

Cincom

SUPRA SERVER PDM

Database Administration Guide
(UNIX & VMS)

P25-2260-49



SUPRA[®] Server PDM Database Administration Guide (UNIX & VMS)

Publication Number P25-2260-49

© 1992–2002 Cincom Systems, Inc.
All rights reserved

This document contains unpublished, confidential, and proprietary information of Cincom. No disclosure or use of any portion of the contents of these materials may be made without the express written consent of Cincom.

The following are trademarks, registered trademarks, or service marks of Cincom Systems, Inc.:

AD/Advantage [®]	iD CinDoc [™]	MANTIS [®]
C+A-RE [™]	iD CinDoc Web [™]	Socrates [®]
CINCOM [®]	iD Consulting [™]	Socrates [®] XML
Cincom Encompass [®]	iD Correspondence [™]	SPECTRA [™]
Cincom Smalltalk [™]	iD Correspondence Express [™]	SUPRA [®]
Cincom SupportWeb [®]	iD Environment [™]	SUPRA [®] Server
CINCOM SYSTEMS [®]	iD Solutions [™]	Visual Smalltalk [®]
 gOOj [™]	intelligent Document Solutions [™]	VisualWorks [®]
	Intermax [™]	

UniSQL[™] is a trademark of UniSQL, Inc.
ObjectStudio[®] is a registered trademark of CinMark Systems, Inc.

All other trademarks are trademarks or registered trademarks of their respective companies.

Cincom Systems, Inc.
55 Merchant Street
Cincinnati, Ohio 45246-3732
U.S.A.

PHONE: (513) 612-2300
FAX: (513) 612-2000
WORLD WIDE WEB: <http://www.cincom.com>

Attention:

Some Cincom products, programs, or services referred to in this publication may not be available in all countries in which Cincom does business. Additionally, some Cincom products, programs, or services may not be available for all operating systems or all product releases. Contact your Cincom representative to be certain the items are available to you.

Release information for this manual

The *SUPRA Server PDM Database Administration Guide (UNIX & VMS)*, P25-2260-49, is dated January 15, 2002. This document supports Release 1.3 of SUPRA Server under UNIX and Release 2.4 of SUPRA Server under VMS environments.

We welcome your comments

We encourage critiques concerning the technical content and organization of this manual. Please take the [survey](#) provided with the online documentation at your convenience.

Cincom Technical Support for SUPRA Server PDM

FAX: (513) 612-2000
Attn: SUPRA Server Support

E-mail: helpna@cincom.com

Phone: 1-800-727-3525

Mail: Cincom Systems, Inc.
Attn: SUPRA Server Support
55 Merchant Drive
Cincinnati, OH 45246-3732
U.S.A.



Contents

About this book	ix
Using this document.....	ix
Document organization	x
Revisions to this manual	xi
Conventions	xii
SUPRA Server documentation series	xv
Overview of SUPRA DBA	17
The SUPRA Directory	18
SUPRA DBA functions	21
Normalizing your data	25
Analyze your organization	27
Analyze your data	31
Convert a data view to unnormalized form.....	32
Convert unnormalized form to first normal form	33
Convert first normal form to second normal form	34
Convert second normal form to third normal form	35
Optimize third normal form.....	37
Design your logical database	40
Select relations.....	41
Determine relationships	42
Chart relationships	43
Create operational relationships	44
Optimize relationships	45
Design your physical database	46
Convert the logical design to a valid SUPRA Server database.....	53
Optimize space usage.....	57
Set timing objectives for major processing cycles	58
Optimize your database design.....	60

Using SUPRA DBA	61
Overview of DBA functions	62
Using DBA function keys	65
Signing on to DBA.....	66
Examining an entity.....	69
Modifying an entity	73
Creating an entity.....	77
Deleting an entity	81
Listing entities	87
Connecting entities to each other	91
Disconnecting entities from each other	94
Entering comments.....	96
Displaying online Help	100
Creating and maintaining a database description	101
Starting SUPRA DBA.....	102
Database Description functions	104
Selecting database description options	108
Defining database details	110
Entering database comments.....	119
Defining data sets	120
Defining physical file attributes	123
Selecting data set options.....	132
Defining Data Set Details.....	134
Defining primary records.....	136
Defining related records.....	145
Using RMS files (VMS)	149
Defining RMS records (VMS)	152
Creating and maintaining indices and secondary keys.....	161
Defining an index	163
Defining a Secondary Key	169
Including data-items in a secondary key.....	175
Entering the data-item type (UNIX).....	177
Creating and maintaining buffers.....	178
Defining a task log	181
Defining a system log	182
Defining and running recovery	183
Creating a recovery point.....	185
Defining task log recovery	186
Defining Recovery Unit Journaling for RMS data sets (VMS)	189
Defining system log recovery.....	190

Maintaining and dumping system logs	194
Setting up automatic system log dumping	197
Dumping the system log file online	208
Stages in a system log dump	211
Running recovery	216
Running recovery from SUPRA DBA	217
Running recovery from the command level (VMS) or at the shell (UNIX)....	226
Running recovery in batch mode (VMS) or as a background process (UNIX)...	227
Validating, compiling, printing, and formatting a database	229
Running Validate from DBA	231
Running Compile and print from DBA	235
Running Print from DBA	240
Running Batch Validate, Compile, and Print from the command level (VMS) or at the shell (UNIX)	244
Running Format/Reformat	250
Running Format from DBA	251
Running format from the command level (VMS) or at the shell (UNIX)	257
Running format in batch (VMS) or as a background process (UNIX)	258
Formatting, populating, and checking indices	262
Running CSTUIDX from DBA	265
Running CSTUIDX from the command level	278
Running CSTUIDX in the batch environment	279
Using the Administration functions	283
Defining users	285
Entering user details	287
Entering user comments	288
Unlocking entities	289
Expanding a related data set	291
VMS: Sample responses to the expand prompts	292
UNIX: Sample responses to the expand prompts	293
Resetting a data set load limit	298
Controlling programs (VMS)	303
Examining the details of a program (VMS)	305
Displaying the views used by a program (VMS)	306
Maintaining comments (VMS)	307
Deleting program information from the directory (VMS)	308
Defining Multiple Physical Databases (MPDs)	309
Copying a database description	311
Modifying a copied database description	313
Validating and compiling a copied database	314

Defining logical data-items, domains, validation tables, and views	315
Defining logical data-items.....	316
Examining or modifying a logical data-item	317
Creating a logical data-item	319
Connecting a domain to a data-item.....	329
Deleting a logical data-item	330
Defining domains.....	331
Creating, examining, or modifying a domain	333
Deleting a domain.....	344
Connecting a domain to a validation table.....	344
Connecting a domain to a data-item.....	346
Defining validation tables.....	349
Entering validation table text.....	350
Entering validation table comments.....	351
Deleting a validation table.....	351
Defining views (VMS)	352
Creating and maintaining views through DBA (VMS).....	355
Customizing your EDT environment (VMS).....	367
Deleting a view (VMS)	369
Defining user authorization (VMS).....	370
Minimum and maximum values	371
VMS environments	371
Database descriptions	371
SUPRA Server physical data manager.....	372
SUPRA Server primary files	372
SUPRA Server related files	373
SUPRA Server index files.....	373
RMS files	374
Validation of use-entered data.....	374
Views	375
SPECTRA query language	376
Supported data types.....	377
Supported data access techniques	377
UNIX environments.....	378
Database descriptions	378
SUPRA Server physical data manager.....	379
SUPRA Server primary files	379
SUPRA Server related files	380
SUPRA Server index files.....	380
Supported data-access techniques	381
Format of entries for DOMAIN-UNIT	383
Index	393

About this book

Using this document

The *SUPRA Server PDM Database Administration Guide (UNIX & VMS)*, P25-2260, documents SUPRA[®] Database Administration (SUPRA DBA). SUPRA DBA is a menu-driven tool that allows you to create and maintain a database description.

This manual shows you how to perform the functions available within SUPRA DBA. The functions that are available depend on the operating environment under which you are running.

Under VMS these functions include:

- ◆ Defining a database description, including secondary keys, domains, validation tables, task and system logs
- ◆ Validating and compiling the completed database description
- ◆ Formatting the physical data files
- ◆ Defining and running recovery
- ◆ Administrative tasks including defining user names and changing the structure of databases and maintaining RDM application programs enrolled on the SUPRA Directory
- ◆ Creating and maintaining views using the DBA EDIT/EDT interface

Under UNIX these functions include:

- ◆ Defining a database description, including data sets, secondary keys, buffers, task and system logs
- ◆ Validating and compiling the completed database description
- ◆ Formatting the physical files
- ◆ Defining and running recovery
- ◆ Administrative tasks including defining user names and changing the structure of live databases

Document organization

The information in this manual is organized as follows:

Chapter 1—Overview of SUPRA DBA

Introduces the SUPRA DBA and what it does.

Chapter 2—Normalizing your data

Describes how to design and set up your database.

Chapter 3—Using SUPRA DBA

Describes how to use SUPRA DBA, the central component of SUPRA Server.

Chapter 4—Creating and maintaining a database description

Describes the screens you use to define database descriptions and their associated data sets, indices, buffers, task and system logs.

Chapter 5—Defining and running recovery

Describes the three recovery techniques (task logging, system logging and shadow recording) and how to use them.

Chapter 6—Validating, compiling, printing, and formatting a database

Describes how to perform these operations.

Chapter 7—Using the Administration functions

Describes how to use the database Administration functions to control the operation of your databases.

Chapter 8—Defining Multiple Physical Databases (MPDs)

Describes the MPD facility, which allows you to define databases that share the same logical structure but are implemented with different physical database files.

Chapter 9—Defining logical data-items, domains, validation tables, and views

Describes how to create the two types of views (base and derived) that make up the conceptual and external schemas.

Appendix—Format of entries for DOMAIN-UNIT

Lists the valid domain units of measurement and their abbreviations.

Index

Revisions to this manual

The following changes have been made for this release:

- ◆ Information on minimum and maximum values for designing and modifying data sets was added under “[Defining database details](#)” on page 110.
- ◆ Information about the Expand utility for UNIX was added as a note under “[Expanding a related data set](#)” beginning on page 291.

Conventions

The following table describes the conventions used in this document series:

Convention	Description	Example
Constant width type	Represents screen images and segments of code.	<pre>PUT 'customer.dat' GET 'miller\customer.dat' PUT '\DEV\RMT0'</pre>
Slashed b (<i>b</i>)	Indicates a space (blank). The example illustrates that four spaces appear between the keywords.	<pre>BEGNbbbbSERIAL</pre>
Brackets []	Indicate optional selection of parameters. (Do not attempt to enter brackets or to stack parameters.) Brackets indicate one of the following situations:	
	A single item enclosed by brackets indicates that the item is optional and can be omitted.	[WHERE <i>search-condition</i>]
	The example indicates that you can optionally enter a WHERE clause.	
	Stacked items enclosed by brackets represent optional alternatives, one of which can be selected.	<pre>[<u>WAIT</u>] [NOWAIT]</pre>
	The example indicates that you can optionally enter either WAIT or NOWAIT. (WAIT is underlined to signify that it is the default.)	

Convention	Description	Example
Braces { }	<p>Indicate selection of parameters. (Do not attempt to enter braces or to stack parameters.) Braces surrounding stacked items represent alternatives, one of which you must select.</p> <p>In the example, you must enter ON or OFF when using the MONITOR statement.</p>	<pre>MONITOR { ON OFF }</pre>
<p><u>Underlining</u> (In syntax)</p>	<p>Indicates the default value supplied when you omit a parameter.</p> <p>The example indicates that if you do not choose a parameter, the system defaults to WAIT.</p> <p>Underlining also indicates an allowable abbreviation or the shortest truncation allowed.</p> <p>The example indicates that you can enter either STAT or STATISTICS.</p>	<pre>[(WAIT)] [(NOWAIT)]</pre> <hr/> <pre><u>STATISTICS</u></pre>
Ellipsis points...	<p>Indicate that the preceding item can be repeated.</p> <p>In the example, you can enter multiple host variables and associated indicator variables.</p>	<pre>INTO :host-variable [:ind- variable],...</pre>

Convention	Description	Example
UPPERCASE lowercase	In most operating environments, keywords are not case-sensitive, and they are represented in uppercase. You can enter them in either uppercase or lowercase.	COPY MY_DATA.SEQ HOLD_DATA.SEQ
	In the UNIX operating environment, keywords are case-sensitive, and you must enter them exactly as shown.	cp *.QAR /backup
<i>Italics</i>	Indicate variables you replace with a value, a column name, a file name, and so on. In the example you must substitute the name of a table.	FROM <i>table-name</i>
Punctuation marks	Indicate required syntax that you must code exactly as presented. () parentheses . period , comma : colon ' ' single quotation marks	<i>(user-id, password, db-name)</i> INFILE 'Cust.Memo' CONTROL LEN4
SMALL CAPS	Represent a required keystroke. Multiple keystrokes are hyphenated.	ALT-TAB
<div style="border: 1px solid black; padding: 2px; display: inline-block; margin-bottom: 5px;">UNIX</div> <div style="border: 1px solid black; padding: 2px; display: inline-block;">VMS</div>	Information specific to a certain operating system is flagged by a symbol in a shadowed box (for example, UNIX) indicating which operating system is being discussed. Skip any information that does not pertain to your environment.	<div style="border: 1px solid black; padding: 2px; display: inline-block; margin-bottom: 10px;">UNIX</div> To delete these files, return to the shell and use the rm command. <div style="border: 1px solid black; padding: 2px; display: inline-block;">VMS</div> To delete these files, return to the command level and use the DELETE command.

