from the operating system
  - Messages from related software
License key and password information

If you have questions about your license key or password, contact Customer Support through one of the following methods:

- Send an e-mail message to customer_support@bmc.com. (In the Subject line, enter SupID:yourSupportContractID, such as SupID:12345.)

- In the United States and Canada, call 1 800 537 1813. Outside the United States and Canada, contact your local support center for assistance.

**Contents**

**About this book** ................................................................. 11
**Related publications** .......................................................... 11
**Conventions** ........................................................................ 12

**Chapter 1** Introduction to PERFORMANCE REPORTER 13
**Running report programs** .................................................. 15
**An overview of PERFORMANCE REPORTER reports** ............. 16
  - Predefined reports ............................................................. 16
  - Custom PRL reports .......................................................... 18

**Chapter 2** Collecting and archiving report data 19
**Collecting report data** ......................................................... 19
  - Transaction detail data (CMRDETL) .................................... 20
  - CTG Transaction detail data (CTGDETL) .............................. 23
  - CICS operational statistics (CMRSTATS) ............................. 23
**Archiving data (CMRPURG)** .................................................. 25
  - Archive-purge process ........................................................ 27
  - Purging summary records from the CMRSUMDN data set .......... 28
**Converting CMRDETL records into CICS 110 SMF format** .......... 29

**Chapter 3** Managing report data 33
**Creating statistical records** .................................................. 33
  - Control statements ............................................................. 34
  - SYSPRINT error messages .................................................. 35
  - Console error messages ...................................................... 36
**Reformatting compressed detail transaction records** ................. 36
  - CMRCMPWN batch decompression program ...................... 37
  - CMRCMPRN decompression program .................................. 39
**Creating reports from summarized data** ................................ 40
  - Summarization utility JCL ................................................... 41
  - Summarization utility control statements ............................. 42
  - Running the summarization utility ....................................... 53
**Creating PRL reports from summary data** ............................ 56
  - Notes on using the summarization utility ............................. 58

**Chapter 4** Resource Tracking and Analysis report 65
Performance Analysis by Netname report (CMRCB14N) .................................................. 114
  Sample report (Performance Analysis by Netname) ..................................................... 114
Performance Analysis for All Users report (CMRCB15N) ................................................... 115
  Sample report (Performance Analysis for All Users) .................................................. 115
File Analysis for All Files report (CMRCB21N) ................................................................. 116
  Sample report (File Analysis for All Files) .................................................................. 116
Rogues report (CMRCB23N) ................................................................................................. 116
  Sample report (COBOL rogues) .................................................................................. 117
Abended Transaction report (CMRCB25N) .......................................................................... 117
  Sample report (Abended Transaction) ............................................................................ 118

Chapter 8  SAS reports ........................................................................................................... 119
Sample SAS programs ......................................................................................................... 120
Control statements ............................................................................................................. 122
SAS report fields .................................................................................................................. 122
Performance Analysis for All Transactions report (CMRSA11N) ...................................... 123
  Sample report (Performance Analysis for All Transactions) ........................................ 123
Performance Analysis by Program report (CMRSA12N) ................................................... 124
  Sample report (Performance Analysis by Program) ....................................................... 124
Performance Analysis by Terminal report (CMRSA13N) .................................................. 125
  Sample report (Performance Analysis by Terminal) ...................................................... 125
Performance Analysis by Node Name report (CMRSA14N) ............................................. 126
  Sample report (Performance Analysis by VTAM Symbolic Node Name) .................. 126
Performance Analysis by User ID report (CMRSA15N) .................................................... 126
  Sample report (Performance Analysis by User ID report) ........................................... 127
File Analysis for All Files report (CMRSA21N) ................................................................. 127
  Sample report (File Analysis for All Files) .................................................................. 127
Rogues report (CMRSA23N) ................................................................................................. 128
  Sample report (SAS rogues) ....................................................................................... 129
Abended Transaction Roster report (CMRSA25N) ............................................................. 129
  Sample report (Abended Transaction Roster) ............................................................... 130
Operational statistics report (CMRSA6F) .......................................................................... 130
  Sample report (Operational statistics) ....................................................................... 131

Chapter 9  User-defined reports (CMRPRLN) ....................................................................... 133
Performance Reporting Language PRL ............................................................................... 134
  Samples ....................................................................................................................... 134
Expressions and operators ................................................................................................. 136
Substrings .......................................................................................................................... 137
Examples ............................................................................................................................ 137
Symbolic variables ............................................................................................................. 138
Chapter 10 PRL control statement verbs 149

Predefined variables .............................................................. 138
Generating PRL reports .......................................................... 139
Job control statements ......................................................... 140
Basic PRL verbs used to create a report ............................... 142
PRL processing ................................................................. 143
Logic flow .............................................................. 143
Record processing .............................................................. 145
Reporting selected data ....................................................... 145
Sorting and reporting data .................................................... 146
Example Reporting average terminal response time ............. 146

Chapter 11 PRL programming techniques 179

Routing report output ........................................................... 179
Defining output with JCL and PRL SET REPORTID ............... 180
Defining output with PRL SET CLASS and SET FORM .......... 181
Formatting report output ....................................................... 181
Predefined data dictionary descriptions ............................... 181
User-defined data .............................................................. 182
About this book

This book contains detailed product information and is intended for system administrators and database administrators (DBAs).

Like most BMC documentation, this book is available in printed and online formats. To request printed books or to view online books and notices (such as release notes and technical bulletins), see the support website at http://www.bmc.com/support.

**Note**

Online books are formatted as PDF or HTML files. To view, print, or copy PDF books, use the free Adobe Reader from Adobe Systems. If your product installation does not install the reader, you can obtain the reader at http://www.adobe.com.

The software also offers online Help. To access Help, press **F1** within any product or click the **Help** button in graphical user interfaces (GUIs).

Related publications

From the BMC Support Central website, you can use the following methods to access related publications that support your product or solution:

- Link to the BMC Documentation Center (https://webapps.bmc.com/infocenter/index.jsp) to browse documentation sets.

- View BMC Quick Course demos (short overviews of selected product concepts, tasks, or features), which are included in the BMC Documentation Center.

- Read individual product documents (books and notices) within the “A – Z Supported Product List.”

You can order hardcopy documentation from your BMC sales representative or from the support site. You can also subscribe to proactive alerts to receive e-mail messages when notices are issued.
Tip
You can access the BMC Support Central site at http://www.bmc.com/support.

Conventions

This document uses the following special conventions:

■ All syntax, operating system terms, and literal examples are presented in this typeface.

■ Variable text in path names, system messages, or syntax is displayed in italic text:
  testsys/instance/fileName

■ This document uses a symbol to show menu sequences. For example, Actions =>
  Create Test instructs you to choose the Create Test command from the Actions
  menu.
PERFORMANCE REPORTER is the MainView for CICS batch report facility. This chapter gives a short introduction to PERFORMANCE REPORTER and the various types of reports you can create.

Figure 1 on page 14 shows the process used to create reports with PERFORMANCE REPORTER and the various types of reports you can create with the sample programs provided by PERFORMANCE REPORTER.
CICS transaction and statistical data are gathered by data collectors and stored on specialized VSAM data sets. Frequently, long-term reports are created from data that has been archived to tape.

Figure 1: Report creation process

Reports are created by running standard JCL batch programs. Samples of these batch programs are located in your site’s BBSAMP data set. As a user, you edit copies of these samples and modify them to your site’s conditions.

Normally, control statements are added to a report job. Some statements select the data from the VSAM data sets that you want to appear in a report. Other control statements specify how a report should be formatted.

The following parts of this book give examples from each report type and the control statements that are needed to produce specific reports.
PERFORMANCE REPORTER provides the capability to create predefined or custom reports.

A predefined report has a fixed format that shows values that apply to a specific range of CICS performance parameters. Your BBSAMP data set contains sample jobs to create predefined CICS, COBOL, and SAS statistical reports.

In most cases, you merely specify the input data set and several control statements in your JCL to produce predefined reports. The control statements select the input data and specify whether the calculations produce a summary or detail report.

Other times, you may want to write programs to create custom reports. You use PERFORMANCE REPORTER’s English-like Performance Reporting Language (PRL) to create custom reports. The following shows an example of a PRL program that creates a custom report summarizing transaction response times by CICS region.

```plaintext
//JOBNAME JOB USER PARAMETERS
/*JOBPARM USER PARAMETERS
//CMRL PROC OUTC='*'.
//    CMRIX='CMR.CMRVx'.
//    C='CICSPRD1'
/*
/** THIS IS SAMPLE JCL TO EXECUTE THE MainView for CICS PERFORMANCE
/** REPORTING LANGUAGE (PRL). CHANGE THE STATEMENTS INDICATED
/** TO REFLECT YOUR INSTALLATION.
/**
/**
//PRL EXEC PGM=CMRPRLN,REGION=4096K
/**
//STEPLIB DD DISP=SHR,DSN=&CMRIX..BBLINK <-- CHANGE AS NEEDED
//SYSPRINT DD SYSOUT=&OUTC
//SYSUDUMP DD SYSOUT=&OUTC
//SYSOUT DD SYSOUT=&OUTC
//ISYSPROG DD SYSOUT=&OUTC,DCB=(LRECL=80,BLKSIZE=3120,RECFM=FBA)
//ISYSPSUT DD SYSOUT=&OUTC,DCB=(LRECL=80,BLKSIZE=3120,RECFM=FBA)
//ISYPDUMP DD SYSOUT=&OUTC,DCB=(LRECL=80,BLKSIZE=3120,RECFM=FBA)
//PNLLIB DD DISP=SHR,DSN=&CMRIX..BBPLIB <-- CHANGE AS NEEDED
/**
/** If using CMRPRL to read CMRSTATS file, uncomment CMRSTATS DD
/** and comment out CMRDETL DD.
/**
//CMRSTATS DD DISP=SHR,DSN=&CMRIX..&C..CMRSTATS <-- EDIT AS NEEDED
//CMRDETL DD DISP=SHR,DSN=&CMRIX..&C..CMRDETL <-- EDIT AS NEEDED
//SORTWK01 DD UNIT=SYSDA,SPACE=(CYL,5) <-- EDIT AS NEEDED
//SORTWK02 DD UNIT=SYSDA,SPACE=(CYL,5) <-- EDIT AS NEEDED
//PEND
//RUNSTEP EXEC CMRL
//SYSIN DD *
SET CLASS = 'R' <-- PERF. REP. CONTROL STATEMENTS
SET ZHEADING = 'SINGLE' <-- PERF. REP. CONTROL STATEMENTS
SET TITLE = 'TRANS RESPONSE FOR ALL TRANSACTIONS' <-- PERF. REP. CONTROL STATEMENTS
SELECT TYPE 6E RECORDS FROM CMRDETL <-- PERF. REP. CONTROL STATEMENTS
    USING T6ETRID T6ERESP T6ECPUR T6EDIST T6EFCWT T6ETSWT T6EPCWT T6EPCWT <-- PERF. REP. CONTROL STATEMENTS
```
After modifying the CMRPRLN batch job to your site’s conditions, include short PRL routines to select data and the format of your report.

### An overview of PERFORMANCE REPORTER reports

This book describes the reports you can create with PERFORMANCE REPORTER, organized into two basic groups: predefined reports and custom reports.

### Predefined reports

As shown in Figure 1 on page 14, PERFORMANCE REPORTER includes sample predefined programs to create CICS, COBOL, and SAS statistical reports.

Each of these groups includes sample programs to create functionally similar reports. Typically, predefined report programs include short-term and long-term reports.

### Short-term reports

Short-term reports show resource usage for the current or immediately preceding CICS execution, as follows:

- The Destination Control Table Report provides either detailed or summarized usage of intra- and extrapartition transient data.

- The File Control Table Report provides either detailed file usage at specific times of the day or total file activity.

  **Note**
  
  The File Control Table Report includes only active table entries.

- The Journal Control Table Report provides either detailed journal and journal buffer usage at specific times of the day or total journal activity.
The Processing Program Table Report provides table-entry statistics that can start with the program name, the highest fetch count, or the highest use count.

*Note*
The Processing Program Table Report includes only active table entries.

The Program Control Table Report provides either detailed transaction usage at specific times of the day or total activity for each transaction.

*Note*
The Program Control Table Report includes only active table entries.

The Temporary Storage Table Report provides the usage of temporary storage and the auxiliary temporary storage data set.

The Terminal Control Table Report provides either terminal activity at specific times of the day or total terminal activity.

*Note*
The Terminal Control Table Report includes only active table entries.

The Transaction Class Usage Report provides class usage within the CICS region.

*Note*
Short-term reports are generated by using PRL input with distributed sample report statements.

**Long-term reports**

Long-term report programs report about tabulated transaction or program resource usage, either individually or grouped as application systems or as histograms of CICS performance.

These reports are as follows:

- The Resource Analysis Report tabulates resource usage and run-time statistics on transactions or programs individually or as a grouped application. The control statements submitted with the batch job control language determine time and date ranges, report input, chronological ordering of the data, and the report output type (a summarized or detailed report or a report of only the resources not meeting service levels).

- The Graphic Histogram Report produces histograms of selected CICS performance parameters or resource usage. The control statements submitted with the batch job define the CICS performance parameters that appear in a report.
The Service Level Analysis Report shows the distribution of transactions into service level ranges.

## Custom PRL reports

With PRL control statements, you can create custom reports from data stored on the CMRDETL and CMRSTATS data sets.

“PRL control statement verbs” on page 149, describes the usage of PRL programs to produce short-term batch reports for CICS users. Using control statements, you specify and format the following information:

- type of data selected from the records stored on CMRDETL or CMRSTATS data sets
- date and time range of selected records
- interval counts for each data item or summarized totals you have specified
- type of CMRDETL or CMRSTATS record processing resulting from a PRL computation or condition
- report titles and column headings
- organization of the report

**Note**

When file expansion is turned on (FILEXPND=YES in CMRSOPT), the structure of expanded resource entries varies by type (that is, a file resource entry differs from a WebSphere MQ entry at the same offsets). A batch program that is processing the first field in each expanded resource entry receives file reads for a file resource entry, select counts for a DB2 resource entry, and open counts for a WebSphere MQ resource entry. All customized programs must process the resource entries separately by using the T6EFILEI field to determine the entry type. Entries of different types should not be processed together.
Collecting and archiving report data

PERFORMANCE REPORTER batch reports are created from data gathered by the MainView for CICS data collectors.

Depending upon the sampling parameters specified when MainView for CICS was customized, CICS data is collected over periodic intervals from monitored targets and then written as records to VSAM databases.

This chapter gives a description of the various data records that are used to create batch reports. An outline of the various steps you must complete to prepare this data is also part of the discussion. “Managing report data” on page 33 gives a more complete description of some of the procedures introduced in this chapter.

This chapter concludes with a section that describes the procedure to archive and purge records from data sets.

Collecting report data

CICS transaction data collected by MainView for CICS is written as Type 6E and Type 6D records to CMRDETL data sets.

MainView for CICS collects CTG transaction data with CTG monitoring and writes the data as Type 6F records to CTGDETL data sets.

CICS statistical data collected is written as SMF 110 statistic records to SMF data sets. The SMF 110 statistic records can be formatted as CMRSTATS records by using batch program CMRSTATS.

The Performance Reporter program uses data dictionaries (CMRDETLN, CTGDETL, and CMRSTATS) that are located in the BBPLIB data set to reformat the stored records for reports.

The MainView for CICS Customization Guide describes how to allocate and initialize the CMRDETL data sets and CTGDETL data sets using the CMRDETL
views. The MainView for CICS PERFORMANCE REPORTER Data Reference describes the format of each type of record that can be stored in a CMRDETL data sets and CTGDETL data sets, as well as the records that can be stored on a CMRSTATS data set.

**Transaction detail data (CMRDETL)**

A detail record is written to the CMRDETL data set for every transaction that occurs in a CICS region monitored by MainView for CICS. This record contains a history of the transaction’s life span.

The record is in a F7 format and it is formatted by BBSAMP member CMRQT6E when it is decompressed. MainView for CICS provides internal compression and external decompression of these records. All timing fields are 12 bytes with 8 bytes containing a value expressed in single microsecond units (providing a capacity of the equivalent of many years).

MainView for CICS knows the CICS release that is being processed and constructs the T6E history record as necessary. When the history record is decompressed before being written to the CMRDETL data set, the proper compression routine is called to compress the record. For record processing through the history views and displays, both records are decompressed and built to the appropriate format.

MainView for CICS provides a pair of decompression programs, CMRCMPRN and CMRCMPWN, that decompress T6E history records to the F7 level.

The following sections explains both the existing and new processing available for the PRL, COBOL and SAS batch reporting facilities as well as the summarization utilities.
The following figure shows an overview of the various steps to prepare detail transaction records for reports.

**Figure 2: MainView for CICS transaction detail records**

```
Figure 2: MainView for CICS transaction detail records
```

```
<table>
<thead>
<tr>
<th>CICS and MainView for CICS data collectors</th>
</tr>
</thead>
<tbody>
<tr>
<td>6E and 6D records</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>BBI-SS address space</td>
</tr>
<tr>
<td>buffer pools</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>CMRDIEL Data set</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>CMRPURG utility</td>
</tr>
<tr>
<td>archived detail records</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>CMRSUMDN utility</td>
</tr>
<tr>
<td>archived summary records</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>summary records</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>printed reports</td>
</tr>
<tr>
<td>third-party reports</td>
</tr>
<tr>
<td>SAS reports</td>
</tr>
</tbody>
</table>
```
Two types of detail transaction records exist. A 6E record is written for every transaction that completes successfully. The CMRDETL data set also contains 6D abended transaction records. They are kept separate from 6E records because their data can be unreliable due to the abended transaction.

Batch jobs that produce the reports select data from the CMRDETL data set by the record ID (the hexadecimal code 6E or 6D) located in the TYPE field of the record. The following example shows a PRL control statement to select a 6E record as input data for a report:

```
SELECT TYPE 6E
```

The MainView for CICS PERFORMANCE REPORTER Data Reference gives a complete description of every field of 6E and 6D transaction records. This section includes two tables that list the fields of the record by offset order and by function.

Before 6E or 6D records are written to the CMRDETL data set, data for application trace (if any), online graphs, and workload services are extracted from it and stored in buffer pools. You can view this data with the MainView for CICS online trace, graph, and workload services. After that, the 6E or 6D records are compressed and written to a CMRDETL data set. You can view the data through MainView for CICS history views and full screen displays.

You can use single or dual CMRDETL data sets. Dual data sets allow you to monitor almost continuously without having to stop and archive the data set when it becomes full. CMRDETL data set switching allows archives to occur without interrupting the data collection process. The MainView for CICS Customization Guide describes how to use the control statements located in member CMRDTL00 of your BBPARM data set to specify the switching conditions for dual CMRDETL data sets.

PERFORMANCE REPORTER batch reports can be prepared from compressed detail transaction records held in the CMRDETL data set or a tape archive. If you are going to use this data to prepare SAS reports or other third-party vendor reports, the records must be decompressed before running the batch report programs. Sample members CMRCJCL and CMRSA10N located in your BBSAMP data set can be edited to run batch jobs that decompress 6E or 6D transaction records.

CMRSUMDN is a utility that summarizes selected detail transaction records. You can summarize transaction activity over periods from minutes to days to create uncompressed summary records stored to tape or a VSAM database. In turn, these summary records can be used to create specialized summary reports.

If you store summary records to a VSAM data set, you can display them online with the MainView for CICS HISTORY service. “Creating reports from summarized data” on page 40 describes how to create summary records from detail transaction records.
CTG Transaction detail data (CTGDETL)

MainView for CICS monitors and collects detail records in the CTGDETL data set for every transaction that occurs in a CICS Transaction Gateway (CTG) region. The detail record contains a history of the transaction’s life span.

There are three types of CTG transactions:

- Synconreturn: these transactions consist of a single call to perform a unit of work on a CICS region
- ExtendedMode: these transactions consist of multiple calls to perform related units of work on a CICS region
- XA: these transactions consist of multiple calls to perform related units of work on one or more CICS regions

Each CTGDETL record contains an entire CTG transaction and consists of a single section. The record includes an overview of the entire transaction in addition to one or more sub-segments representing the individual calls to the CICS region to perform work.

Batch jobs that produce the reports select data from the CTGDETL data set by the record ID (the hexadecimal code 6F), which is located in the TYPE field of the record. For more information about every field of 6F transaction records, refer to the MainView for CICS PERFORMANCE REPORTER Data Reference.

CICS operational statistics (CMRSTATS)

MainView for CICS collects CICS statistics according to parameters specified in the CMRSOPT startup table.

CICS statistical records are first written to the system management facility (SMF) data set with CICS SMF 110 records. Statistical records can then be reformatted to CMRSTATS records by the CMRSTATS batch routine.
Figure 3 on page 24 shows how CICS statistical records are processed for batch reports.

Figure 3: CICS statistical records
Predefined statistical reports select data from the CMRSTATS data set. If you want to
create CICS SMF or other third-party vendor reports from data held in the SMF data
set, you must first filter out MainView for CICS statistical records. A sample member
named CMRSTATX located in your BBSAMP data set removes MainView for CICS
records, leaving SMF 110 records for other reports.

All statistical records contain a field in the header area that identifies the record type.
Records are selected from the CMRSTATS data set by the two-digit hexadecimal
code entered into the TYPE field of the record. Global Performance records (Type
CC) are written to the data set at five-minute intervals. Other types of records are
written according to the startup parameters specified in the CMRSOPT table.

Archiving data (CMRPURG)

CMRPURG is a program that purges and archives data held in CMRDETL and
CTGDETL data sets.

If specified, CMRPURG can also archive and purge selected data to a tape data set
and merge it with other archive tapes.

Refer to the MainView for CICS Customization Guide for information about using
CMRPURG to archive CMRDETL records in a single or dual data set environment.

The execution JCL for CMRPURG is shown in Figure 4 on page 25. Descriptions
corresponding to the numbered statements follow the JCL.

Note
Sample JCL for the CMRPURG program is in the CMRPURGE member of your
BBSAMP data set.

Figure 4: Sample JCL to run to merge, archive, and purge CMRDETL records

```
//jobname  JOB user parameters
/*JOBPARM user parameters
//JOBLIB DD DISP=SHR,DSN=CMR.CMRV5.BBLINK
//ARCHIVE EXEC PGM=CMRPURG
//TAPEIN DD DISP=OLD,DSN=PREV.HISTORY.FILE (1)
//TAPEOUT DD DISP=(NEW,CATLG),DSN=CMR.MERGED.ARCHIVE,UNIT=TAPE (2)
//CMRWRK1 DD DSN=CMR.CMRWRK1,SPACE=(CYL,(10,10)),UNIT=SYSDA (3)
//     DISP=(NEW,DELETE,CATLG)
//CMRDETL DD DISP=SHR,DSN=CMR.CMRV5.CMRDETL (4)
//REPORT DD SYSOUT=* 
//SYSPRINT DD SYSOUT=* 
//SYSIN DD *
DATE=mmdyyyy (5)
HISTORY=YES (6)
```

The numbered items in the following list explain the JCL statements of the example
CMRPURG job shown in Figure 4 on page 25:
1 identifies a data set that is to be merged and archived with the purged records.

2 identifies the output tape data set for the purged records. The default DCB is
   DSORG=PS,RECFM=VB,LRECL=32736,BLKSIZE=32740.

3 specifies a data set, CMRWRK1, to hold all records that are not archived during
   the reorganization phase. The recorder data sets also must be defined with REUSE
   so that reorganization can occur. The records can be written to either tape or disk.

4 identifies the CMRDETL or CTGDETL recorder data sets to be processed.
   Archiving each data set to a separate tape as separate jobs increases access to
   statistical data because there is less volume. More days can then be kept online
   without requiring a purge.

5 specifies the date for purging. This statement must start in column 1. All data
   stamped with this or a previous date is purged and, if requested, archived to the
   output tape data set. DATE=12319999 purges all the data from the recorder data
   set.

   **Note**
   If 12319999 is specified, the CMRWRK1 data set is not required.

The date for purging can also be specified as a number of days preceding today.
   The format is
   DATE=*-nn
   where *nn* is a two-digit number of days before today. Note that the date
   calculated can represent a day in the previous year. The actual date calculated by
   the program is displayed in an informational message on the report output.

6 specifies the type of history tape manipulation to be performed. This statement
   must start in column 1. Valid values are as follows:
   - NO specifies that no input or output history tapes are to be used. All data that has
     a date earlier than the DATE value will be purged.
   - OUT specifies that no input tape is to be used. Only an output tape is to be created
     containing the data to be purged.
   - YES specifies that both input and output tapes are to be used. The data on the
     input tape is to be merged with the data to be purged from the MainView for
     CICS recorder data sets and written to the output tape.

7 LEAVE= specifies whether to leave data in the data set or purge it after archival.
   This statement must start in column 1. Valid values are as follows:
   - NO specifies that data is removed from the data set. The default is NO (it is not
     specified in the sample). NO is required for single CMRDETL environment. It is
not recommended in dual data set environment because data will not be available for online access after the switch.

YES specifies that data remains in the data set after archival. This value is recommended only for dual CMRDETL or CTGDETL data set environment. It provides online access to the data after a data set switch occurs.

**Archive-purge process**

The CMRPURG job performs the following operations:

1. The job reads data from the recorder data sets, merges it with another archive data set (HISTORY=YES), and then writes it to the history output tape (HISTORY=YES or HISTORY=OUT). If HISTORY=NO is specified, the data is deleted and not archived.

2. The job purges data from the recorder data sets.

   **Note**

   Purging is performed here so that abends (such as tape errors) do not leave the recorder data sets in a state of reduced integrity.

The data that is purged is determined by the DATE= control statement. The entire data set can be purged by specifying DATE=12319999, or a portion of the recorder data set can be purged as follows:

- **A.** If only part of a data set is to be purged, you can define the VSAM REUSE parameter when the MainView for CICS recorder data set is defined, as described in the *MainView for CICS Customization Guide*, and allocate a temporary data set, as shown by CMRWRK1 in the previous sample JCL. CMRPURG uses CMRWRK1 to contain temporarily the records that are to remain on the recorder data set. REUSE reorganizes and resets the data sets so that records written to CMRWRK1 from these data sets can be reloaded.

- **B.** CMRPURG processes the CMRDETL or CTGDETL recorder data set. The records stamped with the DATE value or a date previous to the specified date are written as specified by the HISTORY= control statement.
C. The selected records are deleted from the data set if NOREUSE is specified, which can be a time-consuming update to the data set.

If an abend occurs while archiving data, check the messages on the report output to determine at what point it occurred. CMRPURG does not write the same record twice if it exists on both the history and the recorder data sets.

Each phase of the archive-purge process is recorded in the Recorder File Purge Report. Figure 5 on page 28 shows a typical listing for CMRDETL recorder data sets.

**Figure 5: Sample recorder file purge report listings**

<table>
<thead>
<tr>
<th>Report Listing for CMRDETL Recorder File</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>FT290I-DETAIL TRANSACTION RECORDER FILE BEING PROCESSED</td>
<td></td>
</tr>
<tr>
<td>FT293I-BEGINNING ARCHIVE OF REQUESTED RECORDS</td>
<td></td>
</tr>
<tr>
<td>FT294I-ARCHIVE OF REQUESTED RECORDS COMPLETED</td>
<td></td>
</tr>
<tr>
<td>FT295I-BEGINNING PURGE OF REQUESTED RECORDS</td>
<td></td>
</tr>
<tr>
<td>FT296I-PURGE OF REQUESTED RECORDS COMPLETED</td>
<td></td>
</tr>
<tr>
<td>FT297I-CMRWRK1 DATASET FOUND - BEGINNING REORGANIZATION</td>
<td></td>
</tr>
<tr>
<td>FT298I-ENTERING SECOND PHASE OF REORGANIZATION</td>
<td></td>
</tr>
<tr>
<td>FT301I-REORGANIZATION OF RECORDER FILE COMPLETED</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>--------- RECORDER FILE ---------</th>
<th>-- HISTORY_FILE_ACTIVITY --</th>
</tr>
</thead>
<tbody>
<tr>
<td>INPUT</td>
<td>PURGED</td>
</tr>
<tr>
<td>000001667</td>
<td>000001666</td>
</tr>
</tbody>
</table>

Purging summary records from the CMRSUMDN data set

The CMRSUMDN utility summarizes CMRDETL data over selected intervals and places uncompressed summary records in an output VSAM data set. You can create batch reports from the summary records or view the records online with the HISTORY service.

CMRPURG can purge selected summary records from the CMRSUMDN data set. This purging allows you to keep some summary records required for long-term reports and remove others that are no longer needed.

The JCL shown in Figure 4 on page 25 can be used to run the CMRPURG program. An additional WORKLOAD control statement must be added to the job stream beneath the SYSIN DD statement. The WORKLOAD control statement selects which summary records are purged.

The WORKLOAD control statement has the following positional parameters:

```plaintext
WORKLOAD = wkld, group, time
```

where
### Table

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| wkld      | is the name of the workload used to summarize records  
All records are purged by default. |
| group     | is the name of the group of CICS regions combined to create a summary workload  
All groups are purged by default. The group name is specified with the SGROUP control statement of the CMRSUMDN utility. |
| time      | is the summarization interval of summary records  
All summary intervals are purged by default. |

The following examples show how to use the WORKLOAD control statement with the CMRPURG program to purge summary records from the CMRSUMDN data set:

---

**Example**

WORKLOAD=IBMTRANS,*,1440

In the preceding example, daily summary records of all groups of the IBMTRANS workload are purged from a VSAM data set.

---

**Example**

WORKLOAD=,60

In the preceding example, all hourly records from all workloads and groups are purged from a VSAM data set.

---

### Converting CMRDETL records into CICS 110 SMF format

MainView for CICS enables you to convert transaction history records (CMRDETL) into CICS 110 SMF format.

Any MainView for CICS CMRDETL record can be converted regardless of which release of CICS or MainView for CICS produced the record.

Support is provided to create the SMF 110 Performance records for

- CICS Transaction Server 4.1
- CICS Transaction Server 4.2
Conversion is performed by the CMRCMPWN utility program. CMRCMPWN enables you to control the output format by using one of several input parameters. CMRCMPWN supports the following input parameters:

- `FORMAT=4.1` or `4.1Y`
- `FORMAT=4.2` or `4.2Y`
- `FORMAT=5.1` or `5.1Y`
- `FORMAT=5.2` or `5.2Y`
- `FORMAT=5.3` or `5.3Y`

The requested output format is constructed regardless of the CICS release that is used for input. For example, if the T6E input is from CICS/TS 4.2 and you specify `FORMAT=5.1`, CICS TS 5.1 SMF format data records are built. In this case, however, all new CICS TS 5.1 data fields that are unavailable in CICS TS 4.2 are initialized to binary zeros or blanks as appropriate.

The `Y` in the second format sets the date format to `YYYYDDDF`. The default format is `CCYYDDDF`.

**Note**

For more information about the `FORMAT` parameter and sample JCL for running the CMRCMPWN program, see “CMRCMPWN batch decompression program” on page 37.

The converted data contains SMF formatted records: the SMF header followed by the SMF product section followed by the blocked data records. The transaction data records are blocked to fit below 32K with each blocked area preceded by the dictionary field connectors. The SMF header and SMF product section do not have the OPSYS, MCT Options, or Entry Time information.

The first record written is the CICS dictionary record, which is almost equivalent to the complete dictionary for the CICS specified on the `FORMAT=` statement. The difference is in the DBCTL section; instead of 256 bytes of DBCTL data area, the CMRCMPWN SMF dictionary has 14 USER fields with DBCTL data. The dictionary record is followed by the SMF transaction data record. The data section is structured as follows:
The CMRNDATA and CMRNDDB2 areas are defined in the SMF record as if the your site had specified an MCT, including the MainView for CICS MCT definitions in the BBSAMP members CMRNDMCT and CMRNZ2MCT, respectively.

The CICS data area is described by the DFHMNTDS dsect starting at offset 8. The DFHRMI area is described by the DFHMNCBD dsect. The MainView for CICS user counts/clocks areas consist of four 12-byte clocks followed by 10 four-byte counts. These areas contain some MainView for CICS calculated DBCTL information. The dictionary entries for these fields are identified by USER fields 001-004 for the clocks and USER fields 001-010 for the counts. No macro or copy book is available for the format of the USER fields.

The clock fields are (refer to the DFSDSTA control block) as follows:

| USER 001 | number of seconds of elapsed time for DBCTL requests (DBC$SCHT/16) |
| USER 002 | delta between PSB schedule start and stop (DBC$DATN-DBC$DATS) |
| USER 003 | not used |
| USER 004 | elapsed time for database I/O (DBC$TMIO/16) |

The count fields are as follows:

| USER 001 | database GU calls (DBC$GU) |
| USER 002 | database GN calls (DBC$GH) |
| USER 003 | database GNP calls (DBC$GNP) |
| USER 004 | database GHU calls (DBC$GHU) |
| USER 005 | database GHN calls (DBC$GHN) |
| USER 006 | database GHNP calls (DBC$GHNP) |
| USER 007 | database ISRT calls (DFH$ISRT) |
| USER 008 | database DLET calls (DFH$DLET) |
| USER 009 | database REPL calls (DFH$REPL) |
| USER 010 | total database calls (DBC$TOTC) |

The MainView for CICS data area is formatted by the CMRNDATA assembler dsect or the CMRNDSAS SAS field identifications. The dictionary entry is identified by the name of CMRNDATA.
The MainView for CICS DB2 data area is formatted by the CMRNDB2 assembler dsect or the CMRN2SAS SAS field identifications. The dictionary entry is identified by the name of CMMRDB2.

The TCP/IP data area is based on the data provided in the MCT as defined by the TCP/IP Supplied MCT information. MainView for CICS supplies all of the available TCP/IP information, but what is built in the SMF record is based on the TCP/IP information provided by the user in the MCT used by the CICS target.
Managing report data

This chapter describes several procedures to prepare data that appears in batch reports. Separate procedures describe how to

■ create statistical records
■ reformat compressed detail transaction records
■ summarize detail transaction records

Creating statistical records

To create statistical records from data collected by a CICS region, you must run the CMRSTATS program.

The CMRSTATS program reads data directly from an SMF data set or a sequential extract from an SMF data set. The program determines the type of SMF data set that is allocated to the job. Only one VSAM data set is read at a time, and the required SMF data set (SYS1.MAN1, SYS1.MAN2) must be specified.

Sample JCL to run this program is provided in BBSAMP as member CMRSTJCL. You can use the sample JCL along with distributed sample report statements to generate CICS statistical reports.

Figure 6 on page 33 shows the JCL for running the CMRSTATS program.

Figure 6: JCL for CICS statistical reports (CMRSTJCL)

```
// * This sample JCL can be used to extract CMRSTATS statistical
// * data from the SMF data produced by MAINVIEW .
// * You must make some changes to this JCL. The lines that may
// * require changes are indicated by a '<==' marker.
// *JOBLIB DD DSN=MAINVIEW.CMRxx.BBLINK, '<==' change
// DISP=SHR
// * The following step (STATS) extracts the CMRSTATS data from
// * the SMF file and places it into a file that can be used
// * by the Performance Reporter Language (PRL), or other user
// * written programs to produce reports on the performance of
// * your CICS regions.
```
Creating statistical records

The following rules apply to the control statements that you specify with CMRSTATS:

- Each SYSIN statement contains only one control statement.
- The control statement can start anywhere from column 1 but cannot be continued on the next line.

**Note**
You must change some of the JCL shown in Figure 6 on page 33. Lines that might require changes are indicated by <=== markers.
See Table 20 on page 134 for a list of sample programs in BBSAMP.
If you do not supply application names, all CICS data from the SMF data set is processed.
You can select data from the SMF data set that was collected from a specific CICS region.

If both generic and specific application IDs are specified, the specific ID is used for selection.
You can select only one application ID.

Valid control statements are as follows:

Table 1: CMRSTATS control statements

<table>
<thead>
<tr>
<th>Control Statement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FROM TIME=hh:mm:ss</td>
<td>Include SMF data created after the specified time.</td>
</tr>
<tr>
<td>TO TIME=hh:mm:ss</td>
<td>Include SMF data created up to and including the specified time.</td>
</tr>
<tr>
<td>FROM DATE=mm/dd/yyyy</td>
<td>Include SMF data produced from the specified date onward.</td>
</tr>
<tr>
<td></td>
<td><strong>Note</strong>: If you specify a date in the old format (mm/dd/yy), yy is interpreted as 19 yy.</td>
</tr>
<tr>
<td>FROM DATE=*- n</td>
<td>Include the SMF data from today minus n days onward. Yesterday would be *-1.</td>
</tr>
<tr>
<td>TO DATE=mm/dd/yyyy</td>
<td>Include SMF data produced up to and including the specified date.</td>
</tr>
<tr>
<td></td>
<td><strong>Note</strong>: If you specify a date in the old format (mm/dd/yy), yy is interpreted as 19 yy.</td>
</tr>
<tr>
<td>TO DATE=*- n</td>
<td>Include SMF data up to and including today minus n days. Yesterday would be *-1.</td>
</tr>
<tr>
<td>GAPPLID=applid</td>
<td>Include only SMF data for the specified generic application ID.</td>
</tr>
<tr>
<td>SAPPLID=applid</td>
<td>Include only SMF data for the specified specific application ID.</td>
</tr>
<tr>
<td>APPLID=applid</td>
<td>Synonym for SAPPLID. <strong>Note</strong>: Parameter errors cause the run to terminate with a return code of 8 and an appropriate message on SYSPRINT.</td>
</tr>
</tbody>
</table>

**SYSPRINT error messages**

SYSPRINT errors messages are accompanied by a return code of 8. Messages are as follows:
FT581 I Previous parameter was invalid

Explanation: The parameter statement listed above the message on SYSPRINT is not valid.

User response:

FT582I SMFIN DD statement missing or invalid

Explanation: The OPEN for SMFIN failed. Generally, this error means the DD statement is missing.

User response:

FT583I CMRSTATS DD statement missing or invalid

Explanation: The OPEN for the CMRSTATS DD statement failed. Generally, this error means the DD statement is missing. However, it also could mean an invalid BLKSIZE parameter. BLKSIZE must be at least 6000.

User response:

FT584I Insufficient memory to run the program

Explanation: More memory is required to process the data that is used for a report. Either increase the region size or restrict the report to a specific range of data. The date or time ranges normally have little effect on the amount of required memory. Memory is allocated above the 16 MB line.

User response:

Console error messages

The following console error message results from a 2048 ABEND:

FT585I SYSPRINT DD statement missing or invalid.

Explanation: The OPEN for the SYSPRINT data set failed. Generally, this error means the SYSPRINT DD statement is missing.

User response:

Reformatting compressed detail transaction records

Type 6E and 6D records are compressed when they are stored on the CMRDETL data set.
You must decompress these records before they can be read by programs other than PERFORMANCE REPORTER programs. The records can be decompressed into standard MainView for CICS data or a format compatible with CICS/MVS data or CICS TS SMF 110 (CMP) data.

- MainView for CICS offers two programs to decompress the CMRDETL data set records:
- CMRCMPWN can be used in a separate batch job to decompress CMRDETL records to the F7 level
- CMRCMPRN can be called from within a user program to dynamically decompress CMRDETL records to the F7 level

**Note**
Type 6E records produced by earlier versions of MainView for CICS are always decompressed to the record format supported by the current version. Any fields that are new to the earlier version are initialized to zeros or blanks.

---

**CMRCMPWN batch decompression program**

The following figure shows a sample batch job to execute the CMRCMPWN program.

The CMRCJCL job is in your BBSAMP library. See “Reformatting compressed detail transaction records” on page 36 for more information.

Compressed Type 6E records can reside on a current, active CMRDETL data set or be archived to tape. You select either source with the appropriate DD statement and an INPUT= control statement (INPUT=VSAM or INPUT=TAPE). The following figure shows examples of both DD statements.

**Figure 7: JCL for CMRCMPWN batch decompression program**

```plaintext
//S1 EXEC PGM=CMRCMPWN
//STEPLIB DD DSN=CMR.CMRV3.BBLINK,DISP=SHR
//CMRDETL DD DSN=CMR.CMRV3.CMRDETL,DISP=SHR <- IF INPUT=VSAM
//*TAPEIN DD DSN=....ARCHIVE.DATA,DISP=SHR <- IF INPUT=TAPE
//SYSUDUMP DD SYSOUT=* 
//*
// The following OUTPUT DD statement is the data set that contains
// the output from this utility. No DCB information should be
// coded in this DD statement. The DCB information is determined
// by the utility based on the type of output being produced.
//*
//OUTPUT DD DSN=CMR.PROCESS.DATA,DISP=(,CATLG),SPACE=(TRK,(15,15)),
// UNIT=SYSDA
//SYSIN DD *
INPUT=VSAM
FORMAT=CMR
```
Control statements follow the SYSIN DD statement in the job stream. Control statements are as follows:

- The FORMAT statement specifies the format of the decompressed record.
  - FORMAT=CMR builds decompressed detail records that match the MainView for CICS format described by —CMRQT6E for CMRCMPWN
  - FORMAT=4.1 or 4.1Y builds CICS TS 4.1 SMF 110 records
  - FORMAT=4.2 or 4.2Y builds CICS TS 4.2 SMF 110 records
  - FORMAT=5.1 or 5.1Y builds CICS TS 5.1 SMF 110 records
  - FORMAT=5.2 or 5.2Y builds CICS TS 5.2 SMF 110 records
  - FORMAT=5.3 or 5.3Y builds CICS TS 5.3 SMF 110 records
- The date is formatted as ccyyddds where:
  - cc is the 2-digit century value (01 for 2000)
  - yy is the 2-digit year value
  - ddd is the 3-digit day value
  - s is a packed decimal sign value
  - If Y is specified, the date is formatted as yyyyddds where:
    - yyyy is the 4-digit year value
    - ddd is the 3-digit day value
    - s is a packed decimal sign value
- The INPUT control statement specifies the type of input data set holding the compressed CMRDETL records.
  - INPUT=VSAM specifies that compressed records are stored on the current, active CMRDETL data set.
  - INPUT=TAPE specifies that compressed records are archived to a tape data set.
- NEWAPPL specifies a 1- to 8-character alphanumeric name that identifies the APPLID placed in the SMF header portion of decompressed CMRDETL records. This value replaces existing APPLIDs of compressed input records. Decompressed records retain their existing APPLIDs if the NEWAPPL control statement is not specified.
The SELECT control statement selects records to be decompressed by the name of the CICS region that produced the CMRDETL record.

