Programming Interfaces

This Tivoli Information Management for z/OS Data Reporting User’s Guide primarily documents information that is NOT intended to be used as Programming Interfaces of Tivoli Information Management for z/OS.

This Tivoli Information Management for z/OS Data Reporting User’s Guide also documents intended Programming Interfaces that allow the customer to write programs to obtain the services of Tivoli Information Management for z/OS. This information is identified where it occurs, either by an introductory statement to a chapter or section or by the following marking: Programming Interface information.
# Contents

**Preface** ................................................................. xiii

- Who Should Read This Document ........................................ xiv
- Prerequisite and Related Documentation .................................. xiv
- What This Document Contains ............................................ xv
- Typeface Conventions ..................................................... xvi
  - The Use of Panel Style in This Document .......................... xvii
- Contacting Customer Support ............................................ xvii

**Part I. Tivoli Decision Support** ...................................... 1

**Chapter 1. Tivoli Decision Support** ................................. 3

- What Tivoli Decision Support Does ....................................... 3
  - Viewing Data .......................................................... 6
  - Information Management Guide ...................................... 9
- Requirements .............................................................. 11

**Part II. Open Database Connectivity (ODBC) Driver** .......... 13

**Chapter 2. Introduction** ............................................... 15

- ODBC Concepts .......................................................... 16
  - Tivoli Information Management for z/OS ODBC Driver .............. 18
  - ODBC Conformance .................................................. 19

**Chapter 3. Installing the ODBC Driver** ............................ 25

- ODBC Driver Requirements ............................................. 25
- Verifying Your HLAPI/NT Setup ........................................ 26
- Installing the ODBC Driver ............................................. 27
  - Serviceability .......................................................... 30
  - Files Installed ........................................................ 30
  - Uninstalling the ODBC Driver ...................................... 32
  - Ensuring Setup of Data Model Records ............................ 32

**Chapter 4. Creating Data Model Records** .......................... 33

- Data Attribute and Data View Records ................................ 33
  - Create Data Attribute Records for Each Column of Data ........... 33
  - Build a Data View Record .......................................... 35

**Chapter 5. Using the ODBC Driver** .................................. 37
Connecting to the Data Source ..................................................... 37
Using Crystal Reports ................................................................. 37
Using Lotus Approach ................................................................. 38
Using Lotus 1-2-3 ........................................................................ 38
Using Microsoft Access ............................................................... 39
Entering Data Source Information .................................................. 39
Defining Queries ........................................................................... 40

Chapter 6. Changing Driver Options or Attributes ....................... 43
Specifying Advanced Options ..................................................... 43

Chapter 7. Messages .................................................................... 47
Message Formats ........................................................................ 47
ODBC Driver Manager Tracing ..................................................... 48
ODBC Driver Tracing ....................................................................... 48
Messages ..................................................................................... 49
Return and Reason Codes Issued by the ODBC Driver by Function .......................... 52
  SQLAllocConnect ........................................................................ 53
  SQLAllocEnv ............................................................................. 54
  SQLAllocStmt ........................................................................... 55
  SQLBindCol .............................................................................. 56
  SQLBindParameter ..................................................................... 57
  SQLCancel ............................................................................... 58
  SQLColAttributes ....................................................................... 59
  SQLColumns .............................................................................. 60
  SQLConnect ............................................................................. 62
  SQLDescribeCol ....................................................................... 63
  SQLDescribeParam .................................................................... 64
  SQLDisconnect .......................................................................... 65
  SQLDriverConnect ..................................................................... 66
  SQLError .................................................................................. 68
  SQLExecDirect .......................................................................... 69
  SQLExecute ............................................................................... 73
  SQLExtendedFetch ..................................................................... 75
  SQLFetch ................................................................................ 77
  SQLFreeConnect ....................................................................... 79
  SQLFreeEnv .............................................................................. 80
  SQLFreeStmt ............................................................................ 81
  SQLGetConnectOption .............................................................. 82
  SQLGetCursorName .................................................................... 83
Chapter 8. SQL Command Syntax and Parameters ................. 107

Part III. Report Format Facility ....................................... 127

Chapter 9. Introducing the Report Format Facility ............... 131

Chapter 10. Modifying and Creating RFTs ......................... 135
Other Considerations ................................................................. 136
Creating an RFT ................................................................. 136
Creating a New RFT ............................................................... 137
Testing an RFT ................................................................. 139
Adding a Data-Collection Panel ................................................ 142
Adding a Menu Panel ............................................................ 143
Modifying an RFT ............................................................... 143
Performance Considerations .................................................... 143
Virtual Storage Requirements .................................................. 143
Processing Time ............................................................... 144

Chapter 11. RFT Statements .................................................. 147
Syntax and Rules .................................................................... 147
Notation Conventions ............................................................ 147
General Rules ...................................................................... 148
Keywords and Keyword Values ................................................ 148
Special Characters .............................................................. 149
Mixed Data ......................................................................... 149
Prefixes .............................................................................. 149
Variables ............................................................................. 149
Dictionary Index Keys ........................................................... 150
Search Levels ....................................................................... 151
Summary of RFT Statements .................................................. 151

Chapter 12. Definition Statements .......................................... 153
EOD - EEOD ........................................................................... 153
Usage Notes ......................................................................... 153
EOF - EEOF ......................................................................... 156
Usage Notes ......................................................................... 156
FORMAT ................................................................................ 157
Usage Notes ......................................................................... 158
HEADING - EHEADING .......................................................... 160
Usage Notes ......................................................................... 160
HFORMAT - EFORMAT .......................................................... 161
Usage Notes ......................................................................... 161
SECTION - ESECTION ............................................................ 161
Usage Notes ......................................................................... 161
TFORMAT - EFORMAT ............................................................ 163
Usage Notes ......................................................................... 164
TITLE - ETITLE ................................................................. 165

Version 7.1
## Chapter 13. Command Statements

<table>
<thead>
<tr>
<th>Command</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALL</td>
<td>167</td>
</tr>
<tr>
<td>Usage Notes</td>
<td>169</td>
</tr>
<tr>
<td>Examples</td>
<td>172</td>
</tr>
<tr>
<td>DHEADS</td>
<td>173</td>
</tr>
<tr>
<td>DO, EDO, LEAVE</td>
<td>174</td>
</tr>
<tr>
<td>Usage Notes</td>
<td>182</td>
</tr>
<tr>
<td>Examples</td>
<td>182</td>
</tr>
<tr>
<td>EJECT</td>
<td>184</td>
</tr>
<tr>
<td>END</td>
<td>184</td>
</tr>
<tr>
<td>IF - ELSE - EIF</td>
<td>185</td>
</tr>
<tr>
<td>Usage Notes</td>
<td>193</td>
</tr>
<tr>
<td>Examples</td>
<td>193</td>
</tr>
<tr>
<td>PUT</td>
<td>194</td>
</tr>
<tr>
<td>Usage Notes</td>
<td>200</td>
</tr>
<tr>
<td>Examples</td>
<td>203</td>
</tr>
<tr>
<td>SEARCH - ESEARCH</td>
<td>205</td>
</tr>
<tr>
<td>Usage Notes</td>
<td>208</td>
</tr>
<tr>
<td>Examples</td>
<td>209</td>
</tr>
<tr>
<td>SET</td>
<td>210</td>
</tr>
<tr>
<td>Usage Notes</td>
<td>215</td>
</tr>
<tr>
<td>Examples</td>
<td>216</td>
</tr>
<tr>
<td>SETD</td>
<td>217</td>
</tr>
<tr>
<td>Usage Notes</td>
<td>222</td>
</tr>
<tr>
<td>Examples</td>
<td>225</td>
</tr>
<tr>
<td>SPACE</td>
<td>227</td>
</tr>
<tr>
<td>TAB</td>
<td>228</td>
</tr>
</tbody>
</table>

## Chapter 14. Report Exit Routines

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Types of Report Exit Routines</td>
<td>231</td>
</tr>
<tr>
<td>Service Functions</td>
<td>232</td>
</tr>
<tr>
<td>Controlling Report Exit Routines</td>
<td>232</td>
</tr>
<tr>
<td>Data Output Exit Routines</td>
<td>233</td>
</tr>
<tr>
<td>Communication Control Block for the Data Output Exit</td>
<td>233</td>
</tr>
<tr>
<td>Return Codes for Data Output Exits</td>
<td>235</td>
</tr>
<tr>
<td>Restrictions for Data Output Exits</td>
<td>235</td>
</tr>
</tbody>
</table>
CALL Command Exit Routines ................................................... 235
Communication Control Block for CALL Command Exits .................. 236
Return Codes for CALL Command Exits ........................................ 237
CALL Command Exit Restrictions ............................................. 237
Predefined CALL Command Exits ............................................ 237
Search and Count ............................................................. 238
Format Search Argument ..................................................... 239
Format Detailed Data ......................................................... 239
Format Response Chains ..................................................... 240
Check “From” and “To” Dates ................................................ 240
Format List Processor Data .................................................. 241
Exit Routine Service Functions ................................................... 243
Locate Service Function–X’D3’ .............................................. 243
Move Service Function–X’D4’ ............................................... 247
Variable Read Service Function–X’D9’ ..................................... 247
Variable Set Service Function–X’E2’ ...................................... 248
Write Service Function–X’E6’ ............................................... 249
Chapter 15. Defining Output Destinations for Reports .................. 251
SYSOUT, DSNAME, and DDNAME ........................................... 251
Chapter 16. Alternative Methods of Running Reports ......... 255
Running Reports with SRCs .................................................. 255
Running Reports in Batch Mode ............................................. 255
Batch Reports under MVS/ESA ............................................... 255
Writing to SYSPRINT .......................................................... 257
Part IV. Appendixes .................................................................. 259
Appendix A. Standard Reports .................................................. 261
Format for Standard Reports .................................................. 262
General Reports ............................................................... 263
Search Results List Report (BLMZZ11) ..................................... 263
Line Summary Report (BLMZZ12) ............................................. 263
Page Summary Report (BLMZZ13) ............................................. 264
Detail Report (BLMZZ14) ....................................................... 264
Member Names of Standard Reports .......................................... 265
PRINT Record Output .......................................................... 266
Report Samples ............................................................... 267
General Reports ............................................................... 268
Appendix E. Predefined Variables .................................................. 363

Read-Only Variables ............................................................... 363
  System Date and Time Variables .................................. 363
  Day-of-the-Week Variables ........................................... 364
  Weekday Date-Range Variables .................................. 365
  Weekly Date-Range Variables ....................................... 366
  Monthly Date-Range Variables ..................................... 367
  Name-of-the-Month Variables ...................................... 368
  Yearly Date-Range Variables ....................................... 368
  User-Specified Date-Range Variables ............................ 369
  Search-Count Variables ................................................ 371
  Sort Variables ............................................................ 371
  IF-Command Variables ................................................ 372
  Map-Processing Variables ............................................ 373
Read/Write Variables ......................................................... 374
  Example of How to Use Read/Write Variables .................. 375

Appendix F. Relating Publications to Specific Tasks .................... 379

Typical Tasks ................................................................. 379

Appendix G. Tivoli Information Management for z/OS Courses ....... 383

Education Offerings .......................................................... 383
  United States .......................................................... 383
Preface

This book describes how you can interact with Tivoli Information Management for z/OS to report on the operations in your enterprise. Although there are various methods you can use to generate and view reports on the data stored in Tivoli Information Management for z/OS, the approach you select depends on the needs of the users and decision makers in your enterprise. The following Tivoli tools and solutions are described in this book:

- **Tivoli Decision Support**
  Tivoli Decision Support (a Tivoli product offering) is a workstation-based solution that combines the power of popular third-party reporting tools to enable you to see various "views" of data in a wide range of graphical formats on your workstation. It is particularly suited for those who need to perform data analysis or data mining on an ongoing basis. Information is provided in this book to help you understand how Tivoli Decision Support can work with Tivoli Information Management for z/OS data in your environment.

- **Open Database Connectivity (ODBC) driver for Tivoli Information Management for z/OS**
  The ODBC driver is provided with Tivoli Information Management for z/OS to facilitate the retrieval of Tivoli Information Management for z/OS data on a Microsoft Windows NT® workstation. You can use ODBC-enabled workstation applications, such as database and spreadsheet applications, to retrieve and manipulate the data to produce reports. Information is provided in this book on how to install and use the ODBC driver.

- **Report Format Facility of Tivoli Information Management for z/OS**
  The Report Format Facility is a host-based facility provided with Tivoli Information Management for z/OS that enables you to generate standard or customized reports. This book presents general information about the Tivoli Information Management for z/OS PRINT and REPORT commands and the report format tables (RFTs) that define Tivoli Information Management for z/OS reports. It also includes examples of modified reports and reference information on the RFT statements used to code reports.

**Note**
Starting with Tivoli Service Desk for OS/390® Version 1.2, support for the host graphics function of the Report Format Facility, which uses the Graphical Data Display Manager (GDDM®), is no longer included. Customers requiring graphics reports can use the ODBC driver and a user-supplied, ODBC-enabled workstation application capable of producing graphic reports, or use Tivoli Decision Support for Information Management (5697-IMG).

For information about creating World Wide Web-based applications to display Tivoli Information Management for z/OS data, refer to the [Tivoli Information Management for z/OS World Wide Web Interface Guide](#).

There may be references in this publication to versions of Tivoli Information Management for z/OS’s predecessor products. For example:

- **TME 10™ Information/Management Version 1.1**
Who Should Read This Document

This book is intended for the following audience:

- Managers, analysts, or administrators who need a high-level understanding of how Tivoli Decision Support can be used with Tivoli Information Management for z/OS.

  **Note:** Information on installing and using Tivoli Decision Support is provided in Tivoli Decision Support publications.

- Users, analysts, or administrators who need to understand ODBC concepts or install or use the ODBC driver; or, programmers writing their own ODBC applications for use with Tivoli Information Management for z/OS.

- System programmers who create or modify reports or graphs through the Report Format Facility. This book assumes you know how to use Tivoli Information Management for z/OS, including how to use the PRINT and REPORT commands (which are described in the *Tivoli Information Management for z/OS User’s Guide*). It also assumes that you know how to program in assembler language and a high-level programming language, such as COBOL or PL/I.

Prerequisite and Related Documentation

The library for Tivoli Information Management for z/OS Version 7.1 consists of these publications. For a description of each, see *“The Tivoli Information Management for z/OS Library” on page 385.*

- *Tivoli Information Management for z/OS Application Program Interface Guide*, SC31-8737-00
- *Tivoli Information Management for z/OS Client Installation and User’s Guide*, SC31-8738-00
- *Tivoli Information Management for z/OS Data Reporting User’s Guide*, SC31-8739-00
- *Tivoli Information Management for z/OS Desktop User’s Guide*, SC31-8740-00
- *Tivoli Information Management for z/OS Diagnosis Guide*, GC31-8741-00
- *Tivoli Information Management for z/OS Guide to Integrating with Tivoli Applications*, SC31-8744-00
- *Tivoli Information Management for z/OS Integration Facility Guide*, SC31-8745-00
- Tivoli Information Management for z/OS Licensed Program Specification, GC31-8746-00
- Tivoli Information Management for z/OS Master Index, Glossary, and Bibliography, SC31-8747-00
- *Tivoli Information Management for z/OS Messages and Codes*, GC31-8748-00
- *Tivoli Information Management for z/OS Operation and Maintenance Reference*, SC31-8749-00
- *Tivoli Information Management for z/OS Panel Modification Facility Guide*, SC31-8750-00
- *Tivoli Information Management for z/OS Planning and Installation Guide and Reference*, GC31-8751-00
What This Document Contains

This book contains the following sections:

Part 1 – Tivoli Decision Support

- “Tivoli Decision Support” on page 3 provides an overview of Tivoli Decision Support so that you can understand how it can be used to gain insight into your Tivoli Information Management for z/OS data for better decision making.

Part 2 – ODBC Driver

- “Introduction” on page 15 describes how to use the ODBC driver and introduces you to ODBC concepts.
- “Installing the ODBC Driver” on page 25 describes the requirements for the ODBC driver and installation.
- “Creating Data Model Records” on page 33 describes how to use data attribute and data view records to define your tables and columns.
- “Using the ODBC Driver” on page 37 shows how to connect to the Tivoli Information Management for z/OS host from your workstation application.
- “Changing Driver Options or Attributes” on page 43 explains the driver default settings and describes how to change them if necessary.
- “Messages” on page 47 lists the messages associated with the ODBC driver.
- “SQL Command Syntax and Parameters” on page 107 lists the SQL statements supported by the ODBC driver for data retrieval.

Part 3 – Report Format Facility

- “Introducing the Report Format Facility” on page 131 provides an overview of standard reports.
- “Modifying and Creating RFTs” on page 135 and “RFT Statements” on page 147 describe how to create or modify a report (assuming you are already familiar with the standard reports). See “Standard Reports” on page 261 for a list of the member names of
RFTs for the standard reports. Then see “Definition Statements” on page 153 and “Command Statements” on page 167 for information on definition and command statements.

■ “Report Exit Routines” on page 231 describes how to use report exit routines with reports. It shows you how to write routines that retrieve additional data, add output lines to reports, perform calculations, and analyze the data.

■ “Defining Output Destinations for Reports” on page 251 provides information on defining output destinations.

■ “Alternative Methods of Running Reports” on page 255 describes how to run reports using a stored response chain (SRC), or in batch mode.

The appendixes present additional information you can use to control the running of reports and report output destinations and information about customizing the configuration-diagram data extraction RFT.

This product is enabled for DBCS support. As a result, this book uses the following terms:

■ DBCS (double-byte character set)
■ SBCS (single-byte character set)
■ Mixed data

The term mixed data refers to data strings that can contain only DBCS data, only SBCS data, or any combination of DBCS and SBCS data. SBCS data is the same as EBCDIC data. The term mixed case data refers to data strings that can contain uppercase, lowercase, or a combination of uppercase and lowercase SBCS data.

**Typeface Conventions**

This guide uses several typeface conventions for special terms and actions. These conventions have the following meaning:

**Bold** Entries that you must use literally, choices, or options that you select appear in bold. The names of titles or screen objects in graphical windows also appear in bold.

**Italics** Variables and values that you must provide appear in italics. New terms also appear in italics.

**Monospace** Code examples, output, and messages are in monospace font.

The host panels as presented in this book are not meant to be exact replicas of the way a panel might appear on the screen. The information on the panels is correct, but the spacing is not always exact.

Similarly, the examples as presented in this book in the description of the Report Format Facility are not meant to be exact representations of reports and RFTs. At times the spacing has been modified to fit the information on the page.

The reports, RFTs, and panels in this book generally show use of a 4-digit year date format. The actual date format displayed in your reports will depend on the external date format specified by your Tivoli Information Management for z/OS program administrator (or your user profile date format selection) and how you code your RFTs.
The Use of Panel Style in This Document

With Tivoli Information Management for z/OS, you may see changes in the way Tivoli Information Management for z/OS panels are displayed. Two panel styles are available: the standard panel style and the enhanced panel style. The style of panel does not affect the data that must be entered from it.

Except where noted, this book uses the Tivoli Information Management for z/OS standard panel style when showing you how a panel looks.

For more information about the enhanced panel style, refer to the Tivoli Information Management for z/OS Program Administration Guide and Reference, the Tivoli Information Management for z/OS Planning and Installation Guide and Reference, and the Tivoli Information Management for z/OS User's Guide.

Contacting Customer Support

For support inside the United States, for this or any other Tivoli product, contact Tivoli Customer Support in one of the following ways:

- Send e-mail to support@tivoli.com
- Call 1-800-TIVOLI8
- Navigate our Web site at http://www.support.tivoli.com


When you contact Tivoli Customer Support, be prepared to provide identification information for your company so that support personnel can assist you more readily.

The latest downloads and fixes can be obtained at http://www.tivoli.com/infoman.
I — Tivoli Decision Support

Chapter 1. Tivoli Decision Support ........................................... 3
  What Tivoli Decision Support Does........................................... 3
  Viewing Data ..................................................................... 6
  Information Management Guide.......................................... 9
  Requirements. ................................................................... 11
Tivoli Decision Support

This chapter provides an overview of how Tivoli Decision Support integrates with Tivoli Information Management for z/OS. A separately orderable product called Tivoli Decision Support for Information Management (5697-IMG) is required to use Tivoli Decision Support with Tivoli Information Management for z/OS.

**Note:** This chapter does not show you how to install or use Tivoli Decision Support. For that information, refer to the following publications:

- *Tivoli Decision Support Installation Guide*
- *Tivoli Decision Support Administrator Guide*
- *Tivoli Decision Support User’s Guide*
- *Tivoli Decision Support: Using Decision Support Guides*

For instructions on how to install Tivoli Decision Support for Information Management, refer to the following softcopy publications provided with Tivoli Decision Support for Information Management:

- *Release Notes*
- *Tivoli Decision Support: Using the Information Management Guide*

Customization and usage instructions for Tivoli Decision Support for Information Management are also provided in *Tivoli Decision Support: Using the Information Management Guide*.

**What Tivoli Decision Support Does**

Tivoli Decision Support is a workstation-based solution that helps you to gain insight into the data that is collected daily in an enterprise. It helps you translate pieces of data into meaningful knowledge about your operations, your end users, and your relationship with them. Tivoli Decision Support enables you to quickly find and use the data in your enterprise’s databases in a number of key business areas, including:

- **Support center**
- **Problem management**
- **Change management**

For example, your support center database contains information captured as analysts resolve customer requests and manage the support center’s daily functions. This data may include information about product defects, resolution times and rates, or trends in customer requests or interactions with the support center.
Tivoli Decision Support enables businesses to extract information from customer data and automatically builds content that is ready to be used by interested parties. As a decision support framework, it goes beyond traditional reporting tools and static reports by providing value-added functions that make data analysis more effective.

Tivoli Decision Support also handles the delivery and access of the data by facilitating knowledge discovery and user access to the information. Data can be shared with other people in the enterprise through various delivery mechanisms. Data can be printed, extracted to files, or sent to a central repository on a company’s intranet, from which other users can gain access to it.

As shown in Figure 1 on page 5, you can use Tivoli Decision Support to connect to a Tivoli Information Management for z/OS database through the Tivoli Information Management for z/OS ODBC driver, which runs on a Windows NT workstation.

When you issue a request for Tivoli Information Management for z/OS information in Tivoli Decision Support, Tivoli Decision Support retrieves the data to present it in its Discovery Interface in either one of the following ways:

- Directly from the Tivoli Information Management for z/OS database for presentation in a detailed view through Crystal Reports
  
  Crystal Reports is a software package that generates text-based views from standard database queries. Decision Support displays many views in Crystal Reports format.

- From a Cognos PowerPlay cube for presentation in a multidimensional view
  
  Cognos PowerPlay is a third-party analysis and reporting package which must be installed with Tivoli Decision Support. Tivoli Decision Support displays many views in PowerPlay format. A Cognos PowerPlay cube is a file of data that is read from a database (in this case, the Tivoli Information Management for z/OS database). Cubes are built by the Tivoli Decision Support administrator using Cognos PowerPlay.
Tivoli Decision Support integrates with Cognos PowerPlay and Crystal Reports reporting and analysis programs and can present data in several formats. Through the Tivoli Decision Support Discovery Interface, you can view data in a graphical format such as a bar chart, line chart, or pie chart, or as text in a detail report format. The default format depends on the type of view you are using, and you can change the type of view for any of the multidimensional views in the Discovery Interface.

The data presented in a detailed or multidimensional view is retrieved from the Tivoli Information Management for z/OS database through the Tivoli Information Management for z/OS ODBC driver. The ODBC driver communicates with the host through the Tivoli Information Management for z/OS High-Level Application Program Interface (HLAPI) for Windows NT (called the HLAPI/NT) and the Tivoli Information Management for z/OS Multiclient Remote Environment Server (MRES).

Figure 1. Component Overview of Tivoli Information Management for z/OS and Tivoli Decision Support
The MRES provides API functions to the HLAPI/NT client and handles all communication between the client and the Tivoli Information Management for z/OS databases.

Data retrieval is accomplished through data model records which support the underlying cube and report queries. Data model records are stored in a Tivoli Information Management for z/OS database and include data view records and data attribute records. A data view record corresponds to an SQL table for ODBC purposes. Data attribute records correspond to columns in a table and represent the data that is accessible through the table. One data view record exists for each cube.

For more information about the Tivoli Information Management for z/OS ODBC driver, see "Introduction" on page 13. For more information about the Tivoli Information Management for z/OS HLAPI/NT, refer to the Tivoli Information Management for z/OS Client Installation and User’s Guide.

Viewing Data

With Tivoli Decision Support, you can view data from many different angles. You can:

- Analyze data from different perspectives
- Spot trends
- Troubleshoot
- Evaluate resource allocation
- Do projections and forecasts
- Perform other management tasks


Through the graphical Discovery Interface, you can manipulate the view of the data different ways:

- You can view data through a Discovery Guide.

  Discovery Guides are templates that assist users in selecting which questions to ask and locating the data that answers these questions. A Discovery Guide groups your business intelligence data into specialized categories. Various Discovery Guides are available for Tivoli Decision Support, including one for Tivoli Information Management for z/OS data. The Discovery Guide for Tivoli Information Management for z/OS is called Information Management.

  As illustrated in Figure 2 on page 7, the Information Management Guide is organized into three categories: activity management, change management, and problem management. The activity management category enables you to see what activities or tasks are associated with implementing change requests. The change management category enables you to determine how well you are controlling and managing your change process. The problem management category enables you to determine how well you are managing your support center. Data displayed in the activity and change management categories comes from change management records in Tivoli Information Management for z/OS. Problem management data comes from problem management records in Tivoli Information Management for z/OS.
The views available with the Information Management Guide can be used through the topic map.

You can navigate through the data through a topic map.

The topic map lets you see the data organized by categories, topics, and views. The Information Management Guide presents three categories in a topic map: activity management, change management, and problem management.

The scope of what you can see in a topic map is defined by Discovery Guides and the roles that may exist in your organization. A role is a description of a user’s position in the service and support area of the business. Tivoli Decision Support uses roles to help you select the best description of your responsibilities in an organization. By specifying one or more roles, you establish the scope of what you can view. The more roles you select, the more views will be listed in a topic map. Roles help you to search for business information quickly, without having to sift through views that may be of little interest to you.

You can view Tivoli Information Management for z/OS data in Tivoli Decision Support in one of two ways:

- Detailed view
  These views are preformatted to appear in a Crystal Reports template. Detailed views are text-based and two-dimensional, as shown in Figure 3 on page 8.
Multidimensional view

These views are preformatted to appear in one of several PowerPlay templates. Such views can appear in the form of a bar chart, line chart, pie chart, or a crosstab, but other formats are available.

When a multidimensional view is displayed, Tivoli Decision Support gives you access to a variety of PowerPlay’s analysis methods which enable you to analyze different aspects of the data. Using these features, you can search for additional data (called “drilling”), select different portions of data for viewing (filtering), and rearrange the data for comparative purposes.

For example, you can use a multidimensional view to see a pie chart of the types of Tivoli Information Management for z/OS open problem records reported in the past month by reporter (see Figure 4 on page 9). Then, you can:

- Double-click on a particular reporter (such as a specific data center) to see a pie chart of those problems, and “drag down” the By Severity dimension to see how many severity 1 problems are open in that location.
- Double-click on severity 1 problems and drag down the Problem Type dimension to find out what types of severity 1 problems are occurring. The breakout may show that 80 percent of the problems are software related.
- Find out who is assigned to the severity 1 problems by dragging down the Assignee dimension, which presents another pie chart showing which help desk technicians are working on the problems.

Through simple click and drag methods, you quickly select data of interest to you and drill down to details that can facilitate your decision making.

Because the Discovery Interface preformats all views before displaying them, you do not need to create any templates or format any data. Tivoli Decision Support handles those tasks for you. You can create your own templates if necessary.

For more information about the Discovery Interface, refer to the Tivoli Decision Support User’s Guide.

Information Management Guide

The Information Management Guide consists of five cubes that generate a number of multidimensional views which are available in the Discovery Interface. Each cube relies on a set of queries to populate it with data from the Tivoli Information Management for z/OS database. The problem management category consists of the following cubes:

Info Man Problems

The Info Man Problems cube contains information on problem records, regardless of problem status, that you can view in various dimensions to determine trends and statistics. For example, you can view the number of closed or open problems, duplicates, and average severity. You can drill down to various details, such as the reporter and assignee information, and symptom data.
**Info Man Open Problems**

The Info Man Open Problems cube contains information on problem records where the status is not CLOSED. You can view information on the open aging date range (problems open for less than 30 days, open for 30 to 90 days, and so on) and last altered date range.

**Info Man Closed Problems**

The Info Man Closed Problems cube contains information on closed problem records. You can view information such as the problem cause and closure time range, and find out what change request caused the problem.

The change management category consists of the following cubes:

**Info Man Changes**

The Info Man Changes cube contains information on change records, regardless of their change status, that you can view in various dimensions to determine trends and statistics. For example, you can view the changes that are completed, overdue, and upcoming, and the estimated and actual work effort associated with the changes. You can drill down to various details, such as the requester and assignee information, change type, priority, and reason.

**Info Man Activities**

The Info Man Activities cube contains information on activities related to changes, regardless of change status. You can view information on the name, type, and description of the activity, and activity schedule.

Measurements are also provided to help you understand how well your data center is performing problem management and change management activities. For example, for problem management, the following measurements are included:

- Down time associated with problem
- Elapsed business time
- Time spent working on the problem

Change management measurements are provided to help you assess your data center’s performance. For example, the following measurements are included:

- The duration of the change
- Number of days the change is overdue

For more information about the Information Management Guide, such as a detailed description of the multidimensional views and Crystal reports provided for problem management and change management, and a list of all the data elements that compose the cubes, refer to softcopy publication *Tivoli Decision Support: Using the Information Management Guide*. This softcopy publication is available as a portable document file (PDF) with the Information Management Guide.
Requirements

To use Tivoli Decision Support with Tivoli Information Management for z/OS data, you need the following:

- A workstation running under Microsoft Windows NT with the following installed:
  - Tivoli Decision Support 2.0 or later (installed in stand-alone or network mode)
    Refer to the *Tivoli Decision Support User’s Guide* for installation instructions.
  - Tivoli Decision Support for Information Management
    Refer to *Tivoli Decision Support: Using the Information Management Guide* and the *Release Notes*.
  - Tivoli Information Management for z/OS ODBC driver
    See “Installing the ODBC Driver” on page 25 for the prerequisites and installation instructions.
  - Tivoli Information Management for z/OS HLAPI for Windows® NT
    See the *Tivoli Information Management for z/OS Client Installation and User’s Guide* for the prerequisites and installation instructions.

- A multiclient remote environment server (MRES) with TCP/IP or APPC.
  See the *Tivoli Information Management for z/OS Client Installation and User’s Guide* for more information about the MRES.

- Data model records installed on Tivoli Information Management for z/OS to support the Information Management Guide.
  “Data Model Records — Tivoli Decision Support” on page 358 lists the data model records provided to support the Information Management Guide.

**Note:** With this release of Tivoli Information Management for z/OS, data model records needed to support the Information Management Guide are available on the Tivoli Information Management for z/OS host tapes. For general instructions on loading data model records shipped with Tivoli Information Management for z/OS, refer to the *Tivoli Information Management for z/OS Planning and Installation Guide and Reference*. You can use those instructions to load data model records for the Information Management Guide.
# II — Open Database Connectivity (ODBC) Driver

## Chapter 2. Introduction

- ODBC Concepts .............................................................. 16
- Tivoli Information Management for z/OS ODBC Driver ............... 18
- ODBC Conformance .......................................................... 19
  - API Conformance Levels .................................................. 19
  - SQL Conformance Levels ................................................. 20
- ODBC Driver Restrictions ................................................... 23
  - Freeform Text Processing .................................................. 23
  - Multiple Response Field Processing .................................. 23

## Chapter 3. Installing the ODBC Driver

- ODBC Driver Requirements ................................................ 25
- Verifying Your HLAPI/NT Setup .......................................... 26
- Installing the ODBC Driver .................................................. 27
  - Serviceability ............................................................... 30
  - Files Installed ............................................................ 30
  - Uninstalling the ODBC Driver ........................................... 32
  - Ensuring Setup of Data Model Records ................................ 32

## Chapter 4. Creating Data Model Records

- Data Attribute and Data View Records .................................. 33
  - Create Data Attribute Records for Each Column of Data ....... 33
  - Build a Data View Record ............................................... 35

## Chapter 5. Using the ODBC Driver

- Connecting to the Data Source .......................................... 37
  - Using Crystal Reports .................................................... 37
  - Using Lotus Approach .................................................... 38
  - Using Lotus 1-2-3 ........................................................ 38
  - Using Microsoft Access .................................................. 39
  - Entering Data Source Information .................................... 39
  - Defining Queries .......................................................... 40

## Chapter 6. Changing Driver Options or Attributes

- Specifying Advanced Options ......................................... 43

## Chapter 7. Messages

- Message Formats ........................................................... 47
  - ODBC Driver Manager Tracing ......................................... 48
  - ODBC Driver Tracing ..................................................... 48
  - Messages ................................................................. 49
  - Return and Reason Codes Issued by the ODBC Driver by Function . 52
    - SQLAllocConnect ....................................................... 53
    - SQLAllocEnv ........................................................... 54
    - SQLAllocStmt ......................................................... 55
    - SQLBindCol ............................................................. 56
Chapter 8. SQL Command Syntax and Parameters .......................... 107

ODBC Functions Supported .......................................................... 107
Elements Used in SQL Statements ......................................... 118
SQLGetInfo Information Types and Values Returned ............ 119
You have been asked to pull Tivoli Information Management for z/OS data and generate a
customized report for the account representative responsible for handling your company's
customer accounts.

A new product ordering and distribution system was installed recently by your company’s
I/S department, and everyone is anxiously awaiting the outcome. You have heard that a
customer in Pittsburgh had problems placing orders, and that another customer in
Indianapolis did not receive their product delivery on time.

Your account representative has severity 1 situations open in three accounts, and you have
been told to meet with her daily for the next month to review the status of all open
problems. You also need to merge the problem record data with other text and spreadsheets
on your Windows NT workstation to be able to present informative reports to your
managers. Because your managers prefer illustrations, you also want to provide data in
graphical format using the workstation packages provided in your office.

With the Open Database Connectivity (ODBC) driver for Tivoli Information Management
for z/OS, you can exploit the power of your workstation applications through the Tivoli
Information Management for z/OS High-Level Application Program Interface (HLAPI)
Client for Windows NT (HLAPI/NT).

The Tivoli Information Management for z/OS ODBC driver enables you to do the following:

- Use the Tivoli Information Management for z/OS host database as a data source for an
  ODBC-enabled application that can run on a Windows NT workstation.
- Use an ODBC-enabled workstation application (such as a database, spreadsheet, or text
  processing application) to click on fields you want to search on. You can set up a filter
  or query, if needed, to limit the search, and Tivoli Information Management for z/OS
  will execute the search and retrieve the data.
- Set up queries or generate reports in formats that are useful to you.
- Use industry-standard Structured Query Language (SQL) skills to generate customized
  reports, without having to depend on your Tivoli Information Management for z/OS
  administrator to write special report format tables (RFTs) for you.

The information provided in this part of the document is intended for those Tivoli
Information Management for z/OS users seeking alternative ways to retrieve or search data,
or to generate reports from data stored in the Tivoli Information Management for z/OS
databases.
It is assumed that you are familiar with Tivoli Information Management for z/OS, including how to search for Tivoli Information Management for z/OS data. If you are using a workstation database application, it is helpful if you are familiar with SQL and ODBC. Additionally, if you are a business user who needs to prepare information in various formats, it is helpful if you know how to format data within the workstation application you are using.

Although the following publications are not required, they are suggested as additional sources of information:

- **Tivoli Information Management for z/OS User's Guide** — describes how to search Tivoli Information Management for z/OS host data
- **Tivoli Information Management for z/OS Client Installation and User's Guide** — provides more information about the Tivoli Information Management for z/OS HLAPI for Windows NT client feature, the Tivoli Information Management for z/OS servers, and API return and reason codes
- **Tivoli Information Management for z/OS Application Program Interface Guide** — describes API transactions
- **Tivoli Information Management for z/OS Messages and Codes** — lists Tivoli Information Management for z/OS host and HLAPI client messages

## ODBC Concepts

If you are unfamiliar with the concepts of ODBC, you may want to review this section to better understand what ODBC is and how its various components relate to one another.

ODBC is an architected database access interface that enables applications to access data using SQL as a standard language. Applications that are enabled for ODBC can access different database management systems through a consistent set of APIs.

The ODBC approach involves a database driver program, separate from the application, to extract database information. Database drivers are provided by various database vendors or third parties, and are supplied as dynamic link libraries that your ODBC-enabled applications can invoke.

The ODBC interface defines a library of function calls that enable an application to connect to a database management system, execute SQL statements, and retrieve results. The ODBC interface enables you to use standard SQL syntax to search for data and provides a common way to connect and log on to a database management system. It also uses a standard set of error codes (SQLSTATEs) and enables data types to be represented in a standard way.

The components of an ODBC application are described in the following list. Figure 5 on page 18 shows the relationship of these components in a Tivoli Information Management for z/OS environment.

### Application

The application requests a connection or a session with a data source. It sends SQL requests from the workstation to the data source, retrieves the resulting data, and processes errors that are encountered. After the SQL requests are submitted, the application reports the results back to the user if needed.

The application is also responsible for defining the storage areas and format of data obtained through SQL requests. For transaction control, the application also requests commit or rollback operations. (Note that the Tivoli
Information Management for z/OS ODBC driver does not support transactions, so it is always in autocommit mode.)

The connection to the data source is terminated by the application.

Examples of applications are Lotus® Approach®, Lotus 1-2-3®, Microsoft Access, and Tivoli Decision Support.

**Driver Manager**

The driver manager uses the Windows Registry (or the ODBC.INI file in environments other than Windows NT) to map a data source name to a specific driver dynamic link library (DLL). It also provides parameter validation and sequence validation for ODBC calls.

In the Tivoli Information Management for z/OS environment, the ODBC driver manager used is the Microsoft ODBC Driver Manager Version 3.51 for Windows NT. This driver manager is provided with the Tivoli Information Management for z/OS ODBC driver code.

**Driver**

The driver is in charge of receiving control from the driver manager, establishing a connection to the data source, and submitting requests to that source. The driver returns results to the application. It also:

- Returns information about the driver
- Translates errors into standard error codes (the driver translates errors it detects and errors returned by the data source)
- Returns error codes to the application

There are two types of drivers: single-tier and multiple-tier. Single-tier drivers process both ODBC calls and SQL statements. They also access the database directly. Multiple-tier drivers process ODBC calls and pass SQL statements to a data source. The Tivoli Information Management for z/OS ODBC driver, using HLAPI/NT, is a single-tier, client/server implementation.

**Data Source**

The data source provides the overall features and functions provided by a database management system. The data source represents a specific combination of the database management system, remote operating system, and the networking necessary for access. In our case, the database management system is the Tivoli Information Management for z/OS host with networking provided by the HLAPI/NT and Tivoli Information Management for z/OS servers.

Figure 5 on page 18 shows how the ODBC driver interacts with a workstation application and Tivoli Information Management for z/OS.
Tivoli Information Management for z/OS ODBC Driver

The Tivoli Information Management for z/OS ODBC driver enables an ODBC-enabled application to access data in the Tivoli Information Management for z/OS database using standardized SQL. The driver is invoked from an ODBC-enabled application on your workstation through the ODBC Driver Manager to:

- Process ODBC function calls.
- Transform and submit the requests to Tivoli Information Management for z/OS through Tivoli Information Management for z/OS’s HLAPI/NT client feature.
- Return the results to your application.

In a relational database, there are tables, rows, and columns that define the data in the database. These elements also apply to ODBC and are how ODBC-enabled applications reference data in any database management system. However, tables, rows, and columns are not inherent to the Tivoli Information Management for z/OS database design.

Data model records correlate the relational database elements and define this relationship for ODBC. A data view record corresponds to a table. The data view record name is a table name in ODBC functions and SELECT query statements. A data attribute record corresponds to a column in a table. The data attribute records listed in a data view record denote the columns that are accessible through that table. A row is the same as a record in a Tivoli Information Management for z/OS search results list. More information about data model records is available in “Creating Data Model Records” on page 33.

The Tivoli Information Management for z/OS ODBC driver submits API transactions using the Tivoli Information Management for z/OS HLAPI/NT IDBTransactionSubmit function. Only synchronous processing is supported; your application waits upon completion of one function before commencing any other function in the application.
The SQL statement used to initiate a search of the host database is SELECT. The SELECT statement causes the driver to issue a Tivoli Information Management for z/OS API search (HL11 record inquiry) transaction. It also issues retrieve (HL06 retrieve record) transactions to get the requested data from the Tivoli Information Management for z/OS database. The SELECT statement is coded in the ODBC SQLExecDirect or SQLPrepare statements used by the application. The database is accessed as read-only and you cannot create or update data in the Tivoli Information Management for z/OS database through use of this driver.

**ODBC Conformance**

The ODBC standard enables drivers to provide different levels of function. Conformance levels are used to define the function provided. These conformance levels apply to the API interface to ODBC and the SQL statements supported by the driver.

**API Conformance Levels**

API conformance levels are defined as Core, Level 1, and Level 2.

**Core API**

- Allocates and frees environment, connection, and statement handles.
- Connects to data source, uses multiple statements on connection.
- Prepares and executes SQL statements, executes SQL statements immediately.
- Assigns storage for parameters in an SQL statement and result columns.
- Retrieves data from a results set and about a results set.
- Commits or rolls back transactions (not supported by the Tivoli Information Management for z/OS ODBC driver).
- Retrieves error information.

**Level 1 API**

- Provides core API functionality.
- Connects to data sources with driver-specific dialog boxes.
- Sets and inquires values of statement and connection options.
- Sends all or part of a parameter value (useful for long data).
- Retrieves catalog information (columns, special columns, and tables).
- Retrieves information about driver and data source capabilities.

**Level 2 API**

- Provides Core and Level 1 functionality.
- Browses available connections and lists available data sources.
- Sends arrays of parameter values.*
- Retrieves arrays of result column values.
- Uses a scrollable cursor.*
- Retrieves the native form of an SQL statement.
- Retrieves catalog information (privileges, keys, and procedures).*
- Calls a translation API.*
The Tivoli Information Management for z/OS ODBC driver supports Core, Level 1, and some Level 2 ODBC Version 3.51 functions. Level 2 functions noted with an asterisk (*) in the preceding list are not supported by the ODBC driver.

**SQL Conformance Levels**

An ODBC driver can conform to SQL on various levels. A driver can support minimum SQL grammar, core SQL grammar, or extended SQL grammar. The Tivoli Information Management for z/OS ODBC driver supports some minimum SQL grammar as described in the following list. For details on the SQL statements supported, see "SQL Command Syntax and Parameters" on page 107.

- Data manipulation language: simple SELECT statements. For example:

```sql
SELECT * FROM PROBVIEW
SELECT REPORTER, DESCRIPTION FROM PROBVIEW WHERE STATUS='OPEN'
SELECT * FROM PROBVIEW WHERE DATE_ENTERED BETWEEN '1998/05/01' AND '1998/06/01' AND STATUS NOT 'CLOSED'
```

When entering searches for records containing dates, enter the dates in the Tivoli Information Management for z/OS internal date format. Dates must be specified in one of the following formats:

- YYYY/MM/DD
- YYYY-MM-DD
- YYYY/DD/DD

YYYY/DD/DD is the Tivoli Information Management for z/OS internal date format. If you specify dates in YYYY-MM-DD format, they will be converted to the YYYY/MM/DD internal date format.

If you do not know how internal dates are stored in your Tivoli Information Management for z/OS database, issue the GLOSSARY command while logged on to Tivoli Information Management for z/OS, and enter the DATE/ keyword to view the internal date format. Note that the internal date format can be different from the external date format—the date fields you normally see on the panels in Tivoli Information Management for z/OS.

- Expressions: simple comparisons are supported (=, >=, <=, <>). NOT < and NOT > are supported. BETWEEN is the recommended and supported range operator.

The following are not supported: greater than (>), less than (<), NOT >= , NOT <= , and NOT BETWEEN. Addition and subtraction are not supported since Tivoli Information Management for z/OS data is character data.

- Data types: CHAR, VARCHAR, and LONGVARCHAR only. Data types not supported are listed as follows:

<table>
<thead>
<tr>
<th>Type</th>
<th>Equivalent Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIGINT</td>
<td>LONGVARBINARY</td>
</tr>
<tr>
<td>BINARY</td>
<td>NUMERIC</td>
</tr>
<tr>
<td>BIT</td>
<td>REAL</td>
</tr>
<tr>
<td>DATE</td>
<td>SMALLINT</td>
</tr>
<tr>
<td>DECIMAL</td>
<td>TIME</td>
</tr>
<tr>
<td>DOUBLE</td>
<td>TIMESTAMP</td>
</tr>
<tr>
<td>FLOAT</td>
<td>TINYINT</td>
</tr>
<tr>
<td>INTEGER</td>
<td>VARBINARY</td>
</tr>
</tbody>
</table>
The following SQL functions are not supported by the Tivoli Information Management for 
z/OS ODBC driver. If these functions are used, an error message is returned unless 
otherwise noted.

- Data definition language: CREATE TABLE and DROP TABLE are not supported.
- Data manipulation language: INSERT, UPDATE, and DELETE statements are not 
supported.
- Table joins (no multiple tables on a SELECT statement).
  Although table joins are not supported, you can open more than one table at a time. 
Joining tables is not necessary, because you can create a data view record to combine all 
the fields you need for a given record type. For example, you can set up a data view 
record to contain all the fields for problem records, and another data view record to 
contain all the fields for change records.
- Removal of duplicate rows (no SELECT DISTINCT); no error message is returned.
  If your workstation application provides the function, you can remove duplicate rows 
from within your application after the data is retrieved into your application.
- Sorting (ORDER BY); no error message is returned.
  Data is returned in the order in which it was retrieved. You can do additional sorting 
from within your workstation application.
- Full WHERE clause support.
  The following WHERE clause restrictions exist:

  - Parenthetical searching to group elements is not supported. If you enter a search 
    argument like `SELECT * FROM table WHERE STATUS='OPEN' OR (PRIORITY='1' 
    AND REPORTER='JIM')`, the parentheses are stripped, and the search criteria is 
    interpreted as `STATUS='OPEN' OR PRIORITY='1' AND REPORTER='JIM'`, which 
    could cause Tivoli Information Management for z/OS to return unexpected results.
  - Arithmetic support, including date or time math, is not supported.
  - The percent character (%) is a wildcard used to match zero (0) or more of any 
    character. The Tivoli Information Management for z/OS ODBC driver only supports 
    it as a wildcard in the last position of a search argument in a LIKE clause or in a 
catalog function (such as SQLColumns or SQLTables) where it is treated the same as 
Tivoli Information Management for z/OS’s truncation character, the period (.). If the 
% wildcard is not in the last position, it is treated as data.
  - The underscore character (_) is supported as a wildcard in any position of a search 
    argument in a LIKE clause or catalog function. It is treated the same as Tivoli 
    Information Management for z/OS’s ignore character, the asterisk (*).
    Additionally, if you are using a catalog function, and want to have a percent 
    character treated as data in the last position (rather than as a wildcard), you can 
    precede the ‘%’ with the Tivoli Information Management for z/OS ODBC driver’s 
SEARCH_PATTERN_ESCAPE_CHARACTER, the forward slash (/). Note that this 
only applies to catalog functions.

Notes:

1. The data you supply in a WHERE clause is case-sensitive for data retrieval. 
   Typically, data in the Tivoli Information Management for z/OS database is cognized 
in uppercase for search purposes. Enter the data in uppercase to ensure proper 
retrieval of data (for example, specify `STATUS='OPEN'` instead of `STATUS='Open'`).
2. In Tivoli Information Management for z/OS, some fields may use the same p-word. For example, Assignee phone, Reporter phone, and Contact phone all use the PH/p-word. If you include such a field in your SQL statement, be aware that you may receive out-of-context matches. For example:

Record 1 contains: Reporter phone 555-1111 and Assignee phone=555-9999
Record 2 contains: Reporter phone 555-9999 and Assignee phone=555-1111

Record 1 and Record 2 appear in your search results when you search for either:

- `REPORTER_PHONE='555-9999'`
- `REPORTER_PHONE='555-1111'`

Because the freeform search performed has no way of knowing which phone number you are interested in (other than the matching value), you may receive more record matches than you expect.

3. The ODBC driver supports the Tivoli Information Management for z/OS wildcard characters, which are the period (.) and asterisk (*).

- When entering search arguments to retrieve records containing string data, specify each word as a separate search argument. For example, if the description abstract field for a problem record contained the following text:
  
The LAN server is down in Building 600

Enter your search argument as shown:

```
SELECT * FROM table WHERE ABSTRACT='LAN' AND ABSTRACT='SERVER' AND ABSTRACT='DOWN'
```

The order of the data in your search argument does not matter. You could have entered the following to retrieve the same data:

```
SELECT * FROM table WHERE ABSTRACT='SERVER' AND ABSTRACT='DOWN' AND ABSTRACT='LAN'
```

In this example, all records containing data matching those three words in the abstract field, which can include records for LAN server outages in other buildings, are retrieved. If you wanted to limit the search to records containing abstract fields data for a specific building, you can include that data in your search argument (`AND ABSTRACT='600'`).

**Note:** If your search argument contains a string field which is cognized as an unparsed string, you must put an asterisk instead of a space in the search string for that field. For example, in problem records, the Reported by field is cognized as an unparsed string. To search for records reported by 'ANN LIVINGSTON', enter the WHERE clause of the SELECT statement as follows:

```
SELECT RECORD_ID, REQUESTED_BY FROM table
WHERE REQUESTED_BY='ANN*LIVINGSTON'
```

For more information about fields that are cognized as unparsed strings, refer to the [Tivoli Information Management for z/OS User’s Guide](#).

More information about the SQL syntax supported with the ODBC driver is provided in [“SQL Command Syntax and Parameters” on page 107](#).
**ODBC Driver Restrictions**

The following additional restrictions exist with use of the ODBC driver:

- You can only read data from the Tivoli Information Management for z/OS database; you cannot update it.
- The retrieval of Tivoli Information Management for z/OS history data is not supported. History data fields are identified by an `<H>` when you are in display mode in Tivoli Information Management for z/OS.
- The maximum length of a list data field is the lesser of 1024 bytes or 100 list entries.
- If you are looking for a table that is not in the list shown to you, check with your system administrator. The **Allow ODBC access?** field that defines the table must be set to YES in the data view record defined on the Tivoli Information Management for z/OS host.

**Freeform Text Processing**

Some panels in Tivoli Information Management for z/OS enable you to enter lines of freeform text to provide additional notes or information about a record. For example, you can enter notes about the description of a problem or details about its status or resolution. When you use the ODBC driver to retrieve freeform text records, the audit trail data normally associated with freeform text (such as user ID and date information) is not returned to your workstation. For example, if the freeform text data in Tivoli Information Management for z/OS was the following:

```
The problem occurred due to an operating USERID DATE
system change which was implemented over the USERID DATE
weekend.
```

The data returned to you through the ODBC driver is the following:

```
The problem occurred due to an operating
 system change which was implemented over the
 weekend.
```

**Multiple Response Field Processing**

In Tivoli Information Management for z/OS, panels can be customized to accept multiple responses in a data entry field. For example, a name field can be defined as a multiple response field to accept the entries SMITH and JANE in a single field. When such fields are defined, the Tivoli Information Management for z/OS application program interface separates the words with a special separator character, such as a comma, and pads each word to its length with blanks:

```
SMITH ,JANE
```

However, the ODBC driver returns data associated with multiple response fields with a single space between words:

```
SMITH JANE
```

If you are using an MRES with pre-started sessions, your MRES startup parameters should include the following parameter:

```
MULTIPLE_RESPONSE_FORMAT=PHRASE
```

If this parameter is omitted, the data returned from multiple response fields will include padded blanks and the separator character defined for the field.
Installing the ODBC Driver

This section describes what you need to install and use the Tivoli Information Management for z/OS ODBC driver.

ODBC Driver Requirements

To use the Tivoli Information Management for z/OS ODBC driver, you must have the following:

**Host**

- A valid MVS™ TSO user ID for the host system where Tivoli Information Management for z/OS is installed, including a privilege class assignment.

- A Tivoli Information Management for z/OS server installed on your Tivoli Information Management for z/OS host system (a remote environment server or multiclient remote environment server (MRES) using TCP/IP or APPC).

Because the ODBC driver uses program interface data tables (PIDTs) provided by Tivoli Information Management for z/OS, your Tivoli Information Management for z/OS system installer should ensure that either the session-parameters member you use points to the supplied SBLMFMT data set, or that a DD card is used in the JCL for your server (typically RFTDD) for the SBLMFMT data set.

- Basic knowledge of searching Tivoli Information Management for z/OS data.

- Access to a Tivoli Information Management for z/OS program administrator who can define data model records (data attribute and data view records) which are required for use with the Tivoli Information Management for z/OS ODBC driver. Once the administrator sets up the initial data model record to support ODBC access, you can run reports or queries.

**Note:** The data model records provided with Tivoli Information Management for z/OS to support Tivoli Decision Support for Information Management can be used with the ODBC driver even if you are not using Tivoli Decision Support. For a list of these records, see “Data Model Records — Tivoli Decision Support” on page 358. Instructions on how to load these data model records are provided in the Tivoli Information Management for z/OS Planning and Installation Guide and Reference.

**Workstation**

- A workstation running Microsoft Windows NT 4.0 or later.
The Tivoli Information Management for z/OS HLAPI for Windows NT client feature (with its prerequisites) installed on your workstation. The client feature includes support for APPC and TCP/IP connectivity to a Tivoli Information Management for z/OS server.

The Tivoli Information Management for z/OS ODBC driver code.

Installation of the ODBC driver code requires approximately 6 MB of available space on your disk. (This includes space for installing some Windows NT system files or ODBC files that may already be installed.) A README file is provided with the driver with any additional instructions. Upgrades or patches that can be downloaded from a Tivoli Web site may be available for the ODBC driver. For more information, visit Web site http://www.tivoli.com/infoman.

An ODBC-enabled application on your workstation, such as one of the following:

- Microsoft Access 97
- Microsoft Excel 97
- Lotus Approach 97
- Lotus 1-2-3 97
- Seagate Crystal Reports 6.0
- Tivoli Decision Support 2.0 or later
- A user-written application

You should be familiar with the primary functions provided by that application. The application must not use ODBC functions not supported by the Tivoli Information Management for z/OS ODBC driver. The ODBC driver supports Core, Level 1, and some Level 2 ODBC functions.

Although you do not have to be an expert on ODBC principles, you may want to review how ODBC functions with Tivoli Information Management for z/OS. For an overview, see “ODBC Concepts” on page 16.

Verifying Your HLAPI/NT Setup

The HLAPI/NT provides remote access to Tivoli Information Management for z/OS data through a server running a transaction program on an MVS, OS/390, or z/OS host system. The HLAPI/NT requester, running on a Windows NT workstation, provides the application program interface to the HLAPI through the Tivoli Information Management for z/OS server.

Before connecting to the Tivoli Information Management for z/OS data source, take a few minutes to ensure your HLAPI/NT setup is working properly to minimize the risk of connection errors when you attempt to retrieve data.

Ensure that your Tivoli Information Management for z/OS HLAPI/NT requester is active. On your workstation, press Ctrl-Alt-Del and select Task Manager from the Windows NT Security window. Look for the Tivoli Information Management for z/OS HLAPI Requester status in the Applications window to determine whether the requester is already active.

If the Requester is not displayed, you can start it.
1. Select the **Tivoli Information Management for z/OS** folder.

2. Select **HLAPI for Windows NT Requester** to start it, then select **OK**.

   - Run the C sample program provided with the HLAPI/NT client to verify that you can initialize and terminate the HLAPI/NT. The steps required to run the program are described in the [Tivoli Information Management for z/OS Client Installation and User's Guide](#).

**Installing the ODBC Driver**

Assuming that your workstation already has Windows NT and the HLAPI/NT client feature installed, follow these procedures to install the driver.

**Note:** You must install the HLAPI/NT client before installing the ODBC driver. The HLAPI/NT client must be installed to use the driver. You can install the ODBC driver over previous versions of the driver.

The ODBC driver setup program installs the driver and the Microsoft ODBC Driver Manager in the Windows NT system directory if they are not already installed (or if the version of the ODBC SDK is earlier than Version 3.51). The system directory is where your Windows NT operating system is installed, such as `\WINNT\SYSTEM32`.

You must ensure that the driver code is installed on the same machine as the HLAPI/NT client feature. To install the driver, follow the steps below.

1. From the Windows Start menu, choose Run. Type the full path and file name for the setup program. On the Tivoli Information Management for z/OS CD-ROM, the setup program is in the ODBC directory. For example:
   ```
e:\ODBC\En\setup.exe
   ```

2. A welcome window is displayed. To continue with the installation, click **Next**. To cancel, click **Cancel**.

3. Select a destination location to install the ODBC driver README.txt file and other supporting files. The default installation location is `c:\Program Files\Tivoli Systems Inc.\Tivoli Service Desk for OS390 ODBC Driver`. Click **Next** to continue with the installation.

   The actual ODBC driver program files will be installed in your Windows system directory; for example, `c:\WINNT\system32`.

4. On the Start Copying Files window, click **Next** to continue with the installation. A small window is displayed to show the progress of files being copied. The ODBC driver run-time files and required ODBC core components are copied to your Windows NT system directory.

5. The Tivoli Information Management for z/OS Data Source Information window is displayed so that you can configure the data source.

   **Note:** An hourglass may appear on the Data Source Information window. If it does, select the Data Source Information window from the started tasks on your Windows taskbar.

---

**Data Reporting User’s Guide**

27
A default value of ServiceDesk390 is displayed as the Data source name, which you can change. If a ServiceDesk390 data source already exists, the window contains the values used on the last successful connection or installation.

If a data source already exists and no configuration changes are required, click **Cancel**.

Enter values for the Security ID and Application ID fields. The security ID is your MVS TSO user ID (what you normally use to log on to Tivoli Information Management for z/OS). The application ID is a symbolic name that Tivoli Information Management for z/OS uses for the connection session (you can obtain this name from your Tivoli Information Management for z/OS system administrator).

At this stage, the Password field is disabled.

6. Select **Options** from the action bar and select **Advanced settings** to fill out the Privilege class, Session member, and Database profile fields on the Data Source Advanced Options window.
The Privilege class is your Tivoli Information Management for z/OS privilege class name that defines the tasks you can perform in Tivoli Information Management for z/OS. The default is MASTER.

The Session member is the session-parameters member name Tivoli Information Management for z/OS uses for the connection session. The default is BLGSES00.

The Database profile is the name of the file containing the HLAPI/NT database profile. The default name is database.pro. You can specify the file name only, or the fully qualified file name with the drive and path.

Note: If you specify just the file name, the database profile is located using the HLAPI/NT environment variable IDBDBPATH. HLAPI/NT first searches the current directory for the file with that name. If the database profile is not found there, HLAPI/NT searches the directories specified by the IDBDBPATH environment variable.

Additional details on what the advanced options represent are available in “Specifying Advanced Options” on page 43.

Once you complete these required fields on the Advanced Options window, click OK to return to the Data Source Information window, and click OK again.

7. A README text file is displayed. After you close the README window, the Setup Complete window is displayed.
8. You can choose to restart your computer now or later. A restart is required to use the ODBC driver properly. Click **Finish** to exit the setup program.

The ODBC driver specification is written to the Windows Registry under the following key structure:

```
HKEY_LOCAL_MACHINE
SOFTWARE
ODBC
ODBCINST.INI
```

**Notes:**

1. If you receive an error message indicating the IDBHLAPI.DLL file could not be found, it is likely that you did not install the HLAPI/NT. To resolve the error, install the HLAPI/NT.

2. The message **Error updating the Windows System Environment HELP variable** is displayed if an error occurs when the Help Path information (Help variable) is obtained from or updated in the registry. To resolve the error, update your System information through the Windows Control Panel. Add the Windows NT System directory (for example, c:\WINNT\system32;) to your Help environment variable.

**Serviceability**

You can view the version of the ODBC driver installed by checking the Windows Properties Version tab of ODBC driver DLL files (blmodbc.dll, blmocprw.dll, or blmocwrp.dll), or by viewing the SrvLevel.txt file located in the directory you specified when you installed the ODBC driver.

**Files Installed**

When you install the ODBC driver, the following required ODBC components and ODBC driver files are installed in the Windows NT system directory (for example, c:\WINNT\system32).

<table>
<thead>
<tr>
<th><strong>ODBC Driver and VisualAge® Driver DLLs</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>blmodbc.dll</td>
<td>ODBC driver DLL</td>
</tr>
<tr>
<td>blmocprw.dll</td>
<td>ODBC driver window resource DLL</td>
</tr>
<tr>
<td>blmocprw.hlp</td>
<td>ODBC driver window resource DLL</td>
</tr>
<tr>
<td>blmocwrp.dll</td>
<td>ODBC driver HLAPI wrappers DLL</td>
</tr>
<tr>
<td>Blmom35i.dll</td>
<td>renamed VisualAge for C++ for Windows DLL</td>
</tr>
<tr>
<td>Blmoob3i.dll</td>
<td>renamed VisualAge for C++ for Windows DLL</td>
</tr>
<tr>
<td>Blmoou3i.dll</td>
<td>renamed VisualAge for C++ for Windows DLL</td>
</tr>
<tr>
<td>Blmoov3.dll</td>
<td>renamed VisualAge for C++ for Windows DLL</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>ODBC 3.51 Core components</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>ds16gt.dll</td>
<td>16-bit driver setup generic thunking</td>
</tr>
<tr>
<td>ds32gt.dll</td>
<td>32-bit driver setup generic thunking</td>
</tr>
<tr>
<td>mtxdm.dll</td>
<td>connection pooling manager</td>
</tr>
<tr>
<td>odbc16gt.dll</td>
<td>16-bit ODBC generic thunking</td>
</tr>
<tr>
<td>odbc32.dll</td>
<td>32-bit driver manager</td>
</tr>
<tr>
<td>odbc32gt.dll</td>
<td>2-bit ODBC generic thunking</td>
</tr>
<tr>
<td><strong>ODBC 3.51 Core components</strong></td>
<td></td>
</tr>
<tr>
<td>--------------------------------</td>
<td></td>
</tr>
<tr>
<td>odbcad32.exe</td>
<td>32-bit Administrator program (previously, the ODBC Administrator program was not installed)</td>
</tr>
<tr>
<td>odbccp32.cpl</td>
<td>32-bit Administrator control panel applet</td>
</tr>
<tr>
<td>odbccp32.dll</td>
<td>32-bit installer</td>
</tr>
<tr>
<td>odbccr32.dll</td>
<td>ANSI cursor library</td>
</tr>
<tr>
<td>odbccu32.dll</td>
<td>Unicode cursor library</td>
</tr>
<tr>
<td>odbcnt.dll</td>
<td>16-bit installer</td>
</tr>
<tr>
<td>odbctrac.dll</td>
<td>ODBC trace</td>
</tr>
<tr>
<td>odbcinst.hlp</td>
<td>installer help file</td>
</tr>
<tr>
<td>odbcinst.cnt</td>
<td>installer help table of contents file</td>
</tr>
<tr>
<td>olepro32.dll</td>
<td>dependency file required for setup</td>
</tr>
<tr>
<td>oleaut32.dll</td>
<td>dependency file required for setup</td>
</tr>
<tr>
<td>Asycfilt.dll</td>
<td>dependency file required for setup</td>
</tr>
<tr>
<td>stdole2.tlb</td>
<td>dependency file required for setup</td>
</tr>
<tr>
<td>msvcr1.dll</td>
<td>dependency file required for setup</td>
</tr>
<tr>
<td>odbcji32.dll</td>
<td>ODBC driver resource library</td>
</tr>
<tr>
<td>odbcj32.dll</td>
<td>ODBC driver for desktop database</td>
</tr>
<tr>
<td>odbctl32.dll</td>
<td>utility routines</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>OLE DB 2.0 – required component of ODBC 3.5</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>msdadc.dll</td>
</tr>
<tr>
<td>msdaenum.dll</td>
</tr>
<tr>
<td>msdaer.dll</td>
</tr>
<tr>
<td>msdaerr.dll</td>
</tr>
<tr>
<td>msdaps.dll</td>
</tr>
<tr>
<td>msdatt.dll</td>
</tr>
<tr>
<td>msdatl2.dll</td>
</tr>
<tr>
<td>msdaoasp.dll</td>
</tr>
<tr>
<td>MSDASC.DLL</td>
</tr>
<tr>
<td>msasc.hlp</td>
</tr>
<tr>
<td>msasc.cnt</td>
</tr>
<tr>
<td>msasc.txt</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>OLE DB ODBC Provider 2.0</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>msdasql.dll</td>
</tr>
<tr>
<td>msdasqr.dll</td>
</tr>
<tr>
<td>&quot;MSDASQLreadme.txt&quot;</td>
</tr>
</tbody>
</table>
Non-system driver files are copied to the destination specified during installation.

<table>
<thead>
<tr>
<th>Non-system driver files</th>
</tr>
</thead>
<tbody>
<tr>
<td>README.txt</td>
</tr>
<tr>
<td>SrvLevel.txt</td>
</tr>
<tr>
<td>Uninst.isu</td>
</tr>
<tr>
<td>_unodbc.log</td>
</tr>
<tr>
<td>_UNODBC.dll</td>
</tr>
</tbody>
</table>

### Uninstalling the ODBC Driver

To uninstall the ODBC driver (on Windows NT), click **Start** on the Windows taskbar, click **Settings**, and select **Control Panel**. Click the **Add/Remove Programs** icon. On the Install/Uninstall page, select **Tivoli Service Desk for OS390 ODBC Driver**, and then click the **Add/Remove** button. The ODBC driver files are removed from the Windows NT system directory and supporting files such as the README.txt are removed from the location specified during installation. The ServiceDesk390 data source is also removed.

### Ensuring Setup of Data Model Records

Confirm that your Tivoli Information Management for z/OS administrator has set up data model records for ODBC access. Without them, you cannot use the Tivoli Information Management for z/OS ODBC driver. For more information on data model records, see “Creating Data Model Records” on page 33.
Creating Data Model Records

Before using your application to retrieve Tivoli Information Management for z/OS data through ODBC, ensure that data model records are created or loaded on the Tivoli Information Management for z/OS host. Usually, this task is performed by a Tivoli Information Management for z/OS administrator. Data model records can be created for purposes other than ODBC. Check with your Tivoli Information Management for z/OS administrator to determine whether data model records have been created or loaded to support your use of the ODBC driver.

Data model records include data attribute records and data view records. Data attribute records represent each piece of data that will represent a column of data in a database table. Data view records represent the table. Multiple data views can exist, and multiple data attribute records can exist in a data view.

If data model records have not been established, your administrator may ask you what types of Tivoli Information Management for z/OS records you need to retrieve and what data you need to access in those records. The overall process used to set up data model records is outlined in the following section.

Note: You can skip this section if you are a business user who will not be setting up data model records, or if the data model records provided to support use of the ODBC driver with Tivoli Decision Support are already loaded in your Tivoli Information Management for z/OS database.

Data Attribute and Data View Records

More information about establishing data attribute records and data view records is available in the Tivoli Information Management for z/OS Panel Modification Facility Guide (see “Using Data Model Records,” Stages 1 and 2). The following sections summarize the steps you should follow to set up data model records for use with Tivoli Information Management for z/OS.

If you intend to use the data model records provided with Tivoli Information Management for z/OS that are enabled for use with the ODBC driver, refer to the Tivoli Information Management for z/OS Planning and Installation Guide and Reference for instructions on how to load them in your database.

Create Data Attribute Records for Each Column of Data

In the Tivoli Information Management for z/OS Panel Modification Facility Guide, follow Stage 1 in “Using Data Model Records” to create a data attribute record for each data field or freeform text field you want to represent as a column of data in your table. For example, if you wanted five fields from the Problem Reporter entry panel, you would create five data
attribute records. When entering the data attribute record information on the Data Attribute Record Entry panel (BLG0V700), be sure to enter the column name you want to associate with the field in the **ODBC column name field** as shown in Figure 8. This is the name that your ODBC users will know these fields by, so you may want to ensure the name is meaningful. The ODBC column name must not exceed 32 characters. It must start with an alphabetic character; the remaining characters must be alphanumeric or underscore characters.

The ODBC driver supports response and freeform text fields. For the most part, you will be able to use the methods described in Stage 1 of “Using Data Model Records” in the [Tivoli Information Management for z/OS Panel Modification Facility Guide](#) to create your attribute records.

**Note:** In some cases where there is an assisted-entry panel, but the s-word is collected from the caller (such as with phone number fields), you will need to ensure that in addition to entering the assisted-entry panel name in the **Copy assisted-entry panel** field, you also enter the s-word for that field into the data attribute record.

Direct-add fields are not accessible through the Tivoli Information Management for z/OS ODBC driver. Direct-add fields are those such as Date entered, Time entered, and perhaps the record ID; that is, fields that are automatically added to the record. If there is no assisted-entry panel for these fields in create or update mode, you can usually find one in your structured search or quick search path. Use that panel as your “Copy assisted-entry panel” to create a response-type data attribute record for those direct-add fields you want to access through the ODBC driver.
Build a Data View Record

Follow the steps outlined in Stage 2 of “Using Data Model Records” in the [Tivoli Information Management for z/OS Panel Modification Facility Guide](#) to build data view records. A data view record corresponds to a table which is used for SQL queries and also for the ODBC functions. The **Data view record name** is the name that your ODBC users will know the table by. The “columns” associated with that “table” are the data attribute records that you list in the data view record. You may want to build multiple data view records depending on your different record types and the particular data you want to include in the different views.

Be sure to reply YES in the **Allow ODBC access?** field on the Data View Record Entry panel, as shown in [Figure 9](#). A reply of YES enables the data view record to be used as a table by the Tivoli Information Management for z/OS ODBC driver.

When you finish, type END to save or CANCEL to discard any changes.

---

**Figure 9. Data View Record Entry Panel**

Follow the instructions in Stage 2 to enter, on the Data Attribute Records Entry panel (shown in [Figure 10 on page 36](#)), the list of record IDs of the data attribute records associated with the data view record. Since the ODBC driver uses the data view record for search and retrieval transactions only, do not enter YES or the letter ’I’ in the **Reply Required** field. If you do, attempts to access Tivoli Information Management for z/OS using that data view record as a table will fail. Also, do not type anything in the **Not Logic** field. If you do, it is ignored.

When you list the **Data Attribute Record IDs**, the ODBC column names are automatically copied from the data attribute record into a scrollable field on the Data Attribute Records Entry panel. If you ever change an ODBC column name in a data attribute record, ensure that you update the data view records that list the data attribute record and reenter the data.
attribute record ID. Doing this will ensure that the new ODBC column name will be copied into the data view record.

Once you have completed Stages 1 and 2, you have finished defining data model records for ODBC usage. You do not need to perform any of the remaining stages as described in the "Tivoli Information Management for z/OS Panel Modification Facility Guide".

As soon as your users install the Tivoli Information Management for z/OS ODBC driver, they can begin using an ODBC-enabled application to search and retrieve Tivoli Information Management for z/OS data.
Using the ODBC Driver

Assuming that your Tivoli Information Management for z/OS administrator has already set up data model records, your Tivoli Information Management for z/OS ODBC driver is installed, and your HLAPI/NT requester is running, you can connect to the Tivoli Information Management for z/OS data source from within your workstation application.

Connecting to the Data Source

The following instructions describe how to connect to the Tivoli Information Management for z/OS data source from a variety of applications. The actual procedure to link to the Tivoli Information Management for z/OS data source may vary depending on the workstation application you use and the procedures you follow in that application. Be sure to refer to your application's documentation or help text for more information or current instructions on connecting to an ODBC data source.

Using Crystal Reports

To access the Tivoli Information Management for z/OS database from Crystal Reports, follow these steps or refer to the documentation provided with Crystal Reports to connect to a data source:

1. Invoke the Crystal Query Designer.
2. From the File menu, select New. Click on the Use Crystal Query Expert icon.
3. Enter a query name (for example, QUERY#1) in the Query Name field and click on the SQL/ODBC icon.
4. Select ODBC-ServiceDesk390 from the Server Type box and click the OK button. (The default name for the Tivoli Information Management for z/OS data source is ServiceDesk390, but you may have defined it as some other data source name at installation.)
5. The Tivoli Information Management for z/OS Data Source Information window appears. Enter the information requested (your password) and click the OK button. Details on using this dialog are provided in "Entering Data Source Information" on page 39.
   A message box is displayed to indicate a successful logon to the SQL server.
6. In the Choose SQL Table dialog, select a table (data view record) and click the Add button. Then, click the Done button.
7. In the Create Query Expert dialog, click the Next button to see the fields available in the table you selected.
8. On the Fields tab, select the fields you want to use and click on the Add button to add each field to your query. Use the selectable fields available to set up the conditions or
values in your search argument. When you have completed the argument, click the **Preview Query** button to retrieve the data from the host database. You should see all the data you have selected for the report.

### Using Lotus Approach

To access the Tivoli Information Management for z/OS database from Lotus Approach, follow these steps or refer to the documentation provided with Lotus Approach to connect to a data source:

1. From the **File** menu, select **Open/Edit SQL**.
2. In the **Tables** tab, select **Create a New SQL Statement** and click the **Add** button.
3. In the **Open** dialog, select **ODBC Data Sources(*)** from the **Files of type:** field. Select the **ServiceDesk390** folder in the **Look in** field and click the **Open** button. (The default name for the Tivoli Information Management for z/OS data source is **ServiceDesk390**, but you may have defined it as some other data source name at installation.)
4. The **Tivoli Information Management for z/OS Data Source Information** window is displayed. Enter the information requested (your password) and click the **OK** button. See "Entering Data Source Information" on page 39 for more information about using this window.
5. In the Lotus Approach **Open** dialog, click on your Tivoli Information Management for z/OS **userid@ServiceDesk390** folder to see the available data view records that represent tables available for you to use. Select the table that you want to use and click the **Open** button.
6. In the **SQL Assistant** dialog, the selected table is listed. Use the tab functions (such as **Condition, Fields, Sort**) to set up your search argument. When you have completed the argument, click the **Done** button.
7. A form is displayed. To view the data, click on the **Worksheet** tab. You should see all the data you have selected for the worksheet.

### Using Lotus 1-2-3

To access the Tivoli Information Management for z/OS database from Lotus 1-2-3, follow these steps or refer to the documentation provided with Lotus 1-2-3 to connect to a data source:

1. From the **File** menu, select **New Workbook**. On the next dialog, click the **OK** button.
2. From the **Create** menu, select **Database** and then select **Query Table**.
3. In the **Query Table Assistant** dialog, select an **external table** and click the **OK** button.
4. Select where you want the table to be displayed on your spreadsheet.
5. In the **Open** dialog, select **ODBC Data Sources(*)** from the **Files of type:** field. Select the **ServiceDesk390** folder in the **Look in** field and click the **Open** button. (The default name for the Tivoli Information Management for z/OS data source is **ServiceDesk390**, but you may have defined it as some other data source name at installation.)
6. The **Tivoli Information Management for z/OS Data Source Information** window is displayed. Enter the information requested (your password) and click the **OK** button. See "Entering Data Source Information" on page 39 for more information about using this window.
7. In the Lotus 1-2-3 Open dialog, click on your Tivoli Information Management for z/OS userid@ServiceDesk390 folder to see the available data view records that represent tables available for you to use. Select the table that you want to use and click the Open button.

8. In the Worksheet Assistant dialog, select the fields you want to use and click the Done button. You should see all the data you have selected for the worksheet.

Using Microsoft Access

To access the Tivoli Information Management for z/OS database from Microsoft Access, follow these steps or refer to the documentation provided with Microsoft Access to connect to a data source:

1. From the File menu, select New Database.... On the New dialog, click the OK button.

2. On the File New Database dialog, enter a file name (for example, TEST1) in the File name: field and click the Create button.

3. On the Database dialog, select the Tables tab and click the New button, and then select Link Table. On the resulting window, select Files of Type ODBC databases.

4. Select the ServiceDesk390 data source in your application. (The default name for the Tivoli Information Management for z/OS data source is ServiceDesk390, but you may have defined it as some other data source name at installation.)

5. The Tivoli Information Management for z/OS Data Source Information window appears. Enter the information requested (your password) and click the OK button. See "Entering Data Source Information" for more information about using this window.

6. A list of one or more tables is displayed in the Link Tables window. Select a table. A list of column names in that table is displayed in the Select Unique Record Identifier window. The names represent the data attribute records that were set up previously on the Tivoli Information Management for z/OS host. Select the Cancel button.

7. Select the REPORT tab to create a report. You can create a filter or query to limit the contents of the report before running it.

8. Run the report and browse the results. You should see all the columns you have selected for the report.

Entering Data Source Information

The Tivoli Information Management for z/OS Data Source Information window is displayed on your workstation when you attempt to connect to the Tivoli Information Management for z/OS host database to retrieve data without supplying all the necessary input data, such as your password (see Figure 6 on page 28). This is a security panel that prompts you to enter a password. Enter the information requested, and press the OK button.

The fields on the Data Source Information window are described in the following list:

Data Source Name
The name established for the Tivoli Information Management for z/OS data source for your application (1 to 32 characters). The data source name is required and is defined at installation. The default data source name is ServiceDesk390.

Security ID
Your MVS TSO user ID (maximum length 7 characters). Your security ID is required. The value you enter is saved and used as a default the next time the connection panel displays.
Connecting to the Data Source

**Password**
Your MVS password (maximum length 8 characters). Your password is required.

**Application ID**
A symbolic name (1 to 8 characters) that Tivoli Information Management for z/OS uses for the connection session. Your Tivoli Information Management for z/OS administrator should provide you with the name of the application ID to use for ODBC purposes.

The **Security ID**, **Password**, and **Application ID** are required by the HLAPI initialization (HL01) transaction to connect to the data source. The **Security ID** and **Application ID** are specified at installation or retained from the last time you connected to the data source.

When you press the **OK** button, the client HLAPI initialization (HL01) transaction is called to connect to the Tivoli Information Management for z/OS data source.

If a connection cannot be established with Tivoli Information Management for z/OS, the ODBC driver issues a message to your application. An error message is then displayed on your workstation to indicate the problem. The error messages that you see at your workstation will depend on how your particular application handles messages.

If you receive message HICARETC= rc HICAREAS=rs msgtext, refer to “Messages” on page 49 for more information on the message.

If no errors occur, a list of one or more tables is displayed within the context of your application. (What you see will depend on the application you are using.) The tables represent the data view records that were created with the **Allow ODBC access?** field set to YES.

When you exit the database, your host connection is dropped. The ODBC API function SQLDisconnect is called and translated to the HLAPI termination transaction (HL02) to terminate the data source connection.

**Defining Queries**
If you open a table or create a report without creating a filter or query, the following SQL statement is issued to retrieve all records of the type specified in the data view:

```
SELECT * FROM tablename
```

- or -

```
SELECT columnname1,columnname2 ... FROM tablename
```

The `tablename` is the name of a data view record that was set up on the Tivoli Information Management for z/OS host to support ODBC.

You may want to create a filter or query to limit the contents of the report before running it. You can also set up a filter or query after you have created the report, but before running it. By setting up a filter or query in your application, you can obtain only the data you are interested in. When you retrieve data for the report, a live transaction is performed against the host database to search and retrieve the most current records. The time it takes to perform the retrieval will vary depending on the size of your Tivoli Information Management for z/OS database and the amount of data you are retrieving.
Up to 500 records (the default) are retrieved unless the Maximum search hits value is changed in the Data Source Advanced Options window. If you anticipate a large list, you may also want to review the setting of your Host API timeout interval to ensure that it is adequate. You can change the driver defaults for these options as described in Changing Driver Options or Attributes on page 43.

If your Tivoli Information Management for z/OS database contains a significant number of records, you will probably want to set up a filter or a more specific query. For example, instead of getting back all records in a table, you could set up a WHERE clause in your application to limit the data as follows:

```
SELECT * FROM tablename WHERE DATE_ENTERED BETWEEN '1998/05/01' AND '1998/06/01' AND STATUS NOT 'CLOSED'
-or-
SELECT REPORTER, DESCRIPTION FROM tablename WHERE STATUS='OPEN'
```

Refer to your application’s documentation for instructions on how to change the selection criteria of the data.

You can rerun the report with a revised query if needed. A new data inquiry and retrieval transaction takes place to run a revised SELECT statement to pull the data.

When you are satisfied with your report (and have supplied the WHERE clauses you need), you can save the report format in your application and run the report at a later time. Saving the format saves the structure and query of the report (but not the data) and provides a useful way to get refreshed reports periodically.

When you have created a report, you can perform other application functions such as printing the report contents or merging the report with other documents (for example, text documents or spreadsheets). Refer to the documentation provided with your application for instructions.

Likewise, if you wanted to keep a snapshot of the data as it appeared at that time, you can import the data to a file on your workstation, such as a database file, using the import function provided by your application. You can also save the report and its contents.
Changing Driver Options or Attributes

During installation, you can specify driver options or attributes that will be used with your ODBC connection. The options described in this section are used in the host MVS environment and by the HLAPI/NT client. For most users, it is not necessary to change the ODBC driver defaults or advanced login options after the driver is installed. In the event, however, you want to change a driver option, such as the maximum number of search hits or which Tivoli Information Management for z/OS database is to be used, you can specify a new value on one of the driver connection panels displayed on your workstation. The new value you supply is saved and displayed the next time the ODBC driver is invoked.

When your application attempts to connect to the ServiceDesk390 data source, the ODBC driver may display a connection window (Tivoli Information Management for z/OS Data Source Information) where you can type your security ID, password, and application ID. A password is required. The security ID and application ID fields display the values initially set at installation (because they are saved in the Registry) or the values used the last time you successfully connected to the Tivoli Information Management for z/OS host.

If you select Options on the action bar of this window and select Advanced Settings, the Tivoli Information Management for z/OS Data Source Advanced Options panel displays the default login options.

To change the value of an option, type the new value and click the OK button to return to the Tivoli Information Management for z/OS Data Source Information window.

Note: Some ODBC-enabled applications may require that you recreate the database to change the advanced options. To change a field on the Advanced Options panel if the field is disabled, use the ODBC Administrator to reconfigure the options or recreate the database and enter the value desired.

Specifying Advanced Options

The advanced driver options are used to log on to the Tivoli Information Management for z/OS host and define API operating characteristics and database access. A valid value for each option must be specified at installation, on the driver-defined connection string, or through the Advanced Options panel. If the connection string does not contain the DRIVER keyword, whether or not the data source name (DSN) keyword exists, the values specified at installation in the Windows NT Registry are used to initialize these fields.

If you specify this information through driver-defined connection string attribute keywords, rather than through the Advanced Options panel, refer to “SQL Command Syntax and Parameters” on page 107 for keyword details.
The following options represent parameter data blocks (PDBs) on the chain for HLAPI transactions HL01 (initialization), HL06 (record retrieval), and HL11 (record inquiry) as noted. Refer to the [Tivoli Information Management for z/OS Application Program Interface Guide](https://www.ibm.com) for more information on PDBs and HLAPI transactions.

**Privilege class**

The 1–8 character startup privilege class name in Tivoli Information Management for z/OS. The privilege class defines what tasks a user can perform. The application ID entered on the [Tivoli Information Management for z/OS Data Source Advanced Options](https://www.ibm.com) panel must be an eligible user of this privilege class. The default is MASTER. This field is known as PDB name PRIVILEGE_CLASS and is required for the HL01 initialization transaction.

**Session member**

A 7 or 8 character load library session-parameters member name Tivoli Information Management for z/OS uses for the connection session. The default is BLGSESS00. This field is known as PDB name SESSION_MEMBER and is required for the HL01 initialization transaction.

If you are using an MRES with a pre-started session, the value specified in the MRES startup parameter for session member must match the value entered here. If the values do not match, an error occurs.

**Database profile**

The name of the file containing the HLAPI/NT database profile. The default is database.pro. The database profile establishes settings for use with the Tivoli Information Management for z/OS database, such as the Tivoli Information Management for z/OS server host name, client character code set, and log file size.

You can specify either the file name only or you can fully qualify the name with its drive and path. If you specify only a file name, the file is obtained by searching the directories specified by the environment variable IDBDBPATH. This field is known as PDB name DATABASE_PROFILE, and is required for the HL01 initialization transaction. Refer to the [Tivoli Information Management for z/OS Client Installation and User’s Guide](https://www.ibm.com) for more information on the database profile.

**LLAPI message option**

A 1-character designation for the destination of LLAPI messages for the session: P (print data set or SYSOUT), C (message should pass to the HLAPI), or B (both P and C). The default is C. This optional field is known as PDB name APIMSG_OPTION and is used in the HL01 initialization transaction.

If you are using an MRES with a pre-started session, and enter a value for the LLAPI message option that does not match the value specified for the pre-started MRES session, the value entered here is ignored.

**HLAPI message option**

A 1-character designation indicating how the HLAPI processes messages for the session—either P (write messages to the HLAPILOG data set), C (chains messages on the message PDB chain), or B (both P and C). The default is C. This optional field is known as PDB name HLIMSG_OPTION and is used in the HL01 initialization transaction.

If you are using an MRES with a pre-started session, and enter a value for the HLAPI message option that does not match the value specified for the pre-started MRES session, the value entered here is ignored.
Database ID
The name or ID number of the database your application uses for the session. The default is 5. This optional field is known as PDB name DATABASE_ID and is used in the HL01 initialization transaction.

If you are using an MRES with a pre-started session, the value specified in the MRES startup parameter for database ID must match the value entered here. If the values do not match, an error occurs.

Host log spool interval
The time in minutes between API spooling of activity logs. The API spools activity logs when it prints messages. The interval can be from one to 4 digits. The default is 100 minutes. This optional field is known as PDB name SPOOL_INTERVAL and is used in the HL01 initialization transaction.

If you are using an MRES with a pre-started session, and enter a value for the host log spool interval that does not match the value specified for the pre-started MRES session, the value entered here is ignored.

Host API timeout interval
The time in seconds that a transaction can run before a timer interrupt occurs. The interval can be from one to 4 digits. The default is 300 seconds. If you work with large search hit lists, you may want to increase the interval to avoid getting timed out. If a timer interrupt occurs the connection to Tivoli Information Management for z/OS is terminated. This optional field is known as PDB name TIMEOUT_INTERVAL and is used in the HL01 initialization transaction.

If you are using an MRES with a pre-started session, and enter a value for the host API timeout interval that does not match the value specified for the pre-started MRES session, the value entered here is ignored.

HLAPI table count
The maximum number of alias tables and PIDT tables that can be in storage during the Tivoli Information Management for z/OS session. The table count value is used for caching purposes and factors into the overall performance of data retrieval. You can specify any value between 1 and 256. The default is 5 tables, and it is recommended that you not specify a value of less than 5. This optional field is known as PDB name TABLE_COUNT and is used in the HL01 initialization transaction.

If you are using an MRES with a pre-started session, and enter a value for the HLAPI table count that does not match the value specified for the pre-started MRES session, the value entered here is ignored.

Maximum search hits
The maximum number of matches to be returned from a search. The default is 500. This optional field is known as PDB name NUMBER_OF_HITS and is used in the HL11 record inquiry transaction.

Freeform text max lines limit
The maximum number of text lines (text units) that the API can retrieve in the response buffer for each freeform text field. The default is 60 lines. This optional field is known as PDB name TEXT_UNITS and is used in the HL06 record retrieval transaction.

Freeform text line length
The maximum width of a text line or unit that the API can retrieve in the response
buffer. The width can be any value between 1 and 132. The default is 60. This optional field is known as PDB name TEXT_WIDTH and is used in the HL06 record retrieval transaction.

**Freeform text area**

A character string designation indicating whether the bottom block (Bottom) or top block (Top) of text data is retrieved in the HLAPI when the number of text lines or units available exceeds the amount specified in the **Freeform text max lines limit** field. The default is Bottom. This optional field is known as PDB name TEXT_AREA and is used in the HL06 record retrieval transaction.

If you are using an MRES with a pre-started session, and if you are retrieving data from a multiple response field, your MRES startup parameters should include the following parameter:

MULTIPLE_RESPONSE_FORMAT=PHRASE

If this parameter is omitted, the data returned from multiple response fields will include padded blanks and the separator character defined for the field. For more information see “Multiple Response Field Processing” on page 23.
Messages

This chapter describes the following:

- The format of error messages returned by the ODBC driver
- The return codes and reason codes (SQLSTATE values) returned by the Tivoli Information Management for z/OS ODBC driver (see "Return and Reason Codes Issued by the ODBC Driver by Function" on page 52).

Message Formats

The Tivoli Information Management for z/OS ODBC driver may generate messages indicating errors. For errors that do not occur in a data source, the error text format is:

```
[vendor-identifier] [ODBC-component-identifier] component-supplied-text
```

For errors that occur in a data source (for example, ServiceDesk390 or the HLAPI), the error text format is:

```
[vendor-identifier] [ODBC-component-identifier] [data-source-identifier] data-source-supplied-text
```

**vendor-identifier:**
- **Tivoli** Vendor of the component in which the error occurred or that received the error directly from the Tivoli Information Management for z/OS HLAPI.
- **Microsoft** For messages generated by the driver manager. Refer to Microsoft’s online API reference information for a description of SQLSTATE values.

**ODBC-component-identifier:**
- **Tivoli Information Management for z/OS ODBC Driver** The component in which the error occurred or that received the error directly from the Tivoli Information Management for z/OS HLAPI.
- **ODBC Driver Manager** The ODBC component identifier for messages generated by the driver manager.

**data-source-identifier:**
- The data source name as specified in the SQLConnect or SQLDriverConnect function. The data source name is defined at installation. The default name is ServiceDesk390.
component-supplied text
Generated by the driver manager or the Tivoli Information Management for z/OS ODBC driver based on the SQLSTATE.

data-source-supplied text
Generated by the data source based on the SQLSTATE

For example, if you are using Microsoft Access when connecting to Tivoli Information Management for z/OS and the HLAPI/NT requester is not started, the following series of messages can appear at your workstation (one driver manager message followed by three ODBC driver messages). By focusing on the last error message, you can generally identify the cause of the problem and take the appropriate action.

[Microsoft][ODBC Driver Manager]Driver's SQLSetConnectOption failed (#0) [Tivoli]
[Tivoli Information Management for z/OS ODBC Driver] Driver not capable (#0) [Tivoli][Tivoli Information Management for z/OS ODBC Driver] Unable to connect to data source (#0) [Tivoli][Tivoli Information Management for z/OS ODBC Driver] [ServiceDesk390] HICARETC=12 HICAREAS=109 The requester is not available. A requester must be started before HLAPI/NT services can be accessed by a client application. (#1)

ODBC Driver Manager Tracing
You can enable tracing and specify a trace file, or disable tracing, through the ODBC Administrator icon accessible through the Control Panel window on Windows NT 4.0. If tracing is enabled, the driver manager writes the name of each ODBC function called, the value of the input arguments, and the name of the output arguments to the trace file. The default trace log is \SQL.LOG. Your ODBC-enabled workstation application can specify whether or not to perform tracing and to use the trace file. Tracing is performed through the SQLSetConnectOption options SQL_OPT_TRACE and SQL_OPT_TRACEFILE. Refer to your application’s documentation for more information on enabling driver manager tracing.

ODBC Driver Tracing
If you want to see tracing information, you should ensure that the ODBC driver debug tracing is enabled by setting the following Windows NT system environment variables with the values shown. The tracing information provided for the ODBC driver is development-level information that may be useful to Tivoli Customer Support personnel when resolving problems. The setting of these variables can cause the ODBC driver to slow down; therefore, you should do this only when you encounter an ODBC driver problem.

ICLUI_TRACE
ON

ICLUI_TRACEFILE
The fully qualified name of the trace file

ICLUI_TRACETO
FILE
Messages

The following messages are generated by, and are unique to, the Tivoli Information Management for z/OS ODBC driver. Depending on how your ODBC-enabled application handles messages, you may or may not see these messages displayed at your workstation. The text is displayed as the \textit{component-supplied-text} or the \textit{data-source-supplied-text} in the error message format.

Messages are listed in alphabetic order by the message text.

\textbf{Column name is not valid.}

\textbf{Explanation:} The column name is too long.

\textbf{Problem Determination:} The column names supported by the Tivoli Information Management for z/OS data source can be up to 32 characters long. The column names must start with an alphabetic character; remaining characters must be alphanumeric or underscore.

\textbf{HICARETC=rc HICAREAS=rs msgtext.}

\textbf{Explanation:} An error was returned from a HLAPI transaction. The HLAPI return code \textit{rc} and HLAPI reason code \textit{rs} are provided in the message text. The \textit{msgtext} is based on the \textit{rc} and \textit{rs} values.

\textbf{Problem Determination:} Call SQLError to see if there is additional error information. Review the \textit{msgtext} and any additional error information. Refer to the Tivoli Information Management for z/OS Client Installation and User Guide for an explanation of the HLAPI return and reason codes (\textit{rc} and \textit{rs}) and take the appropriate action. If you have to restart the HLAPI/NT requester or the Tivoli Information Management for z/OS server (RES, MRES with APPC, or MRES with TCP/IP), you may need to restart your application.

\textbf{msgtext.}

\textbf{Explanation:} The HLAPI issued a message which was placed on the message PDB chain. The \textit{msgtext} is the text of the Tivoli Information Management for z/OS message.

\textbf{Problem Determination:} Review the \textit{msgtext} in this message and refer to Tivoli Information Management for z/OS Messages and Codes for an explanation. This message will always be preceded by the message HICARETC=rc HICAREAS=rs msgtext. Review the HLAPI return and reason codes (\textit{rc} and \textit{rs}) in the error preceding this error and refer to the Tivoli Information Management for z/OS Client Installation and User's Guide for an explanation of the \textit{rc} and \textit{rs}.

\textbf{Null value cannot be specified in a NOT <, >=, NOT >, or <= WHERE clause.}

\textbf{Explanation:} A null value cannot be specified for a column in a NOT greater than, greater than or equal to, NOT less than, or less than or equal to \textit{search-condition}.

\textbf{Problem Determination:} The value you specified for a column in a NOT greater than, greater than or equal to, NOT less than, or less than or equal to \textit{search-condition} was null. You cannot specify a null value for any of these \textit{search-conditions}.

\textbf{NULL/NOT NULL clause can be specified only for a column that is a prefixed field in Tivoli Information Management for z/OS.}

\textbf{Explanation:} A column name you specified in a NULL or NOT NULL \textit{search-condition} is not a prefixed field in Tivoli Information Management for z/OS.

\textbf{Problem Determination:} A column name used in a NULL or NOT NULL \textit{search-condition} must correspond to a prefixed field in Tivoli Information Management for z/OS. Do not use a NULL or NOT NULL \textit{search-condition} for that column.
Programming error msgtext.

**Explanation:** The Tivoli Information Management for z/OS ODBC driver encountered an unexpected error. The function that encountered the error is identified in msgtext.

**Problem Determination:** Record the error and retry your application. If the problem persists and if the Tivoli Information Management for z/OS ODBC driver tracing is enabled, check the trace output located in the output destination specified by the ICLUI_TRACETO environment variable for the possible cause. Contact Tivoli Customer Support for further assistance if appropriate.

Table name is not valid.

**Explanation:** The table name is too long.

**Problem Determination:** The table names supported by the Tivoli Information Management for z/OS data source can be up to 8 characters long.

SELECT statement syntax is not valid or syntax is not supported by Tivoli Information Management for z/OS ODBC Driver.

**Explanation:** The argument szSqlStr contained an SQL statement that contained a syntax error or contained syntax that was not supported by the Tivoli Information Management for z/OS ODBC driver.

**Problem Determination:** Review the argument in szSqlStr and the syntax supported by the Tivoli Information Management for z/OS ODBC driver and correct it as necessary. For a description of SQL syntax see [“Elements Used in SQL Statements” on page 118](#).

SELECT syntax error – ‘<’ and ‘>’ comparisons are not supported.

**Explanation:** The argument szSqlStr contained an SQL statement that has syntax not supported by the Tivoli Information Management for z/OS ODBC driver. Comparisons in WHERE clauses cannot be ‘<’ (less than) or ‘>’ greater than.

**Problem Determination:** Review the argument in szSqlStr and correct it as necessary. For a description of SQL syntax see [“Elements Used in SQL Statements” on page 118](#).

SELECT syntax error – 1st and 2nd or 1st and 3rd BETWEEN operands cannot both be parameters.

**Explanation:** The argument szSqlStr contained an SQL statement that has syntax which is not valid. You must be able to infer the data type of the parameter from an operand.

**Problem Determination:** Review the argument in szSqlStr and correct it as necessary. For a description of SQL syntax see [“Elements Used in SQL Statements” on page 118](#).

SELECT syntax error – BETWEEN must start with a column name, a numeric literal, or a parameter.

**Explanation:** The argument szSqlStr contained an SQL statement that contained a syntax error or contained syntax that is not supported by the Tivoli Information Management for z/OS ODBC driver. The BETWEEN clause may have started with a character string literal.

**Problem Determination:** Review the argument in szSqlStr and correct it as necessary. For a description of SQL syntax see [“Elements Used in SQL Statements” on page 118](#).

SELECT syntax error – BETWEEN target and values must not mix character string and numeric literals or parameters and numeric literals.

**Explanation:** The argument szSqlStr contained an SQL statement that contained a syntax error or contained syntax that is not supported by the Tivoli Information Management for z/OS ODBC driver. Numeric literals in a BETWEEN predicate are only supported by the Tivoli Information Management for z/OS ODBC driver when both the target and its values are numeric literals.

**Problem Determination:** Review the argument in szSqlStr and correct it as necessary. For a description of SQL syntax see [“Elements Used in SQL Statements” on page 118](#).
SELECT syntax error – BETWEEN value must be a character string literal, a numeric literal, or a parameter.

Explanation: The argument szSqlStr contained an SQL statement that contained a syntax error or contained syntax that is not supported by the Tivoli Information Management for z/OS ODBC driver. The BETWEEN clause may contain a column name as one of the values in the comparison.

Problem Determination: Review the argument in szSqlStr and correct it as necessary. For a description of SQL syntax see “Elements Used in SQL Statements” on page 118.

SELECT syntax error – BETWEEN values cannot be numeric literals when BETWEEN target is a column name.

Explanation: The argument szSqlStr contained an SQL statement that contained a syntax error or contained syntax that is not supported by the Tivoli Information Management for z/OS ODBC driver. The BETWEEN clause target is a column name and the values are numeric literals. All Tivoli Information Management for z/OS data is character data and, therefore, the comparison is not valid.

Problem Determination: Review the argument in szSqlStr and correct it as necessary. For a description of SQL syntax see “Elements Used in SQL Statements” on page 118.

SELECT syntax error – LIKE data must contain 1 literal string or parameter.

Explanation: The argument szSqlStr contained an SQL statement that has a LIKE clause not supported by the Tivoli Information Management for z/OS ODBC driver. A LIKE clause must contain one literal string denoting the search argument data or parameter denoting the search argument data or field name.

Problem Determination: Review the argument in szSqlStr and correct it as necessary. For a description of SQL syntax see “Elements Used in SQL Statements” on page 118.

SELECT syntax error – NULL/NOT NULL clause must contain a column name or parameter.

Explanation: The argument szSqlStr contained an SQL statement that has a NULL or NOT NULL clause that did not contain a column name or parameter. NULL and NOT NULL clauses must contain a column name or parameter.

Problem Determination: Review the argument in szSqlStr and correct it as necessary. For a description of SQL syntax see “Elements Used in SQL Statements” on page 118.

SELECT syntax error – only one SELECT column table name can be specified and it must match the FROM table name or correlation name.

Explanation: The argument szSqlStr contained an SQL statement that has syntax not supported by the Tivoli Information Management for z/OS ODBC driver. All table names used as column name qualifiers must match the FROM table name and only one FROM table is valid.

Problem Determination: Review the argument in szSqlStr and correct it as necessary. For a description of SQL syntax see “Elements Used in SQL Statements” on page 118.

SELECT syntax error – SELECT column names cannot be literal strings or parameters.

Explanation: The argument szSqlStr contained an SQL statement that has syntax not supported by the Tivoli Information Management for z/OS ODBC driver. SELECT field names cannot be quoted strings or parameters.

Problem Determination: Review the argument in szSqlStr and correct it as necessary. For a description of SQL syntax see “Elements Used in SQL Statements” on page 118.
Messages

SELECT syntax error – WHERE comparison cannot have two parameters.

Explanation: The argument szSqlStr contained an SQL statement that has syntax which is not valid. A WHERE comparison clause cannot consist of two parameters.

Problem Determination: Review the argument in szSqlStr and correct it as necessary. For a description of SQL syntax see Elements Used in SQL Statements on page 118.

SELECT syntax error – WHERE comparison must not contain COUNT(*), 2 column names, 2 character string literals, a character string literal and a numeric literal, a parameter and a numeric literal or a column name and a numeric literal.

Explanation: The argument szSqlStr contained an SQL statement that contained a WHERE clause not supported by the Tivoli Information Management for z/OS ODBC driver. For example, any of the following WHERE clauses will cause this error:

WHERE REPORTER=SEVERITY (two column names)
WHERE "0"="1" (two character string literals)
WHERE "PHONE"=5551212 (a character string literal and numeric literal)
WHERE ? > 6 (a parameter and a numeric literal)
WHERE SEVERITY=4 (a column name and a numeric literal)

Problem Determination: Review the argument in szSqlStr and correct it as necessary. For a description of SQL syntax see Elements Used in SQL Statements on page 118.

A WHERE column is not searchable.

Explanation: A column name you specified in a WHERE search-condition cannot be searched. The field that corresponds to the column name is not cognized in the Tivoli Information Management for z/OS database.

Problem Determination: Do not specify the column in a search-condition.

Return and Reason Codes Issued by the ODBC Driver by Function

The following section lists, by ODBC function, the SQLSTATE values and messages returned by the Tivoli Information Management for z/OS ODBC driver. It is intended for programmers writing ODBC applications for use with the Tivoli Information Management for z/OS ODBC driver, or for support personnel who are involved in identifying and resolving problems.

If the vendor identifier on the messages is Microsoft, refer to Microsoft’s online API reference information for a description of SQLSTATE values and messages.

The following list of return and reason codes represents a subset of those codes that are defined for the ODBC interface. These codes are returned by the Tivoli Information Management for z/OS ODBC driver, and are described here with explanations specific to the Tivoli Information Management for z/OS ODBC driver. Listed with each code is the message text that is displayed in the component-supplied-text or the data-source-supplied-text areas of error messages returned by the driver.

Your application can call the SQLError function to retrieve additional SQLSTATE information about errors associated with the SQL_ERROR or SQL_SUCCESS_WITH_INFO messages.
## SQLAllocConnect

### SQL_SUCCESS (0) X'0000'

**Explanation:** SQLAllocConnect completed successfully.

### SQL_ERROR (-1) X'FFFF'

**Explanation:** An error occurred in SQLAllocConnect.

**Problem Determination:** Review any additional information associated with the error. Driver-specific SQLSTATEs are:

- **IDBPE** - Programming error `msgtext`.
  
  The Tivoli Information Management for z/OS ODBC driver encountered an unexpected error. The function that encountered the error is identified in `msgtext`. Record the error and retry your application. If the problem persists and if the Tivoli Information Management for z/OS ODBC driver tracing is enabled, check the trace output located in the output destination specified by the `ICLUI_TRACETO` environment variable for the possible cause. Contact Tivoli Customer Support for further assistance if appropriate.