SUPRA Server documentation series

SUPRA Server is the advanced relational database management system for high-volume, update-oriented production processing. A number of tools are available with SUPRA Server including DBA Functions, DBAID, precompilers, SPECTRA, and MANTIS. The following list shows the manuals and tools used to fulfill the data management and retrieval requirements for various tasks. Some of these tools are optional. Therefore, you may not have all the manuals listed. For a brief synopsis of each manual, refer to the *SUPRA Server PDM Digest for VMS Systems*, P25-9062.

Overview

- ◆ *SUPRA Server PDM Digest for VMS Systems*, P25-9062

Getting started

- ◆ *SUPRA Server PDM UNIX Installation Guide*, P25-1008
- ◆ *SUPRA Server PDM VMS Installation Guide*, P25-0147
- ◆ *SUPRA Server PDM UNIX Tutorial*, T25-2262
- ◆ *SUPRA Server PDM VMS Tutorial*, T25-2263

General use

- ◆ *SUPRA Server PDM Glossary*, P26-0675
- ◆ *SUPRA Server PDM Messages and Codes Reference Manual (PDM/RDM Support for UNIX & VMS)*, P25-0022

Database administration tasks

- ◆ *SUPRA Server PDM Database Administration Guide (UNIX & VMS)*, P25-2260
- ◆ *SUPRA Server PDM System Administration Guide (VMS)*, P25-0130
- ◆ *SUPRA Server PDM System Administration Guide (UNIX)*, P25-0132*
- ◆ *SUPRA Server PDM Utilities Reference Manual (UNIX & VMS)*, P25-6220

- ◆ *SUPRA Server PDM Directory Views (VMS)*, P25-1120
- ◆ *SUPRA Server PDM Windows Client Support User's Guide*, P26-7500*
- ◆ *SPECTRA Administrator's Guide*, P26-9220**

Application programming tasks

- ◆ *SUPRA Server PDM Programming Guide (UNIX & VMS)*, P25-0240
- ◆ *SUPRA Server PDM System Administration Guide (VMS)*, P25-0130
- ◆ *SUPRA Server PDM System Administration Guide (UNIX)*, P25-0132*
- ◆ *SUPRA Server PDM RDM Administration Guide (VMS)*, P25-8220
- ◆ *SUPRA Server PDM Windows Client Support User's Guide*, P26-7500*
- ◆ *MANTIS Planning Guide*, P25-1315**

Report tasks

- ◆ *SPECTRA User's Guide*, P26-9561**



Manuals marked with an asterisk (*) are listed twice because you use them for different tasks.



Educational material is available from your regional Cincom education department.

1

Overview of SUPRA DBA

SUPRA DBA is a menu-driven tool that allows you to define the internal, conceptual and external schemas. In doing so you can implement efficient database navigation on both the physical and the logical levels. All the database details you enter using SUPRA DBA are stored on the SUPRA Directory. The Directory holds a centralized definition of your database together with the physical access (primary keys, secondary keys, and linkpaths) you have defined. Thus, you create a database system incorporating:

- ◆ High performance
- ◆ Insulation from the physical database and its structure
- ◆ A high level of database security
- ◆ Increased programmer productivity
- ◆ Fourth-generation prototyping capabilities

The SUPRA Directory

The SUPRA Directory provides central control of SUPRA Server processing. It stores and maintains all database information, including database descriptions, data set details, and user authorization. You access the Directory through the SUPRA DBA functions.

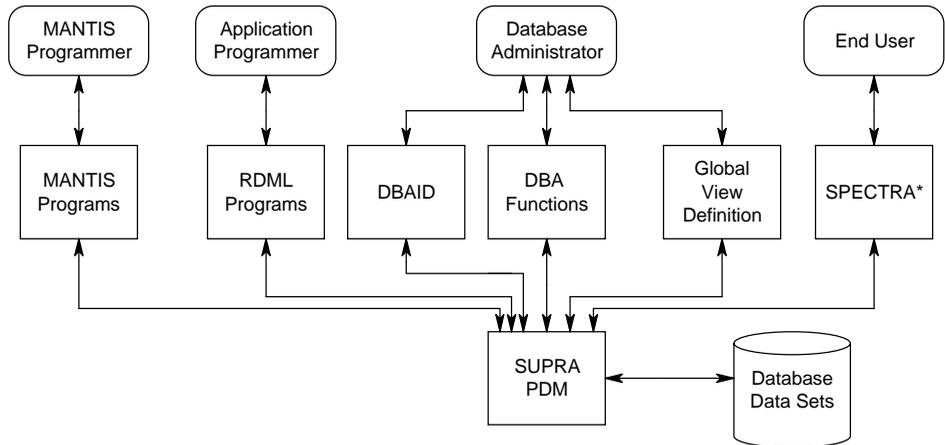
The Directory holds the database information such as entities, relationships, and attributes. Entities are representations. Examples of entities are data sets, data-items, domains, validation tables, and VMS views. Relationships are used to associate pairs of entities. For instance, relationships between data-items and a data set indicate which data-items are defined for the data set. Attributes are details about entities and relationships, such as the number of records that a data set can hold and comments about various entities.

These entities, relationships, and attributes define all of the databases, data sets, users, and so on, available within SUPRA Server. The Directory maintenance functions (referred to as SUPRA DBA functions) allow you to maintain this data and use it to generate database descriptions, manipulate data sets, and maintain user access.

VMS

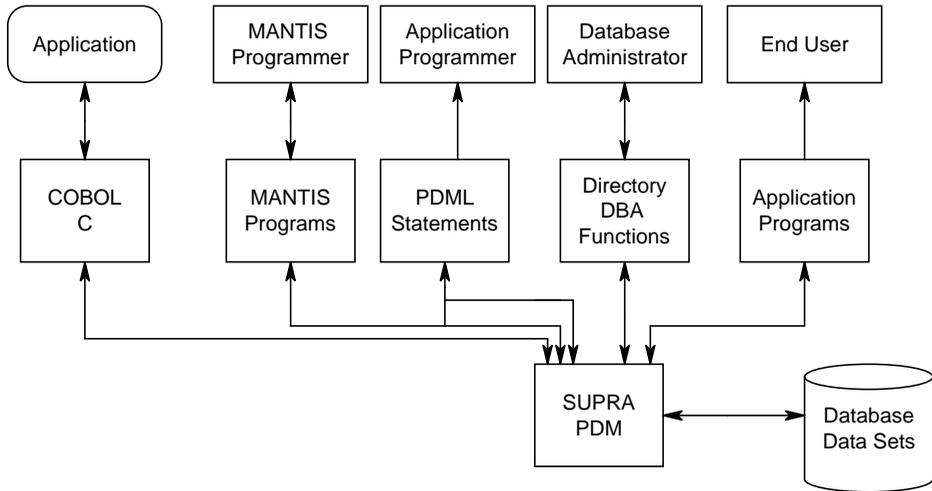
BASIC, COBOL, and FORTRAN preprocessors use this information to generate definitions of views within programs. RDM (the Relational Data Manager) also uses the information to allow particular users to use the requested views at run time and to ensure that the views are up-to-date.

The following figure illustrates the SUPRA Server components and other Cincom tools available in VMS environments. It shows the relationship of the SUPRA DBA functions to the other SUPRA Server components and the type of user for which each component or function is intended.



* VMS VAX only

The following figure illustrates the SUPRA Server components and other Cincom tools available in UNIX environments. It shows the relationship of the SUPRA DBA functions to the other SUPRA Server components, and the type of user for which each component or function is intended.



SUPRA DBA is a menu-driven program.

VMS

DBA will run on any VT100 compatible terminal (this includes upwardly compatible terminals, VT300).

UNIX

DBA will run on any terminal that supports the curses library.

You can press HELP key (function key PF2) at any time to obtain context-related Help text stored in the VMS help library SUPRA_HELP: SUPRA_HELP.HLB, or in the UNIX directory defined by the logical name SUPRA.HELP.

SUPRA DBA functions

SUPRA DBA provides various functions to control development, application, maintenance, and growth of your database (see the following screen). The information following this screen illustration describes the options on the main menu, Function Selection for the DBA:

```

CINCOM SYSTEMS SUPRA DBA-FUNCTION SELECTION FOR THE DBA

                Select required function :

                1 : Database descriptions
                2 : Data sets
(VMS only)    3 : Logical views *
                4 : Logical data-items
                5 : Domains
(VMS only)    6 : Validation tables
                7 : Programs *
                8 : Users
                9 : Unlock functions
               10 : Administration functions

Enter choice no.:

```



* The word *Reserved* displays next to these functions.

Database descriptions

Option 1, Database descriptions, displays the Database Description Function menu through which you create and maintain your databases. See “[Using SUPRA DBA](#)” on page 61 for information on how to move up and down the SUPRA DBA screen hierarchy and how to use some of the generic functions such as Create, Examine, or Delete. “[Creating and maintaining a database description](#)” on page 101, “[Defining and running recovery](#)” on page 183, “[Validating, compiling, printing, and formatting a database](#)” on page 229, and “[Using the Administration functions](#)” on page 283 describe how to create a database, how to define and run database recovery, and how to validate, compile, and print your finished database description.

Data sets

Option 2, Data sets, defines and maintains data sets independently of a database description. See “[Defining data sets](#)” on page 120 for information on data set maintenance.

VMS Logical views

Option 3, Logical views, enables you to perform maintenance on logical views and invoke the SUPRA DBA interface to EDITEDT to define your base and derived views. See “[Defining views \(VMS\)](#)” on page 352 for information on how to use the EDIT/EDT interface. Refer to the *SUPRA Server PDM RDM Administration Guide (VMS)*, P25-8220, for a description of the syntax for base and derived views.

Logical data-items

Option 4, Logical data-items, allows you to assign one or more logical data-items names to each physical data-item, and, if you wish, connects the data-item to a domain. For example, the logical data-item name DATE-OF-BIRTH may be more meaningful to a view or application program than its equivalent physical data-item name of EMPLBRTH. See “[Creating a logical data-item](#)” on page 319 for information on how to define logical data-item names.

Domains

Option 5, Domains, allows you to create and maintain domains containing validation criteria such as data format or type, null values, or reference to a user exit. You can connect a domain to one or more physical data-items from the Domain Function menu. Alternatively, you can make the connection when you define the logical data-item name for the physical data-item. See “[Defining domains](#)” on page 331 for information on how to create and maintain domains.

Validation tables

Option 6, Validation tables, allows you to define a pool of valid values, which you can then connect to a domain. For example, you could create a validation table containing a set of part numbers for one particular component. You can also connect an existing validation table to a domain from the Domain Function menu. See [“Defining validation tables”](#) on page 349 for information on how to create and maintain validation tables.

VMS Programs

Option 7, Programs, controls program information stored in the Directory. You can examine the details of a program, the comments held on the Directory, and the views used. You can also delete the program from the Directory. Note that deleting a program from the Directory does not delete the program; it simply removes the record of it from the SUPRA Directory. See [“Using the Administration functions”](#) on page 283 for additional information.

Users

Option 8, Users, defines and maintains user names on the Directory. You can create users with authorization ranging from read-only to privileged, according to the level of access you want them to have. SUPRA DBA tailors the menus according to the access authorization of the user, displaying only those functions each user is entitled to use. See [“Defining users”](#) on page 285 for additional information.

Unlock functions

Option 9, Unlock functions, resets a locked database description. A database description may become locked if, for instance, a user attempts to access a connected data set during database validation. A locked database is inaccessible until you unlock it using the Unlock function. See [“Unlocking entities”](#) on page 289 for additional information.

Administration functions

Option 10, Administration functions, displays the Administration Function menu. The options on this menu are not directly connected with database and view maintenance. Administrative functions include:

- ◆ Format to create and format the physical files associated with a database description (see [“Running Format/Reformat”](#) on page 250).
- ◆ Utilities to add records, delete records, produce database statistics, or change the record layout without losing existing records (refer to the *SUPRA Server PDM Utilities Reference Manual (UNIX & VMS)*, P25-6220).
- ◆ Expand related data set to increase the size of a related data set (see [“Expanding a related data set”](#) on page 291).
- ◆ Reset data set load limit to change the control interval load limit of a related data set (see [“Resetting a data set load limit”](#) on page 298).
- ◆ Recovery to run system log recovery from within SUPRA DBA (see [“Running recovery from SUPRA DBA”](#) on page 217). You can also invoke system log recovery from the command level (VMS) or at the shell (UNIX) (see [“Running recovery from the command level \(VMS\) or at the shell \(UNIX\)”](#) on page 226).