If the SELECT control statement is not specified, the first CICS ID encountered is used.

SELECT=xxxxxxxx specifies a 1- to 8-character alphanumeric name of a CICS region.

SELECT=* selects all input records.

CMRCMPRN decompression program

The CMRCMPRN program dynamically decompresses CMRDETL records.

Instead of executing CMRCMPWN in batch mode, the CMRCMPRN program can be called from a user program with an assembler language routine, as shown in the following figure.

See “Reformatting compressed detail transaction records” on page 36 for more information.

Figure 8: Assembler routine to dynamically call CMRCMPRN

```
LA R1,PARMLIST
CALL CMRCMPRN
LTR R15,15 Q. GOOD DECOMPRESS
BNZ HANDLERR NO - GO HANDLE ERROR

PARMLIST DS 0D
PARM1 DC A(DFLAG)
PARM2 DC A(RECORD)
DFLAG DC C^D'
```

Standard linkages include the following registers:

- Register 1 contains the address of a parameter list that
  - points to a ‘D’ flag that signifies a decompress request
  - points to the record to be decompressed

The field must be large enough to contain the entire decompressed record. LLBB must be the first record field. It must be set to the current length of the record.

- Register 13 contains the address of a standard register save area.

- Register 14 contains the return address.

- Register 15 contains the return code from the CMRCMPRN program.
Creating reports from summarized data

This section explains how to use PERFORMANCE REPORTER’s CMRSUMDN utilities to summarize detail transaction records stored on your CMRDETL data set.

CMRSUMDN decompresses, processes and creates all records at the F7 level formatted by BBSAMP member CMRQT6E

The summarization batch utility program creates workloads from CICS performance records. You can summarize data collected from multiple CICS regions in intervals from 1 minute to 24 hours. The summary workloads produced by the summarization utility are compatible with the COBOL, SAS, and Performance Reporting Language (PRL) report programs of MainView for CICS.

The following figure illustrates how CMRDETL records are summarized. The summarization utility processes unsummarized records held in a VSAM or sequential data set.

Figure 9: Summarizing CMRDETL records with the CMRSUMDN utility

After the records are summarized over the intervals you requested, the newly created summary workloads are stored on VSAM or sequential output data sets. Summary workloads can be created with data obtained from any supported version of CICS.

If you store your summarized records on a VSAM data set, you can view them online with the MainView for CICS HISTORY service. “Viewing summary records” on page 60 describes the procedure to use the HISTORY service.

Summarizing CMRDETL records gives you several benefits. One benefit is that fewer records are required to produce reports that span long periods of time. For example, producing a monthly report of average daily CPU usage requires only one record per day.
Another benefit of using summarized records is that you can write simpler report programs. Fewer records must be specified for each report interval and the batch program to create the reports generally requires less processing time for each query.

Summarized records are useful for identifying performance trends that become apparent only over long intervals. After you locate CICS problem areas with summarized records, you can use unsummarized data to create reports that give you the detail you need to analyze problems.

**Summarization utility JCL**

The following figure shows an example of JCL to execute the CMRSUMDN programs and summarize report data.

**Figure 10: Sample JCL for creating summary data**

```
//S1 EXEC PGM=CMRSUMDN
//STEPLIB DD DSN=CMR.BBLINK,DISP=SHR
//*
//CMRDETL DD DSN=CMR.DETAIL.DATA,DISP=SHR
//OUTPUT DD DSN=CMR.SUMMARY.DATA,DISP=SHR
//COPYDD DD DSN=CMR.COPY.CMRDETL,DISP=SHR
//REPORT DD SYSOUT=* 
//SYSIN DD *
```

This sample JCL contains the following JCL statements:

- EXEC specifies the name of the program that generates the summarized detail data (PGM=CMRSUMDN).

- //STEPLIB defines the program library containing the MainView for CICS load modules.

- //CMRDETL defines the name of the input data set containing the detail records to be summarized. The detail records can be from a single CICS region or multiple CICS regions. For maximum performance, detail records should be sorted in date and time sequence.

  If necessary, you can use a sort utility to combine multiple detail files into date and time sequence; for example, you can specify

  —SORT FIELDS=(9,20,CH,A) for VSAM

  —SORT FIELDS=(13,20,CH,A) for sequential (to account for the RDW)

- //OUTPUT defines the name of the summary output data set. You must specify the BLKSIZE parameter when creating a new sequential output data set.

  See BBSAMP member CMRDFSUM for guidelines about allocating a VSAM or sequential data set.
When you run multiple executions of the CMRSUMDN program against the same CICS region, workload, and time frame using a VSAM data set:

- Workloads with the same name as existing records are combined with the records.
- New workloads are inserted.
- A maximum of 999 workloads are supported.

//COPYDD, which is optional, defines the name of the selected sequential output data set. “COPY={NO | YES}” on page 52 for more information.

Note
When creating a copy data set, specify RECFM=U, LRECL=0, and BLKSIZE=32768 bytes.

//REPORT defines the SYSOUT data set containing the program statistics compiled by CMRSUMDN.

//SYSIN contains the control statements you must use to specify the input records and summarization options. See “Control statement descriptions” on page 44 for a description of each control statement.

Summarization utility control statements

The control statements described in this section run the summarization utility.

They are coded after the SYSIN DD statement in your JCL. Each control statement uses parameters to specify the report data and summarization options. In the descriptions in this section, default parameters are underlined.

Coding control statements

The global control statement parameters affect all the workloads defined in the summarization job.

Local parameters apply to only a specific workload. You can code summarization control statements in a predefined order that sets both global and local conditions for each of the workloads defined in the job.

BMC recommends coding the control statements in the following order:
1. global control statements
   Global control statements set conditions for the job itself and are not directly related to workloads. The global control statements are NEWAPPL, REPLACE, and COPYAPPL.

2. default control statements
   Default control statements set the default conditions for the workloads specified in the summarization job.
   A default control statement can be any control statement except one that is explicitly defined as global or workload-specific.

3. workload definition control statements
   The workload definition control statements define the workloads in the summarization job. They are coded immediately before the default override and the workload-specific control statements.
   Together, the following workload definition control statements specify the conditions for a specific workload in the summarization job:
   - A. default override control statements
     The workload default override control statements define operating conditions for a specific workload. They may differ from those previously set by the default control statements.
     A default override control statement can be any control statement except one that is defined as global or workload-specific.
   - B. workload-specific control statements
     The following workload-specific control statements specify the input data that is used by a specific workload in the summarization job:
     SKIPFLIO
     PROCFLIO
     EXCLxxxx
     INCLxxxx

The following figure shows the recommended sequence of control statements. These three sets of control statements are repeated for each workload in the summarization job stream.

**Figure 11: Recommended control statement order in CMRSUMDN JCL**

```
REPLACE=YES
TYPE=NOTERM
FILES=TIME
MINUTES=15
WORKLOAD=PAYROLL
INCLTRAN=PAYR
MINUTES=5
EXCLUDER=DRP1CICS
WORKLOAD=ACCOUNTS
MINUTES=10,30
```
Control statement descriptions

The summarization utility has 27 control statements.

These control statements fall into four broad classes based upon how often you are expected to use them.

You will use some control statements in almost every summarization job. Other control statements you will use often, but not for every job. A third class is composed of control statements that are used even less frequently. The last class contains control statements that are used rarely, only to set unique conditions for a summarization job.

The following four sections discuss the summarization control statements, grouped together by frequency of use.

Commonly used control statements

The following control statements in this class are used for almost every summarization job:

Table 2: Commonly used control statements

<table>
<thead>
<tr>
<th>Control statement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WORKLOAD=name</td>
<td>specifies the name assigned to the workload The utility places the workload name in the T6EPGNM (program) field of the summarized record. If you do not explicitly specify a workload name, the default workload name is SYSTEM. <strong>Note:</strong> Up to 999 workloads are supported for a specific VSAM data set.</td>
</tr>
<tr>
<td>Control statement</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>MINUTES={1</td>
<td>5</td>
</tr>
<tr>
<td>1 specifies increments of 1 minute.</td>
<td></td>
</tr>
<tr>
<td>5 specifies increments of 5 minutes.</td>
<td></td>
</tr>
<tr>
<td>10 specifies increments of 10 minutes (the default).</td>
<td></td>
</tr>
<tr>
<td>15 specifies increments of 15 minutes.</td>
<td></td>
</tr>
<tr>
<td>30 specifies increments of 30 minutes.</td>
<td></td>
</tr>
<tr>
<td>60 specifies increments of 1 hour.</td>
<td></td>
</tr>
<tr>
<td>1440 or DAILY specifies increments of 1 day.</td>
<td></td>
</tr>
<tr>
<td>When the MINUTES control statement precedes the WORKLOAD statement, the summarization interval you specify is in effect for all workloads. To specify a unique summarization interval for each workload, place an appropriate MINUTES control statement after each WORKLOAD statement.</td>
<td></td>
</tr>
<tr>
<td>TYPE={ALL</td>
<td>NOTERM</td>
</tr>
<tr>
<td>ALL specifies that all tasks are included in the workload.</td>
<td></td>
</tr>
<tr>
<td>NOTERM specifies that only non-terminal tasks are included in the workload (the default).</td>
<td></td>
</tr>
<tr>
<td>TERM specifies that only terminal tasks are included in the workload. Non-terminal tasks are excluded.</td>
<td></td>
</tr>
<tr>
<td>When the TYPE control statement precedes the WORKLOAD statement, the type of task you specify is in effect for all workloads. To specify a different type of task for each workload, place an appropriate TYPE control statement after each WORKLOAD statement.</td>
<td></td>
</tr>
</tbody>
</table>
**Control statement** | **Description**  
--- | ---  
**SGROUP=list** | specifies a list of CICS regions that are combined to create a workload. The first name in the SGROUP list identifies the summarized records.  
*Note:* SELECT is an alias for SGROUP. That is, you can use SELECT=list to specify a list of CICS regions.  
Only one SGROUP and its identifier are used in a single execution; subsequent references to the same name use the first definition. For example, if you specify the following statements, you get the same region list for the second SGROUP as you do for the first SGROUP:  

```
SGROUP=AORS,CICS1,CICS2,CICS3  
... and other statements  
... SGROUP=AORS  
...  
```

This situation simplifies the definition of CICS system groups. When the SGROUP control statement precedes the WORKLOAD statement, the list of CICS regions you specify is in effect for all workloads. To specify a different list of CICS regions for each workload, place an appropriate SGROUP control statement after each WORKLOAD statement.  
*Note:* It is not necessary to combine all the CICS regions data into a single execution to produce combined workloads. The summarization utility combines data from different executions for the same groups and workloads.

---

**Frequently used control statements**

The following control statements in this class are used often but not for every summarization job.

**EXCLxxxx=list**

Specifies a list of resources to exclude from the summary. The EXCL control statement can be repeated for as many resources as required. EXCLxxxx must be specified within a WORKLOAD.

The xxxx variable must be one of the following values:

- **PLAN** (name of the DB2 plan associated with the transaction)
- **4GLS** (name of the 4GL panel or program associated with the transaction)
- **OPID** (three-character OPID associated with the transaction)
- **TERM** (four-character CICS terminal ID associated with the transaction)
- **TRAN** (transaction ID associated with the transaction)
- **PROG** (program name associated with the transaction)
- **NETN** (up to an 8-character net name associated with the transaction)
- **USER** (up to an 8-character extended user ID associated with the transaction)
- **FILE** (name of a file referenced by the transaction)

**INCLxxxx=list**

Specifies a list of resources to include in the summary. The INCL control statement can be repeated for as many resources as required. INCLxxxx must be specified at the workload level.

The xxxx variable must be selected from the following values:

- **PLAN** (name of the DB2 plan associated with the transaction)
- **4GLS** (name of the 4GL panel or program associated with the transaction)
- **OPID** (three-character OPID associated with the transaction)
- **TERM** (four-character CICS terminal ID associated with the transaction)

**Note**

If you want to combine summarized records gathered in intervals of 10 minutes into hourly records, you must specify INCLTERM=10 to ensure records that overlap time frames are not combined to give invalid data.

- **TRAN** (transaction ID associated with the transaction)
- **PROG** (program name associated with the transaction)
- **NETN** (up to an 8-character net name associated with the transaction)
- **USER** (up to an 8-character extended user ID associated with the transaction)
- **FILE** (name of a file referenced by the transaction)

**NEWAPPL=name**

Specifies a new application ID (APPLID) that is written in the key of the output records.

**Note**

The NEWAPPL control statement is used when you want to view records online. The HISTORY service expects the key of the record to match the name specified in the target field of the display panel. SUMMDATA is the default key. Use NEWAPPL when you want to change the default key.
Less frequently used control statements

The following sections discuss the control statements in this class that are used occasionally.

REPLACE={NO | YES}

REPLACE specifies whether data collected during the execution of the job replaces any data from previous executions that have the same workloads.

This control statement applies to VSAM data sets only. You must use REPLACE to re-create workloads from detail records that have been previously processed. By default, the data is combined with existing data.

FILES={TIME | SUMMARY | COUNT}

FILES specifies the summarization option for the file entries in the records.

If you specify FILES prior to the first WORKLOAD control statement, it defines the default type of summarization for all workloads, as follows:

- TIME (summarizes records by file entries, and the 255 files with the longest elapsed I/O time are written to the summarized file) TIME is the default.
- SUMMARY (summarizes all entries, and only the first 20 entries are written to the summarized file)
- COUNT (summarizes the file entries, and the 20 files with the largest number of I/Os are written to the summarized file).

DETAIL={NO | YES}

DETAIL specifies whether the summarization utility treats files as individual entities or as a single entity for each file type. If you specify DETAIL prior to the first WORKLOAD control statement, it defines the default value for all workloads.

- NO summarizes files entries by a common resource, such as CICS, Software AG ADABAS, DBCTL, DB2, and so forth (the default).
- YES specifies that you want each file entry processed separately.

**Note**

Use DETAIL when you want to determine the overall effect of a given area rather than the overall effect of specific files.
ZERODATA=\{NO | YES\}

ZERODATA specifies whether intervals are written to the summarized file when no tasks are executed for a given workload; for example, when CICS is not active, no tasks are executed.

If you specify ZERODATA before the first WORKLOAD control statement, it defines the default value for all workloads.

Values are as follows:

- **NO** specifies that the utility does not write intervals with zero tasks to the summarized file (the default).
- **YES** specifies that the utility writes intervals with zero tasks to the summarized file.

**FROM DATE=mm/dd/yyyy**

FROM DATE specifies the starting date of the selected records.

---

**Note**

If you specify a date in the old format (mm/dd/yy), yy is interpreted as 19yy.

---

You can specify the starting date using a relative format that is an offset from the current date. For example, to start the selected records with yesterday’s date, specify:

**FROM DATE=*-1**

If you specify FROM DATE before the first WORKLOAD control statement, it defines the default starting date for all workloads.

**TO DATE=mm/dd/yyyy**

TO DATE specifies the ending date of the selected records.

---

**Note**

If you specify a date in the old format (mm/dd/yy), yy is interpreted as 19yy.

---

You can specify the ending date using a relative format that is an offset from the current date. For example, to end the selected records with yesterday’s date, specify:

**TO DATE=*-1**
If you specify TO DATE before the first WORKLOAD statement, the utility provides the default ending date for all workloads at the workload level.

**FROM TIME= hh:mm:ss**

FROM TIME specifies the starting time of the selected records. The format is \texttt{hh:mm:ss}.

If you specify FROM TIME before the first WORKLOAD control statement, it defines the default starting time for all workloads.

**TO TIME= hh:mm:ss**

TO TIME specifies the ending time for selecting records. The format is \texttt{hh:mm:ss}.

If you specify TO TIME before the first WORKLOAD control statement, it defines the default ending time for all workloads.

**START TIME= hh:mm:ss**

START TIME specifies the starting time for the selection interval. You can use this time to specify a period, such as the prime shift for a given workload.

**STOP TIME= hh:mm:ss**

STOP TIME specifies the ending time for the selection interval. STOP TIME can be used with TO DATE to specify an ending date and time.

The following example shows START TIME, STOP TIME, FROM TIME, and TO TIME control statements together in a job stream.

```
FROM DATE=*-5
FROM TIME=12:00:00
TO DATE=*-1
TO TIME=12:00:00
WORKLOAD=TEST1
INCLTRAN=AA*
START TIME=08:00:00
STOP TIME=17:00:00
WORKLOAD=TEST2
INCLTRAN=AA*
FROM DATE=*-5
TO DATE=*-3
```

In this example, the default date and time ranges for the selected records are from noon five days ago through yesterday at noon.

Workload TEST1 uses records that are collected between 8:00 A.M. and 5:00 P.M. On the first day, records are collected from 12:00 P.M. to 5:00 P.M. On the last day, records are collected from 8:00 A.M. to 12:00 P.M.
Workload TEST2 collects records from noon four days ago through the following day at noon.

**MINRESP=nnnnn.nnn**

MINRESP sets a minimum response time threshold of a transaction record in seconds. Records with transaction response times that exceed the minimum threshold are included in the summarization workload. Records with response times less than the threshold are excluded.

**MAXRESP=nnnnn.nnn**

MAXRESP sets a maximum response time threshold of a transaction record in seconds. Records with response times less than the maximum threshold are included in the summarization workload. Records that exceed the maximum response time are excluded.

The MINRESP and MAXRESP control statements are often used together to filter transaction records by response times. You can create summary workloads that contain only those transactions with response times that fall within the interval set by the two control statements.

**MINCPU=nnnnn.nnn**

MINCPU sets a minimum CPU usage threshold of a transaction record in seconds. Records with CPU usage that exceed the minimum threshold are included in the summarization workload. Records with CPU usage less than the threshold are excluded.

**MAXCPU=nnnnn.nnn**

MAXCPU sets a maximum CPU usage threshold of a transaction record in seconds. Records with CPU usage less than the maximum threshold are included in the summarization workload. Records that exceed the maximum threshold are excluded.

The MINCPU and MAXCPU control statements can be used together to filter transaction records by CPU usage. You can create summary workloads that contain only those transactions with CPU usage that falls within the interval set by the two control statements.

**Rarely used control statements**

The control statements in this class set unique conditions that are rarely used for most summarization jobs.
**TABLE**=*name*

TABLE specifies the name of a table created using the CMRRAPM macro, which specifies transactions and programs to be included in the workload.

The TABLE control statement

- is provided for compatibility with the CMRRAPM table
- has limited flexibility

For example, default values are used for all control statements (other than WORKLOAD and INCLxxxx, which are contained within the CMRRAPM definitions).

**COPYAPPL**=*name*

COPYAPPL specifies an application ID (APPLID) that is written in the key of the records copied to the copy data set.

**COPY**={NO | YES}

COPY specifies whether the selected records are copied to the output file specified with the COPYDD DD statement.

When COPY is specified before the first WORKLOAD control statement, it determines whether selected records are copied to the output file for all workloads.

Values are as follows:

- NO specifies that selected records are not copied. NO is the default.
- YES specifies that if a record is selected for summarization, it is also copied to the sequential data set specified with the COPYDD DD statement.

---

**Note**

These copied records are useful when performing detailed analysis of the data without reading the entire detail data set again.

---

**SKIPFLIO**=*list*

SKIPFLIO specifies a list of files that are excluded in the summarized file data. SKIPFLIO must be specified at the workload level.
**Note**
Use of SKIPFLIO does not affect the selection of records, as happens with the EXCLFILE control statement.

**PROCFLIO=list**

PROCFLIO specifies a list of files to include in the summarized file data. Other files are not summarized. Use PROCFLIO to insure that files you specify are reported for a workload. PROCFLIO must be specified at the workload level.

**Note**
Use of PROCFLIO does not affect the selection of records, as happens with the INCLFILE control statement.

**USEREXIT=name**

USEREXIT specifies the name of a user-written routine that is given control when a record is processed during the summarization job. The input record is passed to the exit. The record is tested against conditions set by the user-written routine. The user exit either includes or excludes the record in the summarization job based upon the outcome of the test conditions.

BBSAMP member CMRUSRDRD is an example of this exit.

**Running the summarization utility**

"Running the summarization utility" on page 53 is an example of a job that runs the summarization utility and creates summary workloads.

**Figure 12: Summarization utility JCL and control statements**

```
//USR1SUM JOB (3831), 'USER NAME', CLASS=T, MSGCLASS=R,
// NOTIFY=USR1, TIME=60
//*
//S1 EXEC PGM=CMRSUMDN, REGION=4096K
//STEPLIB DD DSN=CMR34.CMR1.LOAD, DISP=SHR
//CMRDETL DD DSN=CMR1.DETAIL.DATA, DISP=SHR
//*
//SYSUDUMP DD SYSOUT=* 
//OUTPUT DD DSN=CMR.SUMMARY.DATA, DISP=SHR
//REPORT DD SYSOUT=* 
//SYSIN DD *
*
* SAMPLE CMRSUMDN INPUT
* REFER TO MEMBER CMRSUMDN FOR EXECUTION JCL
* THIS MEMBER CREATES DAILY SUMMARY RECORDS FOR 4 WORKLOADS
* MAINVIEW FOR CICS TRANSACTIONS
* CICS TRANSACTIONS
```
These control statements are examples from member CMRSUMD1 of the BBSAMP data set. The job creates four different summary workloads each day. A brief description of each workload is given in the comments section of the job shown in “Running the summarization utility” on page 53. A description of the summarized transaction records created for each workload is shown beneath the SYSIN DD statement.

The report shown in the following figure is produced by the utility. It shows the input CMRDETL records that were summarized across the four workloads defined in the job.

**Figure 13: Summarization utility workload summary report**

<table>
<thead>
<tr>
<th>WORKLOAD</th>
<th>GROUP</th>
<th>ID#</th>
<th>MIN</th>
<th>INPUT</th>
<th>ACCEPTED</th>
<th>OUTPUT</th>
<th>UPDATED</th>
<th>MAX-FILES</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMRTRANS</td>
<td>1 DLY</td>
<td>1.583,507</td>
<td>2,346</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>IBMTRANS</td>
<td>2 DLY</td>
<td>1.583,507</td>
<td>935,254</td>
<td>4</td>
<td>4</td>
<td>6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ALLTRANS</td>
<td>3 DLY</td>
<td>1.583,507</td>
<td>645,907</td>
<td>4</td>
<td>4</td>
<td>6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ALLTERM</td>
<td>4 DLY</td>
<td>1.583,507</td>
<td>1,471,378</td>
<td>4</td>
<td>4</td>
<td>6</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Input records are summarized daily, and the OUTPUT column shows four records were created for each workload over the four-day interval.

**Description of the report fields**

The report fields shown in the figure above are as follows:
### Table 3: CMRSUMDN workload summary report fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WORKLOAD</td>
<td>workload name assigned to the summarization</td>
</tr>
<tr>
<td>GROUP</td>
<td>summarization group of combined CICS regions</td>
</tr>
<tr>
<td>ID #</td>
<td>internal workload ID assigned to the workload and used to make the key unique</td>
</tr>
<tr>
<td>MIN</td>
<td>number of minutes in the summary</td>
</tr>
<tr>
<td>INPUT</td>
<td>number of records in the input file that meet the date, time, and CICS system criteria</td>
</tr>
<tr>
<td>ACCEPTED</td>
<td>number of records in the input file that were accepted for summarization</td>
</tr>
<tr>
<td>OUTPUT</td>
<td>number of summarized records written</td>
</tr>
<tr>
<td>UPDATED</td>
<td>number of summary records that were updated with input records from this job</td>
</tr>
<tr>
<td>MAX-FILES</td>
<td>number of files accessed by the summarized workload</td>
</tr>
</tbody>
</table>

**Note**

When **MAX-FILES** is greater than 20, it indicates some files were not written in the summarized records.

Additional fields appear in the summary report if the REPLACE=YES or USEREXIT=\*name control statements are included in the job. If REPLACE=YES is included, two fields named **DELETED** and **REPLACED** appear in the report. The **DELETED** field shows the number of records that were deleted because the interval contained no records and ZERODATA=NO was specified. The **REPLACED** field shows the number of records held in the output data set that were replaced by new records.

If USEREXIT=\*name is included in the job, two fields named **INCLUDED** and **EXCLUDED** show the number of records that were included or excluded from the summary workloads due to the test conditions set by the user-written routine.

The **INCLUDED** field shows the number of records that were summarized that normally would have been excluded if the exclude decision had not been overridden by the user-written routine. The **EXCLUDED** field shows the number of records that were excluded from summarization that normally would have been included if the include decision had not been overridden by the user-written routine.
Creating PRL reports from summary data

The following figure shows the JCL and control statements to produce a Performance Reporting Language (PRL) report from the summary workloads created by the summarization utility.

The JCL is from BBSAMP member CMRL, which is generic JCL to run PRL jobs. The JCL must be edited to your site’s conditions before running the job.

The PRL control statements added beneath the SYSIN DD statement are from member CMRPRL28 located in the BBSAMP data set. These control statements produce a PRL report from summarized 6E and 6D records.

The seven control statements shown between the USING and REPORT statements specify the data extracted from the summary records that appears in the PRL report. Each of these seven control statements produces a separate, tabular column in the PRL report.

Figure 14: PRL report JCL and control statements

```plaintext
//USR1PRL JOB (NUMBER), 'USER NAME',
//CLASS=F,
//MSGCLASS=R,
//MSGLEVEL=(1,1),
//NOTIFY=USR1
/*JOBPARM L=99, COPIES=1, R=10, S=SYSB
//PRL EXEC PGM=CMRPRLN, REGION=2048K, TIME=180
/*
// THIS IS SAMPLE JCL TO EXECUTE THE MAINVIEW for CICS PERFORMANCE
// REPORTING LANGUAGE (PRL). CHANGE THE STATEMENTS INDICATED
// TO REFLECT YOUR INSTALLATION.
/*
//STEP1 LIB DD DISP=SHR, DSN=CMR34.CMR1.LOAD
//SYSPRINT DD SYSOUT=*  
//SYSUDUMP DD SYSOUT=* 
//ISYSDATA DD SYSOUT=* 
//ISYPARM DD SYSOUT=*, DCB=LRECL=80, BLKSIZE=160, RECFM=FBA
//CMRFDET1 DD DISP=SHR, DSN=CMR1.TEMP.CMRDET1
//SORTWK02 DD UNIT=SYSDA, SPACE=(CYL,30)
//SORTWK03 DD UNIT=SYSDA, SPACE=(CYL,30)
//SYSIN DD *
*
* THIS IS THE SAMPLE REPORT THAT USES THE SUMMARIZED DATA CREATED BY
* CMRSUMDN.
*
* (YOU MAY NEED TO MODIFY THE REPORT OUTPUT CLASS ON THE NEXT STATEMENT)
SET CLASS = 'R'
SET ZHEADING = 'SINGLE'
*
SET TITLE = 'SUMMARIZED DATA ANALYSIS'
SELECT TYPE 6E AND 6D RECORDS FROM CMRDET1
USING CMRDATE -
CMRTIME -
SUMWKLD -
SUMCICS -
T6ETASK -
T6ERESP -
T6ECPUR
```
The online PRL report shown in the following figure is produced by the job shown in Figure 14 on page 56. The PRL report shows the task count, average response time, and CPU usage of the summarized workloads over four daily periods. Notice that the seven tabular columns of the report appear in the same order as the PRL control statements in the job to produce the report.

**Figure 15: PRL report using summary data**

<table>
<thead>
<tr>
<th>RECORDED DATE</th>
<th>RECORDED TIME</th>
<th>WORKLOAD NAME</th>
<th>CICS GROUP</th>
<th>COUNT OF TASKS IN SUMMARY</th>
<th>RESPONSE TIME</th>
<th>CPU REAL TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>09/12/1998</td>
<td>24:00:00</td>
<td>CMRTRANS</td>
<td>CICSPROD</td>
<td>265</td>
<td>0.171</td>
<td>0.057</td>
</tr>
<tr>
<td>09/12/1998</td>
<td>24:00:00</td>
<td>IBMTRANS</td>
<td>CICSPROD</td>
<td>207831</td>
<td>0.067</td>
<td>0.005</td>
</tr>
<tr>
<td>09/12/1998</td>
<td>24:00:00</td>
<td>ALLTRANS</td>
<td>CICSPROD</td>
<td>9017</td>
<td>0.280</td>
<td>0.015</td>
</tr>
<tr>
<td>09/12/1998</td>
<td>24:00:00</td>
<td>ALLTERM</td>
<td>CICSPROD</td>
<td>215355</td>
<td>0.075</td>
<td>0.006</td>
</tr>
<tr>
<td>09/13/1998</td>
<td>24:00:00</td>
<td>CMRTRANS</td>
<td>CICSPROD</td>
<td>866</td>
<td>17.090</td>
<td>0.067</td>
</tr>
<tr>
<td>09/13/1998</td>
<td>24:00:00</td>
<td>IBMTRANS</td>
<td>CICSPROD</td>
<td>614318</td>
<td>0.206</td>
<td>0.005</td>
</tr>
<tr>
<td>09/13/1998</td>
<td>24:00:00</td>
<td>ALLTRANS</td>
<td>CICSPROD</td>
<td>140551</td>
<td>1.042</td>
<td>0.022</td>
</tr>
<tr>
<td>09/13/1998</td>
<td>24:00:00</td>
<td>ALLTERM</td>
<td>CICSPROD</td>
<td>739681</td>
<td>0.295</td>
<td>0.007</td>
</tr>
<tr>
<td>09/14/1998</td>
<td>24:00:00</td>
<td>CMRTRANS</td>
<td>CICSPROD</td>
<td>226</td>
<td>0.135</td>
<td>0.064</td>
</tr>
<tr>
<td>09/14/1998</td>
<td>24:00:00</td>
<td>IBMTRANS</td>
<td>CICSPROD</td>
<td>16229</td>
<td>0.035</td>
<td>0.002</td>
</tr>
<tr>
<td>09/14/1998</td>
<td>24:00:00</td>
<td>ALLTRANS</td>
<td>CICSPROD</td>
<td>38577</td>
<td>1.192</td>
<td>0.018</td>
</tr>
<tr>
<td>09/14/1998</td>
<td>24:00:00</td>
<td>ALLTERM</td>
<td>CICSPROD</td>
<td>46405</td>
<td>0.434</td>
<td>0.014</td>
</tr>
<tr>
<td>09/15/1998</td>
<td>24:00:00</td>
<td>CMRTRANS</td>
<td>CICSPROD</td>
<td>989</td>
<td>0.156</td>
<td>0.064</td>
</tr>
<tr>
<td>09/15/1998</td>
<td>24:00:00</td>
<td>IBMTRANS</td>
<td>CICSPROD</td>
<td>96876</td>
<td>0.026</td>
<td>0.002</td>
</tr>
<tr>
<td>09/15/1998</td>
<td>24:00:00</td>
<td>ALLTRANS</td>
<td>CICSPROD</td>
<td>457762</td>
<td>1.017</td>
<td>0.016</td>
</tr>
<tr>
<td>09/15/1998</td>
<td>24:00:00</td>
<td>ALLTERM</td>
<td>CICSPROD</td>
<td>469937</td>
<td>0.750</td>
<td>0.015</td>
</tr>
</tbody>
</table>

**Description of the reports fields**

The fields of the PRL report are as follows:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RECORDED DATE</td>
<td>ending date of the summarization interval in <em>mm/dd/yyyy</em> format</td>
</tr>
<tr>
<td>RECORDED TIME</td>
<td>ending time of the summarization interval in <em>hh:mm:ss</em> format</td>
</tr>
<tr>
<td>WORKLOAD NAME</td>
<td>name of the workloads with summarized data</td>
</tr>
<tr>
<td>CICS GROUP</td>
<td>CICS region from which summary records were created</td>
</tr>
<tr>
<td>COUNT OF TASKS IN SUMMARY</td>
<td>number of tasks summarized for this program</td>
</tr>
<tr>
<td>RESPONSE TIME</td>
<td>average internal response time for the summarized tasks</td>
</tr>
<tr>
<td>CPU REAL TIME</td>
<td>average real CPU time for the summarized tasks</td>
</tr>
</tbody>
</table>
Notes on using the summarization utility

The remainder of this chapter gives several hints about using the summarization utility.

Selecting input records

The EXCLxxxx and INCLxxxx control statements are often used together to select CMRDETL records for summarization.

As their names suggest, EXCLxxxx excludes records and INCLxxxx includes records used by the summarization utility. The parameters of both control statements define the CICS resources whose records are selected for summarization.

Both control statements can be specified more than once for each workload defined in the job. The control statements are interpreted in the following manner during summarization processing:

- If more than one INCLxxxx control statement is specified for a single workload, the parameters are logically ORed together. Records are included if the selected resources meet any parameter condition.

- If more than one EXCLxxxx control statement is specified for a single workload, the parameters are logically ORed together. Records are excluded if the selected resources meet any parameter condition.

- INCLxxxx processing occurs first.
  
  If a resource meets a condition specified by the INCLxxxx control statement, the included records are then tested against the conditions set by the EXCLxxxx control statement.

  Records are summarized if they meet any condition set by the INCLxxxx statement and do not meet any condition set by the EXCLxxxx statement.
The following figure shows the logic of selecting records for summarization when the INCLxxxx and EXCLxxxx control statements are included in the summarization job.

**Figure 16: Selecting records for summarization**

Notice the USEREXIT control statement at the bottom of Figure 16 on page 59. You can set additional selection criteria with a user-written routine that tests each record. This routine is given control when a record is processed during the summarization job. In the example shown in Figure 17, the exit to the user-written routine is passed all records that have been tested previously by the INCLxxxx and EXCLxxxx control statements.

The user-exit routine is the final arbiter of including or excluding records from the summarization job. Records that have been previously excluded can be reinserted into the job to be summarized. Likewise, records that have been selected for summarization can be overridden and excluded from the job.

**Detail file switch**

The summarization utility’s detail file switch provides an opportunity to summarize data. By adding a step after the archive is completed, you can summarize the data from the detail data set as needed.
You can summarize each detail record into multiple workloads in a single execution of the program. Each execution can produce as many workloads as required.

When the output file is VSAM, the summarization program combines data from multiple executions; it is not necessary to summarize only at daily intervals, such as midnight.

**Replacing summary data**

The default summarization program can combine data from the current execution with summary data created earlier if you use a VSAM output data set.

Occasionally, you may need to replace an existing workload with new data; for example, you could have omitted a transaction when creating the workload definitions.

To replace a summary workload, specify **REPLACE=YES** when you run the summarization utility. Only workloads and summarization intervals in the current execution are affected by the REPLACE=YES option.

---

**WARNING**

The REPLACE=YES control statement replaces all workload data with duplicate time frames and also deletes any current intervals that no longer contain data.

---

**Viewing summary records**

If summary records are stored on a VSAM data set, they can be viewed with the MainView for CICS HISTORY service.

Intervals can be selected from the data set and expanded to get varying levels of detail about the records that compose the interval.

To view summary records with the HISTORY service:

**To view the summary records**

1. Create summary records and store them on a VSAM data set. SUMMDATA is the default target name to view summary records. Otherwise, use the NEWAPPL control statement to set the key of the records to a name other than the CICS region.

2. Add SUMMDATA to BBPARM member BBIJNT00 (do not add BBIISP00 block requests).
3 Add SUMMDATA to the BBPARM member CMRDTL00 and point DSN1 to the VSAM data set created in Step 1.

An example follows:

```
TARGET=SUMMDATA,DSN1=SUMMARY.VSAM
```

4 Select the HISTORY service from the MainView for CICS Primary Option Menu.

5 Access the summary records by setting the CICS target to SUMMDATA in the upper right corner of the MainView for CICS History Selection panel.

```
CICS ===> SUMMDATA
```

6 Complete the remaining fields of the History Selection panel to set any filters and the time and date intervals of the records that you want to view.

The following figure shows the four workloads of daily summary records created earlier with the summarization utility. You can get more detail about each workload by expanding the HISTORY views.

**Figure 17: HISTORY Service Display of summary records**

<table>
<thead>
<tr>
<th>COMMAND</th>
<th>CICS ===&gt; SUMMDATA</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATE</td>
<td>TIME</td>
</tr>
<tr>
<td>24:00:00</td>
<td>989</td>
</tr>
<tr>
<td>24:00:00</td>
<td>96,876</td>
</tr>
<tr>
<td>24:00:00</td>
<td>457,762</td>
</tr>
<tr>
<td>24:00:00</td>
<td>469,937</td>
</tr>
</tbody>
</table>

The HISTORY display of summary records varies from the standard detail data. The ending time shown in the HISTORY display is the end of the summarization interval, not the ending time of the detail records associated with the interval.

**Storing Summary Data on VSAM Data Sets**

A VSAM data set provides the best repository of summarized data:

- The space required to maintain a month of summarized data is small. The CMRPURG utility of MainView for CICS can be used to maintain the data set, so you can keep as little or as much data available as you require. “Archiving data (CMRPURG)” on page 25 gives instructions for removing unwanted records from the VSAM data set with the CMRPURG utility.

- Summary records can be viewed with the HISTORY service.

- The summarization program combines new data with existing data only when the output is VSAM. This provides a complete interval of data even when different jobs summarized the data. For example, a daily summary record can comprise data collected from several CMRDETL data sets when switching is active.
You can create combined workloads from detail files within multiple CICS regions without first combining the data. Although you can create combined workloads using a sequential data set, data from multiple executions is not combined and the CICS input data must be sorted into date/time sequence.

The requirement to sort input data is removed. However, performance improves when input data is stored in sorted order.

See BBSAMP member CMRDFSUM for guidelines about allocating a VSAM or sequential data set.

**CMRDETL fields altered by the summarization utility**

The summarization utility alters the values of some CMRDETL record fields during summarization.

The following table lists the changes to these fields after they have been summarized:

<table>
<thead>
<tr>
<th>Field name</th>
<th>Contents of a summarized record</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>TYPE</strong></td>
<td>contains 6D, if any input record during the interval is a 6D abend record (see T6EABCD)</td>
</tr>
<tr>
<td><strong>CMR$SYS</strong></td>
<td>assigned the value from the NEWAPPL control statement</td>
</tr>
<tr>
<td><strong>CMRTIME</strong></td>
<td>contains the time at the end of the interval</td>
</tr>
<tr>
<td><strong>CMRCLOCK</strong></td>
<td>contains the name of the region or SGROUP (if multiple CICS regions are combined)</td>
</tr>
<tr>
<td><strong>CMRRECID</strong></td>
<td>contains a value to make each record unique. This record comprises a minute value of one byte followed by a sequential number assigned to each workload. This number is in packed format</td>
</tr>
<tr>
<td><strong>T6ETRTY</strong></td>
<td>contains an X indicating a summary record</td>
</tr>
<tr>
<td><strong>T6EUSER</strong></td>
<td>contains the name of the CICS region or SGROUP, if multiple CICS regions are combined. The alias, SUMCICS, can be used by PRL reports to specify the data held in the <strong>T6EUSER</strong> field. <strong>Note:</strong> CMRCLOCK also sets <strong>T6EUSER</strong> to the name of the CICS region or SGROUP when multiple CICS regions are combined</td>
</tr>
<tr>
<td><strong>T6ETMID</strong></td>
<td>contains the number of minutes in the summarization interval. The alias, SUMINTV, can be used by PRL reports to specify the data held in the <strong>T6ETMID</strong> field.</td>
</tr>
</tbody>
</table>
Field name | Contents of a summarized record
---|---
T6EPGNM | contains the WORKLOAD=\textit{name}
The alias, SUMWKLD, can be used by PRL reports to specify the data held in the T6EPGNM field.
T6ESTIME | contains the start time of the interval or the time from the start of data when it is the first interval for a given execution
T6ETASK | contains the number of transactions in the summary
T6EABCD | contains the number of transactions in the summary that were from type 6D records (abend)
T6EFCNT | contains the number of files in the summary record, based on the options you specify
T6EFN \textit{nnn} | contains the file type if you specify DETAIL=NO or the actual file name if you specify DETAIL=YES
T6EFT \textit{nnn} | contains the average file time for all requests to the file in T6EFN \textit{nnn}
\textbf{Note:} T6EFT \textit{nnn} does not contain the average time of all transactions. It contains the average time of only those transactions that access the file.
T6EFC \textit{nnn} | contains the average file calls for all requests to the file in T6EFN \textit{nnn}
\textbf{Note:} T6EFC \textit{nnn} does not contain the average count for all transactions. It contains the average count for those transactions that accessed the file.

**Binary time and character fields**

When you process CMRDETL records with the summarization utility, several data types of the resulting summary records vary from the original record as a result of summarization.

Summary record fields that contain binary, time, or character data are processed during summarization in the following manner:

**Table 5: Processing summary record fields during summarization**

<table>
<thead>
<tr>
<th>Data type</th>
<th>Summary records</th>
</tr>
</thead>
<tbody>
<tr>
<td>Binary</td>
<td>contains the average value for all tasks during the interval</td>
</tr>
<tr>
<td>Time</td>
<td>contains the average time of all tasks during the interval</td>
</tr>
<tr>
<td>Character</td>
<td>contains asterisks (*), unless all tasks in the interval are the same When all tasks in the interval are the same, the original characters are used.</td>
</tr>
</tbody>
</table>

**Combining summarized records**

The summarization utility can combine summarized records.
When you process summarized records, you can combine records from multiple workloads to create a single, new workload. You can also combine summarized records together to create a longer interval than the original records. For example, you can combine data summarized in intervals of 10 minutes into hourly summary records.