- **S1001** - Memory allocation failure.
  
  Try closing any unnecessary windows. If the problem persists, restart your system and try the function again. In addition, turning off the ODBC driver tracing and HLAPI/NT logging may improve your resource usage.

### SQL_INVALID_HANDLE (-2) X'FFFE'

**Explanation:** An environment handle that is not valid was encountered and SQLAllocConnect failed.

**Problem Determination:** A programming error occurred. Contact Tivoli Customer Support.
Return and Reason Codes by Function

**SQLAllocEnv**

**SQL_SUCCESS (0) X'0000'**

**Explanation:** SQLAllocEnv completed successfully.

**SQL_ERROR (-1) X'FFFF'**

**Explanation:** An error occurred in SQLAllocEnv.

**Problem Determination:** A memory allocation error occurred. Try closing any unnecessary windows. If the problem persists, restart your system and try the function again.
SQLAllocStmt

SQL_SUCCESS (0) X'0000'

Explanation: SQLAllocStmt completed successfully.

SQL_ERROR (-1) X'FFFF'

Explanation: An error occurred in SQLAllocStmt.

Problem Determination: Review any additional information associated with the error. Driver-specific SQLSTATEs are:

IDBPE  Programming error msgtext.

The Tivoli Information Management for z/OS ODBC driver encountered an unexpected error. The function that encountered the error is identified in msgtext. Record the error and retry your application. If the problem persists and if the Tivoli Information Management for z/OS ODBC driver tracing is enabled, check the trace output located in the output destination specified by the ICLUI_TRACETO environment variable for the possible cause. Contact Tivoli Customer Support for further assistance if appropriate.

S1001  Memory allocation failure.

Try closing any unnecessary windows. If the problem persists, restart your system and try the function again. In addition, turning off the ODBC driver tracing and HLAPI/NT logging may improve your resource usage.

SQL_INVALID_HANDLE (-2) X'FFFE'

Explanation: A connection handle that is not valid was encountered and SQLAllocStmt failed.

Return and Reason Codes by Function

SQLBindCol

SQL_SUCCESS (0) X'0000'

Explanation: SQLBindCol completed successfully.

SQL_ERROR (-1) X'FFFF'

Explanation: An error occurred in SQLBindCol.

Problem Determination: Review any additional information associated with the error. Driver-specific SQLSTATEs are:

IDBPE  Programming error msgtext.

The Tivoli Information Management for z/OS ODBC driver encountered an unexpected error. The function that encountered the error is identified in msgtext. Record the error and retry your application. If the problem persists and if the Tivoli Information Management for z/OS ODBC driver tracing is enabled, check the trace output located in the output destination specified by the ICLUI_TRACETO environment variable for the possible cause. Contact Tivoli Customer Support for further assistance if appropriate.

S1001  Memory allocation failure.

Try closing any unnecessary windows. If the problem persists, restart your system and try the function again. In addition, turning off the ODBC driver tracing and HLAPI/NT logging may improve your resource usage.

S1C00  Driver not capable.

Verify the argument fCType is supported by the driver (see page 107). If it is a supported value, contact Tivoli Customer Support.

SQL_INVALID_HANDLE (2) X'FFFE'

Explanation: A statement handle that is not valid was encountered and SQLBindCol failed.

SQLBindParameter

**SQL_SUCCESS (0) X'0000'**

**Explanation:**  SQLBindParameter completed successfully.

**SQL_ERROR (-1) X'FFFF'**

**Explanation:**  An error occurred in SQLBindParameter.

**Problem Determination:**  Review any additional information associated with the error. Driver-specific SQLSTATEs are:

- **IDBPE**  Programming error msgtext.
  
  The Tivoli Information Management for z/OS ODBC driver encountered an unexpected error. The function that encountered the error is identified in msgtext. Record the error and retry your application. If the problem persists and if the Tivoli Information Management for z/OS ODBC driver tracing is enabled, check the trace output located in the output destination specified by the ICLUI_TRACETO environment variable for the possible cause. Contact Tivoli Customer Support for further assistance if appropriate.

- **S1001**  Memory allocation failure.
  
  Try closing any unnecessary windows. If the problem persists, restart your system and try the function again. In addition, turning off the ODBC driver tracing and HLAPI/NT logging may improve your resource usage.

- **S1C00**  Driver not capable.
  
  Verify the argument fSqlType is supported by the driver (see page 107). If it is a supported value, contact Tivoli Customer Support.

**SQL_INVALID_HANDLE (-2) X'FFFE'**

**Explanation:**  A statement handle that is not valid was encountered and SQLBindParameter failed.

**Problem Determination:**  A programming error occurred. Contact Tivoli Customer Support.
SQL Cancel

SQL_SUCCESS (0) X'0000'

Explanation: SQL Cancel completed successfully.

SQL_INVALID_HANDLE (-2) X'FFFE'

Explanation: A statement handle that is not valid was encountered and SQL Cancel failed.

7. Messages

SQLColAttributes

**SQL_SUCCESS (0) X'0000'**

Explanation: SQLColAttributes completed successfully.

**SQL_SUCCESS_WITH_INFO (1) X'0001'**

Explanation: SQLColAttributes completed successfully, but an error may have occurred.

Problem Determination: Review any additional information associated with the message. Driver-specific SQLSTATEs are:

- **01004** Data truncated.

  Truncation of the string value occurred because the rgbDesc buffer was not large enough. If you are writing an application for use with the ODBC driver, consider making the buffer equal to or greater than the value contained in argument pcbDesc.

**SQL_ERROR (-1) X'FFFF'**

Explanation: An error occurred in SQLColAttributes.

Problem Determination: Review any additional information associated with the error. Driver-specific SQLSTATEs are:

- **24000** Invalid cursor state.

  Ensure that your application is executing the SQL statement in the proper sequence, or contact support personnel for your ODBC-enabled workstation application.

- **IDBPE** Programming error msgtext.

  The Tivoli Information Management for z/OS ODBC driver encountered an unexpected error. The function that encountered the error is identified in msgtext. Record the error and retry your application. If the problem persists and if the Tivoli Information Management for z/OS ODBC driver tracing is enabled, check the trace output located in the output destination specified by the ICLUI_TRACETO environment variable for the possible cause. Contact Tivoli Customer Support for further assistance if appropriate.

- **S1002** Invalid column number.

  Specify a valid icol. You can use SQLNumResultCols to determine the number of columns in the result set.

**SQL_INVALID_HANDLE (-2) X'FFFE'**

Explanation: A statement handle that is not valid was encountered and SQLColAttributes failed.

SQLColumns

SQL_SUCCESS (0) X'0000'

Explanation: SQLColumns completed successfully.

SQL_ERROR (-1) X'FFFF'

Explanation: An error occurred in SQLColumns.

Problem Determination: Review any additional information associated with the error. Driver-specific SQLSTATEs are:

<table>
<thead>
<tr>
<th>SQLSTATE</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>08S01</td>
<td>Communication link failure. The HLAPI/NT connection failed. Call SQLError for additional error information. If the HLAPI/NT requester is not running, start it and restart your application. If the Tivoli Information Management for z/OS server on MVS (RES, MRES with APPC, or MRES with TCP/IP) is not active, start it and restart your application. Check for any errors that the HLAPI/NT may have written to the IDBPProbe.LOG file.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SQLSTATE</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IDB02</td>
<td>HICARETC=rc HICAREAS=rs msgtext. The HLAPI/NT issued a HLAPI return code rc and HLAPI reason code rs. The msgtext is based on the rc and rs values. Review the msgtext. For additional information, refer to the Tivoli Information Management for z/OS Client Installation and User’s Guide for an explanation of the HLAPI return and reason codes and take the appropriate action. If you have to restart the HLAPI/NT requester or the Tivoli Information Management for z/OS server (RES, MRES with APPC, or MRES with TCP/IP), restart your application.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SQLSTATE</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IDBC0</td>
<td>Column name is not valid. The column name is too long. The columns names supported by the Tivoli Information Management for z/OS data source can be up to 32 characters long.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SQLSTATE</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IDBPE</td>
<td>Programming error msgtext. The Tivoli Information Management for z/OS ODBC driver encountered an unexpected error. The function that encountered the error is identified in msgtext. Record the error and retry your application. If the problem persists and if the Tivoli Information Management for z/OS ODBC driver tracing is enabled, check the trace output located in the output destination specified by the ICLUI_TRACETO environment variable for the possible cause. Contact Tivoli Customer Support for further assistance if appropriate.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SQLSTATE</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IDBT0</td>
<td>Table name is not valid. The table name is too long. The table names supported by the Tivoli Information Management for z/OS data source can be up to 8 characters long.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SQLSTATE</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1001</td>
<td>Memory allocation failure. Try closing any unnecessary windows. If the problem persists, restart your system and try the function again. In addition, turning off the ODBC driver tracing and HLAPI/NT logging may improve your resource usage.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SQLSTATE</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1C00</td>
<td>Driver not capable. Do not specify a table qualifier (szTableQualifier) or a table owner (szTableOwner).</td>
</tr>
</tbody>
</table>
SQL.INVALID_HANDLE (-2) X'FFFE'

**Explanation:** A statement handle that is not valid was encountered and SQLColumns failed.

**Problem Determination:** A programming error occurred. Contact Tivoli Customer Support.
SQLConnect

**SQL_SUCCESS (0) X'0000'**

**Explanation:** SQLConnect completed successfully.

**SQL_ERROR (-1) X'FFFF'**

**Explanation:** An error occurred in SQLConnect.

**Problem Determination:** Review any additional information associated with the error. Driver-specific SQLSTATEs are:

- **08001** Unable to connect to data source.
  
  A connection did not occur because all the information needed to perform the connection was not provided. Refer to the [Tivoli Information Management for z/OS Client Installation and User’s Guide](#) for an explanation of the return and reason code as listed in the text of the message IDB01.

- **IDB01** HICARETC=rc, HICAREAS=rs, msgtext.
  
  The HLAPI/NT issued a HLAPI return code rc and HLAPI reason code rs. The message text is based on the rc and rs values.
  
  Call SQLError to see if there is additional error information. Review the msgtext and any additional error information. Refer to the [Tivoli Information Management for z/OS Client Installation and User’s Guide](#) for an explanation of the HLAPI return and reason codes (rc and rs) and take the appropriate action.

- **IDBPE** Programming error msgtext.
  
  The Tivoli Information Management for z/OS ODBC driver encountered an unexpected error. The function that encountered the error is identified in msgtext. Record the error and retry your application. If the problem persists and if the Tivoli Information Management for z/OS ODBC driver tracing is enabled, check the trace output located in the output destination specified by the ICLUI_TRACETO environment variable for the possible cause. Contact Tivoli Customer Support for further assistance if appropriate.

- **S1001** Memory allocation failure.
  
  Try closing any unnecessary windows. If the problem persists, restart your system and try the function again. In addition, turning off the ODBC driver tracing and HLAPI/NT logging may improve your resource usage.

**SQL_INVALID_HANDLE (-2) X'FFFFFF'**

**Explanation:** A connection handle that is not valid was encountered and SQLConnect failed.

**Problem Determination:** A programming error occurred. Contact Tivoli Customer Support.
### SQLDescribeCol

**SQL_SUCCESS (0) X'0000'**

**Explanation:** SQLDescribeCol completed successfully.

**SQL_SUCCESS_WITH_INFO (1) X'0001'**

**Explanation:** SQLDescribeCol completed successfully, but an error may have occurred.

**Problem Determination:** Review any additional information associated with the message. Driver-specific SQLSTATEs are:

- **01004** Data truncated.
  
  Truncation of the column name occurred because the `szColName` buffer was not large enough. If you are writing an application for use with the ODBC driver, consider making the buffer equal to or greater than the value contained in argument `pcbColName`.

**SQL_ERROR (-1) X'FFFF'**

**Explanation:** An error occurred in SQLDescribeCol.

**Problem Determination:** Review any additional information associated with the error. Driver-specific SQLSTATEs are:

- **24000** Invalid cursor state.

  Ensure that your application is executing the SQL statement in the proper sequence, or contact support personnel for your ODBC-enabled workstation application.

**IDBPE Programming error msgtext.**

The Tivoli Information Management for z/OS ODBC driver encountered an unexpected error. The function that encountered the error is identified in `msgtext`. Record the error and retry your application. If the problem persists and if the Tivoli Information Management for z/OS ODBC driver tracing is enabled, check the trace output located in the output destination specified by the `ICLUI_TRACETO` environment variable for the possible cause. Contact Tivoli Customer Support for further assistance if appropriate.

- **S1002** Invalid column number.

  Specify a valid `icol`. You can use `SQLNumResultCols` to determine the number of columns in the result set.

**SQL_INVALID_HANDLE (-2) X'FFFFE'**

**Explanation:** A statement handle that is not valid was encountered and SQLDescribeCol failed.

**Problem Determination:** A programming error occurred. Contact Tivoli Customer Support.
SQLDescribeParam

SQL_SUCCESS (0) X'0000'

Explanation: SQLDescribeParam completed successfully.

SQL_ERROR (-1) X'FFFF'

Explanation: An error occurred in SQLDescribeParam.

Problem Determination: Review any additional information associated with the error. Driver-specific SQLSTATEs are:

S1001 Memory allocation failure.

Try closing any unnecessary windows. If the problem persists, restart your system and try the function again. In addition, turning off the ODBC driver tracing and HLAPI/NT logging may improve your resource usage.

S1093 Invalid Parameter Number

Specify a valid ipar. You can use SQLNumParams to determine the number of parameters in the SQL statement. If necessary, correct the SELECT statement and reissue the statement.

IDBPE Programming error msgtext.

The Tivoli Information Management for z/OS ODBC driver encountered an unexpected error. The function that encountered the error is identified in msgtext. Record the error and retry your application. If the problem persists and if the Tivoli Information Management for z/OS ODBC driver tracing is enabled, check the trace output located in the output destination specified by the ICLUI_TRACETO environment variable for the possible cause. Contact Tivoli Customer Support for further assistance if appropriate.

SQL_INVALID_HANDLE (-2) X'FFFE'

Explanation: A statement handle that is not valid was encountered and SQLDescribeParam failed.

SQLDisconnect

**SQL_SUCCESS (0) X'0000'**

**Explanation:** SQLDisconnect completed successfully.

**SQL_ERROR (-1) X'FFFF'**

**Explanation:** An error occurred in SQLDisconnect.

**Problem Determination:** Review any additional information associated with the error. Driver-specific SQLSTATEs are:

**IDBPE** Programming error msgtext.

The Tivoli Information Management for z/OS ODBC driver encountered an unexpected error. The function that encountered the error is identified in msgtext. Record the error and retry your application. If the problem persists and if the Tivoli Information Management for z/OS ODBC driver tracing is enabled, check the trace output located in the output destination specified by the ICLUI_TRACETO environment variable for the possible cause. Contact Tivoli Customer Support for further assistance if appropriate.

**SQL_INVALID_HANDLE (-2) X'FFFE'**

**Explanation:** A connection handle that is not valid was encountered and SQLDisconnect failed.

**Problem Determination:** A programming error occurred. Contact Tivoli Customer Support.
SQLDriverConnect

SQL_SUCCESS (0) X'0000'
Explanation: SQLDriverConnect completed successfully.

SQL_SUCCESS_WITH_INFO (1) X'0001'
Explanation: SQLDriverConnect completed successfully, but an error may have occurred.
Problem Determination: Correct the error as described by the SQLSTATE and the associated error information. Driver-specific SQLSTATEs are:
01004 Data truncated.
   Truncation of the connection string occurred because the buffer was not large enough. If you are writing an application for use with the ODBC driver, consider making the buffer equal to or greater than the value contained in argument pcbConnStrOut.
01S00 Invalid connection string attribute.
   The Tivoli Information Management for z/OS ODBC driver connected to the data source, even though an error existed in the connection string.

SQL_NO_DATA_FOUND X'0064'
Explanation: The dialog box presented by the SQLDriverConnect function was cancelled. The information required to connect is not available to the Tivoli Information Management for z/OS ODBC driver.
Problem Determination: Repeat the function you were performing, enter the required information in the dialog box, and press OK.

SQL_ERROR (-1) X'FFFFFF'
Explanation: An error occurred in SQLDriverConnect.
Problem Determination: Review any additional information associated with the error. Driver-specific SQLSTATEs are:
08001 Unable to connect to data source.
   A connection did not occur because all the information needed to perform the connection was not provided. Refer to the Tivoli Information Management for z/OS Client Installation and User's Guide for an explanation of the return and reason code as listed in the text of the message IDB01.
IDB01 HICARETC=rc, HICAREAS=rs, msgtext.
   The HLAPI/NT issued a HLAPI return code rc and HLAPI reason code rs. The message text is based on the rc and rs values.
   Call SQLError to see if there is additional error information. Review the msgtext and any additional error information. Refer to the Tivoli Information Management for z/OS Client Installation and User's Guide for an explanation of the HLAPI return and reason codes (rc and rs) and take the appropriate action.
IDBPE Programming error msgtext.
   The Tivoli Information Management for z/OS ODBC driver encountered an unexpected error. The function that encountered the error is identified in msgtext. Record the error and retry your application. If the problem persists and if the Tivoli Information Management for z/OS ODBC driver tracing is enabled, check the trace output located in the output destination specified by the ICLUI_TRACETO environment variable for the possible cause. Contact Tivoli Customer Support for further assistance if appropriate.
S1001 Memory allocation failure.
Try closing any unnecessary windows. If the problem persists, restart your system and try the function again. In addition, turning off the ODBC driver tracing and HLAPI/NT logging may improve your resource usage.

SQL_INVALID_HANDLE (-2) X'FFFFFF'

**Explanation:** A connection handle that is not valid was encountered and SQLDriverConnect failed.

**Problem Determination:** A programming error occurred. Contact Tivoli Customer Support.
**SQLException**

**SQL_SUCCESS (0) X'0000'**

Explanation: SQLError completed successfully.

**SQL_SUCCESS_WITH_INFO (1) X'0001'**

Explanation: SQLError completed successfully, but the buffer for the error message was too short.

Problem Determination: The error message was truncated. To determine that truncation occurred, the application can compare `cbErrorMsgMax` to the actual length of the message text written to `pcbErrorMsg`.

**SQL_NO_DATA_FOUND (100) X'0064'**

Explanation: There is no additional error information to retrieve.

**SQL_ERROR (-1) X'FFFF'**

Explanation: An error occurred in SQLError.

Problem Determination: A programming error occurred. Retry your application. If the problem persists, contact Tivoli Customer Support for further assistance.

**SQL_INVALID_HANDLE (-2) X'FFFE'**

Explanation: SQLError failed due to encountering a connection, environment, or statement handle that is not valid.

Problem Determination: A programming error occurred. Contact Tivoli Customer support.
SQLExecDirect

SQL_SUCCESS (0) 'X'0000'  
**Explanation:** SQLExecDirect completed successfully.

SQL_SUCCESS_WITH_INFO (1) 'X'0001'  
**Explanation:** SQLExecDirect completed successfully, but an error may have occurred.  
**Problem Determination:** Review any additional information associated with the message. Driver-specific SQLSTATEs are:  
IDB02 HICARETC=rc, HICAREAS=rs, msgtext.  
The HLAPI/NT issued a return code rc and HLAPI reason code rs. The message text is based on the rc and rs values. Review the msgtext and any additional error information. Refer to the [Tivoli Information Management for z/OS Client Installation and User's Guide](#) for an explanation of the HLAPI return and reason codes. If the rc is 12 and the rs is 35, there are records in the Tivoli Information Management for z/OS database that have duplicate record IDs. Duplicate records are skipped and are not included in the result set.

SQL_ERROR (-1) 'X'FFFF'  
**Explanation:** An error occurred in SQLExecDirect.  
**Problem Determination:** Review any additional information associated with the error. Driver-specific SQLSTATEs are:  
08S01 Communication link failure.  
The HLAPI/NT connection failed. Call SQLError for additional error information. If the HLAPI/NT requester is not running, start it and restart your application. If the Tivoli Information Management for z/OS server on MVS (RES, MRES with APPC, or MRES with TCP/IP) is not active, start it and restart your application. Check for any errors that the HLAPI/NT may have written to the IDBPROBE.LOG file.  
IDB02 HICARETC=rc, HICAREAS=rs, msgtext.  
The HLAPI/NT issued a HLAPI return code rc and HLAPI reason code rs. The message text is based on the rc and rs values.  
Call SQLError to see if there is additional error information. Review the msgtext and any additional error information. Refer to the [Tivoli Information Management for z/OS Client Installation and User's Guide](#) for an explanation of the HLAPI return and reason codes (rc and rs) and take the appropriate action. If you have to restart the HLAPI/NT requester or the Tivoli Information Management for z/OS server (RES, MRES with APPC, or MRES with TCP/IP), you will need to restart your application.  
IDBE0 NULL/NOT NULL clause can only be specified for a column that is a prefixed field in Tivoli Information Management for z/OS.  
A column name you specified is not a prefixed field. Do not use a NULL or NOT NULL search-condition for that column.  
IDBE1 Null value cannot be specified in a NOT, <, >, =, NOT >, or <= WHERE clause.  
The value you specified for a column in one of these search-conditions was null. You cannot specify a null value for any of these search-conditions.  
IDBE2 A WHERE column is not searchable.  
A column name you specified in a WHERE search-condition is not searchable. The field is not cognized in the Tivoli Information Management for z/OS database. Do not specify the column in a search-condition.  
IDBMS msgtext.
The HLAPI issued a message that was placed on the message PDB chain. The msgtext is the text of the Tivoli Information Management for z/OS message. IDBMS is always preceded by IDBO2. Review the msgtext in IDBMS and refer to Tivoli Information Management for z/OS Messages and Codes for an explanation. Also, review the HLAPI return and reason codes (rc and rs) in the IDBO2 error preceding the IDBMS error and refer to the Tivoli Information Management for z/OS Client Installation and User’s Guide for an explanation of the rc and rs.

**IDBPE**  Programming error msgtext.

The Tivoli Information Management for z/OS ODBC driver encountered an unexpected error. The function that encountered the error is identified in msgtext. Record the error and retry your application. If the problem persists and if the Tivoli Information Management for z/OS ODBC driver tracing is enabled, check the trace output located in the output destination specified by the ICLU\_TRACETO environment variable for the possible cause. Contact Tivoli Customer Support for further assistance if appropriate.

**IDBS0**  SELECT statement syntax is not valid or syntax is not supported by Tivoli Information Management for z/OS ODBC driver.

The argument szSqlStr contained an SQL statement that contained a syntax error or syntax that is not supported by the Tivoli Information Management for z/OS ODBC driver. Other messages may further identify the error. Review the argument in szSqlStr and the syntax supported by the Tivoli Information Management for z/OS ODBC driver and correct it as necessary. For a description of SQL syntax see Elements Used in SQL Statements on page 118.

**IDBS1**  SELECT syntax error – WHERE comparison must not contain COUNT(*), 2 column names, 2 character string literals, a character string literal and a numeric literal, a parameter and a numeric literal or a column name and a numeric literal.

The argument szSqlStr contained an SQL statement that contained a WHERE clause not supported by the Tivoli Information Management for z/OS ODBC driver. Review the argument in szSqlStr and correct it as necessary. For a description of SQL syntax see Elements Used in SQL Statements on page 118.

**IDBS2**  SELECT syntax error – BETWEEN must start with a column name, a numeric literal, or a parameter.

The argument szSqlStr contained an SQL statement that has a BETWEEN clause not supported by the Tivoli Information Management for z/OS ODBC driver. The first expression in a BETWEEN clause must be either a column name, numeric literal, or a parameter (indicated by a ? character). Review the argument in szSqlStr and correct it as necessary. For a description of SQL syntax see Elements Used in SQL Statements on page 118.

**IDBS3**  SELECT syntax error – BETWEEN value must be a character string literal, a numeric literal, or a parameter.

The argument szSqlStr contained an SQL statement that has a BETWEEN clause not supported by the Tivoli Information Management for z/OS ODBC driver. The second and third expressions in a BETWEEN clause must be a quoted string or a parameter (indicated by a ? character). Review the argument in szSqlStr and correct it as necessary. For a description of SQL syntax see Elements Used in SQL Statements on page 118.

**IDBS4**  SELECT syntax error – SELECT column names cannot be literal strings or parameters.

The argument szSqlStr contained an SQL statement that has syntax not supported by the Tivoli Information Management for z/OS ODBC driver. SELECT field names cannot be quoted strings or parameters (indicated by a ? character). Review the argument in szSqlStr and correct it as necessary. For a description of SQL syntax see Elements Used in SQL Statements on page 118.

**IDBS5**  SELECT syntax error – NULL/NOT NULL clause must contain a column name or parameter.

The argument szSqlStr contained an SQL statement that has a LIKE clause not supported by the Tivoli Information Management for z/OS ODBC driver. A LIKE clause must contain one literal string denoting the search argument data or parameter denoting the search argument data or field name. Review the argument in szSqlStr and correct it as necessary. For a description of SQL syntax see Elements Used in SQL Statements on page 118.

**IDBS6**  SELECT syntax error – NULL/NOT NULL clause must contain a column name or parameter.
The argument szSqlStr contained an SQL statement that has a NULL or NOT NULL clause that did not contain a column name or parameter. NULL and NOT NULL clauses must contain a column name or parameter. Review the argument in szSQLStr and correct it as necessary. For a description of SQL syntax see "Elements Used in SQL Statements" on page 118.

IDBS7
SELECT syntax error – ‘<’ and ‘>’ comparisons are not supported.
The argument szSqlStr contained an SQL statement that has syntax not supported by the Tivoli Information Management for z/OS ODBC driver. Comparisons in WHERE clauses cannot be ‘<’ (less than) or ‘>’ greater than. Review the argument in szSqlStr and correct it as necessary. For a description of SQL syntax see "Elements Used in SQL Statements" on page 118.

IDBS8
SELECT syntax error – WHERE comparison cannot have two parameters.
The argument szSqlStr contained an SQL statement that has syntax not supported by the Tivoli Information Management for z/OS ODBC driver. A WHERE clause cannot consist of two parameters (indicated by a ? character). Review the argument in szSqlStr and correct it as necessary. For a description of SQL syntax see "Elements Used in SQL Statements" on page 118.

IDBS9
SELECT syntax error – 1st and 2nd or 1st and 3rd BETWEEN operands cannot both be parameters.
The argument szSqlStr contained an SQL statement that has syntax not supported by the Tivoli Information Management for z/OS ODBC driver. Review the argument in szSqlStr and correct it as necessary. For a description of SQL syntax see "Elements Used in SQL Statements" on page 118.

IDBSA
SELECT syntax error – BETWEEN target and values must not mix character string and numeric literals or parameters and numeric literals.
The argument szSqlStr contained an SQL statement that has syntax not supported by the Tivoli Information Management for z/OS ODBC driver. Review the argument in szSqlStr and correct it as necessary. For a description of SQL syntax see "Elements Used in SQL Statements" on page 118.

IDBSB
SELECT syntax error – BETWEEN values cannot be numeric literals when BETWEEN target is a column name.
The argument szSqlStr contained an SQL statement that has syntax not supported by the Tivoli Information Management for z/OS ODBC driver. Review the argument in szSqlStr and correct it as necessary. For a description of SQL syntax see "Elements Used in SQL Statements" on page 118.

IDBSC
SELECT syntax error – only one SELECT column table name can be specified and it must match the FROM table name or correlation name.
The argument szSqlStr contained an SQL statement that has syntax not supported by the Tivoli Information Management for z/OS ODBC driver. Review the argument in szSqlStr and correct it as necessary. For a description of SQL syntax see "Elements Used in SQL Statements" on page 118.

S0002
Base table not found.
Specify a valid table name. You can use SQLTables to get a list of valid tables.

S0022
Column not found.
Specify valid column names in your select-list and in your WHERE search-conditions. You can use SQLColumns to get a list of valid columns for the table. If you have access to Tivoli Information Management for z/OS, you can also view the data view record (table name) and data attribute records and their ODBC column names.

S1001
Memory allocation failure.
Try closing any unnecessary windows. If the problem persists, restart your system and try the function again. In addition, turning off the ODBC driver tracing and HLAPI/NT logging may improve your resource usage.
**SQL_INVALID_HANDLE (-2) X'FFFE'**

**Explanation:** A statement handle that is not valid was encountered and SQLExecDirect failed.

**Problem Determination:** A programming error occurred. Contact Tivoli Customer Support.

---

**SQL_NEED_DATA (99) X'0063'**

**Explanation:** Data is needed for statement parameters.

**Problem Determination:** The application needs to send the parameter data through SQLParamData and SQLPutData.
SQLExecute

**SQL_SUCCESS (0) X'0000'**

*Explanation:* SQLExecute completed successfully.

**SQL_SUCCESS_WITH_INFO (1) X'0001'**

*Explanation:* SQLExecute completed successfully, but an error may have occurred.

*Problem Determination:* Review any additional information associated with the message. Driver-specific SQLSTATEs are:

- **IDB02**
  
  The HLAPI/NT issued a return code `rc` and HLAPI reason code `rs`. The message text is based on the `rc` and `rs` values. Review the `msgtext` and any additional error information. Refer to the [Tivoli Information Management for z/OS Client Installation and User's Guide](#) for an explanation of the HLAPI return and reason codes. If the `rc` is 12 and the `rs` is 35, there are records in the Tivoli Information Management for z/OS database that have duplicate record IDs. Duplicate records are skipped and are not included in the result set.

**SQL_ERROR (-1) X'FFFF'**

*Explanation:* An error occurred in SQLExecute.

*Problem Determination:* Review any additional information associated with the error. Driver-specific SQLSTATEs are:

- **08S01** Communication link failure.
  
  The HLAPI/NT connection failed. Call SQLError for additional error information. If the HLAPI/NT requester is not running, start it and restart your application. If the Tivoli Information Management for z/OS server on MVS (RES, MRES with APPC, or MRES with TCP/IP) is not active, start it and restart your application. Check for any errors that the HLAPI/NT may have written to the IDBPROBE.LOG file.

- **IDB02**
  
  The HLAPI/NT issued a HLAPI return code `rc` and HLAPI reason code `rs`. The message text is based on the `rc` and `rs` values.

  Call SQLError to see if there is additional error information. Review the `msgtext` and any additional error information. Refer to the [Tivoli Information Management for z/OS Client Installation and User's Guide](#) for an explanation of the HLAPI return and reason codes `(rc` and `rs`) and take the appropriate action. If you have to restart the HLAPI/NT requester or the Tivoli Information Management for z/OS server (RES, MRES with APPC, or MRES with TCP/IP), restart your application.

- **IDBE0**

  NULL/NOT NULL clause can only be specified for a column that is a prefixed field in Tivoli Information Management for z/OS.

  A column name you specified in a NULL or NOT NULL clause is not a prefixed field. Do not use a NULL or NOT NULL *search-condition* for that column.

- **IDBE1**

  Null value cannot be specified in a NOT, <, >, =, NOT >, or <= WHERE clause.

  The value you specified for a column in one of these *search-conditions* was null. You cannot specify a null value for any of these *search-conditions*.

- **IDBMS**

  The HLAPI issued a message that was placed on the message PDB chain. The `msgtext` is the text of the Tivoli Information Management for z/OS message. IDBMS is always preceded by IDB02. Review the `msgtext` in IDBMS and refer to the [Tivoli Information Management for z/OS Messages and Codes](#) for an explanation. Also, review the HLAPI return and reason codes `(rc` and `rs`) in the IDB02 error preceding the IDBMS error and refer to the [Tivoli Information Management for z/OS Client Installation and User's Guide](#) for an explanation of the `rc` and `rs`. 

---

*Data Reporting User’s Guide* 73
Return and Reason Codes by Function

**IDBPE**  Programming error `msgtext`.

The Tivoli Information Management for z/OS ODBC driver encountered an unexpected error. The function that encountered the error is identified in `msgtext`. Record the error and retry your application. If the problem persists, contact Tivoli Customer Support for further assistance.

**IDBS1**  SELECT syntax error – WHERE comparison must not contain COUNT(*), 2 column names, 2 character string literals, a character string literal and a numeric literal, a parameter and a numeric literal or a column name and a numeric literal.

The argument `szSqlStr` contained an SQL statement that contained a WHERE clause not supported by the Tivoli Information Management for z/OS ODBC driver. The WHERE comparison contained a numeric literal and both operands in the WHERE clause were not numeric literals. Review the argument in `szSqlStr` and correct it as necessary. For a description of SQL syntax see [“Elements Used in SQL Statements”](#) on page 118.

**IDBSA**  SELECT syntax error – BETWEEN target and values must not mix character string and numeric literals or parameters and numeric literals.

The argument `szSqlStr` contained an SQL statement that has syntax not supported by the Tivoli Information Management for z/OS ODBC driver. Review the argument in `szSqlStr` and correct it as necessary. For a description of SQL syntax see [“Elements Used in SQL Statements”](#) on page 118.

**S1001**  Memory allocation failure.

Try closing any unnecessary windows. If the problem persists, restart your system and try the function again. In addition, turning off the ODBC driver tracing and HLAPI/NT logging may improve your resource usage.

---

**SQL_INVALID_HANDLE (-2) X'FFFE'**

**Explanation:** A statement handle that is not valid was encountered and SQLExecute failed.

**Problem Determination:** A programming error occurred. Contact Tivoli Customer Support.

---

**SQL_NEED_DATA (99) X'0063'**

**Explanation:** Data is needed for statement parameters.

**Problem Determination:** The application needs to send the parameter data through SQLParamData and SQLPutData.
SQLExtendedFetch

 SQL_SUCCESS (0) X'0000'
 Explanation: SQLExtendedFetch completed successfully.

 SQL_SUCCESS_WITH_INFO (1) X'0001'
 Explanation: SQLExtendedFetch completed successfully, but an error may have occurred.
 Problem Determination: Review any additional information associated with the message. Driver-specific SQLSTATEs are:
 01004  Data truncated.

  The application can compare pcbValue to cbValueMax (specified in SQLBindCol) to determine which column or columns were truncated. If pcbValue is greater than or equal to cbValueMax, then truncation occurred. Rebind the column with SQLBindCol and use a larger buffer. If you are writing an application for use with the ODBC driver, consider making the buffer equal to or greater than the value contained in argument cbValueMax.

 01S01  Error in row.

  Review the contents of the status array values to determine the row in error. Review the additional error information for the cause of the row error and correct the problem as appropriate.

 SQL_NO_DATA_FOUND (100) X'0064'
 Explanation: There is no additional data to retrieve.

 SQL_ERROR (-1) X'FFFF'
 Explanation: An error occurred in SQLExtendedFetch.
 Problem Determination: Review any additional information associated with the error. Driver-specific SQLSTATEs are:
 07006  Restricted data type attribute violation.

  The fCType specified on SQLBindCol was not supported by the Tivoli Information Management for z/OS ODBC driver. You must specify a valid fCType.

 22003  Numeric value out of range.

  The length of the data is greater than the maximum length of the buffer to store the output data. Rebind the column with SQLBindCol and increase the size of the data buffer.

 22005  Error in assignment.

  The data type of a column was character and the fCType specified on SQLBindCol was not supported by the Tivoli Information Management for z/OS ODBC driver. To return columns of character data, you must specify one of the following fCTypes on SQLBindCol:

  SQL_C_CHAR
  SQL_C_DEFAULT
  SQL_C_BINARY

  Note: All data in a Tivoli Information Management for z/OS result set is character.

 22008  Datetime field overflow.

  The fCType specified on SQLBindCol was not supported by the Tivoli Information Management for z/OS ODBC driver. To return columns of character data, you must specify one of the following fCTypes on SQLBindCol:
Note: All data in a Tivoli Information Management for z/OS result set is character.

24000 Invalid cursor state.

Ensure that your application is executing the SQL statement in the proper sequence, or contact support personnel for your ODBC-enabled workstation application.

IDBPE Programming error msgtext.

The Tivoli Information Management for z/OS ODBC driver encountered an unexpected error. The function that encountered the error is identified in msgtext. Record the error and retry your application. If the problem persists and if the Tivoli Information Management for z/OS ODBC driver tracing is enabled, check the trace output located in the output destination specified by the ICLUI_TRACETO environment variable for the possible cause. Contact Tivoli Customer Support for further assistance if appropriate.

S1002 Invalid column number.

Use SQLNumResultCols to determine the number of columns in the result set. Use SQLBindCol to bind or unbind the columns in error.

S1C00 Driver not capable.

This error only applies to Integer and Smallint data type columns in non-Tivoli Information Management for z/OS result sets. If the column’s data type is Integer, specify an fCType from the following list of supported values:

SQL_C_DEFAULT
SQL_C_LONG
SQL_C_SLONG
SQL_C_FLOAT
SQL_C_DOUBLE
SQL_C_BINARY
SQL_C_CHAR

If the column’s data type is Smallint, specify an fCType from the following list of supported values:

SQL_C_DEFAULT
SQL_C_SHORT
SQL_C_SSHORT
SQL_C_LONG
SQL_C_SLONG
SQL_C_FLOAT
SQL_C_DOUBLE
SQL_C_BINARY
SQL_C_CHAR

SQL_INVALID_HANDLE (-2) X'FFE'

Explanation: A statement handle that is not valid was encountered and SQLExtendedFetch failed.

### SQLFetch

**SQL_SUCCESS (0) X'0000'**

**Explanation:** SQLFetch completed successfully.

**SQL_SUCCESS_WITH_INFO (1) X'0001'**

**Explanation:** SQLFetch completed successfully, but an error may have occurred.

**Problem Determination:** Review any additional information associated with the message. Driver-specific SQLSTATEs are:

- **01004** Data truncated.
  
  The application can compare `pcbValue` to `cbValueMax` (specified in SQLBindCol) to determine which column or columns were truncated. If `pcbValue` is greater than or equal to `cbValueMax`, then truncation occurred. Rebind the column with SQLBindCol and use a larger buffer. If you are writing an application for use with the ODBC driver, consider making the buffer equal to or greater than the value contained in argument `cbValueMax`.

**SQL_NO_DATA_FOUND (100) X'0064'**

**Explanation:** There is no additional data to retrieve.

**SQL_ERROR (-1) X'FFFF'**

**Explanation:** An error occurred in SQLFetch.

**Problem Determination:** Review any additional information associated with the error. Driver-specific SQLSTATEs are:

- **07006** Restricted data type attribute violation.
  
  The `fCType` specified on SQLBindCol was not supported by the Tivoli Information Management for z/OS ODBC driver. You must specify a valid `fCType`.

- **22003** Numeric value out of range.
  
  The length of the data is greater than the maximum length of the buffer to store the output data. Rebind the column with SQLBindCol and increase the size of the data buffer.

- **22005** Error in assignment.
  
  The data type of a column was character and the `fCType` specified on SQLBindCol was not supported by the Tivoli Information Management for z/OS ODBC driver. To return columns of character data, you must specify one of the following `fCTypes` on SQLBindCol:

```
SQL_C_CHAR
SQL_C_DEFAULT
SQL_C_BINARY
```

**Note:** All data in a Tivoli Information Management for z/OS result set is character.

- **22008** Datetime field overflow.
  
  The `fCType` specified on SQLBindCol was not supported by the Tivoli Information Management for z/OS ODBC driver. To return columns of character data, you must specify one of the following `fCTypes` on SQLBindCol:

```
SQL_C_CHAR
SQL_C_DEFAULT
SQL_C_BINARY
```

**Note:** All data in a Tivoli Information Management for z/OS result set is character.
24000  Invalid cursor state.

Ensure that your application is executing the SQL statement in the proper sequence, or contact support personnel for your ODBC-enabled workstation application.

IDBPE  Programming error msgtext.

The Tivoli Information Management for z/OS ODBC driver encountered an unexpected error. The function that encountered the error is identified in msgtext. Record the error and retry your application. If the problem persists and if the Tivoli Information Management for z/OS ODBC driver tracing is enabled, check the trace output located in the output destination specified by the ICLUI_TRACETO environment variable for the possible cause. Contact Tivoli Customer Support for further assistance if appropriate.

S1002  Invalid column number.

Use SQLNumResultCols to determine the number of columns in the result set. Use SQLBindCol to bind or unbind the columns in error.

S1C00  Driver not capable.

This error only applies to Integer and Smallint data type columns in non-Tivoli Information Management for z/OS result sets. If the column’s data type is Integer, specify an fCType from the following list of supported values:

- SQL_C_DEFAULT
- SQL_C_LONG
- SQL_C_SLONG
- SQL_C_FLOAT
- SQL_C_DOUBLE
- SQL_C_BINARY
- SQL_C_CHAR

If the column’s data type is Smallint, specify an fCType from the following list of supported values:

- SQL_C_DEFAULT
- SQL_C_SHORT
- SQL_C_SSHORT
- SQL_C_LONG
- SQL_C_SLONG
- SQL_C_FLOAT
- SQL_C_DOUBLE
- SQL_C_BINARY
- SQL_C_CHAR

SQL_INVALID_HANDLE (-2) X'FFFE'

Explanation:  A statement handle that is not valid was encountered and SQLFetch failed.

## SQLFreeConnect

### SQL_SUCCESS (0) X'0000'

**Explanation:** SQLFreeConnect completed successfully.

### SQL_ERROR (-1) X'FFFF'

**Explanation:** An error occurred in SQLFreeConnect.

**Problem Determination:** Review any additional information associated with the error. Driver-specific SQLSTATEs are:

- **IDBPE**: Programming error `msgtext`.

  The Tivoli Information Management for z/OS ODBC driver encountered an unexpected error. The function that encountered the error is identified in `msgtext`. Record the error and retry your application. If the problem persists and if the Tivoli Information Management for z/OS ODBC driver tracing is enabled, check the trace output located in the output destination specified by the ICLUI_TRACETO environment variable for the possible cause. Contact Tivoli Customer Support for further assistance if appropriate.

### SQL_INVALID_HANDLE (-2) X'FFFF'

**Explanation:** A connection handle that is not valid was encountered and SQLFreeConnect failed.

**Problem Determination:** A programming error occurred. Contact Tivoli Customer Support.
SQLFreeEnv

SQL_SUCCESS (0) X'0000'

Explanation: SQLFreeEnv completed successfully.

SQL_ERROR (-1) X'FFFF'

Explanation: An error occurred in SQLFreeEnv.

Problem Determination: Review any additional information associated with the error. Driver-specific SQLSTATEs are:

IDBPE Programming error msgtext.

The Tivoli Information Management for z/OS ODBC driver encountered an unexpected error. The function that encountered the error is identified in msgtext. Record the error and retry your application. If the problem persists and if the Tivoli Information Management for z/OS ODBC driver tracing is enabled, check the trace output located in the output destination specified by the ICLUI_TRACETO environment variable for the possible cause. Contact Tivoli Customer Support for further assistance if appropriate.

SQL_INVALID_HANDLE (-2) X'FFFE'

Explanation: An environment handle that is not valid was encountered and SQLFreeEnv failed.

**SQLFreeStmt**

**SQL_SUCCESS (0) X'0000'**

*Explanation:* SQLFreeStmt completed successfully.

**SQL_ERROR (-1) X'FFFF'**

*Explanation:* An error occurred in SQLFreeStmt.

*Problem Determination:* Review any additional information associated with the error. Driver-specific SQLSTATEs are:

**IDBPE** Programming error msgtext.

The Tivoli Information Management for z/OS ODBC driver encountered an unexpected error. The function that encountered the error is identified in msgtext. Record the error and retry your application. If the problem persists and if the Tivoli Information Management for z/OS ODBC driver tracing is enabled, check the trace output located in the output destination specified by the ICLUI_TRACETO environment variable for the possible cause. Contact Tivoli Customer Support for further assistance if appropriate.

**S0192** Option type out of range.

If the component-identifier is 'Tivoli Information Management for z/OS ODBC Driver', a programming error exists. This error should be handled by the ODBC driver manager. Contact Tivoli Customer Support.

**SQL_INVALID_HANDLE (-2) X'FFFE'**

*Explanation:* A statement handle that is not valid was encountered and SQLFreeStmt failed.

*Problem Determination:* A programming error occurred. Contact Tivoli Customer Support.
**SQLGetConnectOption**

**SQL_SUCCESS (0) X'0000'**

**Explanation:** SQLGetConnectOption completed successfully.

---

**SQL_ERROR (-1) X'FFFFF'**

**Explanation:** An error occurred in SQLGetConnectOption.

**Problem Determination:** Review any additional information associated with the error. Driver-specific SQLSTATEs are:

**IDBPE** Programming error msgtext.

The Tivoli Information Management for z/OS ODBC driver encountered an unexpected error. The function that encountered the error is identified in msgtext. Record the error and retry your application. If the problem persists and if the Tivoli Information Management for z/OS ODBC driver tracing is enabled, check the trace output located in the output destination specified by the ICLUI_TRACETO environment variable for the possible cause. Contact Tivoli Customer Support for further assistance if appropriate.

**S1C00** Driver not capable.

Specify an fOption from the list of supported values:

- SQL_ACCESS_MODE
- SQL_AUTOCOMMIT

---

**SQL_INVALID_HANDLE (-2) X'FFFE'**

**Explanation:** A connection handle that is not valid was encountered and SQLGetConnectOption failed.

**Problem Determination:** A programming error occurred. Contact Tivoli Customer Support.
SQLGetCursorName

SQL_SUCCESS (0) 'X'0000'

Explanation: SQLGetCursorName completed successfully.

SQL_ERROR (-1) 'X'FFFF'

Explanation: An error occurred in SQLGetCursorName.

Problem Determination: Review any additional information associated with the error. Driver-specific SQLSTATEs are:

IDBPE  Programming error msgtext.

The Tivoli Information Management for z/OS ODBC driver encountered an unexpected error. The function that encountered the error is identified in msgtext. Record the error and retry your application. If the problem persists and if the Tivoli Information Management for z/OS ODBC driver tracing is enabled, check the trace output located in the output destination specified by the ICLUI_TRACETO environment variable for the possible cause. Contact Tivoli Customer Support for further assistance if appropriate.

IM001  Driver does not support this function.

Cursors are not supported by the Tivoli Information Management for z/OS ODBC driver since it does not support positioned updates or deletes. Do not use this function.

SQL_INVALID_HANDLE (-2) 'X'FFFE'

Explanation: A statement handle that is not valid was encountered and SQLGetCursorName failed.

SQLGetData

SQL_SUCCESS (0) X'0000'

Explanation: SQLGetData completed successfully.

SQL_SUCCESS_WITH_INFO (1) X'0001'

Explanation: SQLGetData completed successfully, but an error may have occurred.

Problem Determination: Review any additional information associated with the message. Driver-specific SQLSTATEs are:

- 01004  Data truncated.
  
  Because not all the data was retrieved in a single call, it was truncated. Your application can use the same column number, icol, to retrieve subsequent parts of the data until SQLGetData returns SQL_SUCCESS (0), indicating that all data for the column has been retrieved. SQLGetData returns SQL_NO_DATA_FOUND (100) when it is called for a column after all the data has been retrieved, and before data is retrieved for a subsequent column. The application can ignore excess data by proceeding to the next result column.

SQL_ERROR (-1) X'FFFF'

Explanation: An error occurred in SQLGetData.

Problem Determination: Review any additional information associated with the error. Driver-specific SQLSTATEs are:

- 07006  Restricted data type attribute violation.
  
  The fCType specified on SQLBindCol was not supported by the Tivoli Information Management for z/OS ODBC driver. You must specify a valid fCType.

- 22003  Numeric value out of range.
  
  The length of the data is greater than the maximum length of the buffer to store the output data. Rebind the column with SQLBindCol and increase the size of the data buffer.

- 22005  Error in assignment.
  
  The data type of a column was character and the fCType specified on SQLBindCol was not supported by the Tivoli Information Management for z/OS ODBC driver. To return columns of character data, you must specify one of the following fCTypes on SQLBindCol:

  SQL_C_CHAR
  SQL_C_DEFAULT
  SQL_C_BINARY

  Note: All data in a Tivoli Information Management for z/OS result set is character.

- 22008  Datetime field overflow.
  
  The fCType specified on SQLBindCol was not supported by the Tivoli Information Management for z/OS ODBC driver. To return columns of character data, you must specify one of the following fCTypes on SQLBindCol:

  SQL_C_CHAR
  SQL_C_DEFAULT
  SQL_C_BINARY

  Note: All data in a Tivoli Information Management for z/OS result set is character.

- 24000  Invalid cursor state.
Ensure that your application is executing the SQL statement in the proper sequence, or contact support personnel for your ODBC-enabled workstation application.

**IDBPE**  Programming error *msgtext*.

The Tivoli Information Management for z/OS ODBC driver encountered an unexpected error. The function that encountered the error is identified in *msgtext*. Record the error and retry your application. If the problem persists and if the Tivoli Information Management for z/OS ODBC driver tracing is enabled, check the trace output located in the output destination specified by the ICLUI_TRACETO environment variable for the possible cause. Contact Tivoli Customer Support for further assistance if appropriate.

**S1002**  Invalid column number.

Use SQLNumResultCols to determine the number of columns in the result set. Applications can call SQLGetData only for unbound columns with numbers greater than the number of the last bound column. Also, within a single row of data, the column number in each call to SQLGetData must be greater than or equal to the column number in the previous call.

**S1C00**  Driver not capable.

This error only applies to Integer and Smallint data type columns in non-Tivoli Information Management for z/OS result sets. If the column’s data type is Integer, specify an *fCType* from the following list of supported values:

- SQL_C_DEFAULT
- SQL_C_LONG
- SQL_C_SLONG
- SQL_C_FLOAT
- SQL_C_DOUBLE
- SQL_C_BINARY
- SQL_C_CHAR

If the column data type is Smallint, specify an *fCType* from the following list of supported values:

- SQL_C_DEFAULT
- SQL_C_SHORT
- SQL_C_SSHORT
- SQL_C_LONG
- SQL_C_SLONG
- SQL_C_FLOAT
- SQL_C_DOUBLE
- SQL_C_BINARY
- SQL_C_CHAR

**SQL_INVALID_HANDLE** (-2) X'FFFE’

**Explanation:** A statement handle that is not valid was encountered and SQLGetData failed.

**Problem Determination:** A programming error occurred. Contact Tivoli Customer Support.

**SQL_NO_DATA_FOUND** (100) X'0064’

**Problem Determination:** All data for the *icol* was retrieved.
### SQLGetInfo

<table>
<thead>
<tr>
<th>Code</th>
<th>Explanation</th>
<th>Problem Determination</th>
</tr>
</thead>
<tbody>
<tr>
<td>SQL_SUCCESS (0) X'0000'</td>
<td>SQLGetInfo completed successfully.</td>
<td></td>
</tr>
<tr>
<td>SQL_SUCCESS_WITH_INFO X'0001'</td>
<td>SQLGetInfo completed successfully, but an error may have occurred.</td>
<td>Data truncated. If possible, enlarge the storage pointed to by <code>rgbInfoValue</code> in your application.</td>
</tr>
<tr>
<td>SQL_ERROR (-1) X'FFFF'</td>
<td>An error occurred in SQLGetInfo.</td>
<td>Review any additional information associated with the error. Driver-specific SQLSTATEs are:</td>
</tr>
<tr>
<td></td>
<td>IDBPE Programming error <code>msgtext</code></td>
<td>The Tivoli Information Management for z/OS ODBC driver encountered an unexpected error. The function that encountered the error is identified in <code>msgtext</code>. Record the error and retry your application. If the problem persists and if the Tivoli Information Management for z/OS ODBC driver tracing is enabled, check the trace output located in the output destination specified by the ICLUI_TRACETO environment variable for the possible cause. Contact Tivoli Customer Support if appropriate.</td>
</tr>
<tr>
<td></td>
<td>S1001 Memory allocation failure.</td>
<td>Try closing any unnecessary windows. If the problem persists, restart your system and try the function again. In addition, turning off the ODBC driver tracing and HLAPI/NT logging may improve your resource usage.</td>
</tr>
<tr>
<td></td>
<td>S1096 Information type out of range.</td>
<td>Verify that the application supports ODBC Version 2.5. The Tivoli Information Management for z/OS driver is an ODBC Version 3.51 driver. If the <code>component-identifier</code> is ‘Tivoli Information Management for z/OS ODBC Driver’, a programming error exists. This error should be handled by the ODBC driver manager. Contact Tivoli Customer Support.</td>
</tr>
<tr>
<td></td>
<td>S1C00 Driver not capable.</td>
<td>Correct or modify the application, if possible, to use only the information types that are valid for the version of ODBC supported by the driver.</td>
</tr>
<tr>
<td>SQL_INVALID_HANDLE (-2) X'FFFE'</td>
<td>A connection handle that is not valid was encountered and SQLGetInfo failed.</td>
<td>A programming error occurred. Contact Tivoli Customer Support.</td>
</tr>
</tbody>
</table>
SQLGetStmtOption

SQL_SUCCESS (0) X'0000'

Explanation: SQLGetStmtOption completed successfully.

SQL_ERROR (-1) X'FFFF'

Explanation: An error occurred in SQLGetStmtOption.

Problem Determination: Review any additional information associated with the error. Driver-specific SQLSTATEs are:

IDBPE Programming error msgtext.

The Tivoli Information Management for z/OS ODBC driver encountered an unexpected error. The function that encountered the error is identified in msgtext. Record the error and retry your application. If the problem persists and if the Tivoli Information Management for z/OS ODBC driver tracing is enabled, check the trace output located in the output destination specified by the ICLUI_TRACETO environment variable for the possible cause. Contact Tivoli Customer Support for further assistance if appropriate.

S1C00 Driver not capable.

Specify an fOption from the list of supported values:

SQL_CONCURRENCY
SQL_CURSOR_TYPE
SQL_ROW_NUMBER

SQL_INVALID_HANDLE (-2) X'FFFE'

Explanation: A statement handle that is not valid was encountered and SQLGetStmtOption failed.

SQLGetTypeInfo

SQL_SUCCESS (0) X'0000'

Explanation: SQLGetTypeInfo completed successfully.

SQL_ERROR (-1) X'FFFF'

Explanation: An error occurred in SQLGetTypeInfo.

Problem Determination: Review any additional information associated with the error. Driver-specific SQLSTATEs are:

IDBPE | Programming error msgtext.

The Tivoli Information Management for z/OS ODBC driver encountered an unexpected error. The function that encountered the error is identified in msgtext. Record the error and retry your application. If the problem persists and if the Tivoli Information Management for z/OS ODBC driver tracing is enabled, check the trace output located in the output destination specified by the ICLUI_TRACETO environment variable for the possible cause. Contact Tivoli Customer Support for further assistance if appropriate.

S1001 | Memory allocation failure.

Try closing any unnecessary windows. If the problem persists, restart your system and try the function again. In addition, turning off the ODBC driver tracing and HLAPI/NT logging may improve your resource usage.

S1C00 | Driver not capable.

Verify the argument fSqlType is supported by the driver. If it is a supported argument value, contact Tivoli Customer Support.

SQL_INVALID_HANDLE (-2) X'FFFE'

Explanation: A statement handle that is not valid was encountered and SQLGetTypeInfo failed.

**SQLNativeSql**

**SQL_SUCCESS (0) X'0000'**

Explanation: SQLNativeSql completed successfully.

**SQL_ERROR (-1) X'FFFF'**

Explanation: An error occurred in SQLNativeSql.

**Problem Determination:** Review any additional information associated with the error. Driver-specific SQLSTATEs are:

- **37000** Syntax error or access violation.
  
  The argument `szSqlStr` contained an SQL statement that contained a syntax error or syntax that is not supported by the Tivoli Information Management for z/OS ODBC driver. Review the argument and the syntax supported by the Tivoli Information Management for z/OS ODBC driver and make any necessary corrections.

- **IDBPE** Programming error `msgtext`.
  
  The Tivoli Information Management for z/OS ODBC driver encountered an unexpected error. The function that encountered the error is identified in `msgtext`. Record the error and retry your application. If the problem persists and if the Tivoli Information Management for z/OS ODBC driver tracing is enabled, check the trace output located in the output destination specified by the `ICLUI_TRACETO` environment variable for the possible cause. Contact Tivoli Customer Support for further assistance if appropriate.

- **S1001** Memory allocation failure.
  
  Try closing any unnecessary windows. If the problem persists, restart your system and try the function again. In addition, turning off the ODBC driver tracing and HLAPI/NT logging may improve your resource usage.
**SQLNumParams**

**SQL_SUCCESS (0) X'0000'**

*Explanation:* SQLNumParams completed successfully.

**SQL_ERROR (-1) X'FFFF'**

*Explanation:* An error occurred in SQLNumParams.

*Problem Determination:* Review any additional information associated with the error. Driver-specific SQLSTATEs are:

**IDBPE** Programming error msgtext.

The Tivoli Information Management for z/OS ODBC driver encountered an unexpected error. The function that encountered the error is identified in msgtext. Record the error and retry your application. If the problem persists and if the Tivoli Information Management for z/OS ODBC driver tracing is enabled, check the trace output located in the output destination specified by the ICLUI_TRACETO environment variable for the possible cause. Contact Tivoli Customer Support for further assistance if appropriate.

**SQL_INVALID_HANDLE (-2) X'FFFE'**

*Explanation:* A statement handle that is not valid was encountered and SQLNumParams failed.

*Problem Determination:* A programming error occurred. Contact Tivoli Customer Support.
### SQLNumResultCols

**SQL_SUCCESS (0) X'0000'**

**Explanation:** SQLNumResultCols completed successfully.

**SQL_ERROR (-1) X'FFFF'**

**Explanation:** An error occurred in SQLNumResultCols.

**Problem Determination:** Review any additional information associated with the error. Driver-specific SQLSTATEs are:

- `IDBPE` Programming error `msgtext`.

  The Tivoli Information Management for z/OS ODBC driver encountered an unexpected error. The function that encountered the error is identified in `msgtext`. Record the error and retry your application. If the problem persists and if the Tivoli Information Management for z/OS ODBC driver tracing is enabled, check the trace output located in the output destination specified by the `ICLUI_TRACETO` environment variable for the possible cause. Contact Tivoli Customer Support for further assistance if appropriate.

**SQL_INVALID_HANDLE (-2) X'FFFE'**

**Explanation:** A statement handle that is not valid was encountered and SQLNumResultCols failed.

**Problem Determination:** A programming error occurred. Contact Tivoli Customer Support.
SQLParamData

SQL_SUCCESS (0) X'0000'
Explanation:  SQLParamData completed successfully.

SQL_ERROR (-1) X'FFFF'
Explanation:  An error occurred in SQLParamData.

Problem Determination:  Review any additional information associated with the error. Driver-specific SQLSTATEs are:

IDBPE  Programming error msgtext.
The Tivoli Information Management for z/OS ODBC driver encountered an unexpected error. The function that encountered the error is identified in msgtext. Record the error and retry your application. If the problem persists and if the Tivoli Information Management for z/OS ODBC driver tracing is enabled, check the trace output located in the output destination specified by the ICLUI_TRACETO environment variable for the possible cause. Contact Tivoli Customer Support for further assistance if appropriate.

S1001  Memory allocation failure.
Try closing any unnecessary windows. If the problem persists, restart your system and try the function again. In addition, turning off the ODBC driver tracing and HLAPI/NT logging may improve your resource usage.

SQL_INVALID_HANDLE (-2) X'FFFF'
Explanation:  A statement handle that is not valid was encountered and SQLParamData failed.


SQL_NEED_DATA (99) X'00063'
Explanation:  Data is needed for statement parameters.

Problem Determination:  The application needs to send the parameter data through SQLPutData.

Note:  SQLParamData can also return any SQLSTATE by either the SQLExecDirect or SQLExecute function which was used to execute the SQL statement.
SQLPrepare

SQL_SUCCESS (0) X'0000'

Explanation: SQLPrepare completed successfully.

SQL_ERROR (-1) X'FFFF'

Explanation: An error occurred in SQLPrepare.

Problem Determination: Review any additional information associated with the error. Driver-specific SQLSTATEs are:

08S01 Communication link failure.

The HLAPI/NT connection failed. Call SQLError for additional error information. If the HLAPI/NT requester is not running, start it and restart your application. If the Tivoli Information Management for z/OS server on MVS (RES, MRES with APPC, or MRES with TCP/IP) is not active, start it and restart your application. Check for any errors that the HLAPI/NT may have written to the IDBPROBE.LOG file.

IDB02 HICARETC=rc HICAREAS=rs msgtext.

The HLAPI/NT issued a HLAPI return code rc and HLAPI reason code rs. The msgtext is based on the rc and rs values.

Review the msgtext. For additional information, refer to the Tivoli Information Management for z/OS Client Installation and User’s Guide for an explanation of the HLAPI return and reason codes and take the appropriate action. If you have to restart the HLAPI/NT requester or the Tivoli Information Management for z/OS server (RES, MRES with APPC, or MRES with TCP/IP), you will need to restart your application.

IDBE2 A WHERE column is not searchable.

A column name you specified in a WHERE search-condition is not searchable. The field is not recognized in the Tivoli Information Management for z/OS database. Do not specify the column in a search-condition.

IDBE5 Programming error msgtext.

The Tivoli Information Management for z/OS ODBC driver encountered an unexpected error. The function that encountered the error is identified in msgtext. Record the error and retry your application. If the problem persists and if the Tivoli Information Management for z/OS ODBC driver tracing is enabled, check the trace output located in the output destination specified by the ICLUI_TRACETO environment variable for the possible cause. Contact Tivoli Customer Support for further assistance if appropriate.

IDBS0 SELECT statement syntax is not valid or syntax is not supported by Tivoli Information Management for z/OS ODBC driver.

The argument szSqlStr contained an SQL statement that contained a syntax error or syntax that is not supported by the driver. Review the argument in szSqlStr and the syntax supported by the Tivoli Information Management for z/OS ODBC driver and correct it as necessary. For a description of SQL syntax see "Elements Used in SQL Statements" on page 118.

IDBS1 SELECT syntax error – WHERE comparison must not contain COUNT(*), 2 column names, 2 character string literals, a character string literal and a numeric literal, a parameter and a numeric literal or a column name and a numeric literal.

The argument szSqlStr contained an SQL statement that contained a WHERE clause not supported by the Tivoli Information Management for z/OS ODBC driver. Review the argument in szSqlStr and correct it as necessary. For a description of SQL syntax see "Elements Used in SQL Statements" on page 118.

IDBS2 SELECT syntax error – BETWEEN must start with a column name, a numeric literal, or a parameter.

The argument szSqlStr contained an SQL statement that has a BETWEEN clause not supported by the Tivoli Information Management for z/OS ODBC driver. The first expression in a BETWEEN clause...
must be either a column name, numeric literal, or a parameter (indicated by a ? character). Review the argument in szSqlStr and correct it as necessary. For a description of SQL syntax see "Elements Used in SQL Statements" on page 118.

IDBS3 SELECT syntax error – BETWEEN value must be a character string literal, a numeric literal, or a parameter.

The argument szSqlStr contained an SQL statement that has a BETWEEN clause not supported by the Tivoli Information Management for z/OS ODBC driver. The second and third expressions in a BETWEEN clause must be a quoted string or a parameter (indicated by a ? character). Review the argument in szSqlStr and correct it as necessary. For a description of SQL syntax see "Elements Used in SQL Statements" on page 118.

IDBS4 SELECT syntax error – SELECT column names cannot be literal strings or parameters.

The argument szSqlStr contained an SQL statement that has syntax not supported by the Tivoli Information Management for z/OS ODBC driver. SELECT field names cannot be quoted strings or parameters (indicated by a ? character). Review the argument in szSqlStr and correct it as necessary. For a description of SQL syntax see "Elements Used in SQL Statements" on page 118.

IDBS5 SELECT syntax error – LIKE data must contain 1 literal string or parameter.

The argument szSqlStr contained an SQL statement that has a LIKE clause not supported by the Tivoli Information Management for z/OS ODBC driver. A LIKE clause must contain one literal string denoting the search argument data or parameter denoting the search argument data or field name. Review the argument in szSqlStr and correct it as necessary. For a description of SQL syntax see "Elements Used in SQL Statements" on page 118.

IDBS6 SELECT syntax error – NULL/NOT NULL clause must contain a column name or parameter.

The argument szSqlStr contained an SQL statement that has a NULL or NOT NULL clause that did not contain a column name or parameter. NULL and NOT NULL clauses must contain a column name or parameter. Review the argument in szSqlStr and correct it as necessary. For a description of SQL syntax see "Elements Used in SQL Statements" on page 118.

IDBS7 SELECT syntax error – ‘<’ and ‘>’ comparisons are not supported.

The argument szSqlStr contained an SQL statement that has syntax not supported by the Tivoli Information Management for z/OS ODBC driver. Comparisons in WHERE clauses cannot be ‘<’ (less than) or ‘>’ (greater than). Review the argument in szSqlStr and correct it as necessary. For a description of SQL syntax see "Elements Used in SQL Statements" on page 118.

IDBS8 SELECT syntax error – WHERE comparison cannot have two parameters.

The argument szSqlStr contained an SQL statement that has syntax not supported by the Tivoli Information Management for z/OS ODBC driver. A WHERE clause cannot consist of two parameters (indicated by a ? character). Review the argument in szSqlStr and correct it as necessary. For a description of SQL syntax see "Elements Used in SQL Statements" on page 118.

IDBS9 SELECT syntax error – 1st and 2nd or 1st and 3rd BETWEEN operands cannot both be parameters.

The argument szSqlStr contained an SQL statement that has syntax not supported by the Tivoli Information Management for z/OS ODBC driver. Review the argument in szSqlStr and correct it as necessary. For a description of SQL syntax see "Elements Used in SQL Statements" on page 118.

IDBSA SELECT syntax error – BETWEEN target and values must not mix character string and numeric literals or parameters and numeric literals.

The argument szSqlStr contained an SQL statement that has syntax not supported by the Tivoli Information Management for z/OS ODBC driver. Review the argument in szSqlStr and correct it as necessary. For a description of SQL syntax see "Elements Used in SQL Statements" on page 118.

IDBSB SELECT syntax error – BETWEEN values cannot be numeric literals when BETWEEN target is a column name.

The argument szSqlStr contained an SQL statement that has syntax not supported by the Tivoli Information Management for z/OS ODBC driver. Review the argument in szSqlStr and correct it as necessary. For a description of SQL syntax see "Elements Used in SQL Statements" on page 118.
IDBSC SELECT syntax error – only one SELECT column table name can be specified and it must match the FROM table name or correlation name.

The argument szSqlStr contained an SQL statement that has syntax not supported by the Tivoli Information Management for z/OS ODBC driver. Review the argument in szSqlStr and correct it as necessary. For a description of SQL syntax see "Elements Used in SQL Statements" on page 118.

S0002 Base table not found.

Specify a valid table name. You can use SQLTables to get a list of valid tables.

S0022 Column not found.

Specify valid column names in your select-list and in your WHERE search-conditions. You can use SQLColumns to get a list of valid columns for the table. If you have access to Tivoli Information Management for z/OS, you can also view the data view record (table name) and data attribute records and their ODBC column names.

S1001 Memory allocation failure.

Try closing any unnecessary windows. If the problem persists, restart your system and try the function again. In addition, turning off the ODBC driver tracing and HLAPI/NT logging may improve your resource usage.

SQL_INVALID_HANDLE (-2) X'FFFE'

**Explanation:** A statement handle that is not valid was encountered and SQLPrepare failed.

**Problem Determination:** A programming error occurred. Contact Tivoli Customer Support.
SQLPutData

SQL_SUCCESS (0) X'0000'

Explanation: SQLPutData completed successfully.

SQL_ERROR (-1) X'FFFF'

Explanation: An error occurred in SQLPutData.

Problem Determination: Review any additional information associated with the error. Driver-specific SQLSTATEs are:

22003 Numeric value out of range.

Ensure that the C data type for the parameter is character or binary and that it is bound to a column with a character or binary data type.

IDBPE Programming error msgtext.

The Tivoli Information Management for z/OS ODBC driver encountered an unexpected error. The function that encountered the error is identified in msgtext. Record the error and retry your application. If the problem persists and if the Tivoli Information Management for z/OS ODBC driver tracing is enabled, check the trace output located in the output destination specified by the ICLUI_TRACETO environment variable for the possible cause. Contact Tivoli Customer Support for further assistance if appropriate.

S1001 Memory allocation failure.

Try closing any unnecessary windows. If the problem persists, restart your system and try the function again. In addition, turning off the ODBC driver tracing and HLAPI/NT logging may improve your resource usage.

S1090 Invalid string or buffer length.

Provide a valid length in the cbValue argument for the amount of data sent in the buffer.

SQL_INVALID_HANDLE (-2) X'FFFE'

Explanation: A statement handle that is not valid was encountered and SQLPutData failed.

SQLRowCount

SQL_SUCCESS (0) X'0000'

Explanation: SQLRowCount completed successfully.

SQL_ERROR (-1) X'FFFF'

Explanation: An error occurred in SQLRowCount.

Problem Determination: Review any additional information associated with the error. Driver-specific SQLSTATEs are:

IDBPE Programming error msgtext.

The Tivoli Information Management for z/OS ODBC driver encountered an unexpected error. The function that encountered the error is identified in msgtext. Record the error and retry your application. If the problem persists and if the Tivoli Information Management for z/OS ODBC driver tracing is enabled, check the trace output located in the output destination specified by the ICLUI_TRACETO environment variable for the possible cause. Contact Tivoli Customer Support for further assistance if appropriate.

SQL_INVALID_HANDLE (-2) X'FFFE'

Explanation: A statement handle that is not valid was encountered and SQLRowCount failed.

SQLSetConnectOption

SQL_SUCCESS (0) X'0000'

Explanation:  SQLSetConnectOption completed successfully.

SQL_SUCCESS_WITH_INFO (1) X'0001'

Explanation:  SQLSetConnectOption completed successfully, but an error may have occurred.

Problem Determination:  Review any additional information associated with the error. Driver-specific SQLSTATEs are:

- **01S02** Option value changed.
  A value similar to the value of the vParam argument was substituted because the Tivoli Information Management for z/OS ODBC driver did not support the value of the vParam argument. If the fOption is SQL_CONCURRENCY and the input value (vParam) is not SQL_CONCUR_READ_ONLY, the driver substitutes SQL_CONCUR_READONLY for the vParam.
  If the fOption is SQL_CURSOR_TYPE and the input value (vParam) is not SQL_CURSOR_FORWARD_ONLY, the driver substitutes SQL_CURSOR_FORWARDONLY for the vParam.

SQL_ERROR (-1) X'FFFF'

Explanation:  An error occurred in SQLSetConnectOption.

Problem Determination:  Review any additional information associated with the error. Driver-specific SQLSTATEs are:

- **IDBPE** Programming error msgtext.
  The Tivoli Information Management for z/OS ODBC driver encountered an unexpected error. The function that encountered the error is identified in msgtext. Record the error and retry your application. If the problem persists and if the Tivoli Information Management for z/OS ODBC driver tracing is enabled, check the trace output located in the output destination specified by the ICLUI_TRACETO environment variable for the possible cause. Contact Tivoli Customer Support for further assistance if appropriate.

- **S1C00** Driver not capable.
  Specify an fOption from the list of supported values:

  - SQL_ACCESS_MODE
  - SQL_AUTOCOMMIT
  - SQL_BIND_TYPE
  - SQL_CONCURRENCY
  - SQL_CURSOR_TYPE
  - SQL_ROWSET_SIZE

SQL_INVALID_HANDLE (-2) X'FFFE'

Explanation:  A connection handle that is not valid was encountered and SQLSetConnectOption failed.

SQLSetCursorName

SQL_SUCCESS (0) X'0000'

Explanation: SQLSetCursorName completed successfully.

SQL_ERROR (-1) X'FFFF'

Explanation: An error occurred in SQLSetCursorName.

Problem Determination: Review any additional information associated with the error. Driver-specific SQLSTATEs are:

IDBPE Programming error msgtext.

The Tivoli Information Management for z/OS ODBC driver encountered an unexpected error. The function that encountered the error is identified in msgtext. Record the error and retry your application. If the problem persists and if the Tivoli Information Management for z/OS ODBC driver tracing is enabled, check the trace output located in the output destination specified by the ICLUI_TRACETO environment variable for the possible cause. Contact Tivoli Customer Support for further assistance if appropriate.

IM001 Driver does not support this function.

Cursors are not supported by the Tivoli Information Management for z/OS ODBC driver since it does not support positioned updates or deletes. Do not use this function.

SQL_INVALID_HANDLE (-2) X'FFFE'

Explanation: A statement handle that is not valid was encountered and SQLSetCursorName failed.

SQLSetStmtOption

SQL_SUCCESS (0) X'0000'
Explanation: SQLSetStmtOption completed successfully.

SQL_SUCCESS_WITH_INFO (1) X'0001'
Explanation: SQLSetStmtOption completed successfully, but an error may have occurred.
Problem Determination: Review any additional information associated with the message. Driver-specific SQLSTATEs are:

01S02 Option value changed.

A value similar to the value of the vParam argument was substituted because the Tivoli Information Management for z/OS ODBC driver did not support the value of the vParam argument. If the fOption is SQL_CONCURRENCY and the input value (vParam) is not SQL_CONCUR_READ_ONLY, the driver substitutes SQL_CONCUR_READ_ONLY for the vParam.

If the fOption is SQL_CURSOR_TYPE and the input value (vParam) is not SQL_CURSOR_FORWARD_ONLY, the driver substitutes SQL_CURSOR_FORWARD_ONLY for the vParam.

SQL_ERROR (-1) X'FFFF'
Explanation: An error occurred in SQLSetStmtOption.
Problem Determination: Review any additional information associated with the error. Driver-specific SQLSTATEs are:

IDBPE Programming error msgtext.
The Tivoli Information Management for z/OS ODBC driver encountered an unexpected error. The function that encountered the error is identified in msgtext. Record the error and retry your application. If the problem persists and if the Tivoli Information Management for z/OS ODBC driver tracing is enabled, check the trace output located in the output destination specified by the ICLUI_TRACETO environment variable for the possible cause. Contact Tivoli Customer Support for further assistance if appropriate.

S1C00 Driver not capable.
Specify an fOption from the list of supported values:

SQL_BIND_TYPE
SQL_CONCURRENCY
SQL_CURSOR_TYPE
SQL_ROWSET_SIZE

SQL_INVALID_HANDLE (-2) X'FFFE'
Explanation: A statement handle that is not valid was encountered and SQLSetStmtOption failed.
SQLSpecialColumns

SQL_SUCCESS (0) X'0000'

Explanation: SQLSpecialColumns completed successfully.

SQL_ERROR (-1) X'FFFF'

Explanation: An error occurred in SQLSpecialColumns.

Problem Determination: Review any additional information associated with the error. Driver-specific SQLSTATEs are:

IDBPE  Programming error msgtext.

The Tivoli Information Management for z/OS ODBC driver encountered an unexpected error. The function that encountered the error is identified in msgtext. Record the error and retry your application. If the problem persists and if the Tivoli Information Management for z/OS ODBC driver tracing is enabled, check the trace output located in the output destination specified by the ICLUI_TRACETO environment variable for the possible cause. Contact Tivoli Customer Support for further assistance if appropriate.

S1001  Memory allocation failure.

Try closing any unnecessary windows. If the problem persists, restart your system and try the function again. In addition, turning off the ODBC driver tracing and HLAPI/NT logging may improve your resource usage.

SQL_INVALID_HANDLE (-2) X'FFFE'

Explanation: A statement handle that is not valid was encountered and SQLSpecialColumns failed.

Return and Reason Codes by Function

SQLStatistics

SQL_SUCCESS (0) X'0000'

Explanation: SQLStatistics completed successfully.

SQL_ERROR (-1) X'FFFF'

Explanation: An error occurred in SQLStatistics.

Problem Determination: Review any additional information associated with the error. Driver-specific SQLSTATEs are:

IDBPE Programming error msgtext.

The Tivoli Information Management for z/OS ODBC driver encountered an unexpected error. The function that encountered the error is identified in msgtext. Record the error and retry your application. If the problem persists and if the Tivoli Information Management for z/OS ODBC driver tracing is enabled, check the trace output located in the output destination specified by the ICLUI_TRACETO environment variable for the possible cause. Contact Tivoli Customer Support for further assistance if appropriate.

S1001 Memory allocation failure.

Try closing any unnecessary windows. If the problem persists, restart your system and try the function again. In addition, turning off the ODBC driver tracing and HLAPI/NT logging may improve your resource usage.

SQL_INVALID_HANDLE (-2) X'FFFE'

Explanation: A statement handle that is not valid was encountered and SQLStatistics failed.

SQLTables

SQL_SUCCESS (0) X'0000'

**Explanation:** SQLTables completed successfully.

SQL_ERROR (-1) X'FFFF'

**Explanation:** An error occurred in SQLTables.

**Problem Determination:** Review any additional information associated with the error. Driver-specific SQLSTATEs are:

**08S01** Communication link failure.

The HLAPI/NT connection failed. Call SQLError for additional error information. If the HLAPI/NT requester is not running, start it and restart your application. If the Tivoli Information Management for z/OS server on MVS (RES, MRES with APPC, or MRES with TCP/IP) is not active, start it and restart your application. Check for any errors that the HLAPI/NT may have written to the IDBPROBE.LOG file.

**IDB02** HICARETC=rc HICAREAS=rs msgtext.

The HLAPI/NT issued a HLAPI return code rc and HLAPI reason code rc. The msgtext is based on the rc and rs values.

Review the msgtext. For additional information, refer to the [Tivoli Information Management for z/OS Client Installation and User’s Guide](#) for an explanation of the HLAPI return and reason codes and take the appropriate action. If you have to restart the HLAPI/NT requester or the Tivoli Information Management for z/OS server (RES, MRES with APPC, or MRES with TCP/IP), restart your application.

**IDBPE** Programming error msgtext.

The Tivoli Information Management for z/OS ODBC driver encountered an unexpected error. The function that encountered the error is identified in msgtext. Record the error and retry your application. If the problem persists and if the Tivoli Information Management for z/OS ODBC driver tracing is enabled, check the trace output located in the output destination specified by the ICLUI_TRACETO environment variable for the possible cause. Contact Tivoli Customer Support for further assistance if appropriate.

**IDBT0** Table name is not valid.

The table name is too long. The table names supported by the Tivoli Information Management for z/OS data source can be up to 8 characters long.

**S1C00** Driver not capable.

Do not specify a table qualifier (szTableQualifier) or a table owner (szTableOwner).

**S1001** Memory allocation failure.

Try closing any unnecessary windows. If the problem persists, restart your system and try the function again. In addition, turning off the ODBC driver tracing and HLAPI/NT logging may improve your resource usage.
SQL_INVALID_HANDLE (-2) X'FFFE'

Explanation: A statement handle that is not valid was encountered and SQLTables failed.

SQLTransact

**SQL_SUCCESS (0) X'0000'**

**Explanation:** SQLTransact completed successfully.

**SQL_ERROR (-1) X'FFFF'**

**Explanation:** An error occurred in SQLTransact.

**Problem Determination:** Review any additional information associated with the error. Driver-specific SQLSTATEs are:

- **IDBPE** Programming error *msgtext*.
  
  The Tivoli Information Management for z/OS ODBC driver encountered an unexpected error. The function that encountered the error is identified in *msgtext*. Record the error and retry your application. If the problem persists and if the Tivoli Information Management for z/OS ODBC driver tracing is enabled, check the trace output located in the output destination specified by the ICLUI_TRACEETO environment variable for the possible cause. Contact Tivoli Customer Support for further assistance if appropriate.

- **S1C00** Driver not capable.
  
  Specify an *fType* from the list of supported values:

  SQL_COMMIT

**SQL_INVALID_HANDLE (-2) X'FFFF'**

**Explanation:** A statement handle that is not valid was encountered and SQLTransact failed.

**Problem Determination:** A programming error occurred. Contact Tivoli Customer Support.
SQL Command Syntax and Parameters

If you are a programmer and plan to write your own ODBC-enabled database application, refer to the following section for information on the SQL statements and elements supported by Tivoli Information Management for z/OS.

For more information on writing C language applications to the open database connectivity interface, or for more information on driver support, refer to Microsoft’s online API reference information.

ODBC Functions Supported

The following alphabetical list contains the ODBC Core, Level 1, and Level 2 functions supported by the Tivoli Information Management for z/OS ODBC driver. For a description of parameter specifications, refer to Microsoft’s online API reference information. The Tivoli Information Management for z/OS ODBC driver may not support all of the functionality described in this section. Unique processing specific to the Tivoli Information Management for z/OS ODBC driver is noted here by function.

- SQLAllocConnect
- SQLAllocEnv
- SQLAllocStmt
- SQLBindCol
  
  The Tivoli Information Management for z/OS ODBC driver does not support bookmarks.
  
  Although SQLBindCol allows all fCType, the Tivoli Information Management for z/OS driver only supports conversion on a subset of them. Conversion is done when the data is fetched so verification of the fCType is not done until SQLFetch is issued. See SQLFetch for a list of fCType supported by the Tivoli Information Management for z/OS ODBC driver.

  **Note:** Result sets returning Tivoli Information Management for z/OS data only return character data. To bind columns for a Tivoli Information Management for z/OS result set, use an fCType of SQL_C_CHAR, SQL_C_DEFAULT or SQL_BINARY.

- SQLBindParameter
  
  The input SQL data type arguments (fSqlType) supported by the Tivoli Information Management for z/OS ODBC driver are:
  
  - SQL_CHAR
  - SQL_LONGVARCHAR
  - SQL_VARCHAR
Otherwise, the driver returns SQL_ERROR, SQLSTATE S1C00 (Driver Not capable).

When the driver executes the SQL Statement on SQLExecute or SQLExecDirect, the driver converts any parameter values from the fCType specified on SQLBindParameter to the fSqlType of SQL_CHAR, if needed. The conversion is as documented in Microsoft’s online API reference information, with the following exceptions:

- Converting fCType of SQL_C_DATE to the fSqlType of SQL_CHAR, the resulting character data is in the “YYYY/MM/DD” format.
- Converting fCType of SQL_C_TIME to the fSqlType of SQL_CHAR, the resulting character data is in the “HH:MM” format.
- Converting fCType of SQL_C_TIMESTAMP to the fSqlType of SQL_CHAR, the resulting character data is in the “YYYY/MM/DD HH:MM” format.

SQLCancel
SQLColAttributes
SQLColumns

The Tivoli Information Management for z/OS ODBC driver and the Tivoli Information Management for z/OS data source do not support the use of table qualifiers or table owners. If a table qualifier (szTableQualifier) or table owner (szTableOwner) is specified, the driver returns SQL_ERROR, SQLSTATE S1C00 (Driver not capable), indicating a table qualifier or owner was specified and the Tivoli Information Management for z/OS ODBC driver or data source does not support qualifiers or owners.

Note: The percent (%) metacharacter is only supported as a search character in the last position of the table name search pattern (szTableName) and the last position of the column name search pattern (szColumnName). A percent in any other position of the table name search pattern or the column name search pattern is treated as data.

SQLConnect
SQLDescribeCol
SQLDescribeParam
SQLDisconnect
SQLDriverConnect

The syntax of the ODBC-defined connection string is described in Microsoft’s online API reference information.

If the connection string contains the DRIVER keyword, the Tivoli Information Management for z/OS driver cannot use the information stored in the Windows Registry at installation or on a previous SQLDriverConnect to augment the attribute-values specified in the connection string. Therefore, all required attribute-keywords must be specified on the connection string. If a required attribute-keyword is not supplied on the connection string, and the SQLDriverConnect fDriverCompletion value is SQL_DRIVER_NOPROMPT, the driver returns SQL_ERROR.

Otherwise, the Tivoli Information Management for z/OS ODBC driver uses the information in the Windows Registry to augment the information passed on the connection string as the initial values which are displayed in the dialog box during the SQLDriverConnect function, if the fDriverCompletion flag is SQL_DRIVER_PROMPT.
or if any information is missing and the fDriverCompletion flag is SQL_DRIVER_COMPLETE or SQL_DRIVER_COMPLETE_REQUIRED.

The driver uses the value of the first occurrence of a keyword if the keyword is repeated in the connection string. Any change to these values is saved in the Windows Registry on completion.

Note: The attribute-value, including a blank value, for required attribute-keywords is specified on the corresponding control PDB for the transaction. Any errors are returned by the SQLDriverConnect function, as a result of the HL01 transaction.

If there is a blank attribute-value for an optional attribute-keyword, the corresponding control PDB is not specified on the transaction.

**DSN**
The attribute-value supplied at installation for data source name. It must not consist solely of blanks and cannot contain the backslash (\) character. The default value is ‘ServiceDesk390’.

**DRIVER**
The description of the driver: ‘Tivoli Information Management for z/OS Driver’.

**UID**
Optional attribute-keyword, but an attribute-value is required to connect to the data source. This data value is used for the SECURITY_ID on the HL01 initialization transaction.

**PWD**
The PWD attribute-value is required on all invocations of SQLDriverConnect. There is no default value and the PWD attribute-value is not saved in the Windows NT Registry for subsequent invocations.

In addition, the SQLDriverConnect function has the following Tivoli Information Management for z/OS driver-defined attribute-keywords. These keywords correspond to the fields listed on the Tivoli Information Management for z/OS Data Source Advanced Options panel (see Figure 7 on page 29) unless otherwise noted.

- **APPLICATION_ID** — Optional, but an attribute-value is required to connect to the data source. This data value is used for the HL01 initialization transaction and corresponds to the Application ID field on the Tivoli Information Management for z/OS Data Source Information panel (see Figure 6 on page 28).

- **PRIVILEGE_CLASS** — Optional, but an attribute-value is required to connect to the data source. This data value is used for the HL01 initialization transaction and corresponds to the Privilege class field on the Advanced Options panel.

- **SESSION_MEMBER** — Optional, but an attribute-value is required to connect to the data source. This data value is used for the HL01 initialization transaction and corresponds to the Session member field on the Advanced Options panel.

- **DATABASE_PROFILE** — Optional, but an attribute-value is required to connect to the data source. This data value is used for the HL01 initialization transaction and corresponds to the Database profile field on the Advanced Options panel.

- **APIMSG_OPTION** — An attribute-value is optional. This data value is used for the HL01 initialization transaction and corresponds to the LLAPI message option field on the Advanced Options panel. If you are using an MRES with a pre-started session,
and enter a value for this option that does not match the value specified for the pre-started MRES, the value entered here is ignored.

- **HLIMSG_OPTION** — An attribute-value is optional. This data value is used for the HL01 initialization transaction and corresponds to the HLAPI message option field on the Advanced Options panel. If you are using an MRES with a pre-started session, and enter a value for this option that does not match the value specified for the pre-started MRES, the value entered here is ignored.

- **DATABASE_ID** — An attribute-value is optional. This data value is used for the HL01 initialization transaction and corresponds to the Database ID field on the Advanced Options panel. If you are using an MRES with a pre-started session, and enter a value for this option that does not match the value specified for the pre-started MRES, the value entered here is ignored.

- **SPOOL_INTERVAL** — An attribute-value is optional. This data value is used for the HL01 initialization transaction and corresponds to the Host log spool interval field on the Advanced Options panel. If you are using an MRES with a pre-started session, and enter a value for this option that does not match the value specified for the pre-started MRES, the value entered here is ignored.

- **TABLE_COUNT** — Optional, but an attribute-value is required to connect to the data source, and it must be greater than zero. This data value is used for the HL01 initialization transaction and corresponds to the HLAPI table count field on the Advanced Options panel. If you are using an MRES with a pre-started session, and enter a value for this option that does not match the value specified for the pre-started MRES, the value entered here is ignored.

- **TIMEOUT_INTERVAL** — An attribute-value is optional. This data value is used for the HL01 initialization transaction and corresponds to the Host API timeout interval field on the Advanced Options panel. If you are using an MRES with a pre-started session, and enter a value for this option that does not match the value specified for the pre-started MRES, the value entered here is ignored.

- **NUMBER_OF_HITS** — An attribute-value is optional. This data value is used for the HL11 record inquiry transaction and corresponds to the Maximum search hits field on the Advanced Options panel. The default value is 500.

- **TEXT_UNITS** — An attribute-value is optional. This data value is used for the HL06 record retrieval transaction and corresponds to the Freeform text max lines limit field on the Advanced Options panel.

- **TEXT_WIDTH** — An attribute-value is optional. This data value is used for the HL06 record retrieval transaction and corresponds to the Freeform text line length field on the Advanced Options panel.

- **TEXT_AREA** — An attribute-value is optional. This data value is used for the HL06 record retrieval transaction and corresponds to the Freeform text area field on the Advanced Options panel.

### SQL Error

### SQLExecDirect

See [Elements Used in SQL Statements” on page 118](#) for the syntax of the SQL SELECT statement to be executed.

ODBC date and time escape clauses in an SQL SELECT statement WHERE clause are translated to the Tivoli Information Management for z/OS date and time format. Dates in
the ODBC date escape clause are transformed from YYYY-MM-DD format to the 4-digit year internal date format YYYY/MM/DD. Times in the ODBC time escape clause are transformed from HH:MM:SS format to HH:MM format. In addition, if a search field is a date with data in the format YYYY-MM-DD, it is translated to YYYY/MM/DD format.

When the driver executes the SQL Statement, any bound parameter data is retrieved from the storage buffers. For data-at-execution parameters, the data is sent with SQLPutData. The driver converts any parameter values from the $fCType$ specified on SQLBindParameter to the $fSqlType$, if needed. See the discussion of SQLBindParameter on page 107 for additional information regarding parameter conversion.

- **SQLExecute**
  When the driver executes the SQL Statement, any bound parameter data is retrieved from the storage buffers. For data-at-execution parameters, the data is sent with SQLPutData. The driver converts any parameter values from the $fCType$ specified on SQLBindParameter to the $fSqlType$, if needed. See the discussion of SQLBindParameter on page 107 for additional information regarding parameter conversion.

- **SQLExtendedFetch**
  SQLExtendedFetch fetches one or more rows of data for all columns that were bound to storage locations using SQLBindCol. SQLExtendedFetch also converts the data to the C data type specified in the $fCType$ argument on SQLBindCol. SQLExtendedFetch only supports conversion on a subset of $fCTypes$ that can be specified on SQLBindCol. The conversions supported depend on the type of data being returned.

Result sets returning Tivoli Information Management for z/OS data only return character data. To bind columns for a Tivoli Information Management for z/OS result set, you must use an $fCType$ of SQL_C_CHAR, SQL_C_DEFAULT, or SQL_BINARY. The same is true for any character data type columns for other result sets; for example, the TABLE_NAME column in a SQLTables result set. If you specified an $fCType$ on SQLBindCol of any of the following, SQLExtendedFetch returns SQL_ERROR, SQLSTATE 22005 (Error in assignment):

- SQL_C_BIT
- SQL_C_STINYINT
- SQL_C_UTINYINT
- SQL_C_TINYINT
- SQL_C_SSHORT
- SQL_C_USHORT
- SQL_C_SHORT
- SQL_C_SLONG
- SQL_C_ULONG
- SQL_C_FLOAT
- SQL_C_DOUBLE

If the data is character and you specified an $fCType$ on SQLBindCol of any of the following, SQLExtendedFetch returns SQL_ERROR, SQLSTATE 22008 (Datetime field overflow):

- SQL_C_DATE
- SQL_C_TIME
- SQL_C_TIMESTAMP
Note: Although the Tivoli Information Management for z/OS database contains dates and times, the data is character data and the date and time formats are controlled at the host by the database administrator. The ODBC driver converts dates to YYYY-MM-DD format.

**SQLFetch**

SQLFetch fetches a row of data for all columns that were bound to storage locations using SQLBindCol. SQLFetch also converts the data to the C data type specified in the fCType argument on SQLBindCol. SQLFetch only supports conversion on a subset of fCTypes that can be specified on SQLBindCol. The conversions supported depend on the type of data being returned.

Result sets returning Tivoli Information Management for z/OS data only return character data. To bind columns for a Tivoli Information Management for z/OS result set, you must use an fCType of SQL_C_CHAR, SQL_C_DEFAULT, or SQL_BINARY. The same is true for any character data type columns for other result sets; for example, the TABLE_NAME column in a SQLTables result set. If you specified an fCType on SQLBindCol of any of the following, SQLFetch returns SQL_ERROR, SQLSTATE 22005 (Error in assignment):

- SQL_C_BIT
- SQL_C_STINYINT
- SQL_C_UTINYINT
- SQL_C_TINYINT
- SQL_C_SSHORT
- SQL_C_USHORT
- SQL_C_SHORT
- SQL_C_SLONG
- SQL_C_ULONG
- SQL_C_FLOAT
- SQL_C_DOUBLE

If the data is character and you specified an fCType on SQLBindCol of any of the following, SQLFetch returns SQL_ERROR, SQLSTATE 22008 (Datetime field overflow):

- SQL_C_DATE
- SQL_C_TIME
- SQL_C_TIMESTAMP

Note: Although the Tivoli Information Management for z/OS database contains dates and times, the data is character data and the date and time formats are controlled at the host by the database administrator. The ODBC driver converts dates to YYYY-MM-DD format.

Non-Tivoli Information Management for z/OS result sets that are supported by the ODBC Tivoli Information Management for z/OS driver may have columns that contain Integer or Smallint data as well as columns that have character data. (Non-Tivoli Information Management for z/OS result sets are those that are returned by an ODBC function such as SQLTables, and are not Tivoli Information Management for z/OS database data.) For example, the PRECISION column in a result set for SQLColumns has a data type of Integer and the SCALE column has a data type of Smallint. If the
data type of a column in a non-Tivoli Information Management for z/OS result set is Smallint, the Tivoli Information Management for z/OS driver supports the following fCType:

- SQL_C_DEFAULT
- SQL_C_SHORT
- SQL_C_SSHORT
- SQL_C_LONG
- SQL_C_SLONG
- SQL_C_FLOAT
- SQL_C_DOUBLE
- SQL_C_BINARY
- SQL_C_CHAR

SQLFetch returns SQL_ERROR, SQLSTATE S1C00 (Driver not capable) for other fCTypes.

If the data type of a column in a non-Tivoli Information Management for z/OS result set is Integer, the Tivoli Information Management for z/OS driver supports the following fCType:

- SQL_C_DEFAULT
- SQL_C_LONG
- SQL_C_SLONG
- SQL_C_FLOAT
- SQL_C_DOUBLE
- SQL_C_BINARY
- SQL_C_CHAR

SQLFetch returns SQL_ERROR, SQLSTATE S1C00 (Driver not capable) for other fCTypes.

To determine the data types for columns in non-Tivoli Information Management for z/OS result sets, like SQLColumns and SQLTables, refer to Microsoft’s online API reference information.

- SQLFreeConnect
- SQLFreeEnv
- SQLFreeStmt
- SQLGetConnectOption
  The input options (fOption) supported by the Tivoli Information Management for z/OS ODBC driver are:
  - SQL_ACCESS_MODE
  - SQL_AUTOCOMMIT

If an fOption that the driver does not support is specified, the driver returns SQL_ERROR, SQLSTATE S1C00 (Driver not capable).

- SQLGetCursorName
  Cursors are not supported by the Tivoli Information Management for z/OS ODBC driver since it does not support positioned updates or deletes. The driver returns SQL_ERROR, SQLSTATE IM001 (Driver does not support this function).

- SQLGetData
If the icol is 0, the driver returns SQL_ERROR, SQLSTATE S1C00 (Driver not capable).

If the icol is valid, the driver returns data from the column for the current row in the result set to the specified storage location using the conversion type specified by fCType. The conversions that are supported depend on the data type of the requested column.

Columns in a Tivoli Information Management for z/OS result set only contain character data. To get data for a column in a Tivoli Information Management for z/OS result set, you must specify an fCType of SQL_C_CHAR, SQL_C_DEFAULT, or SQL_BINARY. The same is true for any character data type columns for other result sets as with, for example, the TABLE_NAME column in a SQLTables result set.

If the data is character and you specify an fCType of any of the following, SQLGetData returns SQL_ERROR, SQLSTATE 22005 (Error in assignment):

- SQL_C_BIT
- SQL_C_STINYINT
- SQL_C_UTINYINT
- SQL_C_TINYINT
- SQL_C_SSHORT
- SQL_C_USHORT
- SQL_C_SHORT
- SQL_C_SLONG
- SQL_C_ULONG
- SQL_C_FLOAT
- SQL_C_DOUBLE

If the data is character and you specify an fCType of any of the following, SQLGetData returns SQL_ERROR, SQLSTATE 22008 (Datetime field overflow):

- SQL_C_DATE
- SQL_C_TIME
- SQL_C_TIMESTAMP

Note: Although the Tivoli Information Management for z/OS database contains dates and times, the data is character data and the date and time formats are controlled at the host by the database administrator. The ODBC driver converts dates to YYYY-MM-DD format.

Non-Tivoli Information Management for z/OS result sets that are supported by the Tivoli Information Management for z/OS driver may have columns that contain Integer or Smallint data as well as columns that have character data. (Non-Tivoli Information Management for z/OS result sets are those that are returned by an ODBC function such as SQLTables, and are not Tivoli Information Management for z/OS database data.) For example, the PRECISION column in a result set for SQLColumns has a data type of Integer and the SCALE column has a data type of Smallint. If the data type of a column in a non-Tivoli Information Management for z/OS result set is Smallint, the Tivoli Information Management for z/OS ODBC driver supports the following fCTypes:

- SQL_C_DEFAULT
- SQL_C_SHORT
- SQL_C_SSHORT
- SQL_C_LONG
- SQL_C_SLONG
- SQL_C_FLOAT
SQLGetData returns SQL_ERROR, SQLSTATE S1C00 (Driver not capable) for other fCTypes.

If the data type of a column in a non-Tivoli Information Management for z/OS result set is Integer, the Tivoli Information Management for z/OS ODBC driver supports the following fCTypes:

- SQL_C_DEFAULT
- SQL_C_LONG
- SQL_C_SLONG
- SQL_C_FLOAT
- SQL_C_DOUBLE
- SQL_C_BINARY
- SQL_C_CHAR

SQLGetData returns SQL_ERROR, SQLSTATE S1C00 (Driver not capable) for other fCTypes.

To determine the data types for columns in non-Tivoli Information Management for z/OS result sets, like SQLColumns and SQLTables, refer to Microsoft’s online API reference information.

**SQLGetFunctions**

This function is implemented by the driver manager. It is not implemented in the Tivoli Information Management for z/OS ODBC driver.

**SQLGetInfo**

See [“SQLGetInfo Information Types and Values Returned” on page 119](#) for a list of information types and values returned. For a description of the information types, refer to Microsoft’s online API reference information.

**SQLGetStmtOption**

The input options (fOption) supported by the Tivoli Information Management for z/OS ODBC driver are:

- SQL_BIND_TYPE
- SQL_CONCURRENCY
- SQL_CURSOR_TYPE
- SQL_ROW_NUMBER
- SQL_ROWSET_SIZE

If an fOption that the driver does not support is specified, the driver returns SQL_ERROR, SQLSTATE S1C00 (Driver not capable).

**SQLGetTypeInfo**

The input SQL data type arguments (fSqlType) supported by the Tivoli Information Management for z/OS ODBC driver are:

- SQL_CHAR
- SQL_LONGVARCHAR
- SQL_VARCHAR
- SQL_ALL_TYPES
The driver returns an empty result set for all other SQL data type ($fSqlType) arguments.

- **SQLNativeSQL**
  ODBC date and time escape clauses in a SQL SELECT statement WHERE clause (or stand-alone) are translated to the Tivoli Information Management for z/OS date and time format. Dates in the ODBC date escape clause are transformed from a YYYY-MM-DD format to the 4-digit year internal date format YYYY/MM/DD. Times in the ODBC time escape clause are transformed from HH:MM:SS format to HH:MM format.

  **Note:** Because the ODBC timestamp escape clause is not supported, it returns a syntax error (37000) if it is used as input to the SQLNativeSQL function.

- **SQLNumParams**
- **SQLNumResultCols**
- **SQLParamData**
- **SQLPrepare**
  See [Elements Used in SQL Statements” on page 118](#) for the syntax of the SQL SELECT statement to be executed.
  ODBC date and time escape clauses in a SQL SELECT statement WHERE clause are translated to the Tivoli Information Management for z/OS date and time format. Dates in the ODBC date escape clause are transformed from a YYYY-MM-DD format to the 4-digit year internal date format YYYY/MM/DD. Times in the ODBC time escape clause are transformed from HH:MM:SS format to HH:MM format. In addition, if a search field is a date with data in the format YYYY-MM-DD, it is translated to YYYY/MM/DD format.

- **SQLPutData**
- **SQLRowCount**
- **SQLSetConnectOption**
  The input options ($fOption) supported by the Tivoli Information Management for z/OS ODBC driver are:
  ```
  SQL_ACCESS_MODE
  SQL_AUTOCOMMIT
  SQL_BIND_TYPE
  SQL_CONCURRENCY
  SQL_CURSOR_TYPE
  SQL_ROWSET_SIZE
  ```
  If the $fOption is SQL_CONCURRENCY and the input value ($vParam) is not SQL_CONCUR_READ_ONLY, the driver substitutes SQL_CONCUR_READ_ONLY for the $vParam and returns SQL_SUCCESS_WITH_INFO, SQLSTATE 01S02 (Option value changed).

  If the $fOption is SQL_CURSOR_TYPE and the input value ($vParam) is not SQL_CURSOR_FORWARD_ONLY, the driver substitutes SQL_CURSOR_FORWARD_ONLY for the $vParam and returns SQL_SUCCESS_WITH_INFO, SQLSTATE 01S02 (Option value changed).

  If an $fOption that the driver does not support is specified, the driver returns SQL_ERROR, SQLSTATE S1C00 (Driver not capable).
SQLSetCursorName

Cursors are not supported by the Tivoli Information Management for z/OS ODBC driver since it does not support positioned updates or deletes. The driver returns SQL_ERROR, SQLSTATE IM001 (Driver does not support this function).

SQLSetStmtOption

The input options (fOption) supported by the Tivoli Information Management for z/OS ODBC driver are:

- SQL_BIND_TYPE
- SQL_CONCURRENCY
- SQL_CURSOR_TYPE
- SQL_ROWSET_SIZE

If the fOption is SQL_CONCURRENCY and the input value (vParam) is not SQL_CONCUR_READ_ONLY, the driver substitutes SQL_CONCUR_READ_ONLY for the vParam and returns SQL_SUCCESS_WITH_INFO, SQLSTATE 01S02 (Option value changed).

If the fOption is SQL_CURSOR_TYPE and the input value (vParam) is not SQL_CURSOR_FORWARD_ONLY, the driver substitutes SQL_CURSOR_FORWARD_ONLY for the vParam and returns SQL_SUCCESS_WITH_INFO, SQLSTATE 01S02 (Option value changed).

If an fOption that the driver does not support is specified, the driver returns SQL_ERROR, SQLSTATE S1C00 (Driver not capable).

SQLSpecialColumns

The Tivoli Information Management for z/OS ODBC driver returns an empty result set.

SQLStatistics

The Tivoli Information Management for z/OS ODBC driver returns an empty result set.

SQLTables

The Tivoli Information Management for z/OS ODBC driver and the Tivoli Information Management for z/OS data source do not support the use of table qualifiers or table owners. If a table qualifier (szTableQualifier) or table owner (szTableOwner) is specified, the driver returns SQL_ERROR, SQLSTATE S1C00 (Driver not capable), indicating that a table qualifier or owner was specified and the Tivoli Information Management for z/OS ODBC driver or data source does not support either.

Note: The percent (%) metacharacter is only supported as a search character in the last position of the table name search pattern (szTableName). A percent in any other position of the table name search pattern is treated as data.