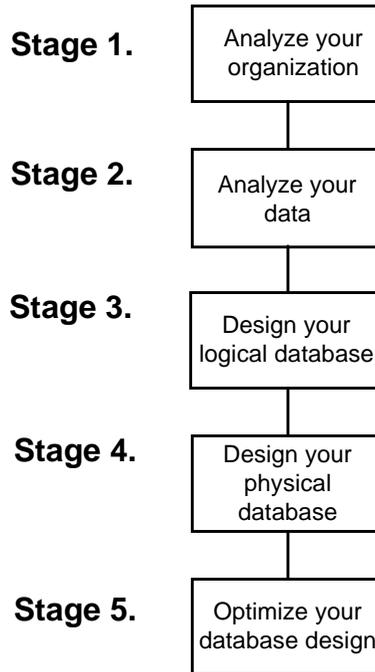
2

Normalizing your data

A SUPRA Server database is made up of data sets, each holding groups of data records. Each customer, for example, is represented by a customer record on the customer data set. Such a record details the customer name, address, credit rating, and other data associated directly with the customer. Likewise, invoice records are held on the invoice data set, product records on the product data set, and so on. The records on each data set relate to those on other data sets in a way that models the data of the organization.

As Database Administrator (DBA), you design, implement, and control the organization's database. The Physical Data Manager (PDM) works most efficiently with data that has undergone the database design process. The following figure presents an overview of this process.

Database design requires that you consider more than just physical factors about data storage and manipulation. Logical considerations about the basic structure of the data should precede physical considerations. Both the logical and physical design processes are critical to producing an accurate and efficient database.



This chapter uses a sample database representing a fictitious company called Burrys to illustrate the database design process. First, the chapter describes the stages to develop an accurate logical database design. These stages are:

- ◆ Analyze your organization
- ◆ Analyze your data and convert to at least third normal form
- ◆ Design your logical database

This chapter then explains how to derive an efficient physical database design from your logical database design.

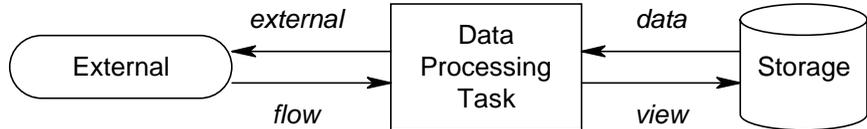
This chapter does not teach you how to design a relational database for your SUPRA Server system. It only increases your awareness of the need for such a process.

Analyze your organization

The first stage of database design analyzes the flow of data throughout the organization. Organizational analysis has three steps:

1. Identify operational functions.
2. Model operational functions and generate data views.
3. List attributes for each data view.

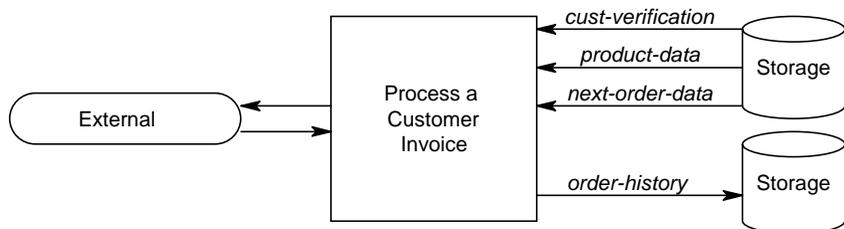
An operational function is a task or transaction that constitutes an essential process in the organization. Processing a customer invoice, for example, is an operational function. You can represent an operational function with the model shown in the following figure:



Because you are concerned with the design and implementation of a database and not with the development of some data-processing application for that database, the focus is on the flow of data between the data-processing task and the storage system — the data view. A customer invoice, such as that shown in the following figure, may represent the transaction:

CUSTOMER: ACME Radio & T.V. 1311 Queen Road Cambridge, MA #EO1649		INVOICE NO. 1376		
BRANCH 2014		DATE 15-JUN-99		
		SALESPERSON 1074		
PRODUCT	DESCRIPTION	QTY	PRICE	VALUE
BZ-2372-L	TELEVISION	1	296.99	296.99
ZB-3621-D	ELECTRIC PLUGS	6	.30	1.80
			TOTAL	298.79

A data view is a subset of the data required to perform a task or to process a transaction. To process a customer invoice, the following data views must go from storage to processing: customer-verification, product-data, and next-order-data. As shown in the following figure, order-history is the data view that must return to storage. Modeling this operational function generates four data views:



Each data view generated has a list of attributes. Using the data views generated by this modeling with the customer invoice, you complete this example of organizational analysis by listing the attributes for each data view:

Customer-verification

- Customer number
- Customer name
- Customer address

Product-data

- Product code
- Product description
- Invoice product quantity
- Invoice product price
- Invoice product value

Next-order-data

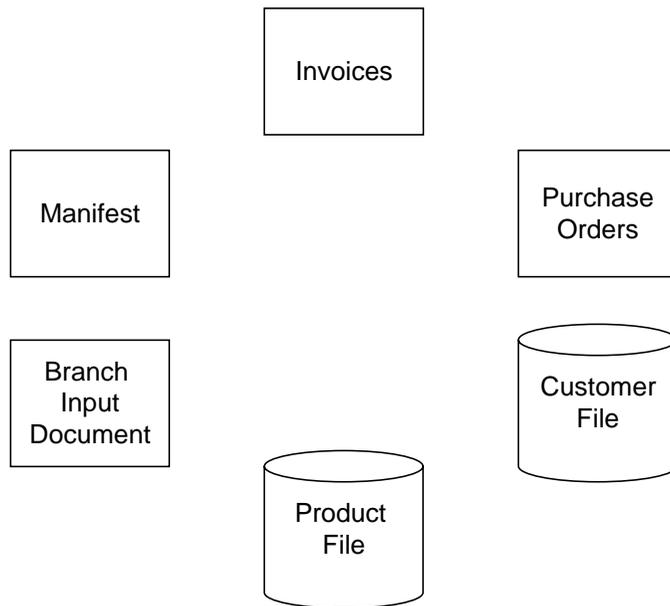
- Order-number

Order-history

- Invoice number
- Customer number
- Customer name
- Customer address
- Invoice date
- Branch number
- Salesman number
- Product code
- Product description
- Invoice product quantity
- Invoice product price
- Invoice product value
- Invoice total

Product quantity, product price, and product value are now preceded with invoice since data occurrences of these attributes vary from invoice to invoice—the product quantity is specifically the quantity of that product listed on a given invoice.

To develop a working database design, you must review all sources of data in your organization in the same way. Like the invoice, data sources are usually found on transaction records, forms, and other routine paperwork. You should also consider people and their tasks, skills, and job knowledge, as well as current computer-based processes, major departmental reports, and so on. The following figure shows the Burrys data sources:



Analyze your data

The second stage of database design is data analysis. Data views are the building blocks of data analysis. Although data views are useful at this early stage, they are often insufficient in the following ways:

- ◆ They depend directly on application processes
- ◆ They reflect data redundancies and ambiguities, they are imprecisely defined
- ◆ They are based on outdated computer-file structures

Data analysis eliminates the insufficiencies of data views by determining the relational structure of the data. This structure provides the foundation for an accurate relational database design. From such a design, you develop a database that will be independent of application programs and file management systems, contain controlled data redundancies, allow new applications to be added without affecting existing applications, and allow easy insertion, deletion, updating, and retrieval of data.

There are several methods of data analysis. The examples in this chapter use third normal form analysis, based on the work of E. F. Codd (IBM, 1970). Third normal form analysis denotes the normalization of data.

Normalization is a way of grouping data attributes on the basis of their fundamental associations. Attribute groups derived from each data view list are called relations because each group is marked by a key attribute, or logical key, to which all other attributes in the group relate.

Normalization derives relations that represent the logical structure of the data. Making changes to a database holding normalized data is easy and efficient.

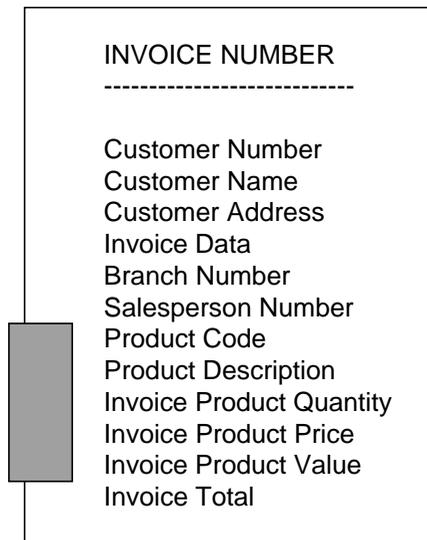
Normalization uses five steps:

1. Convert a data view to unnormalized form.
2. Convert unnormalized form to first normal form.
3. Convert first normal form to second normal form.
4. Convert second normal form to third normal form.
5. Optimize third normal form.

Convert a data view to unnormalized form

Beginning with the list of attributes associated with our example data view labeled order-history (see “Analyze your organization” on page 27), first identify and underline the logical key. The logical key is the attribute you use to access the other attributes in the data view. Examine the process using the view to determine the logical key. For order-history, the invoice number uniquely identifies all the order details required by the data view. Invoice number is therefore the logical key for order-history.

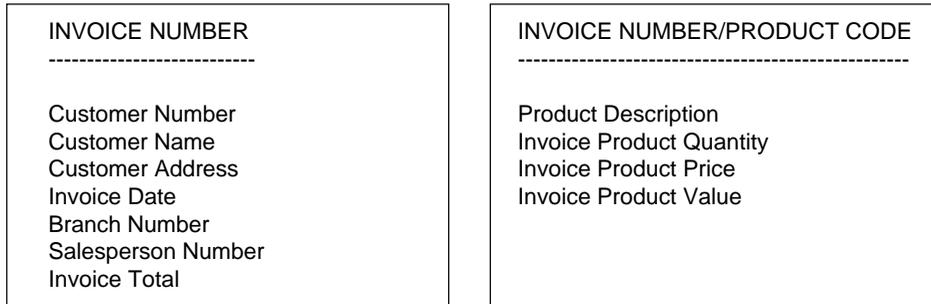
You must also identify repeating groups within the data view. The line items (product code, product description, invoice product quantity, invoice product price, and invoice product value) can repeat many times on an invoice form. To indicate this, place a bar next to that group as shown in the following figure:



Convert unnormalized form to first normal form

To convert the unnormalized relation to first normal form, remove repeating groups of attributes from the original group, setting up a new relation for each repeating group. Add the logical key of the new relation to that of the original, forming the compound key

Invoice-Number/Product-Code as shown in the following figure:

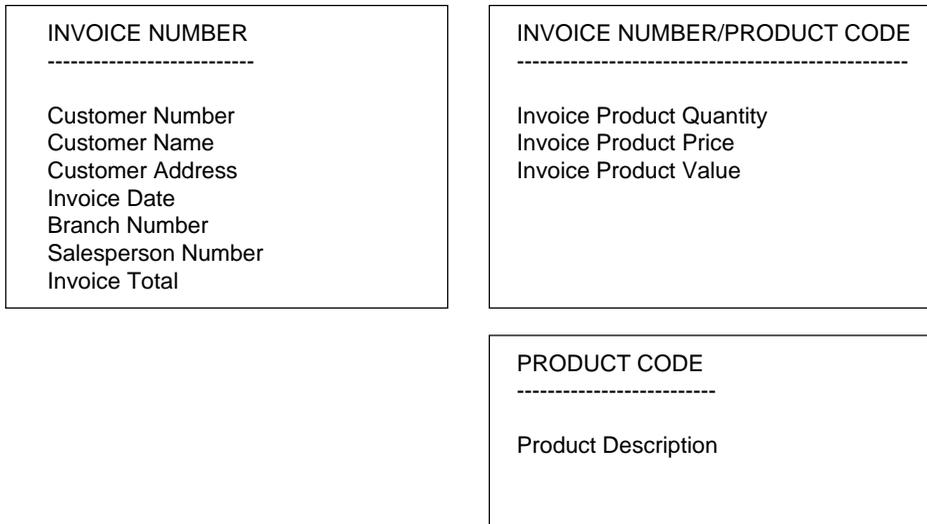


Convert first normal form to second normal form

To derive the second normal form, remove attributes that depend on only part of a compound key from a compound key relation. Generally, all attributes in a relation must depend solely on the entire key to that relation, whether the key is single or compound.

In this example, the product description depends entirely on the product code and has no association with the invoice number. Therefore, remove this attribute from the compound key relation and create a new relation, using product code as the new single key.