**Note**

When you are processing summarized records, you must select records by time period; for example, if you are combining summarized 10-minute records into hourly records, you must specify **INCLTERM=10**.

After executing the summarization utility, the **T6ETMID** field contains the number of minutes in the summarization interval. This prevents records that overlap the summarization interval from being combined and creating invalid data.

The INCLTERM control statement is explained in “INCLxxxx=list” on page 47.
Resource Tracking and Analysis report

The MainView for CICS resource analysis program (CMRRAPR) creates the Resource Tracking and Analysis report. This report provides information about resource usage by CICS transactions and programs. A report may show summary, detail, or service data for either an individual transaction or program or a combination of transactions or programs.

You select the transactions or programs that appear in the Resource Tracking and Analysis report. You declare how transactions or programs are combined by specifying parameters with the CMRRAPM macro in member CMRURAPM in BBSAMP. After you have created the macro statements, they must be assembled and linked using member CMRASM in BBSAMP to create table CMRRAPM. You can change the name of the table by modifying the MODNAME value in CMRASM, otherwise the default name of CMRRAPM is used.

**Note**
If you change the name of the CMRRAPM table, you can use the TABLE control statement to associate the name CMRRAPR to generate the report. Refer to “Report control statements” on page 68 for a description of the TABLE control statement.

The remaining sections of this chapter describe the macro assembly statements, JCL, and control statements used to produce a Resource Tracking and Analysis report.

Creating a Resource Tracking and Analysis report

To create a Resource Tracking and Analysis report, submit the JCL in member CMRRAP in BBSAMP to execute the CMRRAPR report program and any associated control statements as a batch job.
The sample job stream shown in Figure 18 on page 66 produces the Resource Tracking and Analysis report, shown in Figure 20 on page 69.

**Figure 18: Resource Tracking and Analysis report--input JCL**

```
//jobname JOB user parameters
/*JOBPARM user parameters
//JOBLIB DD DISP=SHR,DSN=CMR.CMRV5.BBLINK
//CMRRAPR EXEC PGM=CMRRAPR
//SYSOUT DD SYSOUT=* 
//SYSPRINT DD SYSOUT=* 
//REPORT DD SYSOUT=* 
//CMRDETL DD DISP=SHR,DSN=CMR.CMRV5.CMRDETL 
//SORTWK01 DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSIN DD *
```

The required JCL to produce this report consists of the following statements:

**Table 6: Required JCL for Resource Tracking and Analysis report**

<table>
<thead>
<tr>
<th>JCL</th>
<th>Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>JOB</td>
<td>initiates the job</td>
</tr>
<tr>
<td>EXEC</td>
<td>specifies the name of the program that generates the report (PGM=CMRRAPR)</td>
</tr>
<tr>
<td>//JOBLIB</td>
<td>defines the program library containing the MainView for CICS load modules (BBLINK)</td>
</tr>
<tr>
<td>//CMRDETL</td>
<td>defines a VSAM disk or tape data set containing records for each CICS transaction (CMRDETL)</td>
</tr>
<tr>
<td></td>
<td>You can also use //CMRDETL.nn statements to define additional VSAM disk or tape data sets, where nn is 01 to 99 and must be specified as consecutive numbers.</td>
</tr>
<tr>
<td>//REPORT</td>
<td>defines the report output for the predefined reports</td>
</tr>
<tr>
<td>//SORTWKnn</td>
<td>identifies one to nine work data sets that can be defined for CMRRAPR data sorting</td>
</tr>
<tr>
<td></td>
<td>nn is a numeric value (01-09).</td>
</tr>
<tr>
<td>//SYSPRINT</td>
<td>defines the output class for a sort utility</td>
</tr>
<tr>
<td></td>
<td>The name of the DD statement is determined at CICS system generation.</td>
</tr>
<tr>
<td>//TAPEIN</td>
<td>defines the archive data set produced by CMRPURG, described in “Archiving data (CMRPURG)” on page 25, as input to the report program</td>
</tr>
<tr>
<td>//SYSOUT</td>
<td>defines the output class</td>
</tr>
<tr>
<td>//SYSIN</td>
<td>defines PERFORMANCE REPORTER control statements as input to the batch program</td>
</tr>
</tbody>
</table>

---

Creating a Resource Tracking and Analysis report

MainView for CICS PERFORMANCE REPORTER User Guide
Sample CMRRAPM macro assembly statements

Figure 19 on page 67 shows an example of BBSAMP (CMRRAPM) macro assembly statements that are needed to create the Resource Tracking and Analysis report.

Figure 19: CMRRAPM macro assembly statements

```
CMRRAPM TYPE=INITIAL
CMRRAPM TYPE=ENTRY,GROUP=xxxxxxxxxxxx,
     MEMBERS=(xxxxxxxx,...,xxxxxxxx), SERVICE=nnn
     (Additional TYPE=ENTRY statements, as needed)
CMRRAPM TYPE=FINAL
END
```

The following table describes each CMRRAPM macro assembly statement:

Table 7: Description of the CMRRAPM macro assembly statements

<table>
<thead>
<tr>
<th>Statement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TYPE=INITIAL</td>
<td>generates a group of MainView for CICS modules</td>
</tr>
<tr>
<td></td>
<td>This statement must be entered first and placed on a separate line by itself. Specify CMR=YES to create a group that includes all MainView for CICS transaction IDs (the default); otherwise, specify CMR=NO.</td>
</tr>
<tr>
<td>TYPE=ENTRY</td>
<td>defines one group of transactions or programs and the response time to be compared with the specified service level</td>
</tr>
<tr>
<td></td>
<td>One TYPE=ENTRY statement must be specified for each group to be reported, where:</td>
</tr>
<tr>
<td></td>
<td>■ GROUP= is any 1- to 12-character symbolic identifier for this group.</td>
</tr>
<tr>
<td></td>
<td>■ MEMBERS= is the list of transaction IDs or program names (PANL, MAPS, or PSBS, if they have been monitored with the transaction monitor) to be included in this group. A skip character is a plus sign (+), which can be used in IDs or names to indicate generic subgroups of transactions or programs to be included. There is no limit on the number of IDs or names specified. However, a given ID or name can exist in only one group.</td>
</tr>
<tr>
<td></td>
<td>■ SERVICE= is a service level value to which the response time for each transaction or program is to be compared, expressed in tenths of seconds. The default is 10 (1.0 seconds).</td>
</tr>
</tbody>
</table>
TYPE=FINAL ends the generation parameters
It must be the last statement and must be placed on a line by itself.

Note: BBSAMP member CMRASM contains sample code to assemble and link-edit the CMRRAPM module. CMRRAPM must be linked non-reentrant.

Report control statements

All control statements are optional.

If specified, each statement must begin in column 1. Table 8 on page 68 describes each control statement.

Table 8: Report control statements

<table>
<thead>
<tr>
<th>Statement</th>
<th>Description</th>
</tr>
</thead>
</table>
| FROM DATE   | starting date of the data to be included in the report, in \textit{mm/dd/yyyy} format
If the statement is omitted, no data is rejected as too old.

\textbf{Note}: If you specify a date in the old format (\textit{mm/dd/yy}), \textit{yy} is interpreted as \textit{19yy}.

| TO DATE     | ending date of the data to be included in the report, in \textit{mm/dd/yyyy} format
If the statement is omitted, no data is rejected as too new.

\textbf{Note}: If you specify a date in the old format (\textit{mm/dd/yy}), \textit{yy} is interpreted as \textit{19yy}.

| FROM TIME   | starting time of the data to be included in the report, in \textit{hh:mm:ss} format
If the statement is omitted, no data is rejected as too early

| TO TIME     | ending time of the data to be included in the report, in \textit{hh:mm:ss} format
If the statement is omitted, no data is rejected as too late.

| SORTWK      | number of sort work areas to be used
Valid values are 1 to 8. If a value greater than 1 is specified, a corresponding number of DD or DLBL statements must appear in the JCL.
INPUT

- type of input data set to be used and can be specified as one of the following values:
  - VSAM (VSAM data set—the default)
  - TAPE (sequential tape or disk file)

TYPE

- type of report to be generated, which can be any one of the following values:
  - SUMMARY reports three lines of transaction execution data for each transaction or program by calendar date. This value is the default. A sample of this type of report is in Figure 20 on page 69.
  - DETAIL reports three lines of transaction execution data for each transaction or program.
  - SERVICE specifies DETAIL information, but only for transactions or programs that did not meet their service levels.

TABLE

- module name of the customized CMRRAPM table
  The default is CMRRAPM.

---

Sample report

An example of the Transaction Resource Tracking and Analysis report is shown in Figure 20 on page 69. The report is created from the JCL shown in Figure 18 on page 66.

Figure 20: Transaction Resource Tracking and Analysis report

---

Chapter 4 Resource Tracking and Analysis report 69
### Report structure and conventions

The report shown in Figure 20 on page 69 is produced by the CMRRAPR program.

The report run type, SUMMARY, is printed in the top left corner on line 3 below the date and time fields. A summary report shows a summary of transaction performance in chronological order. A subtotal by transaction ID is marked with asterisks (******) in the DATE column before the next transaction begins. Five transaction IDs are included in this report: DRGI, EMNU, FINC, PROV, and RELG.

A detail report shows performance data for every transaction. A service report is also a type of detail report, except it is limited to transactions or programs that did not meet their service levels. Otherwise, the format of both reports is the same as the summary report shown in Figure 20 on page 69.

G:PAYROLL is the name of the group formed by programs DRGI, EMNU, FINC, PROV, and RELG. Values in this row represent the sum of the subtotals reported for DRGI, EMNU, FINC, PROV, and RELG.
The areas marked ** SELECTED ** and * UNSELECTED * show totals for programs whose IDs were either matched ( SELECTED ) or not matched ( UNSELECTED ) against the CMRRAPM table. Both program IDs in this report were matched in the table:

- The SELECTED row shows values equal to the combined data for programs DRGI, EMNU, FINC, PROV, and RELG in the G:PAYROLL row.

- The UNSELECTED row shows totals for programs other than the selected programs.

The GRAND TOTALS area shows total counts and percentages for all reported programs. In this report, the GRAND TOTALS field shows the sum of all listed values ( SELECTED and UNSELECTED ).

The TRANSACTIONS PER SECOND field at the lower left of the report is the average rate calculated for all transactions that occurred during the period that data was collected for the report. This example represents 15.5 hours; thus, the transactions per second are

\[
\frac{62}{(15.5 \times 60 \times 60)}, \text{ or } 0.001.
\]

**Note**

Any count that exceeds the width of its report field is expressed in millions, indicated by an M suffix. A time that is longer than its report field is forced to a value of 999.999 on the report. The actual time is retained, however, and all values derived from it are maintained correctly.

Column headings in this report are standard; they are described in the following section.

**Field descriptions**

Table 9 on page 72 describes the applicable report fields:
### Table 9: Report fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| **TASK/PGM ID**  | transaction ID or program name being reported  
This field can contain any of the following entries:  
- a 4-character transaction ID (unless the selection matches the program name)  
- a 1- to 12-character identifier assigned to this group of transactions or programs  
  This ID is used when subtotals for a group are summarized.  
- **SELECTED**, indicating a group of transaction or program IDs that were matched to the CMRRAPM table  
  This entry is used when a group break is processed.  
- *UNSELECTED*, indicating a group of transactions or program IDs that were not matched to the CMRRAPM table  
  This entry is used when a group break is processed.  
- GRAND TOTALS, showing a total for the entire report |
| **DATE**         | in a summary report, the Gregorian date of the reported data (with a date format of mm/dd/yyyy)  
in detail or service reports, the following format:  
- Line 1 shows the Gregorian date of the reported activity.  
- Line 2 shows the transaction ending time, in hh:mm:ss format.  
- Line 3 shows the operator identifier that invoked the task or program. |
<p>| <strong>TOTAL NO. OF TASKS</strong> | total number of tasks that reference this task/program |
| <strong>AVG. RESP.</strong>   | average internal response time for this task/program, in seconds |
| <strong>AVG. CPU</strong>     | average elapsed time in seconds that the task was dispatched during each use of this task/program |
| <strong>AVG. CPU REAL</strong>| average actual CPU time required to execute this task/program, in seconds |
| <strong>AVG. TASK WAIT</strong>| average wait time of this task/program, in seconds |
| <strong>AVG. I/O TIME</strong>| average amount of time required to complete an I/O operation, in seconds (This value is the I/O service time experienced by this task/program.) |</p>
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AVG. NO. OF I/O</td>
<td>average number of file, journal, temporary storage, and DL/I I/Os issued during the execution of this task/program</td>
</tr>
</tbody>
</table>
| AVG. FC/DLI CALLS  | on line 1, shows the average number of file control calls issued during each execution of this task/program  
|                    | on line 2, shows the average number of DL/I calls issued during each execution of this task/program                                            |
| CICS/RESP. LOAD     | on line 1, shows the percentage of CICS CPU used by this task/program  
|                    | on line 2, shows the percentage of the total CICS response time used by this task/program                                                    |
| SERVICE LEVEL/PCT/CNT | on line 1, shows the service level of this task/program  
|                     | on line 2, shows the percentage of transactions whose response time met this service level time value  
|                     | on line 3, shows the number of transactions whose response time met this service level time value.                                        |
| FCP WAIT            | on line 1, shows the percentage of this task/program’s response time that can be attributed to file control access requests  
|                     | on line 2, shows the average amount of time attributed to file control access requests, in seconds  
|                     | on line 3, shows the total number of file control access requests issued                                                                  |
| TSP WAIT            | on line 1, shows the percentage of this task/program’s response time attributed to temporary storage access requests  
|                     | on line 2, shows the average amount of time attributed to temporary storage access requests, in seconds  
|                     | on line 3, shows the total number of temporary storage access requests issued                                                               |
| DLI WAIT            | on line 1, shows the percentage of this task/program’s response time attributed to DL/I access requests  
|                     | on line 2, shows the average amount of time attributed to DL/I access requests, in seconds  
|                     | on line 3, shows the total number of DL/I access requests issued                                                                             |
| JCP WAIT            | on line 1, shows the percentage of this task/program’s response time attributed to journal control access requests  
|                     | on line 2, shows the average amount of time attributed to journal control access requests, in seconds  
|                     | on line 3, shows the total number of journal control access requests issued                                                                  |
| ECB WAIT            | on line 1, shows the percentage of this task/program’s response time attributed to user event control block (ECB) wait requests  
|                     | on line 2, shows the average amount of time attributed to ECB wait requests, in seconds  
|                     | on line 3, shows the total number of ECB wait requests issued                                                                               |
Graphic histogram reports

The CMRCHRT member of your BBSAMP data set contains sample JCL and control statements to create histograms from data collected by MainView for CICS. These reports graphically represent selected CICS resources as horizontal histograms.

Over 260 data elements recorded by MainView for CICS can be combined to create tabular historical reports and histograms. Using control statements, you select the resources and the type of processing shown in your histograms.

Creating a histogram report

The CMRCHRT batch program creates a histogram report.

Both the JCL and control statements to run the CMRCHRT program are described in the following sections.

The sample job shown in Figure 21 on page 75 produces a histogram report for all versions of CICS.

Figure 21: Example JCL to create histogram reports

```
//jobname JOB user parameters
/*JOBPARM user parameters
//CMRCHRT EXEC PGM=CMRCHRT
//JOBLIB DD DISP=SHR,DSN=CMR.Vnn.BBLINK
//CMRDETL DD DISP=SHR,DSN=CMR.Vnn.CMRDETL
//CMRSTAT DD DISP=SHR,DSN=CMR.Vnn.CMRSTATS
//TAPEIN DD DISP=SHR,DSN=CMR.Vnn.CMRSTATS
//REPORT DD SYSOUT=*  
//SORTWK01 DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSPRINT DD SYSOUT=*  
//SYSOUT DD SYSOUT=*  
//SYSIN DD *
  (control statements)
```

The JCL to execute the CMRCHRT program consists of the following statements:
### Table 10:

<table>
<thead>
<tr>
<th>JCL Statement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>JOB</td>
<td>initiates the job</td>
</tr>
<tr>
<td>EXEC</td>
<td>specifies the name of the program (PGM=CMRCHRT) that generates the report</td>
</tr>
<tr>
<td>//JOBLIB</td>
<td>defines the program library containing the MainView for CICS load modules (BBLINK)</td>
</tr>
<tr>
<td>//CMRDETL</td>
<td>defines a VSAM disk or tape data set containing T6E detail transaction records</td>
</tr>
<tr>
<td></td>
<td>You can also use //CMRDETLnn statements to define additional VSAM disk or tape data sets, where nn is 01 to 99 and must be specified as consecutive numbers.</td>
</tr>
<tr>
<td>//CMRSTAT</td>
<td>defines the VSAM data set or sequential file containing records of CICS operation statistics (CMRSTATS)</td>
</tr>
<tr>
<td>//TAPEIN</td>
<td>defines the sequential disk or tape file containing records of CICS operation statistics (CMRSTATS)</td>
</tr>
<tr>
<td>//REPORT</td>
<td>defines the report output for the predefined reports</td>
</tr>
<tr>
<td>//SORTWKnn</td>
<td>identifies one to nine work data sets that can be defined for CMRCHRT data sorting</td>
</tr>
<tr>
<td></td>
<td>Define nn as a numeric value from 01 to 09.</td>
</tr>
<tr>
<td>//SYSPRINT</td>
<td>defines the output class for a sort utility</td>
</tr>
<tr>
<td></td>
<td>The name of the DD statement is determined at CICS system generation.</td>
</tr>
<tr>
<td>//SYSOUT</td>
<td>defines the output class</td>
</tr>
<tr>
<td>//SYSIN</td>
<td>defines PERFORMANCE REPORTER control statements as input to the batch program</td>
</tr>
</tbody>
</table>

CMRCHRT runs with the following restrictions:

- Data must be created from the SMF records selected by using CMRSTATS options as follows:
  - GAPPLID or SAPPLID must be used to segregate SMF data by CICS regions into separate files. CMRCHRT cannot select by APPLID.

- Only three class types of CMRCHRT can be run:
  - CLASS=CICS
  - CLASS=APPLICATION
Control statements

Control statements identify the CICS resources and associated performance statistics that appear in histogram reports.

A control statement set is a group of control statement keywords and parameters that specify one histogram report. A set can contain valid keywords in any sequence, but each keyword can be specified only once in each set.

A histogram control statement set is structured as follows:

Table 11: Histogram control statement set structure

<table>
<thead>
<tr>
<th>Structure</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>keyword=value</td>
<td><em>(optional)</em> sets up date and time ranges, titles, graph scales, and so on</td>
</tr>
</tbody>
</table>
|                 | for this report
|                 | Defaults are used if these keywords are not specified.                     |
| class           | *(one required)* specifies the CICS resource appearing in the histogram    |
| data=type       | *(one required)* specifies the type of data appearing in the histogram      |
| END             | *(required)* ends this report request                                       |

Control statements follow these rules:

- Each control statement must begin in column 1.

- If a control statement keyword appears more than once in a set, only its last specification is used.

- If a control statement keyword is not specified in a set, its default is used.

- Multiple control statement sets can be specified in a single execution of the CMRCHRT program to produce multiple histograms. However, separate jobs must be submitted to produce multiple histograms if report data is on tape (see “Archiving data (CMRPURG)” on page 25).

- Comments can be added to control statements by leaving at least one blank after the statement and then starting the comment.
Any statement with an asterisk (*) in column 1 is considered to be a comment and is ignored.

Keyword=value statements

The following keywords specify the range of data, titles, calculation methods, and scales of each histogram report. If any keyword is omitted, the default is used.

TITLE

TITLE specifies the title of the report; up to 60 characters are allowed, including blanks. The default is 60 blanks.

FROM DATE and TO DATE

FROM DATE and TO DATE specify the date range (in \textit{mm/dd/yyyy} format) to select input data. If used, these keywords must be respecified in every control statement set. The defaults are FROM DATE=09/09/0000 and TO DATE=12/31/9999; data from all available dates appears in the histograms.

\textbf{Note}

If you specify a date in the old format (\textit{mm/dd/yy}), \textit{yy} is interpreted as 19\textit{yy}.

FROM TIME and TO TIME

FROM TIME and TO TIME specify the daily time range (in \textit{hh:mm:ss} format) to select input data. If used, these keywords must be respecified in every control statement set. The defaults are FROM TIME=00:00:00 and TO TIME=24:00:00; data from all available times appears in the histograms.

MINUTES

MINUTES represents the length of time the requested file data is to be accumulated, in minutes. The accumulated data is used to create a line of report data. Any value from 1 through 99999 is valid; the default is 1 (minute).

The keywords MINUTES and \texttt{APPL=(ALL)} are mutually exclusive. For more information about \texttt{APPL=(ALL)}, see “Applications class” on page 81.

BREAK={DAILY | HOURLY | MONTHLY}

BREAK divides the date and time ranges into fixed report intervals.
The keywords BREAK= and APPL=(ALL) are mutually exclusive. For more information on APPL=(ALL), see “Applications class” on page 81. Valid values for BREAK are as follows:

- DAILY, the default, specifies that each day’s data starts on a new report page.
- HOURLY specifies that each hour’s data starts on a new report page.
- MONTHLY specifies that each month’s data starts on a new report page.

\[ \text{CALC} = \{ \text{PERCENT} \mid \text{AVERAGE} \mid \text{TOTAL} \} \]

CALC specifies the calculation method that is used to present the data. Valid values are as follows:

- PERCENT, the default, reports percentages.
- AVERAGE reports averages.
- TOTAL reports total counts.

**Note**
The CALC keyword is ignored if you specify DATA=PAGING.

\[ \text{SCALE} = \{ 100 \mid nnnnn \mid \text{FLOAT} \} \]

SCALE defines histogram scale values. Valid values are as follows:

<table>
<thead>
<tr>
<th>Values</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>calculates the scale to accommodate the largest reported value</td>
</tr>
<tr>
<td>nnnnn</td>
<td>specifies the highest scale value&lt;br&gt;Valid scalar numbers include&lt;br&gt;</td>
</tr>
<tr>
<td></td>
<td>■ from 1 to 99999 for counts&lt;br&gt; ■ up to 100 for percent&lt;br&gt; ■ up to 68700 for time; the default scale is 100&lt;br&gt; Anything beyond the maximum range is truncated.</td>
</tr>
<tr>
<td>FLOAT</td>
<td>calculates the scale to accommodate the largest reported value</td>
</tr>
</tbody>
</table>
COMPANY=\textit{name}

COMPANY specifies the company name to be displayed at the top of the histogram. The default is BMC Software.

When specified, this parameter remains in effect for the entire program execution or until another COMPANY keyword is encountered.

INPUT=\{VSAM | TAPE\}

INPUT identifies the organization of the input data set. Valid values are as follows:

- VSAM (VSAM data set—the default)
- TAPE (sequential tape or disk file)

When specified, this parameter remains in effect either for the entire program execution or until another INPUT keyword is encountered.

\underline{Note}

If INPUT=TAPE is specified, you must specify a TAPEIN DD statement in your JCL.

TABLE=\textit{name}

TABLE specifies the module name of the modified CMRRAPM table to be used by the Applications Class GROUP parameter. TABLE must be specified before the associated GROUP parameter. The default is CMRRAPM. For more information, see “Applications class” on page 81.

When specified, this parameter remains in effect either for the entire program execution or until another TABLE keyword is encountered.

ENDDATA=\{DATE | FILE\}

ENDDATA specifies whether end of data should correspond with the TO DATE (ENDDATA=DATE) or end of file (ENDDATA=FILE) keyword. The default is DATE.

When specified, this parameter remains in effect for the entire execution of the program or until another ENDDATA keyword is encountered.

ENDDATA is required if input data is

- selected from an archive tape that contains data merged from multiple CICS regions (see “Archiving data (CMRPURG)” on page 25)
CLASS and DATA control statements

One CLASS and one DATA statement are required for each histogram report.

The CLASS control statement identifies the resource to be reported; the DATA statement identifies the data to be graphed for that class. Each CLASS statement has its own set of data types, and only one DATA type can be selected for each histogram report.

Applications class

The Applications Class control statements identify a specific type of application or group of applications that appear in histogram reports.

The format is as follows:

\[
\text{APPL} = \{ \text{name} | (\text{ALL}) \}
\]

where

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>APPL</td>
<td>is any of the following applications:</td>
</tr>
<tr>
<td></td>
<td>■ TRAN (transaction name)</td>
</tr>
<tr>
<td></td>
<td>■ PROG (program name)</td>
</tr>
<tr>
<td></td>
<td>■ TERM (terminal name)</td>
</tr>
<tr>
<td></td>
<td>■ PANL (DMS panel name)</td>
</tr>
<tr>
<td></td>
<td>■ MAPS= Storage map name</td>
</tr>
<tr>
<td></td>
<td>■ PSBS= PSB name</td>
</tr>
<tr>
<td></td>
<td>■ GROUP= A group of transactions or programs defined in the CMRRAPM table specified in the TABLE= parameter</td>
</tr>
<tr>
<td>name</td>
<td>specifies an application name</td>
</tr>
<tr>
<td>(ALL)</td>
<td>requests all the members in the specified class</td>
</tr>
</tbody>
</table>

Note: If you specify ALL, it must be enclosed in parentheses.
The keywords MINUTES, APPL=(ALL), and BREAK are mutually exclusive. For further information about these keywords, refer to "Keyword=value statements" on page 78.

One of the following DATA statements must follow the APPL statement, beginning in column 1:

Table 12:

<table>
<thead>
<tr>
<th>Data statement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATA=RESPONSE</td>
<td>response time</td>
</tr>
<tr>
<td>DATA=ALL-WAIT</td>
<td>total transaction wait time</td>
</tr>
<tr>
<td>DATA=CPU</td>
<td>CPU time</td>
</tr>
<tr>
<td>DATA=FC-WAIT</td>
<td>file I/O wait time</td>
</tr>
<tr>
<td>DATA=FC-IOCNT</td>
<td>number of file I/Os</td>
</tr>
<tr>
<td>DATA=TC-WAIT</td>
<td>terminal wait time</td>
</tr>
<tr>
<td>DATA=TC-IOCNT</td>
<td>number of terminal I/Os</td>
</tr>
<tr>
<td>DATA=JC-WAIT</td>
<td>journal I/O wait time</td>
</tr>
<tr>
<td>DATA=JC-IOCNT</td>
<td>number of journal I/Os</td>
</tr>
<tr>
<td>DATA=TS-WAIT</td>
<td>temporary storage I/O wait</td>
</tr>
<tr>
<td>DATA=TS-IOCNT</td>
<td>number of temporary storage I/Os</td>
</tr>
<tr>
<td>DATA=ECB-WAIT</td>
<td>user event wait time</td>
</tr>
<tr>
<td>DATA=ECB-WCNT</td>
<td>number of user event wait requests</td>
</tr>
<tr>
<td>DATA=SUSPENDT</td>
<td>suspend wait time</td>
</tr>
<tr>
<td>DATA=SUSPENDC</td>
<td>number of times suspended</td>
</tr>
<tr>
<td>DATA=FCPCALLS</td>
<td>number of file control (FCP) calls or requests</td>
</tr>
<tr>
<td>DATA=DLICALLS</td>
<td>number of DL/I calls</td>
</tr>
<tr>
<td>DATA=DL/IWAIT</td>
<td>DL/I I/O wait time</td>
</tr>
<tr>
<td>DATA=DL/IOCNT</td>
<td>number of DL/I/Os</td>
</tr>
</tbody>
</table>

**Service levels class**

The Service Level Class control statements report about service level performance.

The format is

```
SERVICE
```
No DATA control statements are used with the Service Level Class control statement.

**CICS system class**

CICS system information is reported by using system data journaled by CICS. The format is

CICS

One of the following DATA control statements must follow the CICS statement, beginning in column 1:

- DATA=CICSCPU (total CICS CPU time and user CPU time)
- DATA=KCP-CPU (task control CPU time)
- DATA=KCP-CPUR (task control CPU real time)
- DATA=KCP-CPUC (task control dispatch count)
- DATA=TCP-CPU (terminal control CPU time)
- DATA=TCP-CPUR (terminal control CPU real time)
- DATA=TCP-CPUC (terminal control dispatch count)
- DATA=JCP-CPU (journal control CPU time)
- DATA=JCP-CPUR (journal control CPU real time)
- DATA=JCP-CPUC (journal control dispatch count)
- DATA=USERCPU (user CPU time)
- DATA=USERCPUR (user CPU real time)
- DATA=USERCPUC (user dispatch count)
- DATA=PGIN (CICS page-in operations)
- DATA=PGOT (CICS page-out operations)
- DATA=PAGING (CICS page-in and page-out rate per second)
- DATA=TASKRATE (transaction rate per second)
### Processing program table class

The Processing Program Table reports about a specific PPT entry or all PPT entries.

The format is

```
PPT={xxxxxxx | (ALL)}
```

where xxxxxx is a valid PPT entry name or (ALL), which reports on all the entries in the PPT.

One of the following DATA statements must follow the PPT statement, beginning in column 1:

- DATA=USECNT (number of times a program was used)
- DATA=FETCH (number of times a program was fetched from the OS/MVS load library)

**Note**

If the CALC keyword is specified as CALC=PERCENT and DATA=FETCH is specified here, a FETCH-PERCENT report is produced.

### Program Control Table (PCT) class

The Program Control Table Class control statement reports about a specific PCT entry or all PCT entries.

The format is

```
PCT={xxxx | (ALL)}
```

where xxxx is a valid PCT entry name or (ALL), which reports on all the entries in the PCT.

One of the following DATA statements must follow the PCT statement, beginning in column 1:

- DATA=USECNT (transaction usage count)
- DATA=STGVIOL (number of storage violations)
- DATA=ADDSTG (number of additional storage requests)
Note
If the CALC keyword is specified as CALC=PERCENT and DATA=ADDSTG is specified here, the report shows the percentage of time additional storage was needed.

File Control Table class

The File Control Table Class control statement reports about a specific FCT entry or all entries.

The format is

\[ \text{FCT=xxxxxxx | (ALL)} \]

where xxxxxxx is a valid FCT entry name. ALL reports on all the entries in the FCT.

One of the following DATA statements must follow the FCT statement, beginning in column 1:

- DATA=READ (file READ requests)
- DATA=READNEXT (file READNEXT requests (BROWSE))
- DATA=READ-UPD (file READ FOR UPDATE requests)
- DATA=UPDATE (file UPDATE requests)
- DATA=INSERT (file INSERT (add) requests)
- DATA=DELETE (file DELETE requests)
- DATA=DATAEXCP (data component EXCP count)
- DATA=INDXEXCP (index component EXCP count)
- DATA=CI-SPLIT (count of control interval (CI) splits)
- DATA=CA-SPLIT (count of control area (CA) splits)

Note
The CALC=PERCENT statement uses file control requests as a divisor. For example, using DATA=DATAEXCP produces a histogram charting the percentage of physical I/O requests in relation to logical I/O requests.
Common System Area class

The Common System Area Class control statement reports about CSA data.

The format is

```
CSA
```

One of the following DATA statements must follow the CSA statement, beginning in column 1:

- `DATA=GETMAINS (count of GETMAIN requests)`
- `DATA=FREEMAIN (count of FREEMAIN requests)`
- `DATA=SOSCNT (times CICS went short-on-storage (SOS))`
- `DATA=DUMPS (number of transaction dumps)`
- `DATA=ASRA (number of program checks)`
- `DATA=STGWAIT (number of times transactions waited for storage)`
- `DATA=MAXCNT (number of times CICS maximum task condition was reached)`
- `DATA=STGVIOL (number of storage violations)`
- `DATA=DTBLOGS (number of dynamic transaction log records written)`
- `DATA=DTBSPILL (number of DTB log records that spilled to temporary auxiliary storage (AUX))`

**Note**
The CALC=PERCENT statement uses the total CICS transaction count as a divisor.

Transaction Class Table class

The Transaction Class Table Class control statements report about a transaction class or all transaction classes.

The format is

```
CSO={xx | (ALL)}
```

where `xx` is any transaction class ID from 1 through 10 or (ALL), which reports on all transaction classes.
One of the following DATA statements must follow the CSO statement, beginning in column 1:

- DATA=MAXCNT (maximum setting for the class)
- DATA=HWMCNT (high-water mark reached)

**Destination Control Table class**

The Destination Control Table Class control statements report about one destination ID or all destination IDs.

The format is

\[ \text{DCT=}\{\text{xxxx } | \text{(ALL)}\} \]

where xxxx is any destination ID or (ALL), which reports on all destination IDs.

The only valid DATA statement allowed following the DCT statement is as follows:

DATA=USECNT (number of times the destination was used)

**Journal Control Table class**

The Journal Control Table Class control statement reports about one journal number or all journal numbers.

The format is

\[ \text{JCT=}\{\text{xx } | \text{(ALL)}\} \]

where xx is any journal number from 1 through 99 or (ALL), which reports on all journal numbers.

One of the following DATA statements must follow the JCT statement, beginning in column 1:

- DATA=RECCNT (number of records written)
- DATA=BLKCNT (number of blocks written)
- DATA=FULLCNT (number of times buffer was full)
- DATA=SHIFTCNT (number of buffer shift-ups)
- DATA=AVEBUF (average block size written)
DATA=MAXBUF (maximum block size allowed)

**Note**

If the CALC keyword is set to CALC=PERCENT and DATA=RECCNT is specified here, RECCNT is used as the divisor.
If CALC=PERCENT and AVEBUF or MAXBUF are specified here, MAXBUF is used as the divisor.

---

**DL/I class**

DL/I Class control statements report about DL/I data.

The format is

```
DLZ
```

One of the following DATA statements must follow the DLZ statement, beginning in column 1:

- DATA=PSBCNT (number of PSBs in the system)
- DATA=PSTCNT (number of PSTs in the system)
- DATA=DBDCNT (number of DBDs in the system)
- DATA=SUSPENDS (number of suspended tasks)
- DATA=SCHEDULE (number of scheduling requests)
- DATA=MAXCNT (maximum task count)
- DATA=DEADLOCK (number of deadlock occurrences)
- DATA=DUPPSBS (number of duplicate PSBs created)

**Note**

These fields use total DL/I transaction count as a divisor when CALC=PERCENT is specified.

- DATA=READS (number of READ requests)
- DATA=READBUF (number of READs satisfied by data in the buffer pools)
- DATA=EXCPIN (number of input EXCP commands)
- DATA=EXCPOUT (number of output EXCP commands)
DATA=ALTERS (number of buffer alterations)

DATA=IOERRORS (number of I/O errors)

**Note**
These fields use total buffer requests as a divisor when CALC=PERCENT is specified.

DATA=GETUNIQ (number of GET UNIQUE type requests)

DATA=GETNEXT (number of GET NEXT type requests)

**Note**
These fields use total user DL/I calls as a divisor when CALC=PERCENT is specified.

---

**Terminal Control Table class**

The Terminal Control Table Class control statement reports about one terminal or all terminals.

The format is

\[
\text{TCT} = \{ \text{xxxx} \mid (\text{ALL}) \}
\]

where \text{xxxx} is any terminal ID or (ALL), which reports on all terminal IDs.

One of the following DATA statements must follow the TCT statement, beginning in column 1:

- DATA=INPUTS (number of times the terminal communicates with CICS)
- DATA=OUTPUTS (number of times CICS communicates with the terminal)
- DATA=TRANSCNT (number of transactions)
- DATA=TERMERRS (number of terminal failures, such as a PROG470 abend)
- DATA=OPERERRS (number of application program failures)

**Note**
These fields use TRANSCCNT as a divisor when CALC=PERCENT is specified.
Temporary Storage Table class

The Temporary Storage Table Class control statement specifies a histogram report that shows the usage of temporary storage data. The format is

TST

One of the following DATA statements must follow the TST statement, beginning in column 1:

- DATA=PUTSS (number of PUT requests (MAIN or AUX))
- DATA=PUTSQ (number of PUTQ requests (MAIN or AUX))
- DATA=PUTSA (number of PUT or PUTQ requests (AUX))
- DATA=IOERROR (number of I/O errors)
- DATA=EXTENDS (number of TSGID extensions)
- DATA=SUSPENDS (number of suspensions)
- DATA=COMPRESS (number of compressions)
- DATA=AVAILCI (number of available control intervals)
- DATA=TOTALCI (total control intervals in data set)
- DATA=VIRTHWM (virtual storage use high-water mark)

Note

All of these fields except AVAILCI use the total temporary storage requests as a divisor when CALC=PERCENT is specified; AVAILCI uses TOTALCI as the divisor.

END statement

The END statement completes a control statement set for a requested histogram report.

Control statements for another histogram report can follow an END statement.

An END statement is required for every requested histogram report specified with a control statement set.
Sample reports

Table 13 on page 91 lists some sample histogram reports that can be produced with the CMRCHRT program. These reports offer a broad overview of CICS performance and are recommended for tuning purposes. The remainder of this section contains descriptions and illustrations of the sample reports.

Table 13: Sample histogram reports

<table>
<thead>
<tr>
<th>Sample report title</th>
<th>Sample control statements</th>
<th>Section reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Response Time for All Transactions</td>
<td>TRAN=(ALL)DATA=RESPONSECALC=AVERAGESCALE=10END</td>
<td>“Response time for all transactions” on page 91</td>
</tr>
<tr>
<td>CPU Used by CICS Execution</td>
<td>CICSINPUT=TAPEDATA=CICSCPUCALC=PERCENTMINUTES=60FROM TIME=08:30:00TO TIME=17:00:00END</td>
<td>“CPU used by CICS execution” on page 92</td>
</tr>
<tr>
<td>CICS Paging Rate/Second</td>
<td>CICSINPUT=TAPEDATA=PAGINGMINUTES=60FROM DATE=3/24/2000TO DATE=3/24/2000SCALE=10END</td>
<td>“CICS paging rate/second” on page 93</td>
</tr>
<tr>
<td>CICS Transaction Rate/Second</td>
<td>CICSINPUT=TAPEDATA=TASKRA TECALC=AVERAGEMINUTES=60SCALE=10END</td>
<td>“CICS transaction rate/second” on page 94</td>
</tr>
</tbody>
</table>

Response time for all transactions

This report shows histograms representing the average response time over the selected interval for all transactions.

Data

By default, all records stored on the CMRDETL data set are used as input to produce histogram reports.

The following control statements are specified:

- **TRAN=(ALL)** reports on all transactions.
- **DATA=RESPONSE** reports transaction response times.
- **CALC=AVERAGE** calculates averages.
- SCALE=10 sets the histogram scale to 10 seconds.
- END ends the report request.

The report produced by these control statements is shown in Figure 22 on page 92.

![Figure 22: Response time for all transactions](image)

### CPU used by CICS execution

This report shows histograms representing the hourly average percent CPU usage by CICS.

### Data

By default, all records on the CMRSTATS data set are used to produce histogram reports.

In this example, the histogram is limited to data collected during work hours 8:30 A.M. to 4:30 P.M.

The following control statements are specified:

- CICS reports on CICS system data.
- **INPUT=TAPE** uses sequential CMRSTATS data set.
- **DATA=CICSCPU** reports CICS and user CPU time usage.
- **MINUTES=60** accumulates data for 60-minute periods.
- **FROM TIME=08:30:00** uses data starting at 8:30 A.M.
- **TO TIME=17:30:00** uses data ending at 5:30 P.M.
- **END** ends the report request.

The report produced by these statements is shown in **Figure 23 on page 93.**

**Figure 23: CPU used by CICS execution**

An asterisk (*) represents the combined total when it exceeds the graphed total due to truncation of stacked values.

### CICS paging rate/second

This report shows histograms representing the paging activity caused by CICS execution.

### Data

One day's worth of input is used from the CMRSTATS data set.

The following control statements are specified:

- CICS reports on CICS system data.
- **INPUT=TAPE** uses sequential CMRSTATS data set.
- **DATA=PAGING** uses CICS paging data.
MINUTES=60 accumulates data for 60-minute periods.

FROM DATE=09/17/1999 selects data starting on September 17, 1999.

TO DATE=09/17/1999 does not select data after September 17, 1999.

SCALE=10 sets the histogram scale to 10 seconds.

END ends the report request.

The report produced by these statements is shown in Figure 24 on page 94.

Figure 24: CICS paging rate/second (page-in + page-out)

An asterisk (*) represents the combined total when it exceeds the graphed total due to truncation of stacked values.

CICS transaction rate/second

This report shows histograms representing the average transaction rate per second over the selected interval.

Data

By default, all records on the CMRSTATS data set are used as input to produce histogram reports.

The following control statements are specified:

- CICS reports on CICS system data.
- INPUT=TAPE uses sequential CMRSTATS data set.
- DATA=TASKRATE reports transaction rate per second.
- **CALC=AVERAGE** calculates averages.
- **MINUTES=60** accumulates data for 60-minute periods.
- **SCALE=10** sets the histogram scale to 10 seconds.
- **END** ends the report request.

The report produced by these control statements is shown in **Figure 25 on page 95**.