SQLTransact

The Tivoli Information Management for z/OS ODBC driver does not support transactions. The only input type (fType) supported by the driver is:

- SQL_COMMIT

If the fType is SQL_ROLLBACK, the driver returns SQL_ERROR, SQLSTATE S1C00 (Driver not capable), indicating the Tivoli Information Management for z/OS ODBC driver or data source does not support the ROLLBACK operation.
**Elements Used in SQL Statements**

Refer to "SQL Conformance Levels" on page 20 for a description of the SQL functions supported by the Tivoli Information Management for z/OS ODBC driver and explanation of restrictions.

```
SELECT [ALL | DISTINCT] select-list
FROM table-name
[WHERE search-condition]
    [order-by-clause]
```

DISTINCT is ignored; duplicate rows are not removed. The `order-by-clause` is ignored.

```
between-predicate ::= expression BETWEEN expression AND expression
Note: NOT BETWEEN is not supported.
```

```
boolean-factor ::= [NOT] boolean-primary
boolean-primary ::= predicate | (search-condition)
boolean-term ::= boolean-factor [AND boolean-term]
```

```
character-string-literal ::= `[character]`
Each word in a character-string-literal must be specified as a separate search-condition.
```

```
column-name ::= [[table-name | correlation-name].] column-identifier
Note: The maximum column name length is 32 characters. The column name must start with an alphabetic character and remaining characters must be alphanumeric or underscore.
```

```
column-identifier ::= uppercase-letter [digit | uppercase-letter _ ]...
```

```
comparison-operator ::= < | = | > | = | = | = | = | < >
Note: > and < are allowed with NOT. NOT with <=, or NOT with >=, are not allowed.
```

```
comparison-predicate ::= expression comparison-operator expression
```

```
date-separator ::= - (dash)
Note: The `-` (dash) is translated to a `/` (slash) by the Tivoli Information Management for z/OS ODBC driver.
```

```
date-value ::= years-value date-separator months-value date-separator days-value
```

```
days-value ::= digit digit digit
```

```
digit ::= 0|1|2|3|4|5|6|7|8|9
```

```
dynamic-parameter ::= ?
```

Parameter markers are not allowed in the following circumstances:

- In a `select-list`
- As both expressions in a `comparison-predicate`
- As both the first and second operands of a BETWEEN operation
- As both the first and third operands of a BETWEEN operation

```
expression ::= term
```

```
factor ::= primary
```

```
literal ::= character-string-literal | numeric-literal | ODBC-date-time-extension
Note: A numeric-literal is valid only in a WHERE clause, where the comparison contains all numeric-literals. An ODBC-date-time-extension is valid only in a WHERE clause.
```

```
like-predicate ::= expression [NOT] LIKE pattern-value
```

```
months-value ::= digit digit
```

```
null-predicate ::= column-name IS [ NOT] NULL
```

```
ODBC-date-literal ::= ODBC-std-esc-initiator d date-value ODBC-std-esc-terminator | ODBC-ext-esc-initiator d date-value ODBC-ext-esc-terminator
```

```
ODBC-date-time-extension ::= ODBC-date-literal | ODBC-time-literal
```

**Table 1.1: ODBC Functions Supported**

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SELECT</td>
<td>SQL SELECT statement.</td>
</tr>
<tr>
<td>FROM</td>
<td>SQL FROM statement.</td>
</tr>
<tr>
<td>WHERE</td>
<td>SQL WHERE clause.</td>
</tr>
<tr>
<td>ORDER BY</td>
<td>SQL ORDER BY clause.</td>
</tr>
<tr>
<td>DISTINCT</td>
<td>SQL DISTINCT clause.</td>
</tr>
<tr>
<td>BETWEEN</td>
<td>SQL BETWEEN clause.</td>
</tr>
<tr>
<td>NOT BETWEEN</td>
<td>NOT SQL BETWEEN clause.</td>
</tr>
<tr>
<td>LIKE</td>
<td>SQL LIKE clause.</td>
</tr>
<tr>
<td>IS NULL</td>
<td>SQL IS NULL clause.</td>
</tr>
<tr>
<td>DATE</td>
<td>SQL DATE function.</td>
</tr>
<tr>
<td>TIME</td>
<td>SQL TIME function.</td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td>SQL TIMESTAMP function.</td>
</tr>
</tbody>
</table>

**Version 7.1**
ODBC Functions Supported

8. SQL Command Syntax and Parameters

SQLGetInfo Information Types and Values Returned

The following information types apply to the ODBC function SQLGetInfo, and represent the values to be returned for each value of the input parameter fInfoType, the type of information requested.

**SQL_ACCESSIBLE_PROCEDURES**

"N"; does not apply to driver or data source. Driver does not support procedures.

**SQL_ACCESSIBLE_TABLES**

"Y"; you are authorized to perform a SELECT against all tables returned by SQLTables, although you may not be authorized to retrieve the data for all tables. If you are not authorized to retrieve the data, you will receive a SQL_ERROR return code on a subsequent SQLExecute or SQLExecDirect function.

**SQL_ACTIVE_CONNECTIONS**

0x00; there is no specified limit or the limit is unknown.

**SQL_ACTIVE_STATEMENTS**

0x00; there is no specified limit or the limit is unknown.

**SQL_ALTER_TABLE**

0x0000; does not apply to driver or data source. Driver does not support the ALTER TABLE statement.
ODBC Functions Supported

**SQL_BOOKMARK_PERSISTENCE**

0x0000; does not apply to driver or data source.

**SQL_COLUMN_ALIAS**

"N"

**SQL_CONCAT_NULL_BEHAVIOR**

0x00; does not apply to driver or data source.

**SQL_CONVERT_BIGINT**

0x0000; does not apply to driver or data source. Driver does not support the CONVERT scalar function.

**SQL_CONVERT_BINARY**

0x0000; does not apply to driver or data source. Driver does not support the CONVERT scalar function.

**SQL_CONVERT_BIT**

0x0000; does not apply to driver or data source. Driver does not support the CONVERT scalar function.

**SQL_CONVERT_CHAR**

0x0000; does not apply to driver or data source. Driver does not support the CONVERT scalar function.

**SQL_CONVERT_DATE**

0x0000; does not apply to driver or data source. Driver does not support the CONVERT scalar function.

**SQL_CONVERT_DECIMAL**

0x0000; does not apply to driver or data source. Driver does not support the CONVERT scalar function.

**SQL_CONVERT_DOUBLE**

0x0000; does not apply to driver or data source. Driver does not support the CONVERT scalar function.

**SQL_CONVERT_FLOAT**

0x0000; does not apply to driver or data source. Driver does not support the CONVERT scalar function.

**SQL_CONVERT_INTEGER**

0x0000; does not apply to driver or data source. Driver does not support the CONVERT scalar function.

**SQL_CONVERT_LONGVARBINARY**

0x0000; does not apply to driver or data source. Driver does not support the CONVERT scalar function.

**SQL_CONVERT_LONGVARCHAR**

0x0000; does not apply to driver or data source. Driver does not support the CONVERT scalar function.

**SQL_CONVERT_NUMERIC**

0x0000; does not apply to driver or data source. Driver does not support the CONVERT scalar function.

**SQL_CONVERT_REAL**

0x0000; does not apply to driver or data source. Driver does not support the CONVERT scalar function.
SQL_CONVERT_SMALLINT
0x0000; does not apply to driver or data source. Driver does not support the CONVERT scalar function.

SQL_CONVERT_TIME
0x0000; does not apply to driver or data source. Driver does not support the CONVERT scalar function.

SQL_CONVERT_TIMESTAMP
0x0000; does not apply to driver or data source. Driver does not support the CONVERT scalar function.

SQL_CONVERT_TINYINT
0x0000; does not apply to driver or data source. Driver does not support the CONVERT scalar function.

SQL_CONVERT_VARBINARY
0x0000; does not apply to driver or data source. Driver does not support the CONVERT scalar function.

SQL_CONVERT_VARCHAR
0x0000; does not apply to driver or data source. Driver does not support the CONVERT scalar function.

SQL_CONVERT_FUNCTIONS
0x0000; does not apply to driver or data source. Driver does not support the scalar conversion functions.

SQL_CORRELATION_NAME
SQL_CN_ANY (0x0002); correlation names can be any valid user-defined name.

SQL_CURSOR_COMMIT_BEHAVIOR
0x00; does not apply to driver or data source. Commit transactions not supported.

SQL_CURSOR_ROLLBACK
0x00; does not apply to driver or data source. Rollback transactions not supported

SQL_DATA_SOURCE_NAME
″ServiceDesk390″; the default data source name.

SQL_DATA_SOURCE_READ_ONLY
″N″; the Tivoli Information Management for z/OS data source is not read-only.

SQL_DATABASE_NAME
Name of the current database in use.

SQL_DBMS_NAME
″Tivoli Information Management for z/OS″

SQL_DBMS_VER
02.01.0000 Tivoli Information Management for z/OS 2.1

SQL_DEFAULT_TXN_ISOLATION
0x0000; does not apply to driver or data source. Driver does not support transactions.

SQL_DRIVER_NAME
BLMODBC

SQL_DRIVER_ODBC_VER
03.51
SQL_DRIVER_VER
02.01.0000

SQL_EXPRESSIONS_IN_ORDERBY
"N"; data source does not support expressions in ORDER BY list.

SQL_FETCH_DIRECTION
SQL_FD_FETCH_FIRST, SQL_FD_FETCH_NEXT

SQL_FILE_USAGE
SQL_FILE_NOT_SUPPORTED (0x00); files are not applicable.

SQL_GETDATA_EXTENSIONS
0x0000; no SQLGetData extensions are supported.

SQL_GROUP_BY
SQL_GB_NOT_SUPPORTED (0x00); GROUP BY clauses are not supported.

SQL_IDENTIFIER_CASE
SQL_IC_UPPER; Identifiers in SQL are case insensitive and are stored upper case in system catalog.

SQL_IDENTIFIER_QUOTE_CHAR
" " (blank); data source does not support quoted identifiers.

SQL_KEYWORDS
APPLICATION_ID,SESSION_MEMBER,PRIVILEGE_CLASS,
TABLE_COUNT,APIMSG_OPTION,HLIMSG_OPTION,
TIMEOUT_INTERVAL,SPOOL_INTERVAL,DATABASE_ID,
DATABASE_PROFILE,NUMBER_OF_HITS,TEXT_UNITS,
TEXT_WIDTH, TEXT_AREA

SQL_LIKE_ESCAPE_CLAUSE
"N"; driver does not support a LIKE predicate escape character.

SQL_LOCK_TYPES
0x0000; does not apply to driver. Driver does not support SQLSetPos.

SQL_MAX_BINARY_LITERAL_LEN
0x0000; there is no maximum length or the length is unknown.

SQL_MAX_CHAR_LITERAL_LEN
0x0000; there is no maximum length or the length is unknown.

SQL_MAX_COLUMN_NAME_LEN
32

SQL_MAX_COLUMNS_IN_GROUP_BY
0x00; does not apply to driver or data source. Driver does not support GROUP BY clause.

SQL_MAX_COLUMNS_IN_INDEX
0x00; does not apply to driver or data source. Driver does not support indexes.

SQL_MAX_COLUMNS_IN_ORDER_BY
0x00; does not apply to driver or data source. Driver does not support ORDER BY clause.

SQL_MAX_COLUMNS_IN_SELECT
0x00; there is no specified limit or the limit is unknown.
SQL_MAX_COLUMNS_IN_TABLE
0x00; there is no specified limit or the limit is unknown.

SQL_MAX_CURSOR_NAME_LEN
0x00; does not apply to driver or data source. Driver does not support cursor names.

SQL_MAX_INDEX_SIZE
0x0000; does not apply to driver or data source. Driver does not support indexes.

SQL_MAX_OWNER_NAME_LEN
0x00; does not apply to driver or data source. Driver does not support owner names.

SQL_MAX_PROCEDURE_NAME_LEN
0x00; does not apply to driver or data source. Driver does not support procedures.

SQL_MAX_QUALIFIER_NAME_LEN
0x00; does not apply to driver or data source. Driver does not support qualifier names.

SQL_MAX_ROW_SIZE
0x0000; there is no specified limit or the limit is unknown.

SQL_MAX_ROW_SIZE_INCLUDES_LONG
"N"

SQL_MAX_STATEMENT_LEN
0x0000; there is no specified limit or the limit is unknown.

SQL_MAX_TABLE_NAME_LEN
8

SQL_MAX_TABLES_IN_SELECT
1

SQL_MAX_USER_NAME_LEN
8

SQL_MULT_RESULT_SETS
"N"

SQL_MULT_ACTIVE_TXN
"N"

SQL_NEED_LONG_DATA_LEN
"N"

SQL_NON_NULLABLE_COLUMNS
0x00; does not apply to driver or data source.

SQL_NULL_COLLATION
0x00; does not apply to driver or data source.

SQL_NUMERIC_FUNCTIONS
0x0000; does not apply to driver or data source. Driver does not support scalar numeric functions.

SQL_ODBC_API_CONFORMANCE
SQL_OAC_LEVEL1

SQL_ODBC_SAG_CLI_CONFORMANCE
SQL_OSCC_COMPLIANT
### ODBC Functions Supported

<table>
<thead>
<tr>
<th>Function</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SQL_ODBC_SQL_CONFORMANCE</td>
<td>SQL_OSC_MINIMUM</td>
<td></td>
</tr>
<tr>
<td>SQL_ODBC_SQL_IIF</td>
<td>&quot;N&quot;</td>
<td></td>
</tr>
<tr>
<td>SQL_OJ_CAPABILITIES</td>
<td>0x0000</td>
<td>Driver does not support outer joins.</td>
</tr>
<tr>
<td>SQL_ORDER_BY_COLUMNS_IN_SELECT</td>
<td>&quot;N&quot;</td>
<td>driver does not support ORDER BY clause.</td>
</tr>
<tr>
<td>SQL_OUTER_JOINS</td>
<td>&quot;N&quot;</td>
<td></td>
</tr>
<tr>
<td>SQL_OWNER_TERM</td>
<td>&quot;&quot;&quot;</td>
<td>does not apply to driver or data source. Driver does not support owner names.</td>
</tr>
<tr>
<td>SQL_OWNER_USAGE</td>
<td>0x0000</td>
<td>does not apply to driver or data source. Driver does not support owner names.</td>
</tr>
<tr>
<td>SQL_POS_OPERATIONS</td>
<td>0x0000</td>
<td>does not apply to driver or data source. Driver does not support SQLSetPos.</td>
</tr>
<tr>
<td>SQL_POSITIONED_STATEMENTS</td>
<td>0x0000</td>
<td>does not apply to driver or data source. Driver does not support any positioned SQL statements.</td>
</tr>
<tr>
<td>SQL_PROCEDURE_TERM</td>
<td>&quot;&quot;&quot;</td>
<td>does not apply to driver or data source. Driver does not support procedures.</td>
</tr>
<tr>
<td>SQL_PROCEDURES</td>
<td>&quot;N&quot;</td>
<td>does not apply to driver or data source. Driver does not support procedures.</td>
</tr>
<tr>
<td>SQL_QUALIFIER_LOCATION</td>
<td>0x00</td>
<td>does not apply to driver or data source. Driver does not support qualifiers.</td>
</tr>
<tr>
<td>SQL_QUALIFIER_NAME_SEPARATOR</td>
<td>&quot;&quot;&quot;</td>
<td>does not apply to driver or data source. Driver does not support qualifiers.</td>
</tr>
<tr>
<td>SQL_QUALIFIER_TERM</td>
<td>&quot;&quot;&quot;</td>
<td>does not apply to driver or data source. Driver does not support qualifiers.</td>
</tr>
<tr>
<td>SQL_QUALIFIER_USAGE</td>
<td>0x0000</td>
<td>does not apply to driver or data source. Driver does not support qualifiers.</td>
</tr>
<tr>
<td>SQL_QUOTED_IDENTIFIER_CASE</td>
<td>0x00</td>
<td>does not apply to driver or data source. Driver does not support quoted identifiers.</td>
</tr>
<tr>
<td>SQL_ROW_UPDATES</td>
<td>&quot;N&quot;</td>
<td></td>
</tr>
<tr>
<td>SQL_SCROLL_CONCURRENCY</td>
<td>0x0000</td>
<td>does not apply to driver or data source.</td>
</tr>
<tr>
<td>SQL_SCROLL_OPTIONS</td>
<td>0x0000</td>
<td>does not apply to driver or data source.</td>
</tr>
<tr>
<td>SQL_SEARCH_PATTERN_ESCAPE</td>
<td>&quot;/&quot;</td>
<td>forward slash</td>
</tr>
</tbody>
</table>
ODBC Functions Supported

SQL_SERVER_NAME
"

SQL_SPECIAL_CHARACTERS
"#$/@"

SQL_STATIC_SENSITIVITY
0x0000; does not apply to driver or data source. Driver does not support SQLSetPos.

SQL_STRING_FUNCTIONS
0x0000; does not apply to driver or data source. Driver does not support any scalar string functions.

SQL_SUBQUERIES
0x0000; does not apply to driver or data source. Driver does not support subqueries.

SQL_SYSTEM_FUNCTIONS
0x0000; does not apply to driver or data source. Driver does not support scalar system functions.

SQL_TABLE_TERM
"data view record"

SQL_TIMEDATE_ADD_INTERVALS
0x0000; does not apply to driver or data source. Driver does not support timestamp intervals.

SQL_TIMEDATE_DIFF_INTERVALS
0x0000; does not apply to driver or data source. Driver does not support timestamp intervals.

SQL_TIMEDATE_FUNCTIONS
0x0000; does not apply to driver or data source. Driver does not support scalar date and time functions.

SQL_TXN_CAPABLE
SQL_TC_NONE; Transactions not supported.

SQL_TXN_ISOLATION_OPTION
0x0000; does not apply to driver or data source. Driver does not support transactions.

SQL_UNION
0x0000; does not apply to driver or data source. Driver does not support the UNION clause.

SQL_USER_NAME
""; does not apply to driver or data source.
### III — Report Format Facility

**Chapter 9. Introducing the Report Format Facility**  
131

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introducing Report Format Tables</td>
<td>132</td>
</tr>
<tr>
<td>RFT Statements</td>
<td>132</td>
</tr>
<tr>
<td>RFT Sections</td>
<td>133</td>
</tr>
<tr>
<td>How the Report Format Facility Processes RFTs</td>
<td>133</td>
</tr>
<tr>
<td>Storing RFTs</td>
<td>133</td>
</tr>
<tr>
<td>Standard RFTs</td>
<td>134</td>
</tr>
<tr>
<td>Report Exit Routines</td>
<td>134</td>
</tr>
</tbody>
</table>

**Chapter 10. Modifying and Creating RFTs**  
135

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Planning Your RFT</td>
<td>135</td>
</tr>
<tr>
<td>Deciding Whether to Modify or Create RFTs</td>
<td>135</td>
</tr>
<tr>
<td>Expanding Date Fields</td>
<td>135</td>
</tr>
<tr>
<td>Modifying Panels</td>
<td>136</td>
</tr>
<tr>
<td>Other Considerations</td>
<td>136</td>
</tr>
<tr>
<td>Creating an RFT</td>
<td>136</td>
</tr>
<tr>
<td>Creating a New RFT</td>
<td>137</td>
</tr>
<tr>
<td>Testing an RFT</td>
<td>139</td>
</tr>
<tr>
<td>Adding a Data-Collection Panel</td>
<td>142</td>
</tr>
<tr>
<td>Adding a Menu Panel</td>
<td>143</td>
</tr>
<tr>
<td>Modifying an RFT</td>
<td>143</td>
</tr>
<tr>
<td>Performance Considerations</td>
<td>143</td>
</tr>
<tr>
<td>Virtual Storage Requirements</td>
<td>144</td>
</tr>
<tr>
<td>Processing Time</td>
<td>144</td>
</tr>
</tbody>
</table>

**Chapter 11. RFT Statements**  
147

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Syntax and Rules</td>
<td>147</td>
</tr>
<tr>
<td>Notation Conventions</td>
<td>147</td>
</tr>
<tr>
<td>General Rules</td>
<td>148</td>
</tr>
<tr>
<td>Keywords and Keyword Values</td>
<td>148</td>
</tr>
<tr>
<td>Special Characters</td>
<td>149</td>
</tr>
<tr>
<td>Mixed Data</td>
<td>149</td>
</tr>
<tr>
<td>Prefixes</td>
<td>149</td>
</tr>
<tr>
<td>Variables</td>
<td>149</td>
</tr>
<tr>
<td>Dictionary Index Keys</td>
<td>149</td>
</tr>
<tr>
<td>Search Levels</td>
<td>149</td>
</tr>
<tr>
<td>Summary of RFT Statements</td>
<td>150</td>
</tr>
</tbody>
</table>

**Chapter 12. Definition Statements**  
153

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>EOD - EEOD</td>
<td>153</td>
</tr>
<tr>
<td>Usage Notes</td>
<td>153</td>
</tr>
<tr>
<td>EOF - EEOF</td>
<td>156</td>
</tr>
<tr>
<td>Usage Notes</td>
<td>156</td>
</tr>
<tr>
<td>FORMAT</td>
<td>157</td>
</tr>
<tr>
<td>Usage Notes</td>
<td>158</td>
</tr>
<tr>
<td>HEADING - EHEADING</td>
<td>160</td>
</tr>
<tr>
<td>Usage Notes</td>
<td>160</td>
</tr>
<tr>
<td>HFORMAT - EFORMAT</td>
<td>161</td>
</tr>
</tbody>
</table>
Chapter 13. Command Statements ...................................................... 167
CALL .......................................................... 167
Usage Notes 169
Examples 172
DHEADS .......................................................... 173
DO, EDO, LEAVE .................................................. 174
Usage Notes 182
Examples 182
EJECT .............................................................. 184
END ............................................................... 184
IF - ELSE - EIF .................................................... 185
Usage Notes 193
Examples 193
PUT ................................................................. 194
Usage Notes 200
Examples 203
SEARCH - ESEARCH ............................................. 205
Usage Notes 208
Examples 209
SET ................................................................. 210
Usage Notes 215
Examples 216
SETD ............................................................... 217
Usage Notes 222
Examples 225
SPACE ............................................................ 227
TAB ................................................................. 228

Chapter 14. Report Exit Routines ..................................................... 231
Types of Report Exit Routines ............................................................ 231
Service Functions 232
Controlling Report Exit Routines .................................................. 232
Data Output Exit Routines ............................................................. 233
Communication Control Block for the Data Output Exit .............. 233
Return Codes for Data Output Exits .............................................. 235
Restrictions for Data Output Exits ............................................... 235
CALL Command Exit Routines ...................................................... 235
Communication Control Block for CALL Command Exits ............ 236
Return Codes for CALL Command Exits .................................. 237
CALL Command Exit Restrictions .............................................. 237
Predefined CALL Command Exits .............................................. 237
Search and Count ................................................................ 238
Format Search Argument .......................................................... 239
Format Detailed Data ............................................................... 239
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Format Response Chains</td>
<td>240</td>
</tr>
<tr>
<td>Check “From” and “To” Dates</td>
<td>240</td>
</tr>
<tr>
<td>Format List Processor Data</td>
<td>241</td>
</tr>
<tr>
<td>Exit Routine Service Functions</td>
<td>243</td>
</tr>
<tr>
<td>Locate Service Function–X'D3'</td>
<td>243</td>
</tr>
<tr>
<td>Move Service Function–X'D4'</td>
<td>247</td>
</tr>
<tr>
<td>Variable Read Service Function–X'D9'</td>
<td>247</td>
</tr>
<tr>
<td>Variable Set Service Function–X'E2'</td>
<td>248</td>
</tr>
<tr>
<td>Write Service Function–X'E6'</td>
<td>249</td>
</tr>
</tbody>
</table>

**Chapter 15. Defining Output Destinations for Reports**

SYSOUT, DSNAME, and DDNAME. ................................................... 251

**Chapter 16. Alternative Methods of Running Reports**

Running Reports with SRCs ......................................................... 255
Running Reports in Batch Mode .................................................. 255
Batch Reports under MVS/ESA .................................................... 255
Writing to SYSPRINT ............................................................... 257
Introducing the Report Format Facility

**Programming Interface information**: The Report Format Facility part of this document contains information about Programming Interfaces that enable you to obtain the services of Tivoli Information Management for z/OS.

The Report Format Facility is a part of Tivoli Information Management for z/OS that helps you collect operational and management information about your organization. The Report Format Facility provides you with a standard set of reports, and you have the option to modify these reports or create your own. The Report Format Facility also allows you to print individual records and browse reports. Examples of reports produced with the Report Format Facility are provided in "Report Samples" on page 267.

**Note**

Starting with Tivoli Service Desk for OS/390 Version 1.2, support for the host graphics function of the Report Format Facility, which uses the Graphical Data Display Manager (GDDM), is no longer included. Customers requiring graphics reports can use the ODBC driver and a user-supplied, ODBC-enabled workstation application capable of producing graphic reports, or use Tivoli Decision Support for Information Management (5697-IMG).

To understand the Report Format Facility, you should be familiar with the following publications:

- **Tivoli Information Management for z/OS User’s Guide**
  Describes how to run the REPORT command to generate standard Tivoli Information Management for z/OS reports.

- **Tivoli Information Management for z/OS Problem, Change, and Configuration Management**
  Describes the Tivoli Information Management for z/OS online facilities that are available to enter and track problems, changes, and inventory. The standard reports that can be generated from data collected through those facilities are described in Standard Reports of this manual.

Typical uses of the Report Format Facility include:

- Printing detailed reports about various records in the database
- Listing outstanding problems or changes
- Displaying transactions on an hour-by-hour basis.
You can generate a report at any time, either interactively or as a batch job. You can also create a report and browse it on your workstation to check whether it contains accurate information or whether it is formatted properly.

### Introducing Report Format Tables

The Report Format Facility uses a report format table (RFT) to generate and format each report. An RFT contains a series of “statements” that tell the Report Format Facility what data you want to collect and how you want it formatted. Figure 11 shows a basic RFT; Figure 12 shows the resultant report.

```
/* SECTION TO PRINT ALL RECORDS WITH STATUS=OPEN */
SECTION
TITLE /* start the section title */
PUT COLUMN(1) VALUE(DATE &ZECDATE) /* report date */
PUT COLUMN(27) VALUE(LIST OPEN RECORDS) /* report title */
PUT COLUMN(1) VALUE(TIME &ZCTIME) /* report time */
PUT COLUMN(27) VALUE(------------------) /* underlines */
SPACE LINES(2)
ETITLE /* end the section title */
HEADING /* start column header */
PUT COLUMN(1) VALUE(RECORD ID) /* record id column header */
PUT COLUMN(15) VALUE(STATUS) /* status column header */
PUT COLUMN(30) VALUE(DESCRIPTION) /* description col hdr. */
PUT COLUMN(1) VALUE(---------) /* underscores for rec id */
PUT COLUMN(15) VALUE(------) /* underscores for status */
PUT COLUMN(30) VALUE(-----------) /* underscores for desc. */
EHEADING /* end column header */
SEARCH ARGUMENT(STAC/OPEN) /* for each record with */
/* status=open... */
PUT COLUMN(1) DATA(RNID/.) /* put record id in col 1 */
/* -identified by pword */
PUT COLUMN(15) DATA(STAC/.) /* put status in col 15 */
/* -identified by pword */
PUT COLUMN(30) DATA(!S0E0F) /* put description in col 30 */
/* -identified by sword index*/
ESEARCH /* loop back to SEARCH until */
/* all records processed */
ESSECTION
```

**Figure 11. A Basic RFT**

```
DATE 07/06/1999 LIST OPEN RECORDS
TIME 11:02:49 ------------------
RECORD ID STATUS DESCRIPTION
--------- ------ -----------
000000020 OPEN Sample description for record 20
000000028 OPEN Sample description for record 28
```

**Figure 12. The Printed Report**

### RFT Statements

Report Format Facility provides a comprehensive set of statements that give you extensive control over collecting data and formatting reports. The two types of RFT statements are:
Definition statements
Define segments of data that remain constant either for the whole RFT or for a section. TITLE and HEADING are definition statements.

Command statements
Control record processing such as retrieving, formatting, and printing. PUT and SEARCH are command statements.

Some statements also have keywords that increase the flexibility of the statement. The first PUT statement in Figure 11 on page 132 has two keywords: COLUMN (specifies the column where printing is to start) and VALUE (specifies what to print). The purpose and syntax of each RFT statement is described in detail in "RFT Statements" on page 147.

RFT Sections
An RFT must be divided into sections. This makes it easier for the Report Format Facility to process RFTs and allows you to structure them logically.

Note: Sorting a large number of records can create a storage problem; therefore, dividing an RFT into sections is one way to reduce storage requirements.

A simple RFT, such as the example in Figure 11 on page 132, needs only one section (defined by the SECTION and ESECTION statements). However, more complex RFTs can be split into several sections.

How the Report Format Facility Processes RFTs
The Report Format Facility processes each section of an RFT sequentially and independently. In general, the Report Format Facility keeps only the data needed to process the current section in virtual storage. Report Format Facility processes each section in at least two phases:

1. During the input phase, the Report Format Facility reads the section’s statements. The Report Format Facility parses and checks the syntax of each statement and processes definition statements. Then it converts all data from these statements for use in the next phase.

2. During the output phase, the Report Format Facility processes command statements and generates the report. The Report Format Facility searches the database, reads the records into virtual storage, and writes data to the output data set, which is the report.

After the Report Format Facility has processed one section, it carries some items, such as title, heading, and variable definitions, forward to the next section, and deletes the data that was already processed. The Report Format Facility then repeats the process on the next section of the RFT.

Storing RFTs
You store RFTs in partitioned data sets (PDSs). The RFT data definition (DDNAME) specified in the session-parameters member indicates to the Report Format Facility how to find a requested RFT. Refer to the description of the BLGPARMS RFTDS keyword in the Tivoli Information Management for z/OS Planning and Installation Guide and Reference for more information.
Standard RFTs

Tivoli Information Management for z/OS supplies a number of standard RFTs as a part of the Report Format Facility. These RFTs allow you to generate frequently required reports. If, however, the standard reports supplied with Tivoli Information Management for z/OS do not meet the needs of your organization, you can create your own RFTs by:

- Modifying the standard RFTs supplied with Tivoli Information Management for z/OS
- Writing new ones from scratch.

Standard reports are described in detail in "Standard Reports" on page 261. The rest of this book describes the RFT language and provides examples of typical RFTs.

Report Exit Routines

Report exit routines enable you to perform tasks that you cannot do using the Report Format Facility. During processing, the Report Format Facility gives temporary processing control to the report exit routine. The routine performs its task and, when finished, hands control back to the Report Format Facility.

A typical use of a report exit routine is to process data gathered by the Report Format Facility, for example, perform a calculation on the gathered data. The Report Format Facility then uses the processed data in the report.

Report exit routines also give you direct access to service functions used only by the Report Format Facility to process an RFT. The example in Figure 13 illustrates a series of RFT statements that uses all of the service functions available to the Report Format Facility.

```
PUT COLUMN(1) VALUE(LITERAL) /* Move output (X'D4') */
SET NAME(SWORD) VALUE(ISOEOF) /* Set variable (X'E2') */
PUT COLUMN(1) DATA(&SWORD.) /* Write output (X'E6') */
/* Read variable (X'D9') */
/* Locate Record Data (X'D3') */
```

Figure 13. An example of User Exit Services

Report exit routines are described in detail in "Report Exit Routines" on page 231.
Modifying and Creating RFTs

This chapter explains important factors that you need to know before you modify or create an RFT.

Creating a report involves two major tasks:
- Planning what you want in the report and how you want to present it
- Creating the RFT.

Planning Your RFT

First, decide the features and format you want for the report. You can look at examples of standard reports and at the examples in this book to get an idea of what the Report Format Facility can do. After deciding which features you want to include, you must decide whether to modify an existing RFT or to create a new one.

If your installation uses graphic character substitutions for the @, l, ¬, and ! characters, you need to use the substitution characters in any reports you create or modify. Review all the RFTs, including any supplied by Tivoli, that you intend to use to determine whether they need to be modified to use the graphic substitution characters. Be sure to use the graphic substitution characters in any new RFTs you create. Refer to the Tivoli Information Management for z/OS Planning and Installation Guide and Reference for more information on graphic substitution characters.

Deciding Whether to Modify or Create RFTs

If a standard report requires only a minor change, such as a different set of titles or an additional heading, you can modify a standard RFT instead of creating a new one. Often, you can successfully modify a report by adding or changing a few RFT statements. However, if you need a report that is completely different from any of the standard reports, you will probably have to design a new RFT.

Expanding Date Fields

The standard reports shipped with Tivoli Information Management for z/OS have expanded 10-character date fields to enable you to display dates with 4-digit years. The expanded date fields can accommodate any of the external date formats provided by Tivoli Information Management for z/OS. Date formats less than 10 characters are left-justified on the reports. If you have a customized external date format that is longer than 10 characters, the date data is truncated. Ensure that the date fields on your reports have sufficient room to display data if you decide to change the format of date presentation.

The date format displayed in your reports is dependent on the external date format specified in the session-parameters member by the BLGPARMS macro. Users can, however, override this format in the user profile by selecting another date format from a list of formats.
Planning Your RFT

supported by Tivoli Information Management for z/OS. (The option to override the date
format is provided in the user and database defaults option in the user profile.) The Tivoli
Information Management for z/OS panels and program interface data tables can support the
entry and display of any of the 22 supported external date formats. For details on how the
external date format is set, refer to the discussion of the BLGPARMS macro in the Tivoli
Information Management for z/OS Planning and Installation Guide and Reference.

Note: The standard RFTs shipped with Tivoli Information Management for z/OS support
expanded (10–character) date fields; however, the following RFT requires additional
modification on your part if you intend to use it. Instructions on what modifications
to make are provided in the RFT.

BLMZZ31

Modifying Panels
Creating a User RFT report is usually part of a larger task, such as creating a new record
type, that involves modifying Tivoli Information Management for z/OS panels. When you
create a new report, you probably want to change the various dialogs that can result in the
printed reports as well as give your users the option to select the report directly.

You can also add a new panel to collect data for the new report. (See page “Adding a
Data-Collection Panel” on page 143 for an example of such a panel.) The Tivoli Information
Management for z/OS Panel Modification Facility Guide contains detailed instructions for
creating and modifying Tivoli Information Management for z/OS panels to modify the report
panel flow.

Other Considerations
Before you create an RFT, you must know this information:

■ How to use your facility’s system editor, such as ISPF/PDF, to create or copy and edit a
partitioned data set member
■ The name of, and have access to, the RFT partitioned data set
■ The factors that affect system performance. These are described in “Performance
Considerations” on page 143.

Creating an RFT
The RFT you create must be a member of the RFT partitioned data set so that Tivoli
Information Management for z/OS can find it when you request the report. If you want to
create a new RFT, create a new RFT member in the partitioned data set. If you want to
modify a standard RFT, you can either copy the standard RFT member to a new member in
the partitioned data set or modify the existing member.

You can build the new RFT in a data set that is not currently a member of the partitioned
data set. However, you must add the new data set as a member of the RFT partitioned data
set before you request the report. Refer to the Tivoli Information Management for z/OS
Planning and Installation Guide and Reference for detailed instructions on defining and
using the RFT data set.
Creating a New RFT

Assume that your RFT data set is YOUR.RFTS.CUSTOM and that you want to create an RFT named ASSIGNEE. This report prints a title based on data specified by conditional processing.

Begin with ISPF and edit YOUR.RFTS.CUSTOM(ASSIGNEE).

In the member, type the RFT statements. The first statement in the RFT is SECTION. The next statement is the TITLE statement. Only one TITLE statement is allowed between a SECTION statement and an ESECTION statement, but you can change the title by using conditional processing. The IF and ELSE statements perform conditional processing.

Add EIF statements to end the conditional processing. Add an ETITLE statement to end the title. Add HEADING and PUT statements to write headings for the columns on each report.

When adding PUT statements for date columns, ensure that your columns are wide enough to accommodate the external date formats that can be selected by users, as well as the date format specified on the BLGPARMS macro DATEFMT keyword. For example, if your users will be entering dates in MM/DD/YYYY format, your date columns should accommodate 10 characters.
Next, add the SEARCH argument to retrieve the records for DWHITE and write any data in the report. Include a SPACE statement to space one line between records. Specify MINLINES as 2 so that, if there are fewer than two lines on the page, a new page is started. Be sure to include an ESEARCH statement to end the SEARCH argument. Then, add an EJECT statement to repeat the report heading for your next SEARCH argument. Add the SEARCH argument to retrieve the records for JJSMITH and write any data in the report. Include a SPACE statement to space one line between records for JJSMITH. Add ESEARCH and ESECTION statements. Save the member. Then start Tivoli Information Management for z/OS to test the RFT.
"RFT Statements" on page 147 describes each RFT statement in detail.

Testing an RFT

You can test the RFT by issuing the REPORT command and requesting the report. If you have not modified the panel flow, request the new report as follows:

Enter REPORT from any point in a Tivoli Information Management for z/OS dialog. Tivoli Information Management for z/OS displays the Report Entry panel.

Select USER RFT; this is an installation-defined report. Type 8 and press Enter.

```
EDIT ---- YOUR.RFTS.CUSTOM(ASSIGNEE) - 01.00 ----------------- COLUMNS 001 072
COMMAND ===>
SCROLL ===> PAGE
****** TOP OF DATA ****************************
'----' PUT COL(32) DATA(STAC/.) /* CURRENT STATUS */
'----' PUT COL(42) DATA(PERA/.) /* ASSIGNEE */
'----'
'----' SPACE LINES(1) EXECUTE(C) MINLINES(2)
'----'
'----' ESEARCH /* END OF SEARCH */
'----' ESECTION /* END OF SECTION */
****** BOTTOM OF DATA ****************************
```

Enter the member name of the RFT you created. For this example, the name is ASSIGNEE.
After you enter the member name of the RFT, report processing begins if you have defined the report output destination in your profile. If not, Tivoli Information Management for z/OS displays the Standard Report Output Destination panel so you can enter destination information. See “Defining Output Destinations for Reports” on page 251 for more information about output destinations.

If your destination information did not specify BROWSE=YES for the report output data set, the system returns to the panel where you issued the REPORT command and displays the appropriate messages after report processing ends.

If your destination information did specify BROWSE=YES for the report output data set, the report output data set is presented in browse mode after report processing ends.

If your output destination is YOUR.REPORTS.STANDARD and your destination information specifies BROWSE=YES, this panel appears.
Note: This report output data set shows use of a 4-digit year external date format (MM/DD/YYYY). The external date format is set by your Tivoli Information Management for z/OS program administrator, although you can override it in your user profile through the user and database defaults option.

Scroll down to see the next part of the report.

Continue scrolling to see all of the report.
Adding a Data-Collection Panel

If you create an RFT that requires data to be entered while it is processing, you must create a panel to collect this data. The collected data might include start and end dates or the name of the person requesting the report. Add the panel to the report panel flow so that it follows the Report Format Table panel (BLG6RFTN), unless you modify the report panel flow. If you modify the panel flow, add the data-collection panel to the flow so that it follows the selection panel you created. For more information on modifying panels, refer to the Tivoli Information Management for z/OS Panel Modification Facility Guide.

This is an example of a data-collection panel.
Adding a Menu Panel

If your new RFT is going to be used by more than one person (or you have created several new RFTs), consider modifying the panel flow so that users can easily select the RFTs from a suitable menu panel. For more information on modifying panels, refer to the Tivoli Information Management for z/OS Panel Modification Facility Guide.

Modifying an RFT

The steps for modifying an RFT are the same as the steps for creating a new RFT. However, instead of typing all the RFT statements into a member in an edit session, you begin by copying an RFT. Then you change statements to achieve the processing and report format you want.

To decide which RFT to use as a base, see "Standard Reports" on page 261, for a list of the RFTs shipped with Tivoli Information Management for z/OS. An alternative way to decide which RFT to use as a base is to look at reports from each standard RFT and choose the report that most nearly meets your needs. Then look on the second line of the report title in the upper right, just below the word page. Printed there is the member name of the RFT that produced the report.

"RFT Examples" on page 289, contains a progressive scenario that illustrates first creating the Periodic Change Status RFT (BLMZZ31), then modifying it to accumulate and print risk-assessment totals. That appendix also contains examples of RFTs that produce the following reports:

- Changes and related activities
- Problem record description
- Problems entered overnight
- Problems that exceed outage standards
- Problems with three or more assignees.

Performance Considerations

When you write an RFT, you must be aware of the factors that affect system performance. These are divided into two categories:

- Virtual storage requirements
- Processing time.

Virtual Storage Requirements

The amount of virtual storage required to process a report depends on the following factors (listed in the order in which they occur within an RFT):

- The number of command statements in each section of the RFT. The Report Format Facility converts each command statement into an internal representation of that command.
- The number of nested SEARCH statements. For each nested SEARCH statement, Tivoli Information Management for z/OS reads a record into storage.
- The presence of the SORT keyword on the SEARCH statement. The SORT keyword influences both the output-phase CPU processing time and the amount of virtual storage required to process the report.
The average number of fields, the amount of freeform text, and the size of the journal
history for the records processed by the report. This determines the amount of storage
required to contain the records.

The number of CALL command exits contained in an RFT section and the amount of
storage each requires. The Report Format Facility must load each exit routine and create
a communication control block for it. In addition, storage is required for processing the
exit routine.

The amount of storage required by any user data output exit. This includes the size of
the load module, the communication control block, and any additional storage the exit
requires.

The number and length of any user variables specified in the RFT. Each variable
requires 16 bytes plus the amount of storage specified (or the default amount) for the
data it is to contain.

The amount of string data specified with VALUE keywords. Each data string must be
contained in storage.

The presence of TITLE and HEADING definition sections. The TITLE and HEADING
definitions reserve a minimum of 2K bytes of storage to contain their internal command
representations and data strings.

The presence of an end-of-file processing section. The end-of-file definition reserves a
minimum of 2K bytes of storage for its internal command representations and data
strings.

To conserve virtual storage, do not include the end-of-file processing section in any
report section other than the last.

The presence of history or text format statements and the number of times they are
redefined from one RFT section to the next.

Processing Time

The time required to process a report depends on the following factors (listed in the order in
which they occur within an RFT).

The number of statements in the RFT directly influences both the input-phase and
output-phase CPU processing time.

The number of dictionary index keys used in the RFT influences the input-phase CPU
processing time.

The number of nested SEARCH statements influences the output-phase CPU processing
time.

The presence of the SORT keyword on the SEARCH statement has varying effects:

- Perform a sort for a search level only if you specify the SORT keyword on that
  SEARCH statement. Thus, report performance can be significantly improved at the
  expense of having unsorted records by removing the SORT keyword from all
  SEARCH statements (or at least limiting the number of multiple sorts).

- When you do specify the SORT keyword, the Report Format Facility reads each
  database record identified in the associated search-results list, extracts the data
  identified by the sort fields, and builds sort records using this data. It then attaches
  the IBM® Sort/Merge Licensed Program (or an equivalent) to sort these records.
After the records are sorted, the Report Format Facility reads the records one at a time, and the records of the associated database are again read into storage and processed by the Report Format Facility.

You can specify in the session-parameters member the unit names and track size to be used for allocation of the SORTIN, SORTOUT, and SORTWORK data sets used by SORT/MERGE. For optimum performance, specify a virtual input/output (VIO) unit name for the SORTIN and SORTOUT data sets and a non-VIO DASD unit name for the SORTWORK data set. Refer to the Tivoli Information Management for z/OS Planning and Installation Guide and Reference for more details about the session-parameters member.
This chapter describes the syntax and general rules that apply to RFT statements.

Syntax and Rules

Basic Syntax of an RFT Statement

[ label: ] statement [ keyword (value) ] [ keyword (value) ].../* comment */

label

is an optional 1- to 8-character single-byte character set (SBCS ¹) alphanumeric label you assign to this statement, followed by a colon. It can begin in any column, but it must be first. The Report Format Facility ignores labels; they are strictly for your convenience.

statement

is the name of the command or definition statement (up to 15 alphabetic characters). It can begin in any column (1 - 71). Blanks between the colon following a label and a statement are optional.

keyword (value)

is a parameter that consists of a predefined word (up to 15 alphabetic characters) and its associated data (value). If you include the keyword, you must also specify the value, in parentheses. Keywords are discussed further in “Keywords and Keyword Values” on page 148.

/* comment */

is an optional explanation added to or interspersed among the statements. Comments must be preceded by a slash-asterisk combination (/*), and ended by an asterisk-slash combination (*/). Comments can contain any characters except the asterisk-slash combination (*/).

Notation Conventions

The following conventions are used to describe keywords and values:

- Braces { } represent alternatives; you must choose one.
- Brackets [ ] represent optional entries; you can choose one or none.
- A vertical bar (|) indicates alternatives; you must choose one.

¹. If your installation uses only SBCS data (that is, EBCDIC data) and does not use DBCS data, you do not need to be concerned with references to SBCS, DBCS, or mixed data.
Ellipsis (...) indicates that you can repeat the keywords.

An underscored value represents the default. If you do not specify a keyword and its associated value, the Report Format Facility uses the default value.

In a statement name or a keyword, uppercase letters designate the minimum truncation, and the rest of the name is in lowercase.

Three vertical dots between two statements indicate that other statements can intervene.

Enter uppercase values as shown; replace lowercase values with your own data.

**General Rules**

The following general rules apply:

- At least one blank is required between a statement name and the first keyword. A blank between the keyword and the left parenthesis is optional; however, no blanks are allowed between the left parenthesis and the beginning of the keyword value.

- When you code multiple keywords on a statement, you can separate the keywords by blanks, but this is not required.

- The same keyword may appear on more than one statement, but the meaning of the keyword can vary from statement to statement.

- A null statement is one that contains only a label, only a comment, a label and a comment, or blanks. The Report Format Facility ignores null statements.

- The alphabetic characters on a statement can be in upper, lower, or mixed case. However, when the Report Format Facility reads the statement, it converts all data not enclosed in single quotation marks to uppercase. See "Keywords and Keyword Values" for more information on the use of single quotation marks.

- Use columns 1 through 71 for statement data. Each statement can include 1 to 10 physical records.

- Use column 72 to continue a statement from one line to the next. Use a nonblank SBCS character, including the shift in (SI) character, to note the continuation. A component byte of a double-byte character set (DBCS) character in this column is not valid.

  When you want to continue a statement, additional keywords or their values (enclosed in parentheses) can begin in any column of the next physical record. However, if a keyword or value is started on one physical record and goes up to column 71, the continuation of that keyword or value must begin in the first column of the next physical record.

- Columns 73 through 80 can be used for sequence numbers. To help identify erroneous statements, the Report Format Facility appends to many of the error messages whatever data is in columns 73 through 80 of the last physical record of a statement. If the data in columns 73-80 is not valid mixed data, SBCS periods are printed in the error message.

**Keywords and Keyword Values**

Some RFT statements can include keywords that provide additional information to the Report Format Facility. The values on these keywords provide a symbolic representation of record data. Some keywords are optional; others are required. Some keyword values can contain mixed data.

Keyword values cannot be used interchangeably; each keyword has a specific syntax.
Depending on the type of keyword, a keyword value can be a digit, an alphabetic or special character, a character string (with or without imbedded blanks), a variable, a prefix, a dictionary index key representing a p-word or s-word, or a combination of these.

**Special Characters**

If the following SBCS characters are part of a value, you must enclose them in single SBCS quotes, as shown:
- Left parenthesis, '('
- Right parenthesis, ')'
- Exclamation point, '!''
- Ampersand, '&'
- Leading blanks, '
- Single quote, `'`

**Mixed Data**

Most keywords that allow character string values allow mixed data. (The PAD keyword value that can be specified in the PUT and CALL command statements is an exception. It must be a single SBCS character.)

The statement and keyword combinations that allow mixed data values are:

<table>
<thead>
<tr>
<th>Statement</th>
<th>Keyword</th>
</tr>
</thead>
<tbody>
<tr>
<td>FORMAT</td>
<td>VALUE</td>
</tr>
<tr>
<td>SEARCH</td>
<td>ARGUMENT</td>
</tr>
<tr>
<td>IF</td>
<td>VALUE</td>
</tr>
<tr>
<td>PUT</td>
<td>VALUE</td>
</tr>
<tr>
<td>CALL</td>
<td>INPUT</td>
</tr>
<tr>
<td>SET</td>
<td>VALUE</td>
</tr>
<tr>
<td>SETD</td>
<td>VALUE</td>
</tr>
<tr>
<td>DO</td>
<td>VALUE</td>
</tr>
</tbody>
</table>

**Prefixes**

Prefixes are short, standard words that represent different types of data and enable you to create keyword search arguments. A prefix consists of 1 to 5 alphanumeric characters followed by a slash (/) or an underscore (_). The first character must be alphabetic. For example, “STAC/” represents the status of a problem.

When it processes an RFT, Tivoli Information Management for z/OS uses prefixes to search the database, to identify sort fields, and to retrieve data from records. The prefix must be followed by a period (.) unless specific data is appended to it. Prefixes are listed with other field characteristics in "Field Characteristics" on page 319.

**Variables**

Variables are represented by a name consisting of an ampersand (&) followed by 1 to 8 SBCS alphanumeric characters, such as &ABC. See the description of the NAME keyword on the SET command statement ("SET" on page 210) and the description of the parameters on the BLGOXCNT call exit (on page 238) for exceptions to the use of the ampersand with variable names. Either a period or a space can be used as a variable delimiter.

The SET command statement defines the data represented by a variable name. The data can be null (no data) or 1 to 255 characters. The Report Format Facility substitutes the data for the variable name during the output phase of report processing. However, it substitutes only one level of value for the variable name. For example, if the current value of variable &ABC is &DEF (also a variable name), and the &DEF variable has a current value of GHI,
the Report Format Facility substitutes &DEF for &ABC, not the second-level value of GHI. If you are placing variables within mixed data, the variables must also contain mixed data.

If you do not want a character string containing an ampersand to be considered a variable name, you must enclose the ampersand (or the ampersand and part or all of the character string) in single quotation marks (‘&ABC’).

You can concatenate data to the left or right of a variable name. For example, the variable name 123&ABC has data concatenated to the left. When you concatenate data to the right of a variable name, you must put a period immediately after the variable name (&ABC.123). If a period is part of the data, you must put two periods after the variable name (&ABC..123). In data substitution, the Report Format Facility removes the first period and appends the remaining data to the substituted data. When one or more blanks immediately follow the variable name (&ABC 123), the period is optional. A shift out (SO) character can also be used immediately after the variable name (&ABC<W1W2W3>).

Variable names and the substituted data can be predefined by the Report Format Facility, or you can define them during Tivoli Information Management for z/OS installation. Variables defined by the Report Format Facility have “Z” as the first character after the ampersand. Most predefined variables are read-only (that is, they cannot be changed), but some are read/write. See "Predefined Variables" on page 363 for a complete list.

**Dictionary Index Keys**

Dictionary index keys identify p-words or s-words that are in the Tivoli Information Management for z/OS dictionary data set. The p-words and s-words are associated with data fields. Dictionary index keys are of two types:

- A prefix index, represented by a “P”, followed by 4 hexadecimal characters (for example, P0A49)
- An s-word index, represented by an “S”, followed by 4 hexadecimal characters (for example, S0A49).

You must prefix a dictionary index key with an exclamation point (!) for the Report Format Facility to recognize it, for example, !S0A49. If you do not want a character string containing an exclamation point to be considered a dictionary index key, you must enclose the exclamation point (or the exclamation point and part or all of the character string) in single quotation marks (‘!S0A49’).

The Report Format Facility substitutes the p-word or s-word associated with a dictionary index key for that dictionary index key during input-phase processing. Therefore, you cannot change the dynamics of dictionary index keys during output-phase processing. When the Report Format Facility substitutes a prefix for a prefix index, it automatically appends a period to the prefix as part of the substitution process.

If you concatenate data to the left of a dictionary index key (~!S0A49), it has no effect on the substitution of prefixes or s-words. However, concatenation of data to the right of a dictionary index key is not permitted; you must put at least one blank or a closing right parenthesis after the dictionary index key.

Because the Report Format Facility does not have to read the dictionary data set for prefixes as it does for dictionary index keys, consider using prefixes instead of dictionary index keys wherever possible. See "Field Characteristics" on page 319 for a list of the prefixes, prefix indexes, s-word indexes, and the data fields that each represent.
Search Levels

Each section of a report can include from one to ten search levels.

The search requested on the Tivoli Information Management for z/OS REPORT command is the level-zero search. If an RFT is a print-record RFT, the PRINT command provides the level-zero search. The results of this search become the basis for subsequent search levels in the RFT.

The SEARCH and ESEARCH command statements specify level-one through level-nine searches. During the output phase, the Report Format Facility searches the database using the argument specified on the SEARCH statement. The Report Format Facility processes each command statement between the SEARCH and ESEARCH statements once for each record retrieved. In other words, the Report Format Facility loops between SEARCH and ESEARCH until it processes all records. See “SEARCH - ESEARCH” on page 205 for more information on search processing.

For each record retrieved at the outer level (level 1, for example), Tivoli Information Management for z/OS performs a search at the next inner level (level 2) and processes the resulting records according to the command statements defined for that inner level (level 2). Looping and nested searches are illustrated in the example on page 206.

Summary of RFT Statements

Table 1 lists each RFT statement, identifies it as either a command statement or a definition statement, and lists the page number where the statement is described.

<table>
<thead>
<tr>
<th>RFT Statement</th>
<th>Command Statement</th>
<th>Definition Statement</th>
<th>See Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALL</td>
<td>X</td>
<td></td>
<td>167</td>
</tr>
<tr>
<td>DHEADS</td>
<td>X</td>
<td></td>
<td>173</td>
</tr>
<tr>
<td>DO, EDO, LEAVE</td>
<td>X</td>
<td></td>
<td>174</td>
</tr>
<tr>
<td>EJECT</td>
<td>X</td>
<td></td>
<td>184</td>
</tr>
<tr>
<td>END</td>
<td>X</td>
<td></td>
<td>184</td>
</tr>
<tr>
<td>EOD - EEOD</td>
<td></td>
<td>X</td>
<td>153</td>
</tr>
<tr>
<td>EOF - EEOF</td>
<td></td>
<td>X</td>
<td>156</td>
</tr>
<tr>
<td>FORMAT</td>
<td></td>
<td>X</td>
<td>157</td>
</tr>
<tr>
<td>HEADING - EHEADING</td>
<td></td>
<td>X</td>
<td>160</td>
</tr>
<tr>
<td>HFORMAT - EFORMAT</td>
<td></td>
<td>X</td>
<td>161</td>
</tr>
<tr>
<td>IF - ELSE - EIF</td>
<td>X</td>
<td></td>
<td>185</td>
</tr>
<tr>
<td>PUT</td>
<td>X</td>
<td></td>
<td>194</td>
</tr>
<tr>
<td>SEARCH - ESEARCH</td>
<td>X</td>
<td></td>
<td>205</td>
</tr>
<tr>
<td>SECTION - ESECTION</td>
<td></td>
<td>X</td>
<td>161</td>
</tr>
<tr>
<td>SET</td>
<td>X</td>
<td></td>
<td>210</td>
</tr>
<tr>
<td>SETD</td>
<td>X</td>
<td></td>
<td>217</td>
</tr>
<tr>
<td>SPACE</td>
<td>X</td>
<td></td>
<td>227</td>
</tr>
</tbody>
</table>
### Table 1. Summary of RFT Statements (continued)

<table>
<thead>
<tr>
<th>RFT Statement</th>
<th>Command Statement</th>
<th>Definition Statement</th>
<th>See Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>TAB</td>
<td>X</td>
<td></td>
<td>228</td>
</tr>
<tr>
<td>TFORMAT - EFORMAT</td>
<td></td>
<td>X</td>
<td>163</td>
</tr>
<tr>
<td>TITLE - ETITLE</td>
<td></td>
<td>X</td>
<td>163</td>
</tr>
</tbody>
</table>
The Report Format Facility has two types of statements:

- Definition statements, which define segments of data that remain constant either for the whole RFT or for a section
- Command statements, which control record processing such as retrieving, formatting, and printing.

This chapter describes the definition statements in detail. The statements are listed in alphabetical order.

**EOD - EEOD**

The end-of-data definition statement defines what the Report Format Facility must do each time it reads and processes the last database record, or when it finds an END command statement for a given search level. The EOD statement begins an end-of-data procedure, and the EEOD statement ends the end-of-data procedure.

### Syntax of EOD - EEOD

```
EOD [ Execute {C|U} ]
  ...
  ...
EEOD
```

**Execute**

indicates whether end-of-data applies if no records are processed for the corresponding level.

- **C** requests conditional processing; end-of-data is bypassed if no records are processed.  
  - C is the default.

- **U** requests unconditional processing; end-of-data processing occurs even when no records are processed for the corresponding search level.

**Usage Notes**

- You cannot define end-of-data for search level zero. Within each search level other than level zero, you can define end-of-data only once.

- You must specify end-of-data within the level to which it applies (between corresponding SEARCH and ESEARCH command statements). However, it is only specified within a conditional processing section (between an IF and an EIF) if you put the entire search level within the conditional section.
The only valid command statements that you can specify between EOD and EEOD are CALL, DO, DHEADS, EDO, EIF, EJECT, ELSE, IF, PUT, SET, SETD, SPACE, and TAB. The Report Format Facility processes these statements each time end-of-data occurs.

**Note:** If you use a CALL or PUT statement within the end-of-data definitions to obtain data from a database record, no data is found because no record is in storage at the time end-of-data is processed.

End-of-data can contain multiple command statements. The number of statements that can appear between an EOD and an EEOD statement is not limited, except by storage.

**Figure 15** illustrates valid end-of-data sections. **Figure 16 on page 155** illustrates end-of-data sections that are not valid.
The example “Calculating Totals for Each Report Section (EOD)” on page 292 illustrates additional uses of the EOD statement.

Figure 17 is an example of how to use the EOD - EEOD definition statement. This RFT searches all closed problems that were last modified in 1997. The elapsed time from when a record was created to when it was last modified is accumulated across all of these records. After the last record is processed, the statements between EOD and EEOD are processed to compute the average elapsed time per record. If the last modification for each record changed the status to closed, then this average represents the average lifetime of the records retrieved.
EOF - EEOF

The end-of-file definition statement defines what the Report Format Facility must do after it processes the last section of the RFT. The EOF statement begins an end-of-file definition. The EEOF statement ends the end-of-file definition. The EOF and EEOF statements must be paired within a section.

Syntax of EOF - EEOF

```
EOF [ Execute {C|U}]  EEOF
```

<table>
<thead>
<tr>
<th>Execute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>requests conditional processing; end-of-file is bypassed if no records are processed. C is the default.</td>
</tr>
<tr>
<td>U</td>
<td>requests unconditional processing; end-of-file processing occurs even if no records are processed for any of the report sections.</td>
</tr>
</tbody>
</table>

Usage Notes

- Within the RFT, define end-of-file only once, but it can be in any section (preferably the last because the end-of-file statements remain in storage until the Report Format Facility processes the last section).
The only command statements that are valid between EOF and EEOF are CALL, DHEADS, EIF, EJECT, ELSE, IF, PUT, SET, SETD, SPACE, and TAB. The Report Format Facility processes these statements when an end-of-file occurs.

**Note:** If you use a CALL or PUT statement within the end-of-file definitions to obtain data from a database record, you will find no data because no record is in storage when an end-of-file occurs.

- End-of-file can contain multiple command statements. Storage is the only limit on the number of statements that can appear between an EOF and an EEOF statement.
- End-of-file is applicable to an entire report; you must specify it within a section, but not within any other pair of statements.

The example “Building a Summary Line (EOF)” on page 293 illustrates the use of the end-of-file statements.

**FORMAT**

The FORMAT (F) definition statement defines the format for history or freeform text data. You place FORMAT statements between HFORMAT/TFORMAT and EFORMAT statements. See “HFORMAT - EFORMAT” on page 161 and “TFORMAT - EFORMAT” on page 163.

You use FORMAT statements to define which history or freeform text control data items you want to include with the history or freeform text data and the sequence in which you want these items to appear.

### Syntax of FORMAT

```
Format {
  History {
    (prefix )
    { (dictionary index key) }
    }
  }

  Text {
    (prefix )
    { (prefix index) }
    }

  Value{(string)}
}
```

- **History** identifies history data to be formatted.
- **prefix** is a 1- to 5-character alphanumeric code followed by a slash (/) or an underscore (_). The first character must be alphabetic. A period must follow the slash or underscore to retrieve data from the record. The prefix identifies the history control data item.
- **dictionary index key** is !P or !S, followed by 4 hexadecimal characters, which identify a p-word or an s-word in the dictionary data set associated with the history control data item.
  
  See “Keywords and Keyword Values” on page 148 for a complete description of prefixes and dictionary index keys.
- **Text** identifies freeform text control information to be formatted.
prefix

is a 1- to 5-character alphanumeric code followed by a slash (/) or an underscore (_). The first character must be alphabetic. A period must follow the slash or underscore to retrieve data from the record. The prefix identifies the freeform text control data item.

prefix index

is !P, followed by 4 hexadecimal characters, which identify a prefix in the dictionary data set.

Value

identifies user-specified data to be formatted.

string

is any user-specified data. If the following characters are considered data, you must enclose them in single quotes:

- Left parenthesis, '('
- Right parenthesis, ')'
- Exclamation point, '!
- Ampersand, '&'
- Leading blanks, '

To have a single quote appear as data, you must code two of them. The Report Format Facility removes the extra quote during the input phase of processing. The statements in Figure 18 illustrate how to code a single quote.

```
FORMAT VALUE('SINGLE QUOTE: ''')   */ SINGLE QUOTE: ' */
FORMAT VALUE('''')                */ '' */
```

*Figure 18. How to Code a Single Quote*

The length of a string of data is limited by the maximum continuation for the statement. (Each statement can consist of ten 71-character lines.)

Length

indicates the number of characters to be placed on the output line.

nnn

is a decimal number between 1 and 255. The Report Format Facility truncates items longer than nnn. If the data item is shorter than the length specified here, it is padded on the right with blanks.

Usage Notes

- For each piece of data that you format, you must code a HISTORY, TEXT, or VALUE keyword. You can use the HISTORY keyword only on those FORMAT statements that appear between an HFORMAT and an EFORMAT statement. Likewise, you can use the TEXT keyword only on those FORMAT statements that appear between a TFORMAT and an EFORMAT statement. You can use the VALUE keyword with the FORMAT statement for either the HFORMAT or TFORMAT statement. However, you cannot use variables on the FORMAT statement.

- Use multiple FORMAT statements to include multiple control data items on the history or freeform text data formats. (For example, to include date, time, and user ID in the history format requires three FORMAT statements using the HISTORY keyword.)
Storage is the only limit on the number of FORMAT statements that can be included between an HFORMAT or TFORMAT statement and its corresponding EFORMAT statement.

To get all of the history data or all of the freeform text for a particular item, specify horizontal or vertical iteration on the PUT or CALL statements.

- On the TEXT keyword, the Report Format Facility accepts only the following prefixes and their associated prefix indexes:
  
  **DATM/**
  - Date the freeform text line was last modified.

  **TIMM/**
  - Time the freeform text line was last modified.

  **USER/**
  - TSO user ID of the user who last modified the freeform text line.

  **CLAE/**
  - Privilege class of the user who last modified the freeform text line. This is not part of the control data in the Tivoli-supplied panels; you must modify the appropriate control panel. For information on how to modify control panels, refer to the *Tivoli Information Management for z/OS Panel Modification Facility Guide*.

- You can use the VALUE keyword to define the separation between the data items included in the data format. For example, if you want “DATE - history data” to appear for every history item, specify a FORMAT statement with a VALUE(’ - ’) keyword to make “ - ” appear between the date and the history data item.

- **HFORMAT** data is used only for processing a PUT or CALL statement that includes a HISTORY keyword.

- **TFORMAT** data is used only for processing a PUT or CALL statement that includes a TEXT keyword.

```sql
HFORMAT /* Format common insert */
  FORMAT VAL(ASSD:) LEN(6) /* Insert 'ASSD: ' */
  FORMAT HISTORY(DATM/.) LEN(11)
  FORMAT HISTORY(USER/.) LEN(9)
EFORMAT

TFORMAT /* Text data format */
  FORMAT VAL(TXT:) LEN(5) /* Insert 'TXT: ' */
  FORMAT TEXT(DATM/.) LEN(11)
  FORMAT TEXT(USER/.) LEN(9)
EFORMAT

SEARCH ...
  IF HISTORY(PERA/.) OP(=) VAL(&ZIFDATA) /* If assignee name */
    PUT ...
      PUT HISTORY(PERA/.) OP(V) COL(47) /* Formatted history insert */
    IF TEXT(!S0E07) OP(=) VAL(&ZIFDATA)
      PUT ...
        PUT COL(30) TEXT(!S0E07) OP(H) /* Formatted text */
    EIF
  EIF
ESEARCH
```

Figure 19. Example Using FORMAT Statements
The HEADING (HE) statement begins the definition of report headings. You can define headings to be report subtitles or to identify columnar data in a report. The EHEADING (EH) statement ends the definition of report headings. The HEADING and EHEADING statements must be paired within a section.

### Syntax of HEADING - EHEADING

```
HEading
.
.
.
EHeading
```

### Usage Notes

- A HEADING statement followed immediately by an EHEADING statement eliminates a previously defined heading.
- When you define headings, the Report Format Facility writes them at the top of each page immediately after the report title. You can also write headings anywhere on the page by using the DHEADS command statement.
- Within a section, you can define headings only once, but you can redefine them in a subsequent section.
- The only difference between titles and headings is that you can print a heading anywhere on a page, but a title only appears at the top of a page. Except for that difference, you can use a title or heading definition, because both are optional.
- Like titles, headings are applicable to an entire section. You can specify them only between SECTION and ESECTION statements.
- The only command statements that are valid between HEADING and EHEADING are CALL, DO, EDO, EIF, ELSE, IF, PUT, SET, SETD, SPACE, and TAB. These statements are processed each time headings are required.

**Note:** If you use a CALL or PUT statement within the heading definitions to obtain data from a database record, the resulting data might not be what you expect: the record from which that data is taken is the current record meeting the search-level criteria active at the time the headings are processed. To ensure that you display the correct data, set variables outside the headings and use them with the PUT statements, or provide conditional processing within the titles based on the current setting of the &ZCURLVL variable.

- Headings can contain multiple command statements. Storage is the only limit on the number of statements that can appear between a HEADING and an EHEADING statement.
The example "Defining Titles and Headings" on page 289 illustrates the use of the heading definitions.

## HFORMAT - EFFORMAT

The HFORMAT (HF) definition statement begins the definition of the history data format. You can define the format of the history and freeform text data (used by the PUT and CALL command statements) for a report. Substringing (FOR, FROM, WFOR, WFROM keywords) is ignored on the PUT statement with the HISTORY keyword when HFORMAT is active. The EFORMAT (EF) statement ends the history data format definition.

An EFORMAT statement must follow an HFORMAT statement. Only FORMAT statements can be specified between them. See “FORMAT” on page 157 for more information on the FORMAT definition statement.

### Syntax of HFORMAT - EFFORMAT

```
HFormat
  ...
  ...
EFormat
```

See Figure 19 on page 159 for an example of the format of the HFORMAT - EFFORMAT definition statement.

### Usage Notes

- You can define a history format only once within a section.
- Because history formats apply to an entire section, you must specify them only between SECTION and ESECTION statements, and not within any other pair of statements. You specify only FORMAT statements between the start and end of the definition.
- HFORMAT data is used only when processing a PUT or CALL statement that includes a HISTORY keyword.
- Once a data format is defined, it is used for all subsequent sections until it is redefined. When redefined, the format is replaced by a new one when required.
- An HFORMAT statement followed immediately by an EFORMAT statement eliminates a previously defined format.
- The data format definition statements let you specify whether to print the control information collected for each history item before the history data. Refer to the Tivoli Information Management for z/OS Panel Modification Facility Guide for details about what data makes up this control information and how to change what is collected.

## SECTION - ESECTION

An RFT has one or more sections, depending on your preference when designing the report. If the report is very simple, you almost certainly need only one section; however, you can use multiple sections to:

- Consolidate several reports into one
- Change titles and headings between sections
Use less storage when processing a large report: the Report Format Facility reads one section at a time into virtual storage.

A SECTION definition statement defines the beginning of each report section. Therefore, the first non-comment, non-GRAPH statement in each RFT must be a SECTION statement.

An ESECTION definition statement ends a report section. Like other RFT definition statements, the SECTION and ESECTION statements must be paired. All definition and command statements between SECTION and ESECTION are part of that report section. You must include all statements, except null statements, between SECTION and ESECTION.

### Syntax of SECTION - ESECTION

```
SECTION [ Separation(nnn) ] [ Name(name) ] [ Print {Y|N} ] [ Test {Y|N|S} ]
.
.
ESECTION
```

**Separation**

defines the number of blanks that separate fields on the report output line. The Report Format Facility uses this separation amount for the CALL and PUT commands when you specify neither the COLUMN nor the SEPARATION keyword on those commands.

*nnn* is a decimal number between 0 and the current line length minus 1. The current line length is defined in the usage notes for the PUT command of the Report Format Facility. The default separation is one blank.

**Name**

indicates that an exit routine is given control during output phase processing. You can use an exit routine to modify data before writing it to the report data set, and to add or delete data lines in the report data set.

*name* is a 1- to 8-character alphanumeric identifier for a user-written load module. If you do not specify the Name keyword, no user exit is given control. For further details, see "Report Exit Routines" on page 231.

**Print**

indicates whether the RFT statements are to be printed as part of the report.

*Y* requests that all statements following the SECTION statement, down to and including the corresponding ESECTION statement, be written to the output data set exactly as they appear in the RFT.

The statements are written as 80-character records and, if the line length specified in your profile is less than 80 characters, the statements are truncated. Any input-phase error messages produced for a statement are printed on the lines immediately following the statement. If required, a page eject occurs before and after writing the RFT statements to separate them from any output-phase data that is written to the output data set. The Report Format Facility produces no titles or headings for the pages containing RFT statements.

*N* indicates that no statements are to be printed with the report. N is the default.
Test

is a debugging aid that tests a report section for errors.

Y requests that all the Report Format Facility error messages issued during the input phase be written to the report data set as they occur. If the input phase is unsuccessful (for example, the Report Format Facility issues an error message indicating that processing has stopped), the output phase is bypassed.

If the input phase is successful, the output phase is processed, but no output is produced. Only the Report Format Facility error messages (that is, no data) are written to the output data set. If a Report Format Facility error occurs during output-phase processing, an appropriate error message is written to the output data set and processing continues. If a Report Format Facility error occurs, such as no virtual storage available or a database access failure, processing stops.

All input-phase error messages written to the output data set, after the first one, might not be valid because they result from the nonacceptance of an earlier statement. For example, you receive an error message on a TITLE statement and later a message that states an ETITLE is not valid because it is not paired with a TITLE statement. ETITLE is in error only because TITLE was not accepted earlier.

N indicates that the report is being run. No test is performed. N is the default.

S indicates that no processing is performed at all; the entire section of the report is skipped.

Figure 21 illustrates code that prints all statements in the section with any error messages they produce. If no errors are found, the input and output phases are processed; however, no output is produced.

SEC PRINT(Y) TEST(Y)

. .

ESEC

Figure 21. Code to Print Statements with Error Messages

Figure 22 illustrates code that gives control to the PROBEXIT data-output exit routine before writing data to the report data set. All statements for this section are shown in the example on page 307.

SECTION NAME(PROBEXIT) /* PROBLEM EXCEPTION REPORT */

. .

ESECTION

Figure 22. Code to Call an Exit Routine

**TFORMAT - EFORMAT**

The TFORMAT (TF) definition statement begins the definition of the freeform text data format used in the CALL and PUT command statements. When TFORMAT is active, the substring keywords FOR, FROM, WFOR, and WFROM are ignored on the PUT statement with the TEXT keyword. The EFORMAT (EF) definition statement ends the freeform text data format definition.
For history data, use the HFORMAT and EFORMAT statements; see "HFORMAT - EFORMAT" on page 161.

An EFORMAT statement must follow a TFORMAT statement. Only FORMAT statements are valid between FORMAT and EFORMAT statements. See "FORMAT" on page 157 for more information on the FORMAT definition statement.

### Syntax of TFORMAT - EFORMAT

```
TFORMAT /* Text data format */

FORMAT VAL(TXT:) LEN(5) /* Insert 'TXT: ' */
FORMAT TEXT(DATM/.) LEN(11)
FORMAT TEXT(USER/.) LEN(9)

EFORMAT .
```

### Usage Notes

- Within a section, you can define a text format only once.
- Because a freeform text format is applicable to an entire section, you must specify it between SECTION and ESECTION statements, but not within any other pair of statements.
- TFORMAT data is used only when processing a PUT or CALL statement that includes a TEXT keyword.
- Once you define a data format, it is used for all subsequent sections until you redefine it. When redefined, the new format replaces the old one the next time it is required.
- A TFORMAT statement followed immediately by an EFORMAT statement eliminates a previously defined format.
- The data format definitions let you specify whether to print the control information collected for each text item. Refer to the [Tivoli Information Management for z/OS Panel Modification Facility Guide](#) for details about the data that makes up this control information and how to change the data that is collected.

Figure 23 illustrates the use of the TFORMAT statement. This example includes FORMAT statements that cause all lines of address text to be printed. The resulting printout is similar to that shown in Figure 24 on page 165.

```
TFORMAT /* Text data format */

FORMAT VAL(TXT:) LEN(5) /* Insert 'TXT: ' */
FORMAT TEXT(DATM/.) LEN(11)
FORMAT TEXT(USER/.) LEN(9)

EFORMAT .

SEARCH ...

IF TEXT(!S0E07) OP(=) VAL(&ZIFDATA)
PUT ...
PUT COL(30) TEXT(!S0E07) OP(H) /* Formatted text */

EIF
ESEARCH
```

*Figure 23. Example of TFORMAT - EFORMAT*
The TITLE and ETITLE definition statements specify the title boundaries. The Report Format Facility places the titles you define at the top of each page of the report.

**Syntax of TITLE - ETITLE**

```
TITLE
.
.
ETITLE
```

**Usage Notes**

- You can define titles for an entire report or for a section of a report. However, you cannot change the title or add a second title within a section, but you can eliminate a defined title. You can also include more than one title within a TITLE - ETITLE definition and control the titles generated by using conditional processing commands (IF, ELSE, EIF).
- You can redefine titles in a subsequent section.
- Because the titles are applicable to an entire section, you must specify them between SECTION and ESECTION statements. You cannot specify titles within any other pair of statements.
- A TITLE statement followed immediately by an ETITLE statement eliminates a previously defined title.
- The command statements that are valid between the TITLE and ETITLE statements are CALL, DO, EDO, EIF, ELSE, IF, PUT, SET, SETD, SPACE, and TAB. The Report Format Facility processes these statements each time titles are required.

**Note:** If you use a CALL or PUT statement within the title definitions to obtain data from a database record, the resulting data might not be what you expect: the record from which data is taken is the current record meeting the search-level criteria active at the time the titles are processed. To ensure that the correct data is displayed, set variables outside the titles and use them with the PUT statements, or provide conditional processing within the titles based on the current setting of the &ZCURLVL variable.

- The title can consist of multiple command statements. Storage is the only limit on the number of statements that can appear between a TITLE and an ETITLE statement.

**Figure 25 on page 166** illustrates a TITLE statement that uses conditional processing to set variables. **Figure 26 on page 166** illustrates a TITLE statement that centers the title, **PROBLEM EXCEPTION REPORT**, at the top of each page of the report. See "Defining Titles and Headings" on page 289 for an example of a multiline title.
SECTION
   TITLE
   PUT DATA(RNID/.) /* Yields current child or parent ID */
   PUT VALUE(&PARENT.) /* Yields current parent ID */
   IF VALUE(&ZCURLVL.) OP(=) VALUE(1) /* ZCURLVL is 1 */
      PUT VALUE('THIS TITLE IS A RESULT OF PARENT SEARCH')
      PUT VALUE('PROCESSING SECTION')
   ELSE /* ZCURLVL is 2 */
      PUT VALUE('THIS TITLE IS A RESULT OF CHILD SEARCH')
      PUT VALUE('PROCESSING SECTION')
   EIF
   ETITLE
   SEARCH ARG(!S0B06 ¬!S0B07) /* Parent change records only */
   SET NAME(PARENT) DATA(RNID/.)
   PUT DATA(RNID/.) /* Yields parent ID */
   SPACE LINES (-1) /* Write output buffer */
   SEARCH ARG(RNOR/&PARENT.) MERGE(N) /* Search for children of parent */
      PUT DATA(RNID/.)
      SPACE LINES (-1) /* Write output buffer */
   ESEARCH
   ESEARCH
   ESECTION

Figure 25. TITLE Statement Using Variables

TITLE
   PUT VALUE('PROBLEM EXCEPTION REPORT') COLUMN(C)
   ETITLE

Figure 26. TITLE Statement for a Centered Single-Line Title
The Report Format Facility has two types of statements:

- Definition statements, which define segments of data that remain constant either for the whole RFT or for a section.
- Command statements, which control record processing such as retrieving, formatting, and printing.

This chapter describes the command statements in detail. The statements are listed in alphabetical order.

**CALL**

The CALL (C) command statement gives control to a user-written or Report Format Facility-defined exit routine and writes the data returned to the report data set. See "CALL Command Exit Routines" on page 235 for information on CALL command exit routines.

### Syntax of CALL

<table>
<thead>
<tr>
<th>Call</th>
<th>Name(name)</th>
<th>Input(data)</th>
<th>Minlines(nnn)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>[Column ((nnn))]</td>
<td>[Separation(nnn)]</td>
<td>[Operator ((H))]</td>
</tr>
<tr>
<td></td>
<td>[((C))]</td>
<td></td>
<td>[((V))]</td>
</tr>
<tr>
<td></td>
<td>[((S))]</td>
<td></td>
<td>[((N))]</td>
</tr>
<tr>
<td></td>
<td>Length(nnn)</td>
<td>Pad(x)</td>
<td>Justify ((L))</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>[(R)]</td>
</tr>
</tbody>
</table>

- **Name**
  
  indicates the exit routine to be given control.

- **name**
  
  is a 1- to 8-character alphanumeric identifier for a user-written load module or a load module supplied by Report Format Facility.

- **Input**
  
  passes up to 255 characters of data to the exit routine.

- **data**
  
  is the input data that is expected by the exit routine. It can consist of prefixes, variables, dictionary index keys, a character string, or any combination of these. Substitution of variable or dictionary index key data occurs before the data is passed to the exit routine. After variable substitution, if the value in the variable is a dictionary index key, it is passed as the index and not as the actual s-word or p-word.
Minlines indicates the minimum number of lines that must be available on the current page to write the data on the page.

\[ \text{nnn} \]

is a decimal number between 1 and the current page length. Page length is specified in the user’s profile or when prompted. The default is one line. If the specified number of lines is not available on the page, the Report Format Facility starts a new page.

Column indicates the beginning column for the data in the report.

\[ \text{nnn} \]

is a decimal number between 1 and the current line length that indicates the beginning column position. The current line length is discussed in the usage notes on page 169.

C centers the data on the entire output line. If there is already data on the output line, that data is written first and the following data is then centered on the next line.

When you specify COLUMN(C), follow this command with a SPACE command or a PUT or CALL command. These commands must specify a COLUMN keyword value that is less than the resulting current column position. This ensures that only the centered data is on the line and that the next PUT or CALL command does not write data to the same output line. See Figure 27 on page 172 and Figure 29 on page 172 for examples.

You cannot use COLUMN(C) with the OPERATOR(H) keyword.

S specifies that the Report Format Facility is to use the default separation amount specified on the SECTION statement to position the data on the output line. The rules for default separation are specified in the usage notes on page 170.

Separation defines the number of blanks that separate fields on the output line. You cannot use this keyword if you use COLUMN(C) or COLUMN(S).

\[ \text{nnn} \]

is a decimal number between 0 and the current line length minus 1. The current line length is defined in the usage notes on page 169. The rules specified in the usage notes on page 170 for default separation apply in this case, except that the specified separation amount is used instead of the default amount from the SECTION statement.

Operator indicates the action performed when the exit routine returns multiple data items.

\[ \text{H} \]

indicates horizontal iteration; this value causes the Report Format Facility to write multiple data items across the output line, beginning in the specified or calculated column position. For an example, see Figure 31 on page 172. If the TAB keyword is used, it specifies the start column position for the multiple data items. If the LENGTH keyword is used, it indicates the total number of characters on the output line, not the length of each individual data item. For more details on horizontal iteration, see the usage notes. You cannot use OPERATOR(H) with the JUSTIFY, COLUMN(C), or PAD keywords.

\[ \text{V} \]

indicates vertical iteration; this value causes the Report Format Facility to write
multiple data items one per line, beginning in the specified or calculated column position. For an example, see Figure 31 on page 172.

N indicates no iteration; this value causes the Report Format Facility to write only the first data item beginning in the specified or calculated column position. N is the default.

Length
indicates the number of characters to be placed on the output line.

nnn
is a decimal number between 0 and 255. A length of zero (LENGTH(0)) suppresses the writing of the returned data to the output line. The precise meaning of the LENGTH keyword depends on the iteration specified with the OPERATOR keyword. For more information, see the usage notes on page 173. For an example that uses the LENGTH keyword, see Figure 32 on page 173.

Pad
defines the character to be inserted in the output field if the data is shorter than the specified length. If you use this keyword, you must also specify a LENGTH.

x is a single alphanumeric SBCS character or a blank. You must enclose a blank in quotes, for example, PAD(‘ ’). The default pad character is a blank. You cannot use a DBCS character for the PAD value.

Justify
indicates how you want the data aligned in the output field. If you specify the JUSTIFY keyword, you must also specify LENGTH. If required, the field is padded with blanks or with the specified pad character.

L indicates left justification; data is aligned at the left column margin. L is the default.

R indicates right justification; data is aligned at the right column margin.

Usage Notes

- You can use multiple CALL commands to place different data items on the same output line.
- The Report Format Facility obtains the page length and the current line length for the report from the user’s profile when the report is run, as follows:
  - The page length is in the Lines per page field.
  - The current line length is calculated using the Logical record length (LRECL) field based on the record format as follows:
    - VA or VBA
      LRECL minus 5
    - V or VB
      LRECL minus 4
    - FA or FBA
      LRECL minus 1
    - F or FB
      Same as LRECL.

For SYSOUT destinations, the record format defaults to VBA.
Each processing of a CALL command can move multiple data items, one at a time, to the output line (buffer). Each data item can be a single character or a string of up to 255 characters.

The Report Format Facility uses a current column pointer to position data on the output line. It indicates the next available position where you can place data on the output line. The current column pointer begins at column 1 for a new line and advances across the line as you place data on the output line.

The keywords on the CALL command work with the current column pointer to position the data items on the output line. In general, the Report Format Facility places data on the current output line as long as the starting column (as specified or defaulted on the CALL command) is at or beyond the current column pointer. The output line is written to the output data set only when one of the following occurs:

- A SPACE or EJECT command is processed.
- A message is written to the output data set.
- An ESECTION (end of section) statement is processed.
- An attempt is made to place data on the output line in front of the current column pointer. For example, suppose the last PUT command placed 8 characters of data on the output line beginning at column 41. The current column pointer is now at column 49. If the next CALL command attempts to place data in column 48, the Report Format Facility writes the current line to the data set before it can put the new data on the output line, or the existing data will be overwritten.

The placement of data is an important consideration when you design your own RFTs. For an example of common problems you might find and how you can avoid them, see "Example 4: Overcoming Data Output Problems" on page 304.

The Report Format Facility uses either the COLUMN or the SEPARATION keyword, or both, to position the data on the output line. If you do not specify either of these keywords, the Report Format Facility positions the data according to the following rules:

1. In column 1, if no data is currently on the output line.
2. In the column calculated by adding the default separation amount (specified on the SECTION statement) to the current column position, if there is data currently on the output line. The current column position is the column position following the last character written to the output line. If DBCS data is present, all SO and SI characters that would stand next to each other on concatenation are removed, unless they have been assigned to column positions.
3. If the calculation in rule 2 exceeds the current line length, the current output line is written to the output data set. Then the new data is put in column 1.

Assuming that you specified horizontal iteration (OPERATOR(H)), Table 2 summarizes the order of precedence for the COLUMN and SEPARATION keywords and the TAB command for data placement on the output line.

Table 2. Data Placement on the Output Line, Depending on COL, SEP, and TAB Specifications

<table>
<thead>
<tr>
<th>Specified COL</th>
<th>Specified SEP</th>
<th>Specified TAB</th>
<th>Data Position of First Item</th>
<th>Data Position of Other Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y</td>
<td>Y</td>
<td>Y/N</td>
<td>Use specified column</td>
<td>Add current column and specified separation</td>
</tr>
<tr>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Use specified column</td>
<td>Use next tab position</td>
</tr>
</tbody>
</table>
Table 2. Data Placement on the Output Line, Depending on COL, SEP, and TAB Specifications (continued)

<table>
<thead>
<tr>
<th>Specified COL</th>
<th>Specified SEP</th>
<th>Specified TAB</th>
<th>Data Position of First Item</th>
<th>Data Position of Other Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>Use specified column</td>
<td>Add current column and default separation</td>
</tr>
<tr>
<td>N</td>
<td>Y</td>
<td>Y/N</td>
<td>Add current column and specified separation</td>
<td>Add current column and specified separation</td>
</tr>
<tr>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>Add current column and default separation</td>
<td>Use next tab position default separation</td>
</tr>
<tr>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Add current column and default separation</td>
<td>Add current column and default separation</td>
</tr>
</tbody>
</table>

If a data item does not fit on the current output line, the current output line is written to the output data set. The data item is then positioned on the next line beginning in the specified or calculated column if you specified the COLUMN keyword or TAB statement, or in column 1 if you did not specify the COLUMN keyword or TAB statement.

■ The following keyword combinations are not valid on the CALL command:
  • COLUMN and SEPARATION without OPERATOR(H)
  • COLUMN(C) with SEPARATION
  • COLUMN(S) with SEPARATION
  • OPERATOR(H) with
    – JUSTIFY
    – PAD
    – COLUMN(C)

■ The following keywords are not valid without LENGTH:
  • JUSTIFY
  • PAD

■ If you specify horizontal iteration with the OPERATOR keyword (OPERATOR (H)), the Report Format Facility determines length as follows:
  • The LENGTH keyword specifies the number of characters you can place on the output line.
  • The default length is the line length minus the specified or calculated column position of the first data item (which cannot exceed 255 characters).
  • If the specified length exceeds the default length, the default is used.
  • If multiple data items fit on the current line, then the Report Format Facility splits the data at a blank (word boundary).
  • If the length of an individual data item exceeds the specified or defaulted length, it is not truncated. Instead, the Report Format Facility breaks the item at the specified or calculated line length and writes it to the output data set. The Report Format Facility then writes the remainder of the data item to the next lines starting in the specified or calculated column if you specified the COLUMN keyword, or in column 1 if you did not. The Report Format Facility makes no attempt to split the data at a blank.

■ If you do not specify horizontal iteration with the OPERATOR keyword, the length of the output line is determined as follows:
The LENGTH keyword specifies the number of characters you may place on the output line for the individual data items.

The default length is the length of the data item (which cannot exceed 255 characters).

A data item that is longer than the specified length is truncated. A data item that is shorter is padded with blanks or the specified pad character.

Examples

```plaintext
PUT VALUE('CENTER') COLUMN(C)  /* Current column position would be */
   /* 44 for 80 column output data set */
PUT VALUE('OFF-CENTER') COLUMN(44)  /* same line */
```

Figure 27. Example 1 Using COLUMN Keyword

The resulting output is similar to that shown in Figure 28.

```
CENTER
OFF-CENTER
```

Figure 28. Output from Example 1 Using COLUMN Keyword

The example in Figure 29 is an alternative to Figure 27.

```plaintext
PUT VALUE('CENTER') COLUMN(C)
SPACE LINES(-1)  /* Kicks out line to data set */
PUT VALUE('OFF-CENTER') COLUMN(44)  /* next line */
```

Figure 29. Example 2 Using COLUMN Keyword

The resulting output is similar to that shown in Figure 30.

```
CENTER
OFF-CENTER
```

Figure 30. Examples of the COLUMN Keyword

Figure 31 illustrates horizontal and vertical iteration.

```
O(H)  horizontal iteration looks like this
O(V)  vertical iteration looks like this
```

Figure 31. Examples of Output with the OPERATOR Keyword
DHEADS

The DHEADS (D) command statement writes the currently defined report headings to the report data set immediately. The headings are defined by HEADING-EHEADING.

Data already on the output line is written to the report before the headings.

If writing the data or headings causes a page eject, the Report Format Facility ignores any unwritten portion of the headings. The unwritten portions are lost, and the entire heading is then written again on the page eject. This action prevents the headings from appearing twice on the new page; headings are automatically written after a page eject.

Syntax of DHEADS

Dheads
DO, EDO, LEAVE

The DO, EDO, and LEAVE command statements control the iterative processing of command statements. The commands to be iterated are those between corresponding DO and EDO statements. The LEAVE command causes the Report Format Facility processing to resume with the statement immediately following the EDO statement of the innermost DO/EDO pair that contains the LEAVE statement.
Syntax of DO, EDO, LEAVE

DO { TYPE {(While)} data-type operator data-type [Length(nnn)] [join] [compare]} { {(Until)} [character-base] [character-base] [compare] } { [word-base] [word-base] } { [word-span] [word-span] } (as detailed below)

TYPE {(Forever) }

data-type operator

{ Data {(prefix) } } { Operator { (= | EQ) } }
{ {variable} } { { (~= | NE) } }
{ {dictionary index key} } { { (< | LT) } }

{ History {(prefix) } } { { (<= | LE) } }
{ {variable} } { { (>= | GE) } }
{ {dictionary index key} } { { (~< | NL) } }
{ } { { (~> | NG) } }

Text { {variable} } { { (s-word index) } } join

Value { {string} } [ Join { ({}) } ]
{ {variable} } [ { (|) } ]

compare

[ Compare { (Character) } ]
[ ]
[ (Decimal) ]
[ (Integer) ]

character-base character-span

[ FROM { {variable} } ] [ FOR { {variable} } ]
{ {nnn} } { {nnn} }
{ () } { () }

word-base word-span

[ WFROM { {variable} } ] [ WFOR { {variable} } ]
{ {nnn} } { {nnn} }
{ () } { () }

.
.
.
[Leave]
.
.
.
EDo

**Type**

identifies the kind of iterative processing to be performed.

**While**

indicates that the test for the specified condition is to be performed before the iterative section is processed. If the condition is false, processing resumes with the statement immediately following the corresponding EDO statement.

**Until**

indicates that the test for the specified condition is to be performed after each
execution of the iterative processing section. If the condition is true, processing
resumes with the statement immediately following the corresponding EDO statement.

**Forever**
indicates unconditional iteration.

**nnnnnnnnnn**
is a nonnegative integer from 0 to 2,147,483,647 indicating the number of iterations
to be performed before processing resumes with the statement immediately following
the corresponding EDO statement.

**Data**
specifies structured data to be compared.

**prefix**
is a 1- to 5-character alphanumeric code followed by a slash (/) or an underscore (_).
The first character must be alphabetic. Use a period after the slash or underscore to
retrieve prefix data from the record. For example, STAC/ . finds any data associated
with the STAC/ prefix. Use any other character to retrieve data that exactly matches
the specified string. For example, STAC/OPEN finds the string whether it is the
STAC/ prefix with associated data of OPEN or the STAC/OPEN string exists in the
description abstract.

**variable**
is a 1- to 8-character alphanumeric name, preceded by an ampersand (&), which
identifies a p-word or an s-word in the dictionary data set. The p-word or s-word
identifies the data item.

**dictionary index key**
is either !P or !S followed by 4 hexadecimal characters, which identifies a p-word or
an s-word in the dictionary data set. The p-word or s-word identifies the data item.

See "Keywords and Keyword Values" on page 148 for more information about prefixes,
variables, and dictionary index keys.

**History**
identifies the history data to be compared.

**prefix**
is a 1- to 5-character alphanumeric code followed by a slash (/) or an underscore (_).
The first character must be alphabetic. A period must follow the slash or underscore
to retrieve data from the record. The prefix identifies the history data item.

**variable**
is a 1- to 8-character alphanumeric name, preceded by an ampersand (&), which
identifies a p-word or an s-word associated with the data item.

**dictionary index key**
is either !P or !S followed by 4 hexadecimal characters, which identifies a p-word or
an s-word in the dictionary data set. The p-word or s-word identifies the data item.

**Text**
identifies freeform text data to be compared.

**variable**
is a 1- to 8-character alphanumeric name, preceded by an ampersand (&), which
identifies an s-word associated with the freeform text item.
s-word index
is !S followed by 4 hexadecimal characters, which represent an s-word in the
dictionary data set. See "Field Characteristics” on page 319 for a list of the s-word
index keys for freeform text.

You can also specify a prefix, a prefix index key, or a variable that contains a prefix
for the TEXT keyword without causing an error message. However, no data is found
because prefixes are not used to define freeform text.

Value
identifies the user-specified data to be compared.

string
is any user-specified data. If the following characters are considered data, you must
enclose them in single quotes:

- Left parenthesis, '('
- Right parenthesis, ')
- Exclamation point, '!
- Ampersand, '&'
- Leading blanks, ' '.

To have a single quote appear as data, you must code two of them. The Report
Format Facility removes the extra quote during the input phase of processing. The
statements in the following example illustrate how to code a single quote.

```
DO TYPE (WHILE)
  PUT VALUE('SINGLE QUOTE: '''') /* SINGLE QUOTE: ' */
  PUT VALUE('''') /* '' */
```

The length of a string is limited by the maximum continuation for the statement.
(Each statement can consist of ten 71-character lines.) However, the value of the
LENGTH keyword also limits the number of characters on which to operate. For
more information on the LENGTH keyword, see page 180.

variable
is a 1- to 8-character alphanumeric name, preceded by an ampersand (&), which
identifies a string. If you specify the equal (= or EQ) or not-equal (~= or NE)
operator, you can use one of the following variables:

&ZIFFIND
is used with the VALUE data type keyword to test for the presence of an item
(within the record) that satisfies the specifications supplied by the other data
type keyword. The test is true if an item is found; it is false if no item is found.

&ZIFDATA
determines whether a record contains data for the specified prefix or s-word, or
whether the specified variable contains data. The Report Format Facility
considers blanks as data. If you specify &ZIFDATA with an equal operator (= or
EQ) and a DATA, HISTORY, or TEXT keyword as the other data type, the
statement is true if data is found in the current record for the specified p-word or
s-word. With a not equal operator (~= or NE), the statement is true if no data
exists in the current record.

When you specify &ZIFDATA with an equal operator and a VALUE keyword as
the other data type, the condition is true if the keyword value is a single variable
and the variable is not null. If the keyword value consists of multiple variables
or a combination of a string and a variable, the condition is always true.
When you specify &ZIFDATA with a not equal operator and a VALUE keyword as the other data type, the condition is true if the keyword value is a single variable and the variable is null. If the keyword value consists of multiple variables or a combination of a string and a variable, the condition is always false.

You cannot compare &ZIFFIND and &ZIFDATA to each other.

&ZIFIPL1 through &ZIFIPL9
are level-initial processing variables that indicate when the first record for a search level is being processed. The variables are always set to a null value except when the first record of the corresponding search level is being processed, or during the unconditional processing of that search level. That is, the variables are set to an X'01' each time the Report Format Facility processes the corresponding SEARCH command during output processing and reset to a null value the first time the Report Format Facility processes the corresponding ESEARCH command.

Specify these variables with an IF statement to provide initialization or “first time” processing within their corresponding levels. You must specify them with the VALUE keyword; you cannot use them with other variables or as part of a string. To test them, use the &ZIFDATA variable.

Compare
determines the procedure to use for comparing the data specified by the two data type keywords (VALUE, TEXT, DATA, or HISTORY). Comparisons are either **logical** or **algebraic**.

**Logical comparison**
is a bit by bit comparison without regard to algebraic signs. Algebraic signs are considered only as characters for CHARACTER. For example, ‘−1’C (‘60F1’X) is logically greater than ‘1’C (‘40F1’X).

If comparing mixed strings, you must ensure that the comparison is meaningful. DBCS numeric characters are treated as character strings and only logical comparisons can be performed on them.

If one of the operands contains more than 15 significant digits or 10 fractional digits, the Report Format Facility performs a logical comparison, regardless of the COMPARE keyword value.

If the Report Format Facility finds no data for the DATA, HISTORY, or TEXT keyword, or if a VALUE keyword specifies a variable with a null value, the Report Format Facility considers the null value to be blanks for logical comparisons.

**Algebraic comparison**
is a comparison with algebraic signs considered. Algebraic signs are considered for both DECIMAL and INTEGER as indicated in the information below.

If one of the operands contains more than 15 significant digits or 10 fractional digits, the Report Format Facility performs a logical comparison, regardless of the COMPARE keyword value.

If the Report Format Facility finds no data for the DATA, HISTORY, or TEXT keyword, or if a VALUE keyword specifies a variable with a null value, the Report Format Facility considers the null value to be 0 for algebraic comparisons.
Character
indicates that the operands are compared logically. Algebraic signs are considered only as characters.

Decimal
indicates that if both operands are decimal (including integers) the comparison is algebraic. If one operand is character or contains DBCS numeric characters, the comparison is logical.

For a decimal comparison, the EBCDIC characters represent a positive or a negative number (base 10) with or without a decimal point (‘−001.2’C or ‘40404060F0F0F14BF240’X). The following steps are performed:

1. Leading and trailing blanks are removed.
2. The operand having the lesser precision is padded with zeroes to match the precision of the other operand.
3. Any decimal points are removed from the operands.
4. The operands are converted to packed decimal format.
5. The operands are compared algebraically.

For example, ‘−001.2’C (‘40404060F0F0F14BF240’X) and ‘1’C (‘F1’X) are converted to ‘012D’X and ‘010F’X. The second operand is then determined to be greater because it is positive.

Integer
indicates that if both operands are integer, the comparison is algebraic. If one operand is character or contains DBCS numeric characters, the comparison is logical.

For an integer comparison, the EBCDIC characters represent a positive or a negative number (base 10) without a decimal point (‘−0012’C or ‘40404060F0F0F1F240’X).

The following steps are performed:
1. Leading and trailing blanks are removed.
2. The operands are converted to packed decimal format.
3. The operands are compared algebraically.

For example, ‘−0012’C (‘40404060F0F0F1F240’X) and ‘1’C (‘F1’X) are converted to ‘012D’X and ‘001F’X. The second operand is then determined to be greater because it is positive.

Integer is the default.

Operator
identifies the logical operator to use for the comparison. The valid SBCS operators are defined in the following list. You can use either expression.

= or EQ
The values are equal.

= or NE
The values are not equal.

< or LT
The first value is less than the second value.

> or GT
The first value is greater than the second value.
The first value is less than or equal to the second value.

The first value is greater than or equal to the second value.

The first value is not less than the second value.

The first value is not greater than the second value.

Length

indicates the number of bytes to be compared.

nnn

is an integer between 1 and 255. A data item that is longer than the specified length is truncated on the right.

When you want to do a logical comparison, if either data item is less than the specified length, it is padded with blanks on the right to this length. However, if you did not specify the LENGTH keyword, the default is the length of the longer data item, with the shorter item right-padded with blanks to this length.

If you want to do an algebraic comparison, the Report Format Facility removes the leading and trailing blanks, along with insignificant digits.

Join

enables you to join two or more IF statements to form one logical expression. The number of statements thus joined is limited only by storage. The conditions given by the joined statements are processed sequentially. As the sequence of conditions is processed, a cumulative setting (true or false) is maintained and ANDed or ORed with the result (true or false) of the test for the next condition according to the operation specified by the JOIN keyword of the current statement. You must add this keyword to all joined statements except the last. Joined statements must be adjacent.

& indicates that the result of the test for the condition specified for the next IF statement is to be logically ANDed with the result of the test for the condition specified for the DO statement.

| indicates that the result of the test for the condition specified for the next IF statement is to be logically ORed with the result of the test for the condition specified for the DO statement.

FROM

indicates the position of the substring starting character. Data referenced by the associated data type keyword is used as source for this operation. If an insufficient number of characters is present to satisfy the given keyword value, no data is returned; therefore, no data is used for this part of the comparison.

variable

is a 1- to 8-character alphanumeric name, preceded by an ampersand (&), which identifies an integer.

nnn

is a nonzero integer between -255 and 255. If the value is positive, characters are counted from the beginning of the data to the end. If the value is negative, characters are counted from the end of the data to the beginning. If you do not specify a value for the FROM keyword, a default value of 1 is assumed.
You can specify a FROM keyword for each data type keyword present in a given DO statement. The order of appearance determines the pairing of FROM and data type keywords.

**FOR** indicates the substring length in characters. Data referenced by the associated data type keyword is used as source for this operation. If the FOR keyword value exceeds the number of characters remaining in the data, only the remaining data is used.

variable is a 1- to 8-character alphanumeric name, preceded by an ampersand (&), which identifies an integer.

**nnn** is an integer between 1 and 255. If you do not specify a value for the FOR keyword, the remaining data is used.

You can specify a FOR keyword for each data type keyword present in a given DO statement. The order of appearance determines the pairing of FOR and data type keywords.

**WFRom** indicates the substring starting word. (A word is a delimited string of nonblanks.) The delimiters identifying word boundaries are SBCS blanks, DBCS blanks, SO, or SI. Data referenced by the associated data type keyword is used as source for this operation. If an insufficient number of words is present to satisfy the given keyword value, no data is returned; therefore, no data is used for this part of the comparison.

variable is a 1- to 8-character alphanumeric name, preceded by an ampersand (&), which identifies an integer.

**nnn** is a nonzero integer between -255 and 255. If the value is positive, words are counted from the beginning of the data to the end. If the value is negative, words are counted from the end of the data to the beginning. If you do not specify a value for the WFRom keyword, a default value of 1 is assumed.

You can specify a WFRom keyword for each data type keyword present in a given DO statement. The order of appearance determines the pairing of WFRom and data type keywords.

**WFor** indicates the substring length in words. Data referenced by the associated data type keyword is used as source for this operation. If the WFor keyword value exceeds the number of words remaining in the data, only the remaining data is used.

variable is a 1- to 8-character alphanumeric name, preceded by an ampersand (&), which identifies an integer.

**nnn** is an integer between 1 and 255. If you do not specify a value for the WFor keyword, the remaining data is used.
You can specify a WFOr keyword for each data type keyword present in a given DO statement. The order of appearance determines the pairing of WFOr and data type keywords.

Usage Notes

- An iterative processing section consists of those commands between a corresponding DO/EDO pair, but not within any other processing sections that may exist within the given DO/EDO pair.
- An EDO statement must follow a DO statement within the same processing section.
- Within each search level (levels 0 through 9), you can nest iterative processing sections up to a maximum depth of 25.
- No definition statements are allowed within an iterative processing section. A DO command statement cannot control the processing of definition statements because the Report Format Facility processes definition statements during the input phase and command statements during the output phase.
- When using FRom and FOr keywords, truncation of mixed data may occur at the left or right substring boundaries, or at both. The truncation of data observes the integrity of mixed data:
  - No DBCS character is split into its component bytes.
  - An SO character must immediately precede any DBCS portions.
  - An SI character must immediately follow any DBCS portions.
  - Generated null portions are blanked.
- When using WFRom and WFOr keywords, multiple word delimiters encountered side by side are treated as one delimiter.
- Character substringing is performed after word substringing is complete. In other words, the data referenced by the associated data type keyword must first undergo specified word substringing before it is used as source data for character substringing.