Invoice product quantity, price, and value remain in the compound key relation since these attributes depend on both segments of the compound key. The following figure shows second normal form:



Convert second normal form to third normal form

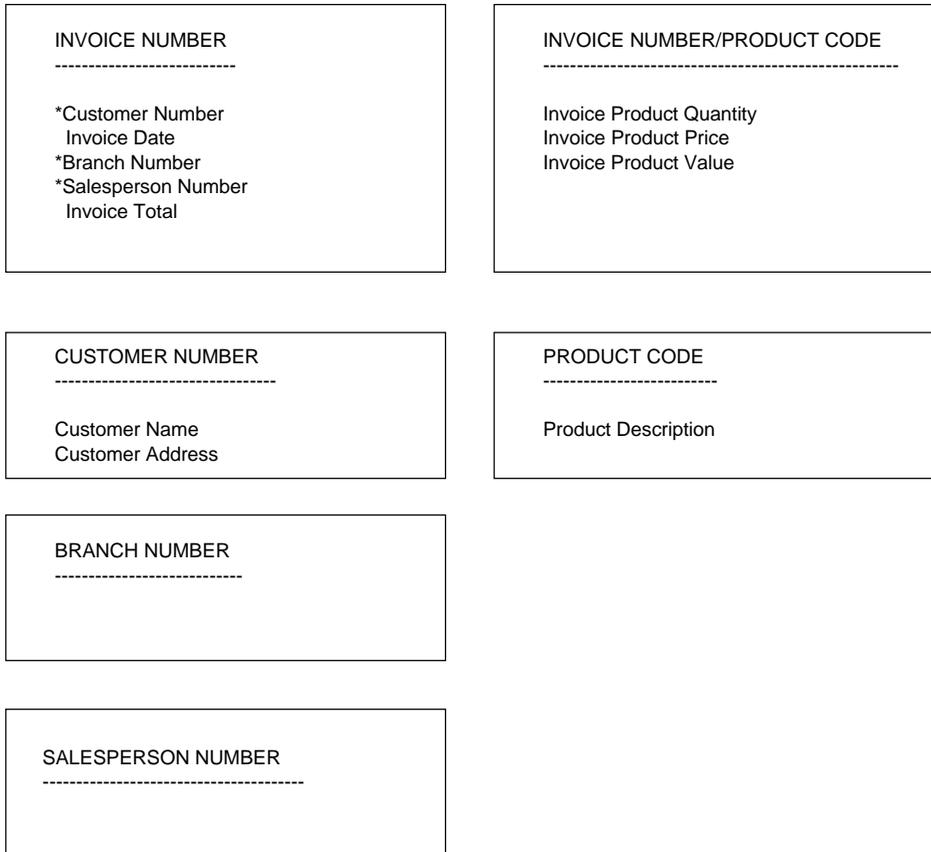
To derive the third normal form, examine relationships between attributes in a single-key relation and remove those only partially dependent on that key. An attribute that only partially depends on a key is not automatically affected by the removal of that key. For example:

- ◆ Invoice total and invoice date are fully dependent on the key invoice number; remove the invoice number and you delete both the invoice total and invoice date.
- ◆ The customer number, customer name, customer address, branch number, and salesperson number only partially depend on the invoice number key. You can remove an invoice number, but the customer, branch, and salesperson details still exist.

Therefore, to derive third normal form, set up a new relation for customer name and address with customer number as the key; likewise, set up new relations for branch number and salesperson number.

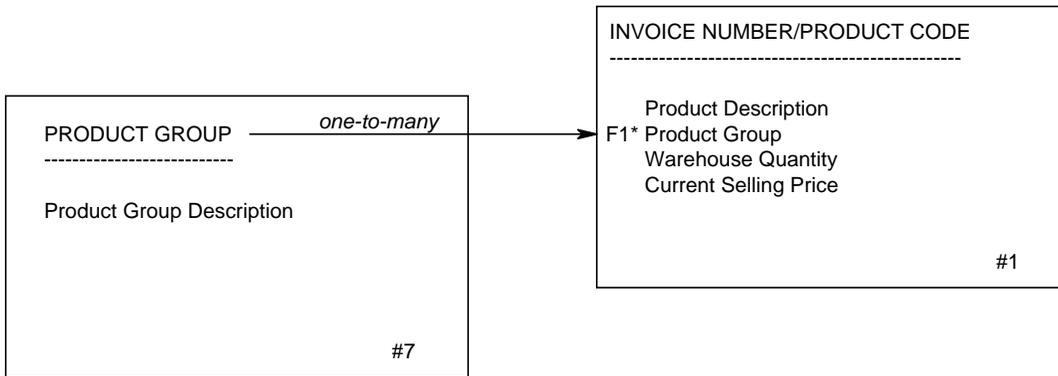
The keys to the three new relations remain in the relation keyed by invoice number. These are marked with asterisks to indicate that they are foreign keys. With a foreign key, you can access a relation without using a linkpath. Foreign keys are the only duplication of data permitted during normalization.

The following figure shows third normal form:

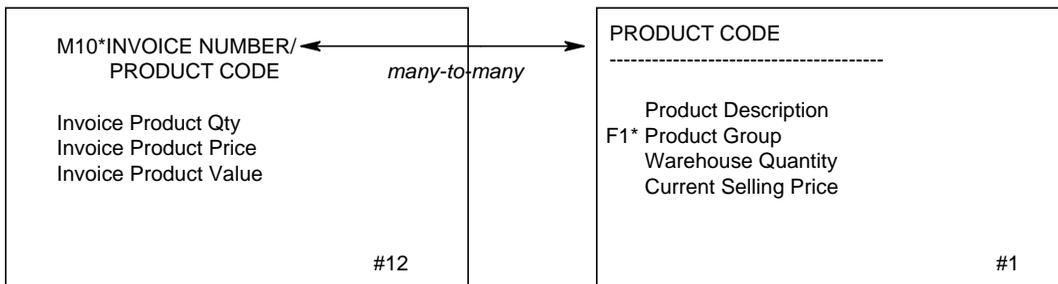


Optimize third normal form

After normalizing all other data views (customer-verification, product-data, next-order-data), some relations have the same key. The final step in normalization merges relations with identical keys. Some attributes might also occur in more than one relation. If such attributes are not used as foreign keys, you must decide to which relation they belong. The following figure shows a one-to-many relationship:

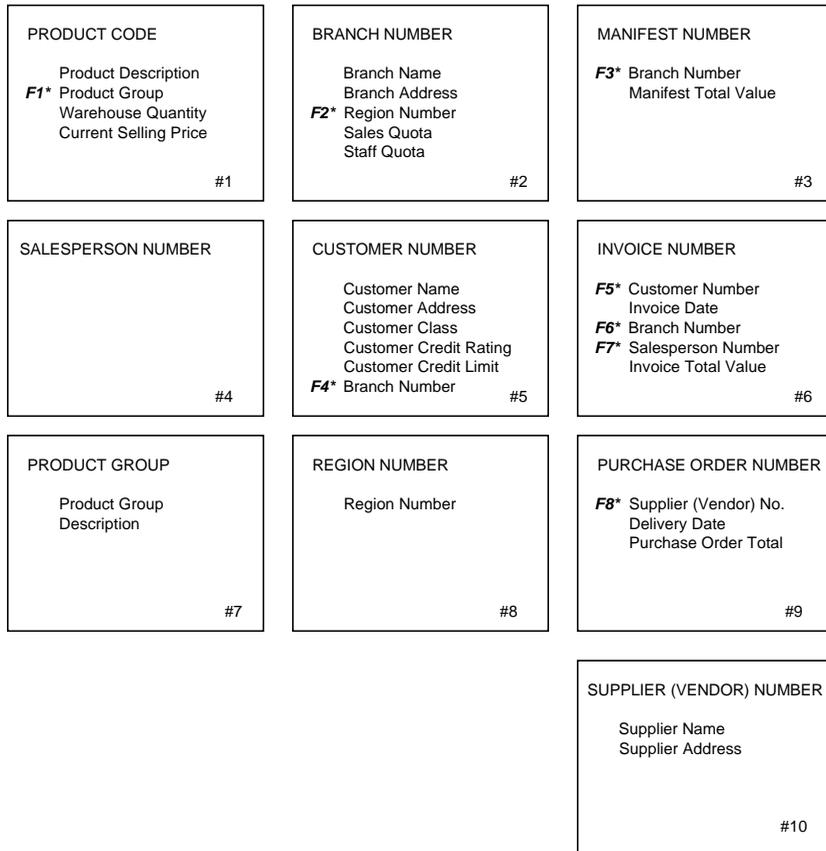


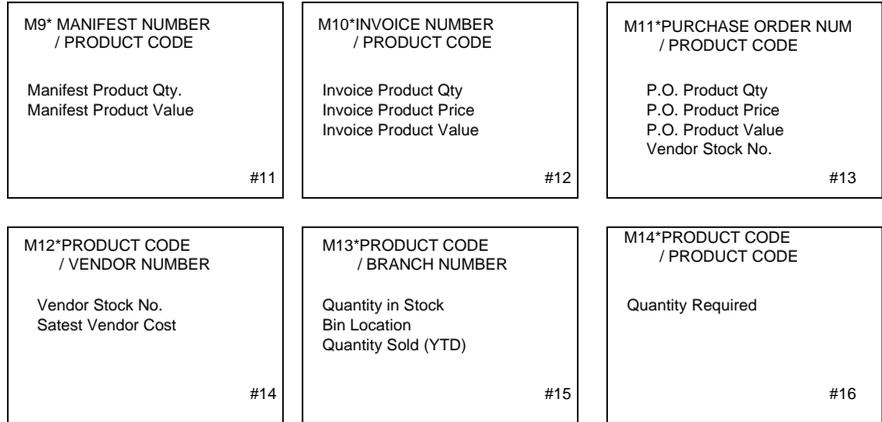
The code *Mn* represents a multiple foreign key with a many-to-many relationship. For example, the foreign key in the relation **INVOICE NUMBER/PRODUCT CODE** denotes a many-to-many relationship between **INVOICE NUMBER** and **PRODUCT CODE**. One **PRODUCT CODE** may appear on many **INVOICE NUMBERS**, and one **INVOICE NUMBER** may contain many **PRODUCT CODES**. The following figure shows a many-to-many relationship:



The following figures show the optimized single-key and compound-key relations for the Burrys example. Foreign keys are marked with an asterisk (*) and the code *F_n* or *M_n*. The code *F_n* represents a foreign key with a one-to-many relationship. For example, the foreign key in the relation PRODUCT CODE denotes a one-to-many relationship from the relation PRODUCT GROUP. Each PRODUCT GROUP may contain many PRODUCT CODES.

OPTIMIZED SINGLE-KEY RELATIONS



OPTIMIZED COMPOUND-KEY RELATIONS

Design your logical database

The third stage of database design is logical database design. The objective is to determine associations between the relations identified through normalization (see the figures in “[Optimize third normal form](#)” on page 37). Each relation represents a fundamental view of the data attributes relating to its key. Such views reflect basic relationships inherent to the data itself. These views are flexible and can be accessed efficiently.

The association between any two relations can take one of two forms: one-to-many or many-to-many. When designing a logical database, consider your data-processing requirements before you consider the physical constraints of the Physical Data Manager (PDM).

Logical database design consists of five steps:

1. Selecting relations.
2. Determining relationships.
3. Charting relationships.
4. Creating operational relationships.
5. Optimizing relationships.

Logical database design generates a schematic of the relations in the database, showing the relationships between relations. The logical design schematic models the organization’s day-to-day operations. You should maintain it to reflect changes that might affect the way the organization views and uses its data. The schematic provides a starting point for all physical database design.

Select relations

In the first step of the logical design process, use the keys identified in data analysis to represent relations. Then transfer the keys to a logical design matrix. The following figure shows the matrix with all of the single keys listed. The numbers refer to the single-key relations presented in the third figure in “Optimize third normal form” on page 37.

Data Group "A"

SUPPLIER (VENDOR) NUMBER #10										
PURCHASE ORDER NUMBER #9										
REGION NUMBER #8										
PRODUCT GROUP #7										
INVOICE NUMBER #6										
CUSTOMER NUMBER #5										
SALESPERSON NUMBER #4										
MANIFEST NUMBER #3										
BRANCH NUMBER #2										
PRODUCT CODE #1										
	#1	#2	#3	#4	#5	#6	#7	#8	#9	#10
										SUPPLIER (VENDOR) NUMBER
										PURCHASE ORDER NUMBER
										REGION NUMBER
										PRODUCT GROUP
										INVOICE NUMBER
										CUSTOMER NUMBER
										SALESPERSON NUMBER
										MANIFEST NUMBER
										BRANCH NUMBER
										PRODUCT CODE

Data Group "B"

Determine relationships

The second step of logical design uses the matrix to identify relationships between relations. In theory, a relationship can exist between any two relations. In practice, however, you select relationships on the basis of the processing requirements of the organization.