**Figure 25: CICS transaction rate/second**

<table>
<thead>
<tr>
<th>MM/DD/YYYY HH:MM</th>
<th>RATE/SEC.</th>
<th>0.00</th>
<th>1.00</th>
<th>2.00</th>
<th>3.00</th>
<th>4.00</th>
<th>5.00</th>
<th>6.00</th>
<th>7.00</th>
<th>8.00</th>
<th>9.00</th>
<th>10.00</th>
</tr>
</thead>
<tbody>
<tr>
<td>08/25/1999 09:28</td>
<td>TTT</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>08/25/1999 10:28</td>
<td>TT</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>08/25/1999 11:28</td>
<td>TTT</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>08/25/1999 12:28</td>
<td>TTTT</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>08/25/1999 13:28</td>
<td>TTT</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>08/25/1999 14:28</td>
<td>TT</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>08/25/1999 15:28</td>
<td>TT</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>08/25/1999 16:28</td>
<td>TTTT</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>08/25/1999 17:28</td>
<td>TT</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
</tbody>
</table>

**LEGEND:**  
- **T** = TERMINAL-ATTACHED-TRANSACTION-RATE  
- **=** TOTAL-TRANSACTION-RATE

Sample reports
Service Level Analysis report

The Service Level Analysis report shows the distribution of transaction response times over a range of service levels (specified by the SERVICE statement in the CMRSOPT table).

This report can be used to monitor the hourly distribution of transactions response times. JCL and a sample Service Level Analysis report are described in the following sections.

Creating a Service Level Analysis report

Figure 26 on page 97 shows the JCL to execute the CMRLTXR program as a batch job and create the Service Level Analysis report.

Figure 26: Service Level Analysis report--input JCL

```
//jobname JOB user parameters
/*JOBPARM user parameters
//CMRLTXR EXEC PGM=CMRLTXR
//JOBLIB DD DISP=SHR,DSN=CMR.CMRV5.BBLINK
//CMRSTAT DD DISP=SHR,DSN=CMR.CMRV5.CMRSTATS
//REPORT DD SYSOUT=* 
//SORTWK01 DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSPRINT DD SYSOUT=* 
//SYSPRINT DD SYSOUT=* 
//SYSIN DD * 
```

The JCL statements are as follows:

- JOB initiates the job.
- EXEC specifies the name of the program (PGM=CMRLTXR) that generates the Service Level Analysis report.
- //JOBLIB defines the program library containing the MainView for CICS load modules (BBLINK).
- //CMRSTAT defines the VSAM or sequential data set containing records of CICS operational statistics (CMRSTATS).
Control statements

Control statements identify the CICS regions and date of the input records that create the Service Level Analysis report.

Valid control statements are as follows:

<table>
<thead>
<tr>
<th>Control statement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TARGET=cicsRegion</td>
<td>name of the CICS region whose input records are used to create the report.</td>
</tr>
</tbody>
</table>
| DATE=mm/dd/yyyy   | date the records that are used in the report were created  
You can specify the starting date by using a relative format that is an offset from the current date. For example, to start the selected records with yesterday’s date, specify  
DATE=*-1 |

**Note**

If you specify a date in the old format (mm/dd/yy), yy is interpreted as 19 yy.

Sample report

Figure 27 on page 99 shows an example of the Service Level Analysis report. Transactions are shown in hourly intervals by counts and percentage for each service level range.
Figure 27: Service Level Analysis report

Table 14 on page 99 describes the fields of the Service Level Analysis report shown in Figure 27 on page 99:

Table 14: Service Level Analysis report fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTVL DATE</td>
<td>gregorian date of the data in <em>mm/dd/yyyy</em> format</td>
</tr>
<tr>
<td>INTVL TIME</td>
<td>time stamp of the data in <em>hh:mm</em> format</td>
</tr>
<tr>
<td>TOTAL TRANS.</td>
<td>total number of terminal attached transactions executed during this time interval</td>
</tr>
</tbody>
</table>
| 0.0 THROUGH 0.5 - 90.0 AND OVER | service level ranges in seconds  
|                     | You can adjust service level ranges (see the description of the CMRSOPT macro in the *MainView for CICS Customization Guide* for additional information) |
| SERVICE COUNTS      | number of transactions completed within the service level specified in the column heading |
| CUMULATIVE CNTS     | cumulative number of transactions completed within the service level specified in the column heading |
| SERVICE PERCENT     | percentage of all transactions completed within the service level specified in the column heading |
| CUMULATIVE PCTS     | cumulative percentage of transactions completed within the service level specified in the column heading |
COBOL reports

The PERFORMANCE REPORTER’s CMRFREDN facility allows COBOL programs to access transaction records stored on the CMRDETL data set.

CMRFREDN reads CMRDETL transaction records and reformat the fields into a COBOL-compatible format. The reformatted records are then passed back to the COBOL program that originally requested the data. You can produce either standard or custom COBOL reports from the reformatted transaction data.

CMRFREDN decompresses, processes and creates all records at the F7 level formatted for COBOL by BBSAMP member CMRCOBCN. See “Transaction detail data (CMRDETL)” on page 20 for more information.

In addition to the CMRFREDN facility, the MainView for CICS BBSAMP data set contains sample COBOL interface programs. These programs create a variety of predefined MainView for CICS reports after they have been edited to meet your site’s conditions.

These programs must be compiled and linked before they can be run. You should be able to modify these sample programs if you have some COBOL programming experience.

Note

When file expansion is turned on (FILEXPND=YES in CMRSOPT), the structure of expanded resource entries varies by type (that is, a file resource entry differs from a WebSphere MQ entry at the same offsets). A batch program that is processing the first field in each expanded resource entry receives file reads for a file resource entry, select counts for a DB2 resource entry, and open counts for a WebSphere MQ resource entry. All customized programs must process the resource entries separately by using the T6EFILEI field to determine the entry type. Entries of different types should not be processed together.

This chapter

- describes how CMRDETL transaction records are accessed and reformatted
- provides JCL examples that explain how to compile, link, and run sample COBOL programs
explains how to modify sample programs to create customized COBOL reports
- describes the controls statements used to create COBOL reports
- provides an example of each predefined report and a description of its fields

**Note**
All COBOL report programs must be compiled again and link-edited again using the current version of the MainView for CICS libraries. This step ensures that the copy books for the current Type 6E record format are incorporated into the programs.

## Accessing detail transaction records

All data fields in detail transaction records are accessible to COBOL programs.

CMRDETL record fields are described in the *MainView for CICS PERFORMANCE REPORTER Data Reference*.

The following COBOL subprograms access CMRDETL records:

- CMRFREDN converts CMRDETL record fields to a recognizable COBOL format and calls the CMRCMPRN program to decompress the records that are compressed in the CMRDETL data set.
- CMRCOBCN is a BBSAMP member that describes the COBOL format of the converted CMRDETL records.

The following sections describe these COBOL subprograms.

### CMRFREDN program

The CMRFREDN program performs the following processes:

- Read CMRDETL records from a VSAM or sequential data set
  - The maximum input record block size is 32K. The input data set can contain any record types produced by MainView for CICS, but only CMRDETL records are passed to the calling COBOL program.
- Reformat CMRDETL record fields into a COBOL-compatible form
  - When reading records from the CMRDETL data set
- Converts record fields to a COBOL-compatible format
Passes a return code back to the calling COBOL program

**Parameters**

Two parameters are passed from a COBOL program to the CMRFREDN program:

- The first parameter defines the function to be performed.
- The second parameter defines the CMRDETL record to be passed back to the calling program.

**CMRFREDN FT messages**

The CMRFREDN program generates FT messages.

You can display descriptions of the messages by using the General Service, MESSAGES service, or by browsing the MainView for CICS BBMLIB data set.

The CMRFREDN program might generate any of the following FT messages:

- FT256 cancels the job if a VSAM open, close, or read operation receives a non-zero return code.
- FT257 cancels the job when
  - the COBOL function passed to CMRFREDN is invalid
  - a read or close operation is requested and the file is closed
  - the function is open and the file is open
- FT258 cancels the job because of an end-of-file error.
- FT260 cancels the job if program CMRFREDN cannot load program CMRCMPRN.
- FT261 cancels the job if program CMRCMPRN passes a nonzero return code to CMRFREDN.

**CMRCOBCN (COBOL copybook) program**

The COBOL copybook contains the formats of the records held in the CMRDETL data set.

The CMRFREDN program passes the CMRDETL records to the calling COBOL program. The COBOL copybook is in BBSAMP member CMRCOBCN.
The CMRCOBCN defines three COBOL 01 data structures:

- CMR-Control
- CMRDETL
- COPYRIGHT

The purpose of these data structures is explained in the following sections.

**CMR–CONTROL**

The CMR–CONTROL data structure passes control fields between the CMRFREDN program and the executing COBOL program. CMR–CONTROL contains the following fields:

- The **Status** field is defined with a COBOL 88-level name to indicate end of file.
- The **Function** field defines the operation to be performed in the CMRFREDN program.

**CMRDETL**

CMRDETL contains the COBOL-compatible definitions of the record fields. The COBOL versions of CMRDETL records use the same field names described in the *MainView for CICS PERFORMANCE REPORTER Data Reference*.

**Note**

Because of COBOL’s processing requirements, the definitions of the data types associated with the record fields vary from the descriptions given in the *MainView for CICS PERFORMANCE REPORTER Data Reference*.

COBOL CMRDETL field formats are as follows:

<table>
<thead>
<tr>
<th>Field</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>Numeric</td>
<td>packed decimal</td>
</tr>
<tr>
<td>Time</td>
<td>seconds</td>
</tr>
<tr>
<td>TYPE</td>
<td>two bytes: 6D or 6E</td>
</tr>
<tr>
<td>T6EQUAL</td>
<td>two bytes, packed decimal</td>
</tr>
</tbody>
</table>

**COPYRIGHT**

The COPYRIGHT data structure contains a copyright statement. The copyright also appears in the compiled COBOL object program.
You can fulfill virtually any need your site has for specialized reports by using the COBOL interface.

The sample COBOL programs use batch processing to create CICS performance reports.

You can use the sample programs to

- generate the sample reports by using your data
- develop your own COBOL programs

When you understand how these sample COBOL programs create reports, you can write your own programs to create CICS performance reports from CMRDETL records.

**Note**

Use of these sample programs requires an understanding of COBOL programming.

The following sections explain how to modify a sample COBOL program, including

- copying and modifying the sample program
- compiling and linking the program
- executing the program

**Copying and modifying the program**

Use the following procedure to copy and modify the program.

1. Copy the sample program. (Sample COBOL programs are listed in Table 15 on page 107.)

2. Copy the standard compile and link JCL from your library.

   If you do not have standard compile and link JCL, member CMRCOBJ1 in your BBSAMP data set contains samples of compile and link JCL that can be used with the sample COBOL programs.

3. Modify the compile and link JCL according to your site standards.
Compiling and linking the program

Use the following procedure to compile and link the program that you modified.

1. Submit the JCL to compile and link the sample program.
2. Verify successful compilation and linking.
3. If you have any JCL errors, make the corrections and resubmit the job.

**Note**
Be sure to compile and link the program using the current version of the MainView for CICS libraries. This step ensures that the copy books for the current Type 6E record format are incorporated into the programs.

Running the program

Use the following procedure to run the program.

1. Copy the sample JCL from member CMRCOBJ2. (The JCL is explained in “CMRCOBJ2—sample execution JCL” on page 108.)
2. Modify the JCL according to your site standards.
3. Enter the control statements you need to limit the input to the program (on the CARDIN DD statement), using your site standards.
4. Submit the job to run the sample program.
5. Verify successful completion.
6. If you have any JCL errors, make the corrections and resubmit the job.

**Note**
Be sure to run the program using the current version of the MainView for CICS libraries.

Sample programs

Two types of sample COBOL programs exist: transaction and file analysis.
The transaction programs create reports that list performance statistics by the transactions recorded in a CICS region:

- Performance Analysis reports contain CICS performance statistics that are summarized with control breaks by
  - transaction ID
  - program name
  - terminal ID
  - node name
  - user ID

- Rogues Report lists transactions with the longest response times.

- Abended Transaction Report lists all transactions that abended in a CICS region.

The file analysis program has control breaks based on the file name field of transaction records. This program creates the File Analysis for All Files report, which lists wait time for CICS files by CICS region.

The following table lists the sample COBOL programs and their associated reports. The table also includes a reference to a section in this book that shows an example of the COBOL report.

### Table 15: Sample COBOL programs

<table>
<thead>
<tr>
<th>Sample report</th>
<th>Sample library member</th>
<th>Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Performance Analysis for All Transactions</td>
<td>CMRCB11N</td>
<td>produces a report that evaluates task performance by transaction ID</td>
<td>“Performance Analysis for All Transactions report (CMRCB11N)” on page 111</td>
</tr>
<tr>
<td>Performance Analysis for All Programs</td>
<td>CMRCB12N</td>
<td>produces a report with control breaks based on the T6EPGNM field</td>
<td>“Performance Analysis by Program report (CMRCB12N)” on page 112</td>
</tr>
<tr>
<td>Performance Analysis by Terminal</td>
<td>CMRCB13N</td>
<td>produces a report that evaluates task performance by terminal ID</td>
<td>“Performance Analysis by Terminal report (CMRCB13N)” on page 113</td>
</tr>
<tr>
<td>Performance Analysis by Netname</td>
<td>CMRCB14N</td>
<td>produces a report that evaluates task performance by VTAM symbolic node name</td>
<td>“Performance Analysis by Netname report (CMRCB14N)” on page 114</td>
</tr>
<tr>
<td>Sample report</td>
<td>Sample library member</td>
<td>Description</td>
<td>Reference</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>-----------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Performance Analysis for All Users</td>
<td>CMRCB15N</td>
<td>produces a report that evaluates task performance by user ID</td>
<td>“Performance Analysis for All Users report (CMRCB15N)” on page 115</td>
</tr>
<tr>
<td>File Analysis for All Files</td>
<td>CMRCB21N</td>
<td>produces a report that evaluates file performance by file name</td>
<td>“File Analysis for All Files report (CMRCB21N)” on page 116</td>
</tr>
<tr>
<td>Rogues Report</td>
<td>CMRCB23N</td>
<td>produces a report that lists the 100 tasks that have the longest response times</td>
<td>“Rogues report (CMRCB23N)” on page 116</td>
</tr>
<tr>
<td>Abended Transaction Report</td>
<td>CMRCB25N</td>
<td>produces a report that lists all transactions that suffered an abend</td>
<td>“Abended Transaction report (CMRCB25N)” on page 117</td>
</tr>
</tbody>
</table>

**CMRCOBJ2—sample execution JCL**

The following figure shows an example of the JCL in member CMRCOBJ2 of your BBSAMP data set.

CMRCOBJ2 runs the sample COBOL programs described in Table 15 on page 107. You must include the program name with the EXEC statement to run a specific sample report program or to run your own COBOL report program.

**Figure 28: JCL execution statements for all COBOL reports**

```plaintext
//jobname  JOB .....  
/** CMRCOBJ2  
/** THIS IS SAMPLE EXECUTION JCL  
/** THIS JCL WILL HAVE TO BE MODIFIED BEFORE USE  
/** THE LOAD LIBRARY MUST BE DEFINED IN THE STEPLIB DD STATEMENT  
/** THE CMRDZTL FILE MUST BE DEFINED  
/** THE CMRDZTL FILE MAY BE A VSAM OR SEQUENTIAL DATA SET  
/** THE CMRDZTL FILE HAS A MAXIMUM BLOCK SIZE OF 32K  
/**  
/** *** CHANGE ALL OCCURANCES OF 'XX' TO THE SUFFIX OF THE APPROPRIATE  
/** *** COBOL SAMPLE REPORT  
/**  
/** EXEC  
//CMRCBXXN EXEC PGM=CMRCBXXN,REGION=3096K  
//STEPLIB DD DSN=SYS1.VSCOBII.SYS.COB2LIB,DISP=SHR  
// DD DSN=.....LOAD,DISP=SHR  
//SORTWK01 DD UNIT=SYSDA,SPACE=(CYL,(5,5))  
//SORTWK02 DD UNIT=SYSDA,SPACE=(CYL,(5,5))  
//SORTWK03 DD UNIT=SYSDA,SPACE=(CYL,(5,5))  
//CMRDZTL DD DISP=SHR,DSN=CMR.CMRV3.CMRDZTL  
//CARDIN DD *  
//REPORT DD SYSOUT=*  
//SYSABOUT DD SYSOUT=*  
//SYSOUT DD SYSOUT=*  
//SYSPRINT DD SYSOUT=*  
```
The sample JCL contains the following statements:

- **JOB** initiates the job.
- **EXEC** specifies the name of the program that creates the report (PGM=), as follows:

<table>
<thead>
<tr>
<th>Program name</th>
<th>Generated report</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMRCB11N</td>
<td>Transaction Analysis for All Transactions</td>
</tr>
<tr>
<td>CMRCB12N</td>
<td>Transaction Analysis by Program</td>
</tr>
<tr>
<td>CMRCB13N</td>
<td>Transaction Analysis by Terminal</td>
</tr>
<tr>
<td>CMRCB14N</td>
<td>Transaction Analysis by Node Name</td>
</tr>
<tr>
<td>CMRCB15N</td>
<td>Transaction Analysis by User ID</td>
</tr>
<tr>
<td>CMRCB21N</td>
<td>File Analysis for All Files</td>
</tr>
<tr>
<td>CMRCB23N</td>
<td>Rogues Report</td>
</tr>
<tr>
<td>CMRCB25N</td>
<td>Abended Transaction</td>
</tr>
</tbody>
</table>

- **//STEPLIB** defines your program library and the MainView for CICS load modules.

- **//SORTWK01** defines the first sort area data set.

- **//SORTWK02** defines the second sort area data set.

- **//SORTWK03** defines the third sort area data set.

- **//CMRDETL** defines VSAM or sequential data sets containing T6E detail transaction records.

- **//CARDIN** defines PERFORMANCE REPORTER control statements to use as input to the report program. Some examples of control statements are as follows:

<table>
<thead>
<tr>
<th>Control statement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CICS=</td>
<td>provides the CICS region name (for example, CICS=cicsprod)</td>
</tr>
<tr>
<td>DATE=</td>
<td>provides the start date for processing in the format yyyyymmdd (for example, DATE=19990917)</td>
</tr>
<tr>
<td>TIME=</td>
<td>provides the start time for processing in the format hhmmss (for example, TIME=081133)</td>
</tr>
</tbody>
</table>

- **//REPORT** defines the report output for the predefined reports.
- //SYSABOUT defines the abbreviated dump data set.
- //SYSOUT defines the output class.
- //SYSPRINT defines the output class for the sort utility. The name of the DD statement is determined at sort system generation.

**COBOL report fields**

The following column headings appear in all sample COBOL reports described in this chapter:

- **CICS REGION** (name of the CICS region associated with the data)
- **RUN DATE** (gregorian date associated with the data in *yyyy/mm/dd* format)
- **TRAN CODE** (transaction identification of the T6ECICS task)
- **PROGRAM NAME** (name of the program associated with the data)
- **TERM ID** (identification number of the terminal)
- **NETNAME** (name of the net node)
- **USER ID** (identification of the CICS user)
- **FILE NAME** (name of the file being accessed)
- **RUN TIME** (time associated with the data in *hh:mm:ss* format)
- **ABEND CODE** (code associated with this abended transaction)
- **TRAN COUNT** (number of transactions executed from this terminal)
- **AVERAGE CPU REAL** (average seconds of CPU real time used by each transaction)
- **AVG FILE WAIT TIME** (average seconds that a transaction waits for called files)
- **AVG FILE CALLS** (average number of times files were called for each transaction)
- **AVG DB WAIT TIME** (average time that a transaction waited for access to a database)
- **AVG DB CALLS** (average number of calls to the database for each transaction)
- **AVERAGE STRG HWM** (average storage high-water mark in bytes during this time interval)
- **STORAGE OCCUPANCY** (usage of storage over time)
  
  A value of 1 is equivalent to 250 bytes held for .25 seconds. This value can be used to evaluate the impact on available storage caused by a transaction.
- **AVG PROG STRG HWM** (average program storage high-water mark reached, in bytes, for all transactions)

### Performance Analysis for All Transactions report (CMRCB11N)

The CMRCB11N sample program produces the Performance Analysis for All Transactions report.

This report has control breaks based on the T6ETRID (transaction ID) field. See “Transaction detail data (CMRDETL)” on page 20 for more information.

Generate this report to evaluate task performance subdivided by transaction.

### Sample report (Performance Analysis for All Transactions)

This topic shows a Performance Analysis for All Transactions sample report.

#### Figure 29: Performance Analysis for All Transactions report
### Performance Analysis by Program report (CMRCB12N)

The CMRCB12N sample program produces the Performance Analysis by Program report.

This report has control breaks based on the T6EPGNNM (program name) field. See "Transaction detail data (CMRDETL)" on page 20 for more information.

Generate this report to evaluate task performance subdivided by program name.

---

**Note**

This report is most useful when a fourth-generation language (4GL) is used.

---

#### Sample report (Performance Analysis by Program)

This topic shows a Performance Analysis by Program sample report.

**Figure 30: Performance Analysis by Program report**

---

<table>
<thead>
<tr>
<th>PROGRAM: CMRCB12N</th>
<th>PERFORMANCE ANALYSIS FOR ALL PROGRAMS</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CICS REGION</strong></td>
<td><strong>RUN DATE</strong></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/17</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/17</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/17</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/17</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/17</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/17</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/17</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/17</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/17</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/17</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/17</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/17</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/17</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/17</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/17</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/17</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/17</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/17</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/17</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/17</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/17</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/17</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/17</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/17</td>
</tr>
</tbody>
</table>

**Date:** 1999/09/24

**Time:** 16:19:46

---

MainView for CICS PERFORMANCE REPORTER User Guide

112
## Performance Analysis by Terminal report (CMRCB13N)

The CMRCB13N sample program produces the Performance Analysis by Terminal report.

This report has control breaks based on the `T6ETMID` (terminal ID) field. See “Transaction detail data (CMRDETL)” on page 20 for more information.

Generate this report to evaluate task performance subdivided by terminal.

### Sample report (Performance Analysis by Terminal)

This topic shows a Performance Analysis by Terminal sample report.

**Figure 31: Performance Analysis by Terminal report**
Performance Analysis by Netname report (CMRCB14N)

The CMRCB14N sample program produces the Performance Analysis by Netname report.

This sample report has control breaks based on the T6ENETNM (network node name) field. See “Transaction detail data (CMRDETL)” on page 20 for more information.

Generate this report to evaluate task performance subdivided by VTAM symbolic node name.

Sample report (Performance Analysis by Netname)

This topic shows a Performance Analysis by Netname sample report.

**Figure 32: Performance Analysis by Netname report**

<table>
<thead>
<tr>
<th>CICS REGION</th>
<th>RUN DATE</th>
<th>NETNAME</th>
<th>TRAN COUNT</th>
<th>AVERAGE RESPONSE</th>
<th>AVERAGE CPU REAL</th>
<th>AVERAGE WAIT TIME</th>
<th>AVG FILE CALLS</th>
<th>AVG DB CALLS</th>
<th>AVERAGE STRG HWM</th>
<th>STORAGE OCCUPANCY</th>
<th>AVG PROG STRG HWM</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRODCICS</td>
<td>1999/09/18</td>
<td>3,442</td>
<td>13.937</td>
<td>.001</td>
<td>.004</td>
<td>.535</td>
<td>.000</td>
<td>.000</td>
<td>7,061</td>
<td>6,831</td>
<td>6,010</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/18</td>
<td>A100A02</td>
<td>79</td>
<td>1.123</td>
<td>.022</td>
<td>.004</td>
<td>1.101</td>
<td>.172</td>
<td>1,367</td>
<td>66,515</td>
<td>872</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/18</td>
<td>A100A03</td>
<td>2</td>
<td>1.051</td>
<td>.004</td>
<td>.000</td>
<td>.500</td>
<td>.000</td>
<td>.000</td>
<td>45,788</td>
<td>44</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/18</td>
<td><em>TOTAL</em></td>
<td>3,523</td>
<td>13.642</td>
<td>.001</td>
<td>.004</td>
<td>.548</td>
<td>.003</td>
<td>.030</td>
<td>8,477</td>
<td>6,693</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/21</td>
<td>1,986</td>
<td>13.340</td>
<td>.001</td>
<td>.006</td>
<td>.898</td>
<td>.080</td>
<td>.000</td>
<td>7,086</td>
<td>4,234</td>
<td>5,915</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/21</td>
<td>A051A03</td>
<td>8</td>
<td>1.427</td>
<td>.014</td>
<td>.000</td>
<td>.375</td>
<td>.172</td>
<td>1,367</td>
<td>52,203</td>
<td>1,623</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/21</td>
<td>A100A03</td>
<td>181</td>
<td>3.620</td>
<td>.029</td>
<td>.016</td>
<td>1.718</td>
<td>.186</td>
<td>8,966</td>
<td>598</td>
<td>175,692</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/21</td>
<td>SMTCA201</td>
<td>25</td>
<td>3.457</td>
<td>.040</td>
<td>.010</td>
<td>1.200</td>
<td>.734</td>
<td>6,640</td>
<td>70,617</td>
<td>1,301</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/21</td>
<td><em>TOTAL</em></td>
<td>2,211</td>
<td>12.079</td>
<td>.004</td>
<td>.007</td>
<td>.969</td>
<td>.037</td>
<td>.463</td>
<td>14,322</td>
<td>3,911</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/17</td>
<td>644</td>
<td>.027</td>
<td>.001</td>
<td>.003</td>
<td>.545</td>
<td>.000</td>
<td>.000</td>
<td>7,012</td>
<td>2</td>
<td>6,084</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/17</td>
<td>SMTAE209</td>
<td>28</td>
<td>2.962</td>
<td>.046</td>
<td>.004</td>
<td>1.000</td>
<td>.355</td>
<td>10,500</td>
<td>71,947</td>
<td>3,245</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/17</td>
<td><em>TOTAL</em></td>
<td>672</td>
<td>.150</td>
<td>.003</td>
<td>.563</td>
<td>.056</td>
<td>.437</td>
<td>9,717</td>
<td>137</td>
<td>12,872</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/23</td>
<td>2,741</td>
<td>1.447</td>
<td>.001</td>
<td>.005</td>
<td>.996</td>
<td>.000</td>
<td>.000</td>
<td>7,066</td>
<td>5,565</td>
<td>5,973</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/23</td>
<td>A100A02</td>
<td>2</td>
<td>3.125</td>
<td>.063</td>
<td>.014</td>
<td>.500</td>
<td>.000</td>
<td>47,016</td>
<td>2,182</td>
<td>77,292</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/23</td>
<td>A100A03</td>
<td>18</td>
<td>1.582</td>
<td>.026</td>
<td>.013</td>
<td>1.333</td>
<td>.786</td>
<td>5,777</td>
<td>70,284</td>
<td>1,797</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/23</td>
<td>LCZHA1E4</td>
<td>20</td>
<td>2.112</td>
<td>.036</td>
<td>.016</td>
<td>1.650</td>
<td>.104</td>
<td>9,150</td>
<td>85,640</td>
<td>2,278</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/23</td>
<td>SMTCA206</td>
<td>9</td>
<td>1.168</td>
<td>.018</td>
<td>.019</td>
<td>1.777</td>
<td>.077</td>
<td>.777</td>
<td>54,978</td>
<td>814</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/23</td>
<td>SMTCA207</td>
<td>17</td>
<td>1.964</td>
<td>.032</td>
<td>.008</td>
<td>1.411</td>
<td>.705</td>
<td>5,294</td>
<td>82,023</td>
<td>2,079</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/23</td>
<td>SMTCA204</td>
<td>163</td>
<td>.994</td>
<td>.035</td>
<td>.001</td>
<td>1.036</td>
<td>.538</td>
<td>6,801</td>
<td>93,258</td>
<td>1,355</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/23</td>
<td>SMTCA210</td>
<td>13</td>
<td>2.649</td>
<td>.028</td>
<td>.016</td>
<td>.615</td>
<td>1.198</td>
<td>10,615</td>
<td>59,868</td>
<td>2,210</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/23</td>
<td><em>TOTAL</em></td>
<td>2,983</td>
<td>1.436</td>
<td>.004</td>
<td>.005</td>
<td>1.007</td>
<td>.050</td>
<td>.535</td>
<td>13,513</td>
<td>5,239</td>
</tr>
<tr>
<td>PRODCICS</td>
<td><em>TOTAL</em></td>
<td>9,389</td>
<td>8.430</td>
<td>.003</td>
<td>.005</td>
<td>.794</td>
<td>.030</td>
<td>.321</td>
<td>11,519</td>
<td>5,107</td>
<td>16,416</td>
</tr>
<tr>
<td>RUN <em>TOTAL</em></td>
<td>9,389</td>
<td>8.430</td>
<td>.003</td>
<td>.005</td>
<td>.794</td>
<td>.030</td>
<td>.321</td>
<td>11,519</td>
<td>5,107</td>
<td>16,416</td>
<td></td>
</tr>
</tbody>
</table>

MainView for CICS PERFORMANCE REPORTER User Guide
The CMRCB15N sample program produces the Performance Analysis for All Users report.

This sample report has control breaks based on the \textbf{T6EUSER} (user ID) field. See “Transaction detail data (CMRDETL)” on page 20 for more information.

Generate this report to evaluate task performance subdivided by user ID.

### Sample report (Performance Analysis for All Users)

This topic shows a Performance Analysis for All Users sample report.

#### Figure 33: Performance Analysis for All Users report

<table>
<thead>
<tr>
<th>PROGRAM:</th>
<th>CMRCB15N</th>
<th>PERFORMANCE ANALYSIS FOR ALL USERS</th>
<th>DATE:</th>
<th>1999/09/24</th>
<th>TIME:</th>
<th>12:00:46</th>
<th>PAGE: 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>CICS</td>
<td>RUN DATE</td>
<td>USER ID</td>
<td>TRAN</td>
<td>AVERAGE</td>
<td>AVERAGE</td>
<td>AVG FILE</td>
<td>AVG DB</td>
</tr>
<tr>
<td>-----------</td>
<td>-----------</td>
<td>---------</td>
<td>------</td>
<td>---------</td>
<td>---------</td>
<td>----------</td>
<td>--------</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/15</td>
<td>33</td>
<td>.298</td>
<td>.030</td>
<td>.002</td>
<td>1.393</td>
<td>.103</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/15</td>
<td>CPS19</td>
<td>2.678</td>
<td>1.253</td>
<td>.001</td>
<td>.004</td>
<td>1.147</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/15</td>
<td>CMR4</td>
<td>174</td>
<td>1.240</td>
<td>.038</td>
<td>.003</td>
<td>1.040</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/15</td>
<td><em>TOTAL</em></td>
<td>2,885</td>
<td>1.242</td>
<td>.004</td>
<td>.004</td>
<td>1.144</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/16</td>
<td>186</td>
<td>.613</td>
<td>.021</td>
<td>.005</td>
<td>1.198</td>
<td>.177</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/16</td>
<td>JEN1</td>
<td>3,402</td>
<td>13.994</td>
<td>.001</td>
<td>.006</td>
<td>2.126</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/16</td>
<td>CMR4</td>
<td>661</td>
<td>.970</td>
<td>.038</td>
<td>.003</td>
<td>.899</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/16</td>
<td><em>TOTAL</em></td>
<td>4,256</td>
<td>11.387</td>
<td>.008</td>
<td>.004</td>
<td>1.909</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/17</td>
<td>1,204</td>
<td>.733</td>
<td>.042</td>
<td>.005</td>
<td>1.151</td>
<td>.309</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/17</td>
<td>CMR4</td>
<td>3,503</td>
<td>4.483</td>
<td>.001</td>
<td>.007</td>
<td>1.774</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/17</td>
<td><em>TOTAL</em></td>
<td>4,756</td>
<td>11.387</td>
<td>.008</td>
<td>.004</td>
<td>1.909</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/18</td>
<td>3,425</td>
<td>14.066</td>
<td>.001</td>
<td>.004</td>
<td>.538</td>
<td>.000</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/18</td>
<td><em>TOTAL</em></td>
<td>3,523</td>
<td>13.642</td>
<td>.001</td>
<td>.004</td>
<td>.548</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/21</td>
<td>233</td>
<td>.858</td>
<td>.027</td>
<td>.013</td>
<td>1.463</td>
<td>.331</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/21</td>
<td>CMR7</td>
<td>1,967</td>
<td>13.468</td>
<td>.001</td>
<td>.006</td>
<td>.907</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/21</td>
<td>CMR8</td>
<td>11</td>
<td>1.313</td>
<td>.016</td>
<td>.014</td>
<td>1.545</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/21</td>
<td><em>TOTAL</em></td>
<td>2,211</td>
<td>12.079</td>
<td>.004</td>
<td>.007</td>
<td>.969</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/17</td>
<td>3</td>
<td>544</td>
<td>.007</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/17</td>
<td>CMR5</td>
<td>636</td>
<td>.027</td>
<td>.001</td>
<td>.003</td>
<td>.550</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/17</td>
<td>CMR1</td>
<td>31</td>
<td>2.634</td>
<td>.042</td>
<td>.004</td>
<td>.903</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/17</td>
<td><em>TOTAL</em></td>
<td>672</td>
<td>.003</td>
<td>.003</td>
<td>.563</td>
<td>.056</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/23</td>
<td>2,719</td>
<td>1.459</td>
<td>.001</td>
<td>.005</td>
<td>1.044</td>
<td>.571</td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/23</td>
<td>CMR6</td>
<td>2,983</td>
<td>1.436</td>
<td>.004</td>
<td>.005</td>
<td>1.007</td>
</tr>
</tbody>
</table>

Chapter 7 COBOL reports 115
# File Analysis for All Files report (CMRCB21N)

The CMRCB21N sample program produces the File Analysis for All Files report.

This report has control breaks based on the **T6EFN** (file name) field. See “Transaction detail data (CMRDETL)” on page 20 for more information.

Generate this report to evaluate file performance.

## Sample report (File Analysis for All Files)

This topic shows a File Analysis for All Files sample report.

### Figure 34: File Analysis for All Files report

<table>
<thead>
<tr>
<th>PROGRAM: CMRCB21N</th>
<th>FILE ANALYSIS FOR ALL FILES</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATE: 1999/09/24</td>
<td></td>
</tr>
<tr>
<td>TIME: 16:36:20</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PAGE;</th>
<th>CICS</th>
<th>REGION</th>
<th>RUN DATE</th>
<th>FILE NAME</th>
<th>FILE TYPE</th>
<th>TOTAL TRANS</th>
<th>TOTAL WAITS</th>
<th>AVG. WAITS PER TRAN</th>
<th>AVERAGE WAIT TIME</th>
<th>AVG. TIME PER WAIT</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/23</td>
<td>CMRSTAT</td>
<td>CICS FCT</td>
<td>148</td>
<td>198</td>
<td>1,337</td>
<td>.096</td>
<td>.071</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/23</td>
<td>CUDCLOG</td>
<td>CICS FCT</td>
<td>1</td>
<td>2</td>
<td>2,000</td>
<td>.035</td>
<td>.017</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/23</td>
<td>CIUCNCTL</td>
<td>CICS FCT</td>
<td>2</td>
<td>7</td>
<td>3,500</td>
<td>.088</td>
<td>.025</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/23</td>
<td>CUPFR01</td>
<td>DB2</td>
<td>1</td>
<td>50</td>
<td>50,000</td>
<td>7.246</td>
<td>.144</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/23</td>
<td>CUPINIT</td>
<td>DB2</td>
<td>3</td>
<td>6</td>
<td>2,000</td>
<td>.862</td>
<td>.431</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/23</td>
<td>CUPSSON</td>
<td>DB2</td>
<td>2</td>
<td>82</td>
<td>41,000</td>
<td>2.875</td>
<td>.070</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/23</td>
<td>DFHRPL</td>
<td>PSEUDO</td>
<td>24</td>
<td>1,829</td>
<td>7,589</td>
<td>.275</td>
<td>.036</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/23</td>
<td>DFHSRC0</td>
<td>PSEUDO</td>
<td>1</td>
<td>1</td>
<td>1,000</td>
<td>.346</td>
<td>.346</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/23</td>
<td>M2DCNLOG</td>
<td>CICS FCT</td>
<td>1</td>
<td>2</td>
<td>2,000</td>
<td>.052</td>
<td>.026</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/23</td>
<td>M2DCNLOG</td>
<td>CICS FCT</td>
<td>3</td>
<td>4</td>
<td>1,333</td>
<td>.046</td>
<td>.035</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/23</td>
<td>M2ISMCTL</td>
<td>CICS FCT</td>
<td>12</td>
<td>27</td>
<td>2,250</td>
<td>.070</td>
<td>.034</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/23</td>
<td>M2PACHP</td>
<td>DB2</td>
<td>4</td>
<td>42</td>
<td>10,500</td>
<td>.811</td>
<td>.077</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/23</td>
<td>M2PCHP1</td>
<td>DB2</td>
<td>2</td>
<td>34</td>
<td>17,900</td>
<td>2.517</td>
<td>.146</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/23</td>
<td>M2PCHP2</td>
<td>DB2</td>
<td>5</td>
<td>112</td>
<td>22,400</td>
<td>3.621</td>
<td>.161</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/23</td>
<td>M2PCHP3</td>
<td>DB2</td>
<td>65</td>
<td>167</td>
<td>2,569</td>
<td>.172</td>
<td>.067</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/23</td>
<td>M2PCHP4</td>
<td>DB2</td>
<td>1</td>
<td>67</td>
<td>67,000</td>
<td>3.647</td>
<td>.054</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/23</td>
<td>M2PCHP5</td>
<td>DB2</td>
<td>4</td>
<td>61</td>
<td>15,250</td>
<td>.354</td>
<td>.023</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/23</td>
<td>M2PCHP6</td>
<td>DB2</td>
<td>2</td>
<td>5</td>
<td>2,500</td>
<td>.031</td>
<td>.012</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/23</td>
<td>M2PCHP7</td>
<td>DB2</td>
<td>2</td>
<td>8</td>
<td>4,000</td>
<td>.336</td>
<td>.084</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/23</td>
<td>M2PCHP8</td>
<td>DB2</td>
<td>15</td>
<td>587</td>
<td>33,800</td>
<td>2.086</td>
<td>.061</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/23</td>
<td>M2PCHP9</td>
<td>DB2</td>
<td>3</td>
<td>59</td>
<td>19,666</td>
<td>1.624</td>
<td>.082</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/23</td>
<td>M2PCHP10</td>
<td>DB2</td>
<td>1</td>
<td>6</td>
<td>6,000</td>
<td>1.951</td>
<td>.325</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/23</td>
<td>M2PCHP11</td>
<td>DB2</td>
<td>1</td>
<td>18</td>
<td>18,000</td>
<td>.412</td>
<td>.023</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/23</td>
<td>M2PCHP12</td>
<td>DB2</td>
<td>4</td>
<td>50</td>
<td>12,500</td>
<td>2.396</td>
<td>.191</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/23</td>
<td>M2PCHP13</td>
<td>DB2</td>
<td>2</td>
<td>14</td>
<td>7,000</td>
<td>.071</td>
<td>.010</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/23</td>
<td>M2PCHP14</td>
<td>DB2</td>
<td>6</td>
<td>104</td>
<td>17,333</td>
<td>2.566</td>
<td>.148</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/23</td>
<td>M2PCHP15</td>
<td>DB2</td>
<td>1</td>
<td>17</td>
<td>17,000</td>
<td>5.219</td>
<td>.307</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/23</td>
<td>M2PRDR01</td>
<td>DB2</td>
<td>2</td>
<td>84</td>
<td>46,000</td>
<td>5.057</td>
<td>.170</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/23</td>
<td>M2PSPS01</td>
<td>DB2</td>
<td>1</td>
<td>4</td>
<td>4,000</td>
<td>2.383</td>
<td>.596</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/23</td>
<td>M2PSPS02</td>
<td>DB2</td>
<td>5</td>
<td>99</td>
<td>19,800</td>
<td>1.682</td>
<td>.084</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/23</td>
<td><em>TOTAL</em></td>
<td></td>
<td>541</td>
<td>3,666</td>
<td>6,776</td>
<td>.430</td>
<td>.063</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Rogues report (CMRCB23N)**

The CMRCB23N sample program produces the Highest Response Times All Transactions Report.
This report is also known as the Rogues Report. The report is produced by

- reading the CMRDETL file
- calculating response times of tasks

**Note**

Response times under 1.000 seconds are bypassed.

- producing a list of the 100 tasks with the longest response times

Generate this report to evaluate tasks with the greatest response times. Modify the report control statements in the JCL stream to limit the input for your report.

See “Transaction detail data (CMRDETL)” on page 20 for more information.

**Sample report (COBOL rogues)**

This topic shows a COBOL rogues sample report.

**Figure 35: COBOL rogues report**

<table>
<thead>
<tr>
<th>PROGRAM: CMRCB23N</th>
<th>HIGHEST RESPONSE TIMES ALL TRANSACTIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>CICS</td>
<td>TRAN CODE</td>
</tr>
<tr>
<td>PRODCICS 1999/09/17 09:06:50</td>
<td>MC00</td>
</tr>
<tr>
<td>PRODCICS 1999/09/17 12:41:21</td>
<td>MC00</td>
</tr>
<tr>
<td>PRODCICS 1999/09/17 12:41:21</td>
<td>MC00</td>
</tr>
<tr>
<td>PRODCICS 1999/09/17 12:41:21</td>
<td>MC00</td>
</tr>
<tr>
<td>PRODCICS 1999/09/17 12:41:21</td>
<td>MC00</td>
</tr>
<tr>
<td>PRODCICS 1999/09/17 12:41:21</td>
<td>MC00</td>
</tr>
<tr>
<td>PRODCICS 1999/09/17 12:41:21</td>
<td>MC00</td>
</tr>
<tr>
<td>PRODCICS 1999/09/17 12:41:21</td>
<td>MC00</td>
</tr>
<tr>
<td>PRODCICS 1999/09/17 12:41:21</td>
<td>MC00</td>
</tr>
<tr>
<td>PRODCICS 1999/09/17 12:41:21</td>
<td>MC00</td>
</tr>
<tr>
<td>PRODCICS 1999/09/17 12:41:21</td>
<td>MC00</td>
</tr>
<tr>
<td>PRODCICS 1999/09/17 12:41:21</td>
<td>MC00</td>
</tr>
<tr>
<td>PRODCICS 1999/09/17 12:41:21</td>
<td>MC00</td>
</tr>
</tbody>
</table>

**Abended Transaction report (CMRCB25N)**

The CMRCB25N sample program produces the Abended Transaction report by
Abended Transaction report (CMRCB25N)

- reading the CMRDETL file
- listing all transactions with an abend code

Generate this report to list all transactions that suffered an abend during the recorded interval. Modify the report control statements in the JCL stream to limit the input for your report.