Examples

In Figure 36, the DO statement repeats those statements enclosed by the DO and EDO command statements while variables A and B have the value 1.

```
DO TYPE(WHILE) VALUE(&A) OPERATOR(=) VALUE(1) JOIN(&)
  IF VALUE(&B) OPERATOR(=) VALUE(1)
    
    
  END
EDO
```

Figure 36. DO Statement with WHILE

In Figure 37 on page 183, the DO statement iterates those statements enclosed by the DO and EDO once. This is because '1..C' ('F14B4B'X) is compared logically to '-1.0'C ('60F14BF0'X), and '1.0'C ('010F'X in packed decimal format) is compared algebraically to '-1.0'C ('010D'X in packed decimal format).
In Figure 38, the DO statement repeats those statements enclosed by the DO and EDO command statements while variable X has the value <D1D2>.

\[
\begin{align*}
&\text{DO TYPE(UNTIL) VALUE(1..) OPERATOR(>) VALUE(-1.0) COMPARE(DECIMAL) JOIN(\&)} \\
&\text{IF VALUE(1.0) OPERATOR(>) VALUE(-1.0) COMPARE(DECIMAL)} \\
&\quad \ldots \\
&\text{EDO}
\end{align*}
\]

**Figure 37. DO Statement with UNTIL and Logical Operator**

In Figure 38, the DO statement repeats those statements enclosed by the DO and EDO command statements while variable X has the value <D1D2>.

\[
\begin{align*}
&\text{DO TYPE(WHILE) VALUE(<D1D2>) OPERATOR(=) VALUE(\&X)} \\
&\quad \ldots \\
&\text{EDO}
\end{align*}
\]

**Figure 38. DO Statement Controlled by a Variable**

In Figure 39, the DO statement illustrates how to calculate the total number of hardware and software problems using the WHILE command.

```
SECTION
  PUT VALUE(DEMO FOR WHILE:) COL(1)
  SET NAME(STRING) VALUE(HW SW) LENGTH(20)
  SET NAME(INDEX) VAL(0)
  DO TYPE(WHILE) VAL(INDEX) OP(<) VAL(2)
    SET NAME(INDEX) VAL(INDEX) OP(+) VAL(1)
    SET NAME(RESULT) VAL(STRING) WFROM(INDEX) WFOR(1)
    PUT VALUE(RESULT.) COL(1)
  CALL NAME(BLGOXCNT) INPUT(!S0B01 TYPE/&RESULT.)
EDO
SPACE LI(2)
ESECTION
```

**Figure 39. Calculating with the DO Statement Using WHILE**

In Figure 40, the DO statement illustrates how to calculate the total number of hardware and software problems using the UNTIL command.

```
SECTION
  PUT VALUE(DEMO FOR UNTIL:) COL(1)
  SET NAME(STRING) VALUE(HW SW) LENGTH(20)
  SET NAME(INDEX) VAL(0)
  DO TYPE(UNTIL) VAL(INDEX) OP(=) VAL(2)
    SET NAME(INDEX) VAL(INDEX) OP(+) VAL(1)
    SET NAME(RESULT) VAL(STRING) WFROM(INDEX) WFOR(1)
    PUT VALUE(RESULT.) COL(1)
  CALL NAME(BLGOXCNT) INPUT(!S0B01 TYPE/&RESULT.)
EDO
SPACE LI(2)
ESECTION
```

**Figure 40. Calculating with the DO Statement Using UNTIL**

In Figure 41 on page 184, the DO statement illustrates how to calculate the total number of hardware and software problems using the FOREVER command.
EJECT

The EJECT (EJ) command statement causes a page eject before the next output line is written.

Any defined titles or headings are automatically processed for the new page. If data is on the output line, it is first written to the report data set and then the page eject occurs before writing the next line.

**Syntax of EJECT**

```
Eject
```

Note that the EJECT command does not produce an immediate page eject. Instead it sets up a request for a page eject on the next write. Blank pages are suppressed. Figure 42 is an example of one line of output per page.

**SECTION**

```
SECTION
  PUT VALUE( DEMO FOR FOREVER:) COL(1)
  SET NAME( STRING) VALUE( HW SW) LENGTH(20)
  SET NAME( INDEX) VAL(0)
  DO TYPE( FOREVER)
    IF VAL( &INDEX ) OP( = ) VAL(2)
      LEAVE
    ELSE
      SET NAME( INDEX ) VAL( &INDEX ) OP( + ) VAL(1)
      SET NAME( RESULT ) VAL( &STRING ) WFROM( &INDEX ) WFOR(1)
      PUT VALUE( &RESULT. ) COL(1)
      CALL NAME( BLGOXCNT ) INPUT( !S0B01 TYPE/&RESULT. )
    EIF
  ED0
  SPACE LI(2)
ESECTION
```

*Figure 41. Calculating with the DO Statement Using FOREVER*

**END**

The END (EN) command statement causes processing of the search level in which it is specified to end. END must be specified between a SEARCH and ESEARCH statement pair.

Although you can use the END command to stop processing at any point in a search level, you will usually want to enclose it within a conditional statement, such as an IF/EIF pair. END stops loop processing of the search level immediately and simulates an end-of-data condition for the level in which it is specified. If you specify an end-of-data definition for
the search level, processing depends on the value of the SEARCH level based on its EXECUTE keyword value. The END command is similar to the LEAVE command for DO/EDO.

Syntax of END

| ENd |

**IF - ELSE - EIF**

The IF, ELSE, and EIF command statements control the conditional processing of command statements.

- The IF (I) command statement defines the beginning of a true or false conditional processing section. It includes two data items to be compared, the type of comparison to be made, and the length of the comparison.

- The ELSE (EL) command statement defines the false conditional processing section.

- The EIF (EI) command statement defines the end of a true or false conditional processing section.

An IF must be followed by a corresponding EIF within the same search level.

If the specified condition is true, the Report Format Facility processes all commands between the IF and the corresponding ELSE (or EIF, if you do not specify ELSE) statements.

If the condition is false, the Report Format Facility skips these commands and processes all commands between the ELSE and the corresponding EIF.

If the condition is false and you do not specify ELSE, the Report Format Facility continues processing with the statement following the EIF command.
# Syntax of IF - ELSE - EIF

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>IF</strong></td>
<td>data-type operator data-type [Length(nnn)] [join]</td>
</tr>
<tr>
<td>[character-base]</td>
<td>[character-base] [compare]</td>
</tr>
<tr>
<td>[character-span]</td>
<td>[character-span]</td>
</tr>
<tr>
<td>[word-base]</td>
<td>[word-base]</td>
</tr>
<tr>
<td>[word-span]</td>
<td>[word-span]</td>
</tr>
</tbody>
</table>

(data-type operator (as detailed below))

<table>
<thead>
<tr>
<th>data-type</th>
<th>operator</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(compare)

<table>
<thead>
<tr>
<th>compare</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(character-base character-span)

<table>
<thead>
<tr>
<th>character-base</th>
<th>character-span</th>
</tr>
</thead>
<tbody>
<tr>
<td>FRom (variable) ]</td>
<td>FOr (variable) ]</td>
</tr>
<tr>
<td>{nnn} ]</td>
<td>{nnn} ]</td>
</tr>
<tr>
<td>() ]</td>
<td>() ]</td>
</tr>
</tbody>
</table>

(join)

<table>
<thead>
<tr>
<th>join</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(word-base word-span)

<table>
<thead>
<tr>
<th>word-base</th>
<th>word-span</th>
</tr>
</thead>
<tbody>
<tr>
<td>WFRom (variable) ]</td>
<td>WFor (variable) ]</td>
</tr>
<tr>
<td>{nnn} ]</td>
<td>{nnn} ]</td>
</tr>
<tr>
<td>() ]</td>
<td>() ]</td>
</tr>
</tbody>
</table>

## Data

specifies structured data for comparison.

**prefix**

is a 1- to 5-character alphanumeric code followed by a slash (/) or an underscore (_). The first character must be alphabetic. Use a period after the slash or underscore to retrieve prefix data from the record. For example, STAC/ finds any data associated with the STAC/ prefix. Use any other character to retrieve data that exactly matches
the specified string. For example, STAC/OPEN finds the string whether it is the STAC/ prefix with associated data of OPEN or the STAC/OPEN string exists in the description abstract.

**variable**
is a 1- to 8-character alphanumeric name, preceded by an ampersand (&), which identifies a p-word or an s-word. The data associated with the p-word or the s-word is used for the comparison. See page 205 for information about data type keywords and how they are used.

**dictionary index key**
is either !P or !S followed by 4 hexadecimal characters, which identifies a p-word or s-word in the dictionary data set. The p-word or s-word identifies the data item.

See “Keywords and Keyword Values” on page 148 for more information about prefixes, variables, and dictionary index keys.

**History**
indicates history data to be compared.

**prefix**
is a 1- to 5-character alphanumeric code followed by a slash (/) or an underscore (_). The first character must be alphabetic. A period must follow the slash or underscore to retrieve data from the record. The prefix identifies the history item.

**variable**
is a 1- to 8-character alphanumeric name, preceded by an ampersand (&), which identifies a p-word or s-word.

**dictionary index key**
is either !P or !S followed by 4 hexadecimal characters, which represent a p-word or s-word in the dictionary data set.

See “Keywords and Keyword Values” on page 148 for more information about prefixes and dictionary index keys.

**Text**
indicates freeform text data to be compared.

**variable**
is a 1- to 8-character alphanumeric name, preceded by an ampersand (&), which identifies an s-word.

**s-word index**
is !S followed by 4 hexadecimal characters, which represents an s-word in the dictionary data set. See “Field Characteristics” on page 319 for a list of the s-word index keys for freeform text.

You can also specify a prefix, a prefix index key, or a variable that contains a prefix for the TEXT keyword without causing an error message. However, no data is found because prefixes are not used to define freeform text.

**Value**
indicates user-specified data to be compared.

**string**
is any user-specified data. If the following characters are considered data, you must enclose them in single quotes:
To have a single quote appear as data, you must code two of them. The Report Format Facility removes the extra quote during the input phase of processing. The statements in the following example illustrate how to code a single quote.

```
PUT VALUE('SINGLE QUOTE: ''') /* SINGLE QUOTE: ' */
PUT VALUE('') /* '' */
```

The length of a string is limited by the maximum continuation for the statement. (Each statement can consist of ten 71-character lines.) However, the value of the LENGTH keyword also limits the number of characters operated on. For more information on the LENGTH keyword, see page 190.

To specify a case-sensitive string for a value, use the CSCHARACTER value of the COMPARE keyword.

A variable is a 1- to 8-character alphanumeric name preceded by an ampersand (&), which identifies a string. If you specify the equal (= or EQ) or not equal (¬= or NE) operator, you can use one of the following variables:

- **&ZIFFIND**
  
  is used with the VALUE data type keyword to test for the presence of an item (within the record) that satisfies the specifications supplied by the other data type keyword. The test is true if an item is found; it is false if no item is found.

- **&ZIFDATA**
  
  determines whether a record contains data for the specified p-word or s-word, or if the specified variable contains data. The Report Format Facility considers blanks as data. If you specify &ZIFDATA with an equal (=) operator and a DATA, HISTORY, or TEXT keyword as the other data type, the statement is true if data is found in the current record for the specified p-word or s-word. With a not equal (¬=) operator, the statement is true if no data exists in the current record.

  When you specify &ZIFDATA with an equal operator and a VALUE keyword as the other data type, the condition is true if the keyword value is a single variable and the variable is not null. If the keyword value consists of multiple variables, or a combination of a string and a variable, the condition is always true.

  When you specify &ZIFDATA with a not equal operator and a VALUE keyword as the other data type, the condition is true if the keyword value is a single variable and the variable is null. If the keyword value consists of multiple variables, or a combination of a string and a variable, the condition is always false.

  You cannot compare &ZIFFIND and &ZIFDATA to each other.

- **&ZIFIPL1 through &ZIFIPL9**
  
  are level-initial processing variables that indicate when the first record for a search level is being processed. The variables are always set to a null value except when the first record of the corresponding search level is being
processed, or during the unconditional processing of that search level. That is, the variables are set to an X'01' each time the Report Format Facility processes the corresponding SEARCH command during output processing and reset to a null value the first time the Report Format Facility processes the corresponding ESEARCH command.

Specify these variables with an IF statement to provide initialization or “first time” processing within their corresponding levels. You must specify them with the VALUE keyword; you cannot use them with other variables or as part of a string. To test them, use the &ZIFDATA variable.

**Compare**

determines the procedure to use for comparing the data specified by the two data type (VALUE, TEXT, DATA, or HISTORY) keywords. Comparisons are either *logical* or *algebraic*.

**Logical comparison**
is a bit by bit comparison without regard to algebraic signs. Algebraic signs are considered only as characters for CHARACTER. For example, ‘−1’C (‘60F1’X) is logically greater than ’1’C (‘40F1’X). If comparing mixed strings, you must ensure that the comparison is meaningful.

DBCS numeric characters are treated as character strings and only logical comparisons can be performed on them.

If one of the operands contains more than 15 significant digits or 10 fractional digits, the Report Format Facility performs a logical comparison, regardless of the COMPARE keyword value.

If the Report Format Facility finds no data for the DATA, HISTORY, or TEXT keyword, or if a VALUE keyword specifies a variable with a null value, the Report Format Facility considers the null value to be blanks for logical comparisons.

**Algebraic comparison**
is a comparison with algebraic signs considered. Algebraic signs are considered for both DECIMAL and INTEGER as indicated in the information below.

If one of the operands contains more than 15 significant digits or 10 fractional digits, the Report Format Facility performs a logical comparison, regardless of the COMPARE keyword value.

If the Report Format Facility finds no data for the DATA, HISTORY, or TEXT keyword, or if a VALUE keyword specifies a variable with a null value, the Report Format Facility considers the null value to be 0 for algebraic comparisons.

**Character**

indicates that the operands are compared logically. Algebraic signs are considered only as characters. The operands are converted to uppercase before the actual comparison is performed.

**CSCharacter**

indicates that the comparison is a logical, case-sensitive character comparison. Arguments for the DATA or HISTORY keywords that are used to locate data within the record are always case insensitive, regardless of the value specified for the COMPARE keyword. If COMPARE(CSCHARACTER) is specified and one or both operands for the comparison are DATA or HISTORY, a case-insensitive find is done.
in the record to locate the data or history information. Then, a case-sensitive comparison is done between that data and the other compare operand.

If COMPARE(CSCHARACTER) is used with the TEXT keyword, a case-sensitive comparison is not performed since case-sensitive searches are not supported in freeform text areas (e.g., multiple-line problem description).

**Decimal**

indicates that if both operands are decimal (including integers) the comparison is algebraic. If one operand is character or contains DBCS numeric characters, the comparison is logical and not case sensitive.

For a decimal comparison, the EBCDIC characters represent a positive or a negative number (base 10) *with or without* a decimal point ('−001.2 'C or '40404060F0F0F14BF240'X). The following steps are performed:

1. Leading and trailing blanks are removed.
2. The operand having the lesser precision is padded with zeroes to match the precision of the other operand.
3. Any decimal points are removed from the operands.
4. The operands are converted to packed decimal format.
5. The operands are compared algebraically.

For example, '−001.2 'C ('40404060F0F0F14BF240'X) and '1'C ('F1'X) are converted to '012D'X and '010F'X. The second operand is then determined to be greater because it is positive.

**Integer**

indicates that if both operands are integer, the comparison is algebraic. If one operand is character or contains DBCS numeric characters, the comparison is logical and not case sensitive.

For an integer comparison, the EBCDIC characters represent a positive or a negative number (base 10) *without* a decimal point ('−0012 'C or '40404060F0F0F1F240'X).

The following steps are performed:

1. Leading and trailing blanks are removed.
2. The operands are converted to packed decimal format.
3. The operands are compared algebraically.

For example, '−0012 'C ('40404060F0F0F1F240'X) and '1'C ('F1'X) are converted to '012D'X and '001F'X. The second operand is then determined to be greater because it is positive.

Integer is the default.

**Operator**

identifies the logical operator to use for the comparison. The valid operators are defined in the following list. You can use either expression.

- **=** or **EQ**
  
The values are equal.

- **=** or **NE**
  
The values are not equal.
The first value is less than the second value.

The first value is greater than the second value.

The first value is less than or equal to the second value.

The first value is greater than or equal to the second value.

The first value is not less than the second value.

The first value is not greater than the second value.

Length
indicates the number of characters compared.

nnn
is an integer between 1 and 255. A data item that is longer than the specified length is truncated on the right.

When you want to do a logical comparison, if either data item is less than the specified length, it is padded with blanks on the right to this length. However, if you do not specify the LENGTH keyword, the default is the length of the longer data item, with the shorter item right-padded with blanks to this length.

Join
enables you to join two or more IF statements to form one logical expression. The number of statements thus joined is limited only by storage. The conditions given by the joined statements are processed sequentially. As the sequence of conditions is processed, a cumulative setting (true or false) is maintained and ANDed or ORed with the result (true or false) of the test for the next condition according to the operation specified by the JOIN keyword of the current statement. You must add this keyword to all joined statements except the last. Joined statements must be adjacent.

& indicates that the result of the test for the condition specified for the next IF statement is to be logically ANDed with the result accumulated thus far.

| indicates that the result of the test for the condition specified for the next IF statement is to be logically ORed with the result accumulated thus far.

FROM
indicates the position of the substring starting character. Data referenced by the associated data type keyword is used as source for this operation. If an insufficient number of characters is present to satisfy the given keyword value, no data is returned; therefore, no data is used for this part of the comparison.

nnn
is a nonzero integer between -255 and 255. If the value is positive, characters are counted from the beginning of the data to the end. If the value is negative, characters are counted from the end of the data to the beginning. If you do not specify a value for the FROM keyword, a default value of 1 is assumed.

variable
is a 1- to 8-character alphanumeric name, preceded by an ampersand (&), which identifies an integer.
You can specify a FROM keyword for each data type keyword present in a given IF statement. The order of appearance determines the pairing of FROM and data type keywords.

**FOR**

indicates the substring length in characters. Data referenced by the associated data type keyword is used as source for this operation. If the FOR keyword value exceeds the number of characters remaining in the data, only the remaining data is used.

**nnn**

is an integer between 1 and 255. If you do not specify a value for the FOR keyword, the remaining data is used.

**variable**

is a 1- to 8-character alphanumeric name, preceded by an ampersand (&), which identifies an integer.

You can specify a FOR keyword for each data type keyword present in a given IF statement. The order of appearance determines the pairing of FOR and data type keywords.

**WFRom**

indicates the substring starting word. (A word is a delimited string of nonblanks.) The delimiters identifying word boundaries may include SBCS blanks, DBCS blanks, SO, or SI. Data referenced by the associated data type keyword is used as the source for this operation. If an insufficient number of words is present to satisfy the given keyword value, no data is returned; therefore, no data is used for this part of the comparison.

**nnn**

is a nonzero integer between -255 and 255. If the value is positive, words are counted from the beginning of the data to the end. If the value is negative, words are counted from the end of the data to the beginning. If you do not specify a value for the WFRom keyword, a default value of 1 is assumed.

**variable**

is a 1- to 8-character alphanumeric name, preceded by an ampersand (&), which identifies an integer.

You can specify a WFRom keyword for each data type keyword present in a given IF statement. The order of appearance determines the pairing of WFRom and data type keywords.

**WFor**

indicates the substring length in words. Data referenced by the associated data type keyword is used as the source for this operation. If the WFor keyword value exceeds the number of words remaining in the data, only the remaining data is used.

**nnn**

is an integer between 1 and 255. If you do not specify a value for the WFor keyword, the remaining data is used.

**variable**

is a 1- to 8-character alphanumeric name, preceded by an ampersand (&), that identifies an integer.
You can specify a WFOr keyword for each data type keyword present in a given IF statement. The order of appearance determines the pairing of WFOr and data type keywords.

Usage Notes

- Within each search level (0 through 9), you can nest IF and EIF pairs to a maximum depth of 50.
- All command statements specified between a pair of IF and EIF statements are considered to be part of that conditional section.
- A conditional processing section consists of those commands between a corresponding IF and EIF, IF and ELSE, or ELSE and EIF pair but not within any other processing sections that may exist within the given statement pair.
- No definition statements are allowed within a conditional processing section. Because the Report Format Facility processes definition statements during the input phase of processing and command statements during the output phase, IF cannot control the processing of definition statements. However, within the same processing section, an EIF must follow all IF or ELSE statements.
- When using FRom and FOr keywords, truncation of mixed data may occur at the left or right substring boundaries, or at both. The truncation of data observes the integrity of mixed data:
  - No DBCS character is split into its component bytes.
  - An SO character must immediately precede any DBCS portions.
  - An SI character must immediately follow any DBCS portions.
  - Generated null portions are blanked.
- When using WFRom and WFOr keywords, multiple word delimiters encountered side by side are treated as one delimiter.
- Character substringing is performed after word substringing is complete. In other words, the data referenced by the associated data type keyword must first undergo specified word substringing before it is used as source data for character substringing.
- If the data type keyword is DATA, HISTORY, or TEST, only the first occurrence of the matching data in the record is used by the statement. If you need to compare subsequent occurrences, use a more specific argument for the data type keyword or use a CALL exit routine with the locate function. See Figure 46 on page 194 for an example of a more specific argument. See "Locate Service Function—X’D3’" on page 243 for information on the locate function.

Examples

In Figure 43, the IF statement checks for fields in the current record with the prefix “PERA/” that contain the value ’SMITH’.

IF DATA(PERA/.) OPERATOR(=) VALUE(SMITH)

Figure 43. IF Statement to Compare First Occurrence

In Figure 44 on page 194, the IF statement checks for fields in the current record with the prefix “PERA/” that contain the mixed case value ’Smith’. If the data in the record is ’SMITH’ or ’sMith’, the IF statement determines that no data exists.
In Figure 45, the IF statement determines whether freeform text exists in the current record.

IF TEXT(!S0E01) OPERATOR(*) VALUE(&ZIFFIND)

Figure 45. IF Statement to Compare First Occurrence

In Figure 46, the IF statement determines whether the prefix “PERA/SMITH” exists with any associated history data in the current record.

IF VALUE(&ZIFDATA) OPERATOR(*) HISTORY(PERA/SMITH)

Figure 46. IF Statement to Compare Multiple Occurrences of PERA/.

In Figure 47, the Report Format Facility combines the conditions of the joined IF statements in the following manner to form one condition:

\[((\text{cond1} \& \text{cond2}) | \text{cond3}) \& \text{cond4}) | \text{cond5}\]

The combinations of conditions that satisfy the accumulated condition above are:

- E = 1
- D = 1, C = 1
- D = 1, B = 1, A = 1

IF VALUE(&A) OPERATOR(*) VALUE(1) JOIN(&) /* condition 1: A = 1 */
IF VALUE(&B) OPERATOR(*) VALUE(1) JOIN() /* condition 2: B = 1 */
IF VALUE(&C) OPERATOR(*) VALUE(1) JOIN(&) /* condition 3: C = 1 */
IF VALUE(&D) OPERATOR(*) VALUE(1) JOIN() /* condition 4: D = 1 */
IF VALUE(&E) OPERATOR(*) VALUE(1) /* condition 5: E = 1 */

Figure 47. Multiple IF Statements Joined

In the IF statement in Figure 48, the cumulative condition is true, because ‘1..'C (‘F14B4B'X) is compared logically to ‘−1.0'C (‘60F14BF0'X), and ‘1.0'C (‘010F'X in packed decimal format) is compared algebraically to ‘−1.0'C (‘010D'X in packed decimal format).

IF VALUE(1..) OPERATOR(>) VALUE(-1.0) COMPARE(DECIMAL) JOIN(&)
IF VALUE(1.0) OPERATOR(>) VALUE(-1.0) COMPARE(DECIMAL)

Figure 48. Joined IF Statements

PUT

The PUT (P) command statement places data on the output line.
### Syntax of PUT

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PUT { Data{(prefix )} }</td>
<td>Data identifies structured data to be placed on the output line.</td>
</tr>
<tr>
<td>PUT { (variable )}</td>
<td>prefix is a 1- to 5-character alphanumeric code followed by a slash (/) or an underscore (_). The first character must be alphabetic. Use a period after the slash or underscore to retrieve prefix data from the record. For example, STAC/. finds any data associated with the STAC/ prefix. Use any other character to retrieve data that exactly matches the specified string. For example, STAC/OPEN finds the string whether it is the STAC/ prefix with associated data of OPEN or the STAC/OPEN string exists in the description abstract.</td>
</tr>
<tr>
<td>PUT { (dictionary index key) }</td>
<td>variable is a 1- to 8-character alphanumeric name preceded by an ampersand (&amp;), which identifies a p-word or s-word associated with the data item.</td>
</tr>
<tr>
<td>PUT { History{(prefix )}}</td>
<td>dictionary index key is either !P or !S followed by 4 hexadecimal characters, which identifies a p-word or s-word in the dictionary data set associated with the data item.</td>
</tr>
<tr>
<td>PUT { (variable )}</td>
<td>data-type</td>
</tr>
<tr>
<td>PUT { (dictionary index key) }</td>
<td>See &quot;Keywords and Keyword Values&quot; on page 148 for more information about prefixes, variables, and dictionary index keys.</td>
</tr>
</tbody>
</table>

### Data

identifies structured data to be placed on the output line.

**prefix**

is a 1- to 5-character alphanumeric code followed by a slash (/) or an underscore (_). The first character must be alphabetic. Use a period after the slash or underscore to retrieve prefix data from the record. For example, STAC/. finds any data associated with the STAC/ prefix. Use any other character to retrieve data that exactly matches the specified string. For example, STAC/OPEN finds the string whether it is the STAC/ prefix with associated data of OPEN or the STAC/OPEN string exists in the description abstract.

**variable**

is a 1- to 8-character alphanumeric name preceded by an ampersand (&), which identifies a p-word or s-word associated with the data item.

**dictionary index key**

is either !P or !S followed by 4 hexadecimal characters, which identifies a p-word or s-word in the dictionary data set associated with the data item.

See "Keywords and Keyword Values" on page 148 for more information about prefixes, variables, and dictionary index keys.

### History

identifies history data to be placed on the output line. When coding the HISTORY keyword, you must specify one of the following values:
prefix
is a 1- to 5-character alphanumeric code followed by a slash (/) or an underscore (_).
The first character must be alphabetic. A period must follow the slash or underscore
to retrieve data from the record. The prefix identifies the history data item.

variable
is a 1- to 8-character alphanumeric name preceded by an ampersand (&), which
identifies a p-word or s-word.

dictionary index key
is either !P or !S followed by 4 hexadecimal characters, which identifies a p-word or
s-word in the dictionary data set.

Text
identifies freeform text data to be placed on the output line. When coding the TEXT
keyword, you must specify one of the following values:

variable
is a 1- to 8-character alphanumeric name preceded by an ampersand (&), which
identifies an s-word.

s-word index
is !S followed by 4 hexadecimal characters, which identifies an s-word in the
dictionary data set. See "Field Characteristics" on page 319 for a list of freeform-text
s-word index keys.

You can also specify a prefix, a prefix index key, or a variable that contains a prefix
for the TEXT keyword without incurring an error message. However, no data is
found because prefixes are not used to define freeform text.

Value
identifies user-specified data to be placed on the output line. When coding the VALUE
keyword, you must specify one of the following values:

variable
is a 1- to 8-character alphanumeric name, preceded by an ampersand (&), which
identifies a string.

string
is any user-specified data. If the following characters are considered as data, you
must enclose them in single quotes:
- Left parenthesis, '('
- Right parenthesis, ')'
- Exclamation point, '!''
- Ampersand, '&'
- Leading blanks, ' '.

If a single quote is to appear as data, you must code two of them. The Report
Format Facility removes the extra quote during the input phase of processing.

If you want a string to appear as lowercase or mixed case letters, you must enclose
it in quotes. If you do not enclose the string in quotes, all lowercase letters are
converted to uppercase letters.
The length of a string is limited by the maximum continuation for the statement. (Each statement can consist of ten 71-character lines.) However, the value of the LENGTH keyword also limits the number of characters to be operated on. For more information, see the usage notes on page 202.

Column
indicates the beginning column for the data in the report.

nnn
is a decimal number between 1 and the current line length that indicates the beginning column length. The current line length is discussed in the usage notes on page 200.

C centers the data on the entire output line. If there is already data on the output line, that data is first written to the output data set and the new data is then centered on the next line.

When you specify COLUMN(C), you must follow this PUT command with a SPACE command or a PUT or CALL command specifying a COLUMN keyword value that is less than the resulting current line position. This ensures that only the centered data is on the line and the next PUT or CALL does not write data to the same output line. You cannot use COLUMN(C) with the OPERATOR(H) keyword.

S specifies that the Report Format Facility is to use the default separation amount specified on the SECTION statement to position the data on the output line. The rules for default separation are specified in the usage notes on page 201.

Separation
defines the number of blanks to use for separating fields on the output line. You cannot use this keyword if you use COLUMN(C) or COLUMN(S).

nnn
is a decimal number between 0 and the current line length minus 1. The current line length is defined in the usage notes on page 200. The rules specified in the usage notes for default separation apply in this case, except that the specified separation amount is used instead of the default amount from the SECTION statement.

Operator
indicates that multiple data items (such as history and text data) are to be retrieved from the record and defines the subsequent action to be performed if multiple items are found.

H indicates horizontal iteration; this value causes the Report Format Facility to write multiple data items across the output line, beginning in the specified or calculated column position. For an example, see Figure 49 on page 203. If the TAB keyword is used, it specifies the start column position for the multiple data items. If the LENGTH keyword is used, it indicates the total number of characters on the output line, not the length of each individual data item. For more details on horizontal iteration, see the usage notes on page 202. You cannot use OPERATOR(H) with the JUSTIFY, COLUMN(C), or PAD keywords.

V indicates vertical iteration; this value causes the Report Format Facility to write multiple data items one per line, beginning in the specified or calculated column position. For an example, see Figure 49 on page 203.

N indicates no iteration; this value causes the Report Format Facility to write only the first data item beginning in the specified or calculated column position. N is the default.
DTFORM(form)
defines the form of a date or time value to return.

UT
returns the Universal Time date or time (internal format).

OLOCAL
returns the original local date or time (internal format).

ULOCAL
returns the date or time in the current user’s local time zone and external date
format. ULOCAL is the default.

Length
indicates the number of characters to be placed on the output line.

nnn
is a decimal number between 1 and 255. The precise meaning of the LENGTH
keyword depends on the iteration specified with the OPERATOR keyword. For more
information, see the usage notes [201].

Pad
defines the character to be inserted in the output field if the data is shorter than the
specified length. If you use this keyword, you must also specify a LENGTH.

x is a single alphanumeric SBCS character or a blank. You must enclose a blank in
quotes, for example, PAD(" "). The default pad character is a blank. You cannot use
DBCS characters for the PAD value.

Justify
indicates how you want data aligned in the output field. If you specify the JUSTIFY
keyword, you must also specify LENGTH. If required, the field is padded with blanks or
with the specified pad character.

L indicates left justification; data is aligned at the left column margin. L is the default.

R indicates right justification; data is aligned at the right column margin.

Minlines
indicates the number of lines that must be available on the current page to write the data
on the page.

nnn
is a decimal number between 1 and the current page length. (Page length is specified
in the user’s profile or when prompted for.) The default is one line.

If the specified number of lines is not available on the page, the Report Format Facility
starts a new page.

PRecision
indicates the number of decimal places to which to round or zero-pad numeric data.
Numeric data with a whole portion of more than 15 significant digits or fractional
portions of more than 10 digits is not affected by this keyword. Nonnumeric data is also
unaffected. If you specify the PRECISION keyword, you must also specify the
LENGTH keyword.

nn is an integer between 0 and 10.

Numeric data is handled in the following manner:
Leading zeros and blanks, as well as trailing blanks, are removed.

If the precision of the data is less than the precision specified, the number is right-padded with zeroes (and possibly a decimal point) to the specified precision.

If the precision of the data is greater than the precision specified, rounding to the specified precision occurs.

If the resulting data is too large for the output field defined by the values specified for the LENGTH and COLUMN keywords, right truncation occurs.

**FROM**

indicates the position of the substring starting character. Data referenced by the associated data type keyword is used as source for this operation. If an insufficient number of characters is present to satisfy the given keyword value, no data is returned as output.

**nnn**

is a nonzero integer between -255 and 255. If the value is positive, characters are counted from the beginning of the data to the end. If the value is negative, characters are counted from the end of the data to the beginning. If you do not specify a value for the FROM keyword, a default value of 1 is assumed.

**variable**

is a 1- to 8-character alphanumeric name, preceded by an ampersand (&), which identifies an integer.

You can specify a FROM keyword for each data type keyword present in a given PUT statement. The order of appearance determines the pairing of FROM and data type keywords.

**FOR**

indicates the substring length in characters. Data referenced by the associated data type keyword is used as source for this operation. If the FOR keyword value exceeds the number of characters remaining in the data, only the remaining data is used.

**nnn**

is an integer between 1 and 255. If you do not specify a value for the FOR keyword, the remaining data is used.

**variable**

is a 1- to 8-character alphanumeric name, preceded by an ampersand (&), which identifies an integer.

You can specify a FOR keyword for each data type keyword present in a given PUT statement. The order of appearance determines the pairing of FOR and data type keywords.

**WFRom**

indicates the substring starting word. (A word is a delimited string of nonblanks.) The delimiters identifying word boundaries may include SBCS blanks, DBCS blanks, SO, or SI. Data referenced by the associated data type keyword is used as source for this operation. If an insufficient number of words is present to satisfy the given keyword value, no data is returned as output.

**nnn**

is a nonzero integer between -255 and 255. If the value is positive, words are counted from the beginning of the data to the end. If the value is negative, words
are counted from the end of the data to the beginning. If you do not specify a value for the WFRom keyword, a default value of 1 is assumed.

**variable**

is a 1- to 8-character alphanumeric name, preceded by an ampersand (&), which identifies a string.

You can specify a WFRom keyword for each data type keyword present in a given PUT statement. The order of appearance determines the pairing of WFRom and data type keywords.

**WFOr**

indicates the substring length in words. Data referenced by the associated data type keyword is used as source for this operation. If the WFOr keyword value exceeds the number of words remaining in the data, only the remaining data is used.

**nnn**

is an integer between 1 and 255. If you do not specify a value for the WFOr keyword, the remaining data is used.

**variable**

is a 1- to 8-character alphanumeric name, preceded by an ampersand (&), which identifies an integer.

You can specify a WFOr keyword for each data type keyword present in a given WFOr statement. Order of appearance determines the pairing of WFOr and data type keywords.

**Usage Notes**

- The Report Format Facility obtains the current line and page lengths for the report from the user’s profile when the report is run, as follows:
  - The page length is in the **Lines per page** field.
  - The current line length is calculated using the **Logical record length** (LRECL) field based on the record format, as follows:
    - **VA or VBA**
      - LRECL minus 5
    - **V or VB**
      - LRECL minus 4
    - **FA or FBA**
      - LRECL minus 1
    - **F or FB**
      - same as LRECL

    For SYSOUT destinations, the record format defaults to VBA.

- Each processing of a PUT command can move multiple data items, one at a time, to the output line (buffer). Each data item can be a single character or a string of up to 255 characters.

  The Report Format Facility uses a current column pointer to position data on the output line. It indicates the next available position where you can place data on the output line. The current column pointer begins at column 1 for a new line and advances across the line as you place the data on the output line.

  The keywords on the PUT command work with the current column pointer to position the data items on the output line.
In general, the Report Format Facility places data on the current output line as long as the starting column (as specified or defaulted on the PUT command) is at or beyond the current column pointer. The output line is written to the output data set only when one of the following occurs:

- A SPACE or EJECT command is processed.
- A message is written to the output data set.
- An ESECTION (end of section) statement is processed.
- An attempt is made to place data on the output line in front of the current column pointer. For example, suppose the last PUT command placed 8 characters of data on the output line beginning at column 41. The current column pointer is now at column 49. If the next PUT command attempts to place data in column 48, the current line must be written to the data set before the new data is put on the output line, or the existing data will be overwritten.

The placement of data is an important consideration when you design your own RFTs. For an example of common problems you might find and how you can avoid them, see “Example 4: Overcoming Data Output Problems” on page 304.

■ When using FRom and FOr keywords, truncation of mixed data may occur either at the left or right substring boundaries, or at both. The truncation of data observes the integrity of mixed data:
  - No DBCS character is split into its component bytes.
  - An SO character must immediately precede any DBCS portions.
  - An SI character must immediately follow any DBCS portions.
  - Generated null portions are blanked.

■ When using WFRom and WFOr keywords, multiple word delimiters encountered side by side are treated as one delimiter.

■ The Report Format Facility uses either the COLUMN or the SEPARATION keyword, or both, to position the data. If you do not specify either of these keywords, the Report Format Facility positions the data according to the following rules:

1. In column 1, if there is no data currently on the output line.

2. In the column calculated by adding the default separation amount (specified on the SECTION statement) to the current column position, if there is data currently on the output line. The current column position is the column position following the last character written to the output line.

   If DBCS data is present, all SO and SI characters that would stand next to each other on concatenation are removed, unless they have been assigned to column positions.

3. If the calculation in rule 2 exceeds the current line length, the current output line is written to the output data set. Then the new data is put in column 1.

Assuming that you specified horizontal iteration (OPERATOR(H)), Table 3 on page 202 summarizes the order of precedence for the COLUMN and SEPARATION keywords and the TAB command for data placement on the output line.
Table 3. Data Placement on the Output Line, Depending on COL, SEP, and TAB Specifications

<table>
<thead>
<tr>
<th>Specified COL</th>
<th>Specified SEP</th>
<th>Specified TAB</th>
<th>Data Position of First Item</th>
<th>Data Position of Other Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y</td>
<td>Y</td>
<td>Y/N</td>
<td>Use specified column</td>
<td>Add current column and specified separation</td>
</tr>
<tr>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Use specified column</td>
<td>Use next tab position</td>
</tr>
<tr>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>Use specified column</td>
<td>Add current column and default separation</td>
</tr>
<tr>
<td>N</td>
<td>Y</td>
<td>Y/N</td>
<td>Add current column and specified separation</td>
<td>Add current column and specified separation</td>
</tr>
<tr>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>Add current column and default separation</td>
<td>Use next tab position default separation</td>
</tr>
<tr>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Add current column and default separation</td>
<td>Add current column and default separation</td>
</tr>
</tbody>
</table>

If a data item does not fit on the current output line, the current output line is written to the output data set. The data item is then positioned on the next line, beginning in the specified or calculated column if the COLUMN keyword was specified, or in column 1 if the column keyword was not specified.

- The following keyword combinations are not valid on the PUT command:
  - COLUMN and SEPARATION without OPERATOR(H)
  - COLUMN(C) with SEPARATION
  - COLUMN(S) with SEPARATION
  - OPERATOR(H) with
    - JUSTIFY
    - PAD
    - COLUMN(C)

- The following keywords are not valid without LENGTH:
  - JUSTIFY
  - PAD

- If you specify horizontal iteration with the OPERATOR(H) keyword, the Report Format Facility determines the length as follows:
  - The default length is the line length minus the specified or calculated column position of the first data item.
  - If the specified length exceeds the default length, the default is used.
  - The LENGTH keyword specifies the number of characters you may place on the output line.
  - If multiple data items fit on the current line, the Report Format Facility splits the data at a blank (word boundary).
  - If the length of an individual data item exceeds the specified or defaulted length, it is not truncated. Instead, the Report Format Facility breaks the item at the specified or calculated line length and writes it to the output data set. The Report Format Facility then writes the remainder of the data item to the next lines, starting in the specified or calculated column if you specified the COLUMN or in column 1 if you did not. The Report Format Facility makes no attempt to split the data at a blank.
If you do not specify horizontal iteration with the OPERATOR keyword, the length of
the output line is determined as follows:

- The default length is the length of the data item (which cannot exceed 255
  characters).
- A data item that is longer than the specified length is truncated. A data item that is
  shorter is padded with blanks or the specified pad character.

Character substringing is performed after word substringing is complete. In other words,
the data referenced by the associated data type keyword must first undergo specified
word substringing before it is used as source data for character substringing.

Substringing (FOR, FROM, WFOR, WFROM keywords) is ignored on the PUT
statement with the HISTORY keyword when HFORMAT is active.

Substringing (FOR, FROM, WFOR, WFROM keywords) is ignored on the PUT
statement with the TEXT keyword when TFORMAT is active.

Examples

Figure 49 illustrates horizontal and vertical iteration.

0(H) horizontal iteration looks like this

0(V) vertical
  iteration
  looks
  like
  this

Figure 49. Examples of Output from the OPERATOR Keyword

Figure 50 and Figure 51 on page 204 illustrate how the PUT statement places data on an
output line. These examples do not illustrate actual report output.

The PUT statement in Figure 50 places the contents of variable DATA in the first column of
the output line for a length of 7 with right justification and precision 2. Blank padding is the
default.

```
PUT VALUE(&DATA) PRECISION(2) LENGTH(7) COLUMN(1) JUSTIFY(R)

VALUE OF OUTPUT
&DATA LINE
12.999 13.00
1234.559 1234.56
0001 1.00
123456.78 123456.
abc abc
```

Figure 50. PUT Statement Using Right Justification

The PUT statement in Figure 51 on page 204 places the contents of variable DATA in the
first column of the output line for a length of 7 with left justification and precision 2.

```
PUT VALUE(&DATA) PRECISION(2) LENGTH(7) COLUMN(1) JUSTIFY(L)

VALUE OF OUTPUT
&DATA LINE
12.999 13.00
1234.559 1234.56
0001 1.00
123456.78 123456.
abc abc
```

Figure 51. PUT Statement Using Left Justification
The PUT statement in Figure 52 places the specified value on the output line beginning in column 3.

```
PUT VALUE(&DATA) PRECISION(2) LENGTH(7) COLUMN(1) JUSTIFY(L) OUTPUT
&DATA  LINE
  12.999  13.00
  1234.559  1234.56
  0001  1.00
  123456.78  123456.
  abc  abc
```

Figure 51. PUT Statement Using Left Justification

The PUT statement in Figure 52 places the specified value on the output line beginning in column 3.

```
PUT VALUE(THIS STATEMENT HAS TOO MANY BLANKS) COLUMN(003)
```

Figure 52. PUT Statement to Begin Value in a Specific Column

The PUT statements in Figure 53 display journal data for the owning privilege class if it exists. The first PUT places the specified value on the output line beginning in column 1. The second PUT retrieves and places each entry horizontally across the page using the default separation.

```
IF HISTORY(CLAO/.) OPERATOR(=) VALUE(&ZIFDATA)
  PUT COLUMN(001) VALUE(OWNING PRIV. CLASS -- ) MINLINES(2)
  PUT COLUMN(023) HISTORY(CLAO/.) OPERATOR(H)
EIF
```

Figure 53. PUT Statements to Display Journal Data

Figure 54 illustrates how to use data type keywords. Other command statements use the data type keywords illustrated similarly; however, neither vertical nor horizontal iteration can be performed for these other statements.
Figure 55 illustrates how to use VARIABLE to display a data type keyword value.

```plaintext
PUT DATA(IS0B59) /* s-word substituted for s-word index and */ /* used as search argument within structured */ /* data portion of record. */ /* Data item retrieved. */

PUT DATA(!P01AC) /* p-word with truncation character */ /* substituted for p-word index and used */ /* as search argument within structured data */ /* portion of record. Data item retrieved. */

PUT DATA(PERS/.) /* p-word with truncation character used */ /* directly as search argument within */ /* structured data portion of record. */ /* Data item retrieved. */

PUT HISTORY(IS0B59) /* s-word substituted for s-word index and */ /* used as search argument within */ /* journal-history portion of record. */ /* History item retrieved. */

PUT HISTORY(PERS/.) /* p-word with truncation character used */ /* directly as search argument within */ /* journal-history portion of record. */ /* History item retrieved. */

PUT TEXT(IS0E01) /* s-word substituted for s-word index and */ /* used as search argument within */ /* freeform text portion of record. */ /* Data item retrieved. */

Figure 54. Examples of PUT with Data Type Keywords

Figure 55. Using a Variable to Display a Keyword Value

SEARCH - ESEARCH

RFT search processing consists of searching the database for a user-specified set of records in the same manner as the Tivoli Information Management for z/OS SEARCH command. When you issue the REPORT command, Tivoli Information Management for z/OS automatically searches the database using the most recent search argument. The records are passed to the Report Format Facility as the first search level (level zero).

Syntax of SEARCH - ESEARCH

```plaintext
SEARCH [ Argument (argument) ] [ MERge[N|P|n] ] [ Execute[C|U] ] [ Database(n) ] [ Sort (field[A|D] field[A|D] ...)] [ MAP [Y|N] ]
```

Also, when you issue the PRINT record command, the entire database is available as the zero level search results, and the requested record is available without any search statements. The SEARCH (SEA) command statement defines the beginning of a search level. The
ESEARCH (ESEA) command statement defines the end of a search level. A SEARCH command must be within a report section and followed by a corresponding ESEARCH command.

A report section can have up to 10 search levels. By using the SEARCH and ESEARCH command statements in the RFT, you specify the boundaries for all search levels except the first one (level zero).

You might perform multiple-level searches for the following reasons:

- To identify parent and child records, such as changes and activities
- To identify records that are associated through record identifier or privilege class name, such as the change that caused the problem or the change that fixed the problem
- To sort the search-results list in an order other than alphanumeric. For example, you could order problem records according to status by doing three searches: initial, open, and closed.

Figure 56 illustrates nested pairs. You can define up to nine (eight nested) search levels using the SEARCH and ESEARCH command statements. You also can define an unlimited number of SEARCH and ESEARCH pairs at the same level. In this case, the pairs are not nested, and only one pair is active at a time for that level. The search result is replaced by the most recent result at the same level.

```
SEARCH /* begins level 1
SEARCH /* begins level 2
ESEARCH /* ends level 2
SEARCH /* begins second level 2
ESEARCH /* ends second level 2
SEARCH /* begins third level 2
ESEARCH /* ends third level 2
ESEARCH /* ends level 1
SEARCH /* begins second level 1
ESEARCH /* ends second level 1
```

Figure 56. Example of Nested Searches

The SEARCH and ESEARCH pairs also represent processing loops. The Report Format Facility processes each of the command statements contained within a search level once for each record returned by the search. Thus, nested pairs of SEARCH and ESEARCH statements define nested processing loops.

The END command statement causes processing of the search level in which it is specified to end. See “END” on page 184 for more information on this command statement.

**Argument**

- **specifies the search argument to be used.**

  **argument**

  is an argument that contains any combination of prefixes, variables, s-word, prefix indexes, and freeform arguments. The search operators defined for the Tivoli Information Management for z/OS SEARCH command are allowed.
If you did not specify the ARGUMENT keyword, the search returns all records in the database.

You can specify a freeform search argument to be case sensitive by enclosing the argument within a set of single quotation marks.

For example, SEARCH ARGUMENT (‘PERS/Smith’) looks for records containing Smith but not SMITH. The search argument SEARCH ARGUMENT (PERS/Smith) (without the quotation marks) looks for all records that have PERS/SMITH cognized.

**MERge**

combines the results of this search with the results of a previous (higher) level search; that is, the merged results contain only records that were found in both searches. If none of the records are in both searches, the merged result is null, the same as if the search had found no records.

**N** indicates no merging of search-level results. You can use MERGE(N) to run a report on the entire database, not just the search results passed to the Report Format Facility at the beginning of report processing.

**P** indicates that the results of this search are to be combined with the results of the previous (higher) search level. If this is level 1, the results are combined with the level 0 search results. However, if the RFT is being processed as the result of a PRINT record command, the zero level search result is the whole database. P is the default.

**n** indicates that the results of this search are to be combined with the results from the level n search. n is a decimal number between 0 and 8 and must be a higher level, which is numerically lower than the current level. For example, if the current level is 3, n must be between 0 and 2. If 0 is specified or defaulted, and the RFT is being processed as the result of a PRINT record command, no merging takes place.

**Execute**

indicates whether the search-level processing applies if no records are returned for the search.

**C** requests conditional processing; the entire search level is bypassed if the search does not find any records, if a merge results in no records, or if the current privilege class does not have display authority for any of the records in the merged results. C is the default.

**U** requests unconditional processing; you always process the search level at least once, even if no records are available.

**Database**

specifies the number of the database to search. If merging is to take place, the database that is the target of the merge must be the same database indicated with the DATABASE keyword.

Only one database is open at a time. Therefore, nesting search levels with multiple databases can cause a loss of report data integrity. This loss can even occur if the databases are specified to be open for the entire processing of the report section.

**n** is an integer between 0 and 9.

**Sort**

specifies the order in which to process the records returned by the search.
field
consists of any combination of prefixes, s-word, prefix indexes, or variables that
contain a p-word or s-word. The p-word or s-word identifies fields that are to be
used to sort the records. You must specify them in the desired sequence and separate
them by at least one blank. For example, if problems are to be sorted by date
occurred and within date occurred by assignee name, the SORT keyword is
expressed as: SORT(DATO/. PERA/.).

Prefixes must be followed by a period to be recognized by the SORT command.

A indicates that the sort order for the associated field is ascending. A is the default.

D indicates that the sort order for the associated field is descending.

If you do not specify the SORT keyword, the Report Format Facility processes the
records as they are found in the database.

MAp
indicates whether map processing is to be done for component configuration records
obtained by this search.

Y requests the following for each component returned by this level:

■ Tivoli Information Management for z/OS automatically searches the entire
database to determine the next components in the hierarchy that are upwardly
connected to the first hierarchy component. This search does not increase the
search level counter.

■ Any components found as a result of this search and all components upwardly
connected to them are processed in hierarchical order. The search is repeated as
many times as necessary to identify all components in the hierarchy.

N requests no map processing. N is the default.

Usage Notes

■ The Report Format Facility considers all command statements you specify between a
pair of SEARCH and ESEARCH statements to be part of that search level. You cannot
use definition statements, other than EOD and EEOD, as part of a search level.

■ A SEARCH statement must be followed by a corresponding ESEARCH within the same
processing section.

■ For the SORT keyword value, the data retrieved from the fields associated with the
prefixes and s-words is sorted to determine the order in which the records are processed.
If a field has more than one value, such as when sorting on a prefix that is included in a
record several times, the record is processed once for each different value. To retrieve
the current value, you must use the predefined sort variables, rather than the sort
keyword value.

For more information on the predefined sort variables, see "Predefined Variables" on
page 363.

■ Because all child records contain the same s-word as their parents, you must use
Boolean NOT logic for the child records if you want to search for the parent records
alone. Table 4 on page 209 lists the required s-word index combinations for each parent.
Table 4. Required S-Word Index Combinations for Parent Records

<table>
<thead>
<tr>
<th>Record Type</th>
<th>Search Argument</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Change</td>
<td>!S0B06 ~ !S0B07</td>
<td>Change, NOT Activity</td>
</tr>
<tr>
<td>Hardware Component</td>
<td>!S0B0F ~ !S0B10 ~ !S0AF8 ~ !S0B1E</td>
<td>Component, NOT Feature, NOT Subcomponent, NOT Connection</td>
</tr>
<tr>
<td>Software Component</td>
<td>!S0B13 ~ !S0B14 ~ !S0B1F</td>
<td>Component, NOT Feature, NOT Connection</td>
</tr>
<tr>
<td>Hardware Model Component</td>
<td>!S0AF4 ~ !S0AF5 ~ !S0AF6</td>
<td>Model Component, NOT Model Feature, NOT Model Subcomponent</td>
</tr>
<tr>
<td>Software Model Component</td>
<td>!S0AFB ~ !S0AFC</td>
<td>Model Component, NOT Model Feature</td>
</tr>
<tr>
<td>Parent Panel Set</td>
<td>!S0B26 ~ !S0B27</td>
<td>Parent, NOT Child</td>
</tr>
</tbody>
</table>

- When you specify dates for the ARGUMENT keyword value, you must enter all dates in the YYYY/MM/DD internal date format used by Tivoli Information Management for z/OS (for example, date/1999/07/26).
- The date indicator specified in the panel that collected the date or a prefix that begins with DAT identifies a date field. If you sort on a date field, the date is converted into internal format before it is sorted.
- If data for a particular field is cognized in mixed case format, be sure to remember to enter any freeform search arguments with the appropriate search operator (that is, enclose each argument within a set of single quotation marks). For example, if the Status, Assignee Name, and System Name fields are cognized in mixed case at your location, enter the search argument as:

```
SEARCH ARGUMENT ('STAC/OPEN' 'PERA/Wilson' 'NASY/Acctg')
```

In this example, only those records with data matching the specified case would be retrieved. (For example, any records with open, WILSON, or acctg would be ignored.)

Regardless of how data is cognized (in mixed case format or uppercase), sorting is performed as if the data is uppercase.

Support for mixed case search arguments is provided for users of Latin translate tables only.

For more information on using search arguments, refer to the Tivoli Information Management for z/OS User’s Guide.

Examples

The SEARCH argument in Figure 57 on page 210 finds all of the Tivoli Information Management for z/OS (DATABASE(5)) problem records in the user-specified date range. The results are merged with the search results of the REPORT command (MERGE(0)) that must also have been issued against database 5. Search processing continues only if at least one record is found that meets the search requirements (EXECUTE(C)). Records are sorted first by date entered and then by record ID. This example is extracted from the Periodic Problem Status RFT (BLMZZ21).
In Figure 58, the first SEARCH finds all change records that have activities associated with them. A variable is set with the change record ID. A second SEARCH retrieves all the activity records that have the change record ID as the parent record. This example is extracted from the Changes with Related Activities RFT (BLMZZ35).

```
SEARCH ARGUMENT(!S0B06 ¬!S0B07 !S0CBC)
  SET NAME(CHANGE) DATA(RNID/.)
  .
  .
  SEARCH ARGUMENT(!S0B07 RNOR/&CHANGE)
```

*Figure 58. SEARCH for Finding Child Records*

In Figure 59, the SORT keyword on the SEARCH statement causes the resulting records to be sorted first by record ID in ascending order, second by date entered in descending order, and third by initial priority in ascending order.

```
SEARCH ARGUMENT(!S0B07 RNOR/&CHANGE)
  SORT(RNID/.,A DATE/.,D PRII/.)
  .
  .
  ESSEARCH
```

*Figure 59. SEARCH with SORT Keyword*

The SEARCH statement in Figure 60 finds all “reported by” names starting with J<D1. This example shows the use of DBCS data in the ARGUMENT keyword.

```
SEARCH ARGUMENT(PERS/J<D1W.>)
  .
  .
  ESSEARCH
```

*Figure 60. SEARCH with DBCS Data in the ARGUMENT Keyword*

SET

The SET command statement defines and initializes variables, and performs arithmetic operations. See “Predefined Variables” on page 363 for information on predefined variables.
The SET command statement defines and initializes variables, and performs arithmetic operations. See "Predefined Variables" on page 363 for information on predefined variables.

**Name**
identifies the variable to receive the data or the result of the arithmetic operation.

**variable**
is a 1- to 8-character alphanumeric name that is a user-defined or a Report Format Facility read/write variable. Unlike variable names on other keywords, the ampersand is not permitted as the first character on the NAME keyword.

You need not define a variable before you use it in the RFT. During the input phase, a variable that you have not defined on a SET command is defined and automatically set to a null value. During the output phase, if the variable has a null value, it is ignored.

**Length**
indicates the number of characters of data the variable represents. If the data is less than this length, it is padded with blanks on the right; if the data exceeds this length, it is truncated from the right.

**nnn**
is an integer between 1 and 255. If you do not specify the LENGTH keyword, the
Report Format Facility default is 16. However, this default length is not directly used in determining the length of the value assigned to the variable.

- First, the largest length (specified or default) is determined across all SET and SETD statements appearing in the current report section for the current variable.
- Then, the largest length is compared to the actual data length, and the lesser value indicates the number of characters of the data that the variable represents.

Data
identifies structured data to be placed into the variable or used in a calculation.

prefix
is a 1- to 5-character alphanumeric code followed by a slash (/) or an underscore (_). The first character must be alphabetic. Use a period after the slash or underscore to retrieve prefix data from the record. For example, STAC/. finds any data associated with the STAC/ prefix. Use any other character to retrieve data that exactly matches the specified string. For example, STAC/OPEN finds the string whether it is the STAC/ prefix with associated data of OPEN or the STAC/OPEN string exists in the description abstract.

variable
is a 1- to 8-character alphanumeric name preceded by an ampersand (&), which identifies a p-word or s-word in the dictionary data set. The p-word or s-word identifies the data item.

dictionary index key
is either !P or !S followed by 4 hexadecimal characters, which identifies a p-word or s-word in the dictionary data set. The p-word or s-word identifies the data item.

See “Keywords and Keyword Values” on page 145 for more information about prefixes, variables, and dictionary index keys.

History
identifies history data to be placed into the variable or used in a calculation.

prefix
is a 1- to 5-character alphanumeric code followed by a slash (/) or an underscore (_). The first character must be alphabetic. A period must follow the slash or underscore to retrieve data from the record. The prefix identifies the history data item.

variable
is a 1- to 8-character alphanumeric name preceded by an ampersand (&), which identifies a p-word or s-word associated with the data item.

dictionary index key
is either !P or !S followed by 4 hexadecimal characters, which identifies a p-word or s-word in the dictionary data set. The p-word or s-word identifies the data item.

Text
identifies freeform text data to be placed into the variable or calculation.

variable
is a 1- to 8-character alphanumeric name, preceded by an ampersand (&), which identifies an s-word associated with the data item.
s-word index

is !S followed by 4 hexadecimal characters, which represents an s-word in the
dictionary data set. (See "Field Characteristics" on page 319 for a list of s-word
index keys for freeform text.)

Note that you must use substringing to use text in a calculation.

You can also specify a prefix, a prefix index key, or a variable that contains a prefix for
the TEXT keyword without incurring an error message. However, no data is found
because prefixes are not used to define freeform text.

Value
tidentifies user-specified data to be placed into the variable or used in a calculation.

variable

is a 1- to 8-character alphanumeric name, preceded by an ampersand (&), which
identifies a string.

string

is any user-specified data. If the following characters are to be considered as data,
you must enclose them in single quotes:

- Left parenthesis, '

- Right parenthesis, '

- Exclamation point, '!'

- Ampersand, '&'

- Leading blanks, ' '.

If a single quote is to appear as data, you must code two of them. The Report
Format Facility removes the extra quote during the input phase of processing.

The length of a string is limited by the maximum continuation for the statement.
(Each statement can consist of ten 71-character lines.) However, the value of the
LENGTH keyword also limits the number of characters to be operated on.

Operator

indicates the type of arithmetic operation to be performed. Arithmetic operations cannot
be performed on DBCS numeric characters.

+  Adds the first data item value to the second.

–  Subtracts the second data item value from the first.

*  Multiplies the first data item value by the second.

%  Divides the first data item value by the second.

As with integer operations, right truncation or padding is performed to match the
size of the result to the size of the output variable, and the result is left justified
in the output variable. A decimal point is imbedded where needed.

/  Divides the first data item value by the second. This is an integer divide; a
remainder is not available.

//  Divides the first data item value by the second and returns only the remainder.

WFRom

indicates the substring starting word. (A word is a delimited string of nonblanks.) The
delimiters identifying word boundaries are SBCS blanks, DBCS blanks, SO, or SI. Data
referred to the associated data type keyword is used as source for this operation. If an insufficient number of words is present to satisfy the given keyword value, no data is returned.

**nnn**

is a nonzero integer between -255 and 255. If the value is positive, words are counted from the beginning of the data to the end. If the value is negative, words are counted from the end of the data to the beginning. If you do not specify a value for the WFRom keyword, a default value of 1 is assumed.

**variable**

is a 1- to 8-character alphanumeric name, preceded by an ampersand (&), which identifies an integer.

You can specify a WFRom keyword for each data type keyword present in a given SET statement. The order of appearance determines the pairing of WFRom and data type keywords.

**WFOr**

indicates the substring length in words. Data referenced by the associated data type keyword is used as source for this operation. If the WFOr keyword value exceeds the number of words remaining in the data, only the remaining data is used.

**nnn**

is an integer between 1 and 255. If you do not specify a value for the WFOr keyword, the remaining data is used.

**variable**

is a 1- to 8-character alphanumeric name, preceded by an ampersand (&), which identifies an integer.

You can specify a WFOr keyword for each data type keyword present in a given SET statement. The order of appearance determines the pairing of WFOr and data type keywords.

**FRom**

indicates the position of the substring starting character. Data referenced by the associated data type keyword is used as source for this operation. If an insufficient number of characters are present to satisfy the given keyword value, no data is returned.

**nnn**

is a nonzero integer between -255 and 255. If the value is positive, characters are counted from the beginning of the data to the end. If the value is negative, characters are counted from the end of the data to the beginning. If you do not specify a value for the FRom keyword, a default value of 1 is assumed.

**variable**

is a 1- to 8-character alphanumeric name, preceded by an ampersand (&), which identifies an integer.

You can specify a FRom keyword for each data type keyword present in a given SET statement. The order of appearance determines the pairing of FRom and data type keywords.

**FOr**

indicates the substring length in characters. Data referenced by the associated data type
keyword is used as source for this operation. If the FOr keyword value exceeds the number of characters remaining in the data, only the remaining data is used.

**nnn**

is an integer between 1 and 255. If you do not specify a value for the FOr keyword, the remaining data is used.

**variable**

is a 1- to 8-character alphanumeric name, preceded by an ampersand (&), which identifies an integer.

You can specify a FOr keyword for each data type keyword present in a given SET statement. The order of appearance determines the pairing of FOr and data type keywords.

**Usage Notes**

- When you request an arithmetic operation, you must precede negative values by a minus sign (–). If an arithmetic operation results in a negative number, a minus sign is automatically appended to the beginning of the variable.

- You cannot perform arithmetic operations with DBCS numeric characters.

- When you specify the OPERATOR keyword, the whole portion of the result must contain no more than 15 significant digits and 10 fractional digits. Exceeding this limit results in an overflow condition. Barring overflow, the result is rounded to 15 or fewer significant digits and 10 or fewer fractional digits.
  - For addition and subtraction, the precision returned is the greater of the precisions of the operands up to 15 significant digits for the overall result.
  - For multiplication, the precision returned is the sum of the precisions of the operands up to 10 significant digits with the additional restriction of 15 significant digits overall.
  - For decimal division, the precision of the returned value is 10 with an overall maximum of 15 significant digits.

- Character substringing is performed after word substringing is complete. In other words, the data referenced by the associated data type keyword must first undergo specified word substringing before it is used as source data for character substringing.

- When using FRom and FOr keywords, truncation of mixed data may occur at the left or right substring boundaries, or at both. The truncation of data observes the integrity of mixed data:
  - No DBCS character is split into its component bytes.
  - An SO character must immediately precede any DBCS portions.
  - An SI character must immediately follow any DBCS portions.
  - Generated null portions are blanked.

- When using WFRom and WFOr keywords, multiple word delimiters encountered side by side are treated as one delimiter.

- The SET command suppresses leading zeroes. Therefore, you can eliminate leading zeroes in data retrieved from the database by adding a value of zero to the data retrieved. See Figure 61 for an example.

- You can use the SET command to perform arithmetic operations on decimal as well as integer values.
Examples

In Figure 61, the first SET statement assigns the record ID for the current record being processed to the variable CHANGE. The second SET statement suppresses leading zeroes in the variable CHANGE. This example assumes the current RNID is numeric.

```
SET NAME(CHANGE) DATA(RNID/.)
SET NAME(CHANGE) VALUE(&CHANGE) OPERATOR(+) VALUE(0)
```

Figure 61. SET Statements to Suppress Leading Zeroes

In Figure 62, the SET statement adds the number of open records from a previous period (PREVOPN) to the total open records (TOTOPN).

```
SET NAME(TOTOPN) VALUE(&TOTOPN) OPERATOR(+) VALUE(&PREVOPN)
```

Figure 62. SET Statement to Add Variables

The SET statements in Figure 63 calculate the percentage of problems that are of type hardware and were entered from 10/01/93 through 10/31/93. The example assumes that a code of HWD is used to identify hardware problems. The internal date format is YYYY/MM/DD.

The first CALL statement finds the number of problem records that were entered in the specified date range. The second CALL statement finds the number of hardware problems among those records.

The first SET statement multiplies the total number of hardware problems by 100. The second SET statement divides the total hardware problems by the total problems. The resulting percentage is a whole number; less than one percent is not detected.

```
CALL NAME(BLGOXCNT) INPUT(!S0B01 DATE/1993/10/01 - DATE/1993/10/31,, - '>&TOTPROB') LENGTH(0)
CALL NAME(BLGOXCNT) INPUT(!S0B01 DATE/1993/10/01 - DATE/1993/10/31 - TYPE/HWD,, '>&TOTHWD') LENGTH(0)
SET NAME(TOTHWD) VALUE(&TOTHWD) OPERATOR(*) VALUE(100)
SET NAME(PCTHWD) VALUE(&TOTHWD) OPERATOR(/) VALUE(&TOTPROB)
```

Figure 63. SET Statements to Calculate a Percentage

In Figure 64, the SET statement first extracts the string ‘mnpqrstuvwxyz’ in the word substringing portion of the operation. Then it extracts the character ‘x’ in the character substringing portion. This result is assigned to variable VAR1.

```
SET NAME(VAR1) VALUE(abcd efgh ijk1 mnpqrstuvwxyz) WFROM(4) WFOR(1)
                     FROM(-3) FOR(1)
```

Figure 64. SET Statement with Word and Character Substringing

In Figure 65, the SET statement divides 3 by 2 and stores the result 1.5 in variable VAR1.
The SETD command statement defines and initializes variables. It also performs date and time arithmetic operations and date format conversions. See "Predefined Variables" on page 363 for information on predefined variables.

**Syntax of SETD**

```
SETD (variable-name) [data-type] [operator] [input] [length] [length]
[character-base] [character-span] [word-base] [word-span]

variable-name
{Adate(variable) [Time(variable)|ITime(variable)]} [DATA {(prefix)}]
{Edate(variable) [Time(variable)|ITime(variable)]} [History {(prefix)}]
{Idate(variable) [Time(variable)|ITime(variable)]} [Value {(variable)}]
{Wday(variable) [Time(variable)|ITime(variable)]} [Value {(string)}]
{Duration(variable)}
{DAYS(variable)}
{Minutes(variable)}
{ITime(variable)}

operator
[Operator {(+)}] [Length{(nnn)}] [INput{(Days)}]
[(-)] [()] [{}]

character-base
[FRom {(variable)}] [FOR {(variable)}]
[{(nnn)}] [{(nnn)}]
[()] [()]

word-base
[WFRom {(variable)}] [WFOr {(variable)}]
[{(nnn)}] [{(nnn)}]
[()] [()]
```
Adate identifies the variable to receive a date result in alphanumeric format; for example, October 2, 1995.

variable is a 1- to 8-character alphanumeric name that is a user-defined or a Report Format Facility read/write variable. The leading ampersand is not permitted for the ADATE keyword.

Edate identifies the variable to receive a date result in external format. The external date format is specified in the DATEFMT keyword of the BLGPARMS session parameters, and can be overridden by users. (Refer to the Tivoli Information Management for z/OS Planning and Installation Guide and Reference for more details.)

variable is a 1- to 8-character alphanumeric name that is a user-defined or a Report Format Facility read/write variable. The leading ampersand is not permitted for the EDATE keyword.

Idate identifies the variable to receive a date result in internal format (YYYY/MM/DD). (Refer to the Tivoli Information Management for z/OS Planning and Installation Guide and Reference for more details about date formats.) The default is YY/MM/DD.

variable is a 1- to 8-character alphanumeric name that is a user-defined or a Report Format Facility read/write variable. The leading ampersand is not permitted for the IDATE keyword.

Wday identifies the variable to receive a day of the week result; for example, Wednesday.

variable is a 1- to 8-character alphanumeric name that is a user-defined or a Report Format Facility read/write variable. The leading ampersand is not permitted for the WDAY keyword.

Time identifies the variable to receive a time result in external format. The default external format is HH:MM. You can use the TIME keyword only in conjunction with a date keyword (ADATE, EDATE, IDATE) or the WDAY keyword.

variable is a 1- to 8-character alphanumeric name that is a user-defined or a Report Format Facility read/write variable. The leading ampersand is not permitted for the TIME keyword.

Duration identifies the variable to receive an extended duration result (DDDD:HH:MM). Negative results are allowed.

variable is a 1- to 8-character alphanumeric name that is a user-defined or a Report Format Facility read/write variable. The leading ampersand is not permitted for the DURATION keyword.
DAYs
identifies the variable to receive a number of days. Negative results are allowed.

variable
is a 1- to 8-character alphanumeric name that is a user-defined or a Report Format Facility read/write variable. The leading ampersand is not permitted for the DAYS keyword.

Minutes
identifies the variable to receive a number of minutes. Negative results are allowed.

variable
is a 1- to 8-character alphanumeric name that is a user-defined or a Report Format Facility read/write variable. The leading ampersand is not permitted for the MINUTES keyword.

ITime
identifies the variable to receive a time result in internal format. The default internal format is HH:MM. You can use the ITIME keyword only with a date keyword (ADATE, EDATE, IDATE) or the WDAY keyword.

variable
is a 1- to 8-character alphanumeric name that is a user-defined or a Report Format Facility read/write variable. The leading ampersand is not permitted for the ITIME keyword.

DATa
identifies structured data used as input.

prefix
is a 1- to 5-character alphanumeric code followed by a slash (/) or an underscore (_). The first character must be alphabetic. Use a period after the slash or underscore to retrieve prefix data from the record. For example, STAC/. finds any data associated with the STAC/ prefix. Use any other character to retrieve data that exactly matches the specified string. For example, STAC/OPEN finds the string whether it is the STAC/ prefix with associated data of OPEN or the STAC/OPEN string exists in the description abstract.

variable
is a 1- to 8-character alphanumeric name, preceded by an ampersand (&), which identifies a p-word or s-word in the dictionary data set. The p-word or s-word identifies the data item.

dictionary index key
is either !P or !S followed by 4 hexadecimal characters, which identifies a p-word or s-word in the dictionary data set. The p-word or s-word identifies the data item.

See “Keywords and Keyword Values” on page 148 for more information about prefixes, variables, and dictionary index keys.

History
identifies history data to be used as input.

prefix
is a 1- to 5-character alphanumeric code followed by a slash (/) or an underscore (_). The first character must be alphabetic. A period must follow the slash or underscore to retrieve data from the record. The prefix identifies the history data item.
**variable**
is a 1- to 8-character alphanumeric name, preceded by an ampersand (&), which identifies a p-word or s-word associated with the data item.

**dictionary index key**
is either !P or !S followed by 4 hexadecimal characters, which identifies a p-word or s-word in the dictionary data set. The p-word or s-word identifies the data item.

**Value**
identifies user-specified data to be used as input.

**variable**
is a 1- to 8-character alphanumeric name, preceded by an ampersand (&), which identifies a string of one of the formats described below.

**string**
is user-specified data of one of the following formats:
- Integer
- External time (user-defined; default is HH:MM)
- Duration (DD:HH:MM)
- Extended duration (DDDD:HH:MM)
- External date (user-defined; default is MM/DD/YYYY).

If the following characters are considered as data, you must enclose them in single quotes:
- Left parenthesis, '('
- Right parenthesis, ')'
- Exclamation point, '!' 
- Ampersand, '&'
- Leading blanks, ' '.

If a single quote is to appear as data, you must code two of them. The Report Format Facility removes the extra quote during the input phase of processing.

The length of a string is limited by the maximum continuation for the statement. (Each statement can consist of ten 71-character lines.)

**Operator**
indicates the type of arithmetic operation to be performed. The default operation is conversion. A conversion is when data is only converted from the source format to the target format. This action occurs when you leave the OPERATOR keyword off.

- + Adds the first data item values to the second.
- - Subtracts the second data item values from the first.

**Length**
indicates the number of characters of data that the variable represents. If the data assigned to the variable is less than this length, it is padded with blanks on the right; if the data exceeds this length, it is truncated from the right.

- nnn

is an integer between 1 and 255. If you do not specify the LENGTH keyword, the Report Format Facility uses a default length of 64 with the EDATE keyword, 20 with the ADATE keyword, and 16 for all other variable names that are valid on a SETD statement. However, this default length is not directly used in determining the length of the value assigned to the variable.
First, the largest length (specified or default) is determined across all SET and SETD statements appearing in the current report section for the current variable.

Then, the largest length is compared to the actual data length, and the lesser value indicates the number of characters of data that the variable represents.

You can specify more than one LENGTH for setting a date and time. The order of appearance determines the pairing of LENGTH and variable-name keywords.

**INput**
qualifies integer data identified by a data type keyword. The INPUT keyword is allowed only for adding or subtracting a number of days, months, or years to or from a date and setting a date result. Therefore, if you specify the INPUT keyword, you must also specify the WDAY, ADATE, EDATE, or IDATE keyword, without the TIME or ITIME keywords.

**Days**
indicates that the integer input to SETD represents a number of days. DAYS is the default.

**Months**
indicates that the integer input to SETD represents a number of months.

**Years**
indicates that the integer input to SETD represents a number of years.

**FRom**
indicates the position of the substring starting character. Data referenced by the associated data type keyword is used as source for this operation. If an insufficient number of characters are present to satisfy the given keyword value, no data is returned.

**nnn**
is an integer between -255 and 255.

- If the value is positive, characters are counted from the beginning of the data to the end.
- If the value is negative, characters are counted from the end of the data to the beginning.
- If you do not specify a value for the FRom keyword, a default value of 1 is assumed.

**variable**
is a 1- to 8-character alphanumeric name, preceded by an ampersand (&), that identifies an integer.

You can specify a FRom keyword for each data type keyword present in a given SETD statement. The order or appearance determines the pairing of FRom and data type keywords.

**FOR**
indicates the substring length in characters. Data referenced by the associated data type keyword is used as source for this operation. If the FOR keyword value exceeds the number of characters remaining in the data, only the remaining data is used.

**nnn**
is an integer between 1 and 255. If you do not specify a value for the FOR keyword, the remaining data is used.
variable
is a 1- to 8-character alphanumeric name, preceded by an ampersand (&), that identifies an integer.

You can specify a FOr keyword for each data type keyword present in a given SETD statement. The order of appearance determines the pairing of FOr and data type keywords.

WFRom
indicates the substring starting word. (A word is a delimited string of nonblanks.) The delimiters identifying word boundaries are SBCS blanks, DBCS blanks, SO, or SI. Data referenced by the associated data type keyword is used as source for this operation. If an insufficient number of words is present to satisfy the given keyword value, no data is returned.

nnn
is a nonzero integer between -255 and 255.
  ■ If the value is positive, words are counted from the beginning of the data to the end.
  ■ If the value is negative, words are counted from the end of the data to the beginning.
  ■ If you do not specify a value for the WFRom keyword, a default value of 1 is assumed.

variable
is a 1- to 8-character alphanumeric name, preceded by an ampersand (&), which identifies an integer.

You can specify a WFRom keyword for each data type keyword present in a given SETD statement. The order of appearance determines the pairing of WFRom and data type keywords.

WFOr
indicates the substring length in words. Data referenced by the associated data type keyword is used as source for this operation. If the WFOr keyword value exceeds the number of words remaining in the data, only the remaining data is used.

nnn
is an integer between 1 and 255. If you do not specify a value for the WFOr keyword, the remaining data is used.

variable
is a 1- to 8-character alphanumeric name, preceded by an ampersand (&), that identifies an integer.

You can specify a WFOr keyword for each data type keyword present in a given SETD statement. The order of appearance determines the pairing of WFOr and data type keywords.

Usage Notes

■ Values of data type keywords must be one of the formats allowed for string input for the VALUE keyword.

The panels shipped with Tivoli Information Management for z/OS support entry of any of the 22 supported external date formats. You can supply a date conversion routine of your own if desired to accommodate your format. For information on date conversion
The date conversion routine also enables you to set the variable specified for the EDATE keyword with a date in your external format. The default external time format is HH:MM, but you can supply a time conversion routine of your own to accommodate your format. For information on time conversion routines, refer to the Tivoli Information Management for z/OS Planning and Installation Guide and Reference. The time conversion routine also enables you to set the variable specified for the TIME keyword with a time in your external format.

If you specify the OPERATOR keyword, you must also specify at least two data type keywords. A SETD statement with no OPERATOR keyword implies that a conversion is to be performed.

Integer values for data type keywords, as well as interim or final integer results of a date/time operation, must be in the range -2 147 483 648 to 2 147 483 647.

When using FRom and FOr keywords, truncation of mixed data may occur at the left or right substring boundaries, or at both. The truncation of data maintains the integrity of mixed data:

- No DBCS character is split into its component bytes.
- An SO character must immediately precede any DBCS portions.
- An SI character must immediately follow any DBCS portions.
- Generated null portions are blanked.

When using WFRom and WFOr keywords, multiple word delimiters encountered side by side are treated as one delimiter.

The SETD command suppresses leading zeroes of integer results.

Time inputs must be nonnegative. For other formats, a minus sign must precede negative inputs, and a minus sign is automatically inserted for negative results.

Tivoli Information Management for z/OS provides various external date formats which support 4-digit years (0000 through 9999), provided that the external date format contains a 4-digit year.

Only one of the values for the data type keywords on a SETD statement can be in integer format.

Addition of dates or date/time pairs is not permitted.

The following represents the four levels of precision for SETD operands and results in descending order:

- Minutes, durations, extended durations, date/time pairs
- Date, days
- Months
- Years

The precision level of operands and results of SETD must be equal. Consider the following:

- The TIME or ITIME keyword is required for adding or subtracting minutes to or from a date/time pair.
- The DAYS keyword is not allowed for subtracting one date/time pair from another. You must use the DURATION or MINUTES keyword.
If one operand for SETD is a date/time pair, the other operand cannot be a date only. It must be a number of minutes, a duration, or a date/time pair.

**Note:** One exception to this rule occurs at the date/day precision level. You can add or subtract a number of months or years to or from a date to yield a date result. This is the only case in which context alone is not sufficient to qualify integer input as minutes, days, months, or years; thus the need for an INPUT keyword.

- The variable-name keywords specified must be adequate to receive all of the resulting data. Consider the following: A date variable-name keyword is required for adding minutes to a date/time pair. Specifying the TIME keyword as the single variable-name keyword is not sufficient.

- Character substringing is performed after word substringing is complete. In other words, the data referenced by the associated data type keyword must first undergo specified word substringing before it is used as source data for character substringing.

- The following conversions are allowed:
  - Duration or extended duration to minutes
  - Minutes to extended duration
  - External date to internal date, alphanumeric date, or day of the week.
  - External time to internal time

- When calculating month-ending dates using the predefined read-only variables, the results may not be as expected. For example, when using the following statement:
  
  ```
  SETD IDATE(MONEND) VALUE(&ZEPMED) OP(-)VALUE(1) INPUT(MONTH) +
  MONEND = 1996/08/30 WHEN ZEPMED = 09/30/1996
  ```

  Although 1996/08/30 is not the last day of the month, it is equal to 09/30/1996 minus 1 month.

- SETD expects three factors that govern the expected order and combination of valid inputs (date, time, duration, integer):
  - The specified variable-name keywords (EDATE, TIME, DURATION, and so forth.)
  - The number of data type keywords
  - The presence of the OPERATOR keyword.

- Because the data type keyword specifications for a SETD statement provide these inputs, it is important that the choice and relative positioning of these specifications adhere to that which SETD expects. [Table 5 on page 225](#) lists, by row, all of the functions SETD can perform, and lists, by column, the inputs SETD expects and order in which they must be presented. To use Table 3, look up the function you want to have performed by starting with the column labeled Variable-Name Keyword (RESULTS). This column indicates the set of result formats that can be obtained for a given operation or row. Read across the row to determine valid data type keywords for the first and second operands and valid operator keywords.
Table 5. SETD Results and Order of Expected Inputs

<table>
<thead>
<tr>
<th>Variable-Name Keywords (RESULTS)</th>
<th>Data Type Keyword data (FIRST OPERAND)</th>
<th>Operator Keyword Value (OPERATION)</th>
<th>Data Type Keyword data (SECOND OPERAND)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADATE/EDATE/IDATE/WDAY &amp; TIME/ITIME</td>
<td>date &amp; time</td>
<td>+/-</td>
<td>duration/minutes</td>
</tr>
<tr>
<td>ADATE/EDATE/IDATE/WDAY</td>
<td>date</td>
<td>+/-</td>
<td>days/months/years</td>
</tr>
<tr>
<td>DURATION/MINUTES</td>
<td>date &amp; time</td>
<td>-</td>
<td>date &amp; time</td>
</tr>
<tr>
<td>DAYS</td>
<td>date</td>
<td>-</td>
<td>date</td>
</tr>
<tr>
<td>DURATION/MINUTES</td>
<td>duration/minutes</td>
<td>+/-</td>
<td>duration/minutes</td>
</tr>
<tr>
<td>DURATION/MINUTES</td>
<td>duration/minutes</td>
<td>none</td>
<td>none</td>
</tr>
<tr>
<td>ADATE/EDATE/IDATE/WDAY</td>
<td>date</td>
<td>none</td>
<td>none</td>
</tr>
<tr>
<td>ITIME</td>
<td>time</td>
<td>none</td>
<td>none</td>
</tr>
</tbody>
</table>

Examples

The third SETD statement in Figure 66 is incorrect because the precision of the results is date, but the precision of the operands is minutes.

```
SETD EDATE(DATE) TIME(TIME) VALUE(&DATE) VALUE(&TIME) OPERATOR(-) +
   VALUE(&MINUTES) /* correct */
```

```
SETD EDATE(DATE) ITIME(TIME) VALUE(&DATE) VALUE(&TIME) OPERATOR(-) +
   VALUE(&MINUTES) /* correct */
```

```
SETD EDATE(DATE) VALUE(&DATE) VALUE(&TIME) OPERATOR(-) +
   VALUE(&MINUTES) /* incorrect */
```

Figure 66. SETD Statement with Results as a Date

The last two SETD statements in Figure 67 are incorrect because the precision of the results is days, but the precision of the operands is minutes.

```
SETD MINUTES(MINUTES) VALUE(&DATE) VALUE(&TIME) OPERATOR(-) +
   VALUE(&MINUTES) /* correct */
```

```
SETD DURATION(DURATION) VALUE(&DATE1) VALUE(&TIME1) OPERATOR(-) +
   VALUE(&DATE2) VALUE(&TIME2) /* correct */
```

```
SETD DAYS(DAYS) VALUE(&DATE1) VALUE(&TIME1) OPERATOR(-) +
   VALUE(&DATE2) VALUE(&TIME2) /* incorrect */
```

```
SETD DAYS(DAYS) VALUE(&DATE) VALUE(&TIME) OPERATOR(-) +
   VALUE(&MINUTES) /* incorrect */
```

Figure 67. SETD Statement with Results in Days

The second SETD statement in Figure 68 on page 226 is incorrect because the precision of the results and second operand is minutes, but the precision of the first operand is days.

```
SETD EDATE(DATE) TIME(TIME) VALUE(&DATE) VALUE(&TIME) OPERATOR(-) +
   VALUE(&MINUTES) /* incorrect */
```

```
SETD EDATE(DATE) ITIME(TIME) VALUE(&DATE) VALUE(&TIME) OPERATOR(-) +
   VALUE(&MINUTES) /* incorrect */
```

```
SETD EDATE(DATE) VALUE(&DATE) VALUE(&TIME) OPERATOR(-) +
   VALUE(&MINUTES) /* incorrect */
```

Figure 68. SETD Statement with Results as a Date
The second SETD statement in Figure 69 is incorrect because the precision of the results and second operand is days, but the precision of the first operand is minutes.

The example in Figure 70 is correct but is an exception because the precision of the results and first operand is days; the precision of the second operand is years.

The SETD statement in Figure 71 subtracts a duration from a date/time pair and assigns the date component of the result to variable VAR1 (mm/dd part only) and the time component to variable VAR2 (hh part only).

The first SETD statement in Figure 74 on page 227 converts a number of minutes into extended duration format and assigns the result to variable VAR1. The second SETD
The SETD statement converts a duration into a number of minutes and assigns the result to variable VAR1. The third SETD statement converts a date into internal format and assigns the result to variable VAR1.

```
SETD DUR(VAR1) VALUE(1500)
SETD MIN(VAR1) VALUE(10:23:59)
SETD IDATE(VAR1) VALUE(07/25/1996)
```

Figure 74. SETD Statements

See Figure 17 on page 156 for another example using SETD.

**SPACE**

The SPACE (SP) command statement inserts blank lines in the report.

**Syntax of SPACE**

```
Space [ Lines (nnn) ] [ Execute { C|U} ] [ Minlines (nnn) ]
```

**Lines**

indicates the number of blank lines to be written.

**nnn**

is a whole number between -1 and 255. The default is 1.

If the output line already has data, that data is written first. For positive values, the blank lines (nnn) are written only if they do not exceed the remaining lines on the page. Any remaining lines after a page eject are ignored because blank lines are never carried forward to a new page.

When ANSI carriage control characters are used, LINES(0) inserts no space and allows for overtyping the previous line. This provides for underscoring of data or darkening of data on impact printers. Any data on the output line is written to the current line of the report data set. Then the data for the next line is written over it.

When ANSI carriage control characters are used, LINES(-1) inserts no space and no overtyping of the previous line occurs. This provides a means of forcing the output line to be written with no additional effect. Any data on the output line is written to the current line of the report data set. Then, the data for the next line is written to the next line.