Using the Burrys example, mark all direct relationships between relations on the logical design matrix below. To process an invoice, for example, you require data about products. Use the optimized single-key relations identified in “Optimize third normal form” on page 37, marking one-to-many relationships as *F_n*, and many-to-many relationships as *M_n* on the matrix where *n* is a reference number to enable you to trace the relationship in the following figure:

Data Group "A"

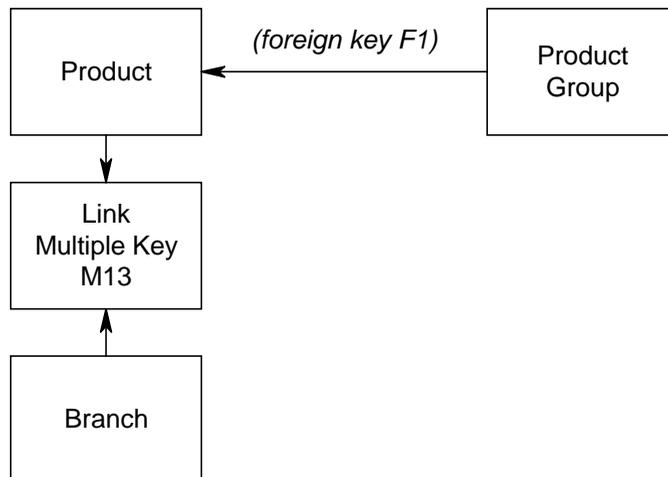
SUPPLIER (VENDOR) NUMBER #10	M12									F8
PURCHASE ORDER NUMBER #9	M11									
REGION NUMBER #8		F2								
PRODUCT GROUP #7										
INVOICE NUMBER #6	M10	F6		F7	F5					
CUSTOMER NUMBER #5		F4								
SALESPERSON NUMBER #4										
MANIFEST NUMBER #3	M9	F3								
BRANCH NUMBER #2	M13									
PRODUCT CODE #1	M14									
	#1	#2	#3	#4	#5	#6	#7	#8	#9	#10
										SUPPLIER (VENDOR) NUMBER
										PURCHASE ORDER NUMBER
										REGION NUMBER
										PRODUCT GROUP
										INVOICE NUMBER
										CUSTOMER NUMBER
										SALESPERSON NUMBER
										MANIFEST NUMBER
										BRANCH NUMBER
										PRODUCT CODE

Data Group "B"

Chart relationships

In the third step of logical design, chart relationships by representing each relation with a box and each relationship by a line between the two relations. Show the nature of the dependency (one-to-many or many-to-many) by drawing an arrow from the key relation to the detail relation. You can illustrate compound-key relations as “link” relations since they form a link between two single-key relations. If you wish, you can also note the one-to-many foreign key relationships and the many-to-many foreign key relationships as identified on the logical design matrix.

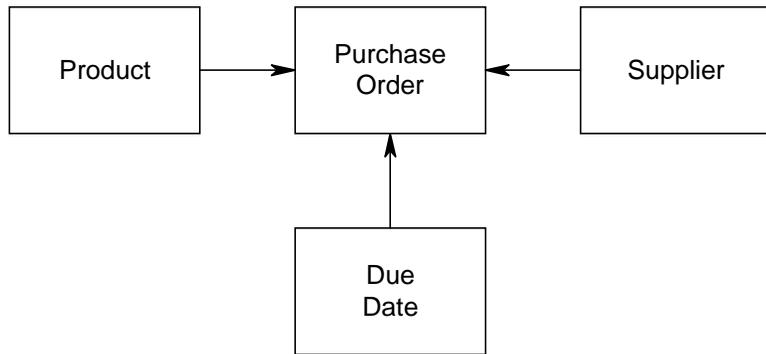
The following figure shows the relationship between product and product group, and between product and branch. An arrow points from product group to product since each product group may have many products. A compound key (product code/branch number) exists between product and branch; therefore, a link relation goes between them.



Create operational relationships

In the fourth step of the logical design process, outline the major processing cycles. In some cases, a processing requirement cannot be met efficiently with existing relations and relationships. Perhaps a requirement can only be met by scanning a whole relation to locate a particular attribute. To avoid this, remove the required attribute from the relation, creating a new relation and relationship. This becomes an “operational relationship.”

For example, assume you must display all purchase orders due on a certain date. To do this, remove the due date from the purchase order group and set up a separate relation as shown in the following figure:



Design your physical database

In the fifth stage of database design, convert the ideal logical database to an optimum physical database. Physical database design consists of three steps:

1. Converting the logical design to a valid physical database design.
2. Considering space constraints and altering the database.
3. Timing the major processing cycles and altering the database.

Before you can create the physical database design, you must understand the basic structure of a SUPRA Server database. SUPRA Server maintains primary, related, and VMS RMS data sets. Physical design determines which relations are primary, which are related, and which are RMS data sets, using these general guidelines:

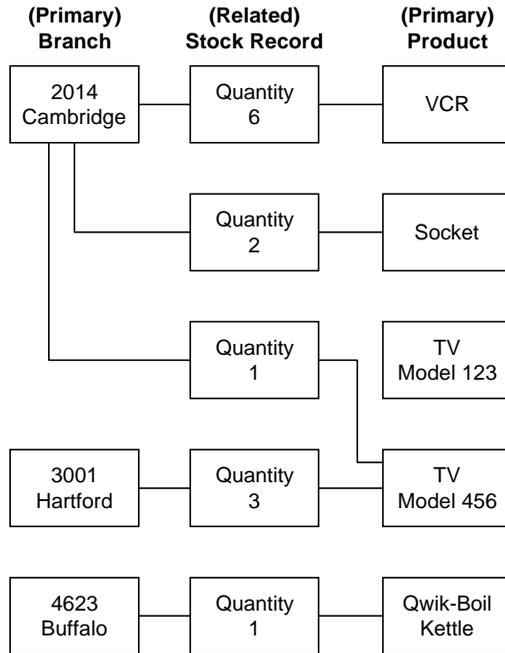
- ◆ Primary data sets are for relations with processing requirements based on keyed access to unique records.
- ◆ Related data sets are for relations with processing requirements based on keyed access to a list of records (associated with a given primary record). A record in a related data set is flexible because it can be associated with more than one list.

VMS

RMS data sets allow you to integrate data files which already exist on your system with the primary and related data sets in your SUPRA Server database. You can index RMS data sets and define foreign keys to connect them to the data sets on the SUPRA Server database.

Suppose you are concerned with the quantity of stock at particular branch offices. In the following figure, the branch information is on a primary data set accessed by a branch number. Stock details are on a related data set. Each branch can have a list of associated stock records. Furthermore, because stock details are for a given product, they are related to the appropriate product records.

You can read the branch record for Cambridge and then the stock details. You can also process data in the reverse direction, reading a product and then the stock details for that product to determine how many items are in stock and in which branch:



Primary data sets. A primary data set is identified by a 4-character name. For example, the primary data sets containing the branch and product records could be named BRCH and PROD. A unique key identifies and accesses each record in a primary data set (a primary record). All primary records in a data set are in the same format as shown in the following figure:

ROOT	KEY	Data and Linkpaths
-------------	------------	---------------------------

The root is an 8-byte field SUPRA Server uses internally. The key is a unique identifier from 1 to 255 bytes long. For example, the key to each branch record can be the branch number. Data can be the address, telephone number, and so on, for each branch. A linkpath is an 8-byte field pointing to a list of related records that can be associated with a primary record.

One primary record can be connected to any number of lists of related records and can be connected to a single related record more than once. Linkpaths in primary records only point to related records; they do not point to other primary records.

Related data sets. A related data set is also identified by a 4-character name. Records in a related data set may vary in length and hold different kinds of data. Related data set records (related records) form lists. A linkpath associates each list with one primary record.

A linkpath in a related record references the location of the associated primary record. A related record is associated with one or more physical keys, and can be in more than one list by having more than one linkpath. The number of different linkpaths in a related record determines the number of ways you can retrieve that record. When the RDM retrieves chained data, it reads the primary record first, followed by the list of related records. The PDM maintains linkpaths; the application programmer need not be concerned with linkpath maintenance.

Using flexible SUPRA Server processing, you can navigate the database by starting to read one list, changing the linkpath, and continuing to read another list in the same related data set.

Related records are identified by a relative record number (RRN). A linkpath holds a pair of RRNs; one points to the next related record and the other points to the previous related record. Whenever a related record is processed, the RRN is identified. You can save RRN values, do other unrelated processing, and then return and continue reading the original list using the saved RRN value.

Before you can create a related record, all the primary records to which it refers must exist. All related data sets share the following characteristics:

- ◆ You can define any number of physical keys and linkpaths in each related record. Therefore, each related record can be a member of several lists.
- ◆ You can associate each unique physical key with any number of related records. Each related record is connected to the previous and the next related record in the list.
- ◆ You can read related records in either forward or reverse sequence, or you can read them directly using a previously saved RRN value.
- ◆ You can give related records a number of different formats and define different relationships based on a unique 2-byte record code (see the following section “Coded record format”).
- ◆ You can create related data sets with or without a coded record format.
- ◆ Related records in a list are stored close together in sequence wherever adjacent locations are available. This makes record lists more compact, improves the performance of retrievals, and reduces physical disk transfers. One physical block can store many records.
- ◆ Multiple physical keys may be used, with each physical key linking the related record to a primary record.

Noncoded related record characteristics. With noncoded related records, all records in a data set are the same physical length and format.

Coded related record characteristics. Coded records can enhance the power and flexibility of the database. For example:

- ◆ The application programmer can perform a code-directed read which returns only those records in a list with the specified code(s)
- ◆ Coded records can be used to structure a data set hierarchically (header records, comment records, etc.)

Coded related records reside in one data set and vary in format and length. When a data set contains more than one record format, you must define the first two bytes as a record code. The PDM examines this code to determine the format of the data which follows.

Coded record format. The following figure shows how coded records are divided into two portions:

- ◆ Base data, has the same format in all records
- ◆ Redefined data, has various formats depending upon the record code

Related Record		Base Data	Redefined Data
Base Data and Redefined Data	Record Code	Common Data Items Possibly including key(s) and linkpath(s)	Redefined Data Item (one data item at level zero)
Record Code AA	AA	Exactly as above	Coded Data Items Possibly including key(s) and linkpath(s)
Coded Record			
Record Code BB	BB	Exactly as above	Coded Data Items Possibly including key(s) and linkpath(s)
Coded Record			



The areas enclosed in bold type represent the coded record as defined in a database description.

The base data portion contains a record code field and any number of physical keys, linkpaths, and data-items in any sequence. Base data and base linkpaths are in every record. Their displacements and lengths do not change.

The redefined data portion can contain a linkpath for some record codes and no linkpath for other record codes. This allows certain types of records to be selectively placed in linkpaths.

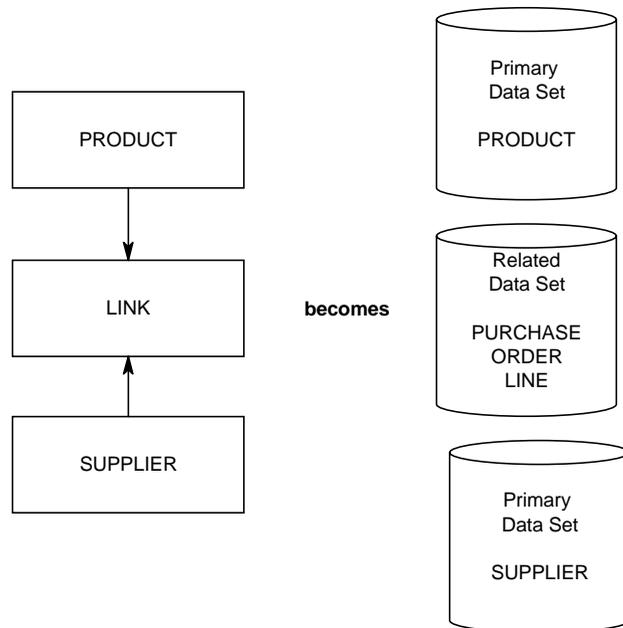
The physical key and linkpath fields in both portions of a coded record provide the same function as in a noncoded record. They associate the redefined portion of the coded record with a primary record uniquely identified by that physical key. Below are some rules for linkpaths, keys, and coded records:

- ◆ Every related record has at least one key and at least one linkpath.
- ◆ One related key is associated with every linkpath.
- ◆ If there is no linkpath in the base data, there must be one in the redefined portion of a coded record.
- ◆ If there is a linkpath in the base data portion, the related key must also be in that portion.
- ◆ If there is a linkpath in the redefined portion of a coded record, the related key must be either in the base data portion or in the redefined data portion of the same coded record.
- ◆ To run the DBA utilities on a related data set with coded records, the linkpath used should be present in every coded record.