See “Transaction detail data (CMRDETL)” on page 20 for more information.

Sample report (Abended Transaction)

This topic shows an Abended Transaction sample report.

Figure 36: Abended Transaction report

<table>
<thead>
<tr>
<th>CICS REGION</th>
<th>RUN DATE</th>
<th>RUN TIME</th>
<th>TRAN CODE</th>
<th>ABEND CODE</th>
<th>RESPONSE</th>
<th>CPU REAL</th>
<th>REAL TIME</th>
<th>WAIT TIME</th>
<th>CALLS</th>
<th>WAIT TIME</th>
<th>CALLS</th>
<th>STRG HWM</th>
<th>OCCUPANCY</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRODCICS</td>
<td>1999/09/05</td>
<td>19:09:07</td>
<td>AEY9</td>
<td>MC00</td>
<td>3.016</td>
<td>.057</td>
<td>.059</td>
<td>1.000</td>
<td>.000</td>
<td>.000</td>
<td>67,344</td>
<td>2,456</td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/15</td>
<td>13:33:20</td>
<td>AEY9</td>
<td>MC00</td>
<td>2.714</td>
<td>.063</td>
<td>.035</td>
<td>1.000</td>
<td>.000</td>
<td>.000</td>
<td>67,344</td>
<td>2,098</td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/15</td>
<td>13:45:31</td>
<td>AEY9</td>
<td>MC00</td>
<td>.110</td>
<td>.014</td>
<td>.000</td>
<td>1.000</td>
<td>.000</td>
<td>.000</td>
<td>74,176</td>
<td>80</td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/15</td>
<td>15:52:03</td>
<td>AEY9</td>
<td>MC00</td>
<td>.321</td>
<td>.014</td>
<td>.000</td>
<td>1.000</td>
<td>.000</td>
<td>.000</td>
<td>66,928</td>
<td>222</td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/08</td>
<td>13:02:52</td>
<td>QUIT</td>
<td>MC00</td>
<td>3.171</td>
<td>.067</td>
<td>.068</td>
<td>6.000</td>
<td>.049</td>
<td>2.000</td>
<td>87,208</td>
<td>3,038</td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/08</td>
<td>14:20:19</td>
<td>QUIT</td>
<td>MC00</td>
<td>.850</td>
<td>.028</td>
<td>.077</td>
<td>6.000</td>
<td>.064</td>
<td>2.000</td>
<td>87,208</td>
<td>758</td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/08</td>
<td>14:15:43</td>
<td>QUIT</td>
<td>MC00</td>
<td>.165</td>
<td>.027</td>
<td>.052</td>
<td>6.000</td>
<td>.006</td>
<td>2.000</td>
<td>87,208</td>
<td>142</td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/08</td>
<td>15:02:38</td>
<td>ATNI</td>
<td>FITX</td>
<td>1.946</td>
<td>.023</td>
<td>.000</td>
<td>1.000</td>
<td>.000</td>
<td>.000</td>
<td>30,962</td>
<td>204,839</td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/09</td>
<td>13:10:39</td>
<td>AEY9</td>
<td>MC00</td>
<td>3.382</td>
<td>.059</td>
<td>.020</td>
<td>1.000</td>
<td>.000</td>
<td>.000</td>
<td>67,344</td>
<td>2,470</td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/15</td>
<td>08:11:48</td>
<td>DB2E</td>
<td>MC00</td>
<td>4.760</td>
<td>.087</td>
<td>.090</td>
<td>6.000</td>
<td>1.120</td>
<td>1.000</td>
<td>116,080</td>
<td>7,269</td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/15</td>
<td>08:19:29</td>
<td>DB2E</td>
<td>MC00</td>
<td>.802</td>
<td>.023</td>
<td>.087</td>
<td>6.000</td>
<td>.230</td>
<td>3.000</td>
<td>64,544</td>
<td>618</td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/15</td>
<td>08:21:03</td>
<td>DB2E</td>
<td>MC00</td>
<td>2.316</td>
<td>.036</td>
<td>.140</td>
<td>6.000</td>
<td>.064</td>
<td>1.000</td>
<td>112,176</td>
<td>3,682</td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/15</td>
<td>08:37:47</td>
<td>DB2E</td>
<td>MC00</td>
<td>.209</td>
<td>.034</td>
<td>.045</td>
<td>6.000</td>
<td>.094</td>
<td>1.000</td>
<td>112,176</td>
<td>243</td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/17</td>
<td>13:12:22</td>
<td>ASRA</td>
<td>MC00</td>
<td>12.296</td>
<td>.133</td>
<td>.080</td>
<td>6.000</td>
<td>3.665</td>
<td>12.000</td>
<td>128,848</td>
<td>20,458</td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/17</td>
<td>13:17:54</td>
<td>ASRA</td>
<td>MC00</td>
<td>1.699</td>
<td>.089</td>
<td>.052</td>
<td>6.000</td>
<td>.013</td>
<td>5.000</td>
<td>111,056</td>
<td>2,574</td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/17</td>
<td>13:25:39</td>
<td>ASRA</td>
<td>MC00</td>
<td>.483</td>
<td>.033</td>
<td>.081</td>
<td>6.000</td>
<td>.042</td>
<td>12.000</td>
<td>124,944</td>
<td>698</td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/17</td>
<td>13:34:35</td>
<td>ASRA</td>
<td>MC00</td>
<td>1.069</td>
<td>.033</td>
<td>.129</td>
<td>6.000</td>
<td>.033</td>
<td>10.000</td>
<td>110,816</td>
<td>1,484</td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/17</td>
<td>13:40:19</td>
<td>LLL</td>
<td>MC00</td>
<td>717.335</td>
<td>.014</td>
<td>.000</td>
<td>.000</td>
<td>96,928</td>
<td>1,029,337</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/17</td>
<td>14:34:24</td>
<td>LLL</td>
<td>MC00</td>
<td>1,712.149</td>
<td>.975</td>
<td>.650</td>
<td>24.000</td>
<td>.000</td>
<td>0.000</td>
<td>103,248</td>
<td>2,461,196</td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/17</td>
<td>14:57:22</td>
<td>ATNI</td>
<td>FITX</td>
<td>.145</td>
<td>.003</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>31,048</td>
<td>28,320</td>
<td></td>
</tr>
<tr>
<td>PRODCICS</td>
<td>1999/09/18</td>
<td>18:46:08</td>
<td>AEY9</td>
<td>MC00</td>
<td>3.243</td>
<td>.060</td>
<td>.020</td>
<td>1.000</td>
<td>.000</td>
<td>.000</td>
<td>67,344</td>
<td>2,567</td>
<td></td>
</tr>
</tbody>
</table>
SAS reports

Sample programs are included in your BBSAMP data set to create SAS reports from data collected by MainView for CICS. Copies of these sample programs can be modified to create unique SAS reports.

Note
When file expansion is turned on (FILEXPND=YES in CMRSOPT), the structure of expanded resource entries varies by type (that is, a file resource entry differs from a WebSphere MQ entry at the same offsets). A batch program that is processing the first field in each expanded resource entry receives file reads for a file resource entry, select counts for a DB2 resource entry, and open counts for a WebSphere MQ resource entry. All customized programs must process the resource entries separately by using the T6EFILEI field to determine the entry type. Entries of different types should not be processed together.

This chapter contains

- control statements to specify the input data for SAS programs
- descriptions of SAS report fields
- examples of SAS reports

Detail transaction records are compressed when they are stored on the CMRDETL data set. You have the choice of creating SAS reports from compressed or decompressed 6E and 6D records. You must run a separate job if you want to create SAS reports with decompressed CMRDETL records.

Member CMRSA10N of your BBSAMP data set decompresses 6E and 6F records. The decompressed output records are compatible with SAS programs. Edit a copy of CMRSA10N and run the program to create decompressed detail transaction records before running your SAS report programs.

Note
Be sure to run your SAS report programs using the current version of the MainView for CICS libraries.
The MainView for CICS SAS reporting facility supports the F7 formats of the T6E history record. See “Transaction detail data (CMRDETL)” on page 20 for information about the F7 record.

The following information is available for the F7 level history records:

- CMRNIFUE is the SAS in-file user exit that reads and decompresses T6E records
- SASI6ExN members in the BBSAMP data set are SAS formats for the various sections of the T6E record
- CMRSAxxN members in the BBSAMP data set are sample programs
- CMRCMPWN reads the CMRDETL records and decompresses them to the F7 format

### Sample SAS programs

The interface routines in MainView for CICS give SAS programs access to records held in the CMRDETL and CMRSTATS data sets.

SAS reports are created from these records. See the MainView for CICS PERFORMANCE REPORTER Data Reference for the format of CMRDETL and CMRSTATS records.

The following table identifies the sample SAS programs and their associated reports. The table also includes a reference to a section in this book that shows an example of the SAS report.

<table>
<thead>
<tr>
<th>Sample report title</th>
<th>Sample library member/program name</th>
<th>Description</th>
<th>Section reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Performance Analysis for All Transactions</td>
<td>CMRSA11N</td>
<td>produces a report with control breaks based on the T6ETRID field</td>
<td>“Performance Analysis for All Transactions report (CMRSA11N)” on page 123</td>
</tr>
<tr>
<td>Sample report title</td>
<td>Sample library member/program name</td>
<td>Description</td>
<td>Section reference</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Performance Analysis by Program     | CMRSA12N                           | produces a report with control breaks based on the T6EPGNNM field  
This report is useful for evaluating task performance subdivided by T6EPGNNM. This is most helpful when a Fourth Generation Language (4GL) is used. The report is almost identical to the report produced by CMRSA11N | “Performance Analysis by Program report (CMRSA12N)” on page 124                   |
| Performance Analysis by Terminal    | CMRSA13N                           | produces a report with control breaks based on the T6ETMID field  
This report is useful for evaluating task performance subdivided by terminal. The report is almost identical to the report produced by CMRSA11N | “Performance Analysis by Terminal report (CMRSA13N)” on page 125                   |
| Performance Analysis by VTAM Symbolic Node Name | CMRSA14N                           | produces a report with control breaks based on the T6ENETNM field  
This report is useful for evaluating task performance subdivided by VTAM symbolic node name. The report is almost identical to the report produced by CMRSA11N | “Performance Analysis by Node Name report (CMRSA14N)” on page 126                   |
| Performance Analysis by User ID     | CMRSA15N                           | produces a report with control breaks based on the T6EUSER field  
This report is useful for evaluating task performance subdivided by user ID. The report is almost identical to the report produced by CMRSA11N | “Performance Analysis by User ID report (CMRSA15N)” on page 126                    |
| File Analysis for All Files         | CMRSA21N                           | produces a report with control breaks based on the T6EFN field  
This report is useful for evaluating file performance by file name. | “File Analysis for All Files report (CMRSA21N)” on page 127                       |
| Rogues Report                      | CMRSA23N                           | produces a report that lists the tasks with the longest response times | “Rogues report (CMRSA23N)” on page 128                                           |
| Abended Transaction Roster         | CMRSA25N                           | produces a report of all transactions that had an abend during execution | “Abended Transaction Roster report (CMRSA25N)” on page 129                       |
| Operational Statistics Report      | CMRSA6F                            | produces a report that lists overall transaction performance across CICS regions | “Operational statistics report (CMRSA6F)” on page 130                             |
Control statements

Control statements are processed by the SAS programs to select data from specific CICS regions.

You also can provide a starting date and a starting time.

Use the following report control statements to select the data that appears in the SAS reports:

<table>
<thead>
<tr>
<th>Control statement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CICS</td>
<td>name of the CICS region providing data for the reports (for example, CICS=cicsprod)</td>
</tr>
<tr>
<td>DATE</td>
<td>processing start date in yyyyymmdd format (for example, DATE=19990924)</td>
</tr>
<tr>
<td>TIME</td>
<td>processing start time in hhmmss format (for example, TIME=082314)</td>
</tr>
</tbody>
</table>

SAS report fields

The following table describes the columnar data shown in the SAS sample reports in this chapter:

Table 17: SAS report fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CICS REGION</td>
<td>name of the CICS region associated with the data</td>
</tr>
<tr>
<td>RUN DATE</td>
<td>gregorian date associated with the data, in yyyy/mm/dd format</td>
</tr>
<tr>
<td>TRAN CODE</td>
<td>transaction identification of the CICS task</td>
</tr>
<tr>
<td>PROGRAM NAME</td>
<td>name of the program associated with the data</td>
</tr>
<tr>
<td>TERM ID</td>
<td>terminal identifier</td>
</tr>
<tr>
<td>NETNAME VTAM</td>
<td>symbolic node name</td>
</tr>
<tr>
<td>USER ID</td>
<td>identification of the CICS user</td>
</tr>
<tr>
<td>FILE NAME</td>
<td>name of the file</td>
</tr>
<tr>
<td>RUN TIME</td>
<td>task termination time associated with the data, in hh:mm:ss format</td>
</tr>
<tr>
<td>ABEND CODE</td>
<td>abend code of an abended transaction</td>
</tr>
<tr>
<td>TRAN COUNT</td>
<td>number of executed transactions</td>
</tr>
</tbody>
</table>
### Field Description

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AVERAGE CPU REAL</td>
<td>average amount of real CPU time used by each transaction (in seconds)</td>
</tr>
<tr>
<td>AVG FILE WAIT TIME</td>
<td>average time waiting for files during this transaction (in seconds)</td>
</tr>
<tr>
<td>AVG FILE CALLS</td>
<td>average number of file calls for each transaction</td>
</tr>
<tr>
<td>AVG DB WAIT TIME</td>
<td>average amount of wait time to access a database for each transaction</td>
</tr>
<tr>
<td>AVG DB CALLS</td>
<td>average number of database calls per transaction</td>
</tr>
<tr>
<td>AVERAGE STRG HWM</td>
<td>average storage high-water mark during this time interval</td>
</tr>
<tr>
<td>STORAGE OCCUPANCY</td>
<td>average storage usage in kilobytes</td>
</tr>
<tr>
<td></td>
<td>A value of 1 is equivalent to 250 bytes held for .25 seconds. This value</td>
</tr>
<tr>
<td></td>
<td>can be used to evaluate the impact on available storage caused by a</td>
</tr>
<tr>
<td></td>
<td>transaction.</td>
</tr>
<tr>
<td>AVG PROG STRG HWM</td>
<td>average program storage high-water mark for all transactions</td>
</tr>
</tbody>
</table>

### Performance Analysis for All Transactions report (CMRSA11N)

The CMRSA11N sample program produces the Performance Analysis for All Transactions report.

This report has control breaks based on the **T6ETRID** (transaction ID) field. See “Transaction detail data (CMRDETL)” on page 20 for information about the F7 record.

Generate this report to evaluate task performance subdivided by transaction.

### Sample report (Performance Analysis for All Transactions)

This topic shows a Performance Analysis for All Transactions sample report.

### Figure 37: Performance Analysis for All Transactions report

<table>
<thead>
<tr>
<th>CICS</th>
<th>REGION</th>
<th>RUN DATE</th>
<th>TRAN</th>
<th>TRANS</th>
<th>COUNT</th>
<th>RESPONSE</th>
<th>AVERAGE CPU REAL</th>
<th>AVERAGE CPU REAL</th>
<th>AVERAGE FILE WAIT TIME</th>
<th>AVERAGE FILE CALLS</th>
<th>AVG DB WAIT TIME</th>
<th>AVG DB CALLS</th>
<th>AVERAGE STRG HWM</th>
<th>STORAGE OCCUPANCY</th>
<th>AVG PROG STRG HWM</th>
</tr>
</thead>
<tbody>
<tr>
<td>CCSN</td>
<td>PRODC11C</td>
<td>1999/09/22</td>
<td>CATD</td>
<td>001</td>
<td>161</td>
<td>.011</td>
<td>.011</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
</tr>
<tr>
<td>CSSN</td>
<td>PRODC11C</td>
<td>1999/09/22</td>
<td>CMFD</td>
<td>001</td>
<td>161</td>
<td>.017</td>
<td>.017</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
</tr>
<tr>
<td>CSSN</td>
<td>PRODC11C</td>
<td>1999/09/22</td>
<td>CRSD</td>
<td>001</td>
<td>161</td>
<td>.017</td>
<td>.017</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
</tr>
<tr>
<td>CSSN</td>
<td>PRODC11C</td>
<td>1999/09/22</td>
<td>CSUO</td>
<td>001</td>
<td>161</td>
<td>.017</td>
<td>.017</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
</tr>
<tr>
<td>CSSN</td>
<td>PRODC11C</td>
<td>1999/09/22</td>
<td>CSIQ</td>
<td>001</td>
<td>161</td>
<td>.017</td>
<td>.017</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
</tr>
<tr>
<td>CSSN</td>
<td>PRODC11C</td>
<td>1999/09/22</td>
<td>CSSQ</td>
<td>001</td>
<td>161</td>
<td>.017</td>
<td>.017</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
</tr>
<tr>
<td>CSSN</td>
<td>PRODC11C</td>
<td>1999/09/22</td>
<td>CSSQ</td>
<td>001</td>
<td>161</td>
<td>.017</td>
<td>.017</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
</tr>
<tr>
<td>CSSN</td>
<td>PRODC11C</td>
<td>1999/09/22</td>
<td>CSSQ</td>
<td>001</td>
<td>161</td>
<td>.017</td>
<td>.017</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
</tr>
<tr>
<td>CSSN</td>
<td>PRODC11C</td>
<td>1999/09/22</td>
<td>CSSQ</td>
<td>001</td>
<td>161</td>
<td>.017</td>
<td>.017</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
</tr>
<tr>
<td>CSSN</td>
<td>PRODC11C</td>
<td>1999/09/22</td>
<td>CSSQ</td>
<td>001</td>
<td>161</td>
<td>.017</td>
<td>.017</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
</tr>
</tbody>
</table>
Performance Analysis by Program report (CMRSA12N)

The CMRSA12N sample program produces the Performance Analysis by Program report.

This report has control breaks based on the T6EPGNM (program name) field. See “Transaction detail data (CMRDETL)” on page 20 for information about the F7 record.

Generate this report to evaluate task performance subdivided by program name.

Note

This report is most useful when a fourth-generation language (4GL) is used.

Sample report (Performance Analysis by Program)

This topic shows a Performance Analysis by Program sample report.

Figure 38: Performance Analysis by Program report
Performance Analysis by Terminal report (CMRSA13N)

The CMRSA13N sample program produces the Performance Analysis by Terminal report.

This report has control breaks based on the T6ETMID (terminal ID) field. See “Transaction detail data (CMRDETL)” on page 20 for information about the F7 record.

Generate this report to evaluate task performance subdivided by terminal.

Sample report (Performance Analysis by Terminal)

This topic shows a Performance Analysis by Terminal sample report.

Figure 39: Performance Analysis by Terminal report

<table>
<thead>
<tr>
<th>PROGRAM: CMRSA13N</th>
<th>PERFORMANCE ANALYSIS BY TERMINAL</th>
<th>10:27 FRIDAY, SEPTEMBER 27, 1999</th>
</tr>
</thead>
<tbody>
<tr>
<td>EICCS</td>
<td>CICS</td>
<td>REGION</td>
</tr>
<tr>
<td>PRODCICS 1999/09/23</td>
<td>230</td>
<td>T2CINIT</td>
</tr>
<tr>
<td>PRODCICS 1999/09/23</td>
<td><em>TOTAL</em></td>
<td></td>
</tr>
<tr>
<td>PRODCICS 1999/09/23</td>
<td>2,741</td>
<td>1.447</td>
</tr>
</tbody>
</table>

Chapter 8 SAS reports 125
Performance Analysis by Node Name report (CMRSA14N)

The CMRSA14N sample program produces the Performance Analysis by Node Name report.

This sample report has control breaks based on the T6NETNM VTAM symbolic node name. See “Transaction detail data (CMRDETL)” on page 20 for information about the F7 record.

Generate this report to evaluate task performance subdivided by VTAM symbolic node name.

Sample report (Performance Analysis by VTAM Symbolic Node Name)

This topic shows a Performance Analysis by VTAM Symbolic Node Name sample report.

Figure 40: Performance Analysis by VTAM Symbolic Node Name report

<table>
<thead>
<tr>
<th>PROGRAM: CMRSA14N</th>
<th>PERFORMANCE ANALYSIS BY VTAM SYMBOLIC NODE NAME</th>
<th>10:27 FRIDAY, SEPTEMBER 27, 1999</th>
</tr>
</thead>
<tbody>
<tr>
<td>CICS REGION RUN DATE</td>
<td>CIC</td>
<td>AVERAGE RESPONSE</td>
</tr>
<tr>
<td>PRODCICS 1999/09/18</td>
<td>3,442</td>
<td>13.937</td>
</tr>
<tr>
<td>PRODCICS 1999/09/18</td>
<td>A100A02</td>
<td>79</td>
</tr>
<tr>
<td>PRODCICS 1999/09/18</td>
<td>A100A03</td>
<td>2</td>
</tr>
<tr>
<td>PRODCICS 1999/09/21</td>
<td>1,986</td>
<td>13.340</td>
</tr>
<tr>
<td>PRODCICS 1999/09/21</td>
<td>A05A03</td>
<td>8</td>
</tr>
<tr>
<td>PRODCICS 1999/09/21</td>
<td>A05A03</td>
<td>181</td>
</tr>
<tr>
<td>PRODCICS 1999/09/21</td>
<td>SMTC201</td>
<td>25</td>
</tr>
<tr>
<td>PRODCICS 1999/09/21</td>
<td>SMTC202</td>
<td>11</td>
</tr>
<tr>
<td>PRODCICS 1999/09/21</td>
<td><em>TOTAL</em></td>
<td>2,711</td>
</tr>
<tr>
<td>PRODCICS 1999/09/22</td>
<td>644</td>
<td>0.027</td>
</tr>
<tr>
<td>PRODCICS 1999/09/22</td>
<td>SMTC209</td>
<td>28</td>
</tr>
<tr>
<td>PRODCICS 1999/09/23</td>
<td>2,741</td>
<td>1.474</td>
</tr>
<tr>
<td>PRODCICS 1999/09/23</td>
<td>A05A02</td>
<td>2</td>
</tr>
<tr>
<td>PRODCICS 1999/09/23</td>
<td>A05A03</td>
<td>18</td>
</tr>
<tr>
<td>PRODCICS 1999/09/23</td>
<td>LCMA4E14</td>
<td>20</td>
</tr>
<tr>
<td>PRODCICS 1999/09/23</td>
<td>SMTCA206</td>
<td>9</td>
</tr>
<tr>
<td>PRODCICS 1999/09/23</td>
<td>SMTCA207</td>
<td>17</td>
</tr>
<tr>
<td>PRODCICS 1999/09/23</td>
<td>SMTCA204</td>
<td>163</td>
</tr>
<tr>
<td>PRODCICS 1999/09/23</td>
<td>SMTCA210</td>
<td>13</td>
</tr>
<tr>
<td>PRODCICS 1999/09/23</td>
<td><em>TOTAL</em></td>
<td>2,983</td>
</tr>
</tbody>
</table>

Performance Analysis by User ID report (CMRSA15N)

The CMRSA15N sample programs produce the Performance Analysis by User ID report.
This sample report has control breaks based on the **T6EUSER** (user ID) field. See “Transaction detail data (CMRDETL)” on page 20 for information about the F7 record.

Generate this report to evaluate task performance subdivided by user ID.

### Sample report (Performance Analysis by User ID report)

This topic shows a Performance Analysis by User ID sample report.

**Figure 41: Performance Analysis by User ID report**

<table>
<thead>
<tr>
<th>PROGRAM: CMRSA15N</th>
<th>PERFORMANCE ANALYSIS BY USER ID</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CICS</strong> REGION</td>
<td><strong>USER ID</strong></td>
</tr>
<tr>
<td>PRODCICS 1999/02/15</td>
<td>33</td>
</tr>
<tr>
<td>PRODCICS 1999/09/15 CICS</td>
<td>2,678</td>
</tr>
<tr>
<td>PRODCICS 1999/09/15 GMH01</td>
<td>174</td>
</tr>
<tr>
<td>PRODCICS 1999/09/16 <em>TOTAL</em></td>
<td>2,885</td>
</tr>
<tr>
<td>PRODCICS 1999/09/16 CICS</td>
<td>186</td>
</tr>
<tr>
<td>PRODCICS 1999/09/16 GMH01</td>
<td>3,409</td>
</tr>
<tr>
<td>PRODCICS 1999/09/16 OH@01</td>
<td>661</td>
</tr>
<tr>
<td>PRODCICS 1999/09/16 <em>TOTAL</em></td>
<td>4,256</td>
</tr>
<tr>
<td>PRODCICS 1999/09/17 CICS</td>
<td>1,204</td>
</tr>
<tr>
<td>PRODCICS 1999/09/17 GMH01</td>
<td>3,503</td>
</tr>
<tr>
<td>PRODCICS 1999/09/17 OH@01</td>
<td>661</td>
</tr>
<tr>
<td>PRODCICS 1999/09/17 <em>TOTAL</em></td>
<td>4,256</td>
</tr>
<tr>
<td>PRODCICS 1999/09/18 CICS</td>
<td>98</td>
</tr>
<tr>
<td>PRODCICS 1999/09/18 GMH01</td>
<td>3,425</td>
</tr>
<tr>
<td>PRODCICS 1999/09/18 OH@01</td>
<td>661</td>
</tr>
<tr>
<td>PRODCICS 1999/09/18 <em>TOTAL</em></td>
<td>672</td>
</tr>
<tr>
<td>PRODCICS 1999/09/19 CICS</td>
<td>233</td>
</tr>
<tr>
<td>PRODCICS 1999/09/19 GMH01</td>
<td>1967</td>
</tr>
<tr>
<td>PRODCICS 1999/09/19 OH@01</td>
<td>11</td>
</tr>
<tr>
<td>PRODCICS 1999/09/19 <em>TOTAL</em></td>
<td>3,523</td>
</tr>
<tr>
<td>PRODCICS 1999/09/20 CICS</td>
<td>2,719</td>
</tr>
<tr>
<td>PRODCICS 1999/09/20 GMH01</td>
<td>2,719</td>
</tr>
<tr>
<td>PRODCICS 1999/09/20 OH@01</td>
<td>3,523</td>
</tr>
<tr>
<td>PRODCICS 1999/09/20 <em>TOTAL</em></td>
<td>2,211</td>
</tr>
<tr>
<td>PRODCICS 1999/09/21 CICS</td>
<td>264</td>
</tr>
<tr>
<td>PRODCICS 1999/09/21 GMH01</td>
<td>2,719</td>
</tr>
<tr>
<td>PRODCICS 1999/09/21 OH@01</td>
<td>3,523</td>
</tr>
<tr>
<td>PRODCICS 1999/09/21 <em>TOTAL</em></td>
<td>2,211</td>
</tr>
</tbody>
</table>

**File Analysis for All Files report (CMRSA21N)**

The CMRSA21N sample program produces the File Analysis for All Files report.

This report has control breaks based on the **T6EFN** (file name) field. See “Transaction detail data (CMRDETL)” on page 20 for information about the F7 record.

Generate this report to evaluate file performance.
A CMRDETL data set may contain two forms of T6E records, original and expanded. Any program that you run against T6E file data needs to know which type of record it is processing. CMRSA21N uses the expanded record flag (field T6EQUAL) to determine whether a given record is expanded or not. This program also uses field T6EOFFFL as a pointer to the file information area. CMRSA21N illustrates how to process both original and expanded format records in the same file.

Sample report (File Analysis for All Files)

This topic shows a File Analysis for All Files sample report.

<table>
<thead>
<tr>
<th>PROGRAM: CMRSA21N</th>
<th>FILE ANALYSIS FOR ALL FILES</th>
<th>10:27 FRIDAY, SEPTEMBER 27, 1999</th>
</tr>
</thead>
<tbody>
<tr>
<td>CICS REGION RUN DATE</td>
<td>FILE NAME</td>
<td>TYPE TOTAL TRANS</td>
</tr>
<tr>
<td>PRODCICS 1999/09/23</td>
<td>CMRSTAT</td>
<td>CICS FCT</td>
</tr>
<tr>
<td>PRODCICS 1999/09/23</td>
<td>CUDCNLOG</td>
<td>CICS FCT</td>
</tr>
<tr>
<td>PRODCICS 1999/09/23</td>
<td>CUPFRO1</td>
<td>DB2</td>
</tr>
<tr>
<td>PRODCICS 1999/09/23</td>
<td>CUPINIT</td>
<td>DB2</td>
</tr>
<tr>
<td>PRODCICS 1999/09/23</td>
<td>CUPSSON</td>
<td>DB2</td>
</tr>
<tr>
<td>PRODCICS 1999/09/23</td>
<td>DFHRPL</td>
<td>PSEUDO</td>
</tr>
<tr>
<td>PRODCICS 1999/09/23</td>
<td>DFHUECB</td>
<td>PSEUDO</td>
</tr>
<tr>
<td>PRODCICS 1999/09/23</td>
<td>M2DCNLOG</td>
<td>CICS FCT</td>
</tr>
<tr>
<td>PRODCICS 1999/09/23</td>
<td>M2DPCT</td>
<td>CICS FCT</td>
</tr>
<tr>
<td>PRODCICS 1999/09/23</td>
<td>M2ISCNTL</td>
<td>CICS FCT</td>
</tr>
<tr>
<td>PRODCICS 1999/09/23</td>
<td>M2PACHP</td>
<td>DB2</td>
</tr>
<tr>
<td>PRODCICS 1999/09/23</td>
<td>M2PIB21</td>
<td>DB2</td>
</tr>
<tr>
<td>PRODCICS 1999/09/23</td>
<td>M2PIQ21</td>
<td>DB2</td>
</tr>
<tr>
<td>PRODCICS 1999/09/23</td>
<td>M2PPNT</td>
<td>DB2</td>
</tr>
<tr>
<td>PRODCICS 1999/09/23</td>
<td>M2MPCC1</td>
<td>DB2</td>
</tr>
<tr>
<td>PRODCICS 1999/09/23</td>
<td>M2MPCC6</td>
<td>DB2</td>
</tr>
<tr>
<td>PRODCICS 1999/09/23</td>
<td>M2MPCC21</td>
<td>DB2</td>
</tr>
<tr>
<td>PRODCICS 1999/09/23</td>
<td>M2MPMD21</td>
<td>DB2</td>
</tr>
<tr>
<td>PRODCICS 1999/09/23</td>
<td>M2MPMD21</td>
<td>DB2</td>
</tr>
<tr>
<td>PRODCICS 1999/09/23</td>
<td>M2MPMD21</td>
<td>DB2</td>
</tr>
<tr>
<td>PRODCICS 1999/09/23</td>
<td>M2MPRA01</td>
<td>DB2</td>
</tr>
<tr>
<td>PRODCICS 1999/09/23</td>
<td>M2MPRA12</td>
<td>DB2</td>
</tr>
<tr>
<td>PRODCICS 1999/09/23</td>
<td>M2MPRA01</td>
<td>DB2</td>
</tr>
<tr>
<td>PRODCICS 1999/09/23</td>
<td>M2MPRA01</td>
<td>DB2</td>
</tr>
<tr>
<td>PRODCICS 1999/09/23</td>
<td>M2MPRC21</td>
<td>DB2</td>
</tr>
<tr>
<td>PRODCICS 1999/09/23</td>
<td>M2MPRT21</td>
<td>DB2</td>
</tr>
<tr>
<td>PRODCICS 1999/09/23</td>
<td>M2MST21</td>
<td>DB2</td>
</tr>
<tr>
<td>PRODCICS 1999/09/23</td>
<td>M2MPSSON</td>
<td>DB2</td>
</tr>
<tr>
<td>PRODCICS 1999/09/23</td>
<td><em>TOTAL</em></td>
<td>DB2</td>
</tr>
</tbody>
</table>

Rogues report (CMRSA23N)

The CMRSA23N sample program creates the Performance Analysis for All Transactions report.

This report is also known as the Rogues Report. CMRSA23N produces the report by

- reading records from the CMRDETL file
- calculating task response times
Response times under 1.000 second are omitted from the report.

- producing a list of the tasks with the longest response times

Generate this report to evaluate tasks with the longest response times. Modify the report control statements in the JCL stream to limit the range of input data for the report.

See “Transaction detail data (CMRDETL)” on page 20 for information about the F7 record.

Sample report (SAS rogues)

This topic shows a SAS rogues sample report.

![Figure 43: SAS rogues report](image)

Abended Transaction Roster report (CMRSA25N)

The CMRSA25N sample program produces the Abended Transaction Roster report by

- reading records from the CMRDETL data set
- listing all transactions with an abend code
The CMRSA25N program fails and no report is created when no abended transactions are in the selected data.

See “Transaction detail data (CMRDETL)” on page 20 for information about the F7 record.

Sample report (Abended Transaction Roster)

This topic shows an Abended Transaction Roster sample report.

**Figure 44: Abended Transaction Roster report**

<table>
<thead>
<tr>
<th>PROGRAM: CMRSA25N</th>
<th>ABENDED TRANSACTION ROSTER</th>
<th>10:27 FRIDAY, SEPTEMBER 06, 1999</th>
</tr>
</thead>
<tbody>
<tr>
<td>CICS</td>
<td>TRANSACTION</td>
<td>DATABASE</td>
</tr>
<tr>
<td>REGION</td>
<td>RUN DATE</td>
<td>RUN TIME</td>
</tr>
<tr>
<td>PRODCICS 1999/09/05</td>
<td>18:09:07</td>
<td>18:09:07</td>
</tr>
<tr>
<td>PRODCICS 1999/09/05</td>
<td>18:11:52</td>
<td>18:11:52</td>
</tr>
<tr>
<td>PRODCICS 1999/09/15</td>
<td>13:33:20</td>
<td>13:33:20</td>
</tr>
<tr>
<td>PRODCICS 1999/09/15</td>
<td>13:41:31</td>
<td>13:41:31</td>
</tr>
<tr>
<td>PRODCICS 1999/09/15</td>
<td>15:52:03</td>
<td>15:52:03</td>
</tr>
<tr>
<td>PRODCICS 1999/09/08</td>
<td>13:02:52</td>
<td>13:02:52</td>
</tr>
<tr>
<td>PRODCICS 1999/09/08</td>
<td>14:03:19</td>
<td>14:03:19</td>
</tr>
<tr>
<td>PRODCICS 1999/09/08</td>
<td>14:15:09</td>
<td>14:15:09</td>
</tr>
<tr>
<td>PRODCICS 1999/09/08</td>
<td>14:53:43</td>
<td>14:53:43</td>
</tr>
</tbody>
</table>

Operational statistics report (CMRSAS6F)

The CMRSAS6F sample program generates a report that lists overall transaction performance across CICS regions.

The program produces the Operational Statistics report by

- reading records from the CMRSTATS data set
- reporting the CPU resource usage and response times of transactions by CICS regions

The program variable PERIOD controls the processing of CMRSTATS records and the appearance of resulting performance statistics shown in the CMRSAS6F report. Based upon the value of the PERIOD variable, performance statistics produced by each transaction record can be listed as individual report lines or they can be...
summarized over several fixed intervals. Permissible values for the PERIOD variable are as follows:

- PERIOD=1 produces a report with one print line for each input record.
- PERIOD=xx produces a report with one print line for each xx minutes of CICS execution time, where xx may be 10, 15, 20, 30, or 60 minutes. The default is 15 minutes.

**Sample report (Operational statistics)**

This topic shows an Operational statistics sample report.

**Figure 45: Operational statistics report**

<table>
<thead>
<tr>
<th>REGION</th>
<th>RUN DATE</th>
<th>RUN TIME</th>
<th>ALL TRANSACTIONS</th>
<th>CPU REAL</th>
<th>AVERAGE CPU REAL</th>
<th>TERMINAL TRANSACTIONS</th>
<th>RESPONSE</th>
<th>AVG. TERM</th>
<th>ACTIVE TASKS</th>
</tr>
</thead>
<tbody>
<tr>
<td>CICSPROD</td>
<td>1999/09/09</td>
<td>08/00/00</td>
<td>118</td>
<td>4.884</td>
<td>0.041</td>
<td>83</td>
<td>0.355</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>CICSPROD</td>
<td>1999/09/09</td>
<td>08/15/00</td>
<td>32</td>
<td>0.844</td>
<td>0.026</td>
<td>1</td>
<td>0.008</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>CICSPROD</td>
<td>1999/09/09</td>
<td>08/30/00</td>
<td>37</td>
<td>1.205</td>
<td>0.033</td>
<td>5</td>
<td>1.165</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>CICSPROD</td>
<td>1999/09/09</td>
<td>08/45/00</td>
<td>31</td>
<td>0.901</td>
<td>0.029</td>
<td>5</td>
<td>0.254</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>CICSPROD</td>
<td>1999/09/09</td>
<td>09/00/00</td>
<td>31</td>
<td>0.896</td>
<td>0.029</td>
<td>0</td>
<td>0.000</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>CICSPROD</td>
<td>1999/09/09</td>
<td>09/15/00</td>
<td>32</td>
<td>0.872</td>
<td>0.027</td>
<td>2</td>
<td>0.100</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>CICSPROD</td>
<td>1999/09/09</td>
<td>09/30/00</td>
<td>32</td>
<td>0.877</td>
<td>0.027</td>
<td>0</td>
<td>0.000</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>CICSPROD</td>
<td>1999/09/09</td>
<td>09/45/00</td>
<td>24</td>
<td>0.865</td>
<td>0.036</td>
<td>0</td>
<td>0.000</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>CICSPROD</td>
<td>1999/09/09</td>
<td>10/00/00</td>
<td>32</td>
<td>0.903</td>
<td>0.028</td>
<td>0</td>
<td>0.000</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>CICSPROD</td>
<td>1999/09/09</td>
<td>10/15/00</td>
<td>31</td>
<td>0.862</td>
<td>0.028</td>
<td>0</td>
<td>0.000</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>CICSPROD</td>
<td>1999/09/09</td>
<td>10/30/00</td>
<td>31</td>
<td>0.860</td>
<td>0.028</td>
<td>0</td>
<td>0.000</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>CICSPROD</td>
<td>1999/09/09</td>
<td>10/45/00</td>
<td>32</td>
<td>0.844</td>
<td>0.026</td>
<td>0</td>
<td>0.000</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>CICSPROD</td>
<td>1999/09/09</td>
<td>11/00/00</td>
<td>24</td>
<td>0.894</td>
<td>0.037</td>
<td>0</td>
<td>0.000</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>CICSPROD</td>
<td>1999/09/09</td>
<td>11/15/00</td>
<td>32</td>
<td>0.862</td>
<td>0.027</td>
<td>0</td>
<td>0.000</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>CICSPROD</td>
<td>1999/09/09</td>
<td>11/30/00</td>
<td>31</td>
<td>0.860</td>
<td>0.028</td>
<td>0</td>
<td>0.000</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>CICSPROD</td>
<td>1999/09/09</td>
<td>11/45/00</td>
<td>31</td>
<td>0.829</td>
<td>0.027</td>
<td>0</td>
<td>0.000</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>CICSPROD</td>
<td>1999/09/09</td>
<td>12/00/00</td>
<td>32</td>
<td>0.890</td>
<td>0.028</td>
<td>0</td>
<td>0.000</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>CICSPROD</td>
<td>1999/09/09</td>
<td>12/15/00</td>
<td>30</td>
<td>1.444</td>
<td>0.048</td>
<td>12</td>
<td>0.515</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>CICSPROD</td>
<td>1999/09/09</td>
<td>12/30/00</td>
<td>31</td>
<td>0.692</td>
<td>0.022</td>
<td>0</td>
<td>0.000</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>CICSPROD</td>
<td>1999/09/09</td>
<td>12/45/00</td>
<td>32</td>
<td>0.827</td>
<td>0.026</td>
<td>0</td>
<td>0.000</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>CICSPROD</td>
<td>1999/09/09</td>
<td>13/00/00</td>
<td>31</td>
<td>0.861</td>
<td>0.028</td>
<td>4</td>
<td>1.571</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>CICSPROD</td>
<td>1999/09/09</td>
<td>13/15/00</td>
<td>31</td>
<td>0.698</td>
<td>0.023</td>
<td>0</td>
<td>0.000</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>CICSPROD</td>
<td>1999/09/09</td>
<td>13/30/00</td>
<td>32</td>
<td>0.595</td>
<td>0.019</td>
<td>0</td>
<td>0.000</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>CICSPROD</td>
<td>1999/09/09</td>
<td>13/45/00</td>
<td>24</td>
<td>0.803</td>
<td>0.033</td>
<td>0</td>
<td>0.000</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>CICSPROD</td>
<td>1999/09/09</td>
<td>14/00/00</td>
<td>32</td>
<td>0.641</td>
<td>0.020</td>
<td>0</td>
<td>0.000</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

**Column descriptions**

Table 18 on page 131 describes the columnar data specific to Figure 45 on page 131.