```
SPACE LINES(-1)
```

Figure 75. SPACE Statement to Achieve Single Spacing

**Execute**

indicates whether or not the SPACE command is to be processed immediately after the titles or headings are processed.

**C** requests conditional processing; SPACE is ignored if titles or headings have just been processed. Or, if titles and headings are not defined, spacing at the top of the page is ignored. C is the default.
requests unconditional processing; SPACE is processed even when titles or headings have just been processed.

**Minlines**

determines whether the current page has enough lines to process the SPACE command.

**nnn**

is a decimal number between 1 and the current page length. If nnn is the same as or less than the number of lines on the page, processing of the SPACE command is based on the EXECUTE keyword.

If the number of lines remaining on the current page is less than nnn, SPACE is ignored. Then the next attempt to write data to the report triggers a page eject and titles and headings (if specified) are written on the next page, followed by the data.

**TAB**

The TAB (TA) command statement defines the start column positions for multiple data items when horizontal iteration has been specified on a CALL or PUT command. See "PUT" on page 194 for an explanation of horizontal iteration.

Until a TAB command is found during the output phase, horizontal iteration uses the separation amount to position multiple data items.

---

**Syntax of TAB**

Tab [ Column (p1 [p2 p3...p16])]

**Column**

specifies up to 16 column positions as a string of numbers, in ascending order, separated by at least one blank.

**p1 - p16**

is a series of integers greater than zero, with the maximum of these (the rightmost) less than the line length. These integers identify the column positions. The line length is defined in the usage notes for the CALL command 169.

You can define tabs more than once within an RFT. A TAB command setting remains in effect for the remainder of the report (all sections) or until you reset it. You can cancel tabs by specifying a TAB command without a COLUMN keyword or with a COLUMN keyword value of zero (TAB or TAB COLUMN(0)).

Figure 76 on page 229 illustrates how to use the TAB statement. The report that results from this RFT is similar to the one shown in Figure 77 on page 229.
TAB COL(1 20 40 60)
/* call the exit BLGOXPSA to print */
/* the search arguments horizontally */
/* using the tab columns */
CALL NAME(BLGOXPSA) COL(1) OP(H)
SPACE LINES(2) EX(U)
/* call the exit BLGOXCNT to count */
/* all hardware problems in 1996 */
/* and return the number in */
/* the variable &tothwprb */
/* do not print data returned */
/* by BLGOXCNT */
CALL NAME(BLGOXCNT) INPUT(!S0B01 DATE/1996/**/**,,tothwprb) Len(0)
/* print the report result */
/* in one line */
PUT VAL('Hardware problems during 1996') COL(1) LEN(50) PAD(.)
PUT VAL(&TOThWPRB.)

Figure 76. TAB Statement

INQUIRY RECS=PROBLEM STATUS DATA .IM00SD002 DATX/**/**/1996
Hardware problems during 1996......................86

Figure 77. Output from TAB Statement
Report Exit Routines

Report exit routines enhance the functionality of the Report Format Facility, because they can perform functions that are not possible with RFT statements alone. With report exit routines you can:

- Add data or output lines to reports
- Determine and flag exception data in reports
- Calculate outage
- Analyze data
- Perform calculations with values that contain special characters, such as periods and colons
- Retrieve or test multiple iterations of history data to check for a specific value or the same value.

All report exit routines in this book are written in assembler; however, you can use report exit routines written in other languages, such as PL/I.

Types of Report Exit Routines

The Report Format Facility uses two types of report exit routines:

**Data output exit**

Provides access to the output buffer and to the following services:

- Write (to write the data from the output buffer to the output data set)
- Read RFT variable
- Set RFT variable.

**CALL command exit**

Provides access to the output buffer and to the following services:

- Move (to move data to the output buffer)
- Read RFT variable
- Set RFT variable
- Data locate (to read the specified data from the current record).

You specify a data output exit (DOE) on the SECTION definition statement. This exit is given control just before the output line of data is written to the report data set for each line of output other than the titles or headings. One exit per section is allowed.
You specify a CALL command exit (CCE) on the CALL command statement. A CALL command routine is given control each time a CALL command statement is processed within a section. Six predefined CALL exit routines are supplied; they are described beginning on page 237.

Data output exits are usually unique to a report because they deal with data on the output line. You can use data output exits in another report only if the output lines for both reports are the same. CALL command exits are report independent because they deal with data that has not yet been placed on the output line.

Service Functions

Five service functions are available to the report exit routines: locate, move, read, set, and write. A CALL command exit can use all the services except write. The data output exits can use only read, set, and write services. See "Exit Routine Service Functions" on page 243 for more information.

The diagram in Figure 78 shows the relationship between an RFT, the report exit routines, and the service functions.

You can find examples of data output and CALL command exit routines in "Example 5: Identifying Problems That Exceed Outage Standards (Data Output Exit)" on page 307 and "Example 6: Identifying Problems with Three or More Assignees (CALL Command Exit)" on page 310.

Controlling Report Exit Routines

The Report Format Facility uses standard linkage to pass control to and from both types of exit routines:

- Register 1 contains the address of the 1-word parameter list that points to the user-exit communication control block.
- Register 13 contains the address of the register save area.
Register 14 contains a return address to the Report Format Facility.

Register 15 contains the entry point address to the exit routine.

The user-exit communication control block begins on a fullword boundary. It is a communications area between the Report Format Facility, the exit routine, and the report service functions. The contents of this control block vary with the type of exit and are included in the individual exit descriptions that follow.

Data Output Exit Routines

You specify the name of the load module for the data output exit routine on the NAME keyword of the SECTION definition statement. When you specify an exit routine, Tivoli Information Management for z/OS loads the module during the input phase, if it is not already in virtual storage. If the previous section used the same exit routine, it is already in storage and continues to be used. If the previous section used a different exit routine, it is deleted from storage when the input phase for the current section is complete.

The exit routine receives control just before the output line is written to the report data set for each line of output other than the titles or headings. The exit routine is never given control during title or heading processing.

The exit routine examines the data in the output line and changes the data or allows it to be written unchanged. Through the use of function codes and the service functions, the exit routine can also read and set variables. In addition, a service function can be used to write the output line to the report data set. This allows the exit routine to add as many additional lines of data to the report data set as desired.

Communication Control Block for the Data Output Exit

The format of the communication control block for data output exit routines is shown in Table 6. For offset and length, the decimal and hexadecimal (in parentheses) values are provided for the control block fields. The names are used for reference purposes only. Names beginning with CPF (common purpose field) are the same for both the data output exits and the CALL command exits. Names beginning with DOE (data output exit) are unique to the data output exit.

Table 6. Format of the Communication Control Block for Data Output Exit Routines

<table>
<thead>
<tr>
<th>Offset</th>
<th>Length</th>
<th>Name</th>
<th>Description</th>
<th>Labels</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 (0)</td>
<td>4 (4)</td>
<td>CPF1</td>
<td>Address of the service function router</td>
<td>FUNCRTN</td>
</tr>
<tr>
<td>4 (4)</td>
<td>4 (4)</td>
<td>DOE1</td>
<td>Address of the output buffer</td>
<td>OUTBUFF</td>
</tr>
<tr>
<td>8 (8)</td>
<td>4 (4)</td>
<td>DOE2</td>
<td>Length of the output buffer</td>
<td>OBUFLLEN</td>
</tr>
<tr>
<td>12 (C)</td>
<td>4 (4)</td>
<td>CPF2</td>
<td>Length of the data in the data buffer</td>
<td>DBUFLLEN</td>
</tr>
<tr>
<td>16 (10)</td>
<td>1 (1)</td>
<td>CPF3</td>
<td>Function code&lt;br&gt; 1101 1001 R - Read a variable&lt;br&gt; 1110 0010 S - Set a variable&lt;br&gt; 1110 0110 W - Write the output line</td>
<td>FUNCBYTE</td>
</tr>
<tr>
<td>17 (11)</td>
<td>1 (1)</td>
<td>CPF4</td>
<td>Call identifier code&lt;br&gt; 0000 0000 First call to routine&lt;br&gt; xxxx xxxx Request last call&lt;br&gt; 1111 1111 Last call to routine</td>
<td>FLAGBYTE</td>
</tr>
</tbody>
</table>
DOE1, the second word in the control block at offset 4, contains the address of the output line buffer. The first byte in this buffer contains the carriage control character. If you do not supply a valid carriage control character, the carriage control character is changed to a blank (print, single space), and the line is written to the output data set. If the output data set you specify does not contain valid carriage control characters, the first character is stripped off just before writing the data to the data set.

DOE2, the third word in the control block at offset 8, contains the length of the output line buffer (including the carriage control character). The output line buffer address and length fields are set in the control block each time your exit is called. They are provided for your information only; the Report Format Facility does not use them. Therefore, changing these fields in the control block has no effect on the output line.

The Report Format Facility sets the call identifier code (CPF4) under two conditions:
- It is set to X'00' when the exit routine is called for the first time within a section.
- If the exit routine changes the code to something other than X'00', the Report Format Facility sets the code to X'FF' and calls the exit routine an extra time just before deleting the exit routine from virtual storage. This extra call is made only if the exit routine sets the code to something other than X'00'.

**Note:** If you use the same exit routine for consecutive report sections, it is not reloaded if you link edited the exit with REUSE specified. If the exit is reusable, the code is not reset for each section.

The primary purpose of the call identifier code is to enable you to write reentrant programs. On the first call (or any call thereafter, because the code continues to be set to X'00' until you reset it) you can obtain storage and place its address in one of the words available to you in the userarea of the control block (DOE6). Next, you must change the code to some value other than X'00' so that your exit is called the extra time. On the last call (CPF4=X'FF'), you can free this storage.

The output line data identifier flags (DOE4) indicate the type of data that is being written to the output data set (the type of data in the output line buffer). When no flags are set (X'00'), normal report data is being written. An X'80' indicates a message is being written, and an X'40' indicates that a blank line (a space) is being written. Tivoli Information Management

---

<table>
<thead>
<tr>
<th>Offset</th>
<th>Length</th>
<th>Name</th>
<th>Description</th>
<th>Labels</th>
</tr>
</thead>
<tbody>
<tr>
<td>18 (12)</td>
<td>1 (1)</td>
<td>DOE3</td>
<td>Reserved</td>
<td>---</td>
</tr>
<tr>
<td>19 (13)</td>
<td>1 (1)</td>
<td>DOE4</td>
<td>Output line data identifier flags</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0000 0000 Data identifier</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1000 0000 Message identifier</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0100 0000 Space identifier</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>xx11 1111 Reserved</td>
<td></td>
</tr>
<tr>
<td>20 (14)</td>
<td>8 (8)</td>
<td>DOE5</td>
<td>Variable name (excluding the ampersand)</td>
<td>VARNAME</td>
</tr>
<tr>
<td>28 (1C)</td>
<td>20 (14)</td>
<td>DOE6</td>
<td>5 words available for your use</td>
<td>USERAREA</td>
</tr>
<tr>
<td>48 (30)</td>
<td>255 (FF)</td>
<td>CPF5</td>
<td>Data buffer</td>
<td>DATABUFF</td>
</tr>
<tr>
<td>303 (12F)</td>
<td>1 (1)</td>
<td>CPF6</td>
<td>Reserved</td>
<td>---</td>
</tr>
<tr>
<td>304 (130)</td>
<td>4 (4)</td>
<td>DOE7</td>
<td>Reserved</td>
<td>---</td>
</tr>
</tbody>
</table>
for z/OS sets these flags before each call to your exit. The identifier flags are for your information only; Tivoli Information Management for z/OS does not check them on return from your exit.

The data output exit provides five words (DOE6) in the communication control block that you can use. The remaining fields are discussed under the individual service function topics beginning on 243.

Return Codes for Data Output Exits

When the exit routine finishes processing, the return code indicates the action the Report Format Facility should take. The return code reflects any serious processing errors returned by the service functions.

<table>
<thead>
<tr>
<th>Return Code</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Write the output line to the report data set.</td>
</tr>
<tr>
<td>4</td>
<td>Delete the data on the output line and do not write it.</td>
</tr>
<tr>
<td>8</td>
<td>Processing for this section ends immediately.</td>
</tr>
<tr>
<td>12</td>
<td>Report processing ends immediately.</td>
</tr>
</tbody>
</table>

Any other return code causes report processing to end immediately.

Restrictions for Data Output Exits

If you use the call identifier code to request that your exit be called before deletion (CPF4=X'FF'), the following restrictions apply:
- The output line data identifier flags are meaningless because no data is being written.
- If you want to write data to the output data set, you must use the write service function.
- A nonzero return code causes processing to end immediately.
- A return code of 0 does not cause the output buffer to be written to the output data set.

CALL Command Exit Routines

Specify the name of the load module for the CALL command exit routine on the NAME keyword of the CALL command statement. When you specify an exit routine, Tivoli Information Management for z/OS loads the exit routine load module during the input phase if it is not already in storage. If a CALL command used the same exit routine in the previously processed section, that routine is already in storage and continues to be used. When the input phase is complete, all CALL command exit routines that you specified in the previous section, but which you did not specify in this section, are deleted from storage.

A CALL command exit receives control each time you process a CALL command during output-phase processing. These routines can:
- Locate data in the record currently being processed
- Read and set variables
- Write data to the report output data line.

Tivoli Information Management for z/OS provides six CALL command exits; you can write additional routines to provide functions that are not currently available.
Communication Control Block for CALL Command Exits

The format of the communication control block for CALL command exit routines is shown in Table 8. For offset and length, the decimal and hexadecimal (in parentheses) values are provided for the control block fields. The names are used for reference purposes only. Names beginning with CPF (common purpose field) are the same for both the data output exits and the CALL command exits. Names beginning with CCE (CALL command exit) are unique to the CALL command exit.

Table 8. Format of the Communication Control Block for CALL Command Exit Routines

<table>
<thead>
<tr>
<th>Offset</th>
<th>Length</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 (0)</td>
<td>4 (4)</td>
<td>CPF1</td>
<td>Address of the service function router</td>
</tr>
<tr>
<td>4 (4)</td>
<td>4 (4)</td>
<td>CCE1</td>
<td>Address of the input keyword data</td>
</tr>
<tr>
<td>8 (8)</td>
<td>4 (4)</td>
<td>CCE2</td>
<td>Length of the input keyword data</td>
</tr>
<tr>
<td>12 (C)</td>
<td>4 (4)</td>
<td>CPF2</td>
<td>Length of the data in the data buffer</td>
</tr>
<tr>
<td>16 (10)</td>
<td>1 (1)</td>
<td>CPF3</td>
<td>Function code</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1101 0011 L - Locate data in the current record</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1101 0100 M - Move data to the output line</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1101 1001 R - Read a variable</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1110 0010 S - Set a variable</td>
</tr>
<tr>
<td>17 (11)</td>
<td>1 (1)</td>
<td>CPF4</td>
<td>Call identifier code</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0000 0000 First call to routine</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>xxxx xxxx Request last call</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1111 1111 Last call to routine</td>
</tr>
<tr>
<td>18 (12)</td>
<td>1 (1)</td>
<td>CCE3</td>
<td>Locate control flags</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1xxx xxxx Reserved</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0100 xxxx Locate structured data</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0100 xxx1 Locate table list structured data</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0010 xxxx Locate journal-history data</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0001 xxxx Locate freeform-text data</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0xxx 1xx0 Locate all control data</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0xxx x10x Locate first occurrence</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0xxx x01x Locate next occurrence</td>
</tr>
<tr>
<td>19 (13)</td>
<td>1 (1)</td>
<td>CCE4</td>
<td>Reserved</td>
</tr>
<tr>
<td>20 (14)</td>
<td>16 (10)</td>
<td>CCE5</td>
<td>Variable name, p-word, or s-word</td>
</tr>
<tr>
<td>36 (24)</td>
<td>12 (C)</td>
<td>CCE6</td>
<td>Available for your use</td>
</tr>
<tr>
<td>48 (30)</td>
<td>255 (FF)</td>
<td>CPF5</td>
<td>Data buffer</td>
</tr>
<tr>
<td>303 (12F)</td>
<td>1 (1)</td>
<td>CPF6</td>
<td>Reserved</td>
</tr>
<tr>
<td>304 (130)</td>
<td>4 (4)</td>
<td>CCE7</td>
<td>Table list s-word length</td>
</tr>
</tbody>
</table>

The second word in the control block (CCE1) contains the address of the data contained in the input keyword, and the third word (CCE2) contains the length of this data. Because of variable substitution, this address and length can change each time your exit routine is given control. For this reason, any change you make to this data can be lost on the next call to your exit routine. When designing an exit routine, you are responsible for defining the syntax (beyond that defined by the Report Format Facility, such as commas defined as delimiters) and for use of this data.

The Report Format Facility sets the call identifier code (CPF4) under two conditions:
- It is set to X’00’ when the exit routine is called for the first time within a section.
If the exit routine changes the code to something other than X'00', the Report Format Facility sets the code to X'FF' and calls the exit routine an extra time just before deleting the exit routine from virtual storage. This extra call is made only if the exit routine sets the code to something other than X'00'.

Note: If you use the same exit routine for consecutive report sections, it is not reloaded if you link edited the exit with REUSE specified. If the exit is reusable, the code is not reset for each section.

The call identifier code enables you to write reentrant programs. On the first call (or any call thereafter, because the code continues to be set to X'00' until you reset it) you can obtain storage and place its address in one of the words available to you in the control block (CCE6). Next, you must change the code to some value other than X'00' so that your exit is called the extra time. On the last call (CPF4=X'FF'), you can free this storage.

The CALL command exit provides three words (CCE6) in the control block that you can use. The remaining fields are discussed under the individual service function topics beginning on page 243.

Return Codes for CALL Command Exits

The return code reflects any serious processing errors returned by the service functions. Any return code other than 0 causes processing to end immediately unless you specify TEST(Y) on the SECTION statement.

CALL Command Exit Restrictions

If you use the call identifier code to request that your exit be called before deletion (CPF4=X'FF'), the following restrictions apply:

1. The input keyword data address and length fields are set to 0.
2. The locate and move service functions are not valid; they return a code of 12 if requested.
3. Returning a nonzero return code causes processing to end immediately, regardless of the setting of the TEST keyword on the present SECTION statement.

Predefined CALL Command Exits

Tivoli Information Management for z/OS supplies six exit routines for use with the CALL command statement:

- BLGOXCNT
  Searches and counts
- BLGOXPSA
  Formats search argument
- BLGOXFSD
  Formats detail data
- BLGOXFSR
  Formats response chains
- BLMOXDCK
  Checks “from” and “to” dates
- BLGOXLST
  Formats list processor data
**Search and Count**

The BLGOXCNT routine searches the database for records that meet a specified search argument and counts the records found. The routine accepts four positional parameters separated by commas as the data for the INPUT keyword. If you omit the first, second, or third parameter, the preceding commas are required. All the parameters are optional.

**Search argument**

follows the same syntax as the search argument on the SEARCH command. If the search argument ends with a dictionary index key and you want to specify one or more of the additional parameters, you must insert at least one blank between the dictionary index key and the comma, or you will get an input-phase error message. If you do not specify a search argument, the routine counts all records returned by the search level specified on the MERGE keyword.

**Merge indicator**

has the same meaning as the MERGE keyword on the SEARCH command when the value \( n \) is specified. The values \( N \) and \( P \) are not valid values for the SEARCH command on a BLGOXCNT call statement. The merge indicator must be less than or equal to the current search level. If you do not specify a merge indicator, the resultant list is not merged, and all of the returned records are counted. If you use a variable for this parameter and you want to specify one or more additional parameters, you must insert either a period or a space between the variable and the comma. If you omit the period or the space before the comma, you get an error message.

**Variable name**

identifies the variable that is to be set with the number of matches found. You can specify it with or without the ampersand. However, if you specify it with the ampersand, remember that you must enclose the ampersand within quotes to prevent its value from being substituted into the input keyword data. If you do not specify a variable name, no variable is set.

Because values enclosed in quotes are not converted to uppercase, the variable name is not valid if you use lowercase characters.

**Database number**

identifies the number (0-9) of the database to search. If merging is to take place, the database that is the target of the merge must be the same database that you used for this CALL command statement. If you use a number that is different from what was on the CALL command statement, you will receive an error message. If you do not specify a database number, the Report Format Facility uses the default database.

The number of matches found is printed on the output line according to the remaining keywords on the CALL statement. The number of matches is always printed unless the length specified by the LENGTH keyword is 0. By setting the length to 0, a variable is set with the number of matches, but no data is printed by the CALL statement.

The CALL command statement illustrated in Figure 79:

- Searches for all records that contain prefix PERA/ and merges the resultant list with the level 0 list (results from REPORT command)
- Starting at column 10, writes the total
- Sets the variable, &ASSIGNED, to the total
- Searches database 5, if level 0 list was from database 5.
Format Search Argument

The BLGOXPSA routine formats and writes the level 0 search argument to the output line. If you do not specify a search argument with the REPORT command, a message stating this fact is printed. The search argument is one of the following:

- The search argument specified by the user before issuing the REPORT command. You can specify the search argument with the panels, the ARGUMENT command, the SEARCH command, or any combination of these three.
- The search argument specified on the REPORT command, if the arguments were not preceded by a plus sign (+).
- A combination of these if the arguments on the REPORT command were preceded by a plus sign (+).

In addition, any of these search arguments would include keywords specified with the ARGUMENT command from a report panel.

The routine requires no input to be passed by the INPUT keyword. The search argument is printed on the output line according to the remaining keywords on the CALL command statement. Because this routine processes multiple data items, specify horizontal iteration on the OPERATOR keyword.

If you did not specify a search argument, an informational message to that effect is written instead. If a PRINT record command calls BLGOXPSA, an informational message is issued and no attempt is made to format the search argument.

The CALL command statement illustrated in Figure 80 writes the entire search argument horizontally starting at column 3. Each data item is separated by two spaces.

CALL NAME(BLGOXPSA) COLUMN(3) SEPARATION(2) OPERATOR(H)

Figure 80. CALL Command Using Format Search Argument Predefined Exit

Format Detailed Data

The BLGOXFSD routine formats all detail data for a record. This includes the following data:

- All nonadministrative structured data. Specifically, data that has the Replace Previous Reply field set to NO or is table list data.
- Phrases.
- Descriptions such as the symptom and resolution structured data for problem records.
- Detail structured data for change records.

Refer to the Tivoli Information Management for z/OS Panel Modification Facility Guide for more information on the meaning of the Replace previous reply field.
The routine requires no input to be passed by the INPUT keyword. The data is written on the output page according to the remaining keywords on the CALL statement. Because this routine processes multiple data items for a record, specify horizontal iteration on the OPERATOR keyword.

Assuming that the CALL command statement illustrated in Figure 81 applies to a problem record, it:

- Retrieves the phrases (this includes TYPE=PROBLEM), prefixes, and data for problem symptoms and resolution, and other data items that are collected from assisted-entry panels that specify NO in the Replace previous reply field.
- Starting at column 5, writes the detailed data horizontally, leaving three spaces between data items.

CALL NAME(BLGOXFSD) COLUMN(5) SEPARATION(3) OPERATOR(H)

Figure 81. CALL Command Using Format Detailed Data Predefined Exit

Format Response Chains

The BLGOXFSR routine formats all responses for an SRC record. This includes selections, data, diverts, commands, freeform responses, and responses added by program exit BLG01146.

The routine requires no input to be passed by the INPUT keyword. The data is written on the output page according to the remaining keywords on the CALL statement. Because this routine processes multiple data items for a record, specify horizontal iteration on the OPERATOR keyword.

The CALL command statement illustrated in Figure 82 writes the SRC responses horizontally starting at column 5.

CALL NAME(BLGOXFSR) COLUMN(5) SEPARATION(2) OPERATOR(H)

Figure 82. CALL Command Using Format Response Chains Predefined Exit

Check “From” and “To” Dates

The BLMOXDCK routine, which is used by the Configuration Subdiagram RFT, determines whether the configuration date you specify on panel BLGOW250 falls within the date ranges specified in a connection record.

The routine first checks the &ZVAR1 variable to determine if you specified a configuration date. If the &ZVAR1 variable is null, BLMOXDCK sets the &ZVAR1T variable to 1 and exits. Otherwise, the routine uses the installation-defined or Tivoli Information Management for z/OS default date conversion routine to convert the date to internal format.

BLMOXDCK then uses the DATX/ and DATJ/ prefixes to find the “from” and “to” dates in the current record and uses the installation-defined or Tivoli Information Management for z/OS default date conversion routine to convert these dates to internal format. If no “from”
date is found, an internal format date of 0000/01/01 is used. If no “to” date is found, an
internal format date of 9999/12/31 is used. If the configuration date you specified falls
between the “from” and “to” dates, the &ZVARIT variable is set to 1. Otherwise, it is set to
0.

Format List Processor Data

The BLGOXLST routine enables you to create a table of list processor data with each
column containing data for the specified list processor field. These columns are in ascending
order by index so that each row corresponds to the nth entries for the specified fields.
Because blank rows are compressed, the row of nth entries can appear before the nth
physical row of the tabular output.

The routine accepts a number of positional parameters as INPUT keyword data which is
separated by commas. The number of parameters is limited only by the maximum number of
continuations for a statement. These parameters are specified in groups, and each group
corresponds to a list processor field. The groups are specified in the order in which they
appear in the output (from left to right).

The group of parameters for a given list processor field consists of the following. All but the
first parameter are optional. The INPUT keyword is required; however, no blanks are
allowed in the INPUT keyword data.

s-word index
An !S followed by 4 hexadecimal characters, which identifies a root s-word of 1 to
8 characters in length. This s-word corresponds to the list processor field that is
included in the output. This parameter cannot be null.

Column
Accepts the same values and has the same meaning as the COLUMN keyword on
the PUT statement; however, the values C and S are not allowed.

Length
Accepts the same values and has the same meaning as the LENGTH keyword on the
PUT statement.

Wfrom
Accepts the same values and has the same meaning as the WFROM keyword on the
PUT statement.

Wfor
Accepts the same values and has the same meaning as the WFOR keyword on the
PUT statement.

From
Accepts the same values and has the same meaning as the FROM keyword on the
PUT statement.

For
Accepts the same values and has the same meaning as the FOR keyword on the
PUT statement.

Justification
Accepts the same values and has the same meaning as the JUSTIFY keyword on the
PUT statement. Must be null unless the length parameter is specified for the current
field.

Pad
Accepts the same values and has the same meaning as the PAD keyword on the
PUT statement. Must be null unless the length parameter is specified for the current
field.
**Precision**

Accepts the same values and has the same meaning as the PRECISION keyword on the PUT statement. Must be null unless the length parameter is specified for the current field.

When data is found for a given field and index (column and row), the BLGOXLST routine simulates a PUT command statement with settings determined by the parameters associated with the current field. Consequently, the same rules that apply to successive PUT statements apply to successive (left to right, top to bottom) field outputs. Also, the SPACE command statement with a keyword specification of LINES(-1) is simulated before and after each row of output.

The following keywords on the CALL statement are ignored when NAME(BLGOXLST) is specified:

- COLUMN
- LENGTH
- OPERATOR
- PAD
- JUSTIFY
- SEPARATION

**SECTION**

**HEADER**

- PUT COLUMN(1) VALUE('RECORD ID') LENGTH(15)
- PUT COLUMN(21) VALUE('DEVICE NAME') LENGTH(15)
- PUT COLUMN(41) VALUE('DEVICE TYPE') LENGTH(15)
- PUT COLUMN(61) VALUE('SERIAL NUMBER') LENGTH(15)
- SPACE LINES(1)

**SEARCH**

- PUT COLUMN(1) DATA(RNID/.) LENGTH(15)
- CALL NAME(BLGOXLST) INPUT(!S1416,21,15,,,,,,,,!S1418,41,15,,,,,,,,!S141C,61,15)

**ESearch**

**ESection**

*Figure 83. CALL Command Using Format List Processor Data Predefined Exit*

<table>
<thead>
<tr>
<th>RECORD ID</th>
<th>DEVICE NAME</th>
<th>DEVICE TYPE</th>
<th>SERIAL NUMBER</th>
</tr>
</thead>
<tbody>
<tr>
<td>RECORD1</td>
<td>DEVICE1</td>
<td>1111111</td>
<td>123456</td>
</tr>
<tr>
<td></td>
<td>DEVICE2</td>
<td>2222222</td>
<td>123457</td>
</tr>
<tr>
<td></td>
<td>DEVICE3</td>
<td>3333333</td>
<td>123458</td>
</tr>
<tr>
<td></td>
<td>DEVICE4</td>
<td>4444444</td>
<td>123459</td>
</tr>
</tbody>
</table>

*Figure 84. Output from CALL Command Using List Processor Data Predefined Exit*

The CALL statement illustrated in *Figure 83* places the list processor data for s-word indexes 1416, 1418, and 141C in columns 21, 41, and 61 respectively. All columns have a width of 15. The output is similar to the output shown in *Figure 84*.

Two RFTs shipped with Tivoli Information Management for z/OS format list processor data. They are BLGPRAL and BLMZZ14.
Exit Routine Service Functions

The diagram on page 232 illustrates the five service functions available to the exit routines. To call these service functions, the exit routine must first specify the appropriate function code in the function code field (CPF3):

L (X'D3')
Located a specific p-word or s-word in the record currently being processed and return any associated data as well as the length of that data. This function is available only to the CALL command exit.

M (X'D4')
Move the specified data to the output data buffer. The data is positioned in the output line according to the keywords specified on the CALL command statement. This function is available only to the CALL command exit.

R (X'D9')
Find the specified variable name and return the associated data as well as the length of that data. This function is available to both the CALL command exit and the data output exit.

S (X'E2')
Set the specified data into the specified variable. This function is available to both the CALL command exit and the data output exit.

W (X'E6')
Write the data in the output line to the output data set. This function is available only to the data-output exit.

After the function code is set in the function code field of the control block, the exit routine, using standard register linkage conventions, must Branch And Link Register (BALR 14,15) to the service function router whose address is contained in the first word (CPF1) of the control block. The exit routine must pass, as an input parameter to the service function router, the same parameter list that it received on input.

The address of the communication control block is passed as the only parameter in a parameter list to the service function router. The address of the parameter list is passed using the standard parameter register, which is Register 1.

The service function router then examines the function code and, if it is valid, calls the corresponding service function routine.

The following sections describe each service function in more detail.

**Locate Service Function—X'D3'**

The locate service function is available only to the CALL command exit routine. It locates the first or next occurrence of a specified p-word or s-word in the record currently being processed by the search level containing the CALL statement.

To understand the locate service function, you must understand the makeup of a Tivoli Information Management for z/OS record. Tivoli Information Management for z/OS records consist of three separate parts:

- **Structured data** The data that you enter on the data-entry and assisted-entry panels. In addition, certain data is entered by selection panels and control panels.
- **Journal-history data** The data that you enter on assisted-entry panels that have yes in their journal reply fields.

- **Freeform-text data** The data that you enter on the freeform-text table panels.

The structured data and journal-history data consist of the individual data items chained together in the order that you entered the data. Likewise, within a given type of text (such as description, status, and address), the text data is also chained together in the order in which it appeared when you entered it. The locate service function scans down these chains looking for a match with the requested item (p-word or s-word). If it finds a match, it returns the data associated with the request.

The locate service function is not case sensitive; that is, it locates matching data regardless of the case used to store the data.

It is also important to remember that there is only one locate service function within the Report Format Facility. All commands share the locate service function as they require. The ramifications of this are discussed on page 245.

Before calling the service function router, the exit routine must set the fields listed in Table 9 in the communication control block:

**Table 9. Fields the Exit Routine Must Set in the Communication Control Block**

<table>
<thead>
<tr>
<th>Field</th>
<th>Length</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPF3</td>
<td>1 byte</td>
<td>Set to “L” (X’D3’) to indicate a locate service function.</td>
</tr>
<tr>
<td>CCE3</td>
<td>1 byte</td>
<td>Set the appropriate locate control flags to indicate the data type, occurrence and amount of data to be returned. (See text below.)</td>
</tr>
<tr>
<td>CCE5</td>
<td>16 bytes</td>
<td>Move the p-word or s-word to be located into this field. For the p-word, a period must follow the slash (/) or underscore (_). If you specify an s-word for this field, the first position (the ‘watermark’ position) must be X’BA’ through X’BF’. For example, enter the s-word UDESCRIPTA as X’BF’DESCRIPTA. The field is then right-padded with blanks.</td>
</tr>
<tr>
<td>CCE7</td>
<td>4 bytes</td>
<td>If the request is to locate table list structured data (CCE3=B’0100xxx1), set this field to the length of the table list s-word (root length + 2).</td>
</tr>
</tbody>
</table>

The locate control flags (CCE3) specify the type of data to be located and which occurrence of the p-word or s-word to find. The locate control flags in the first half of the byte specify the type of data. You must set a single data type flag for each locate service function request:

**0100 xxx0**

Locate structured data. The CCE5 field must contain the p-word or s-word associated with the data you want returned.

**0100 xxx1**

Locate table list structured data. The CCE5 field must contain the table list s-word associated with the data you want returned. Also, the CCE7 field must contain the length of the table list s-word specified in CCE5.
0010 xxx0
Locate journal history data. The CCE5 field must contain the p-word or s-word associated with the history data you want returned.

0001 xxx0
Locate freeform text data. The CCE5 field must contain the s-word associated with the freeform text data you want returned. A specific word within the text cannot be located; only a text type (such as description, status, or address) can be located. The freeform text is returned one line at a time. Blank lines are returned as a single blank.

The locate control flags in the second half of the byte specify which occurrence of the data to return and how much data to return.

You must set a single-occurrence indicator flag for each locate service function request:

0xxx x10x
Locate the first occurrence (entry) of the data item. This flag tells the locate service function to start searching for your request at the top of the data chain. Therefore, use of this flag each time you change the CCE5 field ensures that you always locate the data if it is in the record.

0xxx x01x
Locate the next occurrence of the data item. This flag tells the locate service function to start searching for your request at its current position in the chain. This has the following ramifications:

- If this is the first call to the locate service function for this invocation of the CALL exit routine, the locate service is positioned within the data chain wherever the last request (from a CALL, IF, PUT, or SET command at this search level) for the type of data you are attempting to locate left it, and you may not find the data you are trying to locate.

Refer to the following sample record shown. If the locate service function is currently positioned at STAC/OPEN and you attempt to locate PERA/, the data is returned to you. However, if you attempt to locate PERS/, the data is not found, even though it exists in the record.

PERS/ENDUSER1
STAC/OPEN
GROS/D123
PH/X1234
PERA/STEVENS

- If your last request to the locate service function for data of this type was for a different p-word or s-word, and that data was found, you may or may not find the data you are trying to locate. If the data was not found, the locate service is already at the bottom of that data chain.

- If this is the first request for data of the type you are requesting that has been made by any command since the record was last read, the locate service function starts again at the top of the data chain and tries to locate the first occurrence of the data.
If your last request to the locate service function for data was for the same p-word or s-word, and that data was found, this request will return the next occurrence of that data in the record, if there is one. This is the intended use of the next flag.

0100 xxx1
Locate table list entry. This flag tells the locate service function to disable special character processing, such as * [position ignore] and . [truncation], for the index portion (last 2 bytes) of the table list s-word. Also, field CCE7 specifies the length of the table list s-word (root length + 2).

0xxx 1xx0
Locate all data associated with a history or text data item.

The setting of this flag is optional for history and text data. For structured data, the flag is ignored.

When this flag is on for history or text data, the data item and any journal control data you specify in HFORMAT or TFORMAT definition statements, such as date, time, or user ID, are returned. If you do not include the control data in the format definition, only the actual history or text data item is returned.

For each locate service function request, you must specify one data type and one occurrence flag. When you specify history or text as the type, you must combine the second half of the byte to indicate the amount of data and which occurrence you want to obtain. For example, X'2C' specifies all data associated with the first occurrence of the history data item.

The return codes set by the locate service function are listed in Table 10.

Table 10. Return Codes Set by the Locate Service Function

<table>
<thead>
<tr>
<th>Return Code</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>P-Word or s-word was found.</td>
</tr>
<tr>
<td>4</td>
<td>P-Word or s-word was not found.</td>
</tr>
</tbody>
</table>
| 8           | Field CCE3 or CCE5 is not valid:  
               CCE3 - One or more reserved flags are set.  
               CCE3 - No or multiple data type flags are set.  
               CCE3 - No or multiple occurrence flags are set.  
               CCE5 - Field is blank or 0, or contains mixed data that is not valid. |
| 12          | Function code is not valid. |
| 16          | Processing error. This return code should be returned to the Report Format Facility, because it indicates a serious processing failure and report processing should be stopped. |

If the specified p-word or s-word is found, any data associated with it is returned in the data buffer (CPF5), and the length of the data is placed into the data buffer length field (CPF2). The data buffer is not padded with blanks, so it is important that you use the length returned in the data buffer length field when using the data in the data buffer.

If the s-word is found but no data is associated with it, or if the specified p-word or s-word is not found, the data buffer length field is set to 0. Therefore, you must check the return code to tell the difference.
Move Service Function—X’D4’

The move service function, which is available only to the CALL command exit, moves data to the output line. The MINLINES keyword positions data on the page. The data is positioned within the output line by the COLUMN, LENGTH, JUSTIFY, PAD, OPERATOR, and SEPARATION keywords on the CALL command statement.

If multiple data items are to be written to the output line, it is generally preferred to design the CALL command exit routine to call the move service function several times rather than string the data together and call the move service function only once. This adds flexibility to the exit routine, because you can control the amount and placement of the data items through the use of the various keywords on the CALL command statement that calls the exit routine.

Table 11 lists the fields in the communication control block that the exit routine must set before calling the service function router.

Table 11. Fields the Exit Routine Must Set in the Communication Control Block

<table>
<thead>
<tr>
<th>Field</th>
<th>Length</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPF2</td>
<td>4 bytes</td>
<td>Set this field to the length of the data to be written to the output line.</td>
</tr>
<tr>
<td>CPF3</td>
<td>1 byte</td>
<td>Set to M (X’D4’) to indicate a move service function request.</td>
</tr>
<tr>
<td>CPF5</td>
<td>255 bytes</td>
<td>Move the data to be written to the output line into this field.</td>
</tr>
</tbody>
</table>

The move service function sets the return codes listed in Table 12.

Table 12. Return Codes Set by the Move Service Function

<table>
<thead>
<tr>
<th>Return Code</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Data is written to the output line.</td>
</tr>
<tr>
<td>4</td>
<td>Data is not written because this is the second or subsequent CALL to the move function, and iteration is not specified on the CALL statement.</td>
</tr>
<tr>
<td>8</td>
<td>Data is not written because the LENGTH keyword value is 0, field CPF2 is 0 or negative, or the data in CPF5 contains mixed data that is not valid.</td>
</tr>
<tr>
<td>12</td>
<td>Function code is not valid.</td>
</tr>
<tr>
<td>16</td>
<td>Processing error. This return code should be returned to the Report Format Facility, because it indicates a serious processing failure and report processing should be stopped.</td>
</tr>
</tbody>
</table>

Variable Read Service Function—X’D9’

The variable read service function, which is available to both the CALL command and data output exits, reads a particular variable and returns the data associated with it.

Before calling the service function router, the exit routine must set the fields in the communication control block listed in Table 13 on page 248.
Table 13. Fields the Exit Routine Must Set in the Communication Control Block

<table>
<thead>
<tr>
<th>Field</th>
<th>Length</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPF3</td>
<td>1 byte</td>
<td>Set to R (X’D9’) to indicate a variable read service function request.</td>
</tr>
</tbody>
</table>
| CCE5 or DOE5 | 8 bytes | Move the name of the variable to be read into this field.  
For a CALL command exit, the variable name must be right-padded with at least one blank. For a data output exit, variable names less than 8 characters in length must be right-padded with at least one blank. |

The return codes set by the variable read service function are listed in Table 14.

Table 14. Return Codes Set by the Variable Read Service Function

<table>
<thead>
<tr>
<th>Return Code</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Variable was read successfully.</td>
</tr>
<tr>
<td>4</td>
<td>Variable name is not defined.</td>
</tr>
<tr>
<td>8</td>
<td>Variable name is not SBCS alphanumeric.</td>
</tr>
<tr>
<td>12</td>
<td>Function code is not valid.</td>
</tr>
<tr>
<td>16</td>
<td>Processing error. This return code should be returned to the Report Format Facility, because it indicates a serious processing failure, and report processing should be stopped.</td>
</tr>
</tbody>
</table>

If the specified variable is found, the data associated with it is returned in the data buffer (CPF5), and the length of the data is placed in the data buffer length field (CPF2). If the variable is null (for example, the variable name is defined but no data is associated with it), the data buffer length field is set to 0.

The data buffer is not padded with blanks, so it is important that you use the length returned in the data buffer length field when using the data in the data buffer.

**Variable Set Service Function—X’E2’**

The variable set service function, which is available to both the call command and data output exits, sets data into an existing variable or creates a new variable.

Before calling the service function router, the exit routine must set the fields in the communication control block listed in Table 13.
Table 15. Fields the Exit Routine Must Set in the Communication Control Block

<table>
<thead>
<tr>
<th>Field</th>
<th>Length</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPF2</td>
<td>4 bytes</td>
<td>Move the length of data you want to set into this field.</td>
</tr>
<tr>
<td>CPF3</td>
<td>1 byte</td>
<td>Set to S (X'E2') to indicate a variable set service function request.</td>
</tr>
<tr>
<td>CCE5 or DOE5</td>
<td>8 bytes</td>
<td>Move the name of the variable to be set into this field. For a CALL command exit, the variable name must be right-padded with at least one blank. For a data output exit, variable names less than 8 characters in length must be right-padded with at least one blank.</td>
</tr>
<tr>
<td>CPF5</td>
<td>255 bytes</td>
<td>Move the value to which the variable is to be set into this field. Right-pad the data to the length specified in the length field (CPF2).</td>
</tr>
</tbody>
</table>

The return codes set by the variable set service function are listed in Table 16.

Table 16. Return Codes Set by the Variable Set Service Function

<table>
<thead>
<tr>
<th>Return Code</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Variable was set successfully.</td>
</tr>
<tr>
<td>4</td>
<td>Variable is read-only.</td>
</tr>
<tr>
<td>8</td>
<td>The data length (CPF2) is not 0 to 255, or the variable name (CCE5) exceeds 8 characters in length or is not SBCS alphanumeric.</td>
</tr>
<tr>
<td>12</td>
<td>Function code is not valid, or CPF5 contains mixed data that is not valid.</td>
</tr>
<tr>
<td>16</td>
<td>Processing error. This return code should be returned to the Report Format Facility, because it indicates a serious processing failure, and report processing should be stopped.</td>
</tr>
</tbody>
</table>

Write Service Function—X'E6'

The write service function, which is available only to the data output exit, writes data directly to the report data set.

Before calling the service function router, the exit routine must set the fields in the communication control block listed in Table 17.

Table 17. Fields the Exit Routine Must Set in the Communication Control Block

<table>
<thead>
<tr>
<th>Field</th>
<th>Length</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DOE1</td>
<td>(defined in DOE2)</td>
<td>Move the data to be written to the output line buffer pointed to by this field.</td>
</tr>
<tr>
<td>CPF3</td>
<td>1 byte</td>
<td>Set to W (X'E6') to indicate a write service function request.</td>
</tr>
</tbody>
</table>

The length of the output line buffer (including the carriage control character) is provided in the DOE2 field of the communication control block.
In its calculations to determine when to write titles and headings, the Report Format Facility automatically uses the carriage control character in the first position of the output line. Thus, the exit can place any valid American National Standards Institute (ANSI) carriage control character in the first position of the output line to control forms spacing. However, if you specify anything other than an ANSI carriage control character, it is replaced with an SBCS blank (writes to the next line). The string is adjusted if this could make any DBCS data not valid. If an SO character is specified as a carriage control character, truncation occurs.

If a skip-to-one is specified, or if the Report Format Facility determines that a page eject is required before the output line can be written, the specified carriage control character is replaced with a blank, the page is ejected, the titles and headers are processed, and the output line is written.

If you specify an output data set that does not contain carriage control characters, the characters are stripped off just before writing the data to the report data set.

The return codes set by the write service function are listed in Table 18.

**Table 18. Return Codes Set by the Write Service Function**

<table>
<thead>
<tr>
<th>Return Code</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Data was successfully written to the output data set.</td>
</tr>
<tr>
<td>8</td>
<td>Data was written to the output data set, but the specified ANSI carriage control character was changed to a blank because it was not valid. If the data is mixed and SO was specified as a carriage control character, the data was truncated.</td>
</tr>
<tr>
<td>12</td>
<td>Function code is not valid.</td>
</tr>
<tr>
<td>16</td>
<td>Processing error. This return code should be returned to the Report Format Facility, because it indicates a serious processing failure and report processing should be stopped.</td>
</tr>
</tbody>
</table>

Table 18. Return Codes Set by the Write Service Function
Defining Output Destinations for Reports

When you issue the REPORT or PRINT command, Tivoli Information Management for z/OS must determine the output destination. Each time you issue a REPORT command, Tivoli Information Management for z/OS allocates a data set for the output. The first time you issue a PRINT command in a Tivoli Information Management for z/OS session, Tivoli Information Management for z/OS also allocates a data set for the output. You can keep this data set open or free it. You can use the FREE PRINT command or a setting in your user profile to free this data set. Refer to the Tivoli Information Management for z/OS User's Guide for more information about the FREE command. If you keep the print data set open, the next time you issue the PRINT command, Tivoli Information Management for z/OS appends to the same data set.

Tivoli Information Management for z/OS uses the values defined in your user profile to determine the output destinations and the attributes (such as block size and logical record length) of the output data set. If you do not have an output destination defined in your user profile, Tivoli Information Management for z/OS prompts you for the information. This chapter describes how to define a destination for your REPORT or PRINT output.

SYSOUT, DSNAME, and DDNAME

If the Standard Report field in the Session Defaults panel contains SYSOUT, Tivoli Information Management for z/OS uses the data in the Standard Report SYSOUT Destination Entry panel to obtain the necessary SYSOUT values for running the report. If the Standard Report field contains DSNAME, Tivoli Information Management for z/OS uses the data on the Standard Report Data Set Destination Entry panel to obtain the information for allocating the report data set. If the Standard Report field contains DDNAME, Tivoli Information Management for z/OS obtains the necessary destination information from the Standard Report DDNAME Destination Entry Panel.

When you use the PRINT command, the current output destination (specified in the profile) is selected and remains selected until you free the PRINT data set either by a user profile option, the FREE command, changing the output destination, or ending the Tivoli Information Management for z/OS session. If the current output destination remains selected, any additional PRINT command output is appended to the current output destination. If your profile specifies to keep the print data set open, none of this output is written until the destination is closed by changing the output destination in the profile, ending the Tivoli Information Management for z/OS session, or issuing the FREE PRINT command.

You can dynamically allocate a different data set for printed output, or you can modify the attributes of the default data set identified in your profile. To do so, you must update your profile before issuing the PRINT or REPORT command.
If you issue the REPORT command, and the **Standard Report** field on the Session Defaults panel is blank, Tivoli Information Management for z/OS prompts you for the destination before continuing report processing.

If you select **SYSOUT** on the Standard Report Output Destination panel and you have not previously entered all the required attributes, Tivoli Information Management for z/OS prompts you with the following panel:

Panel BLG0P510 contains the predefined default values for SYSOUT. Refer to the *OS/390 MVS JCL Reference* for information on values for the **Output descriptor name** field.
Selecting DSNAME on the Standard Report Output Destination panel causes the Standard Report Data Set Destination Entry panel to be displayed.

This panel contains the predefined default values for DSNAME under MVS/ESA™. When you enter a data set name in the Data set name field, you must follow standard TSO naming conventions and fully qualify the data set name.

The data set disposition can be NEW, OLD, or MOD (Modify). If you specify MOD and the data set cannot be found, the data set is allocated as NEW. If you do not specify a disposition, the data set is allocated as OLD if it exists, or NEW if it cannot be found. If you are dynamically allocating an existing data set that is not cataloged, you must include the unit name and serial number of the volume on which it resides.

Tivoli Information Management for z/OS supplies a set of predefined default values for many of the attribute fields on the Standard Report Data Set Destination Entry panel. These predefined defaults are listed with each field description in the [Tivoli Information Management for z/OS User's Guide](https://www.ibm.com). After you update the data set information on the data-entry panel, report processing continues.

Selecting DDNAME on the Standard Report Output Destination panel causes a data-entry panel to appear that displays the attributes of the default preallocated data set.

You can modify the DDNAME attributes on this panel.
When you use a preallocated file name (that is, you select DDNAME), the TSO ALLOCATE command must have previously allocated the data set in the TSO environment.

In batch mode, you must allocate this data set with the BLGRPTS DD statement if you specify DDNAME as the standard report destination. See "Running Reports in Batch Mode" on page 255 for more information on batch reports.

Notes:

1. You cannot use the same data set for both print output and report output at the same time. You can use the same data set for both print output and report output provided the data set is closed before issuing a Report command that follows a PRINT command. The print data set is closed by changing the output destination in the user profile, ending the Tivoli Information Management for z/OS session, or issuing the FREE PRINT command.

2. Dynamic allocation does not allow for standard labeled tapes. However, you can use nonlabeled tapes for the report output data set.
This chapter describes how to run reports in batch mode using stored response chains (SRCs).

Running Reports with SRCs
An SRC is a predefined set of panel responses that are appropriate to a particular report. Each time you run an SRC, the same panel responses are repeated automatically. Because an SRC overrides the user’s panel defaults, it always works in the same way, regardless of who runs the SRC. To run a report using an SRC, you prefix the REPORT command with a double semi-colon (;;REPORT). For details on how to create an SRC refer to the Tivoli Information Management for z/OS Program Administration Guide and Reference.

Running Reports in Batch Mode
Batch mode processing eliminates the need for you to be present to start an interactive Tivoli Information Management for z/OS session, run the report, and then log off.

Batch Reports under MVS/ESA
To run a batch report under an MVS/ESA system, run a background job, using standard TSO conventions. The report runs under ISPF batch processing, and ISPF calls Tivoli Information Management for z/OS in the same manner as in an interactive session. As part of the MVS/ESA job control language (JCL), a batch report requires:

- The user’s TSO user ID as the prefix keyword for the TSO PROFILE command if RACF® or an equivalent facility is not installed.
- An SRC or immediate response chain (IRC) specified in the PARM keyword of the ISPSTART command, unless the user’s profile contains a default SRC name.

The SRC or IRC can contain a SEARCH command (with or without a search argument). However, it must include the REPORT command. The response chain must include all data required by the prompting sequence to run the report successfully. The response chain must end with a QUIT.

Tivoli Information Management for z/OS sends the report output to the report destination specified in your profile for the type of report being generated (standard or user RFT, print, or customized).

If an error occurs during batch processing, Tivoli Information Management for z/OS writes the message panel image and any existing message chain (displayed, in an interactive session) to the data set allocated by the SYSPRINT DD statement. If there is no SYSPRINT...
DD statement, Tivoli Information Management for z/OS writes the messages to a dynamically allocated SYSOUT class A data set. In either case, after closing and freeing the print data set, batch processing ends.

**Note:** If you specify the disposition of the report output data set as NEW, you must supply the DCB information by including the DCB keyword subparameters on the report output DD statement. If you copy the DCB information from a catalogued data set or an earlier DD statement, Tivoli Information Management for z/OS uses the default attributes from your user profile instead of the copied attributes. This can cause PUT errors.

You can run reports in batch mode using the JCL given in [Figure 85 on page 257](#), substituting your own values for the lowercase entries.

The JOB statement should match your installation-defined standards. A JOBCAT may be necessary if the data sets are not in the master catalog. A STEPLIB or an ISPLLIB concatenation may be necessary if the Tivoli Information Management for z/OS load modules are not in the system or link pack area.

Most of these JCL statements follow standard MVS/ESA or TSO conventions; however, the following have special meanings:

- **userid** A TSO logon ID. The user ID specified with the PREFIX keyword must be identified in a Tivoli Information Management for z/OS privilege class.

- **blgrpts** The DDNAME of the report output data set (userid.batch.report) specified in your profile. This is required only if allocation via DDNAME is specified in the profile.

- **responses** Prompting sequence responses. Your last response should be the QUIT command (in the form ;quit).

- **name** The name of an SRC that initiates processing on the Primary Options Menu.
When Tivoli Information Management for z/OS writes to SYSPRINT, it formats the data using DCB information you specified on either a SYSPRINT DD statement (that is, LRECL or BLKSIZE) or a TSO ALLOCATE. If you specify an LRECL without a BLKSIZE, Tivoli Information Management for z/OS sets the BLKSIZE to:

\[(14 \times \text{LRECL}) + 4\]

If you do not specify a BLKSIZE or an LRECL, the LRECL is set to:

\[\text{length of output message} + 4\]

The BLKSIZE is set to:

\[(14 \times \text{LRECL}) + 4\]

If you specify a BLKSIZE without an LRECL, the LRECL is set to the smaller of the following statements:

\[\text{length of output message} + 4\]
\[\text{BLKSIZE} - 4\]

In all cases, the LRECL must be less than or equal to BLKSIZE - 4. If it is not, an ABEND occurs when the data set is opened because the data attributes are inconsistent.
IV — Appendixes
Tivoli supplies a number of standard reports with Tivoli Information Management for z/OS. See Table 19 for a list of the standard reports.

### Table 19. Standard Reports

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>General Reports</td>
<td>Present information from records in one of the following report formats:</td>
<td>These reports are described on page &quot;General Reports&quot; on page 263.</td>
</tr>
<tr>
<td></td>
<td>- Search Results List Report</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Line Summary Report</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Page Summary Report</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Detail Report</td>
<td></td>
</tr>
<tr>
<td>Problem Management Reports</td>
<td>Present information from problem management records in one of the following report formats:</td>
<td>Refer to the Tivoli Information Management for z/OS Problem, Change, and Configuration Management for more information.</td>
</tr>
<tr>
<td></td>
<td>- Periodic Status Report</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Calendar Report</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Assignee Report</td>
<td></td>
</tr>
<tr>
<td>Change Management Reports</td>
<td>Present information from change management records in one of the following report formats:</td>
<td>Refer to the Tivoli Information Management for z/OS Problem, Change, and Configuration Management for more information.</td>
</tr>
<tr>
<td></td>
<td>- Periodic Status Report</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Calendar Report</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Approver Summary Report</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Approver Detail Report</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Change Activities Report</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Schedule Report</td>
<td></td>
</tr>
<tr>
<td>Configuration Management Reports</td>
<td>Present information from configuration management records in one of the following report formats:</td>
<td>Refer to the Tivoli Information Management for z/OS Problem, Change, and Configuration Management for more information.</td>
</tr>
<tr>
<td></td>
<td>- Inventory by Location Report</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Component with Related Features Report</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Hardware Configuration Map Report</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Software Configuration Map Report</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Components with Specified Features Report</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Configuration Diagram Report</td>
<td></td>
</tr>
</tbody>
</table>
Table 19. Standard Reports (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
<th>Remarks</th>
</tr>
</thead>
</table>
| Tivoli Inventory Reports | Presents information from an extract of Tivoli Inventory data, as stored in Tivoli Information Management for z/OS, in the following report formats:  
  - Inventory by Tivoli Managed Region (TMR) Summary  
  - Search Results List  
  - Inventory Line Summary  
  - Inventory Page Summary  
  - Inventory Detail Report | Refer to the Tivoli Information Management for z/OS Guide to Integrating with Tivoli Applications for more information.  
  Note: These reports are accessed by selecting the Reports option on the SERVICDESK application's Inventory Menu panel. |

Format for Standard Reports

Although each standard report has unique characteristics, all reports share the following similarities:

- Each page contains 60 lines of 132 characters each. The first line on each page contains the current date on the left, the report title in the center, and the page number on the right. The time the report was generated appears on the second line, directly below the date, and the RFT member name appears on the right directly below the word PAGE.

- The first page contains the search argument, followed by the number of matches found. If the search finds no matches, a message stating that no records met the selection criteria for the report appears on the first page of the report.

- The record data begins on the second page of each report. The appropriate data for each record is displayed according to the report type. Records are sorted on one or more fields, and are presented in alphanumeric or chronological order, depending on the type of data involved. In some reports, the record data is sorted into several sections.

- When you run a standard report, print record report, or draw report, you can specify whether you want the output in uppercase or not. If the **Output in uppercase** field on panel BLG0P520, the Standard Report Data Set Destination Entry panel, is set to 'NO', the data portion of the report displays in the case used to store the data in the SDDS. That is, if the data is stored in all uppercase characters in the SDDS, it will appear on reports in uppercase also. If the data is stored in mixed case format, it will appear on reports in mixed case format. Setting the **Output in uppercase** field to 'YES' causes both the report data and report titles or headings to appear in uppercase. For an illustration of panel BLG0P520, refer to the Tivoli Information Management for z/OS User’s Guide. A description of how the case of data is defined for storage in the SDDS is provided in the Tivoli Information Management for z/OS Panel Modification Facility Guide.

An example of page 1 of the Search Results List report is shown in Figure 86. (Column positions are adjusted to allow the report to fit on this page.)
General Reports

General Reports, supported by Tivoli Information Management for z/OS, are described below.

Search Results List Report (BLMZZ11)

For each record matching the search argument, the Search Results List report shows a single line of data that consists of the record identifier and a description abstract. In the report, the total number of records listed appears in the report header. If you do not have display authority for a record that meets the search criteria, UNAUTHORIZED appears in the description column. Figure 86 is an example of a page of a Search Results List report.

Line Summary Report (BLMZZ12)

The Line Summary report shows a single line of data for each record matching the search argument. The type of data displayed depends on the record type.

Records that meet the search criteria are grouped by record type in the following order:
1. Privilege Class
2. Stored Response Chain (SRC)
3. Call
4. Problem
5. Change
6. Activity
7. Hardware Component
8. Hardware Connection
9. Hardware Feature
10. Hardware Subcomponent
11. Model Hardware Component
12. Model Hardware Feature
13. Model Hardware Subcomponent
14. Software Component
15. Software Connection
16. Software Feature
17. Model Software Component
18. Model Software Feature
19. System
20. Data Center

2. The oldest Tivoli Information Management for z/OS privilege classes, if any, are presented first, followed by those for the remaining versions.
Record type groups are included in the report according to the current privilege class of the user.

**Page Summary Report (BLMZZ13)**

The Page Summary report provides selected data for each record. All records that meet the search criteria are grouped by type and are presented in the same order as in the Line Summary report. For most record types, each report is a full page of data. The record types included in this report are the same as those for the Line Summary report (listed previously).

**Detail Report (BLMZZ14)**

The Detail report presents all the data from every record found in the search. The records that meet the search criteria are grouped by type and presented in the same order and manner as in the Page Summary report. The record types included are the same as those described for the Line Summary report. In addition, the detail report shows the detail data (if present), all freeform text data, and all changes to journalized data.

The detail data is specific to the record type.

- For SRC records, the Detail report also shows the SRC response sequence, any SRC freeform responses, and any text collected in the SRC.
- For problem records, the Detail report includes the symptom and fix data.
- For change requests and activities, the Detail report includes the detailed technical description of the change.
- For hardware components, features, and subcomponents, the Detail report includes engineering change (EC) levels; and for software components and features, the Detail report includes function modification identifications (FMIDs).

The freeform text data in the Detail report includes all the textual comments, such as description, status, or resolution text, entered when the user created or updated the record. The journalized data shows the initial values and all changes that have been made to the journal fields. The fields are presented in alphabetical order, and show each change date and the changed data.
Table 20 lists the member names of the RFTs for the standard reports shipped as part of Tivoli Information Management for z/OS. Samples of standard reports are provided in "Report Samples” on page 267.

Table 20. Member Names of RFTs for Standard Reports

<table>
<thead>
<tr>
<th>Standard Report</th>
<th>Provides</th>
<th>RFT Member Name</th>
<th>Control Panel Name</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>GENERAL</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Search Results List</td>
<td>List of search results</td>
<td>BLGZZ11</td>
<td>BLG0W117</td>
</tr>
<tr>
<td>Line Summary</td>
<td>Line summary of any record</td>
<td>BLMZZ12</td>
<td>BLG0W118</td>
</tr>
<tr>
<td>Page Summary</td>
<td>Page summary of any record</td>
<td>BLMZZ13</td>
<td>BLG0W119</td>
</tr>
<tr>
<td>Detail</td>
<td>All the data for any record, including list processor</td>
<td>BLMZZ14</td>
<td>BLG0W120</td>
</tr>
<tr>
<td><strong>PROBLEM MANAGEMENT</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Periodic Status</td>
<td>Opened, closed, and priority 1 hold-over problems for period</td>
<td>BLMZZ21</td>
<td>BLG0W104</td>
</tr>
<tr>
<td>Calendar</td>
<td>All dates related to problems</td>
<td>BLMZZ22</td>
<td>BLG0W105</td>
</tr>
<tr>
<td>Assignee</td>
<td>Problems assigned</td>
<td>BLMZZ23</td>
<td>BLG0W106</td>
</tr>
<tr>
<td><strong>CHANGE MANAGEMENT</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Periodic Status</td>
<td>Required changes for 3 periods</td>
<td>BLMZZ31</td>
<td>BLG0W107</td>
</tr>
<tr>
<td>Calendar</td>
<td>Key change and activity dates</td>
<td>BLMZZ32</td>
<td>BLG0W108</td>
</tr>
<tr>
<td>Approver Summary</td>
<td>Summary of changes by approver</td>
<td>BLMZZ33</td>
<td>BLG0W109</td>
</tr>
<tr>
<td>Approver Detail</td>
<td>Detail data of above changes</td>
<td>BLMZZ34</td>
<td>BLG0W110</td>
</tr>
<tr>
<td>Changes w/Activities</td>
<td>Changes with related activities</td>
<td>BLMZZ35</td>
<td>BLG0W111</td>
</tr>
<tr>
<td>Change and Activity</td>
<td>Planned changes and activities</td>
<td>BLMZZ36</td>
<td>BLG0W112</td>
</tr>
<tr>
<td>Schedule</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>CONFIGURATION MANAGEMENT</strong></td>
<td>Components grouped by location</td>
<td>BLMZZ41</td>
<td>BLG0W113</td>
</tr>
<tr>
<td>Location</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Component with Features</td>
<td>Components with related features</td>
<td>BLMZZ42</td>
<td>BLG0W114</td>
</tr>
<tr>
<td>Hardware Map</td>
<td>Hierarchy of hardware components</td>
<td>BLMZZ43</td>
<td>BLG0W115</td>
</tr>
<tr>
<td>Software Map</td>
<td>Hierarchy of software components</td>
<td>BLMZZ44</td>
<td>BLG0W116</td>
</tr>
<tr>
<td>Components with Specified Features</td>
<td>Components with specified features</td>
<td>BLMZZ45</td>
<td>BLG0W300</td>
</tr>
<tr>
<td>Diagram Data Extract</td>
<td>Data set for input to DRAW command</td>
<td>BLMZZ46</td>
<td>BLG0W301</td>
</tr>
<tr>
<td><strong>TIVOLI INVENTORY</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Inventory by TMR Summary</td>
<td>Hardware by Tivoli managed region</td>
<td>BLHZZ41</td>
<td>BLH1AZ41</td>
</tr>
<tr>
<td>Search Results List</td>
<td>Hardware listed by record ID</td>
<td>BLHZZ11</td>
<td>BLH1AZ11</td>
</tr>
<tr>
<td>Inventory Line Summary</td>
<td>Summary of hardware by record ID with booted operating system</td>
<td>BLHZZ12</td>
<td>BLH1AZ12</td>
</tr>
<tr>
<td>Inventory Page Summary</td>
<td>Detail data of hardware by record ID</td>
<td>BLHZZ13</td>
<td>BLH1AZ13</td>
</tr>
<tr>
<td>Inventory Detail Report</td>
<td>Detail data of hardware including software installed</td>
<td>BLHZZ14</td>
<td>BLH1AZ14</td>
</tr>
</tbody>
</table>
PRINT Record Output

Output from the PRINT (record) command is also formatted using RFTs. Table 21 shows the RFT names used by the Report Format Facility to format the individual records for printing and the control panel that collects the RFT name for each record type. The RFT name identifies a member of the RFT partitioned data set in MVS/ESA. You need to know the member name if you want to tailor the print output for a record.

Table 21. RFT Member Names and Control Panel Names Used to Print Records

<table>
<thead>
<tr>
<th>Record Type</th>
<th>RFT Member Name</th>
<th>Control Panel Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stored Response Chain (SRC)</td>
<td>BLGPRGR</td>
<td>BLG1A148</td>
</tr>
<tr>
<td>Privilege Class</td>
<td>BLMPRPC</td>
<td>BLG1A101</td>
</tr>
<tr>
<td>Call</td>
<td>BLMPRCA</td>
<td>BLG1A1C8</td>
</tr>
<tr>
<td>Problem</td>
<td>BLMPRPR</td>
<td>BLG1A118</td>
</tr>
<tr>
<td>Change</td>
<td>BLMPRCR</td>
<td>BLG1A128</td>
</tr>
<tr>
<td>Activity</td>
<td>BLMPRAR</td>
<td>BLG1A138</td>
</tr>
<tr>
<td>Hardware Component</td>
<td>BLMPRCH</td>
<td>BLG1A158</td>
</tr>
<tr>
<td>Hardware Feature</td>
<td>BLMPRHF</td>
<td>BLG1A168</td>
</tr>
<tr>
<td>Hardware Connection</td>
<td>BLMPRHC</td>
<td>BLG1A166</td>
</tr>
<tr>
<td>Hardware Subcomponent</td>
<td>BLMPRHS</td>
<td>BLG1A156</td>
</tr>
<tr>
<td>Hardware Financial</td>
<td>BLMPRFH</td>
<td>BLG1A198</td>
</tr>
<tr>
<td>Model Hardware Component</td>
<td>BLMPMHC</td>
<td>BLG1A298</td>
</tr>
<tr>
<td>Model Hardware Feature</td>
<td>BLMPMHF</td>
<td>BLG1A2A8</td>
</tr>
<tr>
<td>Model Hardware Subcomponent</td>
<td>BLMPMHS</td>
<td>BLG1A218</td>
</tr>
<tr>
<td>Software Component</td>
<td>BLMPRCS</td>
<td>BLG1A178</td>
</tr>
<tr>
<td>Software Feature</td>
<td>BLMPRSF</td>
<td>BLG1A188</td>
</tr>
<tr>
<td>Software Connection</td>
<td>BLMPRSC</td>
<td>BLG1A242</td>
</tr>
<tr>
<td>Software Financial</td>
<td>BLMPRFS</td>
<td>BLG1A1A8</td>
</tr>
<tr>
<td>Model Software Component</td>
<td>BLMPMSC</td>
<td>BLG1A278</td>
</tr>
<tr>
<td>Model Software Feature</td>
<td>BLMPMSF</td>
<td>BLG1A288</td>
</tr>
<tr>
<td>Data Center</td>
<td>BLMPRCD</td>
<td>BLG1A1C8</td>
</tr>
<tr>
<td>System</td>
<td>BLMPRSY</td>
<td>BLG1A1D8</td>
</tr>
<tr>
<td>Service</td>
<td>BLMPRSR</td>
<td>BLG1A1B8</td>
</tr>
<tr>
<td>Base Panel Set</td>
<td>BLMPRSPS</td>
<td>BLG1Y118</td>
</tr>
<tr>
<td>Parent Panel Set</td>
<td>BLMPRPPS</td>
<td>BLG1Y618</td>
</tr>
<tr>
<td>Child Panel Set</td>
<td>BLMPRCPS</td>
<td>BLG1Y658</td>
</tr>
<tr>
<td>Rules</td>
<td>BLMPRRU</td>
<td>BLG1A418</td>
</tr>
<tr>
<td>Users</td>
<td>BLGPRUS</td>
<td>BLG1A918</td>
</tr>
<tr>
<td>Alias (includes list processor data)</td>
<td>BLGPRAL</td>
<td>BLG1A1G8</td>
</tr>
<tr>
<td>Logsave</td>
<td>BLGPRBK</td>
<td>BLG1A515</td>
</tr>
<tr>
<td>Command</td>
<td>BLGPRCM</td>
<td>BLG1A1G8</td>
</tr>
<tr>
<td>Data Attribute</td>
<td>BLMPRDA</td>
<td>BLG1A708</td>
</tr>
<tr>
<td>Data View</td>
<td>BLMPRDV</td>
<td>BLG1A608</td>
</tr>
</tbody>
</table>
Table 21. RFT Member Names and Control Panel Names Used to Print Records (continued)

<table>
<thead>
<tr>
<th>Record Type</th>
<th>RFT Member Name</th>
<th>Control Panel Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Validation</td>
<td>BLMPRVR</td>
<td>BLG1A508</td>
</tr>
<tr>
<td>People</td>
<td>BLMPRPE</td>
<td>BLM1B060</td>
</tr>
<tr>
<td>AutoBridge Map</td>
<td>EYMPRMR</td>
<td>EYM1M928</td>
</tr>
<tr>
<td>Tivoli Inventory</td>
<td>BLHPRINV</td>
<td>BLH1A110</td>
</tr>
<tr>
<td>Solution</td>
<td>BLHPRSO</td>
<td>BLH1S008</td>
</tr>
<tr>
<td>Index</td>
<td>BLHPRNX</td>
<td>BLH1N008</td>
</tr>
</tbody>
</table>

Report Samples

This section provides samples of standard reports available with Tivoli Information Management for z/OS. Fictitious data is provided to show you what the reports would look like if you did no special customization. Since some reports can consist of many pages, only a subset of representative pages are shown here. Standard Integration Facility reports are not included in this section. The spacing in the reports may have been modified to fit the information on the page. Dates shown on the reports are in 10-character external date format. If you run standard reports using an 8-character external date format, your date fields will display with 2 extra blank spaces (e.g., 03/30/98__ where __ represents 2 blank spaces).
General Reports

DATE 03/30/1998 *** INFORMATION MANAGEMENT for z/OS SEARCH RESULTS LIST REPORT *** PAGE 1
TIME 11:16:06 ---------------------------------------------------------- BLMZZ11

RECORD ID | DESCRIPTION OF THE '000020' RECORDS THAT MEET THE SEARCH CRITERIA
-----------|---------------------------------------------------
00000005   | CANNOT PRINT IN LOTUS
00000007   | CANNOT CONNECT TO HOST
00000011   | PASSWORD RESET
00000014   | LOAD NOISE FROM DRIVE
00000020   | SCATTERED DATA ON SCREEN
00000031   | HOST CONNECTION FAILED
00000032   | TAPE REWIND PROBLEM
00000037   | USER BEING TIMED OUT
00000048   | ABEND DC2 IN TSO EDIT
00000075   | TERMINAL WILL NOT ACCEPT INPUT
00000135   | CONNECTION TO LAN FAILED
00000301   | UNSCHEDULED IPL - MVS PRONTO
00000373   | CONNECTION TO LAN FAILED
00000612   | LAN IS DOWN
00000613   | CONTROLLER NOT WORKING
00000700   | BATCH JOB FAILING
00000724   | ABEND 008 IN TSO
00000999   | TIMEOUT MESSAGES ON SOME LINES
00001256   | GRAPHICS NOT PRINTING PROPERLY
00001382   | TIMEOUT MESSAGES ON SOME LINES

DATE 03/30/1998 *** INFORMATION MANAGEMENT for z/OS LINE SUMMARY REPORT *** PAGE 1
TIME 12:16:06 ----------------------------------------------------- BLMZZ12

THE FOLLOWING SEARCH ARGUMENT PRODUCED '000108' MATCHES IN THE INFORMATION MANAGEMENT for z/OS DATABASE (DATABASE 5)

SUMMARY OF PRIVILEGE CLASS RECORDS
----------------------------------
AUTHORITIES: A-ASSIGN, C-CLOSE, D-DISPLAY, E-ENTRY, U-UPDATE, X-DELETE
G-PANEL UPDATE, H-DICTIONARY DISPLAY, I-DICTIONARY UPDATE,
J-PANEL COPY, K-PANEL DELETE, L-PANEL LIST, R-PMF REPORTS,
DB-DATABASE ADMINISTRATION, UA-UNIVERSAL PARTITION ACCESS
TB-TSD BRIDGE CLEANUP

<table>
<thead>
<tr>
<th>CLASS</th>
<th>FIRST</th>
<th>CLASS</th>
<th>SRC</th>
<th>PROBLEM</th>
<th>CHANGE</th>
<th>CONFIG</th>
<th>FINANCE</th>
<th>RULES</th>
<th>PEOPLE</th>
<th>SOLUTION</th>
<th>PMF</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAME/ID</td>
<td>USERID</td>
<td>CONTACT NAME</td>
<td>DB</td>
<td>UA</td>
<td>TB</td>
<td>EUXD</td>
<td>EUX</td>
<td>EUDAC</td>
<td>EUX</td>
<td>EUD</td>
<td>EUD</td>
</tr>
<tr>
<td>--------</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>CHNGCONT</td>
<td>SMITH</td>
<td>JIMSMITH</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HELPDESK</td>
<td>JONES</td>
<td>JONES</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INFOUSR</td>
<td>INFOUSR</td>
<td>GREGG</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SYSAADMIN</td>
<td>GRANT</td>
<td>GRANT</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### SUMMARY OF PROBLEM RECORDS

<table>
<thead>
<tr>
<th>PROBLEM NUMB/ID</th>
<th>PROBLEM TYPE</th>
<th>DATE ENTERED</th>
<th>KEY ITEM</th>
<th>REPORTED BY</th>
<th>*** CURRENT *** FIX CHNG</th>
<th>NUMB/ID</th>
<th>ASSIGNEE NAME</th>
<th>DATE ASSIGNED</th>
<th>USER FORM NUM</th>
</tr>
</thead>
<tbody>
<tr>
<td>00000079</td>
<td>SOFTWARE</td>
<td>03/01/1998</td>
<td>PRINTER1</td>
<td>SMITH</td>
<td>CLOS 03 STUDY 00000080</td>
<td>JONESR</td>
<td>03/04/1998</td>
<td>F988</td>
<td></td>
</tr>
<tr>
<td>00000112</td>
<td>DOCUMENT</td>
<td>03/16/1998</td>
<td>DISK47</td>
<td>BLACKM</td>
<td>CLOS 02 TEST 00000080</td>
<td>SMITHM</td>
<td>03/16/1998</td>
<td>F999</td>
<td></td>
</tr>
</tbody>
</table>

### SUMMARY OF CHANGE RECORDS

<table>
<thead>
<tr>
<th>CHANGE NUMB/ID</th>
<th>CHANGE TYPE</th>
<th>DATE REQUIRED</th>
<th>KEY ITEM</th>
<th>REQUESTED BY</th>
<th>*** CURRENT *** PLANNED END DATE</th>
<th>ASSIGNEE NAME</th>
<th>DATE ASSIGNED</th>
<th>APPROVAL STATUS</th>
</tr>
</thead>
</table>

### SUMMARY OF ACTIVITY RECORDS

<table>
<thead>
<tr>
<th>ACTIVITY NUMB/ID</th>
<th>ACTIVITY TYPE</th>
<th>DATE REQUIRED</th>
<th>ACTIVITY NAME</th>
<th>REQUESTED BY</th>
<th>*** CURRENT *** PLANNED END DATE</th>
<th>ASSIGNEE NAME</th>
<th>DATE ASSIGNED</th>
<th>CHANGE NUMB/ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>00000093</td>
<td>HARDWARE</td>
<td>03/14/1998</td>
<td>PCORDER</td>
<td>DAVIES</td>
<td>OPEN 2 STUDY 04/10/1998</td>
<td>MCCARTHR</td>
<td>03/03/1998</td>
<td>00000085</td>
</tr>
<tr>
<td>00000094</td>
<td>PERFORM</td>
<td>03/10/1998</td>
<td>PRINTDOC</td>
<td>SMITHM</td>
<td>INIT 10 HOLDING</td>
<td>BLACKM</td>
<td>03/03/1998</td>
<td>00000086</td>
</tr>
</tbody>
</table>

### SUMMARY OF HARDWARE COMPONENT RECORDS

<table>
<thead>
<tr>
<th>COMPONENT NAME/ID</th>
<th>GENERIC DEVICE</th>
<th>DEVICE TYPE AND MODEL</th>
<th>SERIAL NUMBER</th>
<th>LOCATION CODE</th>
<th>MODEL LINK ID</th>
<th>DATA CTR REC ID</th>
<th>SYSTEM REC ID</th>
<th>FINANCIAL REC ID</th>
<th>**** COMPONENT ****</th>
<th>OWNER</th>
<th>STATUS</th>
<th>DATE OF STATUS</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU3</td>
<td>CPU</td>
<td>CP3000</td>
<td>CP375356</td>
<td>ANNAPOI</td>
<td>DESKTOP</td>
<td>JUDWS10</td>
<td>PCS</td>
<td>BLACKM</td>
<td>ORDER</td>
<td>04/11/1997</td>
<td></td>
<td></td>
</tr>
<tr>
<td>JDLTRK5</td>
<td>CTL</td>
<td>CTLR3745</td>
<td>C222234</td>
<td>ANNAPOI</td>
<td>DESKTOP</td>
<td>JUDWS10</td>
<td>PCS</td>
<td>SMITHM</td>
<td>PLANNED</td>
<td>04/11/1997</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LANONE</td>
<td>LOP</td>
<td>PS/2M545</td>
<td>PS123458</td>
<td>BALTIMOR</td>
<td>DESKTOP</td>
<td>JUDWS10</td>
<td>PCS</td>
<td>MCCARTHR</td>
<td>DOWN</td>
<td>04/11/1997</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MYS1</td>
<td>CPU</td>
<td>CP4000</td>
<td>CP345454</td>
<td>ANNAPOI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>INSTALL</td>
<td>04/11/1997</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### SUMMARY OF PEOPLE RECORDS

<table>
<thead>
<tr>
<th>RECORD ID</th>
<th>PERSON ID</th>
<th>PERSON ID</th>
<th>ROLE</th>
<th>COMPANY NAME</th>
<th>CITY/STATE/PROV</th>
<th>DATE ENTERED</th>
<th>USER</th>
<th>DATE ALTERED</th>
</tr>
</thead>
<tbody>
<tr>
<td>CN12345</td>
<td>Richard Wagner</td>
<td>CUSTOMER</td>
<td>Supermarkets Plus</td>
<td>East Brunswick, NJ</td>
<td>05/26/1999</td>
<td>SCHINDL</td>
<td>05/26/1999</td>
<td></td>
</tr>
<tr>
<td>HDT00001</td>
<td>Yacone TSD390</td>
<td>Services Plus</td>
<td>Raleigh, NC</td>
<td>05/27/1999</td>
<td>SCHINDL</td>
<td>05/27/1999</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### SUMMARY OF SOLUTION RECORDS

<table>
<thead>
<tr>
<th>SOLUTION NUMB/ID</th>
<th>PROBLEM NUMB/ID</th>
<th>SYSTEM</th>
<th>DEVICE</th>
<th>KEY IT</th>
<th>PROGRAM</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>00000007</td>
<td>LAN</td>
<td>ABCNTFS1 server problem</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00000030 00000025</td>
<td>INTRANET WEBSITE</td>
<td>PRODINFO</td>
<td>Link on PRODINFO Web site not working</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00000029 00000028</td>
<td>MVS1 REPORTS 00000044</td>
<td>BILL01</td>
<td>User cannot access CMVC</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The following search argument produced '000108' matches in the Information Management for z/OS Database (Database 5).

<table>
<thead>
<tr>
<th>PRIVILEGE CLASS DATA</th>
<th>PRIVILEGE CLASS USERIDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>USER ID 1 -----------</td>
<td>INFOUSR ID 1 - INFOUSR ID 13 -</td>
</tr>
<tr>
<td>TRANSFER-TO CLASS ---</td>
<td>MASTER ID 2 - ID 14 -</td>
</tr>
<tr>
<td>CONTACT NAME -------</td>
<td>SMITH ID 3 - ID 15 -</td>
</tr>
<tr>
<td>CONTACT DEPARTMENT --</td>
<td>AGFJ ID 4 - ID 16 -</td>
</tr>
<tr>
<td>CONTACT PHONE -------</td>
<td>302-5948 ID 5 - ID 17 -</td>
</tr>
<tr>
<td>LOCATION CODE ------</td>
<td>BLDG4 ID 6 - ID 18 -</td>
</tr>
<tr>
<td>OWNING PRIV. CLASS --</td>
<td>BUCHMAN ID 7 - ID 19 -</td>
</tr>
<tr>
<td>ENTRY PRIV. CLASS ----</td>
<td>MASTER ID 8 - ID 20 -</td>
</tr>
<tr>
<td>DATE ENTERED --------</td>
<td>04/05/1997 ID 9 - ID 21 -</td>
</tr>
<tr>
<td>TIME ENTERED ---------</td>
<td>14:04 ID 10 - ID 22 -</td>
</tr>
<tr>
<td>DATE LAST ALTERED ---</td>
<td>03/31/1998 ID 11 - ID 23 -</td>
</tr>
<tr>
<td>TIME LAST ALTERED ---</td>
<td>13:57 ID 12 - ID 24 -</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CLASS AUTHORITY</th>
<th>SRC AUTHORITY</th>
<th>PROBLEM AUTHORITY</th>
<th>CHANGE AUTHORITY</th>
<th>CONFIGURATION AUTHORITY</th>
<th>FINANCIAL AUTHORITY</th>
<th>PMF AUTHORITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>DISPLAY -</td>
<td>ENTRY --</td>
<td>DISPLAY - YES</td>
<td>DISPLAY - YES</td>
<td>DISPLAY - YES</td>
<td>DISPLAY - YES</td>
<td>PANEL UPD --</td>
</tr>
<tr>
<td>ENTRY ---</td>
<td>UPDATE -</td>
<td>UPDATE -- YES</td>
<td>DELETE -- YES</td>
<td>DELETE -- YES</td>
<td>DELETE -- YES</td>
<td>PANEL COPY --</td>
</tr>
<tr>
<td>UPDATE --</td>
<td>DELETE</td>
<td>DELETE -- YES</td>
<td>CLOSE -- YES</td>
<td>CLOSE -- YES</td>
<td></td>
<td>PANEL DEL --</td>
</tr>
<tr>
<td>DELETE --</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>PANEL LIST --</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RULES AUTHORITY</th>
<th>DATABASE ADMINISTRATION AUTHORITY</th>
<th>ACCESS AUTHORITY</th>
<th>PEOPLE AUTHORITY</th>
<th>SOLUTION AUTHORITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>DISPLAY -</td>
<td>DADMIN ---</td>
<td>DISPLAY -</td>
<td>DISPLAY -</td>
<td></td>
</tr>
<tr>
<td>ENTRY ---</td>
<td>TSD CLEANUP</td>
<td>ENTRY ---</td>
<td>ENTRY ---</td>
<td></td>
</tr>
<tr>
<td>UPDATE --</td>
<td></td>
<td>UPDATE --</td>
<td>UPDATE --</td>
<td></td>
</tr>
<tr>
<td>DELETE --</td>
<td></td>
<td>DELETE --</td>
<td>DELETE --</td>
<td></td>
</tr>
</tbody>
</table>
**PROBLEM REPORTER DATA**

- **REPORTED BY**: SMITH
- **REPORTER DEPT**: ABD1
- **REPORTER PHONE**: 301-345-3457
- **DATE OCCURRED**: 02/02/1998
- **NETWORK NAME**: BACKBONE
- **SYSTEM NAME**: VALUE2
- **PROGRAM NAME**: SORT12
- **DEVICE NAME**: PRINTER1
- **KEY ITEM AFFECTED**: PRINTER1
- **DATE FIX REQUIRED**: 03/03/1998
- **TIME FIX REQUIRED**: 08:41
- **PROBLEM TYPE**: SOFTWARE
- **PROBLEM STATUS**: CLOSED
- **INITIAL PRIORITY**: 03
- **OUTAGE TYPE**: UN REPAIR
- **ERROR CODE**: 123456
- **CLUSTER NAME**: REMOTE
- **CIRCUIT NUMBER**: 45F
- **ENTRY PRIV. CLASS**: BUCHMAN
- **DATE ENTERED**: 03/01/1998
- **TIME ENTERED**: 08:47
- **INTERESTED PRIVILEGE CLASSES**
  - CONSOLE OUTPUT
  - OUTPUT DATA
  - INPUT DATA
  - SOURCE LISTING
  - SOURCe
  - SMP LISTING
  - GRAPH/LOG DATA
  - DIAGNOSTIC OUTPUT
  - OPERATOR FORM
  - PROBABLE CAUSE
  - DUMP DATA SET
  - RESOURCE TYPES

**PROBLEM STATUS DATA**

- **ASSIGNEE NAME**: JONESR
- **ASSIGNEE DEPT**: BY6A
- **ASSIGNEE PHONE**: 689-1234
- **TARGET DATE**: 03/01/1998
- **RESPONSE/TRAVEL**: 02:00:00
- **DATE FIX REQUiRED**: 03/03/1998
- **TIME FIX REQUIRED**: 08:41
- **PROBLEM TYPE**: SOFTWARE
- **CURRENT PHASE**: STUDY
- **CURRENT PRIORITY**: 03
- **SOURCE LISTING**: BLDG4
- **SOURCE**: BLDG4
- **SOURCE LISTING**: BLDG4
- **TRACE DATA**: BLDG4
- **GRAPH/LOG DATA**: BLDG456
- **DIAGNOSTIC OUTPUT**: BLDG4
- **OPERATOR FORM**: BLDG34
- **PROBABLE CAUSE**: HARDWARE
- **DUMP DATA SET**: SMITH.REPORT
- **RESOURCE TYPES**: RESOURCE NAMES

**PROBLEM CLOSE DATA**

- **RESOLVED BY**: JSmith
- **RESOLVER DEPARTMENT**: DB45
- **RESOLVER PHONE**: 678-8954
- **DATE CLOSED**: 02/26/1998
- **TIME CLOSED**: 13:48
- **TOTAL TIME**: 10:00:00
- **DUPLICATE COUNT**: 000
- **OUTAGE**: 01:30:30
- **PROBLEM STATUS**: CLOSED
- **CAUSE CODE**: PROGRAM
- **PROGRAM NAME**: SORT12
- **DEVICE NAME**: PRINTER1
- **ORIGINAL PROBLEM NUM**: BUCHMAN
- **DATE LAST ALTERED**: 03/31/1998
- **TIME LAST ALTERED**: 13:59
- **USER LAST ALTERED**: BUCHMAN
- **VENDOR STATUS**: WAIT
- **VENDOR NUMBER**: 1234567
- **VENDOR PRIORITY**: 04
- **IPCS RECORD NUMBER**: 01234567
- **APAR NUMBER**: 0X00012
- **PTF NUMBER**: UX01234
- **EC NUMBER**: 00034673
- **COMPONENT APARED**: BTAM
- **VENDOR PMR NUMBER**: 01234567
### Change Requester Data

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requested By</td>
<td>ROGERS</td>
</tr>
<tr>
<td>Requester Dept</td>
<td>D40</td>
</tr>
<tr>
<td>Requester Phone</td>
<td>410-654-8954</td>
</tr>
<tr>
<td>Network Name</td>
<td>NOVELL</td>
</tr>
<tr>
<td>System Name</td>
<td>VALUE2</td>
</tr>
<tr>
<td>Device Name</td>
<td>PRINTERS</td>
</tr>
<tr>
<td>Key Item Affected</td>
<td>TAPE05</td>
</tr>
<tr>
<td>Date Required</td>
<td>04/29/1998</td>
</tr>
<tr>
<td>Time Required</td>
<td>09:31</td>
</tr>
<tr>
<td>Change Type</td>
<td>SOFTWARE</td>
</tr>
<tr>
<td>Change Reason</td>
<td>DASD</td>
</tr>
<tr>
<td>Initial Priority</td>
<td>10</td>
</tr>
<tr>
<td>Est. Duration</td>
<td>60:00:00</td>
</tr>
<tr>
<td>Problem Fixed</td>
<td>00000079</td>
</tr>
<tr>
<td>User Form Number</td>
<td>111</td>
</tr>
<tr>
<td>Location Code</td>
<td>BLDG456</td>
</tr>
<tr>
<td>Entry Priv. Class</td>
<td>BUCHMAN</td>
</tr>
<tr>
<td>Date Entered</td>
<td>03/10/1998</td>
</tr>
<tr>
<td>Time Entered</td>
<td>09:37</td>
</tr>
</tbody>
</table>

### Change Status Data

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assignee Name</td>
<td>SMITHJ</td>
</tr>
<tr>
<td>Assignee Department</td>
<td>D40</td>
</tr>
<tr>
<td>Assignee Phone</td>
<td>34232</td>
</tr>
<tr>
<td>Date Assigned</td>
<td>03/03/1998</td>
</tr>
<tr>
<td>Time Assigned</td>
<td>13:30</td>
</tr>
<tr>
<td>Transfer-To Class</td>
<td>MASTER</td>
</tr>
<tr>
<td>Current Priority</td>
<td>10</td>
</tr>
<tr>
<td>Risk Assessment</td>
<td>HIGH</td>
</tr>
<tr>
<td>Change Status</td>
<td>INITIAL</td>
</tr>
<tr>
<td>Approval Status</td>
<td>PENDING</td>
</tr>
<tr>
<td>Approval Status</td>
<td>PENDING</td>
</tr>
<tr>
<td>Planned Start Date</td>
<td>03/06/1998</td>
</tr>
<tr>
<td>Planned Start Time</td>
<td>10:34</td>
</tr>
<tr>
<td>Actual Start Date</td>
<td>03/07/1998</td>
</tr>
<tr>
<td>Actual Start Time</td>
<td>09:35</td>
</tr>
<tr>
<td>Actual Start Date</td>
<td>04/01/1998</td>
</tr>
<tr>
<td>Actual End Date</td>
<td>05:30</td>
</tr>
<tr>
<td>Estimated Effort</td>
<td>0004</td>
</tr>
<tr>
<td>Estimated Duration</td>
<td>60:00:00</td>
</tr>
<tr>
<td>Estimated Impact</td>
<td>5</td>
</tr>
<tr>
<td>Completion Code</td>
<td>5</td>
</tr>
<tr>
<td>Change Status</td>
<td>INITIAL</td>
</tr>
<tr>
<td>Approval Status</td>
<td>PENDING</td>
</tr>
<tr>
<td>Completion Code</td>
<td>5</td>
</tr>
<tr>
<td>Actual Duration</td>
<td>05:07:35</td>
</tr>
<tr>
<td>Unexpected Problems</td>
<td>YES</td>
</tr>
<tr>
<td>Backup Plan Used</td>
<td>YES</td>
</tr>
<tr>
<td>Date Rep. Notified</td>
<td>03/08/1998</td>
</tr>
</tbody>
</table>

### Change Close Data

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Closed By</td>
<td>BUCHMAN</td>
</tr>
<tr>
<td>Closer Department</td>
<td>ABC4</td>
</tr>
<tr>
<td>Closer Phone</td>
<td>567-4562</td>
</tr>
<tr>
<td>Closer Priv. Class</td>
<td>INFOUSR</td>
</tr>
<tr>
<td>Actual Start Date</td>
<td>03/07/1998</td>
</tr>
<tr>
<td>Actual Start Time</td>
<td>09:35</td>
</tr>
<tr>
<td>Completion Date</td>
<td>03/10/1998</td>
</tr>
<tr>
<td>Completion Time</td>
<td>12:35</td>
</tr>
<tr>
<td>Unexpected Problems</td>
<td>YES</td>
</tr>
<tr>
<td>Backup Plan Used</td>
<td>YES</td>
</tr>
<tr>
<td>Date Rep. Notified</td>
<td>03/08/1998</td>
</tr>
</tbody>
</table>

### Requisites

- A10
- 00000012
<table>
<thead>
<tr>
<th>PEOPLE DATA</th>
</tr>
</thead>
<tbody>
<tr>
<td>PERSON NAME -------</td>
</tr>
<tr>
<td>PERSON DEPARTMENT --</td>
</tr>
<tr>
<td>PERSON ROLE -------</td>
</tr>
<tr>
<td>COMPANY NAME -------</td>
</tr>
<tr>
<td>ADDRESS 1 ----------</td>
</tr>
<tr>
<td>CITY/STATE/PROVINCE</td>
</tr>
<tr>
<td>COUNTRY ------------</td>
</tr>
<tr>
<td>POSTAL CODE/ZIP ----</td>
</tr>
<tr>
<td>TSD USER ID --------</td>
</tr>
<tr>
<td>CONTACT METHOD -----</td>
</tr>
<tr>
<td>PHONE NUMBER -------</td>
</tr>
<tr>
<td>MOBILE PHONE NUMBER</td>
</tr>
<tr>
<td>FAX NUMBER ---------</td>
</tr>
<tr>
<td>PAGER NUMBER -------</td>
</tr>
<tr>
<td>E-MAIL ADDRESS -----</td>
</tr>
</tbody>
</table>

| ENTRY PRIV. CLASS -- | MASTER |
| DATE ENTERED ------- | 05/26/1999 |
| TIME ENTERED ------ | 07:29 |
| USER LAST ALTERED - | SCHINDL |

| DATE LAST ALTERED -+ | 05/26/1999 |
| TIME LAST ALTERED - | 07:29 |
THE FOLLOWING SEARCH ARGUMENT PRODUCED '000015' MATCHES IN THE INFORMATION MANAGEMENT for z/OS DATABASE (DATABASE 5)

**Record ID:** INFOUSR  **Privilege Class Description:** INFOUSR

<table>
<thead>
<tr>
<th>PRIVILEGE CLASS DATA</th>
<th>PRIVILEGE CLASS USERIDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>USER ID 1 ----------- INFOUSR</td>
<td>ID 1 - INFOUSR ID 13 -</td>
</tr>
<tr>
<td>TRANSFER-TO CLASS -- MASTER</td>
<td>ID 2 - ID 14 -</td>
</tr>
<tr>
<td>CONTACT NAME -------- SMITH</td>
<td>ID 3 - ID 15 -</td>
</tr>
<tr>
<td>CONTACT DEPARTMENT --- AGFJ</td>
<td>ID 4 - ID 16 -</td>
</tr>
<tr>
<td>CONTACT PHONE ------- 302-5948</td>
<td>ID 5 - ID 17 -</td>
</tr>
<tr>
<td>LOCATION CODE ------- BLDG4</td>
<td>ID 6 - ID 18 -</td>
</tr>
<tr>
<td>OWNING PRIV. CLASS --- BUCHMAN</td>
<td>ID 7 - ID 19 -</td>
</tr>
<tr>
<td>ENTRY PRIV. CLASS ---- MASTER</td>
<td>ID 8 - ID 20 -</td>
</tr>
<tr>
<td>DATE ENTERED -------- 04/05/1997</td>
<td>ID 9 - ID 21 -</td>
</tr>
<tr>
<td>TIME ENTERED -------- 14:04</td>
<td>ID 10 - ID 22 -</td>
</tr>
<tr>
<td>DATE LAST ALTERED --- 03/31/1998</td>
<td>ID 11 - ID 23 -</td>
</tr>
<tr>
<td>TIME LAST ALTERED ---- 13:57</td>
<td>ID 12 - ID 24 -</td>
</tr>
<tr>
<td>USER LAST ALTERED ---- BUCHMAN</td>
<td></td>
</tr>
</tbody>
</table>

**Class Authority**  **SRC Authority**  **Problem Authority**  **Change Authority**  **Configuration Authority**  **Financial Authority**  **PMF Authority**

| DISPLAY - ENTRY -- | DISPLAY - YES | DISPLAY - YES | DISPLAY - YES | DISPLAY - YES | PANEL UPD -- |
| ENTRY --- UPDATE - | ENTRY --- YES | ENTRY --- YES | ENTRY --- YES | ENTRY --- YES | DICT DISP -- |
| UPDATE -- DELETE -- | UPDATE -- YES | UPDATE -- YES | UPDATE -- YES | UPDATE -- YES | DICT UPD --- |
| DELETE -- DELETE -- | DELETE -- YES | DELETE -- YES | DELETE -- YES | DELETE -- YES | PANEL COPY - |
| RULES Authority | DATABASE ADMINISTRATION Authority | ACCESS Authority | PEOPLE Authority | SOLUTION Authority |
| DISPLAY - DBADMIN | DISPLAY - ENTRY -- | DISPLAY - ENTRY --- | ENTRY --- ENTRY --- |
| ENTRY --- TSD CLEANUP | UPDATE -- UPDATE -- | DELETE -- DELETE -- |
| UPDATE -- DELETE -- | |

**Journalized Data**  **Note:** An asterisk on the data fields listed above indicates that the field is journalized.

**Contact Name:** 03/31/1998 13:57 BUCHMAN - SMITH

**Date Last Altered:** 04/05/1997 14:04 SINE - 04/05/1997 14:04 SINE - 03/31/1998 13:57 BUCHMAN - 03/31/1998 13:57 BUCHMAN

**Owning Privilege Class:** 03/31/1998 13:57 BUCHMAN - BUCHMAN

**Time Last Altered:** 04/05/1997 14:04 SINE - 14:04 SINE - 03/31/1998 13:57 BUCHMAN - 13:57 BUCHMAN

**Transfer-To Class:** 03/31/1998 13:57 BUCHMAN - MASTER
### Problem Reporter Data

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reported By</td>
<td>Smith</td>
</tr>
<tr>
<td>Reporter Dept</td>
<td>AB01</td>
</tr>
<tr>
<td>Reporter Phone</td>
<td>301-345-3457</td>
</tr>
<tr>
<td>Date Occurred</td>
<td>02/02/1998</td>
</tr>
<tr>
<td>Time Occurred</td>
<td>13:39</td>
</tr>
<tr>
<td>Network Name</td>
<td>Backbone</td>
</tr>
<tr>
<td>System Name</td>
<td>Value2</td>
</tr>
<tr>
<td>Device Name</td>
<td>Printer1</td>
</tr>
<tr>
<td>Key Item Affected</td>
<td>Printer1</td>
</tr>
<tr>
<td>Date Fix Required</td>
<td>03/03/1998</td>
</tr>
<tr>
<td>Time Fix Required</td>
<td>08:41</td>
</tr>
<tr>
<td>Problem Type</td>
<td>Software</td>
</tr>
<tr>
<td>Problem Status</td>
<td>Closed</td>
</tr>
<tr>
<td>Initial Priority</td>
<td>03</td>
</tr>
<tr>
<td>Outage</td>
<td>01:03:30</td>
</tr>
<tr>
<td>Run Time</td>
<td>01:00:00</td>
</tr>
<tr>
<td>Network Impact</td>
<td>LOSTJ0B</td>
</tr>
<tr>
<td>System Impact</td>
<td>RESTART</td>
</tr>
<tr>
<td>Program Impact</td>
<td>LOSTUSER</td>
</tr>
<tr>
<td>Device Impact</td>
<td>UNUSABLE</td>
</tr>
<tr>
<td>User Form Number</td>
<td>F988</td>
</tr>
<tr>
<td>Location Code</td>
<td>10B</td>
</tr>
<tr>
<td>Outage Type</td>
<td>UN</td>
</tr>
<tr>
<td>Error Code</td>
<td>123456</td>
</tr>
<tr>
<td>Cluster Name</td>
<td>Remote</td>
</tr>
<tr>
<td>Circuit Number</td>
<td>45F</td>
</tr>
<tr>
<td>Entry Priv. Class</td>
<td>BUCHMAN</td>
</tr>
<tr>
<td>Date Entered</td>
<td>03/01/1998</td>
</tr>
<tr>
<td>Time Entered</td>
<td>08:47</td>
</tr>
</tbody>
</table>

### Problem Status Data

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assignee Name</td>
<td>Jonesr</td>
</tr>
<tr>
<td>Assignee Dept</td>
<td>BY6A</td>
</tr>
<tr>
<td>Assignee Phone</td>
<td>689-1234</td>
</tr>
<tr>
<td>Date Opened</td>
<td>02/02/1998</td>
</tr>
<tr>
<td>Time Opened</td>
<td>13:44</td>
</tr>
<tr>
<td>Date Closed</td>
<td>02/26/1998</td>
</tr>
<tr>
<td>Time Closed</td>
<td>13:48</td>
</tr>
<tr>
<td>Total Time</td>
<td>10:00:00</td>
</tr>
</tbody>
</table>

### Problem Close Data

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resolved By</td>
<td>JSmith</td>
</tr>
<tr>
<td>Resolver Dept</td>
<td>DB45</td>
</tr>
<tr>
<td>Resolver Phone</td>
<td>678-8954</td>
</tr>
<tr>
<td>Date Closed</td>
<td>02/26/1998</td>
</tr>
<tr>
<td>Time Closed</td>
<td>13:48</td>
</tr>
<tr>
<td>Duplicate Count</td>
<td>000</td>
</tr>
<tr>
<td>Outage</td>
<td>01:03:30</td>
</tr>
<tr>
<td>Problem Status</td>
<td>CLOSED</td>
</tr>
<tr>
<td>Cause Code</td>
<td>Program</td>
</tr>
<tr>
<td>Device Name</td>
<td>Printer1</td>
</tr>
<tr>
<td>Original Problem Num</td>
<td>000000079</td>
</tr>
<tr>
<td>Cause Change Number</td>
<td>000000080</td>
</tr>
<tr>
<td>Date Rep. Notified</td>
<td>02/10/1998</td>
</tr>
<tr>
<td>Rerun Time</td>
<td>01:00:00</td>
</tr>
<tr>
<td>Vendor PD Time</td>
<td>00:01:00</td>
</tr>
</tbody>
</table>

### Interested Privilege Classes

<table>
<thead>
<tr>
<th>Number</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>15</td>
<td>Console Output</td>
</tr>
<tr>
<td>2</td>
<td>16</td>
<td>Output Data</td>
</tr>
<tr>
<td>3</td>
<td>17</td>
<td>Input Data</td>
</tr>
<tr>
<td>4</td>
<td>18</td>
<td>Source Listing</td>
</tr>
<tr>
<td>5</td>
<td>19</td>
<td>Source</td>
</tr>
<tr>
<td>6</td>
<td>20</td>
<td>SMP Listing</td>
</tr>
<tr>
<td>7</td>
<td>21</td>
<td>Trace Data</td>
</tr>
<tr>
<td>8</td>
<td>22</td>
<td>Graph/Log Data</td>
</tr>
<tr>
<td>9</td>
<td>23</td>
<td>Diagnostic Output</td>
</tr>
<tr>
<td>10</td>
<td>24</td>
<td>Operator Form</td>
</tr>
<tr>
<td>11</td>
<td>25</td>
<td>Probable Cause</td>
</tr>
<tr>
<td>12</td>
<td>26</td>
<td>Dump Data Set</td>
</tr>
<tr>
<td>13</td>
<td>27</td>
<td>Resource Types</td>
</tr>
<tr>
<td>14</td>
<td>28</td>
<td>Resource Names</td>
</tr>
<tr>
<td>Field</td>
<td>Date/Time</td>
<td>Assignee</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>----------------------------------</td>
<td>----------</td>
</tr>
<tr>
<td><strong>RECORD ID:</strong></td>
<td>00000079</td>
<td></td>
</tr>
<tr>
<td><strong>PROBLEM DESCRIPTION:</strong></td>
<td>CAN NOT CONNECT TO HOST</td>
<td></td>
</tr>
<tr>
<td><strong>DETAIL DATA:</strong></td>
<td>0000003E RECS=PROBLEM</td>
<td></td>
</tr>
<tr>
<td><strong>DATE/03/01/1998</strong></td>
<td><strong>TIME/08:47</strong></td>
<td></td>
</tr>
<tr>
<td><strong>CAUSE CODE:</strong></td>
<td>PROGRAM</td>
<td></td>
</tr>
<tr>
<td><strong>CURRENT PHASE:</strong></td>
<td>STUDY</td>
<td></td>
</tr>
<tr>
<td><strong>CURRENT PRIORITY:</strong></td>
<td>03</td>
<td></td>
</tr>
<tr>
<td><strong>DATE FINISHED:</strong></td>
<td>02/25/1998</td>
<td></td>
</tr>
<tr>
<td><strong>DATE LAST ALTED:</strong></td>
<td>03/08/1998 08:46</td>
<td>BUCHMAN</td>
</tr>
<tr>
<td><strong>DATE STARTED:</strong></td>
<td>02/10/1998</td>
<td></td>
</tr>
<tr>
<td><strong>DATE FIX REQUIRED:</strong></td>
<td>03/03/1998</td>
<td></td>
</tr>
<tr>
<td><strong>DATE LAST ALTED:</strong></td>
<td>03/16/1998 10:04</td>
<td></td>
</tr>
<tr>
<td><strong>DATE FINISHED:</strong></td>
<td>03/04/1998</td>
<td></td>
</tr>
<tr>
<td><strong>DATE LAST ALTERED:</strong></td>
<td>03/08/1998 08:46</td>
<td></td>
</tr>
<tr>
<td><strong>DATE FINISHED:</strong></td>
<td>02/25/1998</td>
<td></td>
</tr>
<tr>
<td><strong>DATE LAST ALTED:</strong></td>
<td>03/08/1998 08:46</td>
<td></td>
</tr>
<tr>
<td><strong>DATE LAST ALTERED:</strong></td>
<td>03/31/1998 13:59</td>
<td></td>
</tr>
<tr>
<td><strong>DATE STARTED:</strong></td>
<td>02/10/1998</td>
<td></td>
</tr>
<tr>
<td><strong>INITIAL PRIORITY:</strong></td>
<td>03</td>
<td></td>
</tr>
<tr>
<td><strong>OWNING PRIVILEGE CLASS:</strong></td>
<td>BUCHMAN</td>
<td></td>
</tr>
<tr>
<td><strong>PROBABLE CAUSE:</strong></td>
<td>HARDWARE</td>
<td></td>
</tr>
<tr>
<td><strong>PROBLEM STATUS:</strong></td>
<td>OPEN</td>
<td></td>
</tr>
<tr>
<td><strong>TARGET DATE:</strong></td>
<td>03/03/1998</td>
<td></td>
</tr>
<tr>
<td><strong>TIME FIX REQUIRED:</strong></td>
<td>03/08/1998</td>
<td></td>
</tr>
<tr>
<td><strong>TIME FINISHED:</strong></td>
<td>03/03/1998</td>
<td></td>
</tr>
</tbody>
</table>
DATE 04/03/1998
TIME 08:06:34

RECORD ID: 00000079
PROBLEM DESCRIPTION: CAN NOT CONNECT TO HOST

TIME FIX REQUIRED -------- 03/01/1998 08:47 BUCHMAN - 08:41
TIME LAST ALTERED -------- 03/01/1998 08:47 BUCHMAN - 08:47
TIME STARTED ------------- 03/01/1998 13:51 BUCHMAN - 13:46
TRACKED BY --------------- 03/01/1998 13:51 BUCHMAN - SMITH
TRANSFER-TO CLASS -------- 03/01/1998 13:51 BUCHMAN - MASTER
USER LAST ALTERED -------- 03/01/1998 08:47 BUCHMAN - BUCHMAN
VENDOR PRIORITY ---------- 03/08/1998 08:46 BUCHMAN - 04
VENDOR STATUS ------------ 03/08/1998 08:46 BUCHMAN - WAIT

DATE 06/29/1999
TIME 11:06:34

RECORD ID: CN12345

PEOPLE DATA
PERSON NAME -------* Richard Wagner
PERSON DEPARTMENT -- X74B
PERSON ROLE -------* CUSTOMER
COMPANY NAME ------- Supermarkets Plus
ADDRESS 1 ---------- 3050 Independence Ave.
ADDRESS 2 ----------
CITY/STATE/PROVINCE East Brunswick, NJ
COUNTRY ------------ USA
POSTAL CODE/ZIP ---- 08861
TSD USER ID --------
TSD SITE ID --------

CONTACT DATA
CONTACT METHOD ----* PHONE
PHONE NUMBER ------- 732-555-3333
MOBILE PHONE NUMBER
FAX NUMBER ----------
PAGER NUMBER -------
E-MAIL ADDRESS ----- wagner@supermarketsplus.com

CONTROL DATA
ENTRY PRIV. CLASS -- MASTER
DATE ENTERED ------- 05/26/1999
TIME ENTERED ------- 07:29
DATE LAST ALTERED -*
TIME LAST ALTERED -*
USER LAST ALTERED -* SCHINDL
## Problem Management Reports

**DATE 03/24/1998***
**TIME 13:47:29***

**INFORMATION MANAGEMENT for z/OS PERIODIC PROBLEM STATUS REPORT***

---

**THE FOLLOWING SEARCH ARGUMENT PRODUCED '000016' MATCHES IN THE INFORMATION MANAGEMENT for z/OS DATABASE (DATABASE 5)**

---

**PROBLEMS ENTERED THIS PERIOD: '02/23/1998' THROUGH '03/01/1998'**

<table>
<thead>
<tr>
<th>DATE ENTERED</th>
<th>PROBLEM NUMB/ID</th>
<th>setStatus</th>
<th>PRI</th>
<th>PHASE</th>
<th>IMPACT AFFECTED</th>
<th>ASSIGNEE NAME</th>
<th>DATE ENTERED</th>
<th>PROBLEM DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>02/02/1998</td>
<td>00000034</td>
<td>OPEN</td>
<td>01</td>
<td>TEST</td>
<td>LOSTJOB</td>
<td>PRINTER1 PARKERK</td>
<td>02/30/1998</td>
<td>CANNOT PRINT IN LOTUS</td>
</tr>
<tr>
<td>02/26/1998</td>
<td>00000085</td>
<td>CLOS</td>
<td>03</td>
<td>STUDY</td>
<td>RESTAR</td>
<td>PRINTER1 JONESR</td>
<td>02/03/1998</td>
<td>CAN NOT CONNECT TO HOST</td>
</tr>
</tbody>
</table>

**PROBLEMS CLOSED THIS PERIOD: '02/23/1998' THROUGH '03/01/1998'**

<table>
<thead>
<tr>
<th>DATE CLOSED</th>
<th>PROBLEM NUMB/ID</th>
<th>DUP CNT</th>
<th>CUR PRI</th>
<th>CAUSE</th>
<th>SYSTEM KEY ITEM</th>
<th>DATE ENTERED</th>
<th>PROBLEM DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>03/01/1998</td>
<td>00000079</td>
<td>000</td>
<td>03</td>
<td>PROGRAM</td>
<td>RESTAR PRINTER1 JSMITH</td>
<td>03/01/1998</td>
<td>CAN NOT CONNECT TO HOST</td>
</tr>
</tbody>
</table>

**PRIORITY 01 HOLD-OVER PROBLEMS**

<table>
<thead>
<tr>
<th>DATE ENTERED</th>
<th>PROBLEM NUMB/ID</th>
<th>BYP **</th>
<th>CURRENT **</th>
<th>SYSTEM KEY ITEM</th>
<th>DATE ENTERED</th>
<th>PROBLEM DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>03/01/1998</td>
<td>00000100</td>
<td>YES</td>
<td>01 DORMANT</td>
<td>RESTART DISK47 JSMITH</td>
<td>03/01/1998</td>
<td>SYSTEM WILL NOT DISPLAY MVS1</td>
</tr>
<tr>
<td>02/02/1997</td>
<td>00000101</td>
<td>YES</td>
<td>01 STUDY</td>
<td>LOSTJOB PRINTER3 YOUNG</td>
<td>03/01/1998</td>
<td>LAN IS DOWN</td>
</tr>
</tbody>
</table>

**TOTAL PERIOD PERIOD BY PRIORITY OPEN ENTERED CLOSED HOLD OVER 01 02 03 04 05 06 07 08 09 10 11-99 NONE**

|                | 11 | 2  | 1  | 2  | 2  | 1  | 2  | 0  | 0  | 0  | 0  | 0  | 0  | 4  |

---

**DATE 03/24/1998***
**TIME 13:47:29***

**INFORMATION MANAGEMENT for z/OS PROBLEM CALENDAR REPORT***

---

**THE FOLLOWING SEARCH ARGUMENT PRODUCED '000004' MATCHES IN THE INFORMATION MANAGEMENT for z/OS DATABASE (DATABASE 5)**

---

**DATE ENTERED**
**PROBLEM NUMB/ID**
**KEY ITEM**
**CURRENT DATE**
**DATE**
**DATE**
**DATE**
**DATE**
**DATE**
**DATE**
**DATE**
**LAST ALTERED**

<table>
<thead>
<tr>
<th>DATE ENTERED</th>
<th>PROBLEM NUMB/ID</th>
<th>KEY ITEM</th>
<th>CURRENT DATE</th>
<th>DATE FIX</th>
<th>DATE</th>
<th>DATE</th>
<th>DATE</th>
<th>DATE</th>
<th>DATE</th>
<th>LAST ALTERED</th>
</tr>
</thead>
<tbody>
<tr>
<td>04/05/1997</td>
<td>00000013 DISK01</td>
<td>ASSI</td>
<td>04 04/05/1997 04/05/1997 04/05/1997 04/05/1997 04/05/1997 04/05/1997 04/05/1997 04/05/1997 04/05/1997</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---
The following search argument produced '000003' matches in the Information Management for z/OS database (database 5).

Problems currently assigned to assignee: 'MARYG'

<table>
<thead>
<tr>
<th>DATE</th>
<th>PROBLEM</th>
<th>STAT</th>
<th>PRI</th>
<th>PHASE</th>
<th>NUMB/ID</th>
<th>CURRENT ****</th>
<th>ASSIGNED ***</th>
<th>FINISHED ***</th>
<th>BYP</th>
<th>PROBLEM DESCRIPTION</th>
</tr>
</thead>
</table>

Totals for problems assigned to assignee: 'MARYG'

<table>
<thead>
<tr>
<th>INITIAL</th>
<th>OPEN</th>
<th>CLOSED</th>
<th>01</th>
<th>02</th>
<th>03</th>
<th>04</th>
<th>05</th>
<th>06</th>
<th>07</th>
<th>08</th>
<th>09</th>
<th>10</th>
<th>11-99</th>
<th>NONE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Problems unassigned and not closed

<table>
<thead>
<tr>
<th>DATE</th>
<th>PROBLEM</th>
<th>STAT</th>
<th>PRI</th>
<th>PHASE</th>
<th>NUMB/ID</th>
<th>CURRENT ****</th>
<th>PROBLEM</th>
<th>KEY ITEM</th>
<th>REPORTED BY</th>
<th>DATE FIX</th>
<th>PROBLEM DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>03/16/1998</td>
<td>00000003</td>
<td>OPEN</td>
<td>01</td>
<td>FIXP</td>
<td>HDW</td>
<td>APPL#02</td>
<td>BUCHMAN</td>
<td>03/23/1998</td>
<td>SCATTERED DATA ON SCREEN</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Totals for unassigned problems

<table>
<thead>
<tr>
<th>UNASSIGNED PROBLEMS</th>
<th>INITIAL</th>
<th>OPEN</th>
<th>CLOSED</th>
<th>01</th>
<th>02</th>
<th>03</th>
<th>04</th>
<th>05</th>
<th>06</th>
<th>07</th>
<th>08</th>
<th>09</th>
<th>10</th>
<th>11-99</th>
<th>NONE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>
## Change Management Reports

**DATE 03/10/1998**

**TIME 12:10:58**

**INFORMATION MANAGEMENT for z/OS PERIODIC CHANGE STATUS REPORT**

<table>
<thead>
<tr>
<th>DATE</th>
<th>CHANGE</th>
<th>CHANGE STATUS</th>
<th>CURRENT</th>
<th>KEY ITEM</th>
</tr>
</thead>
<tbody>
<tr>
<td>01/04/1998</td>
<td>00000089</td>
<td>HARDWARE INIT</td>
<td>PEN 01</td>
<td>DORMANT</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>BLACKM</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>REPLACE THE PRINTER IN BLDG4</td>
</tr>
<tr>
<td>01/04/1998</td>
<td>00000091</td>
<td>HARDWARE CLOS</td>
<td>PEN 06</td>
<td>TEST</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>LOW</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>BLACKM</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>LOW</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>REPLACE THE PRINTER IN BLDG2</td>
</tr>
</tbody>
</table>

**CURRENT PERIOD: '02/01/1998' THROUGH '02/28/1998'**

<table>
<thead>
<tr>
<th>DATE</th>
<th>CHANGE</th>
<th>CHANGE STATUS</th>
<th>CURRENT</th>
<th>KEY ITEM</th>
</tr>
</thead>
<tbody>
<tr>
<td>02/13/1998</td>
<td>00000090</td>
<td>HARDWARE OPEN</td>
<td>PEN 05</td>
<td>TEST</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>LOW</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>SMITHJ</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>TAPE45</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>RELOAD TAPE IN DRIVE B</td>
</tr>
</tbody>
</table>

**NEXT PERIOD: '03/01/1998' THROUGH '03/31/1998'**

<table>
<thead>
<tr>
<th>DATE</th>
<th>CHANGE</th>
<th>CHANGE STATUS</th>
<th>CURRENT</th>
<th>KEY ITEM</th>
</tr>
</thead>
<tbody>
<tr>
<td>03/10/1998</td>
<td>00000025</td>
<td>SOFTWARE OPEN</td>
<td>03 TEST</td>
<td>HIGH</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>GREENP</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>PRINTER3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>NEW CHANGE RECORD</td>
</tr>
<tr>
<td>03/23/1998</td>
<td>00000081</td>
<td>HARDWARE INIT</td>
<td>02 DORMANT</td>
<td>LOW</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>ADAMS</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>DISK47</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>CHANGE COM1 TO RS-232</td>
</tr>
<tr>
<td>03/24/1998</td>
<td>00000082</td>
<td>DOCUMENT INIT</td>
<td>01 DORMANT</td>
<td>01</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>MARKSW</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>DISK46</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>REPLACE PRINTER3 WITH PRINTER2</td>
</tr>
</tbody>
</table>

**REPORT PERIOD**

<table>
<thead>
<tr>
<th>REPORT PERIOD</th>
<th>CHANGES REPORTED</th>
<th>OVERDUE</th>
<th>STATUS</th>
<th>COMPLETE</th>
<th>REJECTED</th>
<th>PENDING</th>
<th>APPROVED</th>
<th>NOT REQ'ED</th>
</tr>
</thead>
<tbody>
<tr>
<td>PREVIOUS: 01/01/1998 - 01/31/1998</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>CURRENT: 02/01/1998 - 02/28/1998</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>NEXT: 03/01/1998 - 03/31/1998</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>REPORT TOTALS:</td>
<td>6</td>
<td>3</td>
<td>5</td>
<td>1</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>3</td>
</tr>
</tbody>
</table>
### Changes

<table>
<thead>
<tr>
<th>Date</th>
<th>Change</th>
<th>Type</th>
<th>Current Date</th>
<th>Assigned Name</th>
<th>Date Assigned</th>
<th>Start</th>
<th>Start</th>
<th>End</th>
<th>Start</th>
<th>End</th>
<th>Last Altered</th>
</tr>
</thead>
<tbody>
<tr>
<td>03/10/1998</td>
<td>00000025</td>
<td>SOFTWARE</td>
<td>OPEN</td>
<td></td>
<td>03/10/1998</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>03/24/1998</td>
<td>00000082</td>
<td>DOCUMENT</td>
<td>INIT</td>
<td></td>
<td>03/10/1998</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>04/01/1998</td>
<td>00000083</td>
<td>DOCUMENT</td>
<td>INIT</td>
<td></td>
<td>03/10/1998</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>04/03/1998</td>
<td>00000084</td>
<td>SOFTWARE</td>
<td>OPEN</td>
<td></td>
<td>03/10/1998</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Change Approver Summary

The following change(s) must be approved by privilege class 'BLACKM'.

<table>
<thead>
<tr>
<th>Date</th>
<th>Change</th>
<th>Type</th>
<th>Risk</th>
<th>Est. Eff</th>
<th>Estimated Duration</th>
<th>Assignee Name</th>
<th>Date Assigned</th>
<th>Num</th>
<th>Act</th>
<th>Change Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>04/29/1998</td>
<td>00000080</td>
<td>SOFTWARE</td>
<td>HIGH</td>
<td>0004</td>
<td>60:00:00</td>
<td>BLACKM</td>
<td>03/09/1998</td>
<td>12</td>
<td>03/09/1998</td>
<td>12 PASSWORD RESET</td>
</tr>
<tr>
<td>04/30/1998</td>
<td>00000085</td>
<td>HARDWARE</td>
<td>LOW</td>
<td>0012</td>
<td>05:00:00</td>
<td>SMITHJ</td>
<td>03/09/1998</td>
<td>123</td>
<td>03/09/1998</td>
<td>REPLACE TERMINAL WITH PC</td>
</tr>
<tr>
<td>04/30/1998</td>
<td>00000086</td>
<td>DOCUMENT</td>
<td>03</td>
<td>0004</td>
<td>30:00:00</td>
<td>MCCARTHOR</td>
<td>03/09/1998</td>
<td>236</td>
<td>03/09/1998</td>
<td>REMOVE CHAPTER 3 FROM SOFTWARE DOCUMENTATION</td>
</tr>
</tbody>
</table>

End of change(s) for: BLACKM  
Contact Name: MARY BLACK  
Dept: D40  
Phone: 309-874-3782  
Location: TEXAS
The following '000001' change(s) must be approved by privilege class 'SMITHM'.

<table>
<thead>
<tr>
<th>Record ID: 00000085</th>
<th>Change Description: Replace Terminal with PC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Approver Privilege Class: SMITHM</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Change Requester Data</th>
<th>Change Status Data</th>
<th>Reviewer Class</th>
<th>Approver Status/Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requested By: SMITHM</td>
<td>Assignee Name: SMITHJ</td>
<td>1: P</td>
<td>SMITHM</td>
</tr>
<tr>
<td>Requester Dept: D40</td>
<td>Assignee Department: D40</td>
<td>2: A</td>
<td>INFUSER</td>
</tr>
<tr>
<td>Requester Phone: 410-654-8955</td>
<td>Assignee Phone: 301-345-6739</td>
<td>3:</td>
<td></td>
</tr>
<tr>
<td>Network Name: ADVANTIS</td>
<td>Date Assigned: 03/09/1998</td>
<td>4:</td>
<td></td>
</tr>
<tr>
<td>System Name: VALUE2</td>
<td>Time Assigned: 07:30</td>
<td>5:</td>
<td></td>
</tr>
<tr>
<td>Program Name: SORT</td>
<td>Transfer-To Class: MASTER</td>
<td>6:</td>
<td></td>
</tr>
<tr>
<td>Device Name: PRINTER3</td>
<td>Current Priority: 10</td>
<td>7:</td>
<td></td>
</tr>
<tr>
<td>Key Item Affected: PRINTER3</td>
<td>Coordinator Name: GREGORY</td>
<td>8:</td>
<td></td>
</tr>
<tr>
<td>Date Required: 04/30/1998</td>
<td>Coordinator Dept: D45</td>
<td>9:</td>
<td></td>
</tr>
<tr>
<td>Time Required: 10:31</td>
<td>Coordinator Phone: 345-833-9858</td>
<td>10:</td>
<td></td>
</tr>
<tr>
<td>Change Type: HARDWARE</td>
<td>Risk Assessment: LOW</td>
<td>11:</td>
<td></td>
</tr>
<tr>
<td>Change Status: OPEN</td>
<td>Change Status: OPEN</td>
<td>12:</td>
<td></td>
</tr>
<tr>
<td>Approval Status: PENDING</td>
<td>Current Phase: TEST</td>
<td>13:</td>
<td></td>
</tr>
<tr>
<td>Change Reason: DASD</td>
<td>Approval Status: PENDING</td>
<td>14:</td>
<td></td>
</tr>
<tr>
<td>Initial Priority: 4</td>
<td>Planned Start Date: 03/02/1998</td>
<td>15:</td>
<td></td>
</tr>
<tr>
<td>Est. Duration: 05:00:00</td>
<td>Planned Start Time: 08:00</td>
<td>16:</td>
<td></td>
</tr>
<tr>
<td>Problem Fixed: 00000079</td>
<td>Actual Start Date: 03/02/1998</td>
<td>17:</td>
<td></td>
</tr>
<tr>
<td>User Form Number: 112</td>
<td>Actual Start Time: 09:35</td>
<td>18:</td>
<td></td>
</tr>
<tr>
<td>Location Code: BLDG456</td>
<td>Planned End Date: 04/29/1998</td>
<td>19:</td>
<td></td>
</tr>
<tr>
<td>Entry Priv. Class: BUCHMAN</td>
<td>Planned End Time: 05:00</td>
<td>20:</td>
<td></td>
</tr>
<tr>
<td>Date Entered: 03/10/1998</td>
<td>Estimated Duration: 05:00:00</td>
<td>21:</td>
<td></td>
</tr>
<tr>
<td>Time Entered: 08:39</td>
<td>Estimated Effort: 0012</td>
<td>22:</td>
<td></td>
</tr>
<tr>
<td>Date Last Altered: 03/10/1998</td>
<td>Estimated Duration: 05:00:00</td>
<td>23:</td>
<td></td>
</tr>
<tr>
<td>Time Last Altered: 08:44</td>
<td>Estimated Effort: 0012</td>
<td>24:</td>
<td></td>
</tr>
<tr>
<td>Change Type: HARDWARE</td>
<td>Estimated Duration: 05:00:00</td>
<td>25:</td>
<td></td>
</tr>
</tbody>
</table>

Co-Requisites:
Pre-Requisites:

Detail Data: 00000046 REC=CHANGE DATE/03/10/1998 TIME/08:39 CLAE/SMITHM

Journalyzed Data: ***Note - an asterisk on the data fields listed above indicates that the field is journalyzed. 

Actual Start Date: 03/10/1998 08:39 SMITHM - 03/02/1998
Approval Status: 03/10/1998 08:39 SMITHM - PENDING
Assignee Name: 03/10/1998 08:39 SMITHJ - BLACKM
03/10/1998 08:44 SMITHM - SMITHJ
Change Status: 03/10/1998 08:39 SMITHM - OPEN
Change Type: 03/10/1998 08:39 SMITHM - HARDWARE
## INFORMATION MANAGEMENT for z/OS CHANGE APPROVER DETAIL REPORT

**RECORD ID: 00000085**  
**CHANGE DESCRIPTION:** REPLACE TERMINAL WITH PC  
**APPROVER PRIVILEGE CLASS:** SMITHM

- **CURRENT PHASE:** 03/10/1998 08:39 SMITHM - TEST
- **CURRENT PRIORITY:** 03/10/1998 08:39 SMITHM - 10
- **DATE ASSIGNED:** 03/10/1998 08:39 SMITHM - 03/09/1998
- **DATE LAST ALTERED:** 03/10/1998 08:39 SMITHM - 03/10/1998
- **DATE REQUIRED:** 03/10/1998 08:39 SMITHM - 04/30/1998
- **ESTIMATED DURATION:** 03/10/1998 08:39 SMITHM - 60:00:00  03/10/1998 08:44 SMITHM - 05:00:00
- **ESTIMATED EFFORT:** 03/10/1998 08:39 SMITHM - 0004  03/10/1998 08:44 SMITHM - 0012
- **INITIAL PRIORITY:** 03/10/1998 08:39 SMITHM - 04
- **PLANNED END DATE:** 03/10/1998 08:39 SMITHM - 04/29/1998
- **RISK ASSESSMENT:** 03/10/1998 08:39 SMITHM - HIGH  03/10/1998 08:44 SMITHM - LOW
- **TIME LAST ALTERED:** 03/10/1998 08:39 SMITHM - 08:39  03/10/1998 08:44 SMITHM - 08:44
- **TIME REQUIRED:** 03/10/1998 08:39 SMITHM - 10:31

**END OF CHANGE(S) FOR: SMITHM**  
**CONTACT NAME:** MATT SMITH  
**DEPT:** D40  
**PHONE:** 389-367-6472  
**LOCATION:** MD

---

## INFORMATION MANAGEMENT for z/OS CHANGES WITH RELATED ACTIVITIES REPORT

**DATE 03/10/1998**  
**TIME 13:04:17**

<table>
<thead>
<tr>
<th>STAT</th>
<th>NUMB/ID</th>
<th>DATE REQUIRED</th>
<th>RISK/IMPACT</th>
<th>ACTUAL IMPACT</th>
<th>EST/ACTUAL EFF</th>
<th>PLANNED/ACTUAL START</th>
<th>PLANNED/ACTUAL END DATE</th>
<th>APP</th>
<th>PHASE</th>
<th>PRI</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>OPEN</td>
<td>00000085</td>
<td>04/30/1998 LOW</td>
<td>0012</td>
<td>03/02/1998</td>
<td>04/29/1998 PEN TEST</td>
<td>10 HARDWARE REPLACE TERMINAL WITH PC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OPEN</td>
<td>00000093</td>
<td>03/14/1998 LOW</td>
<td>03/09/1998</td>
<td>MARKSB</td>
<td>HARDWARE THE PC HAS BEEN ORDERED</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OPEN</td>
<td>00000086</td>
<td>04/30/1998 03</td>
<td>0004</td>
<td>03/01/1998</td>
<td>04/29/1998 PEN TEST</td>
<td>10 DOCUMENT REMOVE CHAPTER 3 FROM SOFTWARE DOCUMENTATION</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INIT</td>
<td>00000094</td>
<td>03/10/1998 LOW</td>
<td>03/09/1998</td>
<td>BLACKM</td>
<td>PROBLEMS PRINTING</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### CHANGES NOT SCHEDULED

<table>
<thead>
<tr>
<th>DATE</th>
<th>CHANGE</th>
<th>NUMB/ID</th>
<th>TYPE</th>
<th>KEY ITEM</th>
<th>AFFECTED</th>
<th>CURRENT PHASE</th>
<th>PRI</th>
<th>REQUESTED BY</th>
<th>RISK</th>
<th>EST</th>
<th>EF</th>
<th>ACT</th>
<th>NUMB</th>
<th>ACT CHANGE DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>01/04/1998</td>
<td>HARDWARE</td>
<td>00000089</td>
<td>PRINTER4</td>
<td>DORMANT</td>
<td>01</td>
<td>MCCARTHOR</td>
<td>HIGH</td>
<td>0003</td>
<td>0</td>
<td>REPLACE THE PRINTER IN BLDG4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>02/13/1998</td>
<td>HARDWARE</td>
<td>00000090</td>
<td>TAPE45</td>
<td>TEST</td>
<td>05</td>
<td>JONES</td>
<td>LOW</td>
<td>0020</td>
<td>0</td>
<td>RELOAD TAPE IN DRIVE B</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>03/10/1998</td>
<td>SOFTWARE</td>
<td>00000025</td>
<td>PRINTER3</td>
<td>TEST</td>
<td>03</td>
<td>TAYLOR</td>
<td>HIGH</td>
<td>0010</td>
<td>0</td>
<td>NEW CHANGE RECORD</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>03/23/1998</td>
<td>HARDWARE</td>
<td>00000081</td>
<td>DISK47</td>
<td>DORMANT</td>
<td>02</td>
<td>JONESR</td>
<td>LOW</td>
<td>0020</td>
<td>0</td>
<td>CHANGE COM1 TO RS-232</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>03/24/1998</td>
<td>DOCUMENT</td>
<td>00000082</td>
<td>TAPE5</td>
<td>DORMANT</td>
<td>01</td>
<td>BLACKM</td>
<td>01</td>
<td>0003</td>
<td>0</td>
<td>REPLACE PRINTER3 WITH PRINTER2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>04/01/1998</td>
<td>PERFORM</td>
<td>00000083</td>
<td>DISK46</td>
<td>ACTIVE</td>
<td>04</td>
<td>MCCARTHOR</td>
<td>LOW</td>
<td>0001</td>
<td>0</td>
<td>CHANGE USERID FOR GREGG</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>04/02/1998</td>
<td>SOFTWARE</td>
<td>00000084</td>
<td>TAPE03</td>
<td>ACTIVE</td>
<td>10</td>
<td>SMITHJ</td>
<td>LOW</td>
<td>0002</td>
<td>0</td>
<td>INSTALL SOFTWARE FROM TAPE03</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### ACTIVITIES NOT SCHEDULED

<table>
<thead>
<tr>
<th>DATE</th>
<th>ACTIVITY</th>
<th>REC ID</th>
<th>TYPE</th>
<th>KEY ITEM</th>
<th>AFFECTED</th>
<th>CURRENT PHASE</th>
<th>PRI</th>
<th>REQUESTED BY</th>
<th>PARENT CHANGE</th>
<th>ACTIVITY NAME</th>
<th>ACTIVITY DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>03/10/1998</td>
<td>PERFORM</td>
<td>00000094</td>
<td>PRINTER9</td>
<td>HOLDING</td>
<td>10</td>
<td>SMITHM</td>
<td>00000086</td>
<td>PRINTDOC</td>
<td>PROBLEMS PRINTING</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

Data Reporting User's Guide
## Configuration Management Reports

### Hardware Components at Location Code: 'ANNAPOLI'

<table>
<thead>
<tr>
<th>DEV REC ID</th>
<th>DEV TYPE</th>
<th>SERIAL</th>
<th>ORDER</th>
<th>MODEL</th>
<th>CONNECTS</th>
<th>FINANCIAL</th>
<th>SYSTEM</th>
<th>*** COMPONENT ***</th>
<th>DATE OF</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU CPU3</td>
<td>CP3000</td>
<td>CP375356</td>
<td>3456789</td>
<td>DESKTOP</td>
<td>MVS</td>
<td>PCS</td>
<td>JUDMVS10</td>
<td>BLACKM</td>
<td>ORDER</td>
</tr>
<tr>
<td>CPU MV1</td>
<td>CP4000</td>
<td>CP545454</td>
<td>F459042</td>
<td>WORKSTN</td>
<td>MV1</td>
<td>PCS</td>
<td>JUDMVS10</td>
<td>BLACKM</td>
<td>INSTALL</td>
</tr>
<tr>
<td>CPU VM4</td>
<td>AF498B</td>
<td>AF43854</td>
<td>F373492</td>
<td>WORKSTN</td>
<td>MVS</td>
<td>PCS</td>
<td>JUDMVS10</td>
<td>MATHS</td>
<td>INSTALL</td>
</tr>
<tr>
<td>CPU JDCTLR5</td>
<td>CTR3745</td>
<td>CT22234</td>
<td>D637922</td>
<td>DESKTOP</td>
<td>MV1</td>
<td>PCS</td>
<td>JUDMVS10</td>
<td>SMITHM</td>
<td>PLANNED</td>
</tr>
</tbody>
</table>

### Software Components

<table>
<thead>
<tr>
<th>REC ID/ LOC CD/ NAME</th>
<th>TYPE</th>
<th>SOFTWARE LEVEL</th>
<th>FIX LEVEL</th>
<th>FINANCIAL</th>
<th>DATE OF</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>FILES BLDG183 PPR</td>
<td>APP</td>
<td>5 0002 0002 IMSR119 PCS</td>
<td>TEST</td>
<td>03/14/1998 FILES CONTAINING UPDATED CONFIG RECORDS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>00000106 SCRNSVR</td>
<td>5 0002 0004 MVS3270</td>
<td>PLANNED</td>
<td>A NEW SCREEN SAVER WILL BE ADDED</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MVS A BLDG5 SCP</td>
<td>3 0004 0002 IMSR118 PCS</td>
<td>INSTALL</td>
<td>03/13/1998 SCP CONFIGURATION RECORD</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00000104 SCRNSVR</td>
<td>5 0002 0004 MVS3270</td>
<td>PLANNED</td>
<td>02/28/1998 A NEW SCREEN SAVER WILL BE ADDED</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MVS B BLDG5 TPX</td>
<td>1 0003 0007 IDFR149 PCS</td>
<td>INSTALL</td>
<td>03/13/1998 TPX CONFIGURATION RECORD FOR MVS SYSTEM</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00000110 SCRNSVR</td>
<td>5 0002 0004 MVS3270</td>
<td>PLANNED</td>
<td>A NEW SCREEN SAVER WILL BE ADDED</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRDPRO BLDG183 APP</td>
<td>APP</td>
<td>5 0002 0002 IDFR149 PCS</td>
<td>ORDER</td>
<td>03/14/1998 WORD PROCESSOR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>00000108 SCRNSVR</td>
<td>5 0002 0004 MVS3270</td>
<td>PLANNED</td>
<td>A NEW SCREEN SAVER WILL BE ADDED</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Hardware Configuration Map

<table>
<thead>
<tr>
<th>LEVEL</th>
<th>COMPONENT</th>
<th>DSP</th>
<th>GEN</th>
<th>TYPE</th>
<th>SERIAL</th>
<th>LOCATION</th>
<th>MODEL</th>
<th>CENTER</th>
<th>FINANCIAL</th>
<th>SYSTEM</th>
<th>OWNER</th>
<th>STATUS</th>
<th>DATE OF</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>MVS1</td>
<td>CPU</td>
<td>CP4000</td>
<td>CPU</td>
<td>CP34545</td>
<td>ANNAPOLI</td>
<td>WORKSTN</td>
<td>ANNAPO</td>
<td>PCS</td>
<td>JUDMVS10</td>
<td>DAVIES</td>
<td>INSTALL</td>
<td>04/11/97</td>
</tr>
<tr>
<td>3</td>
<td>JDCTLR5</td>
<td>CTL</td>
<td>CTLR3745</td>
<td>CTL</td>
<td>CT222234</td>
<td>ANNAPOLI</td>
<td>DESKTOP</td>
<td>JUDICAL</td>
<td>PCS</td>
<td>JUDMVS10</td>
<td>SMITHM</td>
<td>PLANNED</td>
<td>04/11/97</td>
</tr>
<tr>
<td>4</td>
<td>LANONE</td>
<td>LOP</td>
<td>PS/2M4545</td>
<td>LOP</td>
<td>PS123458</td>
<td>BALTIMOR</td>
<td>WORKSTN</td>
<td>GBURG</td>
<td>PCS</td>
<td>JUDMYS10</td>
<td>MCCARTHOR</td>
<td>DOWN</td>
<td>04/11/97</td>
</tr>
<tr>
<td>5</td>
<td>W5123459</td>
<td>VID</td>
<td>PS/2M6387</td>
<td>VID</td>
<td>12365434</td>
<td>183R15</td>
<td>DESKTOP</td>
<td>JUDICAL</td>
<td>PCS</td>
<td>JUDMYS10</td>
<td>JJONES</td>
<td>DOWN</td>
<td>04/11/97</td>
</tr>
<tr>
<td>5</td>
<td>W5123460</td>
<td>VID</td>
<td>PS/2M6387</td>
<td>VID</td>
<td>g6757555</td>
<td>h7895f5</td>
<td>DESKTOP</td>
<td>JUDICAL</td>
<td>PCS</td>
<td>JUDMYS10</td>
<td>JJONES</td>
<td>DOWN</td>
<td>04/11/97</td>
</tr>
<tr>
<td>4</td>
<td>NETWORK</td>
<td>LOP</td>
<td>PS/2M6545</td>
<td>LOP</td>
<td>PS124658</td>
<td>BALTIMOR</td>
<td>DESKTOP</td>
<td>ANNAPO</td>
<td>PCS</td>
<td>JUDMYS10</td>
<td>DAVIES</td>
<td>TEST</td>
<td>03/10/98</td>
</tr>
<tr>
<td>2</td>
<td>CPU3</td>
<td>CPU</td>
<td>CP3000</td>
<td>CPU</td>
<td>CP375356</td>
<td>ANNAPOLI</td>
<td>DESKTOP</td>
<td>ANNAPO</td>
<td>PCS</td>
<td>JUDMYS10</td>
<td>BLACKM</td>
<td>ORDER</td>
<td>04/11/97</td>
</tr>
<tr>
<td>2</td>
<td>VM4</td>
<td>CPU</td>
<td>ADF4988</td>
<td>CPU</td>
<td>ADF43854</td>
<td>ANNAPOLI</td>
<td>WORKSTN</td>
<td>JUDICIAL</td>
<td>PCS</td>
<td>JUDMYS10</td>
<td>SMITHJ</td>
<td>INSTALL</td>
<td>04/11/97</td>
</tr>
<tr>
<td>3</td>
<td>CPU3</td>
<td>CPU</td>
<td>CP3000</td>
<td>CPU</td>
<td>CP375356</td>
<td>ANNAPOLI</td>
<td>DESKTOP</td>
<td>ANNAPO</td>
<td>PCS</td>
<td>JUDMYS10</td>
<td>BLACKM</td>
<td>ORDER</td>
<td>04/11/97</td>
</tr>
</tbody>
</table>

### Software Configuration Map

<table>
<thead>
<tr>
<th>LEVEL</th>
<th>COMPONENT</th>
<th>DSP</th>
<th>GEN</th>
<th>TYPE</th>
<th>SERIAL</th>
<th>LOCATION</th>
<th>MODEL</th>
<th>CENTER</th>
<th>FINANCIAL</th>
<th>SYSTEM</th>
<th>OWNER</th>
<th>STATUS</th>
<th>DATE OF</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>APPL01</td>
<td>4</td>
<td>APP</td>
<td>3</td>
<td>0002</td>
<td>0004</td>
<td>3</td>
<td>BLG931</td>
<td>JUDICIAL</td>
<td>PCS</td>
<td>SY5001</td>
<td>SMITH</td>
<td>INSTALL</td>
</tr>
<tr>
<td>1</td>
<td>APPL02</td>
<td>1</td>
<td>APP</td>
<td>2</td>
<td>0001</td>
<td>0004</td>
<td>1</td>
<td>BLG931</td>
<td>JUDICIAL</td>
<td>PCS</td>
<td>SY5001</td>
<td>JONES</td>
<td>TEST</td>
</tr>
</tbody>
</table>

---

Report Samples

Data Reporting User's Guide

---

Page 287
THE FOLLOWING SEARCH ARGUMENT PRODUCED '000004' MATCHES IN THE INFORMATION MANAGEMENT for z/OS DATABASE (DATABASE 5)

SOFTWARE COMPONENTS ASSOCIATED WITH SPECIFIC FEATURES

<table>
<thead>
<tr>
<th>FEATURE NAME</th>
<th>REC ID</th>
<th>TYPE</th>
<th>SOFTWARE LEVEL</th>
<th>FIX LEVEL</th>
<th>FINANCIAL LEVEL</th>
<th>STATUS</th>
<th>DATE OF INSTALLATION</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCRNSVR</td>
<td>00000106</td>
<td>5</td>
<td>0002 0004 MVS3270</td>
<td>PLANNED</td>
<td>A NEW SCREEN SAVER WILL BE ADDED</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FILES</td>
<td>PPR</td>
<td>5</td>
<td>0002 0002 IMSR119</td>
<td>PCS</td>
<td>TEST</td>
<td>03/14/1998</td>
<td>FILES CONTAINING UPDATED CONFIG RECORDS</td>
<td></td>
</tr>
<tr>
<td>MVSA DISPLAY</td>
<td>5</td>
<td>0004 0002 IMSR118</td>
<td>PCS</td>
<td>INSTALL</td>
<td>03/13/1998</td>
<td>SCP CONFIGURATION RECORD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SCRNSVR</td>
<td>00000110</td>
<td>5</td>
<td>0002 0004 MVS3270</td>
<td>PLANNED</td>
<td>A NEW SCREEN SAVER WILL BE ADDED</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MVSA TPX</td>
<td>1</td>
<td>0003 0007 IFSD118</td>
<td>PCS</td>
<td>INSTALL</td>
<td>03/13/1998</td>
<td>TPX CONFIGURATION RECORD_FOR MVS SYSTEM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SCRNSVR</td>
<td>00000108</td>
<td>5</td>
<td>0002 0004 MVS3270</td>
<td>PLANNED</td>
<td>A NEW SCREEN SAVER WILL BE ADDED</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRDPRO APP</td>
<td>5</td>
<td>0002 0002 IDFR149</td>
<td>PCS</td>
<td>ORDER</td>
<td>03/14/1998</td>
<td>WORD PROCESSOR</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
RFT Examples

This appendix gives an example of creating an RFT, the Periodic Change Status report. Then it shows how you might modify the Periodic Change Status report to include risk-assessment totals. It includes several examples of RFTs you might create to do specific tasks. It also includes two examples of report exit routines.

Creating the Periodic Change Status Report

This scenario presents a building-block approach to creating a report. The first example begins by defining titles and headings for the report. The next example positions the output data in the proper columns. Subsequent examples calculate totals and build a summary line for the report. The examples are taken from the Tivoli Information Management for z/OS Periodic Change Status report (BLMZZ31).

Defining Titles and Headings

This example shows you how to define titles and headings.

The Periodic Change Status RFT consists of three period-processing sections for the previous, current, and next periods and a fourth End of File (EOF) processing section that prints cumulative totals for the report.

The RFT code in each of the period-processing sections searches for change-without-activity records and prints one line of data for each change record whose required date falls within the period.

As in all standard reports, actual data begins on the second page; titles and the search arguments appear on the first page.

The RFT code in the following example prints the headings for the previous-period section. The RFT statements for the current-period and next-period headings are almost identical to this. Look at the standard report in Figure 87 on page 291 to see how they are formatted. (In this example, three vertical dots replace statements to print the search argument and report processing statements. Column positions are adjusted here to allow the report to fit on a page.)

Example of Defining Titles and Headings

SECTION PRINT(N) SEPARATION(1) TEST(N)

/* *******************************************/
/* TITLES FOR: */
/* INFORMATION MANAGEMENT for z/OS PERIODIC CHANGE STATUS REPORT */
/* */
Creating the Periodic Change Status Report

TITLE

PUT COLUMN(001) VALUE(DATE &ZECDATE)

PUT COLUMN(032) VALUE(*** INFORMATION MANAGEMENT for z/OS PERIODIC CHANGE STATUS REPORT ***)

PUT COLUMN(132) VALUE(PAGE)
PUT COLUMN(128) VALUE(&ZPAGENO) LENGTH(4) JUSTIFY(R)

PUT COLUMN(001) VALUE(TIME &ZCTIME)

PUT COLUMN(041) VALUE(---------------------------------------------+
PUT COLUMN(123) VALUE(BLMZZ31)
SPACE LINES(2) EXECUTE(U)
ETITLE

 creating the periodic change status report

HEADING

PUT COLUMN(001) VALUE(PREVIOUS PERIOD: ''&ZEPPSD.'' THROUGH ' + 
'&ZEPPED.'')

PUT COLUMN(001) VALUE(---------------------------------------+

SPACE LINES(2) EXECUTE(C)

IF VALUE(&PREVCNT) OPERATOR(>) VALUE(0) /* IF RECORDS FOUND */

PUT COLUMN(001) VALUE(PREVIOUS PERIOD: ''&ZEPPSD.'' THROUGH ' + 
'&ZEPPED.'')

PUT COLUMN(001) VALUE(---------------------------------------+

SPACE LINES(2) EXECUTE(C)

IF VALUE(&PREVCNT) OPERATOR(>) VALUE(0) /* IF RECORDS FOUND */

PUT COLUMN(001) VALUE(---) /* DATE REQUIRED */
PUT COLUMN(013) VALUE(---) /* CHANGE NUMBER/ID */

PUT COLUMN(023) VALUE(---) /* CHANGE TYPE */
PUT COLUMN(034) VALUE(---) /* CHANGE, APP. STAT */

PUT COLUMN(045) VALUE(---) /* CURR PRI., PHASE */
PUT COLUMN(083) VALUE(---) /* ACTUAL IMPACT */

PUT COLUMN(001) VALUE(---) /* DATE REQUIRED */
PUT COLUMN(013) VALUE(---) /* CHANGE NUMBER/ID */

PUT COLUMN(083) VALUE(---) /* DESCRIPTION */

creating the periodic change status report

version 7.1
Example of Report

DATE 07/22/1998 *** INFORMATION MANAGEMENT for z/OS PERIODIC CHANGE STATUS REPORT *** PAGE 2
TIME 9:20:33 ------------------------------------------------------------- BLMZZ31


--------------------------------------------------
DATE CHANGE CHANGE STATUS CURRENT ACTUAL
REQUIRED NUMB/ID TYPE CHNG APP PRI PHASE RISK ASSIGNEE NAME IMPACT CHANGE DESCRIPTION
-------- -------- ------- --------- ---------- ----- --------------- ------ ------------------

Figure 87. Report Headings for Page 2 and Each Subsequent Page

Putting Data on an Output Line

The following example shows how the Periodic Change Status report prints a single line of data for each change record whose value for date required falls within the previous period.

The processing section for the previous period, which is shown here, includes a series of PUT statements that retrieve pieces of data from the record and place them in the specified columns. The RFT code for the current-period and the next-period sections is similar. (In this example, three vertical dots replace EOD processing statements. Column positions are adjusted here to allow the report to fit on a page.)

Example of RFT Code That Puts Data on an Output Line