Convert the logical design to a valid SUPRA Server database

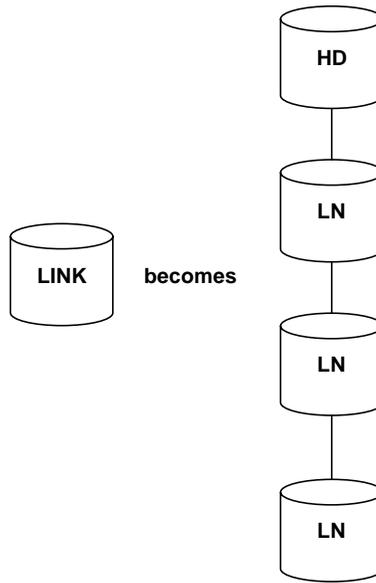
The first step of physical database design is to convert your logical design to a valid SUPRA Server database. During this initial step, examine each relation represented in the logical database design. All relations with single keys (product, supplier) become primary data sets. All relations with compound keys, including those that link two single-key relations, become related data sets.

As shown in the following figure, the relationship between product and supplier is many-to-many. Each supplier provides many products and the same product may be purchased from different suppliers. Therefore, each relation must be implemented as a primary data set linked by a related data set called Purchase Order Line. This allows access of many occurrences in one primary data set from another.

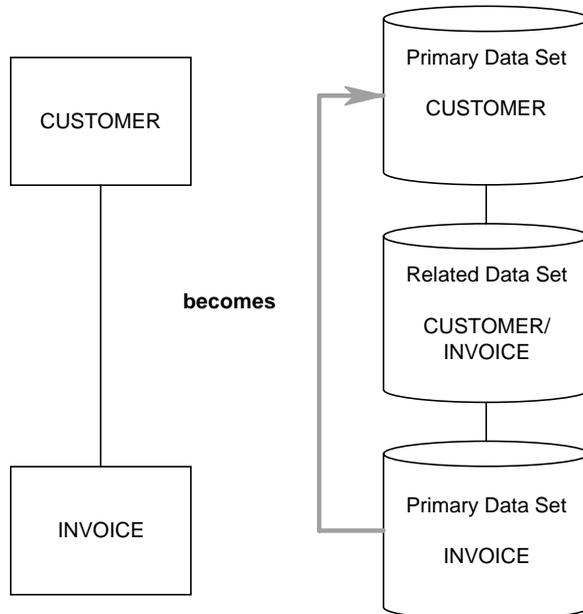


SUPRA Server can process hierarchically ordered data in a related data set with coded records. For example, a record in Purchase Order Line holds base data in the purchase order number and a linkpath. It can process data associated with the purchase order number on a one-to-one basis. It can also process sub-levels of data associated with the purchase order number on a one-to-many basis.

A header record (HD) holds supplier and date information in both record portions. Any number of line records (LN) hold information about the product, quantity, and cost in their redefined portions. As shown in the following figure, the LN coded records form a list directly associated with the HD-coded record:



When two single-key relations are linked in the logical design, you must insert a related data set between them. For example, since both customer and invoice relations become primary data sets, insert a cross-referencing related data set between them. This customer/invoice data set holds only keys and linkpaths, providing access to many invoice records for each customer record. Furthermore, given an invoice record, the associated customer record can be accessed with a foreign key, as the gray line indicates. This bypasses the cross-referencing data set as shown in the following figure:

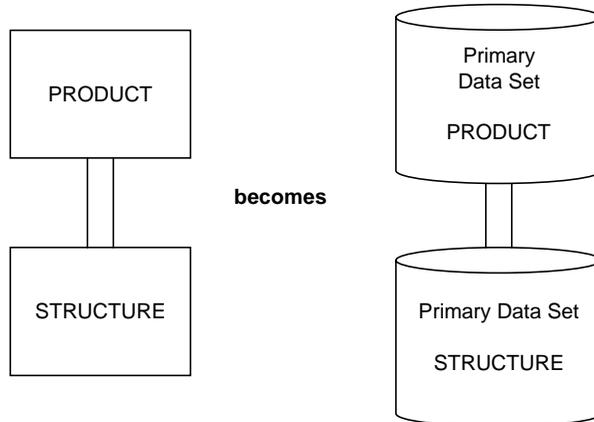


In some cases a relation can be linked to itself as shown in the following figure. This is called a double linkage or structure data set. In the case of Burrys, the product data set (primary) is linked to the structure data set (related).

The structure data set serves as a cross-reference providing data about either a component or the assembly of a component, depending on the need. This method of connecting a primary to a related data set answers two questions:

- ◆ Given a product (component), what parts (assembly) go into it?
- ◆ Given a part, in which products is it used?

Because you can navigate the double linkage in either direction, you can process data concerning either question:



Optimize space usage

The second step in the physical design process sets space objectives. First, calculate the size of each data set. For primary data sets, include the root field, linkpaths, data fields, blocking factors, and synonym rules for RRN calculations. For related data sets, include linkpaths, key field for each linkpath, data fields, record codes, growth factor for related record lists, cylinder load limit, and estimated number of records.

Calculate record lengths by adding the lengths of each data-item. Ensure that space requirements of your SUPRA Server database meet your space objectives. Use the space-saving techniques presented in the *SUPRA Server PDM System Administration Guide (VMS)*, P25-0130, or the *SUPRA Server PDM System Administration Guide (UNIX)*, P25-0132. Space constraints may differ for different physical representations of data sets. Consider the following important space-saving techniques:

- ◆ Codify large keys
- ◆ Omit unnecessary linkpaths
- ◆ Modify record codes

If a large record key is derived, consider creating a new data set where the value can be shortened. For example, using a 9-digit social security number as an employee number is an inefficient use of space if your firm has fewer than ten employees and will have for some time. A 4-digit employee number may be sufficient and requires less space for the employee data set. You can access the social security number as a nonkey field to convert the 4-digit employee number to the social security number.

If a cross-referencing related data set in the design is rarely used, consider substituting foreign keys to provide access between primary data sets. You can gain a significant amount of space and performance by not implementing compound keys and linkpaths that are seldom navigated.

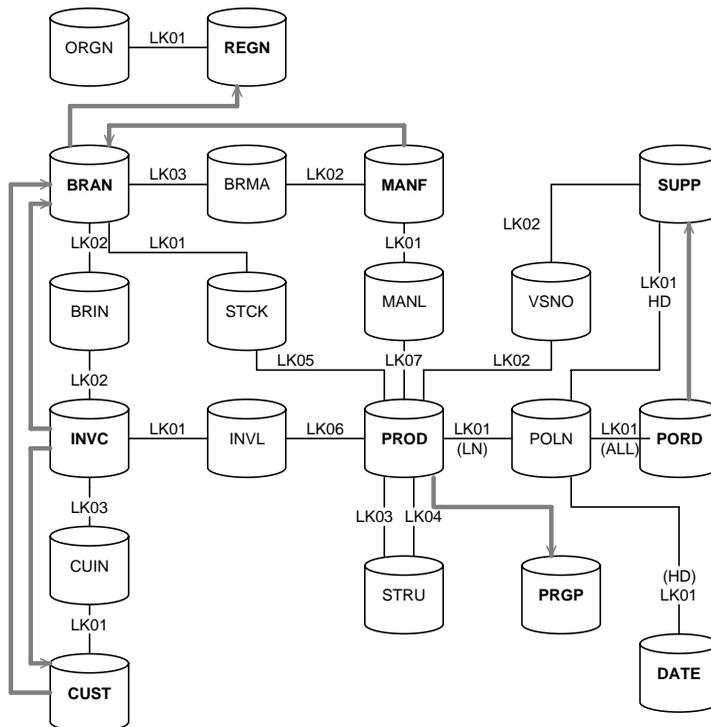
Avoid a very long redefined portion in the HD-coded record. If the redefined portion of the HD-coded record is more than twice the length of the redefined portion in the lower-level coded records, you can reformat the HD-coded record as two coded records. In this case, the first half of the redefined portion is in the first HD occurrence, and the second half is in the second occurrence. This technique avoids excessive filler in the redefined portion of the lower-level coded records.

Set timing objectives for major processing cycles

The third step in the physical design process is timing processing cycles (daily updates, online transactions, etc.). Calculate acceptable time frames for each cycle, considering disk access, program CPU time, logging time, transmission time, SUPRA Server CPU time and transaction time.

Primary data set additions cause synonym maintenance. In addition to adding records, SUPRA Server updates at least two records for each linkpath in the original record. If the time requirements do not meet your objectives, use the performance and space considerations presented in the *SUPRA Server PDM System Administration Guide (VMS)*, P25-0130, or the *SUPRA Server PDM System Administration Guide (UNIX)*, P25-0132.

You might need to repeat steps 2 and 3 of the physical design process until you achieve the optimum design. The following figure illustrates the optimum physical design for the Burrys example:



The names of the primary and related data sets correspond to the data identified in the logical design (see “[Optimize relationships](#)” on page 45). These names are:

Primary data sets	Related data sets
Region = REGN	Branch/Manifest = BRMA
Branch = BRAN	Branch/Invoice = BRIN
Invoice = INVC	Customer/Invoice = CUIN
Customer = CUST	Product/Branch (stock) = STCK
Manifest = MANF	Invoice/Product = INVL
Product = PROD	Manifest/Product = MANL
Product Group = PGRP	Product/Product = STRU
Supplier = SUPP	Product/Supplier = VSNO
Purchase Order = PORD	Product/Purchase Order = POLN

In the previous figure, a bold line represents access using a foreign key for a direct read from one primary data set to another. For example, multiple customer records from CUST can be associated with a branch record from BRAN.

Optimize your database design

In the fifth stage of database design, you optimize the logical and physical design. Optimization includes reviewing space and performance considerations and making changes to the logical and physical designs where necessary.

Optimization has complex, interrelated processes. The most efficient way is to test performance and monitor results. Your objective is maximizing system performance. The *SUPRA Server PDM System Administration Guide (VMS)*, P25-0130, or the *SUPRA Server PDM System Administration Guide (UNIX)*, P25-0132, presents considerations and guidelines for both space and performance optimization.

In practice, you optimize the logical design and implement the physical design. This process involves compromises such as removing relationships between relations when they hinder performance, or combining parts of data groups.

Consider the following factors as you work through the stages of database design:

- ◆ There is no perfect database design for a particular processing situation. A good design must compromise some features for the sake of others.
- ◆ There is no magic formula for transforming related groups of data into the perfect database design.

3

Using SUPRA DBA

SUPRA DBA is a central component of SUPRA Server. You use it to do the following:

- ◆ Define database descriptions
- ◆ Create and format the physical files
- ◆ Connect the database descriptions to the physical files
- ◆ **VMS** Define the base and derived views
- ◆ **VMS** Define and maintain user access to views
- ◆ **VMS** Control application program access to views
- ◆ Define recovery methods
- ◆ Run the DBA utilities

Overview of DBA functions

The DBA functions that allow you to perform these tasks all operate in similar ways. For example, you follow the same procedure when you delete a user name as when you delete a view or data set. Therefore, this chapter describes the following generic DBA functions:

- ◆ Using DBA function keys, PF keys and control key sequences
- ◆ Signing on to DBA
- ◆ Examining an entity
- ◆ Modifying an entity
- ◆ Creating an entity
- ◆ Deleting an entity
- ◆ Listing entities
- ◆ Connecting entities to each other
- ◆ Disconnecting entities from each other
- ◆ Entering comments
- ◆ Displaying online Help

For precise instructions on how to create and maintain databases and associated entities, see [“Creating and maintaining a database description”](#) on page 101.

DBA consists of a hierarchical series of menus and screens. To display screens below your current position in the hierarchy, to do one of the following:

- ◆ Select the appropriate option from the DBA menu displayed and press RETURN to transmit your choice.
- ◆ Respond to a DBA prompt and press RETURN to transmit your choice.

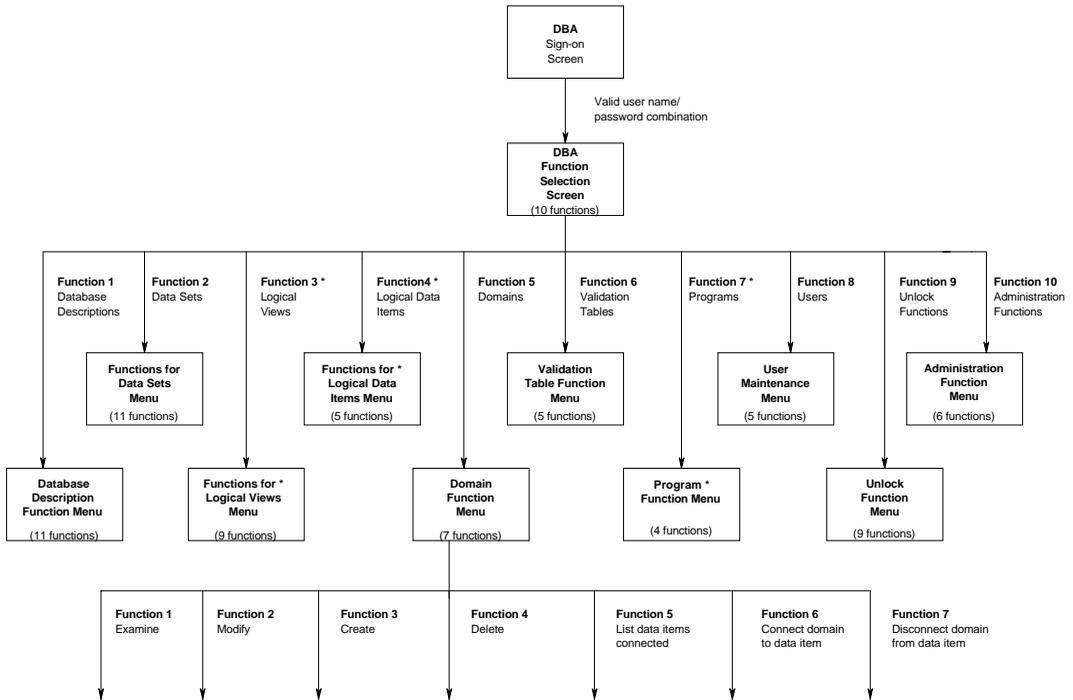
To return to the preceding screen, press function key PF1. You do not have to press RETURN to move back up the screen's hierarchy. The options that are available to you depend on the environment in which you are operating.