**Table 18: Operational statistics report fields**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL TRANSACTIONS</td>
<td>total number of transactions</td>
</tr>
<tr>
<td>TOTAL CPU REAL</td>
<td>total amount of CPU real time used during this transaction</td>
</tr>
<tr>
<td>AVERAGE CPU REAL</td>
<td>average amount of CPU real time used during this transaction</td>
</tr>
<tr>
<td>TERMINAL TRANSACTIONS</td>
<td>number of transactions executed from this terminal during this time interval</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>AVG. TERM RESPONSE</td>
<td>average response time per transactions executed from this terminal during this time interval</td>
</tr>
<tr>
<td>MAXIMUM ACTIVE TASKS</td>
<td>AMXT value from the CICS region at the time this data is collected</td>
</tr>
</tbody>
</table>
User-defined reports (CMRPRLN)

You can create custom reports by writing a program with the MainView for CICS Performance Reporter Language (PRL). With PRL, you select records from the CMRDETL, CTGDETL, and CMRSTATS data sets that hold CICS transaction, CTG transaction, and statistical data. Your program determines how these records are processed and formatted in a printed report.

You create PRL reports by executing a batch job that runs the CMRPRLN program and its associated control statements. CMRPRLN processes these user-written PRL control statements to select the input VSAM records and to define the format of the report.

The Performance Reporting Language supports

- CMRPRLN processing of the F7 format of the CMRDETL history record using the CMRDETLN dictionary format
- CMRPRLN processes the CTGDETL records using the CTGDETL dictionary format
- CMRPRLN processes the CMRSTATS records using the CMRSTATS dictionary format

The most efficient way to execute PRL programs is by running them against small volumes of data during CICS execution or at daily CICS shutdown. Long-term trends can be reported for analysis or debugging by archiving the MainView for CICS-collected data to tape (see “Archiving data (CMRPURG)” on page 25), and then specifying the tape as input to the PRL program (see “Selecting archive data” on page 183). Sample PRL programs for CMRDETL and CMRSTATS are located in your BBSAMP library. They are named CMRPRLnn, where nn is a two-digit suffix that identifies an individual report.

You should execute the sample programs to learn PRL and to obtain copies of the default reports. The JCL for submitting the programs is described in “Generating PRL reports” on page 139.
Performance Reporting Language PRL

A typical PRL program consists of the following statements:

### Table 19: PRL statements

<table>
<thead>
<tr>
<th>Statement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Verbs</td>
<td>perform an action</td>
</tr>
<tr>
<td>Operators</td>
<td>symbolically indicate the action to be performed in a mathematical expression</td>
</tr>
<tr>
<td>Substrings</td>
<td>exist for character string evaluation</td>
</tr>
<tr>
<td>Variables</td>
<td>exist for value substitution</td>
</tr>
<tr>
<td></td>
<td>Variables can be symbolic for user-defined data or predefined for system-provided data.</td>
</tr>
</tbody>
</table>

You write these statements in a PRL program to generate a user-defined batch report. The statements can assign a value to a variable, establish a condition, or control PRL execution.

A PRL program is created with an interactive editor as a member of a data set. The member can be part of a PRL procedure library. BMC Software distributes sample PRL programs as members in the BBSAMP data set.

The program is executed as a batch job that defines the input data and type of report produced.

### Samples

The following table lists the PRL sample programs for CMRDETL and CMRSTATS distributed with MainView for CICS.

### Table 20: PRL samples

<table>
<thead>
<tr>
<th>BBSAMP member</th>
<th>Report type</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMRPRL0</td>
<td>resource usage summary for all transactions</td>
</tr>
<tr>
<td>CMRPR10</td>
<td>service level analysis report for terminal attached transactions</td>
</tr>
<tr>
<td>CMRPR11</td>
<td>file control and suspend time analysis for terminal attached transactions</td>
</tr>
<tr>
<td>CMRPR12</td>
<td>transaction activity graph</td>
</tr>
<tr>
<td>BBSAMP member</td>
<td>Report type</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------</td>
</tr>
<tr>
<td>CMRPRL13</td>
<td>terminal errors</td>
</tr>
<tr>
<td>CMRPRL14</td>
<td>response time analysis for terminal attached transactions</td>
</tr>
<tr>
<td>CMRPRL15</td>
<td>summary of activity for a specific program</td>
</tr>
</tbody>
</table>
| CMRPRL16      | - transaction file control wait analysis  
                   - file wait analysis by field  
                   - transaction to file summary  
                   - dataset activity and placement analysis |
| CMRPRL17      | transaction response time analysis for MRO data |
| CMRPRL18      | - transaction response time analysis  
                   - hourly report of transactions processed  
                   - summary of transaction processing  
                   - summary of transactions within response time |
| CMRPRL19      | total counts by record types from CMRDETL and CMRSTATS files |
| CMRPRL20      | transaction response analysis by response range |
| CMRPRL21      | I/O Activity report in descending I/O response by transaction |
| CMRPRL24      | DB2 plan usage report |
| CMRPRL25      | SQL activity report by DB2 Plan |
| CMRPRL27      | Task exception analysis report |
| CMRPRL28      | summarized data from CMRSUMDN |
| CMRPRL3       | transaction analysis for all transactions |
| CMRPRL30      | DB2 data from variable file entries |
| CMRPRL31      | MQ data from variable file entries |
| CMRPRL32      | DBCTL data from detail file entries when file expansion is turned on |
| CMRPRL33      | session data from variable resource entries |
| CMRPRL34      | program timing data from variable resource entries |
| CMRPRL35      | - transaction/connection analysis  
                   - connection analysis by connection ID  
                   - transaction to connection summary |
<p>| CMRPRL36      | sample transient data queue analysis |
| CMRPRL37      | sample temporary storage queue analysis |
| CMRPRL4       | highest 20 users of CPU time for a given date |
| CMRPRL40      | specific transaction response time by date for a single CICS target |
| CMRPRL41      | transaction summary for the entire day, summarized by CICS target |</p>
<table>
<thead>
<tr>
<th>BBSAMP member</th>
<th>Report type</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMRPRL42</td>
<td>transaction summary for the entire day, summarized by CICS target, with transaction activity for a labeled interval of time</td>
</tr>
<tr>
<td>CMRPRL43</td>
<td>average response time over several days for a single CICS target</td>
</tr>
<tr>
<td>CMRPRL44</td>
<td>average response time over several days for multiple CICS targets</td>
</tr>
<tr>
<td>CMRPRL45</td>
<td>average response time over several days, with an hourly breakdown for a specified time period</td>
</tr>
<tr>
<td>CMRPRL46</td>
<td>sample report to produce CSV output</td>
</tr>
<tr>
<td>CMRPRL5</td>
<td>abended transactions list</td>
</tr>
<tr>
<td>CMRPRL6</td>
<td>service level analysis report</td>
</tr>
<tr>
<td>CMRPRL7</td>
<td>service level analysis graph</td>
</tr>
<tr>
<td>CMRPRL8</td>
<td>summary of terminal attached transactions at 10-minute intervals for any given date</td>
</tr>
<tr>
<td>CMRPX0BD</td>
<td>program control detail report (CMRSTATS)</td>
</tr>
<tr>
<td>CMRPX0BS</td>
<td>program control summary report (CMRSTATS)</td>
</tr>
<tr>
<td>CMRPX0C</td>
<td>transaction class report (CMRSTATS)</td>
</tr>
<tr>
<td>CMRPX22D</td>
<td>terminal control detail report (CMRSTATS)</td>
</tr>
<tr>
<td>CMRPX22S</td>
<td>terminal control summary report (CMRSTATS)</td>
</tr>
<tr>
<td>CMRPX30D</td>
<td>temporary storage detail report (CMRSTATS)</td>
</tr>
<tr>
<td>CMRPX30S</td>
<td>temporary storage summary report (CMRSTATS)</td>
</tr>
<tr>
<td>CMRPX43D</td>
<td>file control detail report (CMRSTATS)</td>
</tr>
<tr>
<td>CMRPX43S</td>
<td>file control summary report (CMRSTATS)</td>
</tr>
</tbody>
</table>

### Expressions and operators

Use any arithmetic, comparative, or logical operator in a statement expression. An expression combines variables, whole numbers, and character strings with operators. Operators are as follows:

<table>
<thead>
<tr>
<th>Operator</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>=</td>
<td>equal to</td>
</tr>
<tr>
<td>&gt;</td>
<td>greater than</td>
</tr>
<tr>
<td>&lt;</td>
<td>less than</td>
</tr>
<tr>
<td>¬</td>
<td>not</td>
</tr>
<tr>
<td>+</td>
<td>plus</td>
</tr>
</tbody>
</table>
### Substrings

Substrings are portions of a character string.

They are selected by specifying a numeric range enclosed in parentheses such as

\((\text{start-}n:\text{length-}n)\)

where

\((\text{start-}n:\text{length-}n)\) is an expression that specifies a count of alphanumeric characters from the beginning of a string and a length:

- \(\text{start-}n\) specifies the start of the substring as indicated by \(n\), which is the number of alphanumeric characters from the beginning of a string or referenced field.
- \(\text{length-}n\) specifies the number of alphanumeric characters for the substring as indicated by \(n\). This is the length of the substring.

### Examples

The following example selects the fourth character of a string and specifies a substring length of three characters.

The substring, FEB, is assigned to the M2 variable.

```
Example
SET Q1 = 'JANFEBMAR'
SET M2 = Q1(4:3)
```

The next example shows how substrings can be used to determine a value, in this case the name of the month:
Example

```plaintext
SET MONS = 'JANFEBMARAPRMAYJUNJULAUGSEPCTNOVDEC'
SET MM   = ZDATE(1:2)
SET V1   = MM - 1
SET V2   = V1 * 3
SET V3   = V2 + 1
SET MNTH = MONS(&V3:3)
```

The substring expression ZDATE(1:2) specifies the first character, for a length of two characters, of the mm/dd/yyyy value passed to ZDATE (03 for example). The substring expression, MONS(&V3:3), selects the portion of the character string determined by the value of the V3 variable and limits the substring to 3 characters. If 03/15/2000 is passed to the ZDATE predefined variable, the variables are assigned the following values:

<table>
<thead>
<tr>
<th>Variable</th>
<th>Assigned value</th>
</tr>
</thead>
<tbody>
<tr>
<td>V1</td>
<td>2</td>
</tr>
<tr>
<td>V2</td>
<td>6</td>
</tr>
<tr>
<td>V3</td>
<td>7</td>
</tr>
<tr>
<td>MNTH</td>
<td>MAR</td>
</tr>
</tbody>
</table>

**Symbolic variables**

PERFORMANCE REPORTER supports symbolic variables.

Variables are symbolic characters used as a substitute for a real value. They are predefined or user-defined. User-defined variables can be any alphanumeric character string that is eight bytes long and starts with an alphabetic character. A symbolic variable name can be preceded by an ampersand (&) to distinguish it from a literal, as described in “PRL control statement syntax” on page 150.

**Predefined variables**

PRL provides predefined, reserved variables.

Use these variables to obtain information about the current CICS environment. The following descriptions of predefined variables show variable characters in uppercase, to indicate that these exact characters must be specified for that variable.
**CDATE (Day Date Month, Year)**

This variable passes the current date in the form:

Thursday 12th September, 2006

It can be used to look for certain conditions on a specific day of the week or month, or to print a more descriptive title. An example follows:

```plaintext
SET TITLE = 'TRANSACTION DAILY REPORT'
SET TITLE1 = &CDATE
```

**ZDATE (mm/dd/yyyy)**

The ZDATE variable passes the current date, expressed as *mm/dd/yyyy*. An example follows:

```plaintext
SELECT TYPE 6E RECORDS FROM CMRDETL
WHERE CMRDATE = &ZDATE
```

**ZTIME (hh:mm:ss)**

The ZTIME variable passes the current time, expressed as *hh:mm:ss*. An example follows:

```plaintext
PRINT 'REPORT GENERATED AT' &ZTIME
```

The following section describes how to create PRL CICS performance reports.

---

### Generating PRL reports

Sample JCL to run a PRL program can be found in the following BBSAMP data set members:

- **CMRL** is used to generate a report from a single CMRDETL VSAM disk, or one or more tape data sets.

- **CMRMVJCL** is used to generate a report from multiple, active VSAM CMRDETL data sets; or one or more tape data sets.

The following figure shows the sample JCL found in CMRL.

**Figure 46: CMRL JCL used for PRL batch execution**

```plaintext
//jobname JOB user parameters
/*/JOBPARM user parameters
//CMRL PROC
```

---

Chapter 9 User-defined reports (CMRPRLN) 139
The following figure shows the sample JCL found in CMRMVJCL.

**Figure 47: CMRMVJCL JCL used for PRL batch execution**

---

**Job control statements**

The JCL for PRL batch execution is used as follows:

- //procname PROC indicates procedural JCL. Defaults can be assigned to parameters in the procedure.

- //stepname EXEC specifies the name of the program that processes user-written PRL statements, which is PGM=CMRPRLN.

---

It might be necessary to specify a region size of 2048K.
//STEPLIB defines the program library containing the MainView for CICS load modules (BBLINK).

**Note**
Alternatively, you can specify a //JOBLIB statement.

//SYSPRINT defines the output class for a sort utility. The name of the DD statement is determined at CICS system generation.

//SYSUDUMP defines the output for a job that abnormally terminates.

//SYSOOT defines the output class.

//ddname is optional and is the name of the Data Definition (DD) statement. The ddname must start with an alphabetic character, can comprise any alphanumeric characters, and is from one to eight characters in length. The DD statement defines the output for a user-defined report. The report is allocated as defined by the DD statement if the ddname is the same as the value specified by the PRL SET REPORTID statement, as described in “SET (assign a value)” on page 168.

Multiple DD statements can be used to direct the output of multiple reports if the ddname for each output allocation is the same as a REPORTID value, as described in “Routing report output” on page 179.

**Note**
A report DD statement, if used, must have DCB specifications that correspond to the report generated (the LRECL size must be large enough to accommodate all the variables specified with the USING statement).

A DD statement is not needed for report output allocation. If the DD statement is omitted and SET CLASS or SET FORM is not used, PERFORMANCE REPORTER allocates a default output class and form number. The defaults are A for the output class and STD for the form number.

//ISYPROG defines an output class for the Performance Reporting Language program listing.

//ISYSOUT defines an output class for the messages.

//PNLLIB defines the data set (BBPLIB) containing the data dictionary (CMRDETLN, CTGDETL, and CMRSTATS members). PERFORMANCE REPORTER uses the data dictionary to generate report column headings for the CMRDETL, CTGDETL, and CMRSTATS records that you select, as described in “Predefined data dictionary descriptions” on page 181.

//CMRSTATS defines a VSAM disk or tape data set containing records of CICS operational statistics (CMRSTATS).
//CMRDETL defines a VSAM disk or tape data set containing records for each CICS transaction (CMRDETL).

//CMRDETnn defines additional VSAM disk or tape data sets that contain records for each CICS transaction (CMRDETL), where nn is 01 to 99 and must be specified as consecutive numbers.

CMRDETLnn can be used to generate PRL reports for multiple CICS targets by specifying multiple, active VSAM CMRDETL files as input.

All of the files defined by the CMRDETL DD and CMRDETLnn DD statements must be the same format (either all VSAM or all sequential).

For sample JCL, see the BSAMP member CMRMVJCL.

---

Note

The order of the CMRDETLnn DD statements is not important, but the CMRDETLnn suffix numbers must be consecutive. The report program stops processing the DD statements when it cannot find the next consecutively numbered CMRDETLnn DD statement.

---

//CTGDETL defines a VSAM disk or tape data set containing records for each CTG transaction (CTGDETL).

//CTGDETnn defines additional VSAM disk or tape data sets that contain records for each CTG transaction (CTGDETL), where nn is 01 to 99 and must be specified as consecutive numbers.

CTGDETLnn can be used to generate PRL reports for multiple CTG targets by specifying multiple, active VSAM CMRDETL files as input.

All of the files defined by the CTGDETL DD and CTGDETLnn DD statements must use the same format (either all VSAM or all sequential).

//SORTWKnn identifies one to nine work data sets that can be defined for data sorting. Define nn as a numeric value from 01 to 09. A minimum of one work data set is required for each user-written report.

//SYSIN defines PRL control statements as input to the batch program.

---

Basic PRL verbs used to create a report

The following verbs generate a basic PRL report.

For a complete description of each PRL verb, refer to “PRL control statement verbs” on page 149.
Table 21: PRL verbs

<table>
<thead>
<tr>
<th>Verb</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>SET</td>
<td>sets the report structure and defines any variables to be processed and reported with the CMRSTATS or CMRDETL data</td>
</tr>
<tr>
<td>SELECT</td>
<td>specifies the CMRSTATS or CMRDETL data to be processed</td>
</tr>
<tr>
<td>USING</td>
<td>specifies the data to be reported</td>
</tr>
<tr>
<td>REPORT</td>
<td>reports the data until end of file</td>
</tr>
<tr>
<td>END</td>
<td>terminates the PRL program</td>
</tr>
</tbody>
</table>

You can select and report abend transaction data by combining the basic statements shown in Figure 48 on page 143 with the JCL described in “Job control statements” on page 140.

Figure 48: Example PRL verbs for creating a report

```plaintext
SET TITLE = 'TRANSACTIONS THAT ABENDED'
SELECT TYPE 6D RECORDS FROM CMRDETL
USING T6ETRID -
    CMRDATE -
    CMRTIME -
    T6EABCD -
    T6EOPID -
    T6ETMID -
    T6ERESP -
    T6ECPUR
REPORT
END
```

PRL processing

As shown in the previous section, some PRL statements are required for proper report execution.

The use of other PRL statements depends upon what you want your report to do. This section describes how PERFORMANCE REPORTER processes a PRL program.

Logic flow

The sequence that follows shows the order in which PRL statements are processed.

Those steps that are noted as optional are user dependent. If the PRL described in an optional step is used, it should be written as sequenced.

1 (optional) Set the titles and size of your report.
2 (optional) Set any variables that you want as constants.

A value assigned to a variable before a SELECT statement does not change unless the variable is used again after the SELECT statement.

Note
If DETAIL, SUMMARY, or USING will be used with a COMPUTE or SET statement, the variable should be defined first (the DEFINE statement).

3 Select the record types as input to your report using the SELECT statement.

4 (optional) Qualify record selection with a WHERE statement.

5 (optional) Evaluate data with a COMPUTE statement.

6 Specify the data to be reported with a USING statement.

7 (optional) Sort the data with ORDER BY.

If DETAIL or SUMMARY is specified, the first variable of ORDER BY must be the same as the first variable of the USING statement. The records are selected and ordered until end of file, and then reported.

8 (optional) Tabulate the data with DETAIL or SUMMARY.

DETAIL or SUMMARY causes a break in the reporting whenever the first variable of the USING statement changes. A summary is computed at each break.

9 Report each selected record using the REPORT statement.

Any variables defined after a SELECT statement and before a REPORT statement are assigned a specified value every time a record is reported and a new record selected, until end of file.

10 (optional) Print footnotes, calculated variables, or comments after all records are reported using the PRINT statement.

Note
PRINT is processed wherever it is used in the PRL program.

11 End the PRL program (the END statement).
Record processing

PERFORMANCE REPORTER processes records chosen in the SELECT statement according to the PRL verbs used between the SELECT and REPORT statements until the end of file is reached.

These records are processed in chronological order unless an ORDER BY statement is used. If an ORDER BY statement is used, the processing takes place according to the PRL verbs used between the SELECT and ORDER BY statements, and the resulting data is processed by the REPORT statement. The END statement specifies the end of file.

Reporting selected data

SELECT requires a REPORT statement to show the results of each record processed by the PRL statements between SELECT and REPORT, until there are no more records.

“Reporting selected data” on page 145 shows an example.

Figure 49: Detail file control table analysis—sample PRL

```
SET TITLE = 'ANALYSIS OF FILE CONTROL TABLE'
SET WIDTH = 132
DEFINE FILFNAM  CHAR 08
DEFINE FILDSRD  EDIT ZZZZ,ZZ9
DEFINE FILDSWRU EDIT ZZZZ,ZZ9
DEFINE FILDSWRA EDIT ZZZZ,ZZ9
DEFINE FILDSDEL EDIT ZZZ,ZZ9
DEFINE FILDSXCP EDIT ZZZ,ZZ9
DEFINE FILDSIXP EDIT ZZZ,ZZ9
*
* ADD STARTING AT CLAUSE TO BEGIN AT A SPECIFIC DATE AND TIME
* 
SELECT TYPE 43 RECORDS FROM CMRSTATS
WHERE FILDSRD > 0
OR FILDSWRU > 0
OR FILDSWRA > 0
OR FILDSDEL > 0
USING CMRDATE -
   CMRTIME -
   FILFNAM -
   FILDSRD -
   FILDSWRU -
   FILDSWRA -
   FILDSDEL -
   FILDSXCP -
   FILDSIXP
ORDER BY FILFNAM
DETAIL
REPORT
```
The PRL statements shown in “Reporting selected data” on page 145 search the CMRSTATS data set for records identified by hexadecimal 43 in the TYPE field. The statements then report each specified field until end of file.

**Sorting and reporting data**

Data shown in PRL reports is in chronological order.

Use the ORDER BY statement to change the order. ORDER BY sorts the data by the record variable specified with the statement.

The data is arranged in ascending order (low to high), unless IN DESCENDING SEQUENCE (high to low) is specified. Adding ORDER BY before REPORT in the preceding example, such as ORDER BY FCTNAME, processes each record between SELECT and ORDER BY until end of file, and then sorts each record by file name (FCTNAME) and reports the sorted data. Only one ORDER BY variable can be used per report set.

---

**Note**

The ORDER BY variable should be the same as the first USING variable if SUMMARY is used after ORDER BY. If it is not, unexpected results can occur.

---

**Example Reporting average terminal response time**

The example in Figure 50 on page 147 shows the use of PRL to average and report terminal response time.
The reported results are shown in Figure 51 on page 148.

**Figure 50: Sample PRL using the COMPUTE statement**

```
** REPORT1 ***************************************************************
*                                                                      *
* RESPONSE TIME SUMMARY FOR ALL TERMINAL ATTACHED TRANSACTIONS -       *
* BASED ON THE CC RECORD TYPE 5 MINUTE RECORDING INTERVAL -            *
************************************************************************
** PRINT CONTROL SECTION ***********************************************
*                                                                      *
************************************************************************
SET CASE     = 'UPPER'
SET CLASS    = 'R'
SET FORM     = 'STD '
SET LINES    = 50
SET REPORTID = 'REPORT1'
SET WIDTH    = '132'
SET TITLE    = 'TRANSACTION RESPONSE TIME'
SET TITLE1   = 'Terminal-Attached Transactions'
SET TITLE2   = '
** USER DEFINED VARIABLES **********************************************
*                                                                      *
* VARIABLES DEFINED BEFORE THE SELECT STATEMENT WILL ONLY BE           *
* INITIALIZED ONCE.                                                    *
*                                                                      *
************************************************************************
************************************************************************
* INCLUDE A RANGE OF DAYS. DAYS AFTER 09/11, IE 09/12. DAYS BEFORE     *
* 09/14, IE 09/13.                                                     *
************************************************************************
SET FDATE    = '09/19/1999'
SET LDATE    = '09/22/1999'
************************************************************************
* INCLUDE A RANGE OF TIMES. HOURS AFTER 7:59 AM AND HOURS BEFORE       *
* 8:00 PM.                                                            *
************************************************************************
SET FTIME    = '07:59:00'
SET LTIME    = '20:00:01'
** SELECT CONTROL SECTION ***********************************************
*                                                                      *
* BEGIN LOOP: THE SELECT STATEMENT BEGINS A REPORT LOOP FOR EACH        *
* RECORD UNTIL END OF FILE.                                            *
*                                                                      *
************************************************************************
SELECT TYPE CC RECORDS FROM CMRSTATS
WHERE CMRDATE > FDATE
AND   CMRDATE < LDATE
AND   CMRTIME > FTIME
AND   CMRTIME < LTIME
** USER AND OTHER PRL VARIABLES ***************************************
*                                                                      *
* VARIABLES DEFINED HERE WILL BE ASSIGNED EVERY TIME A RECORD IS       *
* SELECTED.                                                           *
*                                                                      *
************************************************************************
COMPUTE AVTRMRSP = T6FRSPTM / T6FTRMCT
USING CMRTIME AVTRMRSP T6FTRMCT T6FSSHWM T6FSPGIN T6FSPGOU
REPORT END
```
Figure 51 on page 148 is the result of using **COMPUTE** as shown in the PRL example in Figure 50 on page 147.

**Figure 51: Terminal-attached transaction response time report**

<table>
<thead>
<tr>
<th>RECORDED TIME</th>
<th>AVTRMRSP</th>
<th>TERMINAL TASKS</th>
<th>DSA HWM</th>
<th>PAGE-IN COUNT</th>
<th>PAGEOUT COUNT</th>
</tr>
</thead>
<tbody>
<tr>
<td>12:33:00</td>
<td>0.00000</td>
<td>0</td>
<td>1310</td>
<td>165</td>
<td>26</td>
</tr>
<tr>
<td>12:38:01</td>
<td>0.00000</td>
<td>0</td>
<td>1310</td>
<td>172</td>
<td>31</td>
</tr>
<tr>
<td>12:43:00</td>
<td>0.25196</td>
<td>32</td>
<td>1321</td>
<td>312</td>
<td>27</td>
</tr>
<tr>
<td>12:48:00</td>
<td>0.07513</td>
<td>118</td>
<td>1404</td>
<td>215</td>
<td>67</td>
</tr>
<tr>
<td>12:53:00</td>
<td>0.14303</td>
<td>62</td>
<td>1424</td>
<td>293</td>
<td>124</td>
</tr>
<tr>
<td>12:58:00</td>
<td>0.12650</td>
<td>42</td>
<td>1421</td>
<td>319</td>
<td>67</td>
</tr>
<tr>
<td>13:03:00</td>
<td>3.05300</td>
<td>1</td>
<td>1415</td>
<td>174</td>
<td>18</td>
</tr>
<tr>
<td>13:08:00</td>
<td>0.00000</td>
<td>0</td>
<td>1415</td>
<td>100</td>
<td>20</td>
</tr>
</tbody>
</table>
PRL control statement verbs

The first word of every PRL statement is a verb or a condition (IF, FOR, WHERE) with a verb in the expression. This chapter describes PRL verbs in alphabetical order. Each verb is discussed in a separate section that includes a description of the verb, its uses, syntax, and a coding example.

<table>
<thead>
<tr>
<th>PRL verb</th>
<th>Description</th>
<th>See this topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMPUTE</td>
<td>calculates a mathematical expression</td>
<td>“COMPUTE (calculate mathematical expression)” on page 154</td>
</tr>
<tr>
<td>DEFINE</td>
<td>assigns the data characteristics of a variable</td>
<td>“DEFINE (define variable attributes—optional)” on page 155</td>
</tr>
<tr>
<td>DETAIL</td>
<td>creates a tabular summary at each control break</td>
<td>“DETAIL (tabulate and summarize each record)” on page 157</td>
</tr>
<tr>
<td>DIAGNOSE</td>
<td>creates a system snap dump and diagnostic report</td>
<td>“DIAGNOSE (perform diagnostics)” on page 158</td>
</tr>
<tr>
<td>END</td>
<td>terminates the execution of a PRL program</td>
<td>“END (stop execution)” on page 159</td>
</tr>
<tr>
<td>FOR and NEXT</td>
<td>repeats a series of instructions in a loop</td>
<td>“FOR and NEXT (execute a set of statements on a condition)” on page 159</td>
</tr>
<tr>
<td>GOTO</td>
<td>branches to another location of a PRL program identified by a label</td>
<td>“GOTO (continue processing at label with suffix)” on page 161</td>
</tr>
<tr>
<td>IF and THEN</td>
<td>tests conditions and makes a program branch according to the outcome</td>
<td>“IF and THEN (do statement to test conditions)” on page 162</td>
</tr>
<tr>
<td>LET</td>
<td>assigns a length to variable 1 from the length of variable 2</td>
<td>“LET (set variable length)” on page 163</td>
</tr>
<tr>
<td>ORDER BY</td>
<td>sorts data</td>
<td>“ORDER BY (sort selected records)” on page 164</td>
</tr>
<tr>
<td>PRINT</td>
<td>prints a report</td>
<td>“PRINT (print one-line message)” on page 165</td>
</tr>
</tbody>
</table>
### PRL control statement syntax

PRL is a free-form language with minimal restrictions.

It allows approximately 1200 user-defined variables. The number of allowed PRL statements is limited by the available memory. The left most word of the statement is the verb.

#### Conventions

**To enter statements and comments**

- Start each statement with the verb. Each of the PRL verbs is described in this section. PRL verbs must be written in uppercase.

- Write the PRL statements between columns 1 through 72. Characters between columns 73 and 80 are ignored.

- Use one or more blank spaces to separate the verb and parameters in a statement.

<table>
<thead>
<tr>
<th>PRL verb</th>
<th>Description</th>
<th>See this topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>REPORT</td>
<td>creates a report from the data processed by PRL statements</td>
<td>“REPORT (report selected record)” on page 165</td>
</tr>
<tr>
<td>SCAN</td>
<td>reads a variable and performs an editing function on the data</td>
<td>“SCAN (scan a variable field)” on page 166</td>
</tr>
<tr>
<td>SELECT</td>
<td>selects records from CMRDETL, CTGDETL, and CMRSTATS data sets</td>
<td>“SELECT (find record type)” on page 167</td>
</tr>
<tr>
<td>SET</td>
<td>assigns a value to a variable</td>
<td>“SET (assign a value)” on page 168</td>
</tr>
<tr>
<td>SUMMARY</td>
<td>creates a tabular summary of the first preceding variable in the USING statement</td>
<td>“SUMMARY (total and average selected records)” on page 172</td>
</tr>
<tr>
<td>USING</td>
<td>designates column headings and the data shown in a report based upon record fields</td>
<td>“USING (designate column headings and record fields)” on page 173</td>
</tr>
<tr>
<td>WHERE</td>
<td>qualifies record selection based on whether a condition is met</td>
<td>“WHERE (qualify record selection)” on page 175</td>
</tr>
</tbody>
</table>
Use indentation and blank lines to improve readability.

Describe statements with comments. Comments are indicated by an asterisk (*) in column 1. Each comment should be on a separate line preceding the statement it describes.

Use the END verb followed by a semicolon (END;) to indicate the end of a set of PRL verbs that specify an IF-THEN-BEGIN branching routine. Place each verb on a separate line for readability.

**To terminate execution**

Use the END verb to terminate PRL execution.

**To use labels**

Use labels to direct program flow to another statement.

Labels

- are used with GOTO statements
- must be unique within the PRL program
- should be on a separate line
- can start in any column
- can be any length
- can have special characters but no blank characters
- must end with a colon (:) sign

**To specify a literal**

Enclose a literal in single quotation marks to distinguish it from a variable.

**To specify a variable**

- Begin each variable with an alphabetic character; remaining characters for the variable can be alphanumeric. All variables must be eight characters or less. Predefined variables must be written as shown.
- Put an ampersand (&) sign in front of a variable used within a quoted literal. If an & does not precede any characters enclosed in quotation marks, all characters are processed as literals. For example, the following specification sets V3 to ABCDEF:

```cics
SET V1 = 'ABC'
SET V2 = 'DE'
SET V3 = '&V1&V2.F'
```

If the last statement in this example used 'V1V2.F', V3 would be set to V1V2.F. (See “SET (assign a value)” on page 168.)

**To concatenate variables and literals**

Use a period to concatenate a variable to a literal within single quotes, as shown on the preceding page.

**To assign a numeric value to a variable**

Use a decimal point, a sign, or commas (no blank characters) with a number used in an expression or assigned as a value to a variable, as described in “SET (assign a value)” on page 168.

The numeric value is not enclosed in single quotation marks. An example follows:

```cics
SET DAYS = 7
```

**TO specify a substring**

Use `(start-n :length-n)` to select a portion of a character string, as described in “Substrings” on page 137.

Start-n specifies the number of characters from the beginning of the string. Length-n is the length of the substring. The numerics must be delimited with a colon and enclosed in parentheses, such as (4:2), which counts to the fourth character of a string and limits the substring to two characters.

**To continue the USING or PRINT statement**

If necessary, use a minus (-) sign to continue the USING and PRINT statements one line to another line.

---

**PRL reserved words**

The following list shows PRL reserved words. None of these reserved words should be used as variable names in PRL jobs to create batch reports.
Note that `nnnnnnn` are numbers, `Dnnnnnnnn` are numbers reserved for titles, and `Unnnnnnnn` are numbers reserved for details.

<table>
<thead>
<tr>
<th>&amp;</th>
<th>ALARM</th>
<th>ALTERVS</th>
</tr>
</thead>
<tbody>
<tr>
<td>AND</td>
<td>BLINK</td>
<td>CCMD</td>
</tr>
<tr>
<td>CDATE</td>
<td>COLOR</td>
<td>COMPUTE</td>
</tr>
<tr>
<td>CTIME</td>
<td>CURSOR</td>
<td>CYCLE</td>
</tr>
<tr>
<td>Dnnnnnnnn</td>
<td>DEFINE</td>
<td>DELAY</td>
</tr>
<tr>
<td>DETAIL</td>
<td>DIAGNOSE</td>
<td>DUMP</td>
</tr>
<tr>
<td>END</td>
<td>EXEC</td>
<td>EXIT</td>
</tr>
<tr>
<td>FACT</td>
<td>FCMD</td>
<td>FIND</td>
</tr>
<tr>
<td>FOR</td>
<td>GCMD</td>
<td>GET</td>
</tr>
<tr>
<td>GOTO</td>
<td>GRAF</td>
<td>HEADING</td>
</tr>
<tr>
<td>HELP</td>
<td>HIGHLIGHT</td>
<td>HISTORY</td>
</tr>
<tr>
<td>IF</td>
<td>LET</td>
<td>LIN001</td>
</tr>
<tr>
<td>LOGOFF</td>
<td>NEW</td>
<td>NEXT</td>
</tr>
<tr>
<td>OR</td>
<td>ORDER</td>
<td>PAD</td>
</tr>
<tr>
<td>PANELID</td>
<td>PRINT</td>
<td>PROFCNT</td>
</tr>
<tr>
<td>PROFMEM</td>
<td>PROFRC</td>
<td>PUT</td>
</tr>
<tr>
<td>RAND</td>
<td>REPORT</td>
<td>RETURN</td>
</tr>
<tr>
<td>REVERSE</td>
<td>RVIDEO</td>
<td>SCAN</td>
</tr>
<tr>
<td>SCMD</td>
<td>SELECT</td>
<td>SET</td>
</tr>
<tr>
<td>SUMMARY</td>
<td>SYMBOLISE</td>
<td>SYMBOLIZE</td>
</tr>
<tr>
<td>TCOL</td>
<td>TRACE</td>
<td>TRIM</td>
</tr>
<tr>
<td>TROW</td>
<td>TUTOR</td>
<td>Unnnnnnnn</td>
</tr>
<tr>
<td>UNDERSCORE</td>
<td>USING</td>
<td>WHERE</td>
</tr>
<tr>
<td>ZCARD</td>
<td>ZCICS</td>
<td>ZCSA</td>
</tr>
<tr>
<td>ZCT</td>
<td>ZCURSOR</td>
<td>ZCW</td>
</tr>
<tr>
<td>ZDATE</td>
<td>ZERRMSG</td>
<td>ZIFX</td>
</tr>
<tr>
<td>ZIFY</td>
<td>ZLAST</td>
<td>ZLINE</td>
</tr>
<tr>
<td>ZLINES</td>
<td>ZLOC</td>
<td>ZMEMBER</td>
</tr>
<tr>
<td>ZMSG</td>
<td>ZOPTERR</td>
<td>ZPF</td>
</tr>
</tbody>
</table>
COMPUTE (calculate mathematical expression)

COMPUTE calculates a mathematical expression and places the results in a variable.

The expression specified by the COMPUTE statement uses CMRSTATS, CMRDETL, or CTGDTL records as variables.

The SELECT statement declares the CMRSTATS, CMRDETL, or CTGDETL records that are used as COMPUTE variables. The SELECT statement usually precedes the COMPUTE statement in a PRL program.

The results assigned to the variable by COMPUTE can be reported by naming the variable in the USING statement. The COMPUTE statement usually appears before the USING statement in a PRL program.

Syntax

COMPUTE V1 = expression

where

- V1 is a 1- to 8-byte user-specified alphanumeric character string beginning with an alphabetic character.

- expression sets the user-specified variable to the results of the calculation as expressed by the = sign with any of the following operators:

<table>
<thead>
<tr>
<th>Operator</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>+</td>
<td>adds two record variables together</td>
</tr>
<tr>
<td>-</td>
<td>subtracts one record variable from another</td>
</tr>
<tr>
<td>*</td>
<td>multiplies record variables</td>
</tr>
<tr>
<td>/</td>
<td>divides record variables</td>
</tr>
</tbody>
</table>

Whole numbers with signs, commas, or decimal points can be used with any of these operators. They also can be used alone to represent real numbers.
Usage

When the variable being computed is used with DETAIL, SUMMARY, or USING statements, a DEFINE statement should be inserted before the COMPUTE statement.

The computations on printouts can result in the maximum of a 16-digit mathematical expression.

Example

In the following example, COMPUTE uses a simple addition expression and places the results in TOTALPG. The results of the computation are reported for each selected record until the end of file. The USING statement specifies the user-defined TOTALPG variable as a column heading.

```
SET TITLE = 'DAILY PAGING AND TASK ANALYSIS'
SELECT TYPE CC RECORDS FROM CMRSTATS
COMPUTE TOTALPG = T6FSPPGIN + T6FSPPGOU
USING CMRDATE T6FSPPGIN T6FSPPGOU TOTALPG T6FTRNCT
SUMMARY
REPORT
END
```

DEFINE (define variable attributes—optional)

DEFINE fixes the picture of a variable for the duration of the PRL run.

This occurs on the first use; it is ignored for subsequent uses. The use of this verb causes the variable attribute length and decimal significance to be unalterable. In addition, it provides you with the means to specify an edited pattern for display of fields. This is particularly useful if the variable is to appear on the report with USING.

This function also can be particularly useful when used with the SUMMARY or DETAIL functions. DEFINE ensures that the column width accommodates the totals without decimal truncation or alteration.

DEFINE can be used only with alphabetic or numeric fields. It cannot be used with DATE, TIME, or HEX fields.

Syntax

```
DEFINE variable (CHAR nn | EDIT ZZ,ZZ9.99]
```
where

- **variable** is a symbolic variable. The symbolic variable can be predefined or user-defined. User-defined variables must start with an alphabetic character, can be up to eight characters long, and can comprise any alphanumeric character. For a further discussion about variables, see “SET (assign a value)” on page 168.

- CHAR *nn* identifies the verb as a DEFINE for a character variable; *nn* specifies the number of characters to be shown.

- EDIT *ZZ,ZZ9.99* identifies the verb as a DEFINE for a numeric variable; *ZZ,ZZ9.99* are symbols representing the digits in the number. The description can contain up to 25 characters or symbols (but only 15 digits). The following symbols are supported:

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>'.'</td>
<td>decimal placement and representation</td>
</tr>
<tr>
<td>,</td>
<td>numerical editing</td>
</tr>
<tr>
<td>9</td>
<td>decimal significance</td>
</tr>
<tr>
<td>'Z'</td>
<td>decimal significance (zero suppresses digits)</td>
</tr>
<tr>
<td>''</td>
<td>blanks</td>
</tr>
<tr>
<td>-</td>
<td>sign characters</td>
</tr>
</tbody>
</table>

**Note**

The edit pattern should be at least as long as the default title for that field.

**Usage**

Use the DEFINE statement only once per variable. Define the variable before using it in another statement.

A DEFINE statement should be inserted before a COMPUTE or SET statement when the COMPUTE or SET statements are used with DETAIL, SUMMARY, or USING.

**Example**

```
DEFINE T6ERESP EDIT ZZ9.99
SELECT TYPE 6E RECORDS FROM CMRDETL
USING T6ETRID T6ERESP
SUMMARY
REPORT
END
```
DETAIL (tabulate and summarize each record)

DETAIL requests a tabular summary at each control break based on the first variable mentioned in the USING statement that precedes it.

A count is maintained for each item that has a common value matching the first variable. When the value of the first variable changes, a control break occurs and the totals and averages are printed. If neither SUMMARY nor DETAIL is used, a tabular report with no control breaks is printed.

**Syntax**

DETAIL

**Usage**

DETAIL can be coded only after the USING statement but prior to REPORT. If ORDER BY is used, DETAIL must follow this statement.

**Example**

The report shown in Figure 52 on page 157 is produced when the following PRL is executed:

```plaintext
SET TITLE = 'DETAIL TRANSACTION ROSTER'
SELECT TYPE 6E RECORDS FROM CMRDETL
   USING CMRDATE CMRTIME T6ETRID T6ERESP T6ECPUR T6ETMID T6EOPID
   DETAIL
REPORT
END
```

**Figure 52: Detail report**

<table>
<thead>
<tr>
<th>RECORDED DATE</th>
<th>RECORDED TIME</th>
<th>TRANSACTION NAME</th>
<th>TOTAL</th>
<th>RESPONSE TIME</th>
<th>CPU REAL TIME</th>
<th>TERM ID.</th>
<th>OPR ID.</th>
</tr>
</thead>
<tbody>
<tr>
<td>09/17/1998</td>
<td>10:21:12</td>
<td>ETI0</td>
<td>3.511</td>
<td>0.067</td>
<td>0.067</td>
<td></td>
<td></td>
</tr>
<tr>
<td>09/17/1998</td>
<td>10:31:12</td>
<td>ETI0</td>
<td>0.186</td>
<td>0.021</td>
<td>0.021</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>AVG ****</td>
<td>3.697</td>
<td>0.088</td>
<td>0.088</td>
<td></td>
<td></td>
</tr>
<tr>
<td>09/17/1998</td>
<td>10:21:23</td>
<td>ETMO</td>
<td>5.633</td>
<td>0.321</td>
<td>0.321</td>
<td>0.321</td>
<td>R01</td>
</tr>
<tr>
<td>09/17/1998</td>
<td>10:22:31</td>
<td>ETMO</td>
<td>0.906</td>
<td>0.078</td>
<td>0.078</td>
<td>0.078</td>
<td>R01</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AVG ****</td>
<td>6.539</td>
<td>0.399</td>
<td>0.399</td>
<td></td>
<td></td>
</tr>
<tr>
<td>09/17/1998</td>
<td>10:21:33</td>
<td>ETM2</td>
<td>5.504</td>
<td>0.174</td>
<td>0.174</td>
<td>0.174</td>
<td>R01</td>
</tr>
<tr>
<td>09/17/1998</td>
<td>10:22:14</td>
<td>ETM2</td>
<td>1.353</td>
<td>0.155</td>
<td>0.155</td>
<td>0.155</td>
<td>R01</td>
</tr>
<tr>
<td>09/17/1998</td>
<td>10:22:26</td>
<td>ETM2</td>
<td>1.099</td>
<td>0.138</td>
<td>0.138</td>
<td>0.138</td>
<td>R01</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AVG ****</td>
<td>7.956</td>
<td>0.467</td>
<td>0.467</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Chapter 10  PRL control statement verbs  157
DIAGNOSE (perform diagnostics)

DIAGNOSE forces entry into the PRL diagnostic environment with a system snap dump to produce a diagnostic report that helps BMC Software isolate any problems that might occur.