```rft
/*********************/
/* PROCESSING FOR: */
/* PREVIOUS PERIOD */
/* */
/*********************/

IF VALUE(&ZIPPSD) OPERATOR(=) VALUE(&ZIFDATA)/* IF SECTION REQST
SEARCH ARGUMENT(!S0B06 ¬ !S0B07 DATD/&ZIPPSD - DATD/&ZIPPED)
EXECUTE(C) MAP(N) MERGE(0) SORT(DATD/, RNID/.)
/*********************/
/* DATA FOR: */
/* PREVIOUS PERIOD */
```
Calculating Totals for Each Report Section (EOD)

The Periodic Change Status report uses end-of-data (EOD) processing in each period section to calculate change-status and approval-status totals needed at the end of the report. The previous-period EOD processing is shown in the next example. (Note that the example shows the 4-digit year internal date format.) The RFT code for the current-period and next-period sections is similar.

Example of Calculating Totals

```plaintext
********************************************************************
/* COLLECT PREVIOUS PERIOD TOTALS */
********************************************************************

EOD EXECUTE(C) /* END OF DATA PROCESSING

SET NAME(SCOUNT) VALUE(1) /* SET SECTION PROCESSED
CALL NAME(BLGOXCNT) INPUT(DATD/2001/01/01 - DATD/&ZIPDATE. ¬ STAC/CL+
OSED,1,PREVODUE) LENGTH(0) /* OVERDUE CHANGES
   CALL NAME(BLGOXCNT) INPUT(¬ STAC/CLOSED,1,PREVOPN) +
LENGTH(0) /* OPEN CHANGES
```

Example of Report

```
DATE 07/22/1998 *** INFORMATION MANAGEMENT for z/OS PERIODIC CHANGE STATUS REPORT *** PAGE 2
TIME 9:20:33


<table>
<thead>
<tr>
<th>DATE</th>
<th>CHANGE</th>
<th>CHANGE STATUS</th>
<th>CURRENT ACTUAL</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>REQUIRED</td>
<td>NUMB/ID</td>
<td>TYPE</td>
</tr>
<tr>
<td>07/10/1998</td>
<td>ADD</td>
<td>OPEN</td>
<td>02</td>
</tr>
<tr>
<td>07/15/1998</td>
<td>REPLACE</td>
<td>OPEN</td>
<td>01</td>
</tr>
</tbody>
</table>
```
CALL NAME(BLGOXCNT) INPUT(STAC/CLOSED,1,PREVCOMP) +
LENGTH(0) /* COMPLETED CHANGES
CALL NAME(BLGOXCNT) INPUT(STAP/REJECTED,1,PREVREJ) +
LENGTH(0) /* REJECTED CHANGES
CALL NAME(BLGOXCNT) INPUT(STAP/PENDING,1,PREVPEND) +
LENGTH(0) /* PENDING CHANGES
CALL NAME(BLGOXCNT) INPUT(STAP/APPROVED,1,PREVAPP) +
LENGTH(0) /* APPROVED CHANGES
CALL NAME(BLGOXCNT) INPUT(!S0BF0,1,PREVNONE) +
LENGTH(0) /* NOT REQUIRED
EEO /* END END-OF-DATA
ESEARCH /* END PREVIOUS SEARCH

The EOD processing calculates totals for one of the three sections. The end-of-file (EOF) section accumulates totals provided by each EOD section. If multiple sections were processed, it totals all the processed sections and prints the values at the end of the report. The next example shows how the EOF code in the RFT prints the totals.

Building a Summary Line (EOF)

The Periodic Change Status report uses end-of-file (EOF) processing to place totals on the output line by a series of PUT and CALL commands. The BLGOXCNT (search and count) exit routine totals problems by priority. The EOF section determines whether any of the previous sections were processed. For each section processed, the EOF accumulates and prints the totals provided from each EOD. If multiple sections were processed, the EOF totals all the processed sections and prints the values at the end of the report. The following example shows the RFT code for building a summary line. Figure 88 on page 295 shows the output. The column positions in the output are adjusted to allow the report to fit on a page.

Example of Building a Summary Line

```*/
/****************************************************************************/
/* */
/* REPORT TOTALS */
/* */
/****************************************************************************/

SECTION PRINT(N) SEPARATION(1) TEST(N) /* TOTAL SECTION
HEADING /* DEFINE NULL HEADING
EHEADING /* END NULL HEADING
EOF EXECUTE(C)
. /* STATEMENTS TO PRINT */
. /* HEADINGS FOR */
. /* REPORT TOTALS */
IF VALUE(&PREVCNT) OPERATOR(>) VALUE(0) /* IF PREVIOUS RECS FOUND

/****************************************************************************/
/* */
/* OUTPUT PREVIOUS PERIOD TOTALS */
/* */
/****************************************************************************/

SPACE LINES(1) EXECUTE(C) /* SPACE 1 LINE
PUT VALUE(PREVIOUS: &ZEPPSD - &ZEPPED) COLUMN(5)
PUT COLUMN(044) LENGTH(06) VALUE(&PREVCNT) +
JUSTIFY(R) /* CHANGES REPORTED
PUT COLUMN(057) LENGTH(06) VALUE(&PREVODUE) +
```
Creating the Periodic Change Status Report

JUSTIFY(R) /* OVERDUE CHANGES
PUT COLUMN(068) LENGTH(06) VALUE(&PREVOPN) +

JUSTIFY(R) /* OPEN CHANGES
PUT COLUMN(079) LENGTH(06) VALUE(&PREVCOMP) +

JUSTIFY(R) /* COMPLETE CHANGES
PUT COLUMN(092) LENGTH(06) VALUE(&PREVREJ) +

JUSTIFY(R) /* REJECTED CHANGES
PUT COLUMN(103) LENGTH(06) VALUE(&PREVPEND) +

JUSTIFY(R) /* PENDING CHANGES
PUT COLUMN(114) LENGTH(06) VALUE(&PREVAPP) +

JUSTIFY(R) /* APPROVED CHANGES
PUT COLUMN(125) LENGTH(06) VALUE(&PREVNONE) +

JUSTIFY(R) /* NOT REQUIRED

****************************************************************************
/* */
/* ACCUMULATE PREVIOUS PERIOD TOTALS */
/* */
****************************************************************************

SET NAME(TOTCNT) VALUE(&TOTCNT) OPERATOR(+) VALUE(&PREVCNT)
SET NAME(TOTODUE) VALUE(&TOTODUE) OPERATOR(+) VALUE(&PREVODUE)
SET NAME(TOTOPN) VALUE(&TOTOPN) OPERATOR(+) VALUE(&PREVOPN)
SET NAME(TOTCOMP) VALUE(&TOTCOMP) OPERATOR(+) VALUE(&PREVCOMP)
SET NAME(TOTREJ) VALUE(&TOTREJ) OPERATOR(+) VALUE(&PREVREJ)
SET NAME(TOTPEND) VALUE(&TOTPEND) OPERATOR(+) VALUE(&PREVPEND)
SET NAME(TOTAPP) VALUE(&TOTAPP) OPERATOR(+) VALUE(&PREVAPP)
SET NAME(TOTNONE) VALUE(&TOTNONE) OPERATOR(+) VALUE(&PREVNONE)

EIF /* END PREVIOUS RECORDS
/* EOF PROCESSING */
/* FOR CURRENT AND */
/* NEXT PERIODS */
IF VALUE(&SCOUNT) OPERATOR(>) VALUE(1) /* IF MULTIPLE SECTIONS*/
PRINT REPORT TOTALS

PUT COLUMN(044) VALUE(------) /* UNDERSCORES
PUT COLUMN(057) VALUE(------)
PUT COLUMN(068) VALUE(------)
PUT COLUMN(079) VALUE(------)
PUT COLUMN(092) VALUE(------)
PUT COLUMN(103) VALUE(------)
PUT COLUMN(114) VALUE(------)
PUT COLUMN(125) VALUE(------)
SPACE LINES(1) EXECUTE(C) /* SPACE 1 LINE

PUT COLUMN(044) LENGTH(06) VALUE(&TOTCNT) +
JUSTIFY(R) /* CHANGES REPORTED
PUT COLUMN(057) LENGTH(06) VALUE(&TOTODUE) +
JUSTIFY(R) /* OVERDUE CHANGES
PUT COLUMN(068) LENGTH(06) VALUE(&TOTOPN) +
JUSTIFY(R) /* OPEN CHANGES
PUT COLUMN(079) LENGTH(06) VALUE(&TOTCOMP) +
JUSTIFY(R) /* COMPLETE CHANGES
PUT COLUMN(092) LENGTH(06) VALUE(&TOTREJ) +
JUSTIFY(R) /* REJECTED CHANGES
PUT COLUMN(103) LENGTH(06) VALUE(&TOTPEND) +
JUSTIFY(R) /* PENDING CHANGES

Hayden Corporation

Creating the Periodic Change Status Report

Page 294
Version 7.1
Example of Report (with summary line output)

<table>
<thead>
<tr>
<th>REPORT PERIOD</th>
<th>CHANGES REPORTED</th>
<th>OVERDUE</th>
<th>OPEN</th>
<th>COMPLETE</th>
<th>REJECTED</th>
<th>PENDING</th>
<th>APPROVED</th>
<th>NOT REQ'ED</th>
</tr>
</thead>
<tbody>
<tr>
<td>PREVIOUS: 07/09/1998 - 07/15/1998</td>
<td>4</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>CURRENT: 07/16/1998 - 07/22/1998</td>
<td>2</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>NEXT: 07/23/1998 - 07/29/1998</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>REPORT TOTALS:</td>
<td>7</td>
<td>1</td>
<td>5</td>
<td>2</td>
<td>1</td>
<td>4</td>
<td>2</td>
<td>0</td>
</tr>
</tbody>
</table>

Figure B8. Summary Line Output

Modifying a Standard RFT

In this example, assume you want a report that indicates the total number of low-, medium-, and high-risk change-without-activity records. Assume also that you want the report to include subtotals for the previous, current, and next periods. One possible way to satisfy these requirements is to modify the Periodic Change Status report to calculate risk-assessment totals instead of status totals.

Assume also that you are saving this RFT as member BLMRISK.

The Periodic Change Status RFT consists of three period-processing sections for the previous, current, and next periods and a fourth end-of-file (EOF) processing section that prints cumulative totals for the report.

In the standard RFT, end-of-data (EOD) processing in each period section calculates change-status and approval-status totals for the section. The RFT code for the previous-period sections is shown on page 291, 292, and 293.

First, change the line that writes the RFT member name on the report. That change is shown like this in the following example.

Changing the RFT Member Name

SECTION PRINT(N) SEPARATION(1) TEST(N)

*************** INFORMATION MANAGEMENT for z/OS PERIODIC CHANGE STATUS REPORT

TITLE /* REPORT TITLES */

PUT COLUMN(901) VALUE(DATE &EZCDATE) /* REPORT DATE */

PUT COLUMN(932) VALUE(**** INFORMATION MANAGEMENT for z/OS PERIODIC CHANGE ST+...
Next, change the three CALL statements in the EOD processing section for each period processing section that collects status data. These CALL statements use the BLGOXCNT (search and count) exit routine to calculate the total number of low-, medium-, and high-risk changes whose required date falls within the period.

The code to collect the totals for the previous period is shown in the following example. Statements that change the standard report format are shown like this.

**Adding Risk-Assessment Totals**

```plaintext
/***************************************************************************/
/*! */
/*! COLLECT PREVIOUS PERIOD TOTALS */
/*! */

******************************************************************************/

EOD EXECUTE(C) /* END OF DATA PROCESSING */

SET NAME(SCOUNT) VALUE(1) /* SET SECTION PROCESSED */

CALL NAME(BLGOXCNT) INPUT(IMPR/LOW,1,PREVLLOW) LENGTH(0) /* LOW RISK */

CALL NAME(BLGOXCNT) INPUT(IMPR/MEDIUM,1,PREVMED) LENGTH(0) /* MEDIUM RISK */

CALL NAME(BLGOXCNT) INPUT(IMPR/HIGH,1,PREVHIGH) LENGTH(0) /* HIGH RISK */

CALL NAME(BLGOXCNT) INPUT(STAP/REJECTED,1,PREVREJ) LENGTH(0) /* NOT REQUIRED */

CALL NAME(BLGOXCNT) INPUT(STAP/PENDING,1,PREVPEND) LENGTH(0) /* PENDING CHANGES */

CALL NAME(BLGOXCNT) INPUT(STAP/APPROVED,1,PREVAPP) LENGTH(0) /* APPROVED CHANGES */

CALL NAME(BLGOXCNT) INPUT('IS0BF0',1,PREVNONE) LENGTH(0) /* NOT REQUIRED */

EEOD /* END END-OF-DATA */

ESEARCH /* END PREVIOUS SEARCH */
```

The CALL statements to collect risk-assessment totals for the current-period and next-period EOD processing are not shown; they are essentially identical to those added to the previous-period processing.
Last, change the fourth section of the standard report that prints the change-status and approval-status totals. Change those statements in the EOF processing section that print the risk-assessment totals. In the following example, the lines you need to change in the standard report are shown like this. The corresponding changes for the current and next periods are not shown because they are essentially identical to those changed in the previous-period processing.

**Example of Code for Previous-Period Totals and Cumulative Totals**

```c
/***********************************************************************/
/*                                                             */
/* REPORT TOTALS                                              */
/*                                                             */
/***********************************************************************/
SECTION PRINT(N) SEPARATION(1) TEST(N) /* TOTAL SECTION
HEADING /* DEFINE NULL HEADING
EHEADING /* END NULL HEADING
EOF EXECUTE(C)
IF VALUE(&SCOUNT) OPERATOR(>) VALUE(0) /* IF ANY SECT PROCESSED
SPACE LINES(4) EXECUTE(C) MINLINES(17) /* SPACE 4 LINES
PUT COLUMN(043) VALUE(CHANGES) /* CHANGES REPORTED
PUT COLUMN(056) VALUE(**** RISK ASSESSMENT ****)
PUT COLUMN(091) VALUE(*************** APPROVAL **************)
PUT COLUMN(005) VALUE(REPORT PERIOD) /* REPORT PERIOD
PUT COLUMN(043) VALUE(REPORTED) /* CHANGES REPORTED
PUT COLUMN(056) VALUE(LOW) /* LOW RISK
PUT COLUMN(068) VALUE(MEDIUM) /* MEDIUM RISK
PUT COLUMN(080) VALUE(HIGH) /* HIGH RISK
PUT COLUMN(091) VALUE(REJECTED) /* REJECTED CHANGES
PUT COLUMN(102) VALUE(PENDING) /* PENDING CHANGES
PUT COLUMN(113) VALUE(APPROVED) /* APPROVED CHANGES
PUT COLUMN(124) VALUE(NOT REQ’ED) /* NOT REQUIRED

PUT COLUMN(005) VALUE(---------------------------------)
PUT COLUMN(043) VALUE(--------) /* CHANGES REPORTED
PUT COLUMN(056) VALUE(--------) /* LOW RISK
PUT COLUMN(067) VALUE(--------) /* MEDIUM RISK
PUT COLUMN(078) VALUE(--------) /* HIGH RISK
PUT COLUMN(091) VALUE(--------) /* REJECTED CHANGES
PUT COLUMN(102) VALUE(--------) /* PENDING CHANGES
PUT COLUMN(113) VALUE(--------) /* APPROVED CHANGES
PUT COLUMN(124) VALUE(--------) /* NOT REQUIRED

IF VALUE(&PREVCNT) OPERATOR(>) VALUE(0) /* IF PREVIOUS RECS FOUND
/*****************************/
/*                        */
/* OUTPUT PREVIOUS PERIOD TOTALS                        */
/*                        */
/*****************************/
SPACE LINES(1) EXECUTE(C) /* SPACE 1 LINE
PUT VALUE(PREVIOUS: &ZEPPSD - &ZEPPED) COLUMN(5)

PUT COLUMN(044) LENGTH(06) VALUE(&PREVCNT) +
                JUSTIFY(R) /* CHANGES REPORTED
PUT COLUMN(057) LENGTH(06) VALUE(&PREVLOW) +
                JUSTIFY(R) /* LOW RISK
```
PUT COLUMN(068) LENGTH(06) VALUE(&PREVMED) +
JUSTIFY(R)    /* MEDIUM RISK

PUT COLUMN(079) LENGTH(06) VALUE(&PREVHIGH) +
JUSTIFY(R)    /* HIGH RISK

PUT COLUMN(092) LENGTH(06) VALUE(&PREVREJ) +
JUSTIFY(R)    /* REJECTED CHANGES

PUT COLUMN(103) LENGTH(06) VALUE(&PREVPEND) +
JUSTIFY(R)    /* PENDING CHANGES

PUT COLUMN(114) LENGTH(06) VALUE(&PREVAPP) +
JUSTIFY(R)    /* APPROVED CHANGES

PUT COLUMN(125) LENGTH(06) VALUE(&PREVNONE) +
JUSTIFY(R)    /* NOT REQUIRED

*********************************************************************/
/*
/* ACCUMULATE PREVIOUS PERIOD TOTALS */
/*
*********************************************************************/

SET NAME(TOTCNT) VALUE(&TOTCNT) OPERATOR(+) VALUE(&PREVCNT)

SET NAME(TOTLOW) VALUE(&TOTLOW) OPERATOR(+) VALUE(&PREVLOW)
SET NAME(TOTMED) VALUE(&TOTMED) OPERATOR(+) VALUE(&PREVMED)

SET NAME(TOTHIGH) VALUE(&TOTHIGH) OPERATOR(+) VALUE(&PREVHIGH)
SET NAME(TOTREJ) VALUE(&TOTREJ) OPERATOR(+) VALUE(&PREVREJ)

SET NAME(TOTPEND) VALUE(&TOTPEND) OPERATOR(+) VALUE(&PREVPEND)
SET NAME(TOTAPP) VALUE(&TOTAPP) OPERATOR(+) VALUE(&PREVAPP)

SET NAME(TOTNONE) VALUE(&TOTNONE) OPERATOR(+) VALUE(&PREVNONE)
EIF    /* END PREVIOUS RECORDS

IF VALUE(&CURRCNT) OPERATOR(>) VALUE(0)    /* IF CURRENT RECS FOUND

    /*
    /* OUTPUT CURRENT PERIOD TOTALS */
    /*
    /* ACCUMULATE CURRENT PERIOD TOTALS */
    /*
    /* OUTPUT NEXT PERIOD TOTALS */
    /*
    /* ACCUMULATE NEXT PERIOD TOTALS */
    /*
    EIF    /* END CURRENT RECORDS

    /* UNDERSORES
    */

    PUT COLUMN(057) LENGTH(06) VALUE(&TOTLOW) +
JUSTIFY(R)    /* LOW RISK

    PUT COLUMN(068) LENGTH(06) VALUE(&TOTMED) +
JUSTIFY(R)    /* MEDIUM RISK

    PUT COLUMN(079) LENGTH(06) VALUE(&TOTHIGH) +
JUSTIFY(R)    /* HIGH RISK

    PUT COLUMN(092) LENGTH(06) VALUE(&TOTREJ) +
JUSTIFY(R)    /* REJECTED CHANGES

    PUT COLUMN(103) LENGTH(06) VALUE(&TOTPEND) +
JUSTIFY(R)    /* PENDING CHANGES

    PUT COLUMN(114) LENGTH(06) VALUE(&TOTAPP) +
JUSTIFY(R)    /* APPROVED CHANGES

    PUT COLUMN(125) LENGTH(06) VALUE(&TOTNONE) +
JUSTIFY(R)    /* NOT REQUIRED

*********************************************************************/
/*
/* ACCUMULATE NEXT PERIOD TOTALS */
/*
*********************************************************************/
The report output looks like that shown in the following example.

Example of Report

*** INFORMATION MANAGEMENT for z/OS PERIODIC CHANGE STATUS REPORT ***

DATE 07/08/1998       PAGE 2
TIME 11:04:33       BLMRISK

REPORT PERIOD       CHANGES REPORTED       **** RISK ASSESSMENT ****
PREVIOUS: 06/25/1998 - 07/01/1998  12       7       4       1
CURRENT: 07/02/1998 - 07/08/1998  8       4       2       2
NEXT: 07/09/1998 - 07/15/1998  14       7       6       1

REPORT TOTALS: 34 18 12 4

Figure 89. Report with Risk-Assessment Totals

Additional Examples

This section contains RFTs you can create for specific tasks. You can use these RFTs as guides when you create your own User RFT reports.

Example 1: Identifying Changes and Related Activities

The RFT illustrated in the next example identifies and prints the record IDs of changes that have activity, followed by the activities. It consists of one report section with 10 statements. The numbers on the right are sequence numbers, referenced by the comments in Table 22 on page 300. In an actual RFT, the sequence numbers appear in columns 73 through 80. The column positions in the following example were adjusted so that the sequence numbers appear on the same line as the statement.

RFT to Print Record IDs of Changes with Activity

SECTION
SEARCH ARGUMENT(ISOB06 ~ ISOB07 ISOCBC) SORT(RNID/.) MERGE(0) 00000001
SET NAME(CHANGREC) DATA(RNID/.) 00000002
PUT VALUE(ACTIVITIES FOR RECORD &CHANGREC) COLUMN(1) 00000003
SEARCH ARGUMENT(RNOR/CHANGREC ISOB07) MERGE(N) 00000004
PUT DATA(RNID/.) COLUMN(1) 00000005
ESEARCH 00000006
EJECT 00000007

---

Data Reporting User’s Guide

299
Table 22. Explanations of Sequence Numbers for an RFT

<table>
<thead>
<tr>
<th>Sequence Number</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>00000001</td>
<td>Defines the beginning of a section and causes the Report Format Facility to read all statements in the section.</td>
</tr>
<tr>
<td>00000002</td>
<td>Performs a level-1 search for change records (!S0B06), excluding activity records (!S0B07) that have activity records (S0CBC) associated with them.</td>
</tr>
<tr>
<td>00000003</td>
<td>Sets the variable, CHANGREC, to the change record ID.</td>
</tr>
<tr>
<td>00000004</td>
<td>Substitutes the record ID for &amp;CHANGREC and prints the line.</td>
</tr>
<tr>
<td>00000005</td>
<td>Performs a level-2 search to retrieve activities (!S0B07) that have a parent change ID equal to &amp;CHANGREC. The MERGE(N) keyword indicates a search for all activities.</td>
</tr>
<tr>
<td>00000006</td>
<td>Prints the record ID for each activity record retrieved by the level-2 search.</td>
</tr>
<tr>
<td>00000007</td>
<td>Delimits the statements that are processed for the level-2 search.</td>
</tr>
<tr>
<td>00000008</td>
<td>Causes a page eject after all the activities returned by the level-2 search have been processed. In effect, each change and its associated activities start on a new page.</td>
</tr>
<tr>
<td>00000009</td>
<td>Delimits the commands processed for the level-1 search. The next change record retrieved by the search is processed.</td>
</tr>
<tr>
<td>00000010</td>
<td>Delimits the statements that are part of this report section.</td>
</tr>
</tbody>
</table>

For each change, a report page looks like the one illustrated in Figure 90.

Example of Report

ACTIVITIES FOR CHANGE RECORD 00000783
00000805
00000806
00000807

Figure 90. Output RFT to Print Record IDs of Changes with Activity

Example 2: Problem Record Description

In this example, assume you want a report that presents one line of specific data for each problem record. Also, assume that you want each record to meet the level-0 search argument that you can specify on the Tivoli Information Management for z/OS REPORT command.

The RFT shown in the next example produces the required report. The report output is shown in Figure 91 on page 302.

Problem Record Description RFT

```
SECTION TEST(N) PRINT(N)
/**************************************************************************/
/* TITLES FOR: */
/* PROBLEM DESCRIPTION LIST REPORT */
/* */
Additional Examples

Data Reporting User’s Guide 301
Example of Report

*** PROBLEM RECORD DESCRIPTION REPORT ***

--OCCURRED--

SYSTEM DATE TIME OUTAGE PROBLEM# DEPT ASSIGNEE

---------- --------- ------ -------- -------- ---- --------

MVS1 06/19/1998 09:05 00003987 165 CRAWFORD

GRAPHICS WON'T PRINT PROPERLY

-----------------------------------------------------------------------

MVS1 06/27/1998 15:21 00003916 226 JAMES

GENERATES CODE THAT IS NOT VALID FOR IF STATEMENT

-----------------------------------------------------------------------

Figure 91. Problem Record Description Output

Example 3: Determining Problems Entered Overnight

Assume you want a report that shows all problems entered overnight. “Overnight” is defined in this report as from 7:00 p.m. yesterday through 7:00 a.m. today. This combination of logic cannot be represented in a Tivoli Information Management for z/OS search argument. However, the Report Format Facility makes it possible by performing two searches: one to retrieve problems entered yesterday and one to retrieve problems entered today.

The following RFT produces the required report. Figure 92 on page 304 illustrates the report.

Example of Determining Problems Entered Overnight RFT

SECTION SEP(2)

TITLE

PUT C(001) VALUE(DATE &ZECDATE) /* REPORT DATE */
PUT C(019) VALUE(INFORMATION MANAGEMENT for z/OS OVERNIGHT PROBLEM REPORT)

PUT C(071) VALUE(PAGE)
PUT C(076) VALUE(AMPAGENO) JUSTIFY(R) LENGTH(5) /* PAGE NUMBER */

PUT C(001) VALUE(TIME &ZCTIME) /* REPORT TIME */
SPACE LINES(2)
Example of Report
Example 4: Overcoming Data Output Problems

This example presents an RFT coding problem you may experience and shows how you can avoid this problem.

In general, data is placed on the current output line, as long as the starting column (as specified or defaulted on the PUT and CALL command) is at or beyond the current-column pointer.

If you do not use the SPACE command to force the output line to be written, you could experience undesired results when dealing with titles and headings. The examples that follow illustrate how this can happen and how it can be avoided.

Assume that you want a report that presents a single line of data for each assigned problem record that is not closed. The records matching the search argument are sorted by department assignee. In addition, the report is to include a totals line for each department.

Assume that you want the RFT to print headings at the top of each page of the report, unless the page contains only totals; in that case, do not print headings.

RFT 1 prints the headings incorrectly because of errors in the RFT code. RFT 2 shows how to print the headings correctly.

**RFT 1—Incorrect**

In this RFT, an IF command statement is used to determine whether headings should be for the page.

**Example of Overcoming Data Output Problems (RFT1—Incorrect)**

```
SECTION SEPARATION(1) TEST(N) PRINT(N) /* ASSIGNEE SECTION */

HEADING /* SECTION HEADINGS */
  IF VALUE(&DEPTNM) OPERATOR(=) DATA(GROA/.) /* IF HEADINGS NEEDED */
  PUT COLUMN(001) VALUE(ASSIGNEE DEPT - &DEPTNM.) /* ADDITIONAL HEADING */
  /* STATEMENTS */
  .
  EIF /* END HEADINGS NEEDED */
EHEADING /* END HEADING DEF */
```

Figure 92. Overnight Problem Report Output

---

**Example 4: Overcoming Data Output Problems**

This example presents an RFT coding problem you may experience and shows how you can avoid this problem.

In general, data is placed on the current output line, as long as the starting column (as specified or defaulted on the PUT and CALL command) is at or beyond the current-column pointer.

If you do not use the SPACE command to force the output line to be written, you could experience undesired results when dealing with titles and headings. The examples that follow illustrate how this can happen and how it can be avoided.

Assume that you want a report that presents a single line of data for each assigned problem record that is not closed. The records matching the search argument are sorted by department assignee. In addition, the report is to include a totals line for each department.

Assume that you want the RFT to print headings at the top of each page of the report, unless the page contains only totals; in that case, do not print headings.

RFT 1 prints the headings incorrectly because of errors in the RFT code. RFT 2 shows how to print the headings correctly.

**RFT 1—Incorrect**

In this RFT, an IF command statement is used to determine whether headings should be for the page.

**Example of Overcoming Data Output Problems (RFT1—Incorrect)**

```
SECTION SEPARATION(1) TEST(N) PRINT(N) /* ASSIGNEE SECTION */

HEADING /* SECTION HEADINGS */
  IF VALUE(&DEPTNM) OPERATOR(=) DATA(GROA/.) /* IF HEADINGS NEEDED */
  PUT COLUMN(001) VALUE(ASSIGNEE DEPT - &DEPTNM.) /* ADDITIONAL HEADING */
  /* STATEMENTS */
  .
  EIF /* END HEADINGS NEEDED */
EHEADING /* END HEADING DEF */
```

Figure 92. Overnight Problem Report Output
This first RFT works for all situations except when a department has only one record. When this occurs, the RFT does not produce a heading for the new department. If the database search returns a single record for a department and then a record for a different department, the RFT processing is as follows:

1. When the SEARCH command returns the first record, the &DEPTNM variable has not been set and the first IF following the SEARCH command is true. The second IF is false, because the &TOTPROB variable was initialized to 0. The EJECT command positions the output line counter at the bottom of the page. The &DEPTNM variable is set and tests true on the third IF, so the data from the record is moved to the output line and the &TOTPROB variable is set to one.

2. When the SEARCH command returns the next record, the department is different from the preceding one. Therefore, the first IF following the SEARCH command is again true. This time the second IF is also true. When the SPACE command is found, an attempt is made to write the data for the first record that is still in the output line to the output data set. Because the previous EJECT command positions the output line at the bottom of the page, a page eject occurs and the titles are written on the top of the page. The IF command in the headings is false, so no headings are produced.
The data for the first record is then written on the new page and the totals are moved to the output line.

The EJECT statement forces the totals to be written to the output data set and again positions the output line counter at the bottom of the page.

You can correct this failure to produce headings for departments with a single record in several ways. One way is to add a SPACE statement. This method is shown like this in the RFT in the following example.

**RFT 2—Correct**

**Example of Overcoming Data Output Problems (RFT2—Correct)**

```
SECTION SEPARATION(1) TEST(N) PRINT(N) /* ASSIGNEE SECTION */

HEADING /* SECTION HEADINGS */
IF VALUE(&DEPTNM) OPERATOR(=) DATA(GROA/.) /* IF HEADINGS NEEDED */
  PUT COLUMN(001) VALUE(ASSIGNEE DEPT - &DEPTNM.) /* ADDITIONAL HEADER */
  /* STATEMENTS */
EIF /* END HEADINGS NEEDED */
EHEADING /* END HEADING DEF */

SET NAME(TOTPROB) VALUE(0) /* INITIALIZE COUNTER */
SEARCH ARGUMENT(!S0B01 !S0B9C ¬ STAC/CLOSED) +
  SORT(GROA/. RNID/. ) EXECUTE(C) MERGE(0) MAP(N)

IF VALUE(&DEPTNM); OPERATOR(¬=) DATA(GROA/.) /* IF IT'S A NEW DEPT */
  IF VALUE(&TOTPROB) OPERATOR(>) VALUE(0) /* IF RECORDS PROCESSED */
    SPACE LINES(2) EXECUTE(C) /* SPACE 2 LINES */
    PUT COLUMN(001) VALUE(TOTAL RECORDS = &TOTPROB.)
  EIF /* END RECS PROCESSED */
EJECT
SET NAME(DEPTNM) DATA(GROA/.) /* SET DEPT NAME */
SET NAME(TOTPROB) VALUE(0) /* RESET COUNTER */
EIF

IF VALUE(&DEPTNM) OPERATOR(=) DATA(GROA/.)
  IF VALUE(&TOTPROB) OPERATOR(+) VALUE(1)
    PUT COLUMN(001) LENGTH(08) DATA(RNID/.)
  EIF /* ADDITIONAL DATA */
  /* STATEMENTS */

SPACE LINES(-1) /* WRITE OUTPUT BUFFER */
EIF /* END TOTAL PROCESSING */
EOD EXECUTE(C) /* PROCESS TOTALS */
IF VALUE(&TOTPROB) OPERATOR(>) VALUE(0) /* IF RECORDS PROCESSED */
  SPACE LINES(2) EXECUTE(C)
  PUT COLUMN(001) VALUE(TOTAL RECORDS = &TOTPROB.)
```

**Additional Examples**
Because the SPACE command in this RFT forces the data to be written before the next record is retrieved, the IF in the headings is true and the heading is produced for single records.

**Example 5: Identifying Problems That Exceed Outage Standards (Data Output Exit)**

In this example, assume you want a report that prints a single line of data for priority 1 problems with at least one hour of outage and priority 2 problems with at least three hours of outage.

This example uses both an RFT and a data output exit routine. The RFT uses the data output exit routine to control which records are to be reported. The RFT formats the data and produces the report. The exit routine is shown in the following example; the RFT is shown on page 303, and the report is shown in Figure 93 on page 310.

The exit routine determines whether a problem is priority 1 or 2. If it is priority 1 and the outage is less than one hour, the record is not written to the report data set. Likewise, if it is a priority 2 problem and the outage is less than three hours, the record is not written to the report data set.

**Exit Routine for Problem Exception Report**

```assembly
TITLE 'PROBEXIT – PROBLEM EXCEPTION REPORT EXIT'
PROBEXIT CSECT

USING PROBEXIT,R15
STM R14,R12,12(R13) /* SAVE THE CALLER'S REGISTERS */
B START /* BRANCH AROUND EYECATCHER */
DC CL8'PROBEXIT'

START EQU *
LR R12,R15 /* SET UP BASE REGISTER */
DROP R15
USING PROBEXIT,R12
L R8,0(R1) /* GET PARAMETER LIST */
USING PLIST,R8
ST R13,SAVEAREA+4 /* SAVE CALLER'S SAVE AREA POINTER */
LA R15,SAVEAREA /* GET OUR SAVE AREA ADDRESS */
ST R15,B(R13) /* SAVE OUT ADDRESS IN HIS SAVE AREA */
LR R13,R15 /* ESTABLISH OUT SAVE AREA REGISTER */

SPACE 2
CLC OBUFLEN,MINLEN /* IS THIS A LINE TO BE CHECKED */
BL EXIT0 /* NO, PRINT IT */
L R2,OUTBUFF /* GET OUTPUT BUFFER ADDRESSABILITY */

CLC 22(2,R2),PRI01 /* IS IT PRIORITY 1 */
BE PRIOR1 /* PROCESS IT */

CLC 22(2,R2),PRI02 /* IS IT PRIORITY 2 */
BNE EXIT4 /* NO, DON'T PRINT IT */

CLC 26(2,R2),BLANK /* IF THERE IS NO OUTAGE */
```
BE EXIT4 /* DON'T PRINT THE RECORD */

CLC 26(2,R2),ZERO /* IF OUTAGE WAS IN DAYS */
BNE EXIT0 /* PRINT THE RECORD */

CLC 29(2,R2),THREE /* IF OUTAGE LESS THAN THREE HOURS */
BL EXIT4 /* DON'T PRINT THE RECORD */

B EXIT0 /* OTHERWISE, PRINT IT */

PRIOR1 CLC 26(2,R2),ZERO /* IF OUTAGE WAS NOT IN DAYS */
BE CHKTIME /* CHECK THE OUTAGE TIME */

CLC 26(2,R2),BLANK /* ELSE IF THERE IS NO OUTAGE */
BE EXIT4 /* DON'T PRINT THE RECORD */
B EXIT0 /* OUTAGE IS IN DAYS, PRINT IT */

CHKTIME CLC 29(2,R2),ONE /* IF OUTAGE WAS 1 HOUR OR MORE */
BNL EXIT0 /* PRINT THE RECORD */

CLC 32(2,R2),SIXTY /* IF OUTAGE WAS 1 HOUR OR MORE */
BNL EXIT0 /* PRINT THE RECORD */

EXIT4 L R13,SAVEAREA+4 /* RESTORE SAVE AREA POINTER */
LM R14,R12,12(R13) /* RESTORE CALLER'S REGISTERS */
LA R15,4 /* SET DON'T PRINT RETURN CODE */
BR R14 /* RETURN TO CALLER */

EXIT0 L R13,SAVEAREA+4 /* RESTORE SAVE AREA POINTER */
LM R14,R12,12(R13) /* RESTORE CALLER'S REGISTERS */
SLR R15,R15 /* ZERO REGISTER 15 */
BR R14 /* RETURN TO CALLER */

EJECT

**********************************************************************
* CONSTANT AREA *
**********************************************************************

SPACE 1

SAVEAREA DC 18F'0'

MINLEN DC F'34' /* MINIMUM LINE LENGTH */
SIXTY DC CL2'60' /* SIXTY MINUTES */
THREE DC CL2'03' /* THREE HOURS */
ONE DC CL2'01' /* ONE HOUR */
ZERO DC CL2'00' /* ZERO DAYS */
BLANK DC CL2' ' /* OUTAGE NOT LISTED IN RECORD */
PRI01 DC CL2'01' /* PRIORITY 1 */
PRI02 DC CL2'02' /* PRIORITY 2 */

SPACE 3

**********************************************************************
* REGISTER EQUATES *
**********************************************************************

SPACE 1

R0 EQU 00
R1 EQU 01
R2 EQU 02
R3 EQU 03
R4 EQU 04
R5 EQU 05
R6 EQU 06
The RFT shown in the next example passes control to the Problem Exception Report Exit routine before it writes data to the output data set.

**RFT for Problem Exception Report**

```plaintext
SECTION NAME(PROBEXIT) /* PROBLEM EXCEPTION REPORT */
TITLE /* REPORT TITLE */
    PUT VALUE(PROBLEM EXCEPTION REPORT) COLUMN(28)
ETITLE

SPACE /* REPORT HEADING */
    PUT COLUMN(001) VALUE(PROBLEM)
    PUT COLUMN(011) VALUE(DATE)
    PUT COLUMN(023) VALUE(CUR)
    PUT COLUMN(001) VALUE(NUMB/ID)
    PUT COLUMN(011) VALUE(ENTERED)
    PUT COLUMN(023) VALUE(PRI)
    PUT COLUMN(028) VALUE(OUTAGE)
    PUT COLUMN(038) VALUE(PROBLEM DESCRIPTION)
    PUT COLUMN(001) VALUE(------- ---------- --- -------- ---------+
        ------------------------------------)
    SPACE LINES(1)

EHEADING
SEARCH ARGUMENT(!S0B01) /* GET ALL PROBLEM RECORDS */
```

---

Additional Examples

---

B. RFT Examples
Example of Report

PROBLEM EXCEPTION REPORT

<table>
<thead>
<tr>
<th>PROBLEM</th>
<th>DATE ENTERED</th>
<th>CUR PRI</th>
<th>OUTAGE</th>
<th>PROBLEM DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROBT127</td>
<td>08/24/1996</td>
<td>01</td>
<td>00:02:30</td>
<td>TIMEOUT MESSAGES ON ATLANTA LINES</td>
</tr>
<tr>
<td>PROBT136</td>
<td>08/28/1996</td>
<td>02</td>
<td>01:12:00</td>
<td>ABEND 0C2 IN TSO EDIT</td>
</tr>
</tbody>
</table>

Figure 93. Output from Problem Exception Report

Example 6: Identifying Problems with Three or More Assignees (CALL Command Exit)

In this example, assume you want a report that identifies all problems that have been assigned at least three times and are not closed. One possible way to satisfy these requirements is to use a CALL command exit to find the problems that have been assigned at least three times.

The exit routine shown in this next example disregards problems with one or two assignments. It returns the rest of the problems to the RFT, shown on page 313. The output from the RFT is shown in Figure 94 on page 314.

Exit Routine for Identifying Problems with Three or More Assignees

```assembly
TITLE 'UEXIT – USER CALL COMMAND EXIT ROUTINE'
UEXIT CSECT
    USING UEXIT,R15
    STM R14,R12,12(R13) /* SAVE THE CALLERS REGISTERS */
    B START /* BRANCH AROUND EYECATCHER */
    DC CL8'UEXIT' /* PUT IN MODULE NAME */
    LR R12,R15 /* SET UP BASE REGISTER */
    LA R13,SAVEAREA+4 /* SAVE CALLERS SAVE AREA ADDRESS */
    ST R15,SAVEAREA /* GET OUR SAVE AREA ADDRESS */
    ST R15,8(R13) /* SAVE OUT ADDRESS IN HIS SAVE AREA */
    LR R13,R15 /* ESTABLISH OUT SAVE AREA REGISTER */
```

Additional Examples
SPACE 2
L 4,SXTEEN /* RCODE=16 FOR COMPARISON */
SLR 5,5 /* RCODE=0 FOR COMPARISON */
SLR 2,2 /* INITIALIZE # OF PREFIXES FOUND */
MVC P07,FIRST /* INDICATE THAT THIS IS 1ST TIME THRU */
MVC P05,LOCATE /* INDICATE LOCATE FUNCTION IS REQUESTED */
MVC P09(1),BLANK /* MOVE A BLANK INTO P09 */
MVC P09+1(15),P09 /* BLANK OUT ENTIRE FIELD */
MVC P09(6),PERA /* MOVE IN PREFIX TO BE FOUND */
L 15,P01 /* SET UP TO CALL ROUTER */
BALR 14,15 /* CALL LOCATE FUNCTION PROCESSOR */
CR 15,4 /* IF RETURN CODE IS 16 */
BE SET /* BRANCH OUT OF LOOP */
CR 15,5 /* IF RETURN CODE IS ZERO */
BE INCR /* THE PREFIX WAS FOUND. PROCESS IT */
SLR 15,15 /* ELSE RESET RETURN CODE */
B SET /* AND BRANCH OUT OF LOOP */
INCR A 2,ONE /* ADD 1 TO PREFIX FOUND COUNT */
MVC P07,NEXT /* TURN OFF FIRST TIME THRU SWITCH */
B LOOP /* LOOK FOR ANOTHER PREFIX */
SET CVD 2,WK1 /* CONVERT THE NUMBER OF PREFIXES FOUND */
UNPK WK2(4),WK1(8) /* UNPACK IT */
MVZ WK2+3(1),ZONE /* RESET THE SIGN BIT */
MVC P05,VARSET /* INDICATE SET FUNCTION IS REQUESTED */
MVC P04,FOUR /* SET UP LENGTH OF FIELD TO SET */
MVC P09(1),BLANK /* MOVE A BLANK TO DATA FIELD */
MVC P09+1(15),P09 /* BLANK OUT ENTIRE FIELD */
L 9,P02 /* POINT TO INPUT VARIABLE */
L 6,P03 /* GET THE LENGTH OF INPUT VARIABLE NAME */
BCTR 6,0 /* DECREMENT IT BY ONE FOR MVC */
EX 6,MOVE /* MOVE INPUT VARIABLE TO P09 */
MVC P11(1),BLANK /* MOVE A BLANK TO P11 */
MVC P11+1(255),P11 /* BLANK OUT ENTIRE FIELD */
MVC P11(4),WK2 /* GET VALUE TO SET VARIABLE TO */
L 15,P01 /* SET UP TO CALL ROUTER */
BALR 14,15 /* CALL SET FUNCTION PROCESSOR */
EXIT C 15,SXTEEN /* IF RETURN CODE IS 16 */
BE EXIT16 /* */
L R13,SAVEAREA+4 /* RESTORE SAVE AREA POINTER */
LM R14,R12,12(R13) /* RESTORE CALLERS REGISTERS */
SLR 15,15 /* GOOD RC */
BR R14 /* RETURN TO CALLER */
EXIT16 L R13,SAVEAREA+4 /* RESTORE SAVE AREA POINTER */
LM R14,R12,12(R13) /* RESTORE CALLERS REGISTERS */
LA R15,16 /* LOAD RC 16 INTO R15 */
BR R14 /* RETURN TO CALLER */
EJECT
**CONSTANT AREA**

```asm
SPACE 1

MOVE MVC P09(0),0(R9) /* MOVE INPUT VARIABLE NAME INTO P09 */
SAVEAREA DC 18F'0' /* SAVE AREA FOR CALLERS REGISTERS */
FOUR DC F'4' /* LENGTH OF A FULL WORD */
SXTEEN DC F'16' /* BAD RETURN CODE=16 */
DS 0D /* GET ON DOUBLE WORD BOUNDARY */
WK1 DS D /* AREA FOR CVD FOR NUMBER OF PREFIXES */
WK2 DS F /* UNPACK NUMBER OF PREFIXES */
PERA DC C'PERA/' /* PREFIX VALUE TO LOCATE */
BLANK DC C' ' /* BLANK */
ONE DC F'1' /* INCREMENT FOR PREFIX FOUND */
FIRST DC X'24' /* FIRST TIME THRU SWITCH */
NEXT DC X'22' /* NOT FIRST TIME THRU */
ZONE DC X'FF' /* MASK FOR UNPACK VARIABLE */
LOCATE DC C'L' /* LOCATE MODE INDICATOR */
VARSET DC C'S' /* SET MODE INDICATOR */
VARMOVE DC C'M' /* MOVE MODE INDICATOR */
VERREAD DC C'R' /* READ MODE INDICATOR */

SPACE 3
```

**REGISTER EQUATES**

```asm
SPACE 1

R0 EQU 00
R1 EQU 01
R2 EQU 02
R3 EQU 03
R4 EQU 04
R5 EQU 05
R6 EQU 06
R7 EQU 07
R8 EQU 08
R9 EQU 09
R10 EQU 10
R11 EQU 11
R12 EQU 12
R13 EQU 13
R14 EQU 14
R15 EQU 15
EJECT
```

**PARAMETER LIST DESECT**

```asm
SPACE 1

INPUT DSECT

PO1 DS F /* ADDRESS OF FUNCTION ROUTINE */
```
The RFT in this next example calls the user CALL command exit routine. It passes to the exit routine the results of merging the user-specified search results and opened problems.

**RFT for Identifying Problems with Three or More Assignees**

**SECTION**

**TITLE** /* REPORT TITLES */

PUT COLUMN(C) VALUE(ASSIGNEE TRACKER REPORT)
SPACE LINES(0)

PUT COLUMN(001) VALUE(DATE &ZEDATE)
PUT COLUMN(073) VALUE(PAGE)

PUT COLUMN(078) VALUE(&ZPAGENO) LENGTH(3) JUSTIFY(R)
SPACE LINES(2)

ETITLE /* END REPORT TITLES */

HEADING /* REPORT HEADINGS */

PUT COLUMN(001) VALUE(DATE) /* 1ST HEADING LINE */
PUT COLUMN(013) VALUE(PROBLEM)
PUT COLUMN(023) VALUE(KEY ITEM)
PUT COLUMN(040) VALUE(ASG)

PUT COLUMN(001) VALUE(ENTERED) /* 2ND HEADING LINE */
PUT COLUMN(013) VALUE(NUMBER)
PUT COLUMN(023) VALUE(AFFECTED)
PUT COLUMN(034) VALUE(PRI)
PUT COLUMN(040) VALUE(CNT)
PUT COLUMN(047) VALUE(DESCRIPTION)

PUT COLUMN(001) VALUE(----------) /* 3RD HEADING LINE */
PUT COLUMN(013) VALUE(--------)
PUT COLUMN(023) VALUE(--------)
PUT COLUMN(035) VALUE(---)
PUT COLUMN(040) VALUE(----)
PUT COLUMN(047) VALUE(---------------------------------------)
SPACE LINES(1) /* BLANK LINE */

EHEADING /* END REPORT HEADINGS */

HFORMAT /* FORMAT COMMON INSERT */
FORMAT HISTORY(DATM/) LENGTH(10) /* FORMAT DATES ASSIGNED */
FORMAT VALUE(' ') LENGTH(01)
FORMAT HISTORY(TIMM/) LENGTH(05) /* FORMAT TIMES ASSIGNED */
FORMAT VALUE(' ') LENGTH(01)
Example of Report

DATE 11/22/1998  ASSIGNEE TRACKER REPORT   PAGE 1

<table>
<thead>
<tr>
<th>DATE ENTERED</th>
<th>PROBLEM NUMBER</th>
<th>KEY ITEM AFFECTED</th>
<th>PRI CNT</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>11/16/1998</td>
<td>00008445</td>
<td>TERMINAL</td>
<td>02</td>
<td>3</td>
</tr>
</tbody>
</table>

ASSIGNEE NAME: 11/16/1998 02:30 HDMARGE JACKSON
11/16/1998 04:00 JACKSON CARLSON
11/18/1998 10:15 CARLSON IBMCE

Figure 94. Output for Problems with Three or More Assignees
This appendix describes how to customize the output data set from RFT BLMZZ46 so the DRAW command can produce a configuration diagram of your system.

The RFT extracts the data for a configuration diagram from the configuration database. Together with any search argument you use when invoking the report, it determines the structure and contents of the eventual diagram.

Before attempting to customize the output from BLMZZ46, be sure you thoroughly understand the operation of the RFT.

Understanding Input to the DRAW Command

The data set output by the RFT BLMZZ46 provides the input to the DRAW command. The input is a series of keywords separated by at least one blank. Keywords can continue across report lines, but they cannot exceed 255 characters in length.

Specify the keywords in the order they are presented here.

1. Specify the DIAGRAM(C) keyword.

   DIAGRAM(C) identifies the report as a configuration diagram data set. DRAW command processing ends with an error message if the keyword value is not “C” or if this keyword is not found in the first 200 records.

2. Specify the SUB-DIAGRAM keyword.

   SUB-DIAGRAM indicates the start of the set of keywords (data) associated with the first subdiagram. Do not enter a value for this keyword; DRAW command processing ends with an error message if a keyword value is present.

3. Specify the RNID keyword.

   RNID(component identifier) identifies the start of the set of keywords (data) associated with the component that heads the subdiagram and contains the record identifier (RNID/value) of that component. The value can be 1 to 8 characters long.

4. Following the RNID keyword you can specify two optional keywords that define a table format subdiagram. Use these keywords only if you want to define table format subdiagrams. If these are not present, the subdiagram defaults to hierarchical format. The table format keywords appear as pairs of keywords. Each pair defines one column of the table, as follows:
WIDTH(nn)
specifies the width of the (next) column. *nn* is a numeric value between 5 and 50.

TITLE(column title)
associates a title with the column. The column title can be between 1 and 255 characters long.

5. Following the RNID or the WIDTH/TITLE keywords, you can specify two more optional keywords to define the data associated with the component:

TEXT-IN(text data)
specifies the text data that is to appear inside the component figure. You can repeat the TEXT-IN keyword; each keyword causes a new line but only the first 255 characters of data are processed for all the TEXT-IN keywords for a component. The text data cannot exceed 255 characters in length.

PORTS(nnnn)
specifies the number of ports on the component. If this keyword is present, and the subdiagram is a table format subdiagram, this keyword determines the number of unused ports in the special Port Number column added on the left side of the table. If the subdiagram type is not table format, the system ignores this keyword. If not present for a table format subdiagram, the port number column is omitted. *nnnn* is a numeric value in the range from 1 to 9999.

6. The SORT keyword is also optional.

SORT(sort key)
determines the sequence of the subdiagram within the diagram. If you do not specify this keyword, the subdiagram is sorted based on the input sequence. The sort key can be between 1 and 20 characters long.

7. After the keywords for the head component of the subdiagram, specify the data for each component (and its associated connections) contained in the map as follows:

a. For each map level:

LEVEL(map level)
contains the map level returned by the RFT search. It defines the structure of the diagram. The value for map level must be in the same format as the value of the RFT &ZMAPLVL variable.

b. For each connection:

CONNECTION
indicates that connection data follows. Do not code a value for this keyword.

TEXT-ABOVE(text data)
indicates that the specified text describing the connection appears on the diagram above the connection in a hierarchical format subdiagram. The text data can be between 1 and 255 characters long. Report processing ends with an error message if the keyword value is greater than 255 characters in length.

The TEXT-ABOVE keyword is optional. It is ignored within a table format subdiagram.

TEXT-BELOW(text data)
indicates that the specified text describing the connection appears on the diagram below the connection in a hierarchical format subdiagram. The text data can be between 1 and 255 characters long.
c. For the component:

**RNID(component identifier)**

is the RNID/ value for the component. The component identifier can be between 1 and 8 characters long.

**TEXT-IN(text data)**

specifies text describing the component that is to appear on the diagram inside the component box in a hierarchical format subdiagram. You can repeat the TEXT-IN keyword; each keyword causes a new line, but only the first 255 characters of data is processed for all the TEXT-IN keywords for a component. The text data cannot exceed 255 characters in length.

The TEXT-IN keyword is optional. It is ignored within a table format subdiagram.

**TEXT-n(text data)**

specifies text describing the component that is to appear in column n in a table format subdiagram. (The keyword suffix n is in the range from 1 to 9.) The text data can be between 1 and 255 characters long.

The TEXT-n keyword is optional. It is ignored within a hierarchical format subdiagram. It is also ignored within a table format subdiagram if no column n is defined.

**SORT(sort key)**

determines the sequence of the component and its preceding connections relative to components at the same level in the hierarchy. The sequence and the LEVEL keyword value determine the hierarchy. This is concatenated after the PORT keyword value associated with the first connection for the component for table format subdiagrams. The sort key cannot exceed 20 characters in length.

The SORT keyword is optional.

**OFF-PAGE(component RNID/)**

causes an off-page connector to be drawn on the diagram alongside a component. The reverse off-page connector is drawn next to the component (identified by RNID/) that heads the subdiagram. The component identifier is between 1 and 8
characters long. If no matching component identifier for a subdiagram is found, a warning message is generated on the diagram. 

The OFF-PAGE keyword is optional.

8. Repeat this set of keywords, beginning with SUBDIAGRAM, for each subsequent subdiagram that is to appear in the diagram.

9. Code END to indicate the end of the diagram. This keyword is optional, but it helps you identify the end of a diagram.

Figure 95 is an example of the output from the Configuration Diagram RFT.

DIAGRAM SUB-DIAGRAM RNID(CPU5) TEXT-IN(CPU A 4341 8MB) LEVEL ( 2)
CONNECTION RNID(CHANNEL1) TEXT-IN(CHANO) LEVEL( 3) CONNECTION TEXT-AB0 VE(00E) RNID(LP1) TEST-IN(LINE PRINTER) ...etc.

SUB-DIAGRAM WIDTH(7) TITLE(CABLE) WIDTH(8) TITLE(DEVICE) WIDTH(10) TITLE (LOCATION) WIDTH(13) TITLE(DESCRIPTION) ...etc.

RNID(CONTROLLER ) TEXT IN(3274-A XXXXXXXXXX XXXXXXXXXX) PORTS(224) LEVEL ( 2) CONNECTION TEXT-1(A1) PORT(1) RNID(TERM 612) TEXT-2(3279-A) TEXT -3(H OFFICE) LEVEL( 2) ...etc.

Figure 95. Output from the Configuration Diagram RFT
Field Characteristics

This appendix lists fields from Tivoli Information Management for z/OS record panels and their major characteristics. You can use this information to create reports. The fields are listed in alphanumeric order within record type. The characteristics include the following:

Field name
The label used to identify fields on data-entry or assisted-entry panels.

Prefix
A short, standard word that represents the type of data.

Prefix index key
A dictionary index key that represents a prefix. An exclamation point (!) must precede the index when used in an RFT.

S-Word index
A dictionary index key that represents an s-word. An exclamation point (!) must precede the index when used in an RFT.

Journal Field
A field whose data is kept in the history portion of the record. Journal fields have a YES for this characteristic.

Length
The maximum number of characters for the field. An asterisk (*) indicates the length of the field can vary depending on your external date format.

The panels as shipped with Tivoli Information Management for z/OS contain date fields that support the entry and display of any of the 22 external date formats supported by Tivoli Information Management for z/OS. The panels reside in data set BLM.SBLMPNLS, and the default length of date fields is 10 characters. A default external date format is specified on the DATEFMT keyword of the BLGPARMS macro when the session-parameters member is defined, but users can override this value by making a date format selection in the user profile. Because the Tivoli Information Management for z/OS dictionary data set supports date fields in both 10- and 8-character format, the following tables list the characteristics of date fields for both formats.

The s-word indexes for freeform text data are the same for all record types. Because they do not occur in a defined field, the various kinds of freeform text are not included in the lists on the following pages. Table 23 on page 320 is a list of the freeform text data types and their associated s-word indexes.
### Table 23. Freeform Text Data Types and Their S-Word Indexes

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Structured Index Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description text (except SRCs)</td>
<td>S0E01</td>
</tr>
<tr>
<td>Status text</td>
<td>S0E02</td>
</tr>
<tr>
<td>Resolution text</td>
<td>S0E03</td>
</tr>
<tr>
<td>Notes text</td>
<td>S0E04</td>
</tr>
<tr>
<td>Justification text</td>
<td>S0E05</td>
</tr>
<tr>
<td>Backup plan text</td>
<td>S0E06</td>
</tr>
<tr>
<td>Location/address text</td>
<td>S0E07</td>
</tr>
<tr>
<td>SRC description text</td>
<td>S0E08</td>
</tr>
</tbody>
</table>

### Privilege Class Records

**Table 24. Privilege Class Records.** An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Change assignment</td>
<td>AUTH/</td>
<td>P0072</td>
<td>S0E7E</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Change close</td>
<td>AUTH/</td>
<td>P0072</td>
<td>S0E7F</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Change display</td>
<td>AUTH/</td>
<td>P0072</td>
<td>S0E7D</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Change entry</td>
<td>AUTH/</td>
<td>P0072</td>
<td>S0E7A</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Change delete</td>
<td>AUTH/</td>
<td>P0072</td>
<td>S0E7C</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Change update</td>
<td>AUTH/</td>
<td>P0072</td>
<td>S0E7B</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Class display</td>
<td>AUTH/</td>
<td>P0072</td>
<td>S0E93</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Class entry</td>
<td>AUTH/</td>
<td>P0072</td>
<td>S0E90</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Class name/ID</td>
<td>RNID/</td>
<td>P01E9</td>
<td>S0CCF</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Class delete</td>
<td>AUTH/</td>
<td>P0072</td>
<td>S0E91</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Class update</td>
<td>AUTH/</td>
<td>P0072</td>
<td>S0E92</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Configuration display</td>
<td>AUTH/</td>
<td>P0072</td>
<td>S0E8A</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Configuration entry</td>
<td>AUTH/</td>
<td>P0072</td>
<td>S0E87</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Configuration delete</td>
<td>AUTH/</td>
<td>P0072</td>
<td>S0E89</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Configuration update</td>
<td>AUTH/</td>
<td>P0072</td>
<td>S0E88</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Contact department</td>
<td>GROC/</td>
<td>P014A</td>
<td>S0B9E</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>Contact name</td>
<td>PERC/</td>
<td>P01A9</td>
<td>S0B5C</td>
<td>YES</td>
<td>15</td>
</tr>
<tr>
<td>Contact phone number</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B30</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Date entered</td>
<td>DATE/</td>
<td>P00D8</td>
<td>S0C34</td>
<td></td>
<td>10*</td>
</tr>
<tr>
<td>Date last altered</td>
<td>DATM/</td>
<td>P00DF</td>
<td>S0C35</td>
<td></td>
<td>10*</td>
</tr>
<tr>
<td>Dbadmin</td>
<td>AUTH/</td>
<td>P0072</td>
<td>S14FO</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Description</td>
<td></td>
<td></td>
<td>S0E0F</td>
<td></td>
<td>45</td>
</tr>
<tr>
<td>Eligible user ID 1</td>
<td>CODU/</td>
<td>P00C2</td>
<td>S0D30</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Eligible user ID 2</td>
<td>CODU/</td>
<td>P00C2</td>
<td>S0D31</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Eligible user ID 3</td>
<td>CODU/</td>
<td>P00C2</td>
<td>S0D32</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Eligible user ID 4</td>
<td>CODU/</td>
<td>P00C2</td>
<td>S0D33</td>
<td></td>
<td>8</td>
</tr>
</tbody>
</table>
Table 24. Privilege Class Records (continued). An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eligible user ID 5</td>
<td>CODU/</td>
<td>P00C2</td>
<td>S0D34</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Eligible user ID 6</td>
<td>CODU/</td>
<td>P00C2</td>
<td>S0D35</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Eligible user ID 7</td>
<td>CODU/</td>
<td>P00C2</td>
<td>S0D36</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Eligible user ID 8</td>
<td>CODU/</td>
<td>P00C2</td>
<td>S0D37</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Eligible user ID 9</td>
<td>CODU/</td>
<td>P00C2</td>
<td>S0D38</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Eligible user ID 10</td>
<td>CODU/</td>
<td>P00C2</td>
<td>S0D39</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Eligible user ID 11</td>
<td>CODU/</td>
<td>P00C2</td>
<td>S0D3A</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Eligible user ID 12</td>
<td>CODU/</td>
<td>P00C2</td>
<td>S0D3B</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Eligible user ID 13</td>
<td>CODU/</td>
<td>P00C2</td>
<td>S0D3C</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Eligible user ID 14</td>
<td>CODU/</td>
<td>P00C2</td>
<td>S0D3D</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Eligible user ID 15</td>
<td>CODU/</td>
<td>P00C2</td>
<td>S0D3E</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Eligible user ID 16</td>
<td>CODU/</td>
<td>P00C2</td>
<td>S0D3F</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Eligible user ID 17</td>
<td>CODU/</td>
<td>P00C2</td>
<td>S0D40</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Eligible user ID 18</td>
<td>CODU/</td>
<td>P00C2</td>
<td>S0D41</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Eligible user ID 19</td>
<td>CODU/</td>
<td>P00C2</td>
<td>S0D42</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Eligible user ID 20</td>
<td>CODU/</td>
<td>P00C2</td>
<td>S0D43</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Eligible user ID 21</td>
<td>CODU/</td>
<td>P00C2</td>
<td>S0D44</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Eligible user ID 22</td>
<td>CODU/</td>
<td>P00C2</td>
<td>S0D45</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Eligible user ID 23</td>
<td>CODU/</td>
<td>P00C2</td>
<td>S0D46</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Eligible user ID 24</td>
<td>CODU/</td>
<td>P00C2</td>
<td>S0D47</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Entry privilege class</td>
<td>CLAE/</td>
<td>P007A</td>
<td>S0BB1</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Financial display</td>
<td>AUTH/</td>
<td>P0072</td>
<td>S0E9C</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Financial entry</td>
<td>AUTH/</td>
<td>P0072</td>
<td>S0E99</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Financial delete</td>
<td>AUTH/</td>
<td>P0072</td>
<td>S0E9A</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Financial update</td>
<td>AUTH/</td>
<td>P0072</td>
<td>S0E9B</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Location code</td>
<td>LOCC/</td>
<td>P016D</td>
<td>S0C0C</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Owning privilege class</td>
<td>CLAO/</td>
<td>P007D</td>
<td>S0BB5</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Primary partition name</td>
<td></td>
<td></td>
<td>S011B</td>
<td></td>
<td>9</td>
</tr>
<tr>
<td>Problem assignment</td>
<td>AUTH/</td>
<td>P0072</td>
<td>S0E75</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Problem close</td>
<td>AUTH/</td>
<td>P0072</td>
<td>S0E76</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Problem display</td>
<td>AUTH/</td>
<td>P0072</td>
<td>S0E74</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Problem entry</td>
<td>AUTH/</td>
<td>P0072</td>
<td>S0E71</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Problem delete</td>
<td>AUTH/</td>
<td>P0072</td>
<td>S0E73</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Problem update authority</td>
<td>AUTH/</td>
<td>P0072</td>
<td>S0E72</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>PMF dictionary display</td>
<td>AUTH/</td>
<td>P0072</td>
<td>S0EAC</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>PMF dictionary update</td>
<td>AUTH/</td>
<td>P0072</td>
<td>S0EAB</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>PMF panel copy</td>
<td>AUTH/</td>
<td>P0072</td>
<td>S0EAD</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>PMF panel delete</td>
<td>AUTH/</td>
<td>P0072</td>
<td>S0EAE</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>PMF panel update</td>
<td>AUTH/</td>
<td>P0072</td>
<td>S0EA9</td>
<td></td>
<td>3</td>
</tr>
</tbody>
</table>
Privilege Class Records

Table 24. Privilege Class Records (continued). An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>PMF reports</td>
<td>AUTH/</td>
<td>P0072</td>
<td>S0EB0</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>PMF panel list</td>
<td>AUTH/</td>
<td>P0072</td>
<td>S077C</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Rules display authority</td>
<td>AUTH/</td>
<td>P0072</td>
<td>S1189</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rules entry authority</td>
<td>AUTH/</td>
<td>P0072</td>
<td>S1186</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rules delete authority</td>
<td>AUTH/</td>
<td>P0072</td>
<td>S1188</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rules update authority</td>
<td>AUTH/</td>
<td>P0072</td>
<td>S1187</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Secondary partition list</td>
<td></td>
<td></td>
<td>S0118</td>
<td></td>
<td>9</td>
</tr>
<tr>
<td>Sharing partition list</td>
<td>PTID/</td>
<td>P03CA</td>
<td>S011D</td>
<td></td>
<td>9</td>
</tr>
<tr>
<td>SRC entry authority</td>
<td>AUTH/</td>
<td>P0072</td>
<td>S0EB6</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>SRC delete authority</td>
<td>AUTH/</td>
<td>P0072</td>
<td>S0EB8</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>SRC update authority</td>
<td>AUTH/</td>
<td>P0072</td>
<td>S0EB7</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Time entered</td>
<td>TIME/</td>
<td>P028A</td>
<td>S0C61</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>Time last altered</td>
<td>TIMM/</td>
<td>P028F</td>
<td>S0C62</td>
<td>YES</td>
<td>5</td>
</tr>
<tr>
<td>Transfer-to class</td>
<td>CLAT/</td>
<td>P0081</td>
<td>S0BCC</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>User last altered</td>
<td>USER/</td>
<td>P02C3</td>
<td>S0B5E</td>
<td>YES</td>
<td>8</td>
</tr>
</tbody>
</table>

Stored Response Chain Records—S0B1D

Table 25. Stored Response Chain Records. An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Contact department</td>
<td>GROC/</td>
<td>P014A</td>
<td>S0B9E</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>Contact name</td>
<td>PERC/</td>
<td>P01A9</td>
<td>S0B5C</td>
<td>YES</td>
<td>15</td>
</tr>
<tr>
<td>Contact phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B30</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Date entered</td>
<td>DATE/</td>
<td>P00D8</td>
<td>S0C34</td>
<td></td>
<td>10*</td>
</tr>
<tr>
<td>Date last altered</td>
<td>DATM/</td>
<td>P0805</td>
<td>S0C34</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Description</td>
<td></td>
<td></td>
<td>S0E0F</td>
<td></td>
<td>45</td>
</tr>
<tr>
<td>Entry privilege class</td>
<td>CLAE/</td>
<td>P007A</td>
<td>S0BB1</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Execution class 1</td>
<td>CLAX/</td>
<td>P0084</td>
<td>S0BB6</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Execution class 2</td>
<td>CLAX/</td>
<td>P0084</td>
<td>S0BB7</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Execution class 3</td>
<td>CLAX/</td>
<td>P0084</td>
<td>S0BB8</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Execution class 4</td>
<td>CLAX/</td>
<td>P0084</td>
<td>S0BB9</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Execution class 5</td>
<td>CLAX/</td>
<td>P0084</td>
<td>S0BBA</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Execution class 6</td>
<td>CLAX/</td>
<td>P0084</td>
<td>S0BBB</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Execution class 7</td>
<td>CLAX/</td>
<td>P0084</td>
<td>S0BBC</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Execution class 8</td>
<td>CLAX/</td>
<td>P0084</td>
<td>S0BBD</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Execution class 9</td>
<td>CLAX/</td>
<td>P0084</td>
<td>S0BBE</td>
<td></td>
<td>8</td>
</tr>
</tbody>
</table>
Table 25. Stored Response Chain Records (continued). An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution class 10</td>
<td>CLAX/</td>
<td>P0084</td>
<td>S0BBF</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Execution class 11</td>
<td>CLAX/</td>
<td>P0084</td>
<td>S0BC0</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Execution class 12</td>
<td>CLAX/</td>
<td>P0084</td>
<td>S0BC1</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Execution class 13</td>
<td>CLAX/</td>
<td>P0084</td>
<td>S0BC2</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Execution class 14</td>
<td>CLAX/</td>
<td>P0084</td>
<td>S0BC3</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Execution class 15</td>
<td>CLAX/</td>
<td>P0084</td>
<td>S0BC4</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Execution class 16</td>
<td>CLAX/</td>
<td>P0084</td>
<td>S0BC5</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Execution class 17</td>
<td>CLAX/</td>
<td>P0084</td>
<td>S0BC6</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Execution class 18</td>
<td>CLAX/</td>
<td>P0084</td>
<td>S0BC7</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Location code</td>
<td>LOCC/</td>
<td>P016D</td>
<td>S0C0C</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Owning privilege class</td>
<td>CLAO/</td>
<td>P007D</td>
<td>S0BB5</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Product name</td>
<td></td>
<td></td>
<td>S0C23</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>SRC final responses</td>
<td></td>
<td></td>
<td>S0C31</td>
<td></td>
<td>48</td>
</tr>
<tr>
<td>Starting panel name</td>
<td>PAID/</td>
<td>P01A1</td>
<td>S0C2B</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>SRC name/ID</td>
<td>RNID/</td>
<td>P01E9</td>
<td>S0CCF</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Time entered</td>
<td>TIME/</td>
<td>P028A</td>
<td>S0C61</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>Time last altered</td>
<td>TIMM/</td>
<td>P028F</td>
<td>S0C62</td>
<td>YES</td>
<td>5</td>
</tr>
<tr>
<td>Transfer-to class</td>
<td>CLAT/</td>
<td>P0081</td>
<td>S0BCC</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>User last altered</td>
<td>USER/</td>
<td>P02C3</td>
<td>S0B5E</td>
<td>YES</td>
<td>8</td>
</tr>
</tbody>
</table>

Problem Records—S0032 or S0B01

Note: The following fields in problem records are relevant only if you are using the Tivoli Service Desk Bridge to integrate with the Tivoli Problem Management application of Tivoli Service Desk 6.0: Notify user ID, TSD function state, TSD record ID, TSD refresh date, TSD refresh time, TSD site ID, TSD user ID.

The Gateway ID and TEC Event ID fields are unique to the Problem Service application.