UNIX

Some of the options display the word *Reserved*. If you try to access these options, the system will sound and display this message:

“This function is not implemented.”

The following figure shows part of the DBA screen's hierarchy:



* VMS environments only

Using DBA function keys

After you sign on, DBA displays a main menu from which you select the function you want to perform. Whenever DBA prompts for input, you may either enter data or press one of the keys shown in the following table:

Key	Explanation
function key PF1	Terminates the current action and presents the previous prompt or screen.
RETURN or ENTER	Executes the displayed screen and presents the next appropriate prompt or screen. Press this key without entering any data to bypass the displayed screen. It usually performs like the function key PF1.
CTRL-Z VMS CTRL-Z UNIX	Immediately terminates the current action, unlocks entities and returns the main menu, Function Selection for the DBA. If used from the main menu, returns you to the command level. If used from a Help screen, returns you to DBA.
Function key PF2	Returns a Help screen containing information to help you complete the screen you are processing. If pressed after an error, presents the explanation and action for the error condition.
Function key PF3	Returns a menu or list of available choices.
Function key PF4	Processes the default value or action.
CTRL-U	Deletes the current line.
DELETE	Deletes the last character you entered.



You can only use function keys when the cursor is in the first position of a line. If you press a function key when the cursor is in any other position, DBA will not perform the function and the terminal sounds.

Once you sign on, DBA displays the main menu, Function Selection for the DBA, similar to that shown in the following screen illustration:

```
CINCOM SYSTEMS SUPRA DBA - FUNCTION SELECTION FOR THE DBA
```

```
      Select required function :
```

```

1 : Database descriptions
2 : Data sets
(VMS only) 3 : Logical views *
            4 : Logical data-items
            5 : Domains
(VMS only) 6 : Validation tables
            7 : Programs *
            8 : Users
            9 : Unlock functions
           10 : Administration functions
```

```
Enter choice no.:
```

UNIX

* The word *Reserved* displays next to these functions.

Each of the functions on the main menu, Function Selection for the DBA, gives you access to further options. The following table describes each function and lists the chapter that describes the function in detail:

Function	Purpose	Description
Database descriptions	Maintain database descriptions including associated buffer allocation, data sets, indices and secondary keys, logging files, and so on.	“Creating and maintaining a database description” on page 101
Data sets	Maintain data sets independently of any database description.	“Creating and maintaining a database description” on page 101

Function	Purpose	Description
Logical views VMS	Create, change, or delete base or derived views through EDIT/EDT interface.	“Defining logical data-items, domains, validation tables, and views ” on page 315
Logical data-items	Define the data-items contained in a view.	“Defining logical data-items, domains, validation tables, and views ” on page 315
Domains	Define a domain and connect it to or disconnect it from one or more physical data-items.	“Defining logical data-items, domains, validation tables, and views ” on page 315
Validation tables	Define a validation table and list to the domains to which it is connected.	“Defining logical data-items, domains, validation tables, and views ” on page 315
Programs VMS	Examine which views programs use and maintain program information on the Directory.	“Using the Administration functions” on page 283
Users	Define users and their level of authority.	“Using the Administration functions” on page 283
Unlock function	Make the descriptions of the following available for use on the Directory after an abnormal system close: <ul style="list-style-type: none"> ◆ Databases ◆ Data sets ◆ VMS Views ◆ Programs 	“Using the Administration functions” on page 283
Administration functions	Access the Format function, the DBA utilities, the Expand and Reset functions, Recovery, and the index facilities.	“Defining and running recovery” on page 183, “Validating, compiling, printing, and formatting a database” on page 229, “Using the Administration functions” on page 283, and “Defining Multiple Physical Databases (MPDs)” on page 309

Examining an entity

The Examine function allows you to display details about a selected entity. Because this function does not allow you to alter any details stored on the Directory, it will never set a database description status to “Needs Validation.” Always use the Examine function rather than the Modify function for displaying information.

The following table lists the menus from which you can select the Examine function together with the entity examined:

Menu title	Entity examined
Database Description Function	Database Description
Function for Data Sets	Data set
Index Maintenance Functions	Index
Index Maintenance Options	Secondary key
Logical View Function *	Logical view
Logical Data-item Function	Logical data-item
Domain Function	Domain
Validation Table Function	Validation table
Program Function *	Program
User Maintenance	User

* Available under VMS only.



You access these menus by selecting the appropriate option from the Function Selection for the DBA menu displayed after sign-on.

When you select Examine from any of the menus given in the preceding table, DBA prompts you to enter the name of the entity you wish to examine, for example, a 4-character data set name or a 6-character database name. The following screen shows the prompt that displays lower-left on screen when you select Examine from the Function for Data Sets menu:

```
CINCOM SYSTEMS      SUPRA DBA - FUNCTION FOR DATA SETS

      Function for data sets:
1 : Examine data set
2 : Modify data set
3 : Create primary data set
4 : Create related data set
(VMS only) 5 : Create RMS data set *
6 : Delete data set
7 : List all data sets
8 : Connect an existing data set
9 : Disconnect an existing data set
10 : List databases using data set

      Enter choice no.: 1

Examine data set : CUST
```



* The word *Reserved* displays next to this function.

Note that if you have already carried out a DBA function on an entity of this type (a data set, in this example), DBA will allow you to accept that choice as default with the prompt:

```
Examine data set (<PF4> will select CUST ) :
```

Press function key PF4 to accept the default instead of typing the name.

Remember that you can press function key PF3 to display a list of valid responses when DBA prompts you to specify an entity name.

Once you have selected Examine, DBA carries that function through subsequent screens to give you read-only access to connected entities. The following table lists connected entities and relationships that you can examine through an entity selection from the preceding table:

Original entity	Connected entity or relationship
Database description	Details Comments Data sets incorporating: <ul style="list-style-type: none"> ◆ Details ◆ Comments ◆ Physical file attributes ◆ Records ◆ Physical data-items ◆ Buffers used Buffers Task Log System Log
Data set	Details Comments Physical file attributes Records Physical data-items Indices and secondary keys Buffers used Connection to database description
Index	Attributes Comments Secondary keys
Secondary key	Attributes Comments Data-items connected

Original entity	Connected entity or relationship
Logical view *	View definition Connection to database description User authorization
Domain	Details Comments Connection to physical data-item Connection to validation table
Validation table	Details Comments Connection to domains
Programs *	Details Comments Views used
User	Details Comments Authorization on the Directory

* Available under VMS only.

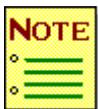
Modifying an entity

The Modify function allows you to change a selected entity. Once you have used Modify to access an entity, DBA sets the associated database description status to “Needs Validation.” Therefore, unless you are certain that you wish to modify entity details, always use the Examine function rather than the Modify function for displaying information.

The following table lists the menus from which you can select the Modify function together with the entity modified:

Menu title	Entity modified
Database Description Function	Database description
Function for Data Sets	Data set
Index Maintenance Functions	Index
Index Maintenance Options	Secondary key
Logical View Function *	Logical view
Logical Data-item Function	Logical data-item
Domain Function	Domain
Validation Table Function	Validation table
User Maintenance	User

* Available under VMS only.



You access these menus by selecting the appropriate option from the Function Selection for the DBA menu displayed after sign-on.

When you select Modify from any of the menus given in the preceding table, DBA prompts you to enter the name of the entity you wish to modify, for example, a user name or a data set name. The following screen shows the prompt displayed when you select Modify from the Database Description Function menu:

```
CINCOM SYSTEMS SUPRA DBA-DATABASE DESCRIPTION FUNCTION

      Function for database descriptions:
1 : Examine
2 : Modify
3 : Create
4 : Delete
5 : Copy
6 : List
7 : Print
8 : Validate
9 : Compile and print
(VMS only) 10 : List logical views for the database *
11 : Data set functions

      Enter choice no.: 2

Modify database description name
: TESTDB
```



* The word *Reserved* displays next to this function.



If you have already carried out a DBA function on an entity of this type (a database in this example), DBA will allow you to accept that choice as default with the prompt:

```
Modify database description name (<PF4> will select TESTDB)
:
```

Press function key PF4 to accept the default instead of typing the name.

Press function key PF3 to display a list of valid responses when DBA prompts you to specify an entity name.

Once you have selected Modify, DBA carries that function through subsequent screens to give you update access to connected entities. The following table lists connected entities and relationships that you can modify through an entity selection from the preceding table:

Original entity	Connected entity or relationship
Database description	Data sets incorporating <ul style="list-style-type: none"> ◆ Details ◆ Comments ◆ Physical file attributes ◆ Buffers used ◆ Records ◆ Physical data-items Buffers Task log System log
Data set	Details Comments Physical file attributes Buffers used Records Physical data-items Indices and secondary keys Connection to database description
Index	Attributes Comments Connection to secondary keys
Secondary key	Attributes Comments Connection to physical data-items

Original entity	Connected entity or relationship
Logical view *	View definition Connection to database description User authorization
Domain	Details Comments Connection to physical data-item Connection to validation table
Validation table	Details Comments
Program *	Comments Views used
User	Details Comments Authorization on the Directory

* Available under VMS only.

Creating an entity

The Create function allows you to create a new entity. The following table lists the menus from which you can select the Create function together with the entity created. You access these menus by selecting the appropriate option from the Function Selection for the DBA menu displayed after sign-on.

Menu title	Entity created
Database Description Function	Database description
Function for Data Sets	Data set
Index Maintenance Function	Index
Index Maintenance Options	Secondary key
Logical View Function *	Logical view
Logical Data-item Function	Logical data-item
Domain Function	Domain
Validation Table Function	Validation table
User Maintenance	User

* Available under VMS only.

When you select Create from any of the menus given in the preceding table, DBA prompts you to give a name to the entity you wish to create, for example, a domain or validation table. The following screen shows the prompt displayed when you select Create from the Function for Data Sets menu:

```
CINCOM SYSTEMS  SUPRA DBA - FUNCTION FOR DATA SETS

      Function for data sets :
1 : Examine data set
2 : Modify data set
3 : Create primary data set
4 : Create related data set
(VMS only) 5 : Create RMS data set *
6 : Delete data set
7 : List all data sets
8 : Connect an existing data set
9 : Disconnect an existing data set
10 : List databases using data set

      Enter choice no.: 3

data set:
```



* The word *Reserved* displays next to this function.



Even if you have already carried out a DBA function on an entity of this type (a validation table in this example), DBA will not offer you any default name. This is because when you create an entity, you always give it a new, unique name. No default is appropriate.

Once you have selected Create, DBA carries that function through subsequent screens so you can create connected entities and establish relationships. The following table lists connected entities you can create through an entity selection from the preceding table.

Original entity	Connected entity or relationship
Database description Comments	Details Data sets incorporating <ul style="list-style-type: none"> ◆ Details ◆ Comments ◆ Physical file attributes ◆ Buffers used ◆ Records ◆ Physical data-items Buffers Task Log System Log
Data set	Details Comments Physical file attributes Buffers used Records Physical data-items Indices and secondary keys Connection to database description
Index	Attributes Comments Secondary keys
Secondary key	Attributes Comments Connection to physical data-items

Original entity	Connected entity or relationship
Logical view *	View definition Connection to database description User authorization
Domain	Details Comments Connection to physical data-item Connection to validation table
Validation Table	Details Comments
Users	Details Comments Authorization on the Directory

* Available under VMS only.

Deleting an entity

The Delete function allows you to remove an entity from the Directory together with any relationships between the deleted entity and other entities. The following table lists the menus from which you can select the Delete function together with the entity deleted. You access these menus by selecting the appropriate option from the Function Selection for the DBA menu displayed after sign-on.

Menu title	Entity deleted
Database Description Function	Database description
Function for Data Sets	Data set
Index Maintenance Functions	Index
Index Maintenance Options	Secondary key
Logical View Function *	Base or derived view
Logical Data-item Function	Logical data-item
Domain Function	Domain
Validation Table Function	Validation table
Program Function *	Program
User Maintenance	User

* Available under VMS only.