The report shows

- name
- length
- assigned value
- data type (E is numeric; blank indicates characters)
- significant decimal places (3 is .001)

Syntax

DIAGNOSE

Usage

The DIAGNOSE statement should be used only for PRL program execution analysis.

Example

The following example shows the use of DIAGNOSE to analyze a previously executed PRL program. DIAGNOSE is added before the END statement.

```
SET TITLE = 'MAINVIEW for CICS COMMON SYSTEM AREA (CSA) ANALYSIS'
SELECT TYPE 11 RECORDS FROM CMRSTATS
   USING CMRDATE CMRTIME CSATSKCT CSAGTMAN CSAFRMAN -
   CSADUMPS CSASOSCT
ORDER BY CSATSKCT
REPORT
***********************************************************************
* Insert DIAGNOSE statement before END if PRL does not execute    *
* properly.  DIAGNOSE produces system snap dump and list of        *
* variable pool elements showing variable attributes.              *
***********************************************************************
DIAGNOSE
END
```
END (stop execution)

END terminates PRL execution.

Syntax

END

Usage

Use END

■ to indicate PRL report termination

■ in the THEN clause of an IF statement to force termination of PRL execution

For example, to get only the first 50 selected records, use END with IF-THEN as follows:

```
SET Y = Y + 1
IF Y > 51 THEN END
```

(See the example for “IF and THEN (do statement to test conditions)” on page 162.)

■ after DIAGNOSE

Example

```
SET TITLE = 'TRANSACTIONS THAT ABENDED'
SELECT TYPE 6D RECORDS FROM CMRDETL
USING T6ETRID -
    CMRDATE -
    CMRTIME -
    T6EABCD -
    T6EOPID -
    T6ETMID -
    T6ERESP -
    T6ECPUR
REPORT
END
```

FOR and NEXT (execute a set of statements on a condition)

FOR and NEXT perform a series of instructions a given number of times in a loop and cause a variable to be used repeatedly.
Syntax

FOR V1 = V2 TO V3
    statement
NEXT V1

where

- V1 is the counter value. The value should be defined using the SET verb.
- V2 is the initial value of the counter. The length attribute of V2 must be the same as the V3 length attribute.
- V3 is the final value of the counter. If a variable name is used, it must have an ampersand (&) prefix.
- statement can be any PRL statement.

The set of statements following the FOR statement are executed until a NEXT statement is encountered and a specific condition is met. There is at least one execution of the series of statements, even if the final counter value is less than the initial value. After each execution, the V1 value is incremented by 1. Processing loops back to the beginning of the set of statements after the FOR statement until the final value is reached. When the counter value is greater than the final value, the statement following the NEXT statement is executed.

Usage

You can

- use FOR and NEXT anywhere within the PRL
  The FOR-NEXT loop usually follows a SELECT statement.
- nest FOR and NEXT loops, as shown in the following example
  The counter within each nested loop must have a unique variable name. The NEXT statement for the inner FOR-NEXT loop must appear before the NEXT statement for the outer FOR-NEXT loop.
- use any of the variables that are described in “Symbolic variables” on page 138 or “Predefined variables” on page 138

Example

FOR &X = 001 TO 100
    FOR &Y = 001 TO 100
        SET Z = X + Y
    NEXT &Y
NEXT &X
GOTO (continue processing at label with suffix)

GOTO branches to the PRL marked by a label ending with a colon (:) sign.

Execution continues at the first statement following the label.

Syntax

GOTO label

where

- GOTO directs program flow to another set of PRL statements.
- label is the name of a set of PRL statements. The label must have a colon (:) suffix.

The label is alphanumeric, can include special characters but no blank characters, and must end with a colon (:) sign. The label should be on a line by itself and must be unique. It can be any length and can start in any column, as follows:

```plaintext
IF V1 = V2 THEN GOTO LABEL
    statement
    statement
LABEL:
    statement
    END
```

Example

```plaintext
SELECT TYPE 6E AND 6D RECORDS FROM CMRDETL 
    WHERE T6ETMID = '    ' 
    IF TYPE = '6D' THEN GOTO ABEND 
    SET CIC = T6ETRID(1:1) 
    IF CIC = 'C' THEN GOTO CICS
ABEND:
    SET REPORTID = 'TRANAB' 
    SET TITLE = 'TRANSACTIONS THAT ABENDED' 
    SET CLASS = 'R' 
    USING CMRDATE CMRTIME T6ERESP T6ECPUR T6ETRID T6EOPID T6ETMID
    REPORT
CICS:
    SET REPORTID = 'CICSTRAN' 
    SET TITLE = 'CICS-SUPPLIED TRANSACTIONS' 
    SET CLASS = 'R' 
    SET HH = &CMRTIME(1:4) 
    SET TIME = '&HH.0 ' 
    USING TIME T6ETMID T6EOPID T6ECPUR T6ETCWT T6ETRID T6ERESP 
    DETAIL 
    REPORT
    END
```
IF and THEN (do statement to test conditions)

The IF and THEN statements test conditions.

The results of the test determine the action to be taken by the PRL program.

**Syntax**

```
IF expression THEN statement | BEGIN
```

where

- IF-THEN statement is all on one line.
- `expression` consists of variables, whole numbers, or character strings connected by operators (see “Expressions and operators” on page 136).

**Note**
The first parameter is used to define the length in an alphabetic compare.

- THEN is required and can be followed by a PRL statement or a BEGIN clause terminated by an END statement with a semicolon (;) suffix.
- `statement` can be any of the following PRL statements: BEGIN, GOTO, END, SET, or EXEC. An example follows:

```
IF V1 = V2 THEN SET V3 = V4
```

- BEGIN indicates that a series of PRL statements follow the IF-THEN condition. More than one statement can be nested within the IF condition when a true condition results from the test. The end of the statements is signaled by an END statement followed by a semicolon (END;):

```
IF V1 = V2 THEN BEGIN
    statement
    statement
END;
```

The semicolon (;) suffix distinguishes this END statement from an END that terminates PRL execution.

END without a semicolon (;) suffix terminates PRL execution as follows:

```
IF V1 > V2 THEN END
```

The entire report generation sequence can be coded in an IF statement. For example, to produce a report that outlines, among other things, the 20 busiest CPU users in your system, keep a variable called COUNT and increment it just before REPORT.
Follow this with a conditional statement that terminates the job when 20 are reported.

**Example**

```plaintext
SET TITLE = 'HIGHEST 20 USERS OF CPU TIME - &ZDATE'
SELECT TYPE 6E RECORDS FROM CMRDETL
WHERE &CMRDATE = &ZDATE
USING T6ETRID -
   CMRDATE -
   CMRTIME -
   T6EUSER -
   T6ETMID -
   T6ERESP -
   T6ECPUR
ORDER BY T6ECPUR IN DESCENDING SEQUENCE
SET &COUNT = &COUNT + 1
IF COUNT > 20 THEN END
REPORT
END
```

**LET (set variable length)**

LET, in combination with the LENGTH parameter, sets `variable1` in a statement to the length of `variable2`.

As a result, the first variable becomes a 3-digit value between 000 and 256, depending on the length of the second variable.

**Syntax**

```plaintext
LET variable1 = LENGTH variable2
```

where

- `variable1` is the name of a variable or array element to receive a length value. `variable1` can be a character string, a numeric variable, or an array element.

- `variable2` is the variable whose length will be assigned to `variable1`.

**Example**

The following example keeps the length of file and transaction names to 70 characters or less:

```plaintext
LET &LTR = LENGTH &TRNID(&K)
IF &LTR > 70 THEN GOTO LAB3
```
ORDER BY (sort selected records)

ORDER BY sorts variables into a different order from that presented by the SELECT verb.

If the report is to be in chronological order, you do not need to sort your data because all of the records are time stamped with CMRDATE and CMRTIME. Occasionally, you might want to arrange the report by transaction code or by response time. This order is useful if you are interested in summaries based on a group of transactions, a range of execution periods, or other similar information.

The ORDER BY statement relies upon the IBM Sort Facility to pass user data to a sort exit as the second parameter in the list. Other sort facilities can be used if they are compatible with the IBM Sort Facility (use standard sort exits E15 and E35).

The ORDER BY statement can be overridden with your own sort control statement, such as

```
//SORTCNTL DD *
  options - -
/*
```

**Syntax**

```
ORDER BY v1 [IN DESCENDING SEQUENCE]
```

**Usage**

- Only one ORDER BY statement is allowed per report set.
- Only one variable can be specified with ORDER BY. It can be either a user-defined or predefined variable.
- If DETAIL or SUMMARY is used, the ORDER BY variable must be the same as the first variable in the USING statement, as shown below.
- **IN DESCENDING SEQUENCE** orders the data from high to low values.

**Example**

```
SELECT TYPE 6E RECORDS FROM CMRDETL
USING T6ETRID -
  T6ERESP -
  T6ECPUR -
  T6EFCWT -
  T6ESHWM -
  T6EPAGCT
ORDER BY T6ETRID
SUMMARY
```
PRINT (print one-line message)

PRINT causes a one-line message, the value of a variable, or a combination of these to be printed in a report whenever the PRINT statement is processed.

Syntax

\[
\text{PRINT} \ [\text{V1}|'\text{string'}|\text{V1..Vn }'\text{string'}|\text{fieldName}]\]

where

- PRINT (with no parameters) prints a blank line.
- V1 is a variable that is assigned a value by the SET statement. The value of the variable is printed.
- 'string' is a literal character string enclosed in single quotation marks. The string can contain alphanumeric, blank, or special characters.
- fieldName is the name of a field in a CMRDETL, CTGDETL, or CMRSTATS record. See MainView for CICS PERFORMANCE REPORTER Data Reference. The field value is printed.

Example

```
SET DOWN = '         * * * MAINVIEW for CICS NOT ACTIVE * * *'
SET TITLE = 'MAINVIEW for CICS GLOBAL PERFORMANCE ANALYSIS'
SELECT TYPE CC RECORDS FROM CMRSTATX
USING CMRDATE CMRTIME T6FSUCP T6FSTCCP T6FSJCCP T6FSKCCP T6FSOSCP
   IF T6FSFLAG = 'L' THEN BEGIN
      PRINT
      PRINT DOWN
      PRINT
      END;
```

REPORT (report selected record)

REPORT produces a report that shows the results of each record processed by the PRL statements between SELECT and REPORT. Each field specified by the USING statement is reported until end of file.
The default report size is 50 lines x 132 print positions. The report size can be changed by SET LINES = and SET WIDTH = as shown by “SET (assign a value)” on page 168. The width of the body of the report is determined by the record fields and variables specified in the USING statement (see “USING (designate column headings and record fields)” on page 173.)

The statement following REPORT receives control when all selection criteria presented by the SELECT verb have been met and the file has reached its end.

**Syntax**

REPORT

**Example**

```plaintext
SELECT TYPE CC RECORDS FROM CMRSTATS
  USING CMRDATE T6FSPGIN T6FSPGOU
  SUMMARY
REPORT
END
```

**SCAN (scan a variable field)**

SCAN scans a variable and edits it.

**Syntax**

```plaintext
SCAN &V n [REMOVING|REPLACING]
  [LEADING|ALL]
  [BLANK|NULL|c]
  [WITH|[BLANK|NULL|c]]
```

where

- &V n is the name of a defined variable that contains the target data to be edited.
- REMOVING | REPLACING indicates the editing to be performed on the data of the target variable.
- LEADING | ALL edits the leading or trailing data to one byte or all the data as indicated.
- WITH must be used if REPLACING is specified.
- BLANK specifies blank characters.
- NULL specifies null characters.
- \( c \) represents any alphanumeric characters.

**Example**

```
SET BB = '\&vbarSELECT TYPE CC RECORDS FROM CMRSTATS
WHERE CMRDATE = '09/17/1998'
SET Y = T6FTRNCT / 5
SET Y = Y + 1
IF Y > THEN 51 THEN SET Y = 52
SCAN &Y REMOVING LEADING 0
SET &X = BB(1:&Y)
USING CMRDATE CMRTIME T6FTRNCT X
REPORT
END
```

**SELECT (find record type)**

SELECT chooses records from CMRDETL, CTGDETL, or CMRSTATS by the record type. Optionally, the starting point of the SELECT can be qualified with a date and time.

**Syntax**

```
SELECT TYPE xx RECORDS FROM dataSet
SELECT TYPE xx xx RECORDS FROM dataSet
SELECT TYPE xx AND xx RECORDS FROM dataSet
SELECT TYPE xx xx xx ... xx RECORDS FROM dataSet
SELECT ALL RECORDS FROM dataSet
SELECT TYPE xx RECORDS FROM dataSet STARTING AT time ON date
```

where

- TYPE xx or ALL names the specific records to be used from the MainView for CICS CMRSTATS, CMRDETL, or CTGDETL data sets; xx is the hexadecimal identifier of the record as described in “Collecting report data” on page 19 and shown in the MainView for CICS PERFORMANCE REPORTER Data Reference. As shown in the syntax example, more than one record type can be selected from a data set with a single SELECT statement. A maximum of nine record types can be specified on one line.

- ALL selects all record types from a single data set. The specified types of records must reside in the same data set. CMRDETL data sets contain Type 6E or Type 6D records. CTGDETL data sets contain Type 6F records. For the record types found in CMRSTATS data sets, see Chapter 3 of the MainView for CICS PERFORMANCE REPORTER Data Reference. A maximum of nine specific record types can be selected.

- RECORDS FROM is a required clause.
**dataSet** can be either CMRSTATS, CMRDETL, or CTGDETL on tape or disk, as defined by the JCL. PERFORMANCE REPORTER searches the data set for a record ID that matches the value specified by SELECT TYPE xx and makes those records available for PRL processing.

**STARTING AT** time ON date specifies when reporting is to begin. MainView for CICS performance data is arranged in chronological order with date and time stamps. The date is always represented as mm/dd/yyyy and the time as hh:mm:ss. To concentrate on a particular time period, add this parameter on the end of the SELECT statement. The selection process will not start until the first record is found. Today’s date is given by using the variable ZDATE. The current time is given by the variable ZTIME.

For example, the following statement reports CICS activity since the start of business today:

```
SELECT TYPE CC RECORDS FROM CMRSTATS STARTING AT 08:00:00 ON &ZDATE
```

This job can be run on any day and the start date does not need to be changed.

**Note**

If records are to be selected from an archive tape, STARTING AT has no effect and should not be used. Records can be selected from an archive tape by using WHERE as follows:

```
SELECT TYPE CC RECORDS FROM CMRSTATS
WHERE CMRDATE = '09/17/1998'
```

**Usage**

Only one SELECT control statement per CMRDETL, CTGDETL, or CMRSTATS per run can be specified in each PRL program. You could receive unpredictable results if you do use more than one SELECT control statement in your program.

**Example**

```
SET TITLE = 'D E T A I L    T R A N S A C T I O N    R O S T E R'
SELECT TYPE 6E RECORDS FROM CMRDETL
   USING CMRDATE CMRTIME T6ERESP T6EFCWTC T6ETRID T6ENETNM
REPORT
END
```

**SET (assign a value)**

SET, in combination with the = operator, assigns a value (characters or numerics) to a variable.
It can be used to specify report headings and the contents of variables and to control the printing of the report.

**Syntax**

```
SET variable = expression
```

where

- `variable` is a symbolic variable. The symbolic variable can be predefined or user-defined. A user-defined variable must start with an alphabetic character, can be up to eight characters long, and can comprise any alphanumeric character. It can be preceded by an ampersand (`&`) to distinguish it from a literal.

- `expression` can be variables, whole numbers, or character strings alone or in combination with operators (see “Expressions and operators” on page 136).

A value that was assigned to a variable before a SELECT statement does not change.

**Usage**

When the variable being set is used with DETAIL, SUMMARY, or USING statements, a DEFINE statement should be inserted before the SET statement.

A symbolic variable can be set to

- the value of another variable, which can be symbolic or predefined (see “Predefined variables” on page 138)

  ```
  SET WEEK = &DAYS
  ```

  **Note**
  
  Ampersand (`&`) is a reserved word and should not be used as a variable value. In other words, `&a` is a variable while `&` or `&&` is not.

- a character string
  
  Enclose characters in single quotation marks. Begin a variable inside quotation marks with an ampersand. Use a period to delimit a variable from a literal. The
characters can be a direct substitute for the variable, a substring, or concatenated with other variables, as shown in the following examples:

**Example 1: Substitution**

In this example, ABCDEF is substituted for the symbolic variable, ALPHA:

```plaintext
SET ALPHA = 'ABCDEF'
```

**Example 2: Substring**

In this example, B is assigned to the symbolic variable, BETA. The substring expression specifies that the second character of the string be substituted for &ALPHA, for a length of one character.

```plaintext
SET BETA = &ALPHA(2:1)
```

**Example 3: Concatenation**

Character strings can be concatenated with other variables and literals. Identify variables with an ampersand prefix and add a period at the end of the variable to delimit it from a literal, as follows:

```plaintext
SET ALPHA = 'A'
SET DELTA = 'DEF'
SET AD = '&ALPHA.BC&DELTA'
```

The AD variable is assigned a value of ABCDEF.

If an ampersand prefix or a period suffix is not used to delimit the variables from the literals, the characters are assigned to the AD variable. An example follows:

```plaintext
SET AD = 'ALPHABCDELTA'
```

In the preceding example, the AD variable is assigned a value of ALPHABCDELTA.

- a numeric integer to be used as a counter

For decimal arithmetic, use the COMPUTE verb.

```plaintext
SET DAYS = 7
```

A number can be an integer from 0 to 9. The number can have one decimal point, a sign, or commas, but it cannot contain any blank characters or be enclosed in single quotation marks.

A numeric value also can be assigned to a variable as a counter. The number of digits is initially set to 00000 as a default. When 99999 is reached, the value resets to 00000. The default digits can be changed with the SET statement. For example, SET number = 01 establishes a two-digit counter that, at 99, resets to 00. An example follows:

```plaintext
SET NUMBER = 01
SET V1 = V2 + NUMBER
```

**Note**

SET does not keep track of decimals or signs.

Report output can be set as follows (all specified values must be enclosed in single quotation marks):
- **CASE** specifies that the output characters be in uppercase or mixed case. The default is uppercase. SET CASE must precede other PRL statements and is written as:

  ```
  SET CASE = 'MIXED'
  SET CASE = 'UPPER'
  ```

- **CLASS** specifies the output class for the report. It is a 1-byte, site-defined character. A is the default.

  **Note**

  Report output also can be allocated with a DD statement, as described in “Routing report output” on page 179, that has the same name as the value of SET REPORTID.

- **FORM** specifies four characters defining the type of paper on which the report is to be printed. STD is the default.

  **Note**

  Instead of specifying SET FORM, a DD allocation that has the same name as the value of SET REPORTID can specify the form type.

- **LINES** specifies the report page length. The default is 50 lines. The minimum value is 001 and the maximum value is 255.

- **REPORTID** specifies a 1- to 8-character alphanumeric ID for the report. The ID is in the upper left corner of the report. The execution job name is the default.

  If a DD output allocation has the same name as the user-specified value for REPORTID, the report is allocated to the DD output, as described in “Routing report output” on page 179.

- **TITLE** specifies the first of four lines to be generated by PRL on every page. Specify any alphanumeric string up to a maximum of 80 characters. The default is MainView for CICS. The title is centered automatically at the top of each page.

- **TITLE1** and **TITLE2** specify a subtitle for the second line (TITLE1) or third line (TITLE2) of every page. The subtitle length is 80 characters. Each specified subtitle is centered automatically at the top of each page. Blank is the default.

- **WIDTH** specifies the width of the report title. The default is 132 print positions. From 32 to 255 print positions can be specified for your report. Report titles are adjusted to the specified width. The USING statement determines the column data to be reported. The reported data is truncated if it does not fit within the specified width. If USING specifies only a few variables and the WIDTH is at a large value, the column data is reported as formatted by the PNLLIB data dictionary. It is not adjusted to the specified WIDTH value.

- **ZHEADING** specifies the print characteristics of the column headings of PRL tabular reports. If ZHEADING is not specified, the default prints each column...
heading three times without a carriage return. The default should be used with line printers to create bold text in the column headings by printing each word three times.

Use ZHEADING = 'SINGLE' to create viewable online reports. Use ZHEADING = 'NONE' to create PRL reports without column headings.

Example

```plaintext
SET CLASS = 'R'
SET FORM = 'C012'
SET WIDTH = 70
SET LINES = 58
SET REPORTID = 'CICSM16'
SET TITLE = 'MAINVIEW for CICS - Monthly System Status'
SET TITLE1 = '        User Logon Times           '

SELECT TYPE 6E RECORDS FROM CMRDETL
   WHERE T6ETMID NOT EQUAL '    '
   SET COUNT = COUNT + 1
   USING CMRDATE CMRTIME T6ETRID T6ERESP T6ECPUR T6ETMID T6EOPID
REPORT
PRINT COUNT
END
```

**SUMMARY (total and average selected records)**

SUMMARY requests a tabular summary based on the first variable mentioned in the USING statement that precedes it.

A count is maintained for each item that has a common value matching the first variable. Only the totals appear on the report, followed by an average, for each and every numeric variable on the print line.

Even after all selection criteria have been met, each record generates a separate report line; the report could become quite large if a large volume of data is being processed.

If you provide summaries, your nominated summary fields will gather and condense all detail records into two lines of print for each control break.

If neither SUMMARY nor DETAIL is used, a tabular report is printed.

**Syntax**

SUMMARY
**Usage**

SUMMARY can be coded only after the USING statement but prior to REPORT. If ORDER BY is used, SUMMARY must follow this statement.

**Note**

Character data can only be included in a summary report if it is specified as the first USING variable.

**Example**

The following example produces the report shown in Figure 53 on page 173:

```plaintext
SET CLASS = 'R'
SET REPORTID = 'RES2RPT2'
SET TITLE = 'SUMMARY TRANSACTION ROSTER'
SELECT TYPE 6E RECORDS FROM CMRDETL
   USING CMRDATE T6ERESP T6ECPUR T6EDIST T6EFCWT T6ESUST T6ETSWT
   SUMMARY
REPORT
END
```

**Figure 53: Summary report**

<table>
<thead>
<tr>
<th>RECORDED</th>
<th>TOTAL</th>
<th>RESPONSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATE</td>
<td>TIME</td>
<td>TIME</td>
</tr>
<tr>
<td>1998/09/17</td>
<td>2</td>
<td>4.808</td>
</tr>
<tr>
<td>AVG *****</td>
<td></td>
<td>0.248</td>
</tr>
<tr>
<td>1998/09/17</td>
<td>2</td>
<td>1.495</td>
</tr>
<tr>
<td>AVG *****</td>
<td></td>
<td>0.297</td>
</tr>
</tbody>
</table>

**USING (designate column headings and record fields)**

USING designates column headings and record contents for each named record field specified with USING.

Only one USING statement is required. The USING statement selects the column heading from an internal data dictionary. To specify your own column heading that is not in the dictionary, set a variable to a literal (maximum of eight alphanumeric characters). Then specify the variable name in the USING statement, as described in “User-defined data” on page 182. Column headings are printed on lines 5 and 6, with the corresponding underlines on line 7 of the report.
Column headings and data appear on the report line in the order specified by the USING statement, reading from left to right. If an undefined field name or variable is specified, it does not appear on the report. If the column headings do not fit within the page width, the headings are truncated.

**Syntax**

```
USING Vn. . . Vn
```

where \textit{Vn} represents the record field names shown in the \textit{MainView for CICS PERFORMANCE REPORTER Data Reference} or user variables defined by SET.

**Note**

Depending on the record field widths, use of more than eight names can cause truncation. More than 11 field names generally cause both truncation and unpredictable results.

**Usage**

To continue a list of variables on the next line, use a hyphen (-). A hyphen can be used only to continue variables to another line, not to continue statements.

**Note**

The variables listed on the USING statement have implied DEFINE statements associated with them that are used unless a previous DEFINE statement for the variable has been specified.

USING fixes the meaning of a variable for the duration of the PRL run. This occurs on the first use; subsequent uses are ignored. The effect of the USING statement is to make the variable attribute length and the decimal significance unalterable.

If the variables listed in a USING statement require special editing characteristics, consider placing a DEFINE statement before the USING statement. For further information on DEFINE, see “DEFINE (define variable attributes—optional)” on page 155.

The maximum number of variables in a USING statement is limited to the number of variables that can fit on a print line. The default print line is 132 characters long.

To change the width of the print line, use

```
SET WIDTH=xxx
```

You can increase the size of the print line to up to 255 print positions.
WHERE (qualify record selection)

WHERE qualifies record selections.

If the condition is matched, control flows to the next statement.

Syntax

WHERE \textit{Vn} = \textit{value}

where

- \textit{Vn} is the field name of a CMRSTATS, CMRDETL, or CTGDETL record. See the MainView for CICS PERFORMANCE REPORTER Data Reference.
- \textit{value} can be a literal character string or a defined variable.

WHERE is written after the SELECT statement and is used with any of the following operators to qualify a selected record:

<table>
<thead>
<tr>
<th>Operator</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>=</td>
<td>equal to</td>
</tr>
<tr>
<td>&gt;</td>
<td>greater than</td>
</tr>
<tr>
<td>&lt;</td>
<td>less than</td>
</tr>
<tr>
<td>\textbackslash{} =</td>
<td>not equal to</td>
</tr>
<tr>
<td>\textbackslash{} &lt;</td>
<td>greater than or equal to</td>
</tr>
<tr>
<td>\textbackslash{} &gt;</td>
<td>less than or equal to</td>
</tr>
</tbody>
</table>
### Operator Description

<table>
<thead>
<tr>
<th>Operator</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AND</td>
<td>NA</td>
</tr>
<tr>
<td>OR</td>
<td>NA</td>
</tr>
</tbody>
</table>

### Usage

The argument in the WHERE clause can be repeated with other arguments, appropriately using the operators AND and OR. Only one variable can be qualified per line. The following example restricts the selection to activity by user DJD, at terminal SC32, with a response time that is greater than two seconds:

```sql
SELECT TYPE 6E RECORDS FROM CMRDETL
WHERE T6ETMID = 'SC32'
AND T6EOPID = 'DJD'
AND T6ERESP  > 2
```

**Note**

AND must be used if NOT (¬) is specified with another qualifying condition for WHERE. An example follows:

```sql
SELECT TYPE 6E RECORDS FROM CMRDETL WHERE CMRDATE = '09/27/1998' AND T6ETMID ¬= ' '
```

Alternative data can be selected for the same variable by adding an OR operator to the right of the WHERE expression. An example follows:

```sql
SELECT TYPE 6E RECORDS FROM CMRDETL
WHERE T6ETRID = 'CSSN' OR 'CSSF' OR 'CSTT'
```

This example selects 6E records created by transactions CSSN, CSSF, or CSTT.

This same WHERE condition could also be repeated with OR as follows:

```sql
WHERE T6ETRID = 'CSSN'
OR T6ETRID = 'CSSF'
OR T6ETRID = 'CSTT'
```

An AND operator with WHERE combines specific data from selected records. The following example selects 6E records that had activity on 09/27/1998 between 10:00 A.M. and 11:00 A.M.:

```sql
SET DATE = '09/27/1998'
SET FTIME = '09:59:59'
SET LTIME = '11:00:01'
SELECT TYPE 6E RECORDS FROM CMRDETL
WHERE CMRDATE = DATE
AND CMRTIME > FTIME
AND CMRTIME < LTIME
```

These PRL statements select data from 6E records based upon the date and time specified in the CMRDATE and CMRTIME fields that
match the 09/27/1998 date assigned to the DATE variable

• have a time stamp (recorded in the CMRTIME common header field) greater than the time assigned to the FTIME variable (start at 10:00)

• have a time stamp less than the time assigned to the LTIME variable (end at 11:00)

Any number of AND or OR conditions can be used with a WHERE statement. However, a condition must be defined with either all ANDs or all ORs because combining them may create a logically ambiguous condition.

If ≠ is used with more than one AND or OR clause, the not condition inverts the logic and should be avoided as follows:

• Use an all positive condition.

• Relegate a single ≠ to the last clause.

The following example shows the use of OR statements to select alternative data from the same field (T6ETRID - transaction identifier). An AND statement adds another required condition to the WHERE condition that further restricts data collected from the field. The expression ≠ is used last and qualifies the record selection to identified terminals.

```
SELECT TYPE 6E RECORDS FROM CMRDETL
  WHERE T6ETRID = 'CSSN' OR 'CSSF' OR 'CSTT'
  AND T6ETMID ≠ '    '
```

**Note**

Use AND last for NOT (¬) conditions.

**Example**

```
SET WIDTH = '80'
SET TITLE = 'TERMINAL ERRORS'

SELECT TYPE 6E RECORDS FROM CMRDETL
  WHERE T6EPGNM = 'DFHACP'
  USING CMRDATE CMRTIME T6ETRID T6ETMID T6EOPID T6ECPUR T6ERESP
REPORT
END
```
WHERE (qualify record selection)
This chapter describes additional programming techniques to produce specialized PRL reports. Using these techniques, you can create programs that do the following functions:

- route report output
- format report output
- select archive data
- expedite record retrieval
- index detail (CMRDETL) records
- index by qualifying variable names
- produce graphs with PRL reports
- summarize data
- sort by more than one field
- report fourth-generation language (4GL) database activity
- create worst transaction response time reports

Routing report output

The output class and format of a user-defined report can be defined with a DD statement in the JCL and the PRL SET REPORTID statement.

The output class and format also can be defined with the PRL statements SET CLASS and SET FORM, described in “SET (assign a value)” on page 168.
If neither the JCL nor the PRL report allocation is specified, PERFORMANCE REPORTER allocates the report to a default output class of A and a default form of STD. The user-written PRL program (DD statement named ISYPROG) and the JCL output listing (MSGCLASS parameter of JOB statement) are allocated as specified in the MVS JCL, described in “Generating PRL reports” on page 139.

**Defining output with JCL and PRL SET REPORTID**

To define report output with JCL, the ddname of the DD statement must be the same as the value of the SET REPORTID statement or its default (the execution job name).

For example, if the PRL program specified SET REPORTID = 'TRANAB', the name of the DD statement defining the report output must be

```plaintext
//TRANAB DD userParametersForReportOutput
```

**Note**
The user parameters must follow the rules and conventions for coding a DD statement, as described in the IBM publication, *MVS Extended Architecture JCL Reference Manual*.

If more than one report is created and DD statements are used to define each report output, there must be a corresponding report ID for each named DD allocation. An example follows:

```plaintext
//TRANAB DD ......  
//TRANSUM DD ......  
//TRANDET DD ......  
```

The preceding specification allocates the report as specified in the JCL if

- a user-written PRL program that contains a `SET REPORTID = 'TRANAB'` declaration for the TRANAB allocation exists

- a user-written PRL program that contains a `SET REPORTID = 'TRANSUM'` declaration for the TRANSUM allocation exists

- a user-written PRL program that contains a `SET REPORTID = 'TRANDET'` declaration for the TRANDET allocation exists

If a DD allocation for the report in the JCL exists and SET REPORTID is not specified or the value is not the same as the ddname of the report output DD statement, the JCL DD allocation is ignored. When no SET REPORTID declaration exists in a PRL program, the execution job name is the default value.
Defining output with PRL SET CLASS and SET FORM

You can define report output with JCL as described in “Defining output with JCL and PRL SET REPORTID” on page 180 or you can use the PRL SET CLASS and SET FORM statements.

However, if SET CLASS and SET FORM allocate the report output and the JCL contains a named DD statement that matches a SET REPORTID value, the report is allocated as specified by the DD statement. The output defaults to a CLASS of A and a form of STD when there is no user-specified allocation.

Formatting report output

A PRL report is formatted according to the parameters specified with the SET and USING statements.

SET can be used to size and title the report and to define a variable to be reported. USING specifies the CMRDETL, CTGDETL, and CMRSTATS records to be reported as reformatted by the BBPLIB data dictionary. See the MainView for CICS PERFORMANCE REPORTER Data Reference. The data dictionary descriptions for each record field are printed as column headings with each specified field name.

Predefined data dictionary descriptions

PERFORMANCE REPORTER provides a data dictionary in a partitioned data set called BBPLIB that automatically creates column headings for the CMRDETL, CTGDETL, and CMRSTATS record fields specified with the PRL.

A description of each CMRDETL, CTGDETL, and CMRSTATS record field is stored in the data dictionary. These descriptions are used in the report as column headings.

The USING statement (discussed in “USING (designate column headings and record fields)” on page 173) and the field names of a CMRDETL, CTGDETL, or CMRSTATS record (shown in the MainView for CICS PERFORMANCE REPORTER Data Reference) specify the data dictionary headings to be printed.
For example, as shown in the “Time versus activity graph” on page 187, the following specification selects the CMRSTATS type CC record and specifies the data dictionary descriptions for the CMRDATE, CMRTIME, and T6FTRNCT fields:

```
SELECT TYPE CC RECORDS FROM CMRSTATS
USING CMRDATE CMRTIME T6FTRNCT
REPORT
END
```

The CMRDATE, CMRTIME, and T6FTRNCT fields correspond to the following column headings:
- RECORDED DATE
- RECORDED TIME
- TOTAL TASKS

**User-defined data**

To define data to be reported, assign a value to a variable, and then specify the variable with the USING statement.
Example

For example, the following specification prints the variable name, TRANTERM, as a column heading, and its assigned value as the column contents:

```sql
SELECT TYPE 6E RECORDS FROM CMRDETL

SET TRANTERM = '&T6ETRID.  &T6ETMID'

USING TRANTERM

REPORT

END
```

The transaction identifier (`&T6ETRID` field of the CMRDETL record) and the terminal identifier (`&T6ETMID` field of the CMRDETL record) are reported in the TRANTERM column.

The variable name must start with an alphabetic character and can comprise any alphanumeric string up to a maximum of eight characters. If the name is longer than the width of the column data, the name is truncated to the column width.

PERFORMANCE REPORTER right-justifies user-defined headings and adds two blank characters at the beginning of the column heading. The column headings can be aligned with the column data either by examining the BBPLIB data dictionary or running the report. The printed report shows where to align the headings. The character string assigned to a column heading variable can then be adjusted to the width of the column data. Or the BBPLIB data dictionary can be examined for the length and type of each CMRDETL, CTGDETL, or CMRSTATS field, which together determine the column width. For a description of the type key definition, see the MainView for CICS PERFORMANCE REPORTER Data Reference.

Selecting archive data

As described in “Archiving data (CMRPURG)” on page 25, CMRDETL, CTGDETL, and CMRSTATS records can be archived to tape. The SELECT statement does not have to be changed to distinguish a tape data set from VSAM; the PRL does this automatically. Therefore, change the CMRDETL, CTGDETL, or CMRSTATS DD statement to point to a tape data set, as shown in Figure 54 on page 183.

**Figure 54: DD statement change to point to a tape data set**

```plaintext
//jobname JOB user parameters                     <-- CHANGE AS NEEDED
/*JOBPARM user parameters                         <-- CHANGE AS NEEDED
//CMRL PROC
//PRL       EXEC PGM=CMRPRLN
//STEPLIB  DD  DISP=SHR,DSN=CMR.CMRV3.BBLINK      <-- CHANGE AS NEEDED
//SYSPRINT DD  SYSOUT=*                           <-- CHANGE AS NEEDED
//SYSUDUMP DD  SYSOUT=*                           <-- CHANGE AS NEEDED
//ISYPROG  DD  SYSOUT=*,DCB=(LRECL=80,BLKSIZE=160,RECFM=FBA)
```
Expediting record retrieval

The data collected in CMRDETL, CTGDETL, and CMRSTATS is arranged in chronological order with date and time stamps.

A record can be selected and the selection process can start at a specific time or date by use of STARTING AT with the SELECT statement; for example, the following specification begins selection of only CC records collected on 09/30/2004 at 8:00 A.M.:

```
SELECT TYPE CC RECORDS FROM CMRSTATS STARTING AT 08:00:00 ON 09/30/2004
```

**Note**

If records are being selected from an archive tape, STARTING AT has no effect and should not be used. Records can be selected from an archive tape by using WHERE. An example follows:

```
SELECT TYPE CC RECORDS FROM CMRSTATS
WHERE CMRDATE = '09/30/2004'
```

The archive data set name must be defined in the JCL as shown in “Archiving data (CMRPURG)” on page 25.
Indexing detail CMRDETL records

CMRDETL record file data can be indexed to obtain information from the T6E and T6D records.

To report the data in these fields, use a FOR-NEXT statement to set up a counter. Within the FOR-NEXT loop, index the fields by the variable counter, assign the value of the indexed fields to variables, and report each variable using the variable names and column headings.

**Note**
A CMRDETL data set may contain two forms of T6E records, original and expanded. Be sure to check the expanded record flag (field T6EQUAL) to determine whether a given record is expanded or not. The sample program CMRPRL16, which is distributed in the BBSAMP data set, illustrates how to process both original and expanded format records in the same file.

Table 22 on page 185 lists the T6E and T6D fields that can be used for indexing.

<table>
<thead>
<tr>
<th>Field name</th>
<th>Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>T6EQUAL</td>
<td>format of the file data, as a decimal value of</td>
</tr>
<tr>
<td></td>
<td>■ 127 or less for an original format record</td>
</tr>
<tr>
<td></td>
<td>■ 128 or greater for an expanded format record</td>
</tr>
<tr>
<td>T6EFCNT</td>
<td>number of files per task</td>
</tr>
<tr>
<td></td>
<td>The maximum value is 40 for expanded format records and 255 for original format records.</td>
</tr>
<tr>
<td>Field name</td>
<td>Use</td>
</tr>
<tr>
<td>------------</td>
<td>-----</td>
</tr>
<tr>
<td><strong>T6EFI</strong></td>
<td>File type, identified by</td>
</tr>
<tr>
<td></td>
<td>■ A (Adabas)</td>
</tr>
<tr>
<td></td>
<td>■ C (CICS)</td>
</tr>
<tr>
<td></td>
<td>■ D (DATA.COM/DB)</td>
</tr>
<tr>
<td></td>
<td>■ K (System 2000)</td>
</tr>
<tr>
<td></td>
<td>■ P (MainView for CICS)</td>
</tr>
<tr>
<td></td>
<td>■ S (SUPRA)</td>
</tr>
<tr>
<td></td>
<td>■ U (user value)</td>
</tr>
<tr>
<td></td>
<td>■ 1 (DL/I)</td>
</tr>
<tr>
<td></td>
<td>■ 2 (DB/2)</td>
</tr>
<tr>
<td><strong>T6EFN</strong></td>
<td>file or database name</td>
</tr>
<tr>
<td><strong>T6EFT</strong></td>
<td>elapsed I/O request time</td>
</tr>
<tr>
<td><strong>T6EFC</strong></td>
<td>number of I/O requests to the file</td>
</tr>
<tr>
<td><strong>T6ERD</strong></td>
<td>number of file read or DB2 select requests</td>
</tr>
<tr>
<td><strong>T6ERU</strong></td>
<td>number of file read for update or DB2 open requests</td>
</tr>
<tr>
<td><strong>T6EWR</strong></td>
<td>number of file write or DB2 close requests</td>
</tr>
<tr>
<td><strong>T6ERW</strong></td>
<td>number of file rewrite or DB2 insert requests</td>
</tr>
<tr>
<td><strong>T6EDL</strong></td>
<td>number of file delete or DB2 delete requests</td>
</tr>
<tr>
<td><strong>T6EUL</strong></td>
<td>number of file unlock or DB2 update requests</td>
</tr>
<tr>
<td><strong>T6ESB</strong></td>
<td>number of file start browse or DB2 fetch requests</td>
</tr>
<tr>
<td><strong>T6ERN</strong></td>
<td>number of file read next or DB2 commit requests</td>
</tr>
<tr>
<td><strong>T6ERP</strong></td>
<td>number of file read previous or DB2 other requests</td>
</tr>
<tr>
<td><strong>T6EEB</strong></td>
<td>number of file end browse requests</td>
</tr>
<tr>
<td><strong>T6ERB</strong></td>
<td>number of file reset browse requests</td>
</tr>
<tr>
<td><strong>T6EOT</strong></td>
<td>number of file other requests</td>
</tr>
<tr>
<td><strong>T6EV1</strong></td>
<td>first volume serial number associated with the file</td>
</tr>
<tr>
<td><strong>T6EV2</strong></td>
<td>second volume serial number associated with the file</td>
</tr>
<tr>
<td><strong>T6EV3</strong></td>
<td>third volume serial number associated with the file</td>
</tr>
</tbody>
</table>
Indexing by qualified variable name

The parenthetical value used to qualify a variable is appended to the variable name.