Table 26. Problem Records. An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assignee department</td>
<td>GROA/</td>
<td>P0148</td>
<td>S0B9C</td>
<td>YES</td>
<td>11</td>
</tr>
<tr>
<td>Assignee name</td>
<td>PERA/</td>
<td>P01A8</td>
<td>S0B5A</td>
<td>YES</td>
<td>40</td>
</tr>
<tr>
<td>Assignee phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B2E</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Assignment number</td>
<td>COUX/</td>
<td>P00CD</td>
<td>S0D19</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Assignment status</td>
<td>STAA/</td>
<td>P026C</td>
<td>S0BEB</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>APAR number</td>
<td>APAR/</td>
<td>P0070</td>
<td>S0E2F</td>
<td></td>
<td>7</td>
</tr>
<tr>
<td>APAR/PTF status</td>
<td>STAF/</td>
<td>P027B</td>
<td>S0BEC</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Bypass available</td>
<td>MISB/</td>
<td>P017C</td>
<td>S0ECA</td>
<td>YES</td>
<td>3</td>
</tr>
</tbody>
</table>
Table 26. Problem Records (continued). An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cause change number</td>
<td>RNCX/</td>
<td>P01DF</td>
<td>S0CD1</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Cause code</td>
<td>CODC/</td>
<td>P00BD</td>
<td>S0C0E</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Checkout application ID</td>
<td>APPL/</td>
<td>P05B3</td>
<td>S14EF</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Circuit number</td>
<td>PH/</td>
<td>P000F</td>
<td>S0D12</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Cluster name</td>
<td>CLSN/</td>
<td>P0086</td>
<td>S0CB9</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Component APARed</td>
<td>PIDS/</td>
<td>P01B1</td>
<td>S0BE3</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>Console output</td>
<td>LOCX/</td>
<td>P016E</td>
<td>S0D69</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>Current phase</td>
<td>CODP/</td>
<td>P00BF</td>
<td>S0C0B</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Current priority</td>
<td>PRIO/</td>
<td>P01B7</td>
<td>S0BE7</td>
<td>YES</td>
<td>2</td>
</tr>
<tr>
<td>Customer PD time</td>
<td>INTC/</td>
<td>P0472</td>
<td>S0D8B</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Data set type</td>
<td>DSTN/</td>
<td>P012B</td>
<td>S0D7B</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>Date assigned</td>
<td>DATA/</td>
<td>P00CE</td>
<td>S0C37</td>
<td>YES</td>
<td>10*</td>
</tr>
<tr>
<td>Date closed</td>
<td>DATR/</td>
<td>P00E7</td>
<td>S0C38</td>
<td></td>
<td>10*</td>
</tr>
<tr>
<td>Date entered</td>
<td>DATE/</td>
<td>P00D8</td>
<td>S0C34</td>
<td></td>
<td>10*</td>
</tr>
<tr>
<td>Date finished</td>
<td>DATF/</td>
<td>P00DB</td>
<td>S0C40</td>
<td>YES</td>
<td>10*</td>
</tr>
<tr>
<td>Date fix required</td>
<td>DATD/</td>
<td>P00D5</td>
<td>S0C49</td>
<td>YES</td>
<td>10*</td>
</tr>
<tr>
<td>Date last altered</td>
<td>DATM/</td>
<td>P00DF</td>
<td>S0C35</td>
<td>YES</td>
<td>10*</td>
</tr>
<tr>
<td>Date last refreshed</td>
<td>DATRF/</td>
<td>P0609</td>
<td>S14F9</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Date occurred</td>
<td>DATO/</td>
<td>P00E3</td>
<td>S0C3D</td>
<td></td>
<td>10*</td>
</tr>
<tr>
<td>Date opened</td>
<td>DATX/</td>
<td>P00ED</td>
<td>S0C3E</td>
<td></td>
<td>10*</td>
</tr>
<tr>
<td>Date reporter notified</td>
<td>DATN/</td>
<td>P00E1</td>
<td>S0C48</td>
<td></td>
<td>10*</td>
</tr>
<tr>
<td>Date started</td>
<td>DATB/</td>
<td>P00D0</td>
<td>S0C43</td>
<td>YES</td>
<td>10*</td>
</tr>
<tr>
<td>Description</td>
<td></td>
<td>S0E0F</td>
<td></td>
<td></td>
<td>45</td>
</tr>
<tr>
<td>Device impact</td>
<td>IMPD/</td>
<td>P0157</td>
<td>S0BE0</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Device name</td>
<td>COMD/</td>
<td>P00C7</td>
<td>S0CA9</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Diagnostic output</td>
<td>LOCX/</td>
<td>P016E</td>
<td>S0D76</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>Dump data set</td>
<td></td>
<td>S0D7A</td>
<td></td>
<td></td>
<td>54</td>
</tr>
<tr>
<td>Duplicate count</td>
<td>COUD/</td>
<td>P00CC</td>
<td>S0D28</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Entry privilege class</td>
<td>CLAE/</td>
<td>P007A</td>
<td>S0BB1</td>
<td></td>
<td>8</td>
</tr>
</tbody>
</table>
Table 26. Problem Records (continued). An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Error code</td>
<td>CODE/</td>
<td>P00BE</td>
<td>S0C0D</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Escalation level</td>
<td>ESCL/</td>
<td>P0443</td>
<td>S0D90</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>EC number</td>
<td>LVLS/</td>
<td>P0172</td>
<td>S0637</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Fix available</td>
<td>MISX/</td>
<td>P0180</td>
<td>S0ECEF</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Fix change number</td>
<td>RNCR/</td>
<td>P01DD</td>
<td>S0CD2</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Gateway ID</td>
<td>GWID/</td>
<td>P04E0</td>
<td>S1260</td>
<td></td>
<td>9</td>
</tr>
<tr>
<td>Graph/log data</td>
<td>LOCX/</td>
<td>P016E</td>
<td>S0D71</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>Initial priority</td>
<td>PRII/</td>
<td>P01B6</td>
<td>S0BE6</td>
<td>YES</td>
<td>2</td>
</tr>
<tr>
<td>Input data</td>
<td>LOCX/</td>
<td>P016E</td>
<td>S0D6B</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>Interested class 01</td>
<td>SP01/</td>
<td>P022A</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Interested class 02</td>
<td>SP02/</td>
<td>P022C</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Interested class 03</td>
<td>SP03/</td>
<td>P022E</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Interested class 04</td>
<td>SP04/</td>
<td>P0230</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Interested class 05</td>
<td>SP05/</td>
<td>P0232</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Interested class 06</td>
<td>SP06/</td>
<td>P0234</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Interested class 07</td>
<td>SP07/</td>
<td>P0236</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Interested class 08</td>
<td>SP08/</td>
<td>P0238</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Interested class 09</td>
<td>SP09/</td>
<td>P023A</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Interested class 10</td>
<td>SP10/</td>
<td>P023C</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Interested class 11</td>
<td>SP11/</td>
<td>P023E</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Interested class 12</td>
<td>SP12/</td>
<td>P0240</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Interested class 13</td>
<td>SP13/</td>
<td>P0242</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Interested class 14</td>
<td>SP14/</td>
<td>P0244</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Interested class 15</td>
<td>SP15/</td>
<td>P0246</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Interested class 16</td>
<td>SP16/</td>
<td>P0248</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Interested class 17</td>
<td>SP17/</td>
<td>P024A</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Interested class 18</td>
<td>SP18/</td>
<td>P024C</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Interested class 19</td>
<td>SP19/</td>
<td>P024D</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Interested class 20</td>
<td>SP20/</td>
<td>P024E</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Interested class 21</td>
<td>SP21/</td>
<td>P024F</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Interested class 22</td>
<td>SP22/</td>
<td>P0250</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Interested class 23</td>
<td>SP23/</td>
<td>P0251</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Interested class 24</td>
<td>SP24/</td>
<td>P0252</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Interested class 25</td>
<td>SP25/</td>
<td>P0253</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Interested class 26</td>
<td>SP26/</td>
<td>P0254</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Interested class 27</td>
<td>SP27/</td>
<td>P0255</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Interested class 28</td>
<td>SP28/</td>
<td>P0256</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>IPCS record number</td>
<td>NUMX/</td>
<td>P0194</td>
<td>S0D16</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Key item affected</td>
<td>COMK/</td>
<td>P00C8</td>
<td>S0CBF</td>
<td></td>
<td>8</td>
</tr>
</tbody>
</table>
Table 26. Problem Records (continued). An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Location code</td>
<td>LOCC/</td>
<td>P016D</td>
<td>S0C0C</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Network impact</td>
<td>IMPN/</td>
<td>P0158</td>
<td>S0BDD</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Network name</td>
<td>COMN/</td>
<td>P00C9</td>
<td>S0CA3</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Notify user ID</td>
<td>NOTID/</td>
<td>P060B</td>
<td>S14FB</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Operator form</td>
<td>LOCX/</td>
<td>P016E</td>
<td>S0D74</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>Original problem number</td>
<td>RNPD/</td>
<td>P01EF</td>
<td>S0CD0</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Outage</td>
<td>INTO/</td>
<td>P0160</td>
<td>S0C8D</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Outage type</td>
<td>TYPO/</td>
<td>P0441</td>
<td>S0C13</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>Output data</td>
<td>LOCX/</td>
<td>P016E</td>
<td>S0D6A</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>Owning privilege class</td>
<td>CLAO/</td>
<td>P007D</td>
<td>S0BBA</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Probable cause</td>
<td>CODM/</td>
<td>P0480</td>
<td>S0C1B</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Problem number</td>
<td>RNID/</td>
<td>P01E9</td>
<td>S0CCF</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Problem status</td>
<td>STAC/</td>
<td>P026D</td>
<td>S0BEE</td>
<td>YES</td>
<td>7</td>
</tr>
<tr>
<td>Problem type</td>
<td>TYPE/</td>
<td>P02AF</td>
<td>S0C09</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Program impact</td>
<td>IMPP/</td>
<td>P0159</td>
<td>S0BDF</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Program name</td>
<td>COMX/</td>
<td>P00CA</td>
<td>S0CA8</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>PTF number</td>
<td>PTFF/</td>
<td>P01BD</td>
<td>S0E30</td>
<td></td>
<td>7</td>
</tr>
<tr>
<td>Repair time</td>
<td>INTX/</td>
<td>P0162</td>
<td>S0D8C</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Reported by</td>
<td>PERS/</td>
<td>P01AC</td>
<td>S0B59</td>
<td></td>
<td>40</td>
</tr>
<tr>
<td>Reporter department</td>
<td>GROS/</td>
<td>P0150</td>
<td>S0B9B</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>Reporter phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B2D</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Rerun time</td>
<td>INTR/</td>
<td>P0161</td>
<td>S0C8E</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Resolved by</td>
<td>PERR/</td>
<td>P01AB</td>
<td>S0B5B</td>
<td></td>
<td>15</td>
</tr>
<tr>
<td>Resolver department</td>
<td>GROR/</td>
<td>P014E</td>
<td>S0B9D</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>Resolver phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B2F</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Resolver privilege class</td>
<td>CLAR/</td>
<td>P007F</td>
<td>S0BB3</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Resource names</td>
<td></td>
<td></td>
<td>S0BE2</td>
<td></td>
<td>54</td>
</tr>
<tr>
<td>Resource types</td>
<td></td>
<td></td>
<td>S0C12</td>
<td></td>
<td>54</td>
</tr>
<tr>
<td>Response/travel time</td>
<td>INTX/</td>
<td>P0162</td>
<td>S0D89</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Source data</td>
<td>LOCX/</td>
<td>P016E</td>
<td>S0D6E</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>Source listing</td>
<td>LOCX/</td>
<td>P016E</td>
<td>S0D6D</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>System impact</td>
<td>IMPS/</td>
<td>P015B</td>
<td>S0BDE</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>System name</td>
<td>NASY/</td>
<td>P018F</td>
<td>S0CA5</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>SMP listing</td>
<td>LOCX/</td>
<td>P016E</td>
<td>S0D6F</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>Target date</td>
<td>DATT/</td>
<td>P00EA</td>
<td>S0C42</td>
<td>YES</td>
<td>10*</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>TEC Event ID</td>
<td>TECID/</td>
<td>P04F7</td>
<td>S14FC</td>
<td></td>
<td>20</td>
</tr>
<tr>
<td>Time assigned</td>
<td>TIMA/</td>
<td>P0281</td>
<td>S0C64</td>
<td>YES</td>
<td>5</td>
</tr>
</tbody>
</table>
Table 26. Problem Records (continued). An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time closed</td>
<td>TIMR/</td>
<td>P0296</td>
<td>S0C65</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>Time entered</td>
<td>TIME/</td>
<td>P028A</td>
<td>S0C61</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>Time finished</td>
<td>TIMF/</td>
<td>P028C</td>
<td>S0C71</td>
<td>YES</td>
<td>5</td>
</tr>
<tr>
<td>Time fix required</td>
<td>TIMD/</td>
<td>P0287</td>
<td>S0C72</td>
<td>YES</td>
<td>5</td>
</tr>
<tr>
<td>Time last altered</td>
<td>TIMM/</td>
<td>P028F</td>
<td>S0C62</td>
<td>YES</td>
<td>5</td>
</tr>
<tr>
<td>Time last refreshed</td>
<td>TIMRF/</td>
<td>P060A</td>
<td>S14FA</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Time occurred</td>
<td>TIMO/</td>
<td>P0292</td>
<td>S0C6a</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>Time opened</td>
<td>TIMX/</td>
<td>P029A</td>
<td>S0C74</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>Time started</td>
<td>TIMB/</td>
<td>P0283</td>
<td>S0C6C</td>
<td>YES</td>
<td>5</td>
</tr>
<tr>
<td>Total time</td>
<td>INTX/</td>
<td>P0162</td>
<td>S0C93</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Trace data</td>
<td>LOCX/</td>
<td>P016E</td>
<td>S0D70</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>Tracked by</td>
<td>PERC/</td>
<td>P01A9</td>
<td>S0B5C</td>
<td>YES</td>
<td>15</td>
</tr>
<tr>
<td>Tracker department</td>
<td>GROC/</td>
<td>P014A</td>
<td>S0B9E</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>Tracker phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B30</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Transfer-to class</td>
<td>CLAT/</td>
<td>P0081</td>
<td>S0BCC</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>TSD function state</td>
<td>S14FF</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TSD record ID</td>
<td>PMREC/</td>
<td>P0608</td>
<td>S14F8</td>
<td></td>
<td>15</td>
</tr>
<tr>
<td>TSD site ID</td>
<td>OSITE/</td>
<td>P0607</td>
<td>S14F7</td>
<td>YES</td>
<td>6</td>
</tr>
<tr>
<td>TSD user ID</td>
<td>TUSER/</td>
<td>P06DB</td>
<td>S15F9</td>
<td>YES</td>
<td>15</td>
</tr>
<tr>
<td>User form number</td>
<td>NUMX/</td>
<td>P0194</td>
<td>S0D14</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>User last altered</td>
<td>USER/</td>
<td>P02C3</td>
<td>S0B5E</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Vendor number</td>
<td>NUMV/</td>
<td>P0192</td>
<td>S0D11</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>Vendor PD time</td>
<td>INTV/</td>
<td>P0473</td>
<td>S0D8A</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Vendor priority</td>
<td>STAP/</td>
<td>P027C</td>
<td>S0BE8</td>
<td>YES</td>
<td>2</td>
</tr>
<tr>
<td>Vendor status</td>
<td>STAV/</td>
<td>P0280</td>
<td>S0BEF</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Vendor PMR number</td>
<td>NUMX/</td>
<td>P0194</td>
<td>S0F52</td>
<td></td>
<td>8</td>
</tr>
</tbody>
</table>

Change Records—S0B06

Note: The Gateway ID and TEC Event ID fields are unique to the Problem Service application.

Table 27. Change Records. An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual duration</td>
<td>INTO/</td>
<td>P0160</td>
<td>S0C97</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Actual effort</td>
<td>EFA/</td>
<td>P0026</td>
<td>S0C96</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>Actual impact</td>
<td>IMPA/</td>
<td>P0154</td>
<td>S0BE1</td>
<td></td>
<td>6</td>
</tr>
<tr>
<td>Actual start date</td>
<td>DATB/</td>
<td>P00D0</td>
<td>S0C43</td>
<td>YES</td>
<td>10*</td>
</tr>
<tr>
<td></td>
<td>DATB/</td>
<td>P0802</td>
<td>S0C43</td>
<td>YES</td>
<td>10</td>
</tr>
</tbody>
</table>
Table 27. Change Records (continued). An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual start time</td>
<td>TIMX/</td>
<td>P029A</td>
<td>S0C70</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>Approval pending 1</td>
<td>SP01/</td>
<td>P022A</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval pending 2</td>
<td>SP02/</td>
<td>P022C</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval pending 3</td>
<td>SP03/</td>
<td>P022E</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval pending 4</td>
<td>SP04/</td>
<td>P0230</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval pending 5</td>
<td>SP05/</td>
<td>P0232</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval pending 6</td>
<td>SP06/</td>
<td>P0234</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval pending 7</td>
<td>SP07/</td>
<td>P0236</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval pending 8</td>
<td>SP08/</td>
<td>P0238</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval pending 9</td>
<td>SP09/</td>
<td>P023A</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval pending 10</td>
<td>SP10/</td>
<td>P023C</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval pending 11</td>
<td>SP11/</td>
<td>P023E</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval pending 12</td>
<td>SP12/</td>
<td>P0240</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval pending 13</td>
<td>SP13/</td>
<td>P0242</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval pending 14</td>
<td>SP14/</td>
<td>P0244</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval pending 15</td>
<td>SP15/</td>
<td>P0246</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval pending 16</td>
<td>SP16/</td>
<td>P0248</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval pending 17</td>
<td>SP17/</td>
<td>P024A</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval pending 18</td>
<td>SP18/</td>
<td>P024C</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval provided 1</td>
<td>SA01/</td>
<td>P0214</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval provided 2</td>
<td>SA02/</td>
<td>P0215</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval provided 3</td>
<td>SA03/</td>
<td>P0216</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval provided 4</td>
<td>SA04/</td>
<td>P0217</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval provided 5</td>
<td>SA05/</td>
<td>P0218</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval provided 6</td>
<td>SA06/</td>
<td>P0219</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval provided 7</td>
<td>SA07/</td>
<td>P021A</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval provided 8</td>
<td>SA08/</td>
<td>P021B</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval provided 9</td>
<td>SA09/</td>
<td>P021C</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval provided 10</td>
<td>SA10/</td>
<td>P021D</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval provided 11</td>
<td>SA11/</td>
<td>P021E</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval provided 12</td>
<td>SA12/</td>
<td>P021F</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval provided 13</td>
<td>SA13/</td>
<td>P0220</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval provided 14</td>
<td>SA14/</td>
<td>P0221</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval provided 15</td>
<td>SA15/</td>
<td>P0222</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval provided 16</td>
<td>SA16/</td>
<td>P0223</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval provided 17</td>
<td>SA17/</td>
<td>P0224</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval provided 18</td>
<td>SA18/</td>
<td>P0225</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval rejected 1</td>
<td>SR01/</td>
<td>P025A</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval rejected 2</td>
<td>SR02/</td>
<td>P025B</td>
<td></td>
<td></td>
<td>8</td>
</tr>
</tbody>
</table>
Table 27. Change Records (continued). An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Approval rejected 3</td>
<td>S03/</td>
<td>P025C</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval rejected 4</td>
<td>S04/</td>
<td>P025D</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval rejected 5</td>
<td>S05/</td>
<td>P025E</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval rejected 6</td>
<td>S06/</td>
<td>P025F</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval rejected 7</td>
<td>S07/</td>
<td>P0260</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval rejected 8</td>
<td>S08/</td>
<td>P0261</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval rejected 9</td>
<td>S09/</td>
<td>P0262</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval rejected 10</td>
<td>S10/</td>
<td>P0263</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval rejected 11</td>
<td>S11/</td>
<td>P0264</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval rejected 12</td>
<td>S12/</td>
<td>P0265</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval rejected 13</td>
<td>S13/</td>
<td>P0266</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval rejected 14</td>
<td>S14/</td>
<td>P0267</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval rejected 15</td>
<td>S15/</td>
<td>P0268</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval rejected 16</td>
<td>S16/</td>
<td>P0269</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval rejected 17</td>
<td>S17/</td>
<td>P026A</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval rejected 18</td>
<td>S18/</td>
<td>P026B</td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approval status</td>
<td>STAP/</td>
<td>P027C</td>
<td>S0BF0</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Approval status*</td>
<td>APST/</td>
<td>P0510</td>
<td>S12DF</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>APST/</td>
<td>P0511</td>
<td>S12DF</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>APST/</td>
<td>P0512</td>
<td>S12DF</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Approver*</td>
<td>SPAR/</td>
<td>P050E</td>
<td>S12DE</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>SAAR/</td>
<td>P050D</td>
<td>S12DE</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>SRAR/</td>
<td>P050F</td>
<td>S12DE</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Approver class 1</td>
<td>CL01/</td>
<td>P0088</td>
<td>S0B96</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approver class 2</td>
<td>CL02/</td>
<td>P008B</td>
<td>S0B85</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approver class 3</td>
<td>CL03/</td>
<td>P008E</td>
<td>S0B86</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approver class 4</td>
<td>CL04/</td>
<td>P0091</td>
<td>S0B87</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approver class 5</td>
<td>CL05/</td>
<td>P0094</td>
<td>S0B88</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approver class 6</td>
<td>CL06/</td>
<td>P0097</td>
<td>S0B89</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approver class 7</td>
<td>CL07/</td>
<td>P009A</td>
<td>S0B8A</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approver class 8</td>
<td>CL08/</td>
<td>P009D</td>
<td>S0B8B</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approver class 9</td>
<td>CL09/</td>
<td>P00A0</td>
<td>S0B8C</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approver class 10</td>
<td>CL10/</td>
<td>P00A3</td>
<td>S0B8D</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approver class 11</td>
<td>CL11/</td>
<td>P00A6</td>
<td>S0B8E</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approver class 12</td>
<td>CL12/</td>
<td>P00AA</td>
<td>S0B8F</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approver class 13</td>
<td>CL13/</td>
<td>P00AC</td>
<td>S0B90</td>
<td></td>
<td>8</td>
</tr>
</tbody>
</table>
Table 27. Change Records (continued). An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Approver class 14</td>
<td>CL14/</td>
<td>P00AF</td>
<td>S0B91</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approver class 15</td>
<td>CL15/</td>
<td>P00B2</td>
<td>S0B92</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approver class 16</td>
<td>CL16/</td>
<td>P00B5</td>
<td>S0B93</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approver class 17</td>
<td>CL17/</td>
<td>P00B8</td>
<td>S0B94</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Approver class 18</td>
<td>CL18/</td>
<td>P00BB</td>
<td>S0B95</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Assignee department</td>
<td>GROA/</td>
<td>P0148</td>
<td>S0B9C</td>
<td>YES</td>
<td>11</td>
</tr>
<tr>
<td>Assignee name</td>
<td>PERA/</td>
<td>P01A8</td>
<td>S0B5A</td>
<td>YES</td>
<td>40</td>
</tr>
<tr>
<td>Assignee phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B2E</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Backup plan used</td>
<td>MISX/</td>
<td>P0180</td>
<td>S0EC9</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Change number</td>
<td>RNID/</td>
<td>P01E9</td>
<td>S0CCF</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Change reason</td>
<td>CODR/</td>
<td>P00C0</td>
<td>S0C0A</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Change status</td>
<td>STAC/</td>
<td>P026D</td>
<td>S0BEE</td>
<td>YES</td>
<td>7</td>
</tr>
<tr>
<td>Change type</td>
<td>TYPE/</td>
<td>P02AF</td>
<td>S0C09</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Checkout application ID</td>
<td>APPL/</td>
<td>P05B3</td>
<td>S14EF</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Closed by</td>
<td>PERR/</td>
<td>P01AB</td>
<td>S0B5B</td>
<td></td>
<td>15</td>
</tr>
<tr>
<td>Closer department</td>
<td>GROR/</td>
<td>P014E</td>
<td>S0B9D</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>Closer phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B2F</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Closer privilege class</td>
<td>CLAR/</td>
<td>P007F</td>
<td>S0BB3</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Co-requisites</td>
<td>RNCC/</td>
<td>P01DB</td>
<td>S0CD4</td>
<td></td>
<td>44</td>
</tr>
<tr>
<td>Completion code</td>
<td>CODC/</td>
<td>P00BD</td>
<td>S0C0E</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Completion date</td>
<td>DATF/</td>
<td>P00DB</td>
<td>S0C40</td>
<td>YES</td>
<td>10*</td>
</tr>
<tr>
<td>Completion time</td>
<td>TIMF/</td>
<td>P028C</td>
<td>S0C71</td>
<td>YES</td>
<td>5</td>
</tr>
<tr>
<td>Coordinator department</td>
<td>GROC/</td>
<td>P014A</td>
<td>S0B9E</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>Coordinator name</td>
<td>PERC/</td>
<td>P01A9</td>
<td>S0B5C</td>
<td>YES</td>
<td>15</td>
</tr>
<tr>
<td>Coordinator phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B30</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Current phase</td>
<td>CODP/</td>
<td>P00BF</td>
<td>S0C0B</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Current priority</td>
<td>PRIO/</td>
<td>P01B7</td>
<td>S0BE7</td>
<td>YES</td>
<td>2</td>
</tr>
<tr>
<td>Date assigned</td>
<td>DATA/</td>
<td>P00CE</td>
<td>S0C37</td>
<td>YES</td>
<td>10*</td>
</tr>
<tr>
<td>Date entered</td>
<td>DATE/</td>
<td>P00D8</td>
<td>S0C34</td>
<td></td>
<td>10*</td>
</tr>
<tr>
<td>Date last altered</td>
<td>DATM/</td>
<td>P00DF</td>
<td>S0C35</td>
<td>YES</td>
<td>10*</td>
</tr>
<tr>
<td>Date requester notified</td>
<td>DATN/</td>
<td>P00E1</td>
<td>S0C48</td>
<td></td>
<td>10*</td>
</tr>
<tr>
<td>Date required</td>
<td>DATD/</td>
<td>P00D5</td>
<td>S0C49</td>
<td>YES</td>
<td>10*</td>
</tr>
<tr>
<td>Description</td>
<td>S0E0F</td>
<td></td>
<td></td>
<td></td>
<td>45</td>
</tr>
</tbody>
</table>
Table 27. Change Records (continued). An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Device name</td>
<td>COMD/</td>
<td>P00C7</td>
<td>S0CA9</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Entry privilege class</td>
<td>CLAE/</td>
<td>P007A</td>
<td>S0BB1</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Estimated duration</td>
<td>INTE/</td>
<td>P015F</td>
<td>S0C95</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Estimated effort</td>
<td>EFE/</td>
<td>P0027</td>
<td>S0C94</td>
<td>YES</td>
<td>4</td>
</tr>
<tr>
<td>Gateway ID</td>
<td>GWID/</td>
<td>P04E0</td>
<td>S1260</td>
<td></td>
<td>9</td>
</tr>
<tr>
<td>Initial priority</td>
<td>PRII/</td>
<td>P01B6</td>
<td>S0BE6</td>
<td>YES</td>
<td>2</td>
</tr>
<tr>
<td>Key item affected</td>
<td>COMK/</td>
<td>P00C8</td>
<td>S0CBF</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Location code</td>
<td>LOCC/</td>
<td>P016D</td>
<td>S0C0C</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Network name</td>
<td>COMN/</td>
<td>P00C9</td>
<td>S0CA3</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Owning privilege class</td>
<td>CLAO/</td>
<td>P007D</td>
<td>S0BB5</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Planned end date</td>
<td>DATT/</td>
<td>P00EA</td>
<td>S0C42</td>
<td>YES</td>
<td>10*</td>
</tr>
<tr>
<td>Planned end time</td>
<td>TIMT/</td>
<td>P0298</td>
<td>S0C6F</td>
<td>YES</td>
<td>5</td>
</tr>
<tr>
<td>Planned start date</td>
<td>DATP/</td>
<td>P00E5</td>
<td>S0C41</td>
<td>YES</td>
<td>10*</td>
</tr>
<tr>
<td>Planned start time</td>
<td>TIMP/</td>
<td>P0294</td>
<td>S0C6E</td>
<td>YES</td>
<td>5</td>
</tr>
<tr>
<td>Prerequisites</td>
<td>RNCP/</td>
<td>P01DC</td>
<td>S0CD5</td>
<td></td>
<td>44</td>
</tr>
<tr>
<td>Problem fixed</td>
<td>RNPR/</td>
<td>P01F1</td>
<td>S0CD3</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Program name</td>
<td>COMX/</td>
<td>P00CA</td>
<td>S0CA8</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Requested by</td>
<td>PERS/</td>
<td>P01AC</td>
<td>S0B59</td>
<td></td>
<td>40</td>
</tr>
<tr>
<td>Requester department</td>
<td>GROS/</td>
<td>P0150</td>
<td>S0B9B</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>Requester phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B2D</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Reviewer* *for list data (BLGLREVR)</td>
<td>CLAX/</td>
<td>P0085</td>
<td>S12DD</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Reviewer class 1</td>
<td>CLAX/</td>
<td>P0084</td>
<td>S0BB6</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Reviewer class 2</td>
<td>CLAX/</td>
<td>P0084</td>
<td>S0BB7</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Reviewer class 3</td>
<td>CLAX/</td>
<td>P0084</td>
<td>S0BB8</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Reviewer class 4</td>
<td>CLAX/</td>
<td>P0084</td>
<td>S0BB9</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Reviewer class 5</td>
<td>CLAX/</td>
<td>P0084</td>
<td>S0BBA</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Reviewer class 6</td>
<td>CLAX/</td>
<td>P0084</td>
<td>S0BBB</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Reviewer class 7</td>
<td>CLAX/</td>
<td>P0084</td>
<td>S0BBC</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Reviewer class 8</td>
<td>CLAX/</td>
<td>P0084</td>
<td>S0BBD</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Reviewer class 9</td>
<td>CLAX/</td>
<td>P0084</td>
<td>S0BBE</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Reviewer class 10</td>
<td>CLAX/</td>
<td>P0084</td>
<td>S0BBF</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Reviewer class 11</td>
<td>CLAX/</td>
<td>P0084</td>
<td>S0BC0</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Reviewer class 12</td>
<td>CLAX/</td>
<td>P0084</td>
<td>S0BC1</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Reviewer class 13</td>
<td>CLAX/</td>
<td>P0084</td>
<td>S0BC2</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Reviewer class 14</td>
<td>CLAX/</td>
<td>P0084</td>
<td>S0BC3</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Reviewer class 15</td>
<td>CLAX/</td>
<td>P0084</td>
<td>S0BC4</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Reviewer class 16</td>
<td>CLAX/</td>
<td>P0084</td>
<td>S0BC5</td>
<td></td>
<td>8</td>
</tr>
</tbody>
</table>
Table 27. Change Records (continued). An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reviewer class 17</td>
<td>CLAX/</td>
<td>P0084</td>
<td>S0BC6</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Reviewer class 18</td>
<td>CLAX/</td>
<td>P0084</td>
<td>S0BC7</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Risk assessment</td>
<td>IMPR/</td>
<td>P015A</td>
<td>S0BF2</td>
<td>YES</td>
<td>6</td>
</tr>
<tr>
<td>Select code* A</td>
<td></td>
<td></td>
<td>0514</td>
<td>S12DC</td>
<td>YES</td>
</tr>
<tr>
<td>Select code* R</td>
<td></td>
<td></td>
<td>0515</td>
<td>S12DC</td>
<td>YES</td>
</tr>
<tr>
<td>System name</td>
<td>NASY/</td>
<td>P018F</td>
<td>S0CA5</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>TEC Event ID</td>
<td>TECID/</td>
<td>P04F7</td>
<td>S14FC</td>
<td></td>
<td>20</td>
</tr>
<tr>
<td>Time assigned</td>
<td>TIMA/</td>
<td>P0281</td>
<td>S0C64</td>
<td>YES</td>
<td>5</td>
</tr>
<tr>
<td>Time entered</td>
<td>TIME/</td>
<td>P028A</td>
<td>S0C61</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>Time last altered</td>
<td>TIMM/</td>
<td>P028F</td>
<td>S0C62</td>
<td>YES</td>
<td>5</td>
</tr>
<tr>
<td>Time required</td>
<td>TIMD/</td>
<td>P0287</td>
<td>S0C72</td>
<td>YES</td>
<td>5</td>
</tr>
<tr>
<td>Transfer-to class</td>
<td>CLAT/</td>
<td>P0081</td>
<td>S0BCC</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Unexpected problems</td>
<td>MISP/</td>
<td>P017E</td>
<td>S0ECB</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>User form number</td>
<td>NUMX/</td>
<td>P0194</td>
<td>S0D14</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>User last altered</td>
<td>USER/</td>
<td>P02C3</td>
<td>S0B5E</td>
<td>YES</td>
<td>8</td>
</tr>
</tbody>
</table>

Activity Records—S0B07

Table 28. Activity Records. An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Activity name</td>
<td>NAMA/</td>
<td>P018B</td>
<td>S0BCB</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Activity reason</td>
<td>CODR/</td>
<td>P00C0</td>
<td>S0C0A</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Activity record ID</td>
<td>RNID/</td>
<td>P01E9</td>
<td>S0CCF</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Activity status</td>
<td>STAC/</td>
<td>P026D</td>
<td>S0BEE</td>
<td>YES</td>
<td>7</td>
</tr>
<tr>
<td>Activity type</td>
<td>TYPE/</td>
<td>P02AF</td>
<td>S0C09</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Actual duration</td>
<td>INTO/</td>
<td>P0160</td>
<td>S0C97</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Actual effort</td>
<td>EFA/</td>
<td>P0026</td>
<td>S0C96</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>Actual impact</td>
<td>IMPA/</td>
<td>P0154</td>
<td>S0BE1</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Actual start date</td>
<td>DATB/</td>
<td>P00D0</td>
<td>S0C43</td>
<td>YES</td>
<td>10*</td>
</tr>
<tr>
<td>Actual start time</td>
<td>TIMX/</td>
<td>P029A</td>
<td>S0C70</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>Assignee department</td>
<td>GROA/</td>
<td>P014E</td>
<td>S0B9C</td>
<td>YES</td>
<td>11</td>
</tr>
<tr>
<td>Assignee name</td>
<td>PERA/</td>
<td>P01A8</td>
<td>S0B5A</td>
<td>YES</td>
<td>40</td>
</tr>
<tr>
<td>Assignee phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B2E</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Backup plan used</td>
<td>MISX/</td>
<td>P0180</td>
<td>S0EC9</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Closed by</td>
<td>PERR/</td>
<td>P01AB</td>
<td>S0B5B</td>
<td></td>
<td>15</td>
</tr>
<tr>
<td>Closer department</td>
<td>GROR/</td>
<td>P014E</td>
<td>S0B9D</td>
<td></td>
<td>11</td>
</tr>
</tbody>
</table>
Table 28. Activity Records (continued). An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Closer phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B2F</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Closer privilege class</td>
<td>CLAR/</td>
<td>P007F</td>
<td>S0BB3</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Co-requisites</td>
<td>RNCC/</td>
<td>P01DB</td>
<td>S0CD4</td>
<td></td>
<td>44</td>
</tr>
<tr>
<td>Completion code</td>
<td>CODC/</td>
<td>P00BD</td>
<td>S0C0E</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Completion date</td>
<td>DATF/</td>
<td>P00DB</td>
<td>S0C40</td>
<td>YES</td>
<td>10*</td>
</tr>
<tr>
<td></td>
<td>DATF/</td>
<td>P0806</td>
<td>S0C40</td>
<td>YES</td>
<td>10</td>
</tr>
<tr>
<td>Completion time</td>
<td>TIMF/</td>
<td>P028C</td>
<td>S0C71</td>
<td>YES</td>
<td>5</td>
</tr>
<tr>
<td>Contact department</td>
<td>GROC/</td>
<td>P014A</td>
<td>S0B9E</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>Contact name</td>
<td>PERC/</td>
<td>P01A9</td>
<td>S0B5C</td>
<td>YES</td>
<td>15</td>
</tr>
<tr>
<td>Contact phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B30</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Current phase</td>
<td>CODP/</td>
<td>P00BF</td>
<td>S0C0B</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Current priority</td>
<td>PRIO/</td>
<td>P01B7</td>
<td>S0BE7</td>
<td>YES</td>
<td>2</td>
</tr>
<tr>
<td>Date assigned</td>
<td>DATA/</td>
<td>P00CE</td>
<td>S0C37</td>
<td>YES</td>
<td>10*</td>
</tr>
<tr>
<td></td>
<td>DATA/</td>
<td>P0801</td>
<td>S0C37</td>
<td>YES</td>
<td>10</td>
</tr>
<tr>
<td>Date entered</td>
<td>DATE/</td>
<td>P00D8</td>
<td>S0C34</td>
<td></td>
<td>10*</td>
</tr>
<tr>
<td></td>
<td>DATE/</td>
<td>P0805</td>
<td>S0C34</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Date last altered</td>
<td>DATM/</td>
<td>P00DF</td>
<td>S0C35</td>
<td>YES</td>
<td>10*</td>
</tr>
<tr>
<td></td>
<td>DATM/</td>
<td>P080D</td>
<td>S0C35</td>
<td>YES</td>
<td>10</td>
</tr>
<tr>
<td>Date requester notified</td>
<td>DATN/</td>
<td>P00E1</td>
<td>S0C48</td>
<td></td>
<td>10*</td>
</tr>
<tr>
<td></td>
<td>DATN/</td>
<td>P080E</td>
<td>S0C48</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Date required</td>
<td>DATD/</td>
<td>P00D5</td>
<td>S0C49</td>
<td>YES</td>
<td>10*</td>
</tr>
<tr>
<td></td>
<td>DATD/</td>
<td>P0804</td>
<td>S0C49</td>
<td>YES</td>
<td>10</td>
</tr>
<tr>
<td>Description</td>
<td></td>
<td></td>
<td>S0E0F</td>
<td></td>
<td>45</td>
</tr>
<tr>
<td>Device name</td>
<td>COMD/</td>
<td>P00C7</td>
<td>S0CA9</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Entry privilege class</td>
<td>CLAE/</td>
<td>P007A</td>
<td>S0BB1</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Estimated duration</td>
<td>INTE/</td>
<td>P015F</td>
<td>S0C95</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Estimated effort</td>
<td>EFE/</td>
<td>P0027</td>
<td>S0C94</td>
<td>YES</td>
<td>4</td>
</tr>
<tr>
<td>Initial priority</td>
<td>PRII/</td>
<td>P01B6</td>
<td>S0BE6</td>
<td>YES</td>
<td>2</td>
</tr>
<tr>
<td>Key item affected</td>
<td>COMK/</td>
<td>P00C8</td>
<td>S0CBF</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Location code</td>
<td>LOCC/</td>
<td>P016D</td>
<td>S0CC0</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Network name</td>
<td>COMN/</td>
<td>P00C9</td>
<td>S0CA3</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Owning privilege class</td>
<td>CLAO/</td>
<td>P007D</td>
<td>S0BB5</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Parent change number</td>
<td>RNOR/</td>
<td>P01ED</td>
<td>S0CD0</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Planned end date</td>
<td>DATT/</td>
<td>P00EA</td>
<td>S0C42</td>
<td>YES</td>
<td>10*</td>
</tr>
<tr>
<td></td>
<td>DATT/</td>
<td>P0814</td>
<td>S0C42</td>
<td>YES</td>
<td>10</td>
</tr>
<tr>
<td>Planned end time</td>
<td>TIMT/</td>
<td>P0298</td>
<td>S0C6F</td>
<td>YES</td>
<td>5</td>
</tr>
<tr>
<td>Planned start date</td>
<td>DATP/</td>
<td>P00E5</td>
<td>S0C41</td>
<td>YES</td>
<td>10*</td>
</tr>
<tr>
<td></td>
<td>DATP/</td>
<td>P0810</td>
<td>S0C41</td>
<td>YES</td>
<td>10</td>
</tr>
<tr>
<td>Planned start time</td>
<td>TIMP/</td>
<td>P0294</td>
<td>S0C6E</td>
<td>YES</td>
<td>5</td>
</tr>
</tbody>
</table>
### Activity Records—S0B07

*Table 28. Activity Records (continued).* An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prerequisites</td>
<td>RNCP/</td>
<td>P01DC</td>
<td>S0CD5</td>
<td></td>
<td>44</td>
</tr>
<tr>
<td>Program name</td>
<td>COMX/</td>
<td>P00CA</td>
<td>S0CA8</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Requested by</td>
<td>PERS/</td>
<td>P01AC</td>
<td>S0B59</td>
<td></td>
<td>40</td>
</tr>
<tr>
<td>Requester department</td>
<td>GROS/</td>
<td>P0150</td>
<td>S0B9B</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>Requester phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B2D</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Risk assessment</td>
<td>IMPR/</td>
<td>P015A</td>
<td>S0BF2</td>
<td>YES</td>
<td>6</td>
</tr>
<tr>
<td>System name</td>
<td>NASY/</td>
<td>P018F</td>
<td>S0CA5</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Time assigned</td>
<td>TIMA/</td>
<td>P0281</td>
<td>S0C64</td>
<td>YES</td>
<td>5</td>
</tr>
<tr>
<td>Time entered</td>
<td>TIME/</td>
<td>P028A</td>
<td>S0C61</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>Time last altered</td>
<td>TIMM/</td>
<td>P028F</td>
<td>S0C62</td>
<td>YES</td>
<td>5</td>
</tr>
<tr>
<td>Time required</td>
<td>TIMD/</td>
<td>P0287</td>
<td>S0C72</td>
<td>YES</td>
<td>5</td>
</tr>
<tr>
<td>Transfer-to class</td>
<td>CLAT/</td>
<td>P0081</td>
<td>S0BCC</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Unexpected problems</td>
<td>MISP/</td>
<td>P017E</td>
<td>S0ECB</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>User last altered</td>
<td>USER/</td>
<td>P02C3</td>
<td>S0B5E</td>
<td>YES</td>
<td>8</td>
</tr>
</tbody>
</table>

### Hardware Component Records—S0B0F

*Table 29. Hardware Component Records.* An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Center record ID</td>
<td>RNDR/</td>
<td>P01E1</td>
<td>S0CDE</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Circuit number</td>
<td>PH/</td>
<td>P000F</td>
<td>S0D12</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Component name/ID</td>
<td>RNID/</td>
<td>P01E9</td>
<td>S0CCF</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Component owner</td>
<td>DEPO/</td>
<td>P00EF</td>
<td>S0B9F</td>
<td>YES</td>
<td>11</td>
</tr>
<tr>
<td>Component status</td>
<td>STAC/</td>
<td>P026D</td>
<td>S0BEE</td>
<td>YES</td>
<td>7</td>
</tr>
<tr>
<td>Connection type</td>
<td>TYPX/</td>
<td>P00BA</td>
<td>S0C04</td>
<td></td>
<td>9</td>
</tr>
<tr>
<td>Contact department</td>
<td>GROC/</td>
<td>P014A</td>
<td>S0B9E</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>Contact name</td>
<td>PERC/</td>
<td>P01A9</td>
<td>S0BSC</td>
<td>YES</td>
<td>15</td>
</tr>
<tr>
<td>Contact phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B30</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Current book value</td>
<td>FEEB/</td>
<td>P01FD</td>
<td>S0DCA</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Current market value</td>
<td>FEEV/</td>
<td>P0200</td>
<td>S0DCB</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>CICS® ID</td>
<td>COMX/</td>
<td>P00CA</td>
<td>S0DOE</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Date entered</td>
<td>DATE/</td>
<td>P00D8</td>
<td>S0C34</td>
<td></td>
<td>10*</td>
</tr>
<tr>
<td>Date last altered</td>
<td>DATM/</td>
<td>P00DF</td>
<td>S0C35</td>
<td>YES</td>
<td>10*</td>
</tr>
<tr>
<td>Date of status</td>
<td>DATC/</td>
<td>P00D2</td>
<td>S0C37</td>
<td>YES</td>
<td>10*</td>
</tr>
</tbody>
</table>
Table 29. Hardware Component Records (continued). An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Date shipped</td>
<td>DATV/</td>
<td>P00EB</td>
<td>SOC45</td>
<td></td>
<td>10*</td>
</tr>
<tr>
<td>Description</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Device type and model</td>
<td>DEVS/</td>
<td>P010A</td>
<td>S097A</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Direction of flow</td>
<td>DIRF/</td>
<td>P0243</td>
<td>SOC05</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Display class</td>
<td>DISC/</td>
<td>P011A</td>
<td>S0D0B</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Entry privilege class</td>
<td>CLAE/</td>
<td>P007A</td>
<td>S0BB1</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Feature names</td>
<td>NAFE/</td>
<td>P0188</td>
<td>S0CBD</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Generic device</td>
<td>TYPD/</td>
<td>P02AA</td>
<td>S0C00</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Hardware EC levels</td>
<td>LVLS/</td>
<td>P0172</td>
<td>S14EC</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Hardware financial ID</td>
<td>RNFR/</td>
<td>P01E6</td>
<td>S0C0D</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Lease begin date</td>
<td>DATX/</td>
<td>P00ED</td>
<td>SOC46</td>
<td></td>
<td>10*</td>
</tr>
<tr>
<td>Lease end date</td>
<td>DATX/</td>
<td>P0818</td>
<td>SOC46</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Line interface address</td>
<td>TYPX/</td>
<td>P00BA</td>
<td>SOC01</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Line phone number</td>
<td>PH/</td>
<td>P000F</td>
<td>S0D1D</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Line protocol</td>
<td>TYPX/</td>
<td>P00BA</td>
<td>SOC00</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>Line set type</td>
<td>TYPX/</td>
<td>P00BA</td>
<td>SOC03</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Line speed</td>
<td>NUMX/</td>
<td>P0194</td>
<td>S0D1A</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>Line type</td>
<td>TYPX/</td>
<td>P00BA</td>
<td>SOC02</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Location code</td>
<td>LOCC/</td>
<td>P016D</td>
<td>SOC0C</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Loop speed</td>
<td>NUMS/</td>
<td>P0206</td>
<td>S0DF9</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>Loop type</td>
<td>TYPX/</td>
<td>P00BA</td>
<td>SOC06</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>LTERM ID for IMS</td>
<td>COMA/</td>
<td>P0312</td>
<td>S0BF0</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Maintenance interval</td>
<td>INTX/</td>
<td>P0162</td>
<td>SOC9B</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>Microcode EC level</td>
<td>LVXL/</td>
<td>P0178</td>
<td>SOE46</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Model link ID</td>
<td>RNMR/</td>
<td>P04AD</td>
<td>S0CE0</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Network name</td>
<td>COMN/</td>
<td>P00C9</td>
<td>S0CA3</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Node name</td>
<td>COMX/</td>
<td>P00CA</td>
<td>S0CB8</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Number of ports</td>
<td>NUMT/</td>
<td>P0247</td>
<td>S0D2D</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>Order number</td>
<td>NUMX/</td>
<td>P0194</td>
<td>S0D13</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>Owning privilege class</td>
<td>CLAO/</td>
<td>P007D</td>
<td>S0BB5</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Program name</td>
<td>COMX/</td>
<td>P00CA</td>
<td>S0CA8</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Purchase date</td>
<td>DATX/</td>
<td>P00ED</td>
<td>S0DED</td>
<td></td>
<td>10*</td>
</tr>
<tr>
<td></td>
<td>DATX/</td>
<td>P0818</td>
<td>S0DED</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Second service ID 1</td>
<td>RNSV/</td>
<td>P04B8</td>
<td>S1299</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Second service ID 2</td>
<td>RNSV/</td>
<td>P04B8</td>
<td>S129A</td>
<td></td>
<td>8</td>
</tr>
</tbody>
</table>
Table 29. Hardware Component Records (continued). An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Second support group 1</td>
<td>SSGP/</td>
<td>P04A9</td>
<td>S0B73</td>
<td></td>
<td>15</td>
</tr>
<tr>
<td>Second support group 2</td>
<td>SSGP/</td>
<td>P04A9</td>
<td>S0B74</td>
<td></td>
<td>15</td>
</tr>
<tr>
<td>Second system ID 1</td>
<td>RNSS/</td>
<td>P04B5</td>
<td>S1291</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Second system ID 2</td>
<td>RNSS/</td>
<td>P04B5</td>
<td>S1292</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Second system ID 3</td>
<td>RNSS/</td>
<td>P04B5</td>
<td>S1293</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Second system ID 4</td>
<td>RNSS/</td>
<td>P04B5</td>
<td>S1294</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Serial number</td>
<td>NUMX/</td>
<td>P0194</td>
<td>S0D17</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Service record ID</td>
<td>RNSR/</td>
<td>P01F6</td>
<td>S0CDC</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Subdiagram marker 1</td>
<td>SDM1/</td>
<td>P020A</td>
<td>S0BCC</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Subdiagram marker 2</td>
<td>SDM2/</td>
<td>P020C</td>
<td>S0BCC</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Subdiagram marker 3</td>
<td>SDM3/</td>
<td>P0235</td>
<td>S0BCC</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Subdiagram marker 4</td>
<td>SDM4/</td>
<td>P0237</td>
<td>S0BCC</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Subdiagram marker 5</td>
<td>SDM5/</td>
<td>P0239</td>
<td>S0BCC</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Subdiagram marker 6</td>
<td>SDM6/</td>
<td>P023B</td>
<td>S0BCC</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Subdiagram marker 7</td>
<td>SDM7/</td>
<td>P023D</td>
<td>S0BCC</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Subdiagram marker 8</td>
<td>SDM8/</td>
<td>P023F</td>
<td>S0BCC</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>System record ID</td>
<td>RNSY/</td>
<td>P01F9</td>
<td>S0CDB</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Time entered</td>
<td>TIME/</td>
<td>P028A</td>
<td>S0C61</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>Time last altered</td>
<td>TIMM/</td>
<td>P028F</td>
<td>S0C62</td>
<td>YES</td>
<td>5</td>
</tr>
<tr>
<td>Transfer-to class</td>
<td>CLAT/</td>
<td>P0081</td>
<td>S0BCC</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Up connected to</td>
<td>RNU1/</td>
<td>P01FE</td>
<td>S0CC8</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>User last altered</td>
<td>USER/</td>
<td>P02C3</td>
<td>S0B5E</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>VPA number</td>
<td>VPAN/</td>
<td>P046C</td>
<td>S0DF7</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>VPA sequence number</td>
<td>VPAS/</td>
<td>P046D</td>
<td>S0DF8</td>
<td></td>
<td>8</td>
</tr>
</tbody>
</table>

Hardware Connection Records—S0B1E

Table 30. Hardware Connection Records. An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cable length</td>
<td>NUMX/</td>
<td>P0194</td>
<td>S0DFB</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Cable number</td>
<td>NUMC/</td>
<td>P02F1</td>
<td>S0DF5</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Channel number</td>
<td>NUML/</td>
<td>P024B</td>
<td>S0DF4</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Component from</td>
<td>RNOR/</td>
<td>P01ED</td>
<td>S0C0D</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Component to</td>
<td>RNCT/</td>
<td>P04AB</td>
<td>S0CDF</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Connection record ID</td>
<td>RNID/</td>
<td>P01E9</td>
<td>S0CCF</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Connection status</td>
<td>STAC/</td>
<td>P026D</td>
<td>S0BEE</td>
<td>YES</td>
<td>7</td>
</tr>
<tr>
<td>Connection type</td>
<td>TYPX/</td>
<td>P00BA</td>
<td>S0C05</td>
<td>YES</td>
<td>8</td>
</tr>
</tbody>
</table>
### Hardware Connection Records—S0B1E

Table 30. **Hardware Connection Records (continued)**. An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Date entered</td>
<td>DATE/</td>
<td>P00D8</td>
<td>S0C34</td>
<td></td>
<td>10*</td>
</tr>
<tr>
<td></td>
<td>DATE/</td>
<td>P0805</td>
<td>S0C34</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Date from</td>
<td>DATX/</td>
<td>P00ED</td>
<td>S0DEB</td>
<td></td>
<td>10*</td>
</tr>
<tr>
<td></td>
<td>DATX/</td>
<td>P0818</td>
<td>S0DEB</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Date last altered</td>
<td>DATM/</td>
<td>P00DF</td>
<td>S0C35</td>
<td>YES</td>
<td>10*</td>
</tr>
<tr>
<td></td>
<td>DATM/</td>
<td>P080D</td>
<td>S0C35</td>
<td>YES</td>
<td>10</td>
</tr>
<tr>
<td>Date to</td>
<td>DATJ/</td>
<td>P01A5</td>
<td>S0DEC</td>
<td></td>
<td>10*</td>
</tr>
<tr>
<td></td>
<td>DATJ/</td>
<td>P08DA</td>
<td>S0DEC</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Description</td>
<td></td>
<td></td>
<td>S0E0F</td>
<td></td>
<td>45</td>
</tr>
<tr>
<td>Device address</td>
<td>NUMA/</td>
<td>P01A0</td>
<td>S0DFD</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Entry privilege class</td>
<td>CLAE/</td>
<td>P007A</td>
<td>S0BB1</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Generic device</td>
<td>TYPD/</td>
<td>P02AA</td>
<td>S0CB0</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Owning privilege class</td>
<td>CLAO/</td>
<td>P007D</td>
<td>S0BB5</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Path ID</td>
<td>CODL/</td>
<td>P0249</td>
<td>S0C07</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Port number</td>
<td>NUMN/</td>
<td>P019D</td>
<td>S0DF6</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Port ID</td>
<td>CODO/</td>
<td>P0136</td>
<td>S0BFB</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Shift number</td>
<td>NUMW/</td>
<td>P019B</td>
<td>S0DFC</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Time entered</td>
<td>TIME/</td>
<td>P028A</td>
<td>S0C61</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>Time last altered</td>
<td>TIMM/</td>
<td>P028F</td>
<td>S0C62</td>
<td>YES</td>
<td>5</td>
</tr>
<tr>
<td>Transfer-to class</td>
<td>CLAT/</td>
<td>P0081</td>
<td>S0BCC</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>User last altered</td>
<td>USER/</td>
<td>P02C3</td>
<td>S0B5E</td>
<td>YES</td>
<td>8</td>
</tr>
</tbody>
</table>

### Hardware Feature Records—S0B10

Table 31. **Hardware Feature Records**. An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Contact department</td>
<td>GROC/</td>
<td>P014A</td>
<td>S0B9E</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>Contact name</td>
<td>PERC/</td>
<td>P01A9</td>
<td>S0B5C</td>
<td>YES</td>
<td>15</td>
</tr>
<tr>
<td>Contact phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B30</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Date entered</td>
<td>DATE/</td>
<td>P00D8</td>
<td>S0C34</td>
<td></td>
<td>10*</td>
</tr>
<tr>
<td></td>
<td>DATE/</td>
<td>P0805</td>
<td>S0C34</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Date last altered</td>
<td>DATM/</td>
<td>P00DF</td>
<td>S0C35</td>
<td>YES</td>
<td>10*</td>
</tr>
<tr>
<td></td>
<td>DATM/</td>
<td>P080D</td>
<td>S0C35</td>
<td>YES</td>
<td>10</td>
</tr>
<tr>
<td>Date of status</td>
<td>DATC/</td>
<td>P00D2</td>
<td>S0C37</td>
<td>YES</td>
<td>10*</td>
</tr>
<tr>
<td></td>
<td>DATC/</td>
<td>P0803</td>
<td>S0C37</td>
<td>YES</td>
<td>10</td>
</tr>
<tr>
<td>Description</td>
<td></td>
<td></td>
<td>S0E0F</td>
<td></td>
<td>45</td>
</tr>
<tr>
<td>Entry privilege class</td>
<td>CLAE/</td>
<td>P007A</td>
<td>S0BB1</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Feature name</td>
<td>NAFE/</td>
<td>P0188</td>
<td>S0CBD</td>
<td></td>
<td>8</td>
</tr>
</tbody>
</table>
Table 31. Hardware Feature Records (continued). An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Feature number</td>
<td>NUMX/</td>
<td>P0194</td>
<td>S0D1E</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>Feature owner</td>
<td>DEPO/</td>
<td>P00EF</td>
<td>S0B9F</td>
<td>YES</td>
<td>11</td>
</tr>
<tr>
<td>Feature record ID</td>
<td>RNID/</td>
<td>P01E9</td>
<td>S0CCF</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Feature status</td>
<td>STAC/</td>
<td>P026D</td>
<td>S0BEE</td>
<td>YES</td>
<td>7</td>
</tr>
<tr>
<td>Feature type</td>
<td>TYPE/</td>
<td>P02AF</td>
<td>S0C09</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Financial record ID</td>
<td>RNFR/</td>
<td>P01E6</td>
<td>S0CDD</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Hardware EC levels</td>
<td>LVLS/</td>
<td>P0172</td>
<td>S14EC</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Lease begin date</td>
<td>DATX/</td>
<td>P00ED</td>
<td>S0C46</td>
<td></td>
<td>10*a</td>
</tr>
<tr>
<td></td>
<td>DATX/</td>
<td>P0818</td>
<td>S0C46</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Lease end date</td>
<td>DATX/</td>
<td>P00ED</td>
<td>S0C47</td>
<td></td>
<td>10*a</td>
</tr>
<tr>
<td></td>
<td>DATX/</td>
<td>P0818</td>
<td>S0C47</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Location code</td>
<td>LOCC/</td>
<td>P016D</td>
<td>S0C0C</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Microcode EC level</td>
<td>LVLX/</td>
<td>P0178</td>
<td>S0E46</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Owning privilege class</td>
<td>CLAO/</td>
<td>P007D</td>
<td>S0BB5</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Parent component ID</td>
<td>RNOR/</td>
<td>P01ED</td>
<td>S0CD0</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Purchase date</td>
<td>DATX/</td>
<td>P00ED</td>
<td>S0DED</td>
<td></td>
<td>10*a</td>
</tr>
<tr>
<td></td>
<td>DATX/</td>
<td>P0818</td>
<td>S0DED</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Serial number</td>
<td>NUMX/</td>
<td>P0194</td>
<td>S0D17</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Time entered</td>
<td>TIME/</td>
<td>P028A</td>
<td>S0C61</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>Time last altered</td>
<td>TIMM/</td>
<td>P028F</td>
<td>S0C62</td>
<td>YES</td>
<td>5</td>
</tr>
<tr>
<td>Transfer-to class</td>
<td>CLAT/</td>
<td>P0081</td>
<td>S0BCC</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>User last altered</td>
<td>USER/</td>
<td>P02C3</td>
<td>S0B5E</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>VPA number</td>
<td>VPAN/</td>
<td>P046C</td>
<td>S0DF7</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>VPA sequence number</td>
<td>VPAS/</td>
<td>P046D</td>
<td>S0DF8</td>
<td></td>
<td>8</td>
</tr>
</tbody>
</table>

Hardware Subcomponent Records—S0AF8

Table 32. Hardware Subcomponent Records. An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Center record ID</td>
<td>RNDR/</td>
<td>P01E1</td>
<td>S0CDE</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Contact department</td>
<td>GROC/</td>
<td>P014A</td>
<td>S0B9E</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>Contact name</td>
<td>PERC/</td>
<td>P01A9</td>
<td>S0B5C</td>
<td>YES</td>
<td>15</td>
</tr>
<tr>
<td>Contact phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B30</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Date entered</td>
<td>DATE/</td>
<td>P00D8</td>
<td>S0C34</td>
<td></td>
<td>10*a</td>
</tr>
<tr>
<td></td>
<td>DATE/</td>
<td>P0805</td>
<td>S0C34</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Date last altered</td>
<td>DATM/</td>
<td>P00DF</td>
<td>S0C35</td>
<td>YES</td>
<td>10*a</td>
</tr>
<tr>
<td></td>
<td>DATM/</td>
<td>P080D</td>
<td>S0C35</td>
<td>YES</td>
<td>10</td>
</tr>
</tbody>
</table>
Table 32. Hardware Subcomponent Records (continued). An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Date of status</td>
<td>DATC/</td>
<td>P00D2</td>
<td>S0C37</td>
<td>YES</td>
<td>10*</td>
</tr>
<tr>
<td>Description</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Date of status</td>
<td>DATC/</td>
<td>P0803</td>
<td>S0C37</td>
<td>YES</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Entry privilege class</td>
<td>CLAE/</td>
<td>P007A</td>
<td>S0BB1</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Hardware EC levels</td>
<td>LVLS/</td>
<td>P00172</td>
<td>S14EC</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Hardware financial ID</td>
<td>RNFR/</td>
<td>P001E6</td>
<td>S0CDD</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Hardware link ID</td>
<td>RNOC/</td>
<td>P04B0</td>
<td>S0CDA</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Lease begin date</td>
<td>DATX/</td>
<td>P00ED</td>
<td>S0C46</td>
<td></td>
<td>10*</td>
</tr>
<tr>
<td>Lease end date</td>
<td>DATX/</td>
<td>P0818</td>
<td>S0C46</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Lease begin date</td>
<td>DATX/</td>
<td>P00ED</td>
<td>S0C47</td>
<td></td>
<td>10*</td>
</tr>
<tr>
<td>Lease end date</td>
<td>DATX/</td>
<td>P0818</td>
<td>S0C47</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Location code</td>
<td>LOCC/</td>
<td>P016D</td>
<td>S0C0C</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Maintenance interval</td>
<td>INTX/</td>
<td>P00162</td>
<td>S0C9B</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>Microcode EC level</td>
<td>LVLT/</td>
<td>P00178</td>
<td>S0E46</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Owning privilege class</td>
<td>CLAO/</td>
<td>P007D</td>
<td>S0BB5</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Purchase date</td>
<td>DATX/</td>
<td>P00ED</td>
<td>S0DED</td>
<td></td>
<td>10*</td>
</tr>
<tr>
<td>Purchase date</td>
<td>DATX/</td>
<td>P0818</td>
<td>S0DED</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Serial number</td>
<td>NUMX/</td>
<td>P0194</td>
<td>S0D17</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Service record ID</td>
<td>RNSR/</td>
<td>P001F6</td>
<td>S0CDC</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Subcomponent owner</td>
<td>DEPO/</td>
<td>P00EF</td>
<td>S0B9F</td>
<td>YES</td>
<td>11</td>
</tr>
<tr>
<td>Subcomponent name/ID</td>
<td>RNID/</td>
<td>P001E9</td>
<td>S0CCF</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Subcomponent status</td>
<td>STAC/</td>
<td>P026D</td>
<td>S0BEE</td>
<td>YES</td>
<td>7</td>
</tr>
<tr>
<td>Subcomponent type</td>
<td>TYPE/</td>
<td>P02AF</td>
<td>S0C09</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>System record ID</td>
<td>RNSY/</td>
<td>P001F9</td>
<td>S0CDB</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Time entered</td>
<td>TIME/</td>
<td>P028A</td>
<td>S0C61</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>Time last altered</td>
<td>TIMM/</td>
<td>P028F</td>
<td>S0C62</td>
<td>YES</td>
<td>5</td>
</tr>
<tr>
<td>Transfer-to class</td>
<td>CLAT/</td>
<td>P0081</td>
<td>S0BCC</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>User last altered</td>
<td>USER/</td>
<td>P02C3</td>
<td>S0BC5</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>VPA number</td>
<td>VPAN/</td>
<td>P046C</td>
<td>S0DF7</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>VPA sequence number</td>
<td>VPAS/</td>
<td>P046D</td>
<td>S0DF8</td>
<td></td>
<td>8</td>
</tr>
</tbody>
</table>

Model Hardware Component Records—S0AF4

Table 33. Model Hardware Component Records. An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Center record ID</td>
<td>RNDR/</td>
<td>P01E1</td>
<td>S0CDE</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Circuit number</td>
<td>PH/</td>
<td>P000F</td>
<td>S0D12</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Component owner</td>
<td>DEPO/</td>
<td>P00EF</td>
<td>S0B9F</td>
<td>YES</td>
<td>11</td>
</tr>
</tbody>
</table>
Table 33. Model Hardware Component Records (continued). An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Component status</td>
<td>STAC/</td>
<td>P026D</td>
<td>S0BEE</td>
<td>YES</td>
<td>7</td>
</tr>
<tr>
<td>Connection type</td>
<td>TYPX/</td>
<td>P00BA</td>
<td>S0C04</td>
<td></td>
<td>9</td>
</tr>
<tr>
<td>Contact department</td>
<td>GROC/</td>
<td>P014A</td>
<td>S0B9E</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>Contact name</td>
<td>PERC/</td>
<td>P01A9</td>
<td>S0B5C</td>
<td>YES</td>
<td>15</td>
</tr>
<tr>
<td>Contact phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B30</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Current book value</td>
<td>FEEB/</td>
<td>P01FD</td>
<td>S0DCA</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Current market value</td>
<td>FEEV/</td>
<td>P0200</td>
<td>S0DCB</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Date entered</td>
<td>DATE/</td>
<td>P00D8</td>
<td>S0C34</td>
<td></td>
<td>10*</td>
</tr>
<tr>
<td></td>
<td>DATE/</td>
<td>P0805</td>
<td>S0C34</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Date last altered</td>
<td>DATM/</td>
<td>P00DF</td>
<td>S0C35</td>
<td>YES</td>
<td>10*</td>
</tr>
<tr>
<td></td>
<td>DATM/</td>
<td>P080D</td>
<td>S0C35</td>
<td>YES</td>
<td>10</td>
</tr>
<tr>
<td>Date of status</td>
<td>DATC/</td>
<td>P00D2</td>
<td>S0C37</td>
<td>YES</td>
<td>10*</td>
</tr>
<tr>
<td></td>
<td>DATC/</td>
<td>P0803</td>
<td>S0C37</td>
<td>YES</td>
<td>10</td>
</tr>
<tr>
<td>Description</td>
<td></td>
<td>S0E0F</td>
<td></td>
<td></td>
<td>45</td>
</tr>
<tr>
<td>Device type and model</td>
<td>DEVS/</td>
<td>P010A</td>
<td>S097A</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Direction of flow</td>
<td>DIRF/</td>
<td>P0243</td>
<td>S0CE5</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Display class</td>
<td>DISC/</td>
<td>P011A</td>
<td>S0D0B</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Entry privilege class</td>
<td>CLAE/</td>
<td>P007A</td>
<td>S0BB1</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Feature names</td>
<td>NAFE/</td>
<td>P0188</td>
<td>S0CBD</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Generic device</td>
<td>TYPD/</td>
<td>P02AA</td>
<td>S0CB0</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Hardware financial ID</td>
<td>RNFR/</td>
<td>P01E6</td>
<td>S0CDD</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Line interface address</td>
<td>TYPX/</td>
<td>P00BA</td>
<td>S0C01</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Line phone number</td>
<td>PH/</td>
<td>P000F</td>
<td>S0D1D</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Line protocol</td>
<td>TYPX/</td>
<td>P00BA</td>
<td>S0C00</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>Line set type</td>
<td>TYPX/</td>
<td>P00BA</td>
<td>S0C03</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>Line speed</td>
<td>NUMX/</td>
<td>P0194</td>
<td>S0D1A</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>Line type</td>
<td>TYPX/</td>
<td>P00BA</td>
<td>S0C02</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Loop speed</td>
<td>NUMS/</td>
<td>P0206</td>
<td>S0DF9</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>Loop type</td>
<td>TYPX/</td>
<td>P00BA</td>
<td>S0C06</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Loop ID</td>
<td>CODA/</td>
<td>P0312</td>
<td>S0BFC</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Maintenance interval</td>
<td>INTX/</td>
<td>P0162</td>
<td>S0C9B</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>Model component name/ID</td>
<td>RNID/</td>
<td>P01E9</td>
<td>S0CCF</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Owning privilege class</td>
<td>CLAO/</td>
<td>P007D</td>
<td>S0BB5</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Second service ID 1</td>
<td>RNSV/</td>
<td>P04B8</td>
<td>S1299</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Second service ID 2</td>
<td>RNSV/</td>
<td>P04B8</td>
<td>S129A</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Second support group 1</td>
<td>SSGP/</td>
<td>P04A9</td>
<td>S0B73</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Second support group 2</td>
<td>SSGP/</td>
<td>P04A9</td>
<td>S0B74</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Second system ID 1</td>
<td>RNSS/</td>
<td>P04B5</td>
<td>S1291</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Second system ID 2</td>
<td>RNSS/</td>
<td>P04B5</td>
<td>S1292</td>
<td></td>
<td>8</td>
</tr>
</tbody>
</table>
### Model Hardware Component Records—S0AF4

Table 33. Model Hardware Component Records (continued). An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Second system ID 3</td>
<td>RNSS/</td>
<td>P04B5</td>
<td>S1293</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Second system ID 4</td>
<td>RNSS/</td>
<td>P04B5</td>
<td>S1294</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Service record ID</td>
<td>RNSR/</td>
<td>P01F6</td>
<td>S0CDC</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>System record ID</td>
<td>RNSY/</td>
<td>P01F9</td>
<td>S0CDB</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Time entered</td>
<td>TIME/</td>
<td>P028A</td>
<td>S0C61</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>Time last altered</td>
<td>TIMM/</td>
<td>P028F</td>
<td>S0C62</td>
<td>YES</td>
<td>5</td>
</tr>
<tr>
<td>Transfer-to class</td>
<td>CLAT/</td>
<td>P0081</td>
<td>S0BCC</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>User last altered</td>
<td>USER/</td>
<td>P02C3</td>
<td>S0B5E</td>
<td>YES</td>
<td>8</td>
</tr>
</tbody>
</table>

### Model Hardware Feature Records—S0AF5

Table 34. Model Hardware Feature Records. An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Contact department</td>
<td>GROC/</td>
<td>P014A</td>
<td>S0B9E</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>Contact name</td>
<td>PERC/</td>
<td>P01A9</td>
<td>S0B5C</td>
<td>YES</td>
<td>15</td>
</tr>
<tr>
<td>Contact phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B30</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Date entered</td>
<td>DATE/</td>
<td>P00D8</td>
<td>S0C34</td>
<td></td>
<td>10*</td>
</tr>
<tr>
<td>Date last altered</td>
<td>DATM/</td>
<td>P00DF</td>
<td>S0C35</td>
<td>YES</td>
<td>10*</td>
</tr>
<tr>
<td>Date of status</td>
<td>DATC/</td>
<td>P00D2</td>
<td>S0C37</td>
<td>YES</td>
<td>10*</td>
</tr>
<tr>
<td>Description</td>
<td></td>
<td></td>
<td>S0E0F</td>
<td></td>
<td>45</td>
</tr>
<tr>
<td>Entry privilege class</td>
<td>CLAE/</td>
<td>P007A</td>
<td>S0BB1</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Feature name</td>
<td>NAFE/</td>
<td>P0188</td>
<td>S0CBD</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Feature number</td>
<td>NUMX/</td>
<td>P0194</td>
<td>S0D1E</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>Feature owner</td>
<td>DEPO/</td>
<td>P00EF</td>
<td>S0B9F</td>
<td>YES</td>
<td>11</td>
</tr>
<tr>
<td>Feature status</td>
<td>STAC/</td>
<td>P026D</td>
<td>S0BEE</td>
<td>YES</td>
<td>7</td>
</tr>
<tr>
<td>Feature type</td>
<td>TYPE/</td>
<td>P02AF</td>
<td>S0C09</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Financial record ID</td>
<td>RNFR/</td>
<td>P01E6</td>
<td>S0CDD</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Model feature record ID</td>
<td>RNID/</td>
<td>P01E9</td>
<td>S0CCF</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Owning privilege class</td>
<td>CLAO/</td>
<td>P007D</td>
<td>S0BB5</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Parent component ID</td>
<td>RNOR/</td>
<td>P01ED</td>
<td>S0CD0</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Storage class</td>
<td>TYPS/</td>
<td>P04A3</td>
<td>S0C17</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>Time entered</td>
<td>TIME/</td>
<td>P028A</td>
<td>S0C61</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>Time last altered</td>
<td>TIMM/</td>
<td>P028F</td>
<td>S0C62</td>
<td>YES</td>
<td>5</td>
</tr>
<tr>
<td>Transfer-to class</td>
<td>CLAT/</td>
<td>P0081</td>
<td>S0BCC</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>User last altered</td>
<td>USER/</td>
<td>P02C3</td>
<td>S0B5E</td>
<td>YES</td>
<td>8</td>
</tr>
</tbody>
</table>
Model Hardware Subcomponent Records—S0AF6

Table 35. Model Hardware Subcomponent Records. An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Center record ID</td>
<td>RNDR/</td>
<td>P01E1</td>
<td>S0CDE</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Contact department</td>
<td>GROC/</td>
<td>P014A</td>
<td>S0B9E</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>Contact name</td>
<td>PERC/</td>
<td>P01A9</td>
<td>S0B5C</td>
<td>YES</td>
<td>15</td>
</tr>
<tr>
<td>Contact phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B30</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Date entered</td>
<td>DATE/</td>
<td>P00D8</td>
<td>S0C34</td>
<td>10*</td>
<td></td>
</tr>
<tr>
<td></td>
<td>DATE/</td>
<td>P0805</td>
<td>S0C34</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Date last altered</td>
<td>DATM/</td>
<td>P00DF</td>
<td>S0C35</td>
<td>YES</td>
<td>10*</td>
</tr>
<tr>
<td></td>
<td>DATM/</td>
<td>P080D</td>
<td>S0C35</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Date of status</td>
<td>DATC/</td>
<td>P00D2</td>
<td>S0C37</td>
<td>YES</td>
<td>10*</td>
</tr>
<tr>
<td></td>
<td>DATC/</td>
<td>P0803</td>
<td>S0C37</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Description</td>
<td></td>
<td></td>
<td></td>
<td>S0E0F</td>
<td>45</td>
</tr>
<tr>
<td>Entry privilege class</td>
<td>CLAE/</td>
<td>P007A</td>
<td>S0BB1</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Hardware financial ID</td>
<td>RNFR/</td>
<td>P01E6</td>
<td>S0CDD</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Maintenance interval</td>
<td>INTX/</td>
<td>P0162</td>
<td>S0C9B</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>Model link ID</td>
<td>RNOC/</td>
<td>P04B0</td>
<td>S0CDA</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Model subcomponent name/ID</td>
<td>RNID/</td>
<td>P01E9</td>
<td>S0CCF</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Owning privilege class</td>
<td>CLAO/</td>
<td>P007D</td>
<td>S0BB5</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Service record ID</td>
<td>RNSR/</td>
<td>P01F6</td>
<td>S0CDC</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Subcomponent owner</td>
<td>DEPO/</td>
<td>P00EF</td>
<td>S0B9F</td>
<td>YES</td>
<td>11</td>
</tr>
<tr>
<td>Subcomponent status</td>
<td>STAC/</td>
<td>P026D</td>
<td>S0BEE</td>
<td>YES</td>
<td>7</td>
</tr>
<tr>
<td>Subcomponent type</td>
<td>TYPE/</td>
<td>P02AF</td>
<td>S0C09</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>System record ID</td>
<td>RNSY/</td>
<td>P01F9</td>
<td>S0CDB</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Time entered</td>
<td>TIME/</td>
<td>P028A</td>
<td>S0C61</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>Time last altered</td>
<td>TIMM/</td>
<td>P028F</td>
<td>S0C62</td>
<td>YES</td>
<td>5</td>
</tr>
<tr>
<td>Transfer-to class</td>
<td>CLAT/</td>
<td>P0081</td>
<td>S0BCC</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>User last altered</td>
<td>USER/</td>
<td>P02C3</td>
<td>S0B5E</td>
<td>YES</td>
<td>8</td>
</tr>
</tbody>
</table>

Software Component Records—S0B13

Table 36. Software Component Records. An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Center record ID</td>
<td>RNDR/</td>
<td>P01E1</td>
<td>S0CDE</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Component owner</td>
<td>DEPO/</td>
<td>P00EF</td>
<td>S0B9F</td>
<td>YES</td>
<td>11</td>
</tr>
<tr>
<td>Component name/ID</td>
<td>RNID/</td>
<td>P01E9</td>
<td>S0CCF</td>
<td></td>
<td>8</td>
</tr>
</tbody>
</table>
Table 36. Software Component Records (continued). An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Component status</td>
<td>STAC/</td>
<td>P026D</td>
<td>S0BEE</td>
<td>YES</td>
<td>7</td>
</tr>
<tr>
<td>Contact department</td>
<td>GROC/</td>
<td>P014A</td>
<td>S0B9E</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>Contact name</td>
<td>PERC/</td>
<td>P01A9</td>
<td>S0B5C</td>
<td>YES</td>
<td>15</td>
</tr>
<tr>
<td>Contact phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B30</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>CICS transaction ID</td>
<td>NUMX/</td>
<td>P0194</td>
<td>S0D1B</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Date entered</td>
<td>DATE/</td>
<td>P00D8</td>
<td>S0C34</td>
<td></td>
<td>10*</td>
</tr>
<tr>
<td>Date last altered</td>
<td>DATM/</td>
<td>P00DF</td>
<td>S0C35</td>
<td>YES</td>
<td>10*</td>
</tr>
<tr>
<td>Date of status</td>
<td>DATC/</td>
<td>P00D2</td>
<td>S0C37</td>
<td>YES</td>
<td>10*</td>
</tr>
<tr>
<td>Description</td>
<td></td>
<td></td>
<td>S0E0F</td>
<td></td>
<td>45</td>
</tr>
<tr>
<td>Display class</td>
<td>DISC/</td>
<td>P011A</td>
<td>S0D0B</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Entry privilege class</td>
<td>CLAE/</td>
<td>P007A</td>
<td>S0BB1</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Execution type</td>
<td>TYPR/</td>
<td>P02B5</td>
<td></td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Feature names</td>
<td>NAFE/</td>
<td>P0188</td>
<td>S0CBD</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Fix level</td>
<td>LEVF/</td>
<td>P0169</td>
<td>S0639</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>IMS transaction ID</td>
<td>NUMX/</td>
<td>P0194</td>
<td>S0D1C</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>License begin date</td>
<td>DATX/</td>
<td>P00ED</td>
<td>S0C46</td>
<td></td>
<td>10*</td>
</tr>
<tr>
<td>License end date</td>
<td>DATX/</td>
<td>P0818</td>
<td>S0C46</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Location code</td>
<td>LOCC/</td>
<td>P016D</td>
<td>S0C0C</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Model link ID</td>
<td>RNMR/</td>
<td>P04AD</td>
<td>S0CE0</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Modification level</td>
<td>LEVM/</td>
<td>P016A</td>
<td>S0ECE</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>Network name</td>
<td>COMN/</td>
<td>P00C9</td>
<td>S0CA3</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Node name</td>
<td>COMX/</td>
<td>P00CA</td>
<td>S0CB8</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Order number</td>
<td>NUMX/</td>
<td>P0194</td>
<td>S0D13</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>Owning privilege class</td>
<td>CLAO/</td>
<td>P007D</td>
<td>S0BB5</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Program name</td>
<td>COMX/</td>
<td>P00CA</td>
<td>S0CA8</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Program type</td>
<td>TYPP/</td>
<td>P02B1</td>
<td>S0CAF</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Program version</td>
<td>LVLX/</td>
<td>P0178</td>
<td>S0ECD</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>Release level</td>
<td>LVLS/</td>
<td>P0172</td>
<td>S0636</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>Second service ID 1</td>
<td>RNSV/</td>
<td>P04B8</td>
<td>S1299</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Second service ID 2</td>
<td>RNSV/</td>
<td>P04B8</td>
<td>S129A</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Second support group 1</td>
<td>SSGP/</td>
<td>P04A9</td>
<td>S0B73</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Second support group 2</td>
<td>SSGP/</td>
<td>P04A9</td>
<td>S0B74</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Second system ID 1</td>
<td>RNSS/</td>
<td>P04B5</td>
<td>S1291</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Second system ID 2</td>
<td>RNSS/</td>
<td>P04B5</td>
<td>S1292</td>
<td></td>
<td>8</td>
</tr>
</tbody>
</table>
### Software Component Records—S0B13

*Table 36. Software Component Records (continued).* An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Second system ID 3</td>
<td>RNSS/</td>
<td>P04B5</td>
<td>S1293</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Second system ID 4</td>
<td>RNSS/</td>
<td>P04B5</td>
<td>S1294</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Service record ID</td>
<td>RNSR/</td>
<td>P01F6</td>
<td>S0CDC</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Software financial ID</td>
<td>RNFR/</td>
<td>P01E6</td>
<td>S0CDD</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Software FMID</td>
<td>LVLS/</td>
<td>P0172</td>
<td>S14EB</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Source language</td>
<td>MISX/</td>
<td>P0180</td>
<td>S0ECC</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>System record ID</td>
<td>RNSY/</td>
<td>P01F9</td>
<td>S0CDB</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Time entered</td>
<td>TIME/</td>
<td>P028A</td>
<td>S0C61</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>Time last altered</td>
<td>TIMM/</td>
<td>P028F</td>
<td>S0C62</td>
<td>YES</td>
<td>5</td>
</tr>
<tr>
<td>Transfer-to class</td>
<td>CLAT/</td>
<td>P0081</td>
<td>S0BBC</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Up connected to</td>
<td>RNU1/</td>
<td>P01FE</td>
<td>S0CC8</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>User last altered</td>
<td>USER/</td>
<td>P02C3</td>
<td>S0B5E</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Vendor component ID</td>
<td>NUMX/</td>
<td>P0194</td>
<td>S0D11</td>
<td></td>
<td>11</td>
</tr>
</tbody>
</table>

### Software Connection Records—S0B1F

*Table 37. Software Connection Records.* An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Component from</td>
<td>RNOR/</td>
<td>P01ED</td>
<td>S0CD0</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Component to</td>
<td>RNCT/</td>
<td>P04AB</td>
<td>S0CDE</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Connection record ID</td>
<td>RNID/</td>
<td>P01E9</td>
<td>S0CCF</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Connection status</td>
<td>STAC/</td>
<td>P026D</td>
<td>S0BEE</td>
<td>YES</td>
<td>7</td>
</tr>
<tr>
<td>Connection type</td>
<td>TYPX/</td>
<td>P00BA</td>
<td>S0C05</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Date entered</td>
<td>DATE/</td>
<td>P00D8</td>
<td>S0C34</td>
<td></td>
<td>10*</td>
</tr>
<tr>
<td>Date from</td>
<td>DATE/</td>
<td>P0805</td>
<td>S0C34</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Date last altered</td>
<td>DATM/</td>
<td>P080D</td>
<td>S0C35</td>
<td>YES</td>
<td>10*</td>
</tr>
<tr>
<td>Date to</td>
<td>DATJ/</td>
<td>P01A5</td>
<td>S0DEC</td>
<td></td>
<td>10*</td>
</tr>
<tr>
<td>Description</td>
<td></td>
<td></td>
<td>S0E0F</td>
<td></td>
<td>45</td>
</tr>
<tr>
<td>Device address</td>
<td>NUMA/</td>
<td>P01A0</td>
<td>S0DFD</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Entry privilege class</td>
<td>CLAE/</td>
<td>P007A</td>
<td>S0BB1</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Owning privilege class</td>
<td>CLAO/</td>
<td>P007D</td>
<td>S0BB5</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Program type</td>
<td>TYPP/</td>
<td>P02B1</td>
<td>S0CAF</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Shift number</td>
<td>NUMW/</td>
<td>P019B</td>
<td>S0DFC</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Time entered</td>
<td>TIME/</td>
<td>P028A</td>
<td>S0C61</td>
<td></td>
<td>5</td>
</tr>
</tbody>
</table>
Table 37. Software Connection Records (continued). An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time last altered</td>
<td>TIMM/</td>
<td>P028F</td>
<td>S0C62</td>
<td>YES</td>
<td>5</td>
</tr>
<tr>
<td>Transfer-to class</td>
<td>CLAT/</td>
<td>P0081</td>
<td>S0BCC</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>User last altered</td>
<td>USER/</td>
<td>P02C3</td>
<td>S0B5E</td>
<td>YES</td>
<td>8</td>
</tr>
</tbody>
</table>

Software Feature Records—S0B14

Table 38. Software Feature Records. An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Contact department</td>
<td>GROC/</td>
<td>P014A</td>
<td>S0B9E</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>Contact name</td>
<td>PERC/</td>
<td>P01A9</td>
<td>S0B5C</td>
<td>YES</td>
<td>15</td>
</tr>
<tr>
<td>Contact phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B30</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Date entered</td>
<td>DATE/</td>
<td>P00D8</td>
<td>S0C34</td>
<td>10*</td>
<td></td>
</tr>
<tr>
<td></td>
<td>DATE/</td>
<td>P0805</td>
<td>S0C34</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Date last altered</td>
<td>DATM/</td>
<td>P00DF</td>
<td>S0C35</td>
<td>YES</td>
<td>10*</td>
</tr>
<tr>
<td></td>
<td>DATM/</td>
<td>P080D</td>
<td>S0C35</td>
<td>YES</td>
<td>10</td>
</tr>
<tr>
<td>Date of status</td>
<td>DATC/</td>
<td>P00D2</td>
<td>S0C37</td>
<td>YES</td>
<td>10*</td>
</tr>
<tr>
<td></td>
<td>DATC/</td>
<td>P0803</td>
<td>S0C37</td>
<td>YES</td>
<td>10</td>
</tr>
<tr>
<td>Description</td>
<td></td>
<td></td>
<td>S0E0F</td>
<td></td>
<td>45</td>
</tr>
<tr>
<td>Entry privilege class</td>
<td>CLAE/</td>
<td>P007A</td>
<td>S0BB1</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Feature name</td>
<td>NAFE/</td>
<td>P0188</td>
<td>S0CBD</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Feature owner</td>
<td>DEPO/</td>
<td>P00EF</td>
<td>S0B9F</td>
<td>YES</td>
<td>11</td>
</tr>
<tr>
<td>Feature record ID</td>
<td>RNID/</td>
<td>P01E9</td>
<td>S0CCF</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Feature status</td>
<td>STAC/</td>
<td>P026D</td>
<td>S0BEE</td>
<td>YES</td>
<td>7</td>
</tr>
<tr>
<td>Feature type</td>
<td>TYPE/</td>
<td>P02AF</td>
<td>S0C09</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Financial record ID</td>
<td>RNFR/</td>
<td>P01E6</td>
<td>S0CDD</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Fix level</td>
<td>LEVF/</td>
<td>P0169</td>
<td>S0639</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>License begin date</td>
<td>DATX/</td>
<td>P00ED</td>
<td>S0C46</td>
<td>10*</td>
<td></td>
</tr>
<tr>
<td></td>
<td>DATX/</td>
<td>P0818</td>
<td>S0C46</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>License end date</td>
<td>DATX/</td>
<td>P00ED</td>
<td>S0C47</td>
<td>10*</td>
<td></td>
</tr>
<tr>
<td>Location code</td>
<td>LOCC/</td>
<td>P016D</td>
<td>S0COC</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Modification level</td>
<td>LEVM/</td>
<td>P016A</td>
<td>S0ECE</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>Owning privilege class</td>
<td>CLAO/</td>
<td>P007D</td>
<td>S0BB5</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Parent component ID</td>
<td>RNOR/</td>
<td>P01ED</td>
<td>S0CDD</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Program version</td>
<td>LV LX/</td>
<td>P0178</td>
<td>S0ECD</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>Release level</td>
<td>LVLS/</td>
<td>P0172</td>
<td>S0636</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>Software FMID</td>
<td>LVLS/</td>
<td>P0172</td>
<td>S14EB</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Source language</td>
<td>MISX/</td>
<td>P0180</td>
<td>S0ECC</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Time entered</td>
<td>TIME/</td>
<td>P028A</td>
<td>S0C61</td>
<td></td>
<td>5</td>
</tr>
</tbody>
</table>
Table 38. Software Feature Records (continued). An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time last altered</td>
<td>TIMM/</td>
<td>P028F</td>
<td>S0C62</td>
<td>YES</td>
<td>5</td>
</tr>
<tr>
<td>Transfer-to class</td>
<td>CLAT/</td>
<td>P0081</td>
<td>S0BCC</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>User last altered</td>
<td>USER/</td>
<td>P02C3</td>
<td>S0B5E</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Vendor component ID</td>
<td>NUMX/</td>
<td>P0194</td>
<td>S0D11</td>
<td></td>
<td>11</td>
</tr>
</tbody>
</table>

Model Software Component Records—S0AFB

Table 39. Model Software Component Records. An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Center record ID</td>
<td>RNDR/</td>
<td>P01E1</td>
<td>S0CDE</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Component owner</td>
<td>DEPO/</td>
<td>P00EF</td>
<td>S0B9F</td>
<td>YES</td>
<td>11</td>
</tr>
<tr>
<td>Component status</td>
<td>STAC/</td>
<td>P026D</td>
<td>S0BEE</td>
<td>YES</td>
<td>7</td>
</tr>
<tr>
<td>Contact department</td>
<td>GROC/</td>
<td>P014A</td>
<td>S0B9E</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>Contact name</td>
<td>PERC/</td>
<td>P01A9</td>
<td>S0B5C</td>
<td>YES</td>
<td>15</td>
</tr>
<tr>
<td>Contact phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B30</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Date entered</td>
<td>DATE/</td>
<td>P00D8</td>
<td>S0C34</td>
<td>10*</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>DATE/</td>
<td>P0805</td>
<td>S0C34</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Date last altered</td>
<td>DATM/</td>
<td>P00DF</td>
<td>S0C35</td>
<td>10*</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>DATM/</td>
<td>P080D</td>
<td>S0C35</td>
<td>YES</td>
<td>10</td>
</tr>
<tr>
<td>Date of status</td>
<td>DATC/</td>
<td>P00D2</td>
<td>S0C37</td>
<td>10*</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>DATC/</td>
<td>P0803</td>
<td>S0C37</td>
<td>YES</td>
<td>10</td>
</tr>
<tr>
<td>Description</td>
<td></td>
<td></td>
<td>S0E0F</td>
<td></td>
<td>45</td>
</tr>
<tr>
<td>Display class</td>
<td>DISC/</td>
<td>P011A</td>
<td>S0D0B</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Entry privilege class</td>
<td>CLAE/</td>
<td>P007A</td>
<td>S0BB1</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Execution type</td>
<td>TYPR/</td>
<td>P02B5</td>
<td></td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Feature names</td>
<td>NAFE/</td>
<td>P0188</td>
<td>S0CBD</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Fix level</td>
<td>LEVF/</td>
<td>P0169</td>
<td>S0639</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Model component name/ID</td>
<td>RNID/</td>
<td>P01E9</td>
<td>S0CCF</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Modification level</td>
<td>LEVM/</td>
<td>P016A</td>
<td>S0ECE</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>Owning privilege class</td>
<td>CLAO/</td>
<td>P007D</td>
<td>S0BB5</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Program type</td>
<td>TYPF/</td>
<td>P02B1</td>
<td>S0CAF</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Program version</td>
<td>LVLX/</td>
<td>P0178</td>
<td>S0ECD</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>Release level</td>
<td>LVLS/</td>
<td>P0172</td>
<td>S0636</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>Second service ID 1</td>
<td>RNSV/</td>
<td>P04B8</td>
<td>S1299</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Second service ID 2</td>
<td>RNSV/</td>
<td>P04B8</td>
<td>S129A</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Second support group 1</td>
<td>SSGP/</td>
<td>P04A9</td>
<td>S0B73</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Second support group 2</td>
<td>SSGP/</td>
<td>P04A9</td>
<td>S0B74</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Second system ID 1</td>
<td>RNSS/</td>
<td>P04B5</td>
<td>S1291</td>
<td></td>
<td>8</td>
</tr>
</tbody>
</table>
Table 39. Model Software Component Records (continued). An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Second system ID 2</td>
<td>RNSS/</td>
<td>P04B5</td>
<td>S1292</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Second system ID 3</td>
<td>RNSS/</td>
<td>P04B5</td>
<td>S1293</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Second system ID 4</td>
<td>RNSS/</td>
<td>P04B5</td>
<td>S1294</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Service record ID</td>
<td>RNSR/</td>
<td>P01F6</td>
<td>S0CDC</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Software financial ID</td>
<td>RNFR/</td>
<td>P01E6</td>
<td>S0CDD</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Source language</td>
<td>MISX/</td>
<td>P0180</td>
<td>S0ECC</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>System record ID</td>
<td>RNSY/</td>
<td>P01F9</td>
<td>S0CDB</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Time entered</td>
<td>TIME/</td>
<td>P028A</td>
<td>S0C61</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>Time last altered</td>
<td>TIMM/</td>
<td>P028F</td>
<td>S0C62</td>
<td>YES</td>
<td>5</td>
</tr>
<tr>
<td>Transfer-to class</td>
<td>CLAT/</td>
<td>P0081</td>
<td>S0BCC</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>User last altered</td>
<td>USER/</td>
<td>P02C3</td>
<td>S0B5E</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Vendor component ID</td>
<td>NUMX/</td>
<td>P0194</td>
<td>S0D11</td>
<td></td>
<td>11</td>
</tr>
</tbody>
</table>

Model Software Feature Records—S0AFC

Table 40. Model Software Feature Records. An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Contact department</td>
<td>GROC/</td>
<td>P014A</td>
<td>S0B9E</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>Contact name</td>
<td>PERC/</td>
<td>P01A9</td>
<td>S0B5C</td>
<td>YES</td>
<td>15</td>
</tr>
<tr>
<td>Contact phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B30</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Date entered</td>
<td>DATE/</td>
<td>P00D8</td>
<td>S0C34</td>
<td></td>
<td>10*</td>
</tr>
<tr>
<td>Date last altered</td>
<td>DATM/</td>
<td>P00DF</td>
<td>S0C35</td>
<td>YES</td>
<td>10*</td>
</tr>
<tr>
<td>Date of status</td>
<td>DATC/</td>
<td>P00D2</td>
<td>S0C37</td>
<td>YES</td>
<td>10*</td>
</tr>
<tr>
<td>Description</td>
<td></td>
<td></td>
<td>S0E0F</td>
<td></td>
<td>45</td>
</tr>
<tr>
<td>Entry privilege class</td>
<td>CLAE/</td>
<td>P007A</td>
<td>S0BB1</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Feature name</td>
<td>NAFE/</td>
<td>P0188</td>
<td>S0CBD</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Feature owner</td>
<td>DEPO/</td>
<td>P00EF</td>
<td>S0B9F</td>
<td>YES</td>
<td>11</td>
</tr>
<tr>
<td>Feature record ID</td>
<td>RNID/</td>
<td>P01E9</td>
<td>S0CCF</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Feature status</td>
<td>STAC/</td>
<td>P026D</td>
<td>S0BEE</td>
<td>YES</td>
<td>7</td>
</tr>
<tr>
<td>Feature type</td>
<td>TYPE/</td>
<td>P02AF</td>
<td>S0C09</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Financial record ID</td>
<td>RNFR/</td>
<td>P01E6</td>
<td>S0CDD</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Fix level</td>
<td>LEVF/</td>
<td>P0169</td>
<td>S0639</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Modification level</td>
<td>LEVM/</td>
<td>P016A</td>
<td>S0ECE</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>Owning privilege class</td>
<td>CLAO/</td>
<td>P007D</td>
<td>S0BB5</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Parent component ID</td>
<td>RNOR/</td>
<td>P01ED</td>
<td>S0CDD</td>
<td></td>
<td>8</td>
</tr>
</tbody>
</table>
**Model Software Feature Records—S0AFC**

*Table 40. Model Software Feature Records (continued).* An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Program version</td>
<td>LVLX/</td>
<td>P0178</td>
<td>S0ECD</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>Release level</td>
<td>LVLS/</td>
<td>P0172</td>
<td>S0636</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>Source language</td>
<td>MISX/</td>
<td>P0180</td>
<td>S0ECC</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Storage class</td>
<td>TYPX/</td>
<td>P04A3</td>
<td>S0C17</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>Time entered</td>
<td>TIME/</td>
<td>P028A</td>
<td>S0C61</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>Time last altered</td>
<td>TIMM/</td>
<td>P028F</td>
<td>S0C62</td>
<td>YES</td>
<td>5</td>
</tr>
<tr>
<td>Transfer-to class</td>
<td>CLAT/</td>
<td>P0081</td>
<td>S0BCC</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>User last altered</td>
<td>USER/</td>
<td>P02C3</td>
<td>S0B5E</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Vendor component ID</td>
<td>NUMX/</td>
<td>P0194</td>
<td>S0D11</td>
<td></td>
<td>11</td>
</tr>
</tbody>
</table>

**Hardware Financial Records—S0B1B**

*Table 41. Hardware Financial Records.* An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Charge out account</td>
<td>NUMX/</td>
<td>P0194</td>
<td>S0D15</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>Charge out rate</td>
<td>NUMX/</td>
<td>P0194</td>
<td>S0DB1</td>
<td></td>
<td>6</td>
</tr>
<tr>
<td>Component count</td>
<td>COUC/</td>
<td>P00CB</td>
<td>S0D29</td>
<td>YES</td>
<td>3</td>
</tr>
<tr>
<td>Date entered</td>
<td>DATE/</td>
<td>P00D8</td>
<td>S0C34</td>
<td></td>
<td>10*</td>
</tr>
<tr>
<td></td>
<td>DATE/</td>
<td>P0805</td>
<td>S0C34</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Date last altered</td>
<td>DATM/</td>
<td>P00DF</td>
<td>S0C35</td>
<td>YES</td>
<td>10*</td>
</tr>
<tr>
<td></td>
<td>DATM/</td>
<td>P080D</td>
<td>S0C35</td>
<td>YES</td>
<td>10</td>
</tr>
<tr>
<td>Depreciation method</td>
<td>TYPX/</td>
<td>P00BA</td>
<td>S0C15</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Depreciation period</td>
<td>INTX/</td>
<td>P0162</td>
<td>S0C98</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>Description</td>
<td></td>
<td></td>
<td>S0E0F</td>
<td></td>
<td>45</td>
</tr>
<tr>
<td>Device type and model</td>
<td>DEVS/</td>
<td>P010A</td>
<td>S097A</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Entry privilege class</td>
<td>CLAE/</td>
<td>P007A</td>
<td>S0BB1</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Financial name</td>
<td>NAFI/</td>
<td>P018A</td>
<td>S0CA6</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Financial record ID</td>
<td>RNID/</td>
<td>P01E9</td>
<td>S0CCF</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Financial type</td>
<td>TYPF/</td>
<td>P04A0</td>
<td>S0C14</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Generic device</td>
<td>TYPD/</td>
<td>P02AA</td>
<td>S0CB0</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Lease type</td>
<td>TYPL/</td>
<td>P02B0</td>
<td>S0DB3</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Maintenance class</td>
<td>CODS/</td>
<td>P00C1</td>
<td>S0D95</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Manufacturer</td>
<td>GROM/</td>
<td>P014C</td>
<td>S0BA4</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>Marketing rep name</td>
<td>PERX/</td>
<td>P01AD</td>
<td>S0B5F</td>
<td></td>
<td>15</td>
</tr>
<tr>
<td>Marketing rep phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B3B</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Maximum accrual percent</td>
<td>NUMX/</td>
<td>P0194</td>
<td>S0DB0</td>
<td></td>
<td>6</td>
</tr>
<tr>
<td>Maximum accrual period</td>
<td>INTX/</td>
<td>P0162</td>
<td>S0C99</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>Maximum VPA quantity</td>
<td>VPAM/</td>
<td>P0470</td>
<td>S0D2C</td>
<td></td>
<td>8</td>
</tr>
</tbody>
</table>
Table 41. Hardware Financial Records (continued). An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum maintenance charge</td>
<td>FEEX/</td>
<td>P0204</td>
<td>S0DD8</td>
<td>YES</td>
<td>7</td>
</tr>
<tr>
<td>Minimum VPA quantity</td>
<td>VPAQ/</td>
<td>P046F</td>
<td>S0D2B</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Monthly charge</td>
<td>FEER/</td>
<td>P0141</td>
<td>S0DB2</td>
<td>YES</td>
<td>7</td>
</tr>
<tr>
<td>Owning privilege class</td>
<td>CLAO/</td>
<td>P007D</td>
<td>S0BB5</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Purchase option percent</td>
<td>NUMX/</td>
<td>P0194</td>
<td>S0DAF</td>
<td></td>
<td>6</td>
</tr>
<tr>
<td>Purchase price</td>
<td>FEEP/</td>
<td>P0140</td>
<td>S0DAE</td>
<td></td>
<td>7</td>
</tr>
<tr>
<td>Residual value</td>
<td>FEEU/</td>
<td>P0202</td>
<td>S0DCC</td>
<td>YES</td>
<td>7</td>
</tr>
<tr>
<td>Service charge out rate</td>
<td>NUMX/</td>
<td>P0194</td>
<td>S0DD7</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>System specialist name</td>
<td>PERX/</td>
<td>P01AD</td>
<td>S0B61</td>
<td></td>
<td>15</td>
</tr>
<tr>
<td>System specialist phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B3D</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Time entered</td>
<td>TIME/</td>
<td>P028A</td>
<td>S0C61</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>Time last altered</td>
<td>TIMM/</td>
<td>P028F</td>
<td>S0C62</td>
<td>YES</td>
<td>5</td>
</tr>
<tr>
<td>Transfer-to class</td>
<td>CLAT/</td>
<td>P0081</td>
<td>S0BCC</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>User last altered</td>
<td>USER/</td>
<td>P02C3</td>
<td>S0B5E</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Vendor name</td>
<td>GROV/</td>
<td>P0152</td>
<td>S0BA3</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>VPA contact name</td>
<td>PERC/</td>
<td>P01A9</td>
<td>S0B72</td>
<td></td>
<td>15</td>
</tr>
<tr>
<td>VPA contact phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B4F</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>VPA duration</td>
<td>VPAD/</td>
<td>P046E</td>
<td>S0D8D</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>VPA name</td>
<td>CODV/</td>
<td>P00A8</td>
<td>S0C08</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>VPA number</td>
<td>VPAN/</td>
<td>P046C</td>
<td>S0DF7</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>VPA start date</td>
<td>DATB/</td>
<td>P00D0</td>
<td>S0C43</td>
<td></td>
<td>10*</td>
</tr>
<tr>
<td></td>
<td>DATB/</td>
<td>P0802</td>
<td>S0C43</td>
<td></td>
<td>10</td>
</tr>
</tbody>
</table>

Software Financial Records—S0B1A

Table 42. Software Financial Records. An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Additional charge</td>
<td>FEEG/</td>
<td>P046B</td>
<td>S0DD5</td>
<td>YES</td>
<td>7</td>
</tr>
<tr>
<td>Basic license identifier</td>
<td>CODB/</td>
<td>P00A5</td>
<td>S0C19</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Basic license location</td>
<td>LOCC/</td>
<td>P016D</td>
<td>S0C18</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Date entered</td>
<td>DATE/</td>
<td>P00D8</td>
<td>S0C34</td>
<td></td>
<td>10*</td>
</tr>
<tr>
<td></td>
<td>DATE/</td>
<td>P0805</td>
<td>S0C34</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Date last altered</td>
<td>DATM/</td>
<td>P00DF</td>
<td>S0C35</td>
<td>YES</td>
<td>10*</td>
</tr>
<tr>
<td></td>
<td>DATM/</td>
<td>P080D</td>
<td>S0C35</td>
<td>YES</td>
<td>10</td>
</tr>
<tr>
<td>Description</td>
<td></td>
<td></td>
<td>S0E0F</td>
<td></td>
<td>45</td>
</tr>
<tr>
<td>Entry privilege class</td>
<td>CLAE/</td>
<td>P007A</td>
<td>S0BB1</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Financial name</td>
<td>NAFI/</td>
<td>P018A</td>
<td>S0CA6</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Financial record ID</td>
<td>RNID/</td>
<td>P01E9</td>
<td>S0CCCF</td>
<td></td>
<td>8</td>
</tr>
</tbody>
</table>
Table 42. Software Financial Records (continued). An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial license chrg (DSLO)</td>
<td>FEED/</td>
<td>P041B</td>
<td>S0DB4</td>
<td>YES</td>
<td>7</td>
</tr>
<tr>
<td>Initial license chrg (BASIC)</td>
<td>FEED/</td>
<td>P041B</td>
<td>S0DCF</td>
<td>YES</td>
<td>7</td>
</tr>
<tr>
<td>Install license applies</td>
<td>MISX/</td>
<td>P0180</td>
<td>S0EE3</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>License type</td>
<td>TYPL/</td>
<td>P02B0</td>
<td>S0C1C</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>Location license applies</td>
<td>MISL/</td>
<td>P0305</td>
<td>S0EE2</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>LPSA charge</td>
<td>FEEL/</td>
<td>P01E4</td>
<td>S0DD3</td>
<td>YES</td>
<td>5</td>
</tr>
<tr>
<td>Maintenance class</td>
<td>CODS/</td>
<td>P00C1</td>
<td>S0D95</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Marketing rep name</td>
<td>PERX/</td>
<td>P01AD</td>
<td>S0B5F</td>
<td></td>
<td>15</td>
</tr>
<tr>
<td>Marketing rep phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B3B</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Maximum VLA quantity</td>
<td>VPAM/</td>
<td>P0470</td>
<td>S0D2C</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Minimum VLA quantity</td>
<td>VPAQ/</td>
<td>P046F</td>
<td>S0D2B</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Multiple license charge</td>
<td>FEEM/</td>
<td>P013F</td>
<td>S0DAD</td>
<td>YES</td>
<td>7</td>
</tr>
<tr>
<td>Multiple license count</td>
<td>COUC/</td>
<td>P00CB</td>
<td>S0D29</td>
<td>YES</td>
<td>3</td>
</tr>
<tr>
<td>One time charge</td>
<td>FEEC/</td>
<td>P040A</td>
<td>S0DDE</td>
<td></td>
<td>7</td>
</tr>
<tr>
<td>Owning privilege class</td>
<td>CLAO/</td>
<td>P007D</td>
<td>S0B65</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Periodic charge interval</td>
<td>INTT/</td>
<td>P00AB</td>
<td>S0D8E</td>
<td>YES</td>
<td>9</td>
</tr>
<tr>
<td>Periodic license charge</td>
<td>FEER/</td>
<td>P0141</td>
<td>S0DCE</td>
<td>YES</td>
<td>7</td>
</tr>
<tr>
<td>Process charge</td>
<td>FEEX/</td>
<td>P0204</td>
<td>S0DD4</td>
<td>YES</td>
<td>7</td>
</tr>
<tr>
<td>System specialist name</td>
<td>PERX/</td>
<td>P01AD</td>
<td>S0B61</td>
<td></td>
<td>15</td>
</tr>
<tr>
<td>System specialist phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B3D</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Test period</td>
<td>INTX/</td>
<td>P0162</td>
<td>S0C9A</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>Time entered</td>
<td>TIME/</td>
<td>P028A</td>
<td>S0C61</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>Time last altered</td>
<td>TIMM/</td>
<td>P028F</td>
<td>S0C62</td>
<td>YES</td>
<td>5</td>
</tr>
<tr>
<td>Transfer-to class</td>
<td>CLAT/</td>
<td>P0081</td>
<td>S0BCC</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Upgrade license charge</td>
<td>FEEF/</td>
<td>P0241</td>
<td>S0D6D</td>
<td>YES</td>
<td>7</td>
</tr>
<tr>
<td>User last altered</td>
<td>USER/</td>
<td>P02C3</td>
<td>S0B5E</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Vendor name</td>
<td>GROV/</td>
<td>P0152</td>
<td>S0BA3</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>VLA contact name</td>
<td>PERC/</td>
<td>P01A9</td>
<td>S0B72</td>
<td></td>
<td>15</td>
</tr>
<tr>
<td>VLA contact phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B4F</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>VLA duration</td>
<td>VPAD/</td>
<td>P046E</td>
<td>S0D8D</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>VLA name</td>
<td>CODV/</td>
<td>P00A8</td>
<td>S0C08</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>VLA number</td>
<td>VPAN/</td>
<td>P046C</td>
<td>S0DF7</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>VLA start date</td>
<td>DATB/</td>
<td>P00D0</td>
<td>S0C43</td>
<td></td>
<td>10^*</td>
</tr>
<tr>
<td></td>
<td>DATB/</td>
<td>P0802</td>
<td>S0C43</td>
<td></td>
<td>10</td>
</tr>
</tbody>
</table>
### Data Center Records—S0B0C

*Table 43. Data Center Records.* An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Center name</td>
<td>NADC/</td>
<td>P0187</td>
<td>S0CA4</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Center record ID</td>
<td>RNID/</td>
<td>P01E9</td>
<td>S0CCF</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Date entered</td>
<td>DATE/</td>
<td>P00D8</td>
<td>S0C34</td>
<td></td>
<td>10*</td>
</tr>
<tr>
<td>Date last altered</td>
<td>DATM/</td>
<td>P00DF</td>
<td>S0C35</td>
<td>YES</td>
<td>10*</td>
</tr>
<tr>
<td>Description</td>
<td>S0E0F</td>
<td></td>
<td></td>
<td></td>
<td>45</td>
</tr>
<tr>
<td>Emergency phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B37</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Entry privilege class</td>
<td>CLAE/</td>
<td>P007A</td>
<td>S0BB1</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Help phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B3F</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Location code</td>
<td>LOCC/</td>
<td>P016D</td>
<td>S0C0C</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Off shift phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B36</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Operations manager name</td>
<td>PERM/</td>
<td>P01AA</td>
<td>S0B6E</td>
<td></td>
<td>15</td>
</tr>
<tr>
<td>Ops manager home phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B43</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Ops manager phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B42</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Owning privilege class</td>
<td>CLAO/</td>
<td>P007D</td>
<td>S0BB5</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Time entered</td>
<td>TIME/</td>
<td>P028A</td>
<td>S0C61</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>Time last altered</td>
<td>TIMM/</td>
<td>P028F</td>
<td>S0C62</td>
<td>YES</td>
<td>5</td>
</tr>
<tr>
<td>Transfer-to class</td>
<td>CLAT/</td>
<td>P0081</td>
<td>S0BCC</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>User last altered</td>
<td>USER/</td>
<td>P02C3</td>
<td>S0B5E</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>1st manager home phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B44</td>
<td>YES</td>
<td>13</td>
</tr>
<tr>
<td>1st shift manager name</td>
<td>PERM/</td>
<td>P01AA</td>
<td>S0B6F</td>
<td></td>
<td>15</td>
</tr>
<tr>
<td>1st shift manager phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B33</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>2nd manager home phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B45</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>2nd shift manager name</td>
<td>PERM/</td>
<td>P01AA</td>
<td>S0B70</td>
<td></td>
<td>15</td>
</tr>
<tr>
<td>2nd shift manager phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B34</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>3rd manager home phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B46</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>3rd shift manager name</td>
<td>PERM/</td>
<td>P01AA</td>
<td>S0B71</td>
<td></td>
<td>15</td>
</tr>
<tr>
<td>3rd shift manager phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B35</td>
<td></td>
<td>13</td>
</tr>
</tbody>
</table>

### System Records—S0B0E

*Table 44. System Records.* An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Center record ID</td>
<td>RNDR/</td>
<td>P01E1</td>
<td>S0CDE</td>
<td></td>
<td>8</td>
</tr>
</tbody>
</table>
### System Records—S0B0E