This qualified name must be used consistently by the TRANS variable within the PRL program, as shown in Figure 55 on page 187.

**Figure 55: Qualified variable names PRL statement example**

```
SET TITLE = ' -*- TOTAL USER TRANSACTION SUMMARY -*-'
SELECT TYPE 6E RECORDS FROM CMRDETL
SET PGMMNAM = T6EPGMNM(1:4)
SET TRAN = T6ETRID(1:1)
SET HOUR = CMRTIME(1:2)
WHERE PGMMNAM = 'FACT'
    AND TRAN = 'C'
    AND CMRDATE = '09/04/1999'
COMPUTE TRANS(&HOUR) = TRANS(&HOUR) + 1
USING T6ETRID T6ERESP T6EDIST T6EDISTC T6EFCWT T6EFCWTC T6ESUST
ORDER BY T6ETRID
SUMMARY
REPORT
PRINT 'TOTAL TRANSACTIONS PER HOUR'
FOR &X = 00 TO 24
    SET &Y = &X + 01
    PRINT 'FROM ' &X ': 00 TO ' &Y ': 00     TOTAL: ' &TRANS(&X)
NEXT &X
END
```

Produce graphs with PRL reports

The following case studies show how to create PRL graph reports with CMRDETL or CMRSTATS data.

The first case study can use any record stored on either data set. The PRL program described in the second study can use CA records only.

**Time versus activity graph**

This topic provides a table.

<table>
<thead>
<tr>
<th>Member</th>
<th>CMRPRL12</th>
</tr>
</thead>
<tbody>
<tr>
<td>Objective</td>
<td>to show graphically the activity over time within a given CICS</td>
</tr>
<tr>
<td>Data</td>
<td>Global Performance Records (Type CC) from the CMRSTATS file</td>
</tr>
</tbody>
</table>
Every 5 minutes, MainView for CICS produces a Global Performance Record containing the task count for the 5-minute interval (T6FTRNCT) along with other fields if T6F=YES is set in CMRSOPT. PRL records these values in chronological order and represents them on a bar chart, as seen in Figure 57 on page 189.

The control statements shown in the following figure specify the input data and the format of the report:

**Figure 56: Time versus activity PRL statements**

```
SET CLASS = 'R'
SET REPORTID = 'PRL12'
SET TITLE = 'T I M E V S A C T I V I T Y'
SET WIDTH = '82'
SET FORM = 'STD'
DEFINE GRAPH CHAR 52
DEFINE T6FTRNCT EDIT ZZZZZ9
DEFINE NUMSTAR EDIT 99999999999999
SET BB = '&vbar SELECT TYPE CC RECORDS FROM CMRSTATS
         WHERE CMRDATE = '09/17/1999'
         COMPUTE NUMSTAR = T6FTRNCT / 10
         COMPUTE NUMSTAR = NUMSTAR + 1
         IF NUMSTAR > 51 THEN BEGIN
             COMPUTE NUMSTAR = 52
         END;
SET &GRAPH = BB(1:&NUMSTAR)
USING CMRDATE -
CMRTIME -
T6FTRNCT -
GRAPH
REPORT
END
```
A variable BB is used to define a string of asterisks (*); the substring feature of the SET statement is used to pick as many asterisks (*) as needed to represent the value. The minimum activity is 15 transactions in the 5-minute period, since MainView for CICS uses this many transactions to monitor the system. T6FTRNCT was divided by 5 to extract activity for a particular day that was not busy. If the line were all asterisks, this would represent a count in excess of 250 transactions in the 5-minute interval. If your daily transaction rate is 50,000, 100,000, or more, divide by 10, 20, or even 30.

A look at the sample output in Figure 57 on page 189 shows the highs and lows of the day’s activity. Use T6FTRMCT instead of T6FTRNCT if you are interested in only terminal activity as opposed to asynchronous tasks; add other CC fields to the USING statement as needed.
Adding a particular time of day in a WHERE clause, as follows, permits you to chart, on a single page, information about peak periods, such as 12:00 P.M. to 12:59 P.M., over several months; see Figure 58 on page 190:

```
SELECT TYPE CC RECORDS FROM CMRSTATS
  SET TIME = CMRTIME(1:2)
WHERE TIME = '12'
```

**Figure 58: Time versus activity over several months**

<table>
<thead>
<tr>
<th>RECORDED DATE</th>
<th>TOTAL TASKS</th>
<th>GRAPH</th>
</tr>
</thead>
<tbody>
<tr>
<td>09/03/1999</td>
<td>102</td>
<td></td>
</tr>
<tr>
<td>09/04/1999</td>
<td>171</td>
<td></td>
</tr>
<tr>
<td>09/05/1999</td>
<td>147</td>
<td></td>
</tr>
<tr>
<td>09/09/1999</td>
<td>113</td>
<td></td>
</tr>
<tr>
<td>09/07/1999</td>
<td>232</td>
<td></td>
</tr>
<tr>
<td>09/10/1999</td>
<td>253</td>
<td></td>
</tr>
<tr>
<td>10/09/1999</td>
<td>199</td>
<td></td>
</tr>
<tr>
<td>10/09/1999</td>
<td>237</td>
<td></td>
</tr>
<tr>
<td>10/13/1999</td>
<td>229</td>
<td></td>
</tr>
<tr>
<td>10/14/1999</td>
<td>273</td>
<td></td>
</tr>
<tr>
<td>10/17/1999</td>
<td>383</td>
<td></td>
</tr>
<tr>
<td>10/18/1999</td>
<td>279</td>
<td></td>
</tr>
<tr>
<td>10/19/1999</td>
<td>312</td>
<td></td>
</tr>
<tr>
<td>10/20/1999</td>
<td>275</td>
<td></td>
</tr>
<tr>
<td>10/21/1999</td>
<td>399</td>
<td></td>
</tr>
<tr>
<td>10/24/1999</td>
<td>360</td>
<td></td>
</tr>
<tr>
<td>10/25/1999</td>
<td>409</td>
<td></td>
</tr>
<tr>
<td>10/26/1999</td>
<td>337</td>
<td></td>
</tr>
<tr>
<td>10/27/1999</td>
<td>424</td>
<td></td>
</tr>
<tr>
<td>10/28/1999</td>
<td>339</td>
<td></td>
</tr>
<tr>
<td>11/02/1999</td>
<td>328</td>
<td></td>
</tr>
<tr>
<td>11/03/1999</td>
<td>288</td>
<td></td>
</tr>
<tr>
<td>11/04/1999</td>
<td>254</td>
<td></td>
</tr>
<tr>
<td>11/05/1999</td>
<td>270</td>
<td></td>
</tr>
</tbody>
</table>

**Service level analysis graph**

This topic provides a table.

<table>
<thead>
<tr>
<th>Member</th>
<th>CMRPRL7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Objective</td>
<td>to graphically represent service levels as defined by CMRSOPT and collected by MainView for CICS</td>
</tr>
<tr>
<td>Data</td>
<td>Service Level Analysis records (Type CA) from the CMRSTATS file Each record contains 18 service level counts. This data is collected only if LTX= is set to more than 0 in CMRSOPT.</td>
</tr>
</tbody>
</table>
Scenario

MainView for CICS produces service level records according to what is set in the LTX SMF parameter in CMROPT. CMROPT has 17 default service levels defined, ranging from half a second to 90 seconds. In addition, an extra service level is kept, recording response times greater than 90 seconds. The ranges of values in the various service levels can be changed through the customization macro CMROPT and CMROPT administrative view.

A different recording frequency can be specified on the Service Level Analysis field of the PERFORMANCE REPORTER Statistics File Recording Frequencies screen. Note that the transaction counts only involve transactions that run at terminals and are, therefore, not biased by fast asynchronous transactions.

The control statements shown in Figure 59 on page 191 show how service levels can be represented. The example in Figure 59 on page 191 is not intended to cover instances where there has been no activity whatsoever. Thus, the WHERE clause presumes that this time frame is idle when there has been no activity in the first three LTX values. If this assumption is incorrect for your site, repeat the OR clause with LTX004, LTX005, and so on as far as is necessary.

The LTX fields in the FOR loop in the above example are represented by as many asterisks (*) as needed by substringing field \texttt{BB}. If there are more than 50 asterisks (*), the plus (+) is added. If you have a lot of asterisks (*) in your report, increase the frequency of CA records on the file by reducing the interval between records through transaction FIC2 or by dividing the transaction count by a scaling factor. Each line containing the LTX values and their graphical representation is then printed.

Ideally, the chart shows a high percentage of transactions appearing in the first service level, with the frequency rapidly diminishing as the higher service level buckets are reached. Any activity above the ninth or tenth service level slot, with a response time of seven seconds or more, is cause for concern. At this point, the time and date from the 6E records can be noted for a detailed analysis.

Figure 59: Service level analysis PRL statements

```pascal
SET CLASS = 'R'
SET REPORTID = 'PRL7'
SET TITLE = 'SERVICE LEVEL ANALYSIS'
SET WIDTH = '86'
SET LINES = '43'
DEFINE NUMTRAN EDIT Z,ZZZ,ZZ9
DEFINE NUMSTAR EDIT 99999999999999
SET BB = '|***************************************************+
SET GRAPHS = '|....5....10...15...20...25...30...35...40...45...50'
SET GRAPH = '
READ:
SELECT TYPE CA RECORDS FROM CMRSTATS
WHERE CMRDATE = '09/17/2007'
AND LTX001 > 0
OR LTX002 > 0
OR LTX003 > 0
USING CMRDATE - CMRTIME
PRINT 
PRINT CMRDATE CMRTIME ' ' GRAPHS
PRINT ' RESPONSE COUNT | (SCALE = 1 : 5 (ROUNDED))'
FOR &IX = 001 TO 018
  COMPUTE NUMSTAR = LTX(&IX) / 5
  COMPUTE NUMSTAR = NUMSTAR + 1
  IF LTIM(&IX) = 0 THEN SET LTIM(&IX) = ' 100.000'
  IF NUMSTAR > 51 THEN BEGIN
    COMPUTE NUMSTAR = 52
  END;
END;
```
As shown in the report in Figure 60 on page 192, the CMRDATE and CMRTIME field contents are reported when the REPORT statement is processed. The headings, RESPONSE, COUNT, and the scale are printed when the PRINT statement is processed. Enclosing a character string in single quotation marks defines the characters to be printed. To print the value of the variable, specify a defined variable, not enclosed in single quotation marks, with PRINT.

**Note**

To use the PRINT statement, as in this example, the number of data lines per page must be defined, for example, SET LINES=21, and the reporting of data within those lines must be controlled, for example, FOR &IX = 001 TO 018. SET LINES determines each page break and generates a new page with titles and column headings.

**Figure 60: Service level analysis output**

<table>
<thead>
<tr>
<th>PRL7</th>
<th>SERVICE LEVEL ANALYSIS</th>
<th>VERS: REL5.4.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>TIME: 12:10:56</td>
<td>DATE: 09/17/1999</td>
<td>PAGE: 1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RECORDED DATE</th>
<th>RECORDED TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>09/17/1999</td>
<td>15:59:00</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RESPONSE</th>
<th>COUNT</th>
<th>(SCALE = 1 : 5 (ROUNDED))</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.500</td>
<td>4,687</td>
<td>***********************************************************+</td>
</tr>
<tr>
<td>1.000</td>
<td>281</td>
<td>***********************************************************+</td>
</tr>
<tr>
<td>1.500</td>
<td>153</td>
<td>*******************************</td>
</tr>
<tr>
<td>2.000</td>
<td>123</td>
<td>*************************</td>
</tr>
<tr>
<td>2.500</td>
<td>36</td>
<td>*******</td>
</tr>
<tr>
<td>3.000</td>
<td>5</td>
<td>*</td>
</tr>
<tr>
<td>4.000</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>5.000</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>6.000</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>7.000</td>
<td>4</td>
<td>*</td>
</tr>
<tr>
<td>8.000</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>9.000</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>10.000</td>
<td>3</td>
<td>**</td>
</tr>
<tr>
<td>20.000</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>30.000</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>60.000</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>90.000</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>34359.738</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

**Summarizing data**

Each selected record is reported on one line, which can cause the report to become quite large.
Use the PRL statement SUMMARY to report all of the records on just two lines whenever there is a control-break in the report, as shown in Figure 61 on page 193 and in Figure 62 on page 193. A control-break is the point in a report when the first variable specified with USING changes. For this reason, it is best to always make the first USING variable the same as the ORDER BY variable.

**Figure 61: SUMMARY PRL statement**

```
SET CLASS = 'R'
SET REPORTID = 'REPORT1'
SET TITLE = 'TRANSACTION ANALYSIS FOR ALL TRANSACTIONS'
SELECT TYPE 6E RECORDS FROM CMRDETL
USING T6ETRID -
  T6ERESP -
  T6ECPUR -
  T6EFCWT -
  T6ESHWM -
  T6EPAGCT
ORDER BY T6ETRID
SUMMARY
REPORT
END
```

The PRL for this report is in the CMRPRL3 member of the sample library (BBSAMP). As shown in Figure 62 on page 193, the first of the two summary lines contains totals for all numeric fields. The second line contains the computed averages. The totals and averages are shown whenever there is a change in the first reported variable. These two lines of totals and averages also are shown when the PRL statement DETAIL is used. One selected record is reported per line, and when the first variable changes, the totals and averages are reported as SUMMARY.

**Note**

Character data only can be included in a summary report if it is specified as the first USING variable.

**Figure 62: Transaction Analysis for All Transactions summary report**

<table>
<thead>
<tr>
<th>TRAN NAME</th>
<th>TOTAL</th>
<th>RESPONSE TIME</th>
<th>CPU REAL TIME</th>
<th>FILE_CONTROL I/O TIME</th>
<th>STORAGE HI_WATER_MARK</th>
<th>PAGE_COUNT</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSGM</td>
<td>52</td>
<td>29.530</td>
<td>0.264</td>
<td>0.000</td>
<td>354984</td>
<td>58</td>
</tr>
<tr>
<td>AVG *****</td>
<td></td>
<td>0.568</td>
<td>0.005</td>
<td>0.000</td>
<td>6827</td>
<td>1</td>
</tr>
<tr>
<td>CSMT</td>
<td>6</td>
<td>4.380</td>
<td>0.062</td>
<td>0.000</td>
<td>72952</td>
<td>13</td>
</tr>
<tr>
<td>AVG *****</td>
<td></td>
<td>0.730</td>
<td>0.018</td>
<td>0.000</td>
<td>12159</td>
<td>2</td>
</tr>
<tr>
<td>CSPG</td>
<td>13</td>
<td>0.418</td>
<td>0.022</td>
<td>0.000</td>
<td>102048</td>
<td>18</td>
</tr>
<tr>
<td>AVG *****</td>
<td></td>
<td>0.732</td>
<td>0.002</td>
<td>0.000</td>
<td>7850</td>
<td>1</td>
</tr>
<tr>
<td>CSSF</td>
<td>33</td>
<td>24.234</td>
<td>0.496</td>
<td>0.000</td>
<td>321368</td>
<td>137</td>
</tr>
<tr>
<td>AVG *****</td>
<td></td>
<td>0.734</td>
<td>0.015</td>
<td>0.000</td>
<td>9738</td>
<td>4</td>
</tr>
<tr>
<td>CSSN</td>
<td>4</td>
<td>2.339</td>
<td>0.021</td>
<td>0.000</td>
<td>31776</td>
<td>7</td>
</tr>
<tr>
<td>AVG *****</td>
<td></td>
<td>0.585</td>
<td>0.005</td>
<td>0.000</td>
<td>7944</td>
<td>2</td>
</tr>
<tr>
<td>CSST</td>
<td>5</td>
<td>0.206</td>
<td>0.017</td>
<td>0.000</td>
<td>39432</td>
<td>0</td>
</tr>
<tr>
<td>AVG *****</td>
<td></td>
<td>0.041</td>
<td>0.003</td>
<td>0.000</td>
<td>7886</td>
<td>0</td>
</tr>
<tr>
<td>DB2</td>
<td>16</td>
<td>9.879</td>
<td>0.224</td>
<td>0.000</td>
<td>99040</td>
<td>1</td>
</tr>
<tr>
<td>AVG *****</td>
<td></td>
<td>0.617</td>
<td>0.014</td>
<td>0.000</td>
<td>6190</td>
<td>0</td>
</tr>
<tr>
<td>DISP</td>
<td>3</td>
<td>7.617</td>
<td>0.756</td>
<td>0.000</td>
<td>39328</td>
<td>37</td>
</tr>
<tr>
<td>AVG *****</td>
<td></td>
<td>2.539</td>
<td>0.262</td>
<td>0.000</td>
<td>13109</td>
<td>12</td>
</tr>
<tr>
<td>FC02</td>
<td>6</td>
<td>5.147</td>
<td>0.088</td>
<td>0.000</td>
<td>88624</td>
<td>8</td>
</tr>
</tbody>
</table>
Sorting by more than one field

To sort by more than one field and report these fields:

- Set a variable to more than one field.
- Specify the variable name with USING and ORDER BY.

An example is shown in Figure 63 on page 194.

**Figure 63: Set variables to sort by more than one field PRL statement**

```plaintext
SET CLASS = 'R'
SET REPORTID = 'REPORT1'
SET TITLE = 'TRANSACTION RESPONSE ANALYSIS BY RESPONSE RANGE'
SELECT TYPE 6E RECORDS FROM CMRDETL
SET &VAR2 = '9> '
IF &T6ERESP < 9 THEN SET &VAR2 = '8-9'
IF &T6ERESP < 8 THEN SET &VAR2 = '7-8'
IF &T6ERESP < 7 THEN SET &VAR2 = '6-7'
IF &T6ERESP < 6 THEN SET &VAR2 = '5-6'
IF &T6ERESP < 5 THEN SET &VAR2 = '4-5'
IF &T6ERESP < 4 THEN SET &VAR2 = '3-4'
IF &T6ERESP < 3 THEN SET &VAR2 = '2-3'
IF &T6ERESP < 2 THEN SET &VAR2 = '1-2'
IF &T6ERESP < 1 THEN SET &VAR2 = '0-1'
SET &VAR3 = T6ETRID
SET ID-RANGE = '&VAR3. &VAR2. '
USING ID-RANGE -
```

---

```
AVG *****          0.858          0.015          0.000            14771                1
FIC2      28922       6788.137        132.645       2275.220        305450768            14799
AVG *****          0.235          0.005          0.079            10561                1
FST2          4          2.539          0.014          0.000            54400               28
AVG *****          0.635          0.003          0.000            13600                7
HST2         72         15.729          1.789          8.092          1259048               23
AVG *****          0.218          0.025          0.112            17487                0
JNL2         56     156338.884         13.032          0.000           409264              397
AVG *****       2791.766          0.233          0.000             7308                7
JPH1        929      10020.464          4.830          0.000        599785928                5
AVG *****         10.786          0.005          0.000           645625                0
SMN2         62         19.769          0.308          0.000           436824               1
AVG *****          0.319          0.005          0.000             7046                1
STAR       5360        308.284         10.726          0.000         32214644                2
AVG *****          0.058          0.002          0.000             6010                0
TSM2         30        964.988          0.327          0.000           91808               17
AVG *****         32.166          0.011          0.000            30394                1
TSM4         16         72.636          0.235          0.000           483792               14
AVG *****           4.540          0.015          0.000            30237                1
TSM5         15        160.559          0.262          0.000           450166                3
AVG *****          10.704          0.017          0.000            30334                0
```
This technique is used in the case study described in “Creating longest transaction response time reports” on page 196.

**Reporting 4GL database activity**

To retrieve and process the fourth-generation language (4GL) database activity collected in the CMRDETL data sets, use PRL and JCL DD statements to specify the input for your reports.

The databases and 4GLs are identified by the following field names used in PRL statements:

- **T6EPGNM** specifies the program name of any 4GL or database (except MANTIS, which is identified through T6EMNTIS) represented by T6EPTYPE.

- **T6EPTYPE** specifies the program name type identified by a 1-character database or 4GL designation.

- **T6EFI** specifies the file type identified by a 1-character file or database name represented by T6EFN.

- **T6EFN** specifies the file or database name.

For example, the statements shown in Figure 64 on page 195, submitted as an MVS batch job, report on Software AG Natural performance on a given day.

**Figure 64: NATURAL performance PRL statements**

```plaintext
SET TITLE = 'NATURAL RESPONSE TIMES'
SET TITLE1 = 'On 09/17/1999'

SELECT TYPE 6E RECORDS FROM CMRDETL
WHERE CMRDATE = '09/17/1999'
AND T6EPTYPE = 'N'

USING CMRTIME -
T6EPGNM -
T6EUSER -
T6ERESP -
T6EFCNT

REPORT
```
Creating longest transaction response time reports

Detailed transaction information stored on the CMRDETL data set can be used to create daily reports.

Sample member CMRPRL14 of your BBSAMP data set creates a report from 6E detail transaction records. This report shows the 30 longest transactions recorded over a daily interval.

Figure 65 on page 196 shows sample PRL statements from member CMRPRL14 to create a report of the transactions with the longest response time. In this example, the control statements restrict the report to terminal transactions that occurred on a single day.

Figure 65: Longest transaction response time PRL statements

```
SET CLASS = 'R'
SET REPORTID = 'PRL14'
SET TITLE = 'WORST RESPONSE TIMES'
SET TITLE1 = 'RESTRICTED TO 09/18/1999 '
SET TITLE2 = 'TERMINAL ATTACHED TRANSACTIONS ONLY'
SELECT TYPE 6E RECORDS FROM CMRDETL
   WHERE CMRDATE = '09/18/1999'
   AND T6ETMID ¬= '    '
   COMPUTE LINE_I/O = T6ECHARI + T6ECHARO
USING CMRTIME -
   T6ETRID -
   T6EPGNM -
   T6RESP -
   T6ECPUR -
   T6ETMID -
   T6EOPID -
   T6FCWCT -
   T6FCCAL -
   LINE_I/O
ORDER BY T6ERESP IN DESCENDING SEQUENCE
   SET COUNT = COUNT + 1
   IF COUNT > 30 THEN GOTO DDD
REPORT
DDD:
PRINT '     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  * '
PRINT '     *                                               *'
PRINT '     *  W O R S T    3 0    R E S P O N S E S        *'
PRINT '     *                                               *'
PRINT '     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  * '
END
```
Figure 66 on page 197 shows the left-most portion of an example report created by these PRL control statements. This report lists 30 transactions with the longest response times over a daily period.

**Figure 66: Longest transaction response times report**

<table>
<thead>
<tr>
<th>RECORDED TIME</th>
<th>TRAN NAME</th>
<th>PROGRAM NAME</th>
<th>RESPONSE TIME</th>
<th>CPU REAL TIME</th>
<th>TERM ID.</th>
<th>OPR ID.</th>
<th>FILE</th>
<th>CONTROL</th>
</tr>
</thead>
<tbody>
<tr>
<td>10:22:23</td>
<td>DGNH</td>
<td>LDG0000</td>
<td>33.515</td>
<td>0.629</td>
<td>SC32</td>
<td>DJD</td>
<td>0.377</td>
<td></td>
</tr>
<tr>
<td>16:55:30</td>
<td>PCXR</td>
<td>PCXRDRVR</td>
<td>20.658</td>
<td>0.028</td>
<td>L072</td>
<td>RGN</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>13:10:21</td>
<td>DRV5</td>
<td>ADRIVER</td>
<td>16.632</td>
<td>0.156</td>
<td>L072</td>
<td>RGN</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>16:55:42</td>
<td>GRSD</td>
<td>GR52500</td>
<td>16.577</td>
<td>0.387</td>
<td>SB24</td>
<td></td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>16:55:58</td>
<td>PCXR</td>
<td>PCXRDRVR</td>
<td>13.263</td>
<td>0.027</td>
<td>L072</td>
<td>RGN</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>17:14:10</td>
<td>GRSD</td>
<td>GR52500</td>
<td>11.990</td>
<td>0.384</td>
<td>SB24</td>
<td></td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>10:22:45</td>
<td>DGNH</td>
<td>SYSEDIR</td>
<td>8.708</td>
<td>0.078</td>
<td>L064</td>
<td></td>
<td>0.068</td>
<td></td>
</tr>
<tr>
<td>17:08:02</td>
<td>GRSD</td>
<td>GR52500</td>
<td>8.537</td>
<td>0.388</td>
<td>SB24</td>
<td></td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>12:03:52</td>
<td>FSET</td>
<td>CMRRMP</td>
<td>8.370</td>
<td>0.078</td>
<td>SB24</td>
<td></td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>17:06:45</td>
<td>GRSD</td>
<td>GR52500</td>
<td>7.713</td>
<td>0.405</td>
<td>SB24</td>
<td></td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>15:58:43</td>
<td>DGNH</td>
<td>SYSEDIR</td>
<td>7.201</td>
<td>0.087</td>
<td>SC32</td>
<td>DJD</td>
<td>4.279</td>
<td></td>
</tr>
<tr>
<td>17:02:34</td>
<td>GRSD</td>
<td>DF20001</td>
<td>6.490</td>
<td>0.020</td>
<td>SC26</td>
<td></td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>15:58:35</td>
<td>GRSD</td>
<td>GR52500</td>
<td>6.402</td>
<td>0.418</td>
<td>SB24</td>
<td></td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>11:28:59</td>
<td>CSPK</td>
<td>DFHRPRK</td>
<td>5.696</td>
<td>0.113</td>
<td>L064</td>
<td></td>
<td>0.068</td>
<td></td>
</tr>
<tr>
<td>16:13:18</td>
<td>GRSD</td>
<td>GR52500</td>
<td>5.368</td>
<td>0.418</td>
<td>SB24</td>
<td></td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>13:07:07</td>
<td>CMTM</td>
<td>DFHEMTP</td>
<td>4.620</td>
<td>0.953</td>
<td>L072</td>
<td>RGN</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>12:02:53</td>
<td>CSMG</td>
<td>DFHGMN</td>
<td>4.612</td>
<td>0.017</td>
<td>SB24</td>
<td></td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>16:52:40</td>
<td>PCXR</td>
<td>PCXRDRVR</td>
<td>4.494</td>
<td>0.026</td>
<td>L072</td>
<td>RGN</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>16:10:26</td>
<td>GRSD</td>
<td>GR52500</td>
<td>4.418</td>
<td>0.384</td>
<td>SB24</td>
<td></td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>10:22:00</td>
<td>PCXR</td>
<td>PCXRDRVR</td>
<td>4.270</td>
<td>0.022</td>
<td>L064</td>
<td></td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>17:14:06</td>
<td>PCXR</td>
<td>PCXRDRVR</td>
<td>4.253</td>
<td>0.025</td>
<td>L072</td>
<td>RGN</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>16:04:32</td>
<td>GRSD</td>
<td>GR52500</td>
<td>4.151</td>
<td>0.395</td>
<td>SB24</td>
<td></td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>15:58:41</td>
<td>GRSD</td>
<td>GR52500</td>
<td>4.034</td>
<td>0.386</td>
<td>SB24</td>
<td></td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>16:42:13</td>
<td>GRSD</td>
<td>GR52500</td>
<td>3.984</td>
<td>0.436</td>
<td>SB24</td>
<td></td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>15:28:49</td>
<td>DGNH</td>
<td>LDG0000</td>
<td>3.949</td>
<td>0.098</td>
<td>SC32</td>
<td>DJD</td>
<td>3.823</td>
<td></td>
</tr>
<tr>
<td>17:12:50</td>
<td>PCXR</td>
<td>PCXRDRVR</td>
<td>3.907</td>
<td>0.035</td>
<td>L072</td>
<td>RGN</td>
<td>0.248</td>
<td></td>
</tr>
<tr>
<td>16:14:21</td>
<td>PCXR</td>
<td>PCXRDRVR</td>
<td>3.615</td>
<td>0.021</td>
<td>L072</td>
<td>RGN</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>16:13:05</td>
<td>FRED</td>
<td>GR52500</td>
<td>3.570</td>
<td>0.380</td>
<td>SB24</td>
<td></td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>13:28:47</td>
<td>CMTM</td>
<td>DFHEMTP</td>
<td>3.561</td>
<td>0.929</td>
<td>L072</td>
<td>RGN</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>13:31:58</td>
<td>GRSD</td>
<td>GR52500</td>
<td>3.559</td>
<td>0.390</td>
<td>SB24</td>
<td></td>
<td>0.000</td>
<td></td>
</tr>
</tbody>
</table>

Transactions are sorted in descending order by response time. The longest transaction is at the top of the report.
Creating longest transaction response time reports
Symbols

& reserved word 168

A

abended transactions, sample PRL report 134
activity for a specific program, sample PRL summary report 134
ALARM reserved word 152
ALTERVS reserved word 152
AND reserved word 152
APPL CLASS control statement 81
applications class control statement 81
APPLID control statement 34
archive data, selecting 183
archiving CMRDETL data
  single data set 27

B

BBPLIB data set
  specifying in JCL 140
  using with PRL 181
BBSAMP
  CMRPRL sample programs 134
  PRL sample programs 134
BBSAMP data set
  using 13
BLINK reserved word 152

C

CALC=PERCENT 85
CCMD reserved word 152
CDATE reserved word 138, 152
CICS 110 SMF format. 29

CICS CLASS control statement 83
CLASS control statements
  APPL 81
  CICS 83
  CSA 86
  CSO 86
  DCT 87
  DLZ 88
  FCT 85
  JCT 87
  overview 81
  PCT 84
  PPT 84
  SERVICE 82
  TCT 89
  TST 90
CMRASM 65
CMRCB11N 106, 111
CMRCB12N 106, 112
CMRCB13N 106, 113
CMRCB14N 106, 114
CMRCB15N 106, 115
CMRCB21N 106, 116
CMRCB23N 106, 116
CMRCB25N 106, 118
CMRCHRT batch program 75
CMRCHRT reports
  control statements
    CLASS 90
    DATA 81, 90
    END 90
    keyword 78
    overview 77
  creating 75
  description 75
  examples 91
  JCL 75
CMRCHRT reports control statements
  CLASS 81
CMRCJCL 20
CMRDETL fields altered 62
combining summarized records 63
code statements
  coding 42
  COPY 51
  COPYAPPL 51
  DETAIL 48
  EXCLxxxx 46
FILES 48
FROM DATE 49
FROM TIME 50
INCLxxxx 47
MAXCPU 51
MAXRESP 51
MINCPU 51
MINRESP 51
MINUTES 45
NEWAPPL 47
overview 42
PROCFLIO 51
REPLACE 48
SGROUP 46
SKIPFLIO 51
START TIME 50
STOP TIME 50
TABLE 51
TO DATE 49
TO TIME 50
TYPE 45
USEREXIT 51
WORKLOAD 44
ZERO_DATA 49
creating PRL reports 56
data types altered 63
description 44
detail file switch 59
JCL 41, 53
purging summary records 28
replacing summary data 60
sample jobstream 53
sample output 53
selecting input records 58
storing summary data 61
usage notes 58
viewing summary records 60
CMRTIME fields 175
CMRURAPM 65
CMRUSRD 51
COBOL reports
accessing CMRDETL records 102
CMRCOBCN program 103
CMRFREDN program 102
common fields 110
code statements 108
customizing 105
description 101
eachmes
Abended Transaction Report 118
File Analysis for All Files 116
overview 106
Performance Analysis by Netname 114
Performance Analysis by Program 112
Performance Analysis by Terminal 113
Performance Analysis by User ID 115
Performance Analysis for All
Transactions 111
Rogues Report 116
JCL 108
collecting data 19
COLOR reserved word 152
Common System Area class control statement 86
compressed CMRDETL records, reformatting 36
COMPUTE PRL statement 154
COMPUTE reserved word 152
code statements
  CLASS 81, 90
  CMRSUMDN utility 42, 51
  COBOL 108
  DATA 81, 90
  histogram 77
  Resource Tracking and Analysis 68
  SAS 122
  Service Level Analysis 98
conventions, documentation 12
converting CMRSTATS records 33
counts by record types from CMRDETL and
CMRSTAT files sample PRL report 134
CSA CLASS control statement 86
CSO CLASS control statement 86
CTGDETL
  transaction detail 23
CTIME reserved word 152
CURSOR reserved word 152
CYCLE reserved word 152

D

DATA control statements
G
GAPPLID control statement 34
GCMD reserved word 152
GET reserved word 152
GOTO PRL statement 161
GOTO reserved word 152
GRAF reserved word 152
graph reports
see also histogram reports 75
using PRL 187
Graphic Histogram Report 17

H
HEADING reserved word 152
Help
online 11
HELP reserved word 152
highest users of CPU time, sample PRL report 134
HIGHLIGHT reserved word 152
histogram reports
creating 75
description 75
examples 91
JCL 75
histogram reports control statements
CLASS 81, 90
DATA 81, 90
END 90
keyword 78
overview 77
HISTORY reserved word 152
HISTORY service 40, 60, 61

I
I/O activity sample PRL report 134
IF and THEN PRL statements 162
IF reserved word 152
INCLxxxx control statement 58
indexing with PRL
by qualified variable[indexing with
PRL:qualified variable] 187
CMRDETL records 185

J
JCL members
CMRL 139
CMRMVJCL 139
CMRPURGE 25
CMRSTATX 23
CMRSTJCL 33
CMRSUMD1 53
Journal Control Table (JCT), CLASS control
statement 87

L
LEAVE= 25
LET PRL statement 163
LET reserved word 152
LIN001 reserved word 152
LOGOFF reserved word 152

M
MODNAME value 65
MQ data from variable file entries sample PRL
report 134

N
NEW reserved word 152
NEWAPPL control statement 60
NEXT reserved word 152

O
online Help 11
OR reserved word 152
ORDER BY PRL statement
description 164
examples 164, 194
ORDER reserved word 152

P
PAD reserved word 152
PANELID reserved word 152
PCT class control statement | 84
---|---
PERFORMANCE REPORTER
- archiving data | 25
- collecting data | 19
- description | 13
- report types | 16
Performance Reporting Language (PRL)

<table>
<thead>
<tr>
<th>Control Statement Verbs</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>COMPUTE</td>
<td>154</td>
</tr>
<tr>
<td>DEFINE</td>
<td>155</td>
</tr>
<tr>
<td>DETAIL</td>
<td>157</td>
</tr>
<tr>
<td>DIAGNOSE</td>
<td>158</td>
</tr>
<tr>
<td>END</td>
<td>159</td>
</tr>
<tr>
<td>example</td>
<td>142, 146</td>
</tr>
<tr>
<td>FOR and NEXT</td>
<td>159</td>
</tr>
<tr>
<td>GOTO</td>
<td>161</td>
</tr>
<tr>
<td>IF and THEN</td>
<td>162</td>
</tr>
<tr>
<td>LET</td>
<td>163</td>
</tr>
<tr>
<td>ORDER BY</td>
<td>164</td>
</tr>
<tr>
<td>overview</td>
<td>149</td>
</tr>
<tr>
<td>PRINT</td>
<td>165</td>
</tr>
<tr>
<td>REPORT</td>
<td>165</td>
</tr>
<tr>
<td>SCAN</td>
<td>166</td>
</tr>
<tr>
<td>SELECT</td>
<td>167</td>
</tr>
<tr>
<td>SET</td>
<td>168</td>
</tr>
<tr>
<td>SUMMARY</td>
<td>172</td>
</tr>
<tr>
<td>syntax</td>
<td>150</td>
</tr>
<tr>
<td>USING</td>
<td>173</td>
</tr>
<tr>
<td>WHERE</td>
<td>175</td>
</tr>
</tbody>
</table>

| JCL | 139 |
| Operators | 136 |
| Overview | 133 |
| Processing Flow | 143 |
| Program Syntax | 134 |
| Programming Techniques | |
| Creating response time reports | 196 |
| Formatting report output | 181 |
| Indexing by qualified variable | 187 |
| Indexing CMRDETL records | 185 |
| Producing graphs | 187 |
| Reporting 4GL database activity | 195 |
| Retrieving selected records | 184 |
| Routing report output | 179 |
| Selecting archive data | 183 |
| Sorting data | 194 |
| Summarizing data | 192 |
| Reserved Words | 152 |
| Sample Programs | 134 |
| Substrings | 137 |
| Using CMRSUMDN utility | 56 |
| Variables | 138 |
| PRINT PRL statement | 165 |
| PRINT reserved word | 152 |
| PRL, See Performance Reporting Language | 133 |
| Processing Program Table (PPT), CLASS control statement | 84 |
| PROFCNT reserved word | 152 |
PROFMEM reserved word 152
PROFRC reserved word 152
program control
detail sample PRL report 134
summary sample PRL report 134
Program Control Table (PCT), CLASS control statement 84
program timing data sample PRL report 134
publications, related 11
PUT reserved word 152

RAND reserved word 152
records
creating CICS statistical 33
purging CMRSUMDN 28
SMF 110 23
Type 6D and 6E transaction detail 20
reformatting compressed CMRDETL records 36
related publications 11
REPLACE= 60
REPORT PRL statement 165
REPORT reserved word 152
reports
creating 13
custom 18
histogram 75
long-term 17
predefined 16
Resource Tracking and Analysis 65
Service Level Analysis 97
short-term 16
reports, COBOL
Abended Transaction Report 118
File Analysis for All Files 116
Performance Analysis by Netname 114
Performance Analysis by Program 112
Performance Analysis by Terminal 113
Performance Analysis by User ID 115
Performance Analysis for All Transactions 111
Rogues Report 116
reports, SAS
Abended Transaction Roster 129
File Analysis for All Files 127
Operational Statistics Report 130
Performance Analysis by Node Name 126
Performance Analysis by Program 124
Performance Analysis by Terminal 125
Performance Analysis by User ID 126
Performance Analysis for All Transactions 123
Rogues Report 128
reserved word 152
reserved words 152
Resource Analysis Report 17
Resource Tracking and Analysis reports
control statements 68
description 65
eexample 69
field descriptions 71
JCL 65
macro assembly statements 67
resource usage, summary sample PRL report 134
response time analysis
by response range sample PRL report 134
for MRO data sample PRL report 134
for terminal attached transactions sample PRL report 134
RETURN reserved word 152
REVERSE reserved word 152
Rogues Reports
COBOL 116
SAS 128
routing PRL output 179
RVIDEO reserved word 152

SAPPLID control statement 34
SAS reports

Index 205
Abended Transaction Roster example 129
common fields 122
collection statements 122
description 119
File Analysis for All Files example 127
Operational Statistics Report example 130
overview 119
Performance Analysis by Node Name example 126
Performance Analysis by Program example 124
Performance Analysis by Terminal example 125
Performance Analysis by User ID example 126
Performance Analysis for All Transactions example 123
Rogues Report example 128
SCAN PRL statement 122
SCAN reserved word 152
SCMD reserved word 152
SELECT PRL statement
description 167
elements 167, 183
SELECT reserved word 152
SERVICE CLASS control statement 82
service level analysis graph, sample PRL report 134
Service Level Analysis reports
collection statements 98
description 97
example 99
field descriptions 99
JCL 97
sample PRL report 134
session data sample PRL report 134
SET PRL statement 168
SET reserved word 152
SMF 110 records 23
SQL activity, by DB2 plan PRL report 134
summarization utility 58
summarized data from CMRSUMDN sample PRL report 134
summarizing CMRDETL records 40
SUMMARY PRL statement
description 172
examples 172, 192
SUMMARY reserved word 152
SUMMDATA 60
SYMBOLIZE reserved word 152
system management facility (SMF) 23

T
T6EDL 186
T6EEB 186
T6EFC 186
T6EFCNT 185
T6EFI 186
T6EFN 186
T6EFT 186
T6EOT 186
T6EQUAL 185
T6ERB 186
T6ERD 186
T6ERN 186
T6ERP 186
T6ERU 186
T6ESB 186
T6EUL 186
T6EV1 186
T6EV2 186
T6EV3 186
T6EWR 186
T6FTRNCT 181
TABLE control statement 65
TCOL reserved word 152
temporary storage
sample PRL detail report 134
sample PRL summary report 134
Temporary Storage Table (TST), CLASS control statement 90
terminal attached transactions sample PRL report 134
terminal control
sample PRL detail report 134
sample PRL summary report 134
Terminal Control Table (TCT), CLASS control statement 89
terminal errors, sample PRL report 134
TO DATE control statement 34
TO TIME control statement 34
TRACE reserved word 152
transaction analysis for all transactions, sample PRL report 134
transaction class
CLASS control statement 86
transaction class sample PRL report 134
Transaction Class Table class control statements 86
transaction file control wait analysis, sample PRL report 134
transaction response time
  sample PRL summary report 134
transaction response time, sample PRL report 134
TRIM reserved word 152
TROW reserved word 152
TUTOR reserved word 152
Type 6D and 6E transaction detail records
  overview 20

U
UNDERSCORE reserved word 152
Unnnnnnn reserved word 152
USEREXIT control statement 58
USING PRL statement
  description 173
  examples 173, 181
USING reserved word 152

V
VSAM
  storing CMRSUMDN data 61

W
WHERE PRL statement 175
WHERE reserved word 152

Z
ZCARD reserved word 152
ZCICS reserved word 152
ZCSA reserved word 152
ZCT reserved word 152
ZCURSOR reserved word 152
ZCW reserved word 152
ZDATE reserved word 138, 152
ZERRMSG reserved word 152
ZIFX reserved word 152
ZIFY reserved word 152
ZLAST reserved word 152
ZLINE reserved word 152
ZLINES reserved word 152
ZLOC reserved word 152
ZMEMBER reserved word 152
ZMSG reserved word 152
ZOPTERR reserved word 152
ZPF reserved word 152
ZREQUEST reserved word 152
ZSCREEN reserved word 152
ZSHOW reserved word 152
ZTIME reserved word 138, 152
ZTITLE reserved word 152
ZTRAN reserved word 152
ZUSER reserved word 152
ZWRAP reserved word 152
ZXDT reserved word 152