Table 44. System Records (continued). An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Contact department</td>
<td>GROC/</td>
<td>P014A</td>
<td>S0B9E</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>Contact name</td>
<td>PERC/</td>
<td>P01A9</td>
<td>S0B5C</td>
<td>YES</td>
<td>15</td>
</tr>
<tr>
<td>Contact phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B30</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Date entered</td>
<td>DATE/</td>
<td>P00D8</td>
<td>S0C34</td>
<td></td>
<td>10*</td>
</tr>
<tr>
<td>Date last altered</td>
<td>DATM/</td>
<td>P00DF</td>
<td>S0C35</td>
<td>YES</td>
<td>10*</td>
</tr>
<tr>
<td>Description</td>
<td>S0E0F</td>
<td></td>
<td></td>
<td></td>
<td>45</td>
</tr>
<tr>
<td>Emergency phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B37</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Entry privilege class</td>
<td>CLAE/</td>
<td>P007A</td>
<td>S0BB1</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Location code</td>
<td>LOCC/</td>
<td>P016D</td>
<td>S0CC</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Owning privilege class</td>
<td>CLAO/</td>
<td>P007D</td>
<td>S0BB5</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>System manager name</td>
<td>PERM/</td>
<td>P01AA</td>
<td>S0B6E</td>
<td></td>
<td>15</td>
</tr>
<tr>
<td>System manager phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B42</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>System name</td>
<td>NASY/</td>
<td>P018F</td>
<td>S0CA5</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>System operator phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B40</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>System record ID</td>
<td>RNID/</td>
<td>P01E9</td>
<td>S0CCF</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Time entered</td>
<td>TIME/</td>
<td>P028A</td>
<td>S0C61</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>Time last altered</td>
<td>TIMM/</td>
<td>P028F</td>
<td>S0C62</td>
<td>YES</td>
<td>5</td>
</tr>
<tr>
<td>Transfer-to class</td>
<td>CLAT/</td>
<td>P0081</td>
<td>S0BCC</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>User last altered</td>
<td>USER/</td>
<td>P02C3</td>
<td>S0B5E</td>
<td>YES</td>
<td>8</td>
</tr>
</tbody>
</table>

### Service Records—S0B19

Table 45. Service Records. An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Date entered</td>
<td>DATE/</td>
<td>P00D8</td>
<td>S0C34</td>
<td></td>
<td>10*</td>
</tr>
<tr>
<td>Date last altered</td>
<td>DATM/</td>
<td>P00DF</td>
<td>S0C35</td>
<td>YES</td>
<td>10*</td>
</tr>
<tr>
<td>Description</td>
<td>S0E0F</td>
<td></td>
<td></td>
<td></td>
<td>45</td>
</tr>
<tr>
<td>Entry privilege class</td>
<td>CLAE/</td>
<td>P007A</td>
<td>S0BB1</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Hardware rep name</td>
<td>PERX/</td>
<td>P01AD</td>
<td>S0B62</td>
<td></td>
<td>15</td>
</tr>
<tr>
<td>Hardware rep phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B38</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Off shift service phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B36</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Owning privilege class</td>
<td>CLAO/</td>
<td>P007D</td>
<td>S0BB5</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Service name</td>
<td>NASE/</td>
<td>P018E</td>
<td>S0CA7</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Service organization name</td>
<td>NAMX/</td>
<td>P018C</td>
<td>S0BA5</td>
<td></td>
<td>11</td>
</tr>
</tbody>
</table>
Table 45. Service Records (continued). An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B3E</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Service record ID</td>
<td>RNID/</td>
<td>P01E9</td>
<td>S0CCF</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Software rep name</td>
<td>PERX/</td>
<td>P01AD</td>
<td>S0B63</td>
<td></td>
<td>15</td>
</tr>
<tr>
<td>Software rep phone</td>
<td>PH/</td>
<td>P000F</td>
<td>S0B39</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Time entered</td>
<td>TIME/</td>
<td>P028A</td>
<td>S0C61</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>Time last altered</td>
<td>TIMM/</td>
<td>P028F</td>
<td>S0C62</td>
<td>YES</td>
<td>5</td>
</tr>
<tr>
<td>Transfer-to class</td>
<td>CLAT/</td>
<td>P0081</td>
<td>S0BCC</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>User last altered</td>
<td>USER/</td>
<td>P02C3</td>
<td>S0B5E</td>
<td>YES</td>
<td>8</td>
</tr>
</tbody>
</table>

Rules Records—S0120

Table 46. Rules Records. An asterisk in the length column indicates the length of the field can vary depending on the external date format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Date entered</td>
<td>DATE/</td>
<td>P00D8</td>
<td>S0C34</td>
<td></td>
<td>10*</td>
</tr>
<tr>
<td>Date last altered</td>
<td>DATM/</td>
<td>P00DF</td>
<td>S0C35</td>
<td>YES</td>
<td>10*</td>
</tr>
<tr>
<td>Entry privilege class</td>
<td>CLAE/</td>
<td>P007A</td>
<td>S0BB1</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Inc/dec priority</td>
<td>RPID/</td>
<td>P044E</td>
<td>S0123</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Initial priority</td>
<td>PRII/</td>
<td>P0463</td>
<td>S0BE6</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>Key item affected</td>
<td>COMK/</td>
<td>P00C8</td>
<td>S0CBF</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Level 1 duration</td>
<td>INTX/</td>
<td>P0162</td>
<td>S0124</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Level 1 ID to notify</td>
<td>USRN/</td>
<td>P0468</td>
<td>S0125</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Level 1 node</td>
<td>COMX/</td>
<td>P00CA</td>
<td>S0126</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Level 2 duration</td>
<td>INTX/</td>
<td>P0162</td>
<td>S0127</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Level 2 ID to notify</td>
<td>USRN/</td>
<td>P0468</td>
<td>S0128</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Level 2 node</td>
<td>COMX/</td>
<td>P00CA</td>
<td>S0129</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Level 3 duration</td>
<td>INTX/</td>
<td>P0162</td>
<td>S012A</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Level 3 ID to notify</td>
<td>USRN/</td>
<td>P0468</td>
<td>S012B</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Level 3 node</td>
<td>COMX/</td>
<td>P00CA</td>
<td>S012C</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Location code</td>
<td>LOCC/</td>
<td>P016D</td>
<td>S0CC0</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Priority adjustment</td>
<td>RADJ/</td>
<td>P044D</td>
<td>S0122</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>Problem type</td>
<td>TYPE/</td>
<td>P02AF</td>
<td>S0C09</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Time entered</td>
<td>TIME/</td>
<td>P028A</td>
<td>S0C61</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>Time last altered</td>
<td>TIMM/</td>
<td>P028F</td>
<td>S0C62</td>
<td>YES</td>
<td>5</td>
</tr>
<tr>
<td>User last altered</td>
<td>USER/</td>
<td>P02C3</td>
<td>S0B5E</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Field Name</td>
<td>Prefix</td>
<td>Prefix Index</td>
<td>S-Word Index</td>
<td>Journal Field</td>
<td>Length</td>
</tr>
<tr>
<td>----------------------------</td>
<td>---------</td>
<td>--------------</td>
<td>--------------</td>
<td>---------------</td>
<td>--------</td>
</tr>
<tr>
<td>Acct version</td>
<td>NWAC/</td>
<td>P0688</td>
<td>S156D</td>
<td>16</td>
<td></td>
</tr>
<tr>
<td>(BIOS) date</td>
<td>BSDT/</td>
<td>P063D</td>
<td>S15E1</td>
<td>16</td>
<td></td>
</tr>
<tr>
<td>Bios ID</td>
<td></td>
<td></td>
<td>S15DE</td>
<td>128</td>
<td></td>
</tr>
<tr>
<td>Bios ID bytes</td>
<td>BSBY/</td>
<td>P063C</td>
<td>S15DF</td>
<td>16</td>
<td></td>
</tr>
<tr>
<td>Bios manufacturer ID</td>
<td>BSMF/</td>
<td>P063E</td>
<td>S15E5</td>
<td>64</td>
<td></td>
</tr>
<tr>
<td>Bios string</td>
<td></td>
<td></td>
<td>S15E0</td>
<td>128</td>
<td></td>
</tr>
<tr>
<td>Booted OS name</td>
<td>OSNM/</td>
<td>P0629</td>
<td>S15C0</td>
<td>16</td>
<td></td>
</tr>
<tr>
<td>Booted OS version</td>
<td>OSVR/</td>
<td>P062A</td>
<td>S15C1</td>
<td>16</td>
<td></td>
</tr>
<tr>
<td>Class created</td>
<td>CLAE/</td>
<td>P007A</td>
<td>S0BB1</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>CLIB major ver</td>
<td>NWMV/</td>
<td>P068D</td>
<td>S156E</td>
<td>16</td>
<td></td>
</tr>
<tr>
<td>CLIB minor ver</td>
<td>NWNV/</td>
<td>P068E</td>
<td>S156F</td>
<td>16</td>
<td></td>
</tr>
<tr>
<td>CLIB revision</td>
<td>NWRV/</td>
<td>P0692</td>
<td>S1570</td>
<td>16</td>
<td></td>
</tr>
<tr>
<td>Computer architecture</td>
<td></td>
<td></td>
<td>S1531</td>
<td>32</td>
<td></td>
</tr>
<tr>
<td>Computer model</td>
<td>MCPU/</td>
<td>P061F</td>
<td>S1532</td>
<td>32</td>
<td></td>
</tr>
<tr>
<td>(Config file) date</td>
<td>CDAT/</td>
<td>P06BC</td>
<td>S15B2</td>
<td>32</td>
<td></td>
</tr>
<tr>
<td>(Config file) name</td>
<td>FNAM/</td>
<td>P0648</td>
<td>S15CE</td>
<td>24</td>
<td></td>
</tr>
<tr>
<td>(Config file) path</td>
<td>FPTH/</td>
<td>P064B</td>
<td>S15CF</td>
<td>64</td>
<td></td>
</tr>
<tr>
<td>(Config file) size</td>
<td>FSIZ/</td>
<td>P064D</td>
<td>S15D0</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>(Config file) time</td>
<td>CTIM/</td>
<td>P06BE</td>
<td>S15B3</td>
<td>32</td>
<td></td>
</tr>
<tr>
<td>Conns used</td>
<td>NWMU/</td>
<td>P068C</td>
<td>S1574</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>Conventional total KB</td>
<td>TTKB/</td>
<td>P06B3</td>
<td>S1541</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>Co-processor model</td>
<td>MCOP/</td>
<td>P0671</td>
<td>S1542</td>
<td>24</td>
<td></td>
</tr>
<tr>
<td>Co-processor type</td>
<td>TCOP/</td>
<td>P066AF</td>
<td>S1543</td>
<td>24</td>
<td></td>
</tr>
<tr>
<td>Current build</td>
<td>NTBD/</td>
<td>P0681</td>
<td>S15D7</td>
<td>32</td>
<td></td>
</tr>
<tr>
<td>Current type</td>
<td>NTCT/</td>
<td>P0682</td>
<td>S15D8</td>
<td>32</td>
<td></td>
</tr>
<tr>
<td>Current version</td>
<td>NTCV/</td>
<td>P0683</td>
<td>S15D9</td>
<td>32</td>
<td></td>
</tr>
<tr>
<td>Cylinders</td>
<td>HDCY/</td>
<td>P064E</td>
<td>S154A</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>Date created</td>
<td>DATE/</td>
<td>P00D8</td>
<td>S0C34</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>Date installed</td>
<td>—</td>
<td>—</td>
<td>S15B4</td>
<td>32</td>
<td></td>
</tr>
<tr>
<td>Date modified</td>
<td>DATM/</td>
<td>P00DF</td>
<td>S0C35</td>
<td>YES</td>
<td></td>
</tr>
<tr>
<td>Device name</td>
<td>NWDV/</td>
<td>P068A</td>
<td>S1571</td>
<td>32</td>
<td></td>
</tr>
<tr>
<td>Expanded (total) KB</td>
<td>EPKB/</td>
<td>P0646</td>
<td>S1545</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>Extended (total) KB</td>
<td>ETKB/</td>
<td>P0647</td>
<td>S1546</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>Free KB</td>
<td>LDFK/</td>
<td>P0666</td>
<td>S155E</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>(Hard disk) access speed</td>
<td>HDSP/</td>
<td>P0653</td>
<td>S1549</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>(Hard disk) serial number</td>
<td>HDSR/</td>
<td>P0654</td>
<td>S1550</td>
<td>32</td>
<td></td>
</tr>
<tr>
<td>Hardware system ID</td>
<td></td>
<td></td>
<td>S1512</td>
<td>32</td>
<td></td>
</tr>
<tr>
<td>Field Name</td>
<td>Prefix</td>
<td>Prefix Index</td>
<td>S-Word Index</td>
<td>Journal Field</td>
<td>Length</td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>---------</td>
<td>--------------</td>
<td>--------------</td>
<td>---------------</td>
<td>--------</td>
</tr>
<tr>
<td>Hardware system ID desc</td>
<td>HDESC/</td>
<td>P06C1</td>
<td>S151C</td>
<td></td>
<td>32</td>
</tr>
<tr>
<td>Heads</td>
<td>HDHD/</td>
<td>P064F</td>
<td>S154B</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Installed HD ID</td>
<td>INHD/</td>
<td>P0659</td>
<td>S1554</td>
<td></td>
<td>32</td>
</tr>
<tr>
<td>Internet (bridge) supp</td>
<td>NWIN/</td>
<td>P068B</td>
<td>S1572</td>
<td></td>
<td>16</td>
</tr>
<tr>
<td>(IPX) address</td>
<td>IPXA/</td>
<td>P065F</td>
<td>S1559</td>
<td></td>
<td>32</td>
</tr>
<tr>
<td>(IPX) full name</td>
<td>IPXN/</td>
<td>P0661</td>
<td>S155A</td>
<td></td>
<td>32</td>
</tr>
<tr>
<td>(IPX) login name</td>
<td>IPXL/</td>
<td>P0660</td>
<td>S155B</td>
<td></td>
<td>32</td>
</tr>
<tr>
<td>(IPX LAN connection) name</td>
<td>IPNM/</td>
<td>P065D</td>
<td>S1589</td>
<td></td>
<td>32</td>
</tr>
<tr>
<td>(IPX LAN connection) number</td>
<td>IPNU/</td>
<td>P065E</td>
<td>S158A</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>(Logical drive) file system</td>
<td>LDFS/</td>
<td>P0667</td>
<td>S155D</td>
<td></td>
<td>16</td>
</tr>
<tr>
<td>(Logical drive) name</td>
<td>LDNM/</td>
<td>P0668</td>
<td>S1560</td>
<td></td>
<td>32</td>
</tr>
<tr>
<td>(Logical drive) serial number</td>
<td>LDSR/</td>
<td>P0669</td>
<td>S1561</td>
<td></td>
<td>32</td>
</tr>
<tr>
<td>(Logical drive) size KB</td>
<td>LDSZ/</td>
<td>P066A</td>
<td>S1562</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Max conns</td>
<td>NWMC/</td>
<td>P066E</td>
<td>S1573</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Max volumes</td>
<td>NWVL/</td>
<td>P0697</td>
<td>S1575</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Model</td>
<td>MHDR/</td>
<td>P0674</td>
<td>S154D</td>
<td></td>
<td>32</td>
</tr>
<tr>
<td>Mount dir</td>
<td>LDDR/</td>
<td>P0665</td>
<td>S155F</td>
<td></td>
<td>16</td>
</tr>
<tr>
<td>(Network node) address</td>
<td>NNAD/</td>
<td>P0625</td>
<td>S159C</td>
<td></td>
<td>32</td>
</tr>
<tr>
<td>(Network node) name</td>
<td>NNNM/</td>
<td>P0626</td>
<td>S159B</td>
<td></td>
<td>32</td>
</tr>
<tr>
<td>NW serial number</td>
<td>NWSR/</td>
<td>P06BB</td>
<td>S159E</td>
<td></td>
<td>12</td>
</tr>
<tr>
<td>Paging space (KB)</td>
<td>PGSP/</td>
<td>P062E</td>
<td>S153C</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>(PC port) base address</td>
<td>PADR/</td>
<td>P069A</td>
<td>S158B</td>
<td></td>
<td>16</td>
</tr>
<tr>
<td>(PC port) number</td>
<td>PNUM/</td>
<td>P069D</td>
<td>S158C</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>(PC port) type</td>
<td>TPOR/</td>
<td>P06B2</td>
<td>S158D</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Physical memory (KB)</td>
<td>PMEM/</td>
<td>P0630</td>
<td>S1536</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Printserver ver</td>
<td>NWPV/</td>
<td>P068F</td>
<td>S1576</td>
<td></td>
<td>16</td>
</tr>
<tr>
<td>Processor model</td>
<td>MPRC/</td>
<td>P0621</td>
<td>S1537</td>
<td></td>
<td>32</td>
</tr>
<tr>
<td>Processor speed</td>
<td>PSPD/</td>
<td>P0632</td>
<td>S1538</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Protocol</td>
<td>NPTC/</td>
<td>P0680</td>
<td>S156C</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Queuing ver</td>
<td>NWQV/</td>
<td>P0690</td>
<td>S1577</td>
<td></td>
<td>16</td>
</tr>
<tr>
<td>Reg org</td>
<td>NTOR/</td>
<td>P0684</td>
<td>S15DA</td>
<td></td>
<td>32</td>
</tr>
<tr>
<td>Reg user</td>
<td>NTUS/</td>
<td>P0687</td>
<td>S15DB</td>
<td></td>
<td>32</td>
</tr>
<tr>
<td>Revision level</td>
<td>NWRL/</td>
<td>P0691</td>
<td>S1578</td>
<td></td>
<td>16</td>
</tr>
<tr>
<td>Scan time (date)</td>
<td>DATZ/</td>
<td>P0614</td>
<td>S1518</td>
<td>YES</td>
<td>10</td>
</tr>
<tr>
<td>Scan time (time)</td>
<td>TIMZ/</td>
<td>P0615</td>
<td>S1519</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Sectors</td>
<td>HDSC/</td>
<td>P0651</td>
<td>S154E</td>
<td></td>
<td>10</td>
</tr>
</tbody>
</table>
### Table 47. Inventory Records (continued)

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Security (restriction) level</td>
<td>NWSL/</td>
<td>P0694</td>
<td>S1579</td>
<td></td>
<td>16</td>
</tr>
<tr>
<td>Seek time</td>
<td>HDSK/</td>
<td>P0652</td>
<td>S154F</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Service pack</td>
<td>NTSP/</td>
<td>P0685</td>
<td>S15DC</td>
<td></td>
<td>32</td>
</tr>
<tr>
<td>Signature file name</td>
<td>SFNM/</td>
<td>P062C</td>
<td>S15E3</td>
<td></td>
<td>32</td>
</tr>
<tr>
<td>Signature file size</td>
<td>SFSZ/</td>
<td>P062D</td>
<td>S15E4</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Size MB</td>
<td>HDSZ/</td>
<td>P0618</td>
<td>S1534</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>(Software) identifier</td>
<td>SW/</td>
<td>P0635</td>
<td>S1513</td>
<td></td>
<td>64</td>
</tr>
<tr>
<td>Software level</td>
<td>NWSF/</td>
<td>P0693</td>
<td>S157A</td>
<td></td>
<td>16</td>
</tr>
<tr>
<td>System root</td>
<td>NTSR/</td>
<td>P0686</td>
<td>S15DD</td>
<td></td>
<td>32</td>
</tr>
<tr>
<td>Time created</td>
<td>TIME/</td>
<td>P028B</td>
<td>S0C61</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Time modified</td>
<td>TIMM/</td>
<td>P0291</td>
<td>S0C62</td>
<td>YES</td>
<td>8</td>
</tr>
<tr>
<td>Tivoli object ID</td>
<td>TOBJ/</td>
<td>P06B0</td>
<td>S151A</td>
<td></td>
<td>32</td>
</tr>
<tr>
<td>Tivoli object ID desc</td>
<td>—</td>
<td>—</td>
<td>S151D</td>
<td></td>
<td>96</td>
</tr>
<tr>
<td>Tivoli object label</td>
<td>OBJ/</td>
<td>P0628</td>
<td>S1511</td>
<td></td>
<td>32</td>
</tr>
<tr>
<td>TMR Label</td>
<td>TMR/</td>
<td>P0638</td>
<td>S1510</td>
<td></td>
<td>32</td>
</tr>
<tr>
<td>TTS level</td>
<td>NWTS/</td>
<td>P0696</td>
<td>S157C</td>
<td></td>
<td>16</td>
</tr>
<tr>
<td>User modified</td>
<td>USER/</td>
<td>P02C4</td>
<td>S0B5E</td>
<td>YES</td>
<td>7</td>
</tr>
<tr>
<td>VAP version</td>
<td>NWVP/</td>
<td>P0698</td>
<td>S157D</td>
<td></td>
<td>16</td>
</tr>
<tr>
<td>Version</td>
<td>NWVR/</td>
<td>P0699</td>
<td>S157E</td>
<td></td>
<td>16</td>
</tr>
<tr>
<td>Version ID</td>
<td>SWVR/</td>
<td>P0637</td>
<td>S15E2</td>
<td></td>
<td>16</td>
</tr>
<tr>
<td>Virtual console ver</td>
<td>NWCV/</td>
<td>P0689</td>
<td>S157F</td>
<td></td>
<td>16</td>
</tr>
<tr>
<td>Volume label</td>
<td>LDVL/</td>
<td>P066B</td>
<td>S1563</td>
<td></td>
<td>64</td>
</tr>
</tbody>
</table>

### Call Records—S0037

### Table 48. Call Records

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Address 1</td>
<td>ADDR/</td>
<td>P06EA</td>
<td>S152A</td>
<td></td>
<td>45</td>
</tr>
<tr>
<td>Address 2</td>
<td>ADDR/</td>
<td>P06EA</td>
<td>S152B</td>
<td></td>
<td>45</td>
</tr>
<tr>
<td>Associated problem ID</td>
<td>RNAP/</td>
<td>P0341</td>
<td>S0CE1</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Brief description</td>
<td></td>
<td></td>
<td>S0E0F</td>
<td></td>
<td>45</td>
</tr>
<tr>
<td>Call taker department</td>
<td>GROA/</td>
<td>P0148 P0149</td>
<td>S0B9C</td>
<td>YES</td>
<td>11</td>
</tr>
<tr>
<td>Call taker ID</td>
<td>ASID/</td>
<td>P0610</td>
<td>S01D3</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Call taker name</td>
<td>PERA/</td>
<td>P0343 P01A8 P060D</td>
<td>S0B5A</td>
<td>YES</td>
<td>40</td>
</tr>
<tr>
<td>Call taker phone</td>
<td>PH/</td>
<td>P0010 P0013</td>
<td>S0B2E</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Call ID</td>
<td>RNID/</td>
<td>P01EC</td>
<td>S0CCF</td>
<td></td>
<td>8</td>
</tr>
</tbody>
</table>
Table 48. Call Records (continued)

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Call fax number</td>
<td>FAX/</td>
<td>P0717</td>
<td>S01D5</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Call type</td>
<td>TYPJ/</td>
<td>P0710</td>
<td>S01D0</td>
<td>YES</td>
<td>6</td>
</tr>
<tr>
<td>Caller’s identifier</td>
<td>CAID/</td>
<td>P0611</td>
<td>S14FD</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>City/State/Province</td>
<td>CITY/</td>
<td>P06EB</td>
<td>S152C</td>
<td></td>
<td>45</td>
</tr>
<tr>
<td>Company name</td>
<td>CO/</td>
<td>P06E9</td>
<td>S15F7</td>
<td></td>
<td>45</td>
</tr>
<tr>
<td>Date entered</td>
<td>DATE/</td>
<td>P0805</td>
<td>S0C34</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Date last altered</td>
<td>DATM/</td>
<td>P00DF</td>
<td>S0C35</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>Department</td>
<td>GROS/</td>
<td>P0150</td>
<td>S0B9B</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>End date</td>
<td>DATF/</td>
<td>P0806</td>
<td>S0C40</td>
<td>YES</td>
<td>10</td>
</tr>
<tr>
<td>End time</td>
<td>TIMF/</td>
<td>P028C</td>
<td>S0C71</td>
<td>YES</td>
<td>5</td>
</tr>
<tr>
<td>Management application entry sword</td>
<td></td>
<td></td>
<td>S002C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>PERS/</td>
<td>P01AC</td>
<td>S0B59</td>
<td>YES</td>
<td>40</td>
</tr>
<tr>
<td>Phone</td>
<td>PH/</td>
<td>P0010</td>
<td>S0B2D</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Postal code/ZIP</td>
<td>PCOD/</td>
<td>P06ED</td>
<td>S152E</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Problem description</td>
<td></td>
<td></td>
<td>S0E01</td>
<td>freeform text</td>
<td></td>
</tr>
<tr>
<td>Start date</td>
<td>DATB/</td>
<td>P00D0</td>
<td>S0C43</td>
<td>YES</td>
<td>10</td>
</tr>
<tr>
<td>Start time</td>
<td>TIMB/</td>
<td>P0283</td>
<td>S0C6C</td>
<td>YES</td>
<td>5</td>
</tr>
<tr>
<td>Status</td>
<td>STAC/</td>
<td>P0272</td>
<td>S0BEE</td>
<td>YES</td>
<td>7</td>
</tr>
<tr>
<td>Time entered</td>
<td>TIME/</td>
<td>P028B</td>
<td>S0C61</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>Time last altered</td>
<td>TIMM/</td>
<td>P0290</td>
<td>S0C62</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>User last altered</td>
<td>USER/</td>
<td>P02C3</td>
<td>S0B5E</td>
<td></td>
<td>8</td>
</tr>
</tbody>
</table>

People Records—S0031

Table 49. People Records

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Address 1</td>
<td>ADDR/</td>
<td>P06EA</td>
<td>S152A</td>
<td></td>
<td>45</td>
</tr>
<tr>
<td>Address 2</td>
<td>ADDR/</td>
<td>P06EA</td>
<td>S152B</td>
<td></td>
<td>45</td>
</tr>
</tbody>
</table>
### Data Model Records — Tivoli Decision Support

The following information pertains to the data model records (specifically, the data attribute records) that are provided with Tivoli Information Management for z/OS to support usage of the Information Management Guide with the Tivoli Decision Support product. For an overview of how Tivoli Information Management for z/OS data is used with Tivoli Decision Support, see "Tivoli Decision Support" on page 3. Data attribute records are listed in order by prefix.

**Note:** In TME 10 Information/Management Version 1.1, the names of these data attribute records contained a pound sign character as the trigger character (for example, BLG#CLAE). In this release of Tivoli Information Management for z/OS, the ampersand character is used instead. The data attribute records are shipped with Tivoli Information Management for z/OS in the SBLMRCDS data set.

The following data view records (available in the BLGLRTDS list in the SBLMRCDS data set) support use of Tivoli Information Management for z/OS with Tivoli Decision Support: BLGPROBS, BLGCLSPR, BLGOPNPR, BLGALLPR, BLGCHANG, BLGACTVY.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Prefix</th>
<th>Prefix Index</th>
<th>S-Word Index</th>
<th>Journal Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>City/State/Province</td>
<td>CITY/</td>
<td>P06EB</td>
<td>S152C</td>
<td></td>
<td>45</td>
</tr>
<tr>
<td>Company name</td>
<td>CO/</td>
<td>P06E9</td>
<td>S15F7</td>
<td></td>
<td>45</td>
</tr>
<tr>
<td>Country</td>
<td>CTRY/</td>
<td>P06EE</td>
<td>S152F</td>
<td></td>
<td>20</td>
</tr>
<tr>
<td>Date entered</td>
<td>DATE/</td>
<td>P0805</td>
<td>S0C34</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Date last altered</td>
<td>DATM/</td>
<td>P080D</td>
<td>S0C35</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>E-mail address</td>
<td>EADD/</td>
<td>P0713</td>
<td>S01D1</td>
<td></td>
<td>45</td>
</tr>
<tr>
<td>Fax number</td>
<td>PHFX/</td>
<td>P06E4</td>
<td>S15F1</td>
<td></td>
<td>21</td>
</tr>
<tr>
<td>Mobile phone number</td>
<td>PHMO/</td>
<td>P06E6</td>
<td>S15F3</td>
<td></td>
<td>21</td>
</tr>
<tr>
<td>Pager number</td>
<td>PHPG/</td>
<td>P06E5</td>
<td>S15F2</td>
<td></td>
<td>21</td>
</tr>
<tr>
<td>People identifier</td>
<td>RNID/</td>
<td>P01EC</td>
<td>S0CCF</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Person department</td>
<td>DEPT/</td>
<td>P06DC</td>
<td>S151E</td>
<td></td>
<td>16</td>
</tr>
<tr>
<td>Person description</td>
<td></td>
<td></td>
<td>S0E01</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Person name</td>
<td>PERNM/</td>
<td>P06DD</td>
<td>S151F</td>
<td>YES</td>
<td>40</td>
</tr>
<tr>
<td>Person role</td>
<td>ROLE/</td>
<td>P06DE</td>
<td>S15F8</td>
<td>YES</td>
<td>10</td>
</tr>
<tr>
<td>Phone number</td>
<td>PHNM/</td>
<td>P06E3</td>
<td>S15F0</td>
<td></td>
<td>21</td>
</tr>
<tr>
<td>Postal code/ZIP</td>
<td>PCOD/</td>
<td>P06ED</td>
<td>S152E</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Preferred contact</td>
<td>PNOT/</td>
<td>P06EF</td>
<td>S01D2</td>
<td>YES</td>
<td>6</td>
</tr>
<tr>
<td>Time entered</td>
<td>TIME/</td>
<td>P028A</td>
<td>S0C61</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>Time last altered</td>
<td>TIMM/</td>
<td>P028F</td>
<td>S0C62</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>TSD site ID</td>
<td>OSITE/</td>
<td>P0607</td>
<td>S14F7</td>
<td></td>
<td>6</td>
</tr>
<tr>
<td>TSD user ID</td>
<td>TUSER/</td>
<td>P06DB</td>
<td>S15F9</td>
<td></td>
<td>15</td>
</tr>
<tr>
<td>User last altered</td>
<td>USER/</td>
<td>P02C3</td>
<td>S0B5E</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Field Name</td>
<td>Data Attribute Record</td>
<td>Prefix</td>
<td>ODBC Column Name</td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------------------------------</td>
<td>-----------------------</td>
<td>---------</td>
<td>---------------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Entry privilege class</td>
<td>BLG&amp;CLAE</td>
<td>CLAE/</td>
<td>ENTRY_PRIVILEGE_CLASS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cause/resolution code</td>
<td>BLG&amp;CODC</td>
<td>CODC/</td>
<td>CAUSE_COMPLETION_CODE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Change reason</td>
<td>BLG&amp;CODR</td>
<td>CODR/</td>
<td>CHANGE_REASON</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Device</td>
<td>BLG&amp;COMD</td>
<td>COMD/</td>
<td>DEVICE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Key item</td>
<td>BLG&amp;COMK</td>
<td>COMK/</td>
<td>KEY_ITEM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Component</td>
<td>BLG&amp;COMX</td>
<td>COMX/</td>
<td>COMPONENT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Duplicate problem count</td>
<td>BLG&amp;COUD</td>
<td>COUD/</td>
<td>DUPLICATE_PROBLEM_COUNT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Actual start date</td>
<td>BLG&amp;DATB</td>
<td>DATB/</td>
<td>DATE_STARTED</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Date requested</td>
<td>BLG&amp;DATD</td>
<td>DATD/</td>
<td>DATE_REQUIRED</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Date entered</td>
<td>BLG&amp;DATE</td>
<td>DATE/</td>
<td>DATE_ENTERED</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Completion date</td>
<td>BLG&amp;DATF</td>
<td>DATF/</td>
<td>DATE_FINISHED</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Date last altered</td>
<td>BLG&amp;DATM</td>
<td>DATM/</td>
<td>DATE_LAST_ALTERED</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Date problem occurred</td>
<td>BLG&amp;DATO</td>
<td>DATO/</td>
<td>DATE_OCCURRED</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Planned start date</td>
<td>BLG&amp;DATP</td>
<td>DATP/</td>
<td>PLANNED_START_DATE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Date closed</td>
<td>BLG&amp;DATR</td>
<td>DATR/</td>
<td>DATE_CLOSED</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Planned end date</td>
<td>BLG&amp;DATT</td>
<td>DATT/</td>
<td>PLANNED_END_DATE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Date opened</td>
<td>BLG&amp;DATX</td>
<td>DATX/</td>
<td>DATE_OPENED</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>BLG&amp;DSAB</td>
<td></td>
<td>DESCRIPTION</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Actual effort</td>
<td>BLG&amp;CEFA</td>
<td>EFA/</td>
<td>ACTUAL_EFFORT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Estimated effort</td>
<td>BLG&amp;CEFE</td>
<td>EFE/</td>
<td>ESTIMATED_EFFORT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Escalation level</td>
<td>BLG&amp;ESCL</td>
<td>ESCL/</td>
<td>ESCALATION_LEVEL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Assignee department</td>
<td>BLG&amp;GROA</td>
<td>GROA/</td>
<td>ASSIGNEE_DEPARTMENT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reporter or requester department</td>
<td>BLG&amp;GROS</td>
<td>GROS/</td>
<td>REQUESTER_DEPARTMENT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Actual impact</td>
<td>BLG&amp;IMPA</td>
<td>IMPA/</td>
<td>ACTUAL_IMPACT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Risk assessment</td>
<td>BLG&amp;IMPR</td>
<td>IMPR/</td>
<td>RISK_ASSESSMENT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Estimated duration</td>
<td>BLG&amp;INTE</td>
<td>INTE/</td>
<td>ESTIMATED_DURATION</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Actual duration</td>
<td>BLG&amp;ADUR</td>
<td>INTO/</td>
<td>ACTUAL_DURATION</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Outage</td>
<td>BLG&amp;INTO</td>
<td>INTO/</td>
<td>OUTAGE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Vendor response/travel time</td>
<td>BLG&amp;INTX</td>
<td>INTX/</td>
<td>VENDOR_RESPONSE_TRAVEL_TIME</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Location of problem</td>
<td>BLG&amp;LOCC</td>
<td>LOCC/</td>
<td>LOCATION</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Activity name</td>
<td>BLG&amp;NAMA</td>
<td>NAMA/</td>
<td>ACTIVITY_NAME</td>
<td></td>
<td></td>
</tr>
<tr>
<td>System</td>
<td>BLG&amp;NASY</td>
<td>NASY/</td>
<td>SYSTEM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Vendor</td>
<td>BLG&amp;NUMV</td>
<td>NUMV/</td>
<td>VENDOR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Assignee name</td>
<td>BLG&amp;PERA</td>
<td>PERA/</td>
<td>ASSIGNEE_NAME</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reported by</td>
<td>BLG&amp;PERS</td>
<td>PERS/</td>
<td>REQUESTED_BY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Phone number</td>
<td>BLG&amp;PHON</td>
<td>PH/</td>
<td>PHONE_NUMBER</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Current priority/severity</td>
<td>BLG&amp;PRIO</td>
<td>PRIO/</td>
<td>CURRENT_PRIORITY_SEVERITY</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 50. Data Attribute Records used with Tivoli Decision Support (continued)

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Attribute Record</th>
<th>Prefix</th>
<th>ODBC Column Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cause change number</td>
<td>BLG&amp;RNCRX</td>
<td>RNCX/</td>
<td>CAUSED_BY_CHANGE_NUMBER</td>
</tr>
<tr>
<td>Record identifier (problem,</td>
<td>BLG&amp;RNID</td>
<td>RNID/</td>
<td>RECORD_ID</td>
</tr>
<tr>
<td>change, or activity)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Parent change record</td>
<td>BLG&amp;RNOK</td>
<td>RNOR/</td>
<td>PARENT_CHANGE_RECORD</td>
</tr>
<tr>
<td>Problem fixed</td>
<td>BLG&amp;RNPR</td>
<td>RNPR/</td>
<td>PROBLEM_FIXED</td>
</tr>
<tr>
<td>Status</td>
<td>BLG&amp;STAC</td>
<td>STAC/</td>
<td>STATUS</td>
</tr>
<tr>
<td>Approval status</td>
<td>BLG&amp;STAP</td>
<td>STAP/</td>
<td>APPROVAL_STATUS</td>
</tr>
<tr>
<td>Time started</td>
<td>BLG&amp;TIMB</td>
<td>TIMB/</td>
<td>TIME_STARTED</td>
</tr>
<tr>
<td>Time entered</td>
<td>BLG&amp;TIME</td>
<td>TIME/</td>
<td>TIME_ENTERED</td>
</tr>
<tr>
<td>Time finished</td>
<td>BLG&amp;TIMF</td>
<td>TIMF/</td>
<td>TIME_FINISHED</td>
</tr>
<tr>
<td>Time last altered</td>
<td>BLG&amp;TIMM</td>
<td>TIMM/</td>
<td>TIME_LAST_ALTERED</td>
</tr>
<tr>
<td>Time occurred</td>
<td>BLG&amp;TIMO</td>
<td>TIMO/</td>
<td>TIME_OCCURRED</td>
</tr>
<tr>
<td>Time closed</td>
<td>BLG&amp;TIMR</td>
<td>TIMR/</td>
<td>TIME_CLOSED</td>
</tr>
<tr>
<td>Time opened</td>
<td>BLG&amp;TIMX</td>
<td>TIMX/</td>
<td>TIME_OPENED</td>
</tr>
<tr>
<td>Type (problem, change, or</td>
<td>BLG&amp;TYPE</td>
<td>TYPE/</td>
<td>TYPE</td>
</tr>
<tr>
<td>activity)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>User last altered</td>
<td>BLG&amp;USER</td>
<td>USER/</td>
<td>USER_LAST_ALTERED</td>
</tr>
</tbody>
</table>

Text Search Records

Data attribute records are provided to support use of OS/390 Text Search.

Index Records

In addition to the fields listed, index records use the following fields: Record ID (RNID/), Date entered (DATE/), Time entered (TIME/), Date last altered (DATM/), Time last altered (TIMM/), Entry privilege class (CLAE/), User last altered (USER/).

Table 51. Data Attribute Records used with OS/390 Text Search Indexes

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Attribute Record</th>
<th>Prefix</th>
<th>S-word Index</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>BLH&amp;DESC</td>
<td>S0E0F</td>
<td></td>
</tr>
<tr>
<td>Indexed record data</td>
<td>BLH&amp;DVRC</td>
<td>DVRC/</td>
<td>S12E2</td>
</tr>
<tr>
<td>RECS=INDEX</td>
<td></td>
<td></td>
<td>S12E2</td>
</tr>
</tbody>
</table>

Solution Records

In addition to the fields listed, solution records use the following fields: Record ID (RNID/), Date entered (DATE/), Time entered (TIME/), Date last altered (DATM/), Time last altered (TIMM/), Entry privilege class (CLAE/), User last altered (USER/).

Table 52. Data Attribute Records used with OS/390 Text Search Indexes

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Attribute Record</th>
<th>Prefix</th>
<th>S-word Index</th>
</tr>
</thead>
<tbody>
<tr>
<td>Problem number</td>
<td>BLH&amp;RNPD</td>
<td>RNPD/</td>
<td>S0CD0</td>
</tr>
<tr>
<td>Description</td>
<td>BLH&amp;DESC</td>
<td></td>
<td>S0E0F</td>
</tr>
</tbody>
</table>
### Table 52. Data Attribute Records used with OS/390 Text Search Indexes (continued)

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Attribute Record</th>
<th>Prefix</th>
<th>S-word Index</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description text</td>
<td>BLM&amp;DTXT</td>
<td></td>
<td>S0E01</td>
</tr>
<tr>
<td>System</td>
<td>BLH&amp;SYST</td>
<td>SYST/</td>
<td>S0CAA</td>
</tr>
<tr>
<td>Component</td>
<td>BLH&amp;COMP</td>
<td>COMP/</td>
<td>S0CA8</td>
</tr>
<tr>
<td>Item</td>
<td>BLH&amp;ITEM</td>
<td>ITEM/</td>
<td>S0CAC</td>
</tr>
<tr>
<td>Module</td>
<td>BLH&amp;MODL</td>
<td>MODL/</td>
<td>S0CAD</td>
</tr>
<tr>
<td>Resolution text</td>
<td>BLM&amp;RTXT</td>
<td></td>
<td>S0E03</td>
</tr>
<tr>
<td>Usage count</td>
<td>BLH&amp;COUX</td>
<td>COUX/</td>
<td>S0D26</td>
</tr>
<tr>
<td>RECS=SOLUTION</td>
<td></td>
<td></td>
<td>S0B05</td>
</tr>
</tbody>
</table>
To help you use the Report Format Facility, Tivoli Information Management for z/OS includes two types of predefined variables: read-only and read/write. All predefined variables begin with the character Z.

**Note:** You cannot increase the number of predefined variables by using the same naming convention. For example, although you can define additional ZSORTn or ZSORTAPn variables, the Report Format Facility cannot initialize them to the data associated with the nth sort field.

### Read-Only Variables

You use read-only variables on RFT statements to control report processing or to provide information for the report. Read-only variables are predefined and initialized by the Report Format Facility and cannot be modified. Therefore, you cannot specify read-only variables with the NAME keyword on a SET command. The thirteen categories of read-only variables are:

1. System date and time
2. Day-of-the-week
3. Weekday date range
4. Weekly date range
5. Monthly date range
6. Name-of-the-month
7. Yearly date range
8. User-specified date range
9. Search count
10. Sort
11. IF command
12. Map processing
13. Miscellaneous

Many of the read-only variables are used in the standard RFTs. However, you can also use these variables in your own RFTs.

### System Date and Time Variables

The Report Format Facility sets the date and time variables using the current system date and time when report processing begins. Yesterday’s and tomorrow’s dates are also calculated and set using the current date.

The three date formats for each date are:

1. Alphanumeric, for example: September 17, 1999
2. Internal format
3. External format

The Report Format Facility always passes the external format dates to the installation’s date conversion exit routine to be converted to the organization’s date format. If you do not specify a conversion routine, or do not choose an external-date format supported by the Tivoli-supplied date conversion routine, the external format dates are in the default format: MM/DD/YYYY. For details on the date conversion routine and how to specify the external date format, refer to the *Tivoli Information Management for z/OS Planning and Installation Guide and Reference*.

The format for date variables is specified on the DATEFMT keyword of the BLGPARMS macro. The format for time variables is specified on the TIMECNV keyword of the BLGPARMS macro. The *Tivoli Information Management for z/OS Planning and Installation Guide and Reference* describes how to use the BLGPARMS macro.

**System Time**

- `&ZCTIME` Time the report processing began (hh:mm:ss)
- `&ZECTIME` Time the report processing began in external format
- `&ZICTIME` Time the report processing began in internal format

**System Dates in Alphanumeric Format**

- `&ZAPDATE` Yesterday’s date
- `&ZACDATE` Date the report processing began
- `&ZANDATE` Tomorrow’s date

**System Dates in External Format**

- `&ZEAPDATE` Yesterday’s date
- `&ZECDATE` Date the report processing began
- `&ZENDATE` Tomorrow’s date

**System Dates in Internal Format**

- `&ZIPDATE` Yesterday’s date
- `&ZICDATE` Date the report processing began
- `&ZINDATE` Tomorrow’s date

**Day-of-the-Week Variables**

The following predefined variables calculate the day of the week for yesterday, today, and tomorrow.

- `&ZPWKDAY` Day of the week for yesterday
- `&ZCWKDAY` Day of the week for today
- `&ZNWKDAY` Day of the week for tomorrow

Figure 96 on page 365 shows the values for these variables when the current system date is Thursday, 12/23/99.
Weekday Date-Range Variables

The Report Format Facility provides the date of the previous or next occurrence of a particular weekday.

Alphanumeric Format Weekday Date-Range Variables

&ZAPSU Previous Sunday’s date
&ZAPM Previous Monday’s date
&ZAPTU Previous Tuesday’s date
&ZAPW Previous Wednesday’s date
&ZAPTH Previous Thursday’s date
&ZAPF Previous Friday’s date
&ZAPSA Previous Saturday’s date
&ZANSU Next Sunday’s date
&ZANM Next Monday’s date
&ZANTU Next Tuesday’s date
&ZANW Next Wednesday’s date
&ZANTH Next Thursday’s date
&ZANF Next Friday’s date
&ZANSA Next Saturday’s date

External Format Weekday Date-Range

&ZEPSU Previous Sunday’s date
&ZEPM Previous Monday’s date
&ZEPTU Previous Tuesday’s date
&ZEPW Previous Wednesday’s date
&ZEPTH Previous Thursday’s date
&ZEPF Previous Friday’s date
&ZEPSA Previous Saturday’s date
&ZENSU Next Sunday’s date
&ZENM Next Monday’s date
&ZENTU Next Tuesday’s date
&ZENW Next Wednesday’s date
&ZENTH Next Thursday’s date
&ZENF Next Friday’s date
&ZENSA Next Saturday’s date

Internal Format Weekday Date-Range Variables

&ZIPSU Previous Sunday’s date
&ZIPM Previous Monday’s date
&ZIPTU Previous Tuesday’s date
&ZIPW Previous Wednesday’s date
&ZIPTH Previous Thursday’s date
&ZIPF Previous Friday’s date

Day of the week for yesterday....Wednesday
Day of the week for today........Thursday
Day of the week for tomorrow.....Friday

Figure 96. Day-of-the-Week Variables Values
&ZIPSA Previous Saturday’s date
&ZINSU Next Sunday’s date
&ZINM Next Monday’s date
&ZINTU Next Tuesday’s date
&ZINW Next Wednesday’s date
&ZINTH Next Thursday’s date
&ZINF Next Friday’s date
&ZINSA Next Saturday’s date

Figure 97 shows the values for these variables when external format MM/DD/YYYY is requested and the current system date is Wednesday, 06/30/1999.

Previous Thursday's date........06/24/1999
Next Thursday's date..........07/01/1999

Figure 97. Weekday Date-Range Variables in External Format

Weekly Date-Range Variables

The weekly date-range variables calculate the start and end dates for the three weeks: previous, current, and next.

The Report Format Facility sets variables based on the current system date when the report processing begins. The current week is calculated using the current date as the week-end date and six days prior to it as the week-start date. The previous week is calculated as the seven-day range prior to the current-week period, and the next week is the seven-day range following the current-week period. These dates are represented in alphanumeric, external, and internal format.

Alphanumeric Format Weekly Date-Range Variables

&ZAPWSD Previous week start date
&ZAPWED Previous week end date
&ZACWSD Current week start date
&ZACWED Current week end date
&ZANWSD Next week start date
&ZANWED Next week end date

External Format Weekly Date-Range Variables

&ZEPWSD Previous week start date
&ZEPWED Previous week end date
&ZECWSD Current week start date
&ZECWED Current week end date
&ZENWSD Next week start date
&ZENWED Next week end date

Internal Format Weekly Date-Range Variables

&ZIPWSD Previous week start date
&ZIPWED Previous week end date
&ZICWSD Current week start date
&ZICWED Current week end date
&ZINWSD  Next week start date
&ZINWED  Next week end date

Figure 98 shows the values for these variables in external format MM/DD/YYYY when the current system date is 9/22/1999.

Previous week start date....9/09/1999
Previous week end date......9/15/1999
Current week start date.....9/16/1999
Current week end date.......9/22/1999
Next week start date.......9/23/1999
Next week end date..........9/29/1999

Figure 98. Weekly Date-Range Variables

Monthly Date-Range Variables
To calculate the date of the first and last days of the previous month, the current month, and the next month, use the following predefined variables.

Alphanumeric Format Monthly Date-Range Variables
&ZAPMSD  Previous month start date
&ZAPMED  Previous month end date
&ZACMSD  Current month start date
&ZACMED  Current month end date
&ZANMSD  Next month start date
&ZANMED  Next month end date

External Format Monthly Date-Range Variables
&ZEPMSD  Previous month start date
&ZEPMED  Previous month end date
&ZECMSD  Current month start date
&ZECMED  Current month end date
&ZENMSD  Next month start date
&ZENMED  Next month end date

Internal Format Monthly Date-Range Variables
&ZIPMSD  Previous month start date
&ZIPMED  Previous month end date
&ZICMSD  Current month start date
&ZICMED  Current month end date
&ZINMSD  Next month start date
&ZINMED  Next month end date

Figure 99 on page 368 shows the values for these variables when external format MM/DD/YYYY is requested and the current system date is 03/31/1999.
Name-of-the-Month Variables

To calculate the names of last month, this month, and next month, use the following list of predefined variables.

&ZPMONTH Name of the previous month
&ZCMONTH Name of the current month
&ZNMONTH Name of the next month

Figure 100 shows the values for these variables when external format MM/DD/YYYY had been requested and the current system date is 03/31/1999.

Name of the last month............February
Name of this month..................March
Name of the next month............April

Figure 100. Name-of-the-Month Variables

Yearly Date-Range Variables

To obtain the 4-digit number for last year, this year, and next year, use the following list of variables:

&ZPYEAR Previous year in YYYY format
&ZCYEAR Current year in YYYY format
&ZNYEAR Next year in YYYY format

Figure 101 shows the values for these variables when the external format MM/DD/YYYY has been requested and the current system date is 03/31/1999.

Previous year........................1998
Current year..........................1999
Next year..............................2000

Figure 101. Yearly Date-Range Variables
User-Specified Date-Range Variables

The user-specified date-range variables are three date ranges that the Report Format Facility generates from the date-range values specified in your profile. You can enter these on the prompting panels in the profile dialog. The maximum value for a date range is 365. The default value is 7.

The three date periods are generated based on the current system date when report processing begins. The current date is used as the current period end date. The Report Format Facility calculates the current period start date by subtracting one less than the number of days specified in your profile for the current period date range from the current period end date. If the current period date range in your profile is 0, no dates are generated for the current period. The formula is:

\[
\text{START} = \text{END} - (\text{RANGE} - 1).
\]

The previous period end date is the date before the current period start date. If there is no current period start date, yesterday's date is used. The Report Format Facility calculates the previous period start date by subtracting one less than the number of days specified in your profile for the previous period date range from the previous period end date. If the previous period date range in your profile is 0, no dates are generated for the previous period.

Tomorrow's date is the next period start date. The Report Format Facility calculates the next period end date by adding one less than the number of days specified in your profile for the next period date range to the next period start date. If the next period date range in your profile is 0, no dates are generated for the next period. These dates are represented in alphanumeric, external, or internal format.

When you request a standard report that uses these dates, the Report Format Facility displays the dates and prompts you to either approve or change these default dates. You can enter different dates for those periods or clear the date fields. If, as the result of these user actions, only one date for a particular range is specified, both variables for that range are set to the same date; this results in a one-day range. You cannot clear all date fields on the report panel that prompts for the date range; you must specify at least one date.

You can create User RFT reports that use the date ranges in your user profile but do not prompt you with the date-range panel when you request the report. In this case, you must change your profile if you want to change the date ranges.

Alphanumeric Format User Date-Range Variables

The Report Format Facility obtains the alphanumeric format user-specified dates by calling the installation-defined or the Tivoli-supplied date-conversion routine. The conversion routine converts the user's external format dates to internal format and then converts the internal dates to alphanumeric format: month day, year. For example, JULY 25, 1999.

Tivoli Information Management for z/OS provides one internal date format: YYYY/MM/DD.

Refer to the Tivoli Information Management for z/OS Planning and Installation Guide and Reference for information on date formats.

&ZAPPSD Previous period start date
&ZAPPED Previous period end date
&ZACPSD Current period start date
&ZACPED Current period end date
&ZANPSD Next period start date
&ZANPED    Nex period end date

External Format User Date-Range Variables
The external format user-specified date variables use the same format as that entered by the user. When the user-specified date ranges are to be collected from panels, you must specify the appropriate s-word index. Table 53 lists each variable and its s-word index. The corresponding s-word indexes apply to both internal and alphanumeric date formats.

Table 53. External Format Date Variables and S-Word Index Keys

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
<th>S-Word Index Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>&amp;ZEPPSD</td>
<td>Previous period start date</td>
<td>S0DE2</td>
</tr>
<tr>
<td>&amp;ZEPPED</td>
<td>Previous period end date</td>
<td>S0DE3</td>
</tr>
<tr>
<td>&amp;ZECPSD</td>
<td>Current period start date</td>
<td>S0DE4</td>
</tr>
<tr>
<td>&amp;ZECPED</td>
<td>Current period end date</td>
<td>S0DE5</td>
</tr>
<tr>
<td>&amp;ZENPSD</td>
<td>Next period start date</td>
<td>S0DE6</td>
</tr>
<tr>
<td>&amp;ZENPED</td>
<td>Next period end date</td>
<td>S0DE7</td>
</tr>
</tbody>
</table>

Internal Format User Date-Range Variables
You can obtain the internal format user-specified dates by calling the installation-defined or Tivoli-supplied date-conversion routine to convert the user’s external format dates to the internal format.

Tivoli Information Management for z/OS provides one internal date format: YYYY/MM/DD. Refer to the Tivoli Information Management for z/OS Planning and Installation Guide and Reference for information on date formats.

&ZIPPSD    Previous period start date
&ZIPPED    Previous period end date
&ZICPSD    Current period start date
&ZICPED    Current period end date
&ZINPSD    Next period start date
&ZINPED    Next period end date

The example in Figure 102 presents user-specified date-range variables in the YYYY/MM/DD internal date format when the current system date is 08/06/1999 and your profile contains the following date ranges:

Previous period date range  5
Current period date range  10
Next period date range     1

Previous period start date....1999/07/23
Previous period end date......1999/07/27
Current period start date....1999/07/28
Current period end date......1999/08/06
Next period start date.......1999/08/07
Next period end date.........1999/08/07

Figure 102. User-Specified Date-Range Variables in Internal Format YYYY/MM/DD
Search-Count Variables

Search variables are used in arithmetic statements and to print the total records found for the indicated level.

The 10 search-count variables contain the number of records that matched the search arguments specified for each of the 10 search levels. The Report Format Facility determines the number of records by counting the records in the resultant list after the search and the merge (if requested) with the list specified by the MERGE keyword. The level-0 search argument is the argument that the user specifies when issuing the Tivoli Information Management for z/OS REPORT command. The SEARCH command specifies the remaining search arguments in the RFT.

You can use the search count variable for level-0 to test whether an RFT is being used by the PRINT or REPORT command. This variable always contains 0 when the PRINT command is issued and usually contains a number greater than 0 when the REPORT command is issued. In the following situations, the REPORT command returns a 0:

- The search argument passed to the RFT does not return any records, but the database contains records.
- No records are in the database.

Following are the predefined search-count variables:

- &ZRECSL0 Number of records found at level 0
- &ZRECSL1 Number of records found at level 1
- &ZRECSL2 Number of records found at level 2
- &ZRECSL3 Number of records found at level 3
- &ZRECSL4 Number of records found at level 4
- &ZRECSL5 Number of records found at level 5
- &ZRECSL6 Number of records found at level 6
- &ZRECSL7 Number of records found at level 7
- &ZRECSL8 Number of records found at level 8
- &ZRECSL9 Number of records found at level 9

Sort Variables

You can use sort variables to obtain the prefix or the data that was used to sort the record at the current level. Where practical, the use of the data contained in the sort-data variable is recommended over extracting the same data from the record, because the retrieval time is faster.

The sort variables are especially useful for determining sort breaks and for doing secondary searches. However, because a separate set of sort variables is maintained for each search level, and a SEARCH statement immediately changes the level, you cannot use the sort variables directly for a subsequent SEARCH statement. If, for example, you wanted to find all activity records associated with each change record, and you coded the following set of SEARCH statements, the second search would find no matches because the &ZSORTAD1 variable would be for the second-level search and would thus be null.

```
SEARCH ARGUMENT(150006 150Cbc = 150007) SORT(RNID/.)
SEARCH ARGUMENT(150007 RNOR/&ZSORTAD1) SORT(RNID/.)
```

To achieve the desired result, you must set an interim variable equal to the sort variable and use that interim variable on the second search statement, as follows:
Sort-Prefix Variables
The sort-prefix variables are useful for determining which particular prefix was used to extract the data when sorting on a prefix that contains an asterisk, such as \texttt{SORT(SP**/.)}, where multiple prefixes would be found for the abbreviated form.

\&ZSORTAP1 Prefix associated with the first sort field
\&ZSORTAP2 Prefix associated with the second sort field
\&ZSORTAP3 Prefix associated with the third sort field
\&ZSORTAP4 Prefix associated with the fourth sort field
\&ZSORTAP5 Prefix associated with the fifth sort field
\&ZSORTAP6 Prefix associated with the sixth sort field

Sort-Data Variables
The sort-data variables contain the current-level data for the first six sort fields specified.

\&ZSORTAD1 Data associated with the first sort field
\&ZSORTAD2 Data associated with the second sort field
\&ZSORTAD3 Data associated with the third sort field
\&ZSORTAD4 Data associated with the fourth sort field
\&ZSORTAD5 Data associated with the fifth sort field
\&ZSORTAD6 Data associated with the sixth sort field

IF-Command Variables
If-command variables are used only on the IF command statement with the equal (=) or not equal (\texttt{\textasciitilde} =) operator; they include the initial processing variables for test and level.

Test Variables
The test variables have no defined value and are used with the IF command only to test for the presence of prefixes, s-words, or data in a record or value.

\&ZIFFIND Checks the current record for the existence of the specified p-word or the associated s-word.
\&ZIFDATA Checks the current record for the existence of data associated with the indicated p-word or s-word, or checks for the existence of data associated with a variable. The Report Format Facility considers blanks as data.

Level-Initial-Processing Variables
The level-initial-processing variables indicate when the first record for a search level is being processed.

An IF statement specifies these variables to provide initialization or “first-time” processing within their corresponding levels. They must be specified with the VALUE keyword and cannot be used with other variables or as part of a string. They must be tested using the \&ZIFDATA variable.
Initial processing indicator for first level  
Initial processing indicator for second level  
Initial processing indicator for third level  
Initial processing indicator for fourth level  
Initial processing indicator for fifth level  
Initial processing indicator for sixth level  
Initial processing indicator for seventh level  
Initial processing indicator for eighth level  
Initial processing indicator for ninth level  

See “IF - ELSE - EIF” on page 185 for more information on these variables.

Map-Processing Variables

Map-processing variables are used only for hierarchical map processing of hardware and software components.

&ZMAPLVL  Indicates the current hierarchical map level. The map level is two 2-digit fields with odd numbers appearing in positions 1 and 2, and even numbers appearing in positions 3 and 4. If the map level exceeds 99, asterisks are substituted in the appropriate two positions. If map processing is not active, the variable is set to blanks.

The following variables indicate the up connection path for map processing. If the up connection exists, the value assigned to the variable is the record identifier of the component to which the current component is connected. If no up connection exists for the path, or if map processing is not active (that is, map processing is not specified on the SEARCH statement line), the value is null.

&ZUPID  Indicates the record identifier of the component to which the current component is connected.

The following variable indicates whether the component returned during map processing is the start (head component) of a map. If the component returned during map processing is at level one for any map that the search will return, this indicator is set to Y; otherwise, it is set to N. If map processing is not active, that is, map processing is not specified on the SEARCH statement line, the value is null.

&ZMAPTOP  Indicates the top of the map.

Miscellaneous Variables

&ZCURLVL  Indicates the search level (0-9) that is currently processing.

&ZDBID  Indicates the current database identifier (0-5 and 7-9).

&ZDBNAME  Indicates the product name associated with the current database, such as MVS, ACCESS, MANAGEMENT.

&ZLINENO  Indicates the current line number on the page. (Note that &ZLINENO is set to the number of the line that was last written to the data set, even though the output buffer can contain data that is to be written to the next line.)

&ZPRIVCL  Indicates the current privilege class.

&ZUSERID  Indicates the current user ID.

&ZUSRDEPT  Indicates the user department contained in the profile.
Read-Only Variables

&ZUSRNAME
Indicates the user name contained in the profile.

Read/Write Variables

The SET command can read and set values for read/write variables.

&ZPAGENO
Indicates the current page number. The Report Format Facility sets and keeps track of the exact page number between TITLE-ETITLE and HEADER-EHEADER. However, the Report Format Facility subtracts one number when tracking page numbers in the body of an RFT. You can reset and read the variable during report processing.

&ZPATHEND
Indicates the path end. Use this variable to end a path during map processing. If this variable has the value Y when the ESEARCH statement is processed, the current path is ended at the last component returned.

Note: Only the current path is ended; this variable does not end map processing. This variable is automatically reset (to N) when the ESEARCH statement is processed.

You can use the variables listed in Table 54 to capture data from prompting panels when the report is requested. You must use PMF to create panels that use the appropriate s-word indexes for the variables.

You can use a variable to collect additional dates or data, or to control report processing. For example, you could build a panel to prompt for the purpose of the report using one or more of these s-word indexes. You could then use the corresponding variables with PUT statements to print the data in the title or headings. Or, you could use a variable to get a subset of a report; you would provide your name as assignee to get only your assignments on the problem assignee report.

If you use a variable to collect an s-word index, the variable resolves to the s-word itself, not the data associated with the s-word. Variables are substituted only one time; to resolve an s-word index to the data would require two substitutions.

Table 54. Read/Write Variable Descriptions and S-Word Index Keys

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
<th>S-Word Index Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>&amp;ZVAR1</td>
<td>User data field 1</td>
<td>S0D54</td>
</tr>
<tr>
<td>&amp;ZVAR2</td>
<td>User data field 2</td>
<td>S0D55</td>
</tr>
<tr>
<td>&amp;ZVAR3</td>
<td>User data field 3</td>
<td>S0D56</td>
</tr>
<tr>
<td>&amp;ZVAR4</td>
<td>User data field 4</td>
<td>S0D57</td>
</tr>
<tr>
<td>&amp;ZVAR5</td>
<td>User data field 5</td>
<td>S0D58</td>
</tr>
<tr>
<td>&amp;ZVAR6</td>
<td>User data field 6</td>
<td>S0D59</td>
</tr>
<tr>
<td>&amp;ZVAR7</td>
<td>User data field 7</td>
<td>S0D5A</td>
</tr>
<tr>
<td>&amp;ZVAR8</td>
<td>User data field 8</td>
<td>S0D5B</td>
</tr>
<tr>
<td>&amp;ZVAR9</td>
<td>User data field 9</td>
<td>S0D5C</td>
</tr>
<tr>
<td>&amp;ZVAR10</td>
<td>User data field 10</td>
<td>S0D5D</td>
</tr>
<tr>
<td>&amp;ZVAR11</td>
<td>User data field 11</td>
<td>S0D5E</td>
</tr>
</tbody>
</table>
Table 54. Read/Write Variable Descriptions and S-Word Index Keys (continued)

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
<th>S-Word Index Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>&amp;ZVAR12</td>
<td>User data field 12</td>
<td>S0D5F</td>
</tr>
<tr>
<td>&amp;ZVAR13</td>
<td>User data field 13</td>
<td>S0D60</td>
</tr>
<tr>
<td>&amp;ZVAR14</td>
<td>User data field 14</td>
<td>S0D61</td>
</tr>
<tr>
<td>&amp;ZVAR15</td>
<td>User data field 15</td>
<td>S0D62</td>
</tr>
<tr>
<td>&amp;ZVAR16</td>
<td>User data field 16</td>
<td>S0D63</td>
</tr>
<tr>
<td>&amp;ZVAR17</td>
<td>User data field 17</td>
<td>S0D64</td>
</tr>
<tr>
<td>&amp;ZVAR18</td>
<td>User data field 18</td>
<td>S0D65</td>
</tr>
<tr>
<td>&amp;ZVAR19</td>
<td>User data field 19</td>
<td>S0D66</td>
</tr>
<tr>
<td>&amp;ZVAR20</td>
<td>User data field 20</td>
<td>S0D67</td>
</tr>
</tbody>
</table>

Example of How to Use Read/Write Variables

To learn how to add these variables to a panel, refer to the [Tivoli Information Management for z/OS Panel Modification Facility Guide](#) for information on modifying report processing.

You could create a panel like the one illustrated here that uses &ZVARn predefined variables.

The RFT on page B73 uses variables &ZVAR1, &ZVAR2, &ZVAR3, and &ZVAR4, which have the same contents as fields 3 through 6 from this panel.

---

RFT Using Read/Write Variables

```c
/* START OF SPECIFICATIONS ***************************************************/
/* */
/* */
/* */
/* CALL REPORT */
/* */
```
FUNCTION: 

THIS REPORT FORMAT TABLE PRODUCES A CALL REPORT BASED ON THE TR TYPE (RECYCLE, SLOW, INFO, STATUS) 

ZICPSD S0DE4 Current period start date (external) 
ZICPED S0DE5 Current period end date (external) 
ZVAR1 S0D54 RECYCLE 
ZVAR2 S0D55 SLOW 
ZVAR3 S0D56 INFO 
ZVAR4 S0D57 STATUS 

MODULE TYPE: REPORT FORMAT TABLE 

*** END OF SPECIFICATIONS ****************************************** 

CALL REPORT 

SECTION SEPARATION(1) TEST(N) PRINT(N) /* ENTERED SECTION */ 

SET TYPE, RECYCLE IS DEFAULT */ 

SET NAME(TRTYPE) VALUE(RECYCLE) 
IF VALUE(&ZVAR2) OP(=) VALUE(&ZIFDATA) 
   SET NAME(TRTYPE) VALUE(SLOW) 
EIF 

IF VALUE(&ZVAR3) OP(=) VALUE(&ZIFDATA) 
   SET NAME(TRTYPE) VALUE(INFO) 
EIF 

IF VALUE(&ZVAR4) OP(=) VALUE(&ZIFDATA) 
   SET NAME(TRTYPE) VALUE(STATUS) 
EIF 

SET NAME(ICPSD) VALUE(&ZICPSD) 
SET NAME(ICPED) VALUE(&ZICPED) 
/* IF ONE NOT SPECIFIED, SET IT TO THE OTHER */ 

IF VALUE(&ZICPSD) OP(=) VALUE(&ZIFDATA) 
   SET NAME(ICPSD) VALUE(&ZICPED) 
EIF 

IF VALUE(&ZICPED) OP(=) VALUE(&ZIFDATA) 
   SET NAME(ICPED) VALUE(&ZICPSD) 
EIF 

/* IF NEITHER START NOR END SPECIFIED, SET TO . */ 

IF VALUE(&ZICPSD) OP(=) VALUE(&ZIFDATA) JOIN(&) 
   IF VALUE(&ZICPED) OP(=) VALUE(&ZIFDATA) 
      SET NAME(ICPSD) VALUE('') 
      SET NAME(ICPED) VALUE('.') 
   EIF 

TITLE /* REPORT TITLES 

PUT COLUMN(070) VALUE(PAGE) 
PUT COLUMN(075) VALUE(&ZPAGENO) LENGTH(4) JUSTIFY(R) 

PUT COLUMN(001) VALUE(DATE &ZICDATE) /* REPORT DATE 

/* REPORT TITLES */
PUT COLUMN(001) VALUE(ID: &ZUSERID) /* USERID

PUT COLUMN(026) VALUE(***)
PUT COLUMN(036) VALUE(&TRTYPE CALL REPORT)

PUT COLUMN(068) VALUE(***)
PUT COLUMN(026) VALUE(***)

PUT COLUMN(036) VALUE(FOR &ZICPSD - &ZICPED)
PUT COLUMN(068) VALUE(***)

SPACE LINES(2) EXECUTE(U) /* SPACE 2 LINES
ETITLE /* END OF TITLE

/*********************************************************************/
/* HEADING FOR: */
/* CALL REPORT */
/* */
/*********************************************************************/

HEADING /* SECTION HEADINGS
PUT COLUMN(003) VALUE(DATE) /* DATE OCCURRED
PUT COLUMN(013) VALUE(TIME) /* TIME OCCURRED
IF VALUE(&TRTYPE) OP(=) VALUE(RECYCLE) JOIN(|) /* RECYCLE? OR
IF VALUE(&TRTYPE) OP(=) VALUE(INFO) /* INFO? OR
PUT COLUMN(021) VALUE(NODE) /* NODE NAME
PUT COLUMN(031) VALUE(APPL) /* APPLICATION
PUT COLUMN(042) VALUE(DESCRIPTION) /* DESCRIPTION
EIF
IF VALUE(&TRTYPE) OP(=) VALUE(SLOW) /* IF SLOW REPORT
PUT COLUMN(021) VALUE(APPL) /* APPLICATION
PUT COLUMN(033) VALUE(CITY) /* RESP CITY
PUT COLUMN(054) VALUE(DESCRIPTION) /* DESCRIPTION
EIF
IF VALUE(&TRTYPE) OP(=) VALUE(STATUS) /* IF STATUS REPORT
PUT COLUMN(021) VALUE(APPL) /* APPLICATION
PUT COLUMN(033) VALUE(DESCRIPTION) /* DESCRIPTION
EIF

PUT COLUMN(001) VALUE(=)
PUT COLUMN(041) VALUE(=)

SPACE LINES(1) EXECUTE(U) /* SPACE 1 LINE
EHEADING /* END OF HEADINGS

/*********************************************************************/
/* PROCESSING FOR: */
/* CALL REPORT */
/* */
/*********************************************************************/

SET NAME(SORTVAR) VALUE(NASY/.) /* SET TO SORT ON NODE
IF VALUE(&TRTYPE) OP(=) VALUE(SLOW) JOIN(|) /* IF SLOW REPORT
IF VALUE(&TRTYPE) OP(=) VALUE(STATUS) /* IF STATUS REPORT
SET NAME(SORTVAR) VALUE(COMX/.) /* SORT ON APPLICATION
EIF
SEARCH ARGUMENT(!S0032 TYPE/&TRTYPE DATO/&ZICPSD - DATO/&ZICPED) +
SORT(&SORTVAR) EXECUTE(U) MERGE(0) MAP(N)

/*******************************************************************/
/* DATA FOR: 
/* CALL REPORT 
/* 
**************************************************************************
PUT COLUMN(001) LENGTH(10) DATA(DATO/.) /* DATE OCCURRED
PUT COLUMN(013) LENGTH(05) DATA(TIMO/.) /* TIME OCCURRED
IF VALUE(&TRTYPE) OP(=) VALUE(RECYCLE) JOIN(|)
IF VALUE(&TRTYPE) OP(=) VALUE(INFO)
   PUT COLUMN(020) LENGTH(08) DATA(NASY/.) /* NODE NAME
   PUT COLUMN(030) LENGTH(08) DATA(COMX/.) /* APPLICATION
   PUT COLUMN(040) LENGTH(42) DATA(!S0E0F) /* DESCRIPTION
   EIF
IF VALUE(&TRTYPE) OP(=) VALUE(SLOW)
   PUT COLUMN(020) LENGTH(08) DATA(COMX/.) /* APPLICATION
   PUT COLUMN(030) LENGTH(20) DATA(LOCC/.) /* CITY
   PUT COLUMN(052) LENGTH(30) DATA(!S0E0F) /* DESCRIPTION
   EIF
IF VALUE(&TRTYPE) OP(=) VALUE(STATUS)
   PUT COLUMN(020) LENGTH(08) DATA(COMX/.) /* APPLICATION
   PUT COLUMN(030) LENGTH(45) DATA(!S0E0F) /* DESCRIPTION
   EIF
ESSEARCH /* END ENTERED SEARCH
ESECTION /* END ENTERED SECTION

Figure 103 illustrates the output that is produced when an ‘X’ is entered for field 5 of the sample panel.
Relating Publications to Specific Tasks

Your data processing organization can have many different users performing many different tasks. The books in the Tivoli Information Management for z/OS library contain task-oriented scenarios to teach users how to perform the duties specific to their jobs.

The following table describes the typical tasks in a data processing organization and identifies the Tivoli Information Management for z/OS publication that supports those tasks. See “The Tivoli Information Management for z/OS Library” on page 385 for more information about each book.

**Typical Tasks**

**Table 55. Relating Publications to Specific Tasks**

<table>
<thead>
<tr>
<th>If You Are:</th>
<th>And You Do This:</th>
<th>Read This:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Planning to Use Tivoli Information Management for z/OS</td>
<td>Identify the hardware and software requirements of Tivoli Information Management for z/OS. Identify the prerequisite and corequisite products. Plan and implement a test system.</td>
<td>Tivoli Information Management for z/OS Planning and Installation Guide and Reference</td>
</tr>
<tr>
<td>Installing Tivoli Information Management for z/OS</td>
<td>Install Tivoli Information Management for z/OS. Define and initialize data sets. Create session-parameters members.</td>
<td>Tivoli Information Management for z/OS Planning and Installation Guide and Reference Tivoli Information Management for z/OS Integration Facility Guide</td>
</tr>
<tr>
<td></td>
<td>Define and create multiple Tivoli Information Management for z/OS BLX-SPs.</td>
<td>Tivoli Information Management for z/OS Planning and Installation Guide and Reference</td>
</tr>
<tr>
<td></td>
<td>Define and create APPC transaction programs for clients.</td>
<td>Tivoli Information Management for z/OS Client Installation and User’s Guide</td>
</tr>
<tr>
<td></td>
<td>Define coupling facility structures for sysplex data sharing.</td>
<td>Tivoli Information Management for z/OS Planning and Installation Guide and Reference</td>
</tr>
<tr>
<td>Diagnosing problems</td>
<td>Diagnose problems encountered while using Tivoli Information Management for z/OS</td>
<td>Tivoli Information Management for z/OS Diagnosis Guide</td>
</tr>
<tr>
<td>If You Are:</td>
<td>And You Do This:</td>
<td>Read This:</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>----------------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Administering Tivoli Information Management for z/OS</td>
<td>Manage user profiles and passwords. Define and maintain privilege class records. Define and maintain rules records.</td>
<td>Tivoli Information Management for z/OS Program Administration Guide and Reference.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Tivoli Information Management for z/OS Integration Facility Guide.</td>
</tr>
<tr>
<td></td>
<td>Create or modify GUI workstation applications that can interact with Tivoli Information Management for z/OS. Install the Tivoli Information Management for z/OS Desktop on user workstations.</td>
<td>Tivoli Information Management for z/OS Desktop User's Guide.</td>
</tr>
<tr>
<td>Maintaining Tivoli Information Management for z/OS</td>
<td>Set up access to the data sets. Maintain the databases. Define and maintain privilege class records.</td>
<td>Tivoli Information Management for z/OS Planning and Installation Guide and Reference.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Tivoli Information Management for z/OS Program Administration Guide and Reference.</td>
</tr>
<tr>
<td></td>
<td>Define and maintain the BLX-SP. Run the utility programs.</td>
<td>Tivoli Information Management for z/OS Operation and Maintenance Reference.</td>
</tr>
<tr>
<td>Programming applications</td>
<td>Use the application program interfaces.</td>
<td>Tivoli Information Management for z/OS Application Program Interface Guide.</td>
</tr>
<tr>
<td></td>
<td>Use the application program interfaces for Tivoli Information Management for z/OS clients.</td>
<td>Tivoli Information Management for z/OS Client Installation and User's Guide.</td>
</tr>
<tr>
<td></td>
<td>Create Web applications using or accessing Tivoli Information Management for z/OS data.</td>
<td>Tivoli Information Management for z/OS World Wide Web Interface Guide.</td>
</tr>
</tbody>
</table>
Table 55. Relating Publications to Specific Tasks (continued)

<table>
<thead>
<tr>
<th>If You Are:</th>
<th>And You Do This:</th>
<th>Read This:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customizing Tivoli Information Management for z/OS</td>
<td>Design and implement a Change Management system. Design and implement a Configuration Management system. Design and implement a Problem Management system.</td>
<td>Tivoli Information Management for z/OS Problem, Change, and Configuration Management</td>
</tr>
<tr>
<td></td>
<td>Design, create, and test terminal simulator panels or terminal simulator EXECs. Customize panels and panel flow.</td>
<td>Tivoli Information Management for z/OS Terminal Simulator Guide and Reference</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Tivoli Information Management for z/OS Panel Modification Facility Guide</td>
</tr>
<tr>
<td></td>
<td>Design, create, and test Tivoli Information Management for z/OS formatted reports.</td>
<td>Tivoli Information Management for z/OS Data Reporting User's Guide</td>
</tr>
<tr>
<td></td>
<td>Create a bridge between NetView® and Tivoli Information Management for z/OS applications. Integrate Tivoli Information Management for z/OS with Tivoli distributed products.</td>
<td>Tivoli Information Management for z/OS Guide to Integrating with Tivoli Applications</td>
</tr>
<tr>
<td>Assisting Users</td>
<td>Create, search, update, and close change, configuration, or problem records. Browse or print Change, Configuration, or Problem Management reports.</td>
<td>Tivoli Information Management for z/OS Problem, Change, and Configuration Management</td>
</tr>
<tr>
<td></td>
<td>Use the Tivoli Information Management for z/OS Integration Facility.</td>
<td>Tivoli Information Management for z/OS Integration Facility Guide</td>
</tr>
<tr>
<td>Using Tivoli Information Management for z/OS</td>
<td>Learn about the Tivoli Information Management for z/OS panel types, record types, and commands. Change a user profile.</td>
<td>Tivoli Information Management for z/OS User's Guide</td>
</tr>
<tr>
<td></td>
<td>Learn about Problem, Change, and Configuration Management records.</td>
<td>Tivoli Information Management for z/OS Problem, Change, and Configuration Management</td>
</tr>
<tr>
<td></td>
<td>Receive and respond to Tivoli Information Management for z/OS messages.</td>
<td>Tivoli Information Management for z/OS Messages and Codes</td>
</tr>
<tr>
<td></td>
<td>Design and create reports.</td>
<td>Tivoli Information Management for z/OS Data Reporting User's Guide</td>
</tr>
</tbody>
</table>
Tivoli Information Management for z/OS Courses

Education Offerings

Tivoli Information Management for z/OS classes are available in the United States and in the United Kingdom. For information about classes outside the U.S. and U.K., contact your local IBM representative or visit http://www.training.ibm.com on the World Wide Web.

United States

IBM Education classes can help your users and administrators learn how to get the most out of Tivoli Information Management for z/OS. IBM Education classes are offered in many locations in the United States and at your own company location.

For a current schedule of available classes or to enroll, call 1-800-IBM TEACh (1-800-426-8322). On the World Wide Web, visit:

http://www.training.ibm.com

to see the latest course offerings.

United Kingdom

In Europe, the following public courses are held in IBM’s central London education centre at the South Bank at regular intervals. On-site courses can also be arranged.

For course schedules and to enroll, call Enrollments Administration on 0345 581329, or send an e-mail note to:

  contact_educ_uk@vnet.ibm.com

On the World Wide Web, visit:

  http://www.europe.ibm.com/education-uk

to see the latest course offerings.
Where to Find More Information

The Tivoli Information Management for z/OS library is an integral part of Tivoli Information Management for z/OS. The books are written with particular audiences in mind. Each book covers specific tasks.

The Tivoli Information Management for z/OS Library

The publications shipped automatically with each Tivoli Information Management for z/OS Version 7.1 licensed program are:

- Tivoli Information Management for z/OS Application Program Interface Guide
- Tivoli Information Management for z/OS Client Installation and User’s Guide*
- Tivoli Information Management for z/OS Data Reporting User’s Guide*
- Tivoli Information Management for z/OS Desktop User’s Guide
- Tivoli Information Management for z/OS Diagnosis Guide*
- Tivoli Information Management for z/OS Guide to Integrating with Tivoli Applications*
- Tivoli Information Management for z/OS Integration Facility Guide*
- Tivoli Information Management for z/OS Licensed Program Specification
- Tivoli Information Management for z/OS Master Index, Glossary, and Bibliography
- Tivoli Information Management for z/OS Messages and Codes
- Tivoli Information Management for z/OS Operation and Maintenance Reference
- Tivoli Information Management for z/OS Panel Modification Facility Guide
- Tivoli Information Management for z/OS Planning and Installation Guide and Reference
- Tivoli Information Management for z/OS Problem, Change, and Configuration Management*
- Tivoli Information Management for z/OS Reference Summary
- Tivoli Information Management for z/OS Terminal Simulator Guide and Reference
- Tivoli Information Management for z/OS User’s Guide
- Tivoli Information Management for z/OS World Wide Web Interface Guide

Note: Publications marked with an asterisk (*) are shipped in softcopy format only.

Also included is the Product Kit, which includes the complete online library on CD-ROM.

To order a set of publications, specify order number SBOF-7028-00.

Additional copies of these items are available for a fee.

Publications can be requested from your Tivoli or IBM representative or the branch office serving your location. Or, in the U.S., you can call the IBM Publications order line directly by dialing 1-800-879-2755.
The following descriptions summarize all the books in the Tivoli Information Management for z/OS library.

**Tivoli Information Management for z/OS Application Program Interface Guide**, SC31-8737-00, explains how to use the low-level API, the high-level API, and the REXXX interface to the high-level API. This book is written for application and system programmers who write applications that use these program interfaces.

**Tivoli Information Management for z/OS Client Installation and User's Guide**, SC31-8738-00, describes and illustrates the setup and use of Tivoli Information Management for z/OS’s remote clients. This book shows you how to use Tivoli Information Management for z/OS functions in the AIX®, CICS, HP-UX, OS/2®, Sun Solaris, Windows NT, and OS/390 UNIX® System Services environments. Also included in this book is complete information about using the Tivoli Information Management for z/OS servers.

**Tivoli Information Management for z/OS Data Reporting User's Guide**, SC31-8739-00, describes various methods available to produce reports using Tivoli Information Management for z/OS data. It describes Tivoli Decision Support for Information Management (a Discovery Guide for Tivoli Decision Support), the Open Database Connectivity (ODBC) Driver for Tivoli Information Management for z/OS, and the Report Format Facility. A description of how to use the Report Format Facility to modify the standard reports provided with Tivoli Information Management for z/OS is provided. The book also illustrates the syntax of report format tables (RFTs) used to define the output from the Tivoli Information Management for z/OS REPORT and PRINT commands. It also includes several examples of modified RFTs.

**Tivoli Information Management for z/OS Desktop User's Guide**, SC31-8740-00, describes how to install and use the sample application provided with the Tivoli Information Management for z/OS Desktop. The Tivoli Information Management for z/OS Desktop is a Java-based graphical user interface for Tivoli Information Management for z/OS. Information on how to set up data model records to support the interface and instructions on using the Desktop Toolkit to develop your own Desktop application are also provided.

**Tivoli Information Management for z/OS Diagnosis Guide**, GC31-8741-00, explains how to identify a problem, analyze its symptoms, and resolve it. This book includes tools and information that are helpful in solving problems you might encounter when you use Tivoli Information Management for z/OS.

**Tivoli Information Management for z/OS Guide to Integrating with Tivoli Applications**, SC31-8744-00, describes the steps to follow to make an automatic connection between NetView and Tivoli Information Management for z/OS applications. It also explains how to customize the application interface which serves as an application enabler for the NetView Bridge and discusses the Tivoli Information Management for z/OS NetView AutoBridge. Information on interfacing Tivoli Information Management for z/OS with other Tivoli management software products or components is provided for Tivoli Enterprise Console, Tivoli Global Enterprise Manager, Tivoli Inventory, Tivoli Problem Management, Tivoli Software Distribution, and Problem Service.

**Tivoli Information Management for z/OS Integration Facility Guide**, SC31-8745-00, explains the concepts and structure of the Integration Facility. The Integration Facility provides a task-oriented interface to Tivoli Information Management for z/OS that makes the
Tivoli Information Management for z/OS applications easier to use. This book also explains how to use the panels and panel flows in your change and problem management system.

_Tivoli Information Management for z/OS Master Index, Glossary, and Bibliography_, SC31-8747-00, combines the indexes from each hardcopy book in the Tivoli Information Management for z/OS library for Version 7.1. Also included is a complete glossary and bibliography for the product.

_Tivoli Information Management for z/OS Messages and Codes_, GC31-8748-00, contains the messages and completion codes issued by the various Tivoli Information Management for z/OS applications. Each entry includes an explanation of the message or code and recommends actions for users and system programmers.

_Tivoli Information Management for z/OS Operation and Maintenance Reference_, SC31-8749-00, describes and illustrates the BLX-SP commands for use by the operator. It describes the utilities for defining and maintaining data sets required for using the Tivoli Information Management for z/OS licensed program, Version 7.1.

_Tivoli Information Management for z/OS Panel Modification Facility Guide_, SC31-8750-00, gives detailed instructions for creating and modifying Tivoli Information Management for z/OS panels. It provides detailed checklists for the common panel modification tasks, and it provides reference information useful to those who design and modify panels.

_Tivoli Information Management for z/OS Planning and Installation Guide and Reference_, GC31-8751-00, describes the tasks required for installing Tivoli Information Management for z/OS. This book provides an overview of the functions and optional features of Tivoli Information Management for z/OS to help you plan for installation. It also describes the tasks necessary to install, migrate, tailor, and start Tivoli Information Management for z/OS.

_Tivoli Information Management for z/OS Problem, Change, and Configuration Management_, SC31-8752-00, helps you learn how to use Problem, Change, and Configuration Management through a series of training exercises. After you finish the exercises in this book, you should be ready to use other books in the library that apply more directly to the programs you use and the tasks you perform every day.

_Tivoli Information Management for z/OS Program Administration Guide and Reference_, SC31-8753-00, provides detailed information about Tivoli Information Management for z/OS program administration tasks, such as defining user profiles and privilege classes and enabling the GUI user interface.

_Tivoli Information Management for z/OS Reference Summary_, SC31-8754-00, is a reference booklet containing Tivoli Information Management for z/OS commands, a list of p-words and s-words, summary information for PMF, and other information you need when you use Tivoli Information Management for z/OS.

_Tivoli Information Management for z/OS Terminal Simulator Guide and Reference_, SC31-8755-00, explains how to use terminal simulator panels (TSPs) and EXECs (TSXs) that let you simulate an entire interactive session with a Tivoli Information Management for z/OS program. This book gives instructions for designing, building, and testing TSPs and TSXs, followed by information on the different ways you can use TSPs and TSXs.
**Tivoli Information Management for z/OS User's Guide**, SC31-8756-00, provides a general introduction to Tivoli Information Management for z/OS and databases. This book has a series of step-by-step exercises to show beginning users how to copy, update, print, create, and delete records, and how to search a database. It also contains Tivoli Information Management for z/OS command syntax and descriptions and other reference information.

**Tivoli Information Management for z/OS World Wide Web Interface Guide**, SC31-8757-00, explains how to install and operate the features available with Tivoli Information Management for z/OS that enable you to access a Tivoli Information Management for z/OS database using a Web browser as a client.

Other related publications include the following:

**Tivoli Decision Support: Using the Information Management Guide** is an online book (in portable document format) that can be viewed with the Adobe Acrobat Reader. This book is provided with Tivoli Decision Support for Information Management (5697-IMG), which is a product that enables you to use Tivoli Information Management for z/OS data with Tivoli Decision Support. This book describes the views and reports provided with the Information Management Guide.

IBM Redbooks™ published by IBM’s International Technical Support Organization are also available. For a list of redbooks related to Tivoli Information Management for z/OS and access to online redbooks, visit Web site [http://www.redbooks.ibm.com](http://www.redbooks.ibm.com) or [http://www.support.tivoli.com](http://www.support.tivoli.com)
**Index**

**Special Characters**
&ZMAaplvl, map level variable 373

**A**
access, ODBC 23
activity records 332
Administrator module, Tivoli Decision Support 4
advanced options, data source 29, 43
ampersand as part of data, requirements 149
application function, ODBC 16
application ID 28, 40
applications, enabled for ODBC 26
approval status, calculating totals 292
arithmetic operations 210, 217

**B**
batch mode
example of report in 256
using to run reports 255
blank lines, inserting 227
BLGOXCNT, predefined CALL command exit 238
BLGOXFSD, predefined CALL command exit 239
BLGOXFSR, predefined CALL command exit 240
BLGOXLST, predefined CALL command exit 241
BLGOXPSA, predefined CALL command exit 239
BLGPARMS macro 145
BLGR8PLA, plan chart 136
BLGR8STE, step chart 136
BLGR8SUR, surface chart 136
BLMZZ31, periodic change status 136
BLMZZ46 (configuration diagram RFT) 315
building a summary line, example of 293

**C**
calculating totals for report section, example of 292
CALL command exit
exit return codes 237
functions 231, 235
predefined exits 237
check “from” and “to” dates 240
description 237
format detailed data 239
format list processor data 241
format response chains 240
format search data 239
CALL command exit (continued)
predefined exits (continued)
search and count 237
restrictions 237
CALL command statement
definition 167
format of data used 163
call identifier code 234
call records 356
case sensitive comparisons 190
change records 327
change status, calculating totals 292
changes and related activities, identifying 299
changing ODBC driver defaults 43
characteristics of fields 319
client features 26
code
cumulative totals 297
previous-period processing 296
previous-period totals 297
cognize unparsed string fields 22
Cognos PowerPlay 4
column position
DBCS data 170
keyword in CALL command statement 170
keyword in PUT command statement 201
command statement 167
CALL 167
controlling
conditional processing 185
iterative processing 174
definition 132, 167
DHEADS 173
DO 174
EDO 174
EEOD 153
EEOF 156
EFORMAT 161, 163
EHEADING 160
EIF 185
EJECT 184
ELSE 185
END 184
EOD 153
EOF 156
ESearch 205
ESECTION 161
ETITLE 165
FORMAT 157
HEADING 160
HFORMAT 161
IF 185
LEAVE 174
PUT 194
SEARCH 205
data output exit (continued)
  description 233
  restrictions 235
  return codes 235
data output problems, example of overcoming 304, 306
data source, ODBC 17, 28, 40, 47
data types supported, SQL 20
data view records 18, 35
database
  connection 37
  ID 45
  profile 44
date escape clause, ODBC 110
date fields 135
date formats
  ODBC driver searches 20
  overriding 135
  returning 198
date operations 217
day-of-the-week variables 364
DBCS
  search example 210
Decision Support, introduction 3
defaults, changing ODBC driver 43
defining queries, SQL 40
defining titles and headings in RFTs, example 289
definition statement
  definition 132
  EFORMAT 163
  EOD 153
  EOF 156
  ESECTION 161
  ETITLE 165
  FORMAT 157
  HEADING 160
  HFORMAT 161
  SECTION 161
  TITLE 165
description, service function exit routines 232
detail report 264
determining problems, example 302
DHEADS command statement 173
dictionary index key 150
disconnecting from host, ODBC driver 40
Discovery Interface 5
DO command statement 174
drilling down to data 8
driver functions, ODBC 17
driver manager, ODBC 17
duplicate row removal, restriction of 21

data center records 351
positioning chart
  CALL command 170
  PUT command 202
data attribute records 18, 33
data collection panel, adding 142
data entries, comparing values 193
data model records 25, 33
data output exit
  communication control block 233

D

data
  center records 351
  positioning chart
    CALL command 170
    PUT command 202
data attribute records 18, 33
data collection panel, adding 142
data entries, comparing values 193
data model records 25, 33
data output exit
  communication control block 233

E

EDO command statement 174
EEOD definition statement 153
EEOF definition statement 156
EFORMAT definition statement 161, 163
EHEADING definition statement 160
EIF command statement 185
**EJECT command statement** 184
**ELSE command statement** 185
**END command statement** 153, 184
**EOD definition statement** 153
**EOF definition statement** 156
**error messages, ODBC driver** 40, 47
**ESEARCH command statement** 205
**ESECTION definition statement** 161
**ETITLE definition statement** 165
**exclamation point as part of data, requirements** 149
**exit routine** 167
**expanding date fields** 135
**expressions, SQL** 20
**external date format** 135, 222
**external time format**
  - default 223

**F**
**field characteristic**
  - length of 319
  - record panel 319
**filters, report** 41
**FORMAT definition statement** 157
**format records for printing, RFT names to** 266
**formats, saving ODBC report** 41
**freeform text**
  - area 46
  - line length 46
  - maximum lines limit 45
  - processing 23
**freeform text data types** 319
**function calls, ODBC interface** 16
**functions, ODBC** 53

**G**
**general reports**
  - detail 264
  - line summary 263
  - page summary 264
  - search results list 263
**GLOSSARY command** 20
**graphic character substitutions** 135
**graphics**
  - Tivoli Decision Support 8
  - workstation alternatives 8, 41

**H**
**hardware**
  - component records 334
  - connection record 336
  - feature records 337
  - financial records 348
  - (continued)
    - map processing 373
    - subcomponent records 338
**HEADING definition statement** 160
**headings**
  - defining 289
  - identifying columnar data with 160
  - report subtitles 160
**HFORMAT definition statement** 161
**history data, format of** 157, 161
**history entries, comparing values** 193
**HLAPI**
  - message option 44
  - requester startup 26
  - table count 45
  - transactions
    - HL01, initialization 40
    - HL02, termination 40
    - HL06, retrieve record 19
    - HL11, record inquiry 19
**HLAPI for Windows NT** 26
  - horizontal iteration 168
  - host API timeout interval 45
  - host log spool interval 45

**I**
**ICLUI environment variables** 48
**IF command statement**
  - definition 185
  - level-initial-processing variables 372
  - test variables 372
**information types for SQLGetInfo** 119
**installation procedure, ODBC driver** 27
**internal date formats** 20
**intranet, report data access** 4
**inventory records** 354
**inventory reports** 265
**ISPF/PDF editor, use of** 136
**iteration**
  - horizontal 197
  - vertical 197

**J**
**JCL statements**
  - special meanings 256
  - using to run reports in batch mode 256
**joining tables** 21
**journal fields, list of** 319

**K**
**key, dictionary index** 150
keyword
  configuration diagram RFT 315
description 148
dictionary index key 150
notational convention 147
prefix 149
special characters 149
variable 149
keywords with mixed data 149

L
last database record, processing 153
last section of RFT, processing 156
leading blanks as part of data, requirements 149
leading zeros, suppressing 215
LEAVE command statement 174
Level 1 and 2 compliance, ODBC 20
line
  length 169, 200
  summary report 263
list processor
  exit routine parameters 241
  formatting data 241
  RFTs 242
LLAPI message option 44
locate control flags 244
locate service function–X'D3'
description 243
locate control flags 244
problem record example 244
return codes 246
  service-function router 244
single-occurrence indicator flags 245
log file, tracing functions 48
logon, ODBC driver 25, 28
Lotus 1-2-3 26, 38
Lotus Approach 26, 38
LRECL (logical record length)
  CALL command 169
  PUT command 200

Microsoft Access 26, 39
Microsoft Excel 26
mixed case comparisons 190
mixed case data 209, 262
mixed data in keywords 149
model hardware
  component records 339
  feature records 341
  subcomponent records 342
model software
  component records 346
  feature records 347
modifying
  panels 136
  RFT 135
  standard RFT, example 295
monthly date-range variables
  alphanumeric format 367
  external format 367
  internal format 367
move service function–X'D4'
description 247
return codes 247
  service function router 247
Multiclient Remote Environment Server (MRES), reporting
  with 6, 25
multiple databases
  nesting search levels 207
  multiple-level searches 206
  multiple sections, creating 161
  multiple-tier driver, ODBC 17
  mutiple response fields, ODBC driver 23

N
name-of-the-month variables 368
names, member, of standard reports 265
nested conditional statements 193
nested search 206
nesting search levels 207
null statements 162

M
macro
  BLGPARMS 145
map-processing
    hardware and software components 373
    variables 373
maximum search hits, ODBC 45
member names of standard reports 265
memory requirements 143
menu panel, adding 143
merge 207
merge, with BLGOXCNT exit 238
merging data, ODBC driver 41
messages, ODBC driver 47

ODBC
  access flag 23
  applications supporting 26
  compliance 20
  concepts 16
  conformance 19
  data source 17
  diagram, components 16
  diagram, with Tivoli Decision Support 4
  driver description 15
  driver functions 17
  driver manager 17
  driver manager tracing 48
ODBC (continued)
  driver tracing 48
  functions supported 107
  return and reason codes, function 53
  role with Tivoli Decision Support 4
  ODBC.INI file 17
Open Database Connectivity (ODBC) 15
OS/390 Text Search support 360
outage standards, identifying problems that exceed 307
output data, writing 170
output destination for reports
  DDNAME 253
  defining 251
  DSNAME 252
  SYSOUT 252
output line, placing data on 194, 291
overcoming data output problems, example 304, 306
overnight, problems entered 302

P
  page
    eject 184
    length 169, 200
      summary report 264
  panels, modification of 136
  parameter data blocks (PDBs) 44
  parenthesis as part of data, requirements 149
  partitioned data set, used with RFTs 136
  password, data source information 28, 40
  people records 357
  performance considerations of RFTs 143
  planning changes to RFT 135
  platforms supported, ODBC driver 25
  positioning of data on output line
    CALL command 170
    PUT command 200
  PowerPlay format 4
  predefined CALL command exits
    BLGOXCNT, search and count 237
    BLGOXFSD, format detail data 239
    BLGOXFSR, format response chains 240
    BLGOXLST, format list processor data 241
    BLGOXPSA, format search argument 239
    BLMOXDCK, check from and to dates 240
  predefined variable
    read-only variables 363
    read/write variables 374
  prefix
    keywords 149
  prefixes
    indexes, list of 319
  prerequisites, ODBC driver 25
  previous-period processing
    code 296
    example 292
    PUT statements 291
  previous-period totals, code for 297
PRINT command
  defining destination 251
  record output 266
printing
  defining destination 251
  privilege class 44
  privilege class records 320
  problem record
    description, example 300
    panel fields 323
  problems
    determining, example 302
  procedure, ODBC driver usage 37
  processing
    RFT 133
    time for an RFT 144
PUT command statement
  format of data in 163
  keyword combinations not valid 202
  syntax 194
  putting data on an output line, example of 291

Q
  query, changing SQL 41

R
  read-only variables
    day-of-the-week variables 364
    description 363
    IF-command variables 372
    map-processing variables 373
    miscellaneous variables 373
    monthly date-range variables 367
    name-of-the-month variables 368
    search-count variables 371
    sort variables 371
    system date and time variables 363
    user-specified date-range variable 369
    weekday date-range variables 365
    weekly date-range variables 366
    yearly date-range variables 368
  read/write variables 374
  record
    panel fields 319
  records
    data attribute 18
    data view 18
  remote environment server 25
  REPORT command, defining destination 251
  report data sets
    writing data 167
    writing headings 173
  report exit routine
    CALL command exit 231, 235
    controlling 232
report exit routine (continued)
  data output exit 231, 233
  introduction 231
  predefined CALL command exits 237
  service function 232
  service functions 243
  types 231
report exit routines
  introduction 134
report format, ODBC driver 41
Report Format Facility
  introduction 131
  uses 131
reports
  alternate methods of running 255
  defining output destination 251
  standard 261
requirements
  ODBC driver 25
  Tivoli Decision Support 11
requirements, virtual storage 143
restrictions, ODBC driver 23
restrictions for data output exits 235
return codes
  CALL command exit 237
  data output exit 235
return codes, ODBC driver 52
RFT
  creating 135, 136, 289
  examples 132, 289
  how to code 137
  introduction 132
  modifying 135, 136, 295
names to format records for printing 266
performance considerations 143
performance time 145
planning changes 135
processing 133
processing time 144
risk-assessment totals 289
sections 133
sequence numbers, explanation of RFT 300
statement
  explanation 147
  introduction to 132
  keywords 148
  notational convention 147
  search level 151
  syntax and rules 147, 148
storage 133
  testing 139
risk-assessment totals
  RFTs 289
roles, Tivoli Decision Support 7
rules records 353

S
s-word
  combinations with SEARCH command 208
  indexes, list of 319
saving report formats, ODBC 41
Seagate Crystal Reports 26, 37
SEARCH command statement 205
search-count variables 371
search levels
  description 151
  ending 184
  SEARCH command 206
search results list report 263
searches, maximum hits (ODBC) 45
searching for records 205
SECTION definition statement 161
sections, introduction to 133
security ID 28
security ID, data source information 39
security logon 25
SELECT statement 118
sequence numbers, explanation of RFT 300
Service Desk Bridge 323
service functions for exit routines
  function codes 243
  locate service function–X’D3’ 243
  move service function–X’D4’ 247
  variable read service function–X’D9’ 247
  variable set service function–X’E2’ 248
  write service function–X’E6’ 249
service records 352
session member 44
session-parameters member 145
SET command statement 210
SETD command statement
  expected inputs 224
  functions 224
  results 224
  syntax 217
setup, verifying HLAPI/NT 26
single-occurrence indicator flags 245
single quote as part of data, requirements 149
single-tier driver, ODBC 17
software
  component records 342
  connection records 344
  feature records 345
  financial records 349
  map processing 373
sort variables
  sort-data 372
  sort-prefix 372
  usage 371
sorting data, ODBC 21
SPACE command statement 227
special characters as part of data, requirements 205
SQL
  statements, elements 118
SQLGetInfo information types and values 119
SQLSTATE codes 52
SRCs, running reports with 255
standard reports
description of 261
format for 262
member names of 265
standard RFT, example of modifying 295
start column positions, defining 228
starting client requester 26
statement
command 167
command/definition, summary 151
definition 153
explanation of RFT 147
introduction 132
JCL, to run reports in batch mode 256
SQL
ten entering 20
syntax errors 50
storage
requirements 143
RFT 133
stored response chain records 322
structured query language (SQL)
command syntax and parameters 107
conformance levels 20
elements used 118
ORDER BY 21
parenthesis with 21
percent character limitation 21
restrictions 21
SELECT DISTINCT restriction 21
statements 41
WHERE clause 21
subtitles 160
summary line, building, example of 293
syntax of RFT statements
comment 147
keyword (value) 147
label 147
statement 147
syntax of SQL commands 107
SYSOUT destination entry panel 251
SYSPRINT
DCB information 257
system date variables
algorithmic format 364
date formats 363
external format 364
internal format 364
system records 351
system time variables
external format 364
internal format 364
system time 364

T
TAB command statement 228
table list, data view record 39, 40
tables, joining 21
testing RFTs 139
text data, format of 157, 161
text entries, comparing values 193
text lines limit, freeform 45
text search records 360
TFORMAT definition statement 163
three or more assignees, identifying problems with 310
time escape clause, ODBC 110
time formats
returning 198
time operations 217
time to process an RFT 144
title
defining 289
TITLE definition statement 165
Tivoli Decision Support 3
Discovery Guides 6
Tivoli Information Management for z/OS, makeup 243
Tivoli Information Management for z/OS data source 40
Tivoli Inventory reports 265
topic map, Tivoli Decision Support 7
totals for report section, example 292
tracing
ODBC driver 48
ODBC driver manager 48
types, data 20

U
universal time 198
unparsed string fields, cognized 22
update restriction, database 19
uppercase data 262
user exit 231
user-specified date-range variable 369
algorithmic format 369
external format with s-word index keys 370
internal format 370

V
variable
defining and initializing 210, 217
keyword 149
predefined 363
day-of-the-week 364
IF-Command 372
map-processing 373
monthly date-range 367
name of the month 368
read-only 363
read/write 374
search-count 371
sort 371
system date 363
system time 363
user-specified date-range 369
variable (continued)
  predefined (continued)
    weekday date-range 365
    weekly date-range 366
    yearly date-range 368
variable-read service function–X’D9’
  description 247
  return codes 248
  service function router 247
variable set service function–X’E2’
  description 248
  return codes 249
  service function router 248
verifying HLAPI/NT setup 26
vertical
  iteration 168
  views, Tivoli Decision Support
    detailed 7
    multidimensional 8
virtual storage requirements 143

W
weekday date-range variables
  alphanumeric format 365
  external format 365
  internal format 365
weekly date-range variables
  alphanumeric format 366
  external format 366
  internal format 366
WHERE clause, limiting data 41
Windows NT 4, 25
write service function–X’E6’
  description 249
  return codes 250
  service function router 249
writing of output data 170, 200

Y
yearly date-range variables 368