When you select Delete from any of the menus given in the preceding table, DBA prompts you to identify the entity you wish to delete, for example, a data set or view. The following screen shows the prompts displayed when you select Delete from the Function for Data Sets menu:

```
CINCOM SYSTEMS  SUPRA DBA - FUNCTION FOR DATA SETS
                Function for data sets :
                1 : Examine data set
                2 : Modify data set
                3 : Create primary data set
                4 : Create related data set
(VMS only)    5 : Create RMS data set *
                6 : Delete data set
                7 : List all data sets
                8 : Connect an existing data set
                9 : Disconnect an existing data set
               10 : List databases using data set
                Enter choice no.: 6
Delete data set : CUST
Confirm deletion of CUST and the data-items that it owns (Y,N) : Y
                SUPRADBA working
```



* The word *Reserved* displays next to this function.

Because Delete is such a powerful function, SUPRA DBA asks for confirmation before proceeding. If you respond Y, DBA flashes the message “SUPRADBA working” on the screen followed by:

```
CUST deleted successfully <PF1>
```

to indicate that the entity has been deleted. Press function key PF1 to return to the previous function menu.



If you have already carried out a DBA function on an entity of this type (a data set in this example), DBA will allow you to accept that choice as default with the prompt:

```
Delete data set (<PF4> will select CUST )
:
```

You can then press function key PF4 to accept the default instead of typing the name. You must still respond to any confirmation prompts. You can press function key PF3 to display a list of valid responses when DBA prompts you to specify an entity name.

The Delete function removes the selected entity from the Directory and the connected entities and relationships to other entities. Consider the following:

- ◆ When you delete a data set, you delete all of the logical data-items it owns, as well as the relationship between the data set and any connected databases. This sets all connected databases to a status of “Needs Validation.”
- ◆ When you delete a database, you delete all the connected views and data sets provided they are not connected to any other database. If a view or data set is connected to more than one database, DBA displays the message:

```
(entity-name) is connected to (data-base-name) so cannot be
deleted <PF1>
```

where:

entity-name is the view or data set connected to more than one database

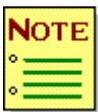
data-base-name is the database to which it is connected.

You must press function key PF1 to continue with the delete, omitting the named entity.

In addition to preventing you from accidentally deleting entities connected to more than one database, DBA prompts for confirmation before deleting data sets connected only to the selected database as follows:

```
Confirm deletion of (data-set) and the data-items that it owns
(Y,N) :
```

This additional security check allows you to delete a database description while retaining one or more of its data sets for use with other database descriptions. A database delete removes associated buffers, task logs, and system logs without prompting for confirmation.



A database delete does not delete the physical files created by the Format function.

VMS To delete these files return to the command level and use the DELETE command.

UNIX To delete these files, return to the shell and use the rm command.

The following table lists connected entities you can delete through an entity selection from the preceding table. An asterisk next to an item indicates that it is available in VMS environments only.

Original entity	Connected entity or relationship
Database description	Details Comments Data sets incorporating <ul style="list-style-type: none"> ◆ Details ◆ Comments ◆ Physical file attributes ◆ Buffers used ◆ Records Physical data-items including <ul style="list-style-type: none"> ◆ Connection to domains * ◆ Logical data-items * Buffers Task log System log Views *
Data set	Details Comments Physical file attributes Buffers used Records Physical data-items including Connection to domains * Indices and secondary keys Logical data-items Connection to database description

Original entity	Connected entity or relationship
Index	Attributes Comments Secondary keys
Secondary key	Attributes Comments Connection to physical data-item
Logical view *	Connection to database description User authorization
Domain comments	Details Comments Connection to physical data-items Connection to validation table
Validation table	Connection to domains
Programs *	Details Comments Connection to views
User	Details Comments Authorization on the Directory

* Available under VMS only.

Listing entities

The List function allows you to list all entities of the specified type. You can select an entity from the list by typing its associated number and pressing RETURN. DBA returns you to the screen from which you selected the List function and sets up the entity you selected as the default entity of that type. The following table lists the menus from which you can select the List function together with the entity listed. You access these menus by selecting the appropriate option from the Function Selection for the DBA menu displayed after sign-on.

Menu title	Entities listed
Database Description Function	All database descriptions
Function for Data Sets	All data sets
Index Maintenance Functions	All indices
Index Maintenance Options	All secondary keys
Logical View Function *	All views
User Maintenance	All users

* Available under VMS only.

When you select List from any of the menus given in the preceding table, DBA displays a list of all entities of the specified type in a standard format. The following screen shows a typical list displayed by selecting List from the User Maintenance menu:

```
CINCOM SYSTEMS  MENU for USER SELECTION

      1 : DATA-DICTIONARY
      2 : DATABASE-DESCRIPTION
      3 : PUBLIC
      4 : TEST-USR
      5 : USR-39

Enter choice number (or hit <PF2> for help):
```

If you make a choice in response to the prompt, DBA sets up your choice as the default for subsequent screens and returns you to the preceding menu. Press function key PF1 to return to the preceding menu without making any selection.

As well as the simple List function described above, DBA allows you to list entities connected to other entities. The following table shows the entities you can list in this way and the menus from which you select the List function.

Menu title	Entity connection listed
Data Set Function	Databases using data set
Data Set Options	Databases using data set
Index Maintenance Functions	Indices for data set
Index Maintenance Options	Secondary keys for index
Logical Data-item Function	Logical data-items for data set
Domain Function	Data-items connected to domain
Validation Table Function	Domains using validation table
Program Function *	Logical views used by program

* Available under VMS only.

When you select the List function from one of these menus, DBA prompts you to specify the connected entity. For example, to list the data-items connected to a data set, you must first specify the name of the data set in response to one of the following prompts:

Enter data set name :

or

Data set name :

Press function key PF3 to display a list of valid responses when DBA prompts you to specify an entity name. Once you have entered the appropriate entity name, DBA displays the list without prompting you to make a selection. Members of lists displayed in this way cannot be used as defaults.

The following screen shows a typical list displayed by selecting the option List databases using data set from the Function for Data Sets menu:

```
CINCOM SYSTEMS      -  DBDESCS USING CUST

                    1 : TESTDB
                    2 : CUSTDB

                    Hit <PF1>
```

Connecting entities to each other

The Connect function allows you to establish a relationship between one entity and another. The following table lists the menus from which you can select the Connect function together with the entities you connect. You access these menus by selecting the appropriate option from the Function Selection for the DBA menu, displayed after sign-on.

Menu title	Entities connected
Function for Data Sets	Data set to database
Index Maintenance Functions	Index to database
Logical View Function *	Logical view to database
Domain Function	Domain to data-item

* Available under VMS only.

The following screen shows the prompts displayed when you connect a data set to a database from the Function for Data Sets menu:

```
CINCOM SYSTEMS      SUPRA DBA - FUNCTION FOR DATA SETS

      Function for data sets:
1 : Examine data set
2 : Modify data set
3 : Create primary data set
4 : Create related data set
(VMS only) 5 : Create RMS data set *
6 : Delete data set
7 : List all data sets
8 : Connect an existing data set
9 : Disconnect an existing data set
10 : List databases descs using data set

      Enter choice no.: 8

Data set name (<PF4> will select CUST): <PF4>

Connect to Database description name: CUSTDB
```



* The word *Reserved* displays next to this function.



If you have already carried out a DBA function on an entity of these types (a domain or data set in this example), DBA will allow you to accept that choice as default with the following prompts:

```
Data set name (<PF4> will select CUST ):
```

You can then press function key PF4 to accept the default instead of typing the name. Remember that you can press function key PF3 to display a list of valid responses when DBA prompts you to specify an entity name.

When you connect a data set to a database, the database description is set to a status of “Needs Validation.”



You can connect any view to any database without affecting the database status.

When you connect a domain to a data-item, DBA performs validation to check that the data-item is not already connected to another domain and that the data-item and domain lengths match. If either of these conditions is not met, the attempted connection fails with an explanatory error message.

Disconnecting entities from each other

The Disconnect function allows you to remove a relationship between one entity and another. The following table lists the menus from which you can select the Disconnect function together with the entities you disconnect. You access these menus by selecting the appropriate option from the Function Selection for the DBA menu displayed after sign-on.

Menu title	Entities disconnected
Function for Data Sets	Data set from database
Index Maintenance Functions	Index from database
Logical View Function *	Logical view from database
Domain Function	Domain from data-item

* Available under VMS only.

When you select Disconnect from any of the menus given in the preceding table, DBA first prompts you to enter the name of the entity you wish to disconnect. After checking that the specified entity exists, DBA prompts you to identify the entity to disconnect it from. The following screen shows the prompts displayed when you disconnect a data set from a database description via the Function for Data Sets menu:

```

CINCOM SYSTEMS      SUPRA DBA - FUNCTION FOR DATA SETS
                    Function for data sets:
                    1 : Examine data set
                    2 : Modify data set
                    3 : Create primary data set
                    4 : Create related data set
(VMS only)         5 : Create RMS data set *
                    6 : Delete data set
                    7 : List all data sets
                    8 : Connect an existing data set
                    9 : Disconnect an existing data set
                   10 : List databases desc using data set

                    Enter choice no.: 9

                    Data set name (<PF4> wil select CUST): <PF4>

Disconnect from database desc name (<PF4> disconnects from them all) :

```



* The word *Reserved* displays next to this function.



If you have already carried out a DBA function on an entity of this type (a data set in this example), DBA will allow you to accept that choice as default with the following prompts:

```
Data set name (<PF4> will select CUST ) :
```

Press function key PF4 to accept the default instead of typing the name.

If you attempt to disconnect two entities that are not connected, DBA displays an error message and prompts you to press function key PF1 to correct the entity specification. You can press function key PF3 to display a list of valid responses when DBA prompts you to specify an entity name.

Entering comments

All of the entities you define through DBA have associated comment screens for you to enter any text data. You access the Comments screen through the Examine, Modify, and Create functions to examine or enter comments for any of the following entities:

- ◆ Database description
- ◆ Data set
- ◆ Index
- ◆ Secondary key
- ◆ Buffer
- ◆ Task log
- ◆ System log
- ◆ **VMS** Base view
- ◆ **VMS** Derived view
- ◆ Logical data-item
- ◆ Domain
- ◆ Validation table
- ◆ **VMS** Program
- ◆ User

The following screen illustration shows comments entered for a user named SENIOR. A screen of comment data is referred to as a window, and you can use multiple windows. You enter information line-by-line, up to sixteen lines of data per window. DBA allocates a sequence number to each line to add, delete, move, or display lines. When displaying a window, the current line is usually positioned near the middle of the screen. When displaying multiple windows, you can page forward or backward.

```
CINCOM SYSTEMS                               Comments for SENIOR

1 SENIOR represents the senior programmer for Development
2 The senior programmer is the only programmer with access authority
3 on this system.
4 Senior can be reached at ext 5990.
5 In case of emergency call the Database Administrator at ext 6100.

action : A,C,D,F,L,M,N,O,P,R, or W - Hit <PF2> for explanation
:
```

DBA displays prompt text and position indicators for every comment action. You select the comment action (add line, delete line, etc.) from a list of comment codes displayed at the bottom of the screen (see the following table). You enter lines of data at the bottom of the screen; DBA displays only one line at a time. When you finish adding one or more lines, you can press RETURN twice to redisplay the entire window to see the revised version.

Also, consider the following:

- ◆ When adding information, the system displays each window when you exit from the function.
- ◆ When deleting lines of data, DBA prompts for the first and last line number and displays confirmation before deleting the lines.

Code	Purpose
A	<p>Add one or more lines of data:</p> <ul style="list-style-type: none">◆ To terminate each line, press RETURN.◆ To add a blank line, press function key PF4 or Space Bar and then RETURN.◆ To terminate the add, press RETURN or function key PF1.
C	<p>Display the current window.</p>
D	<p>Delete one or more lines of data:</p> <ul style="list-style-type: none">◆ To delete multiple lines, specify the first and last line numbers and press RETURN.◆ To delete a single line, specify it as the first and last line number, or press function key PF4 as the last number.
F	<p>Insert text from an external text file. A prompt requests the name of the text file. SUPRA DBA prompts for the number of the line the new text is to follow. SUPRA DBA truncates any line longer than 72 characters.</p>
L	<p>Display the window holding a specified line number. Specify:</p> <ul style="list-style-type: none">◆ The line number and press RETURN.◆ + and a number which SUPRA DBA adds to the current line number to display the new line.◆ - and a number which SUPRA DBA subtracts from the current line number to display the new line. <p>If you specify a nonexistent line number, SUPRA DBA uses the nearest existing line.</p>

Code	Purpose
M	Move one or more lines within the comment. Specify the from and to line numbers. A prompt requests whether to delete the comment data from its original position. Enter Y or N.
N	Display the next window.
O	Output the entire comment to a text file. A prompt requests the name of the text file.
P	Display the previous window.
R	Replace one or more lines of data: <ul style="list-style-type: none"> ◆ SUPRA DBA prompts to delete the specified line and then to add the new data ◆ To delete a single line, specify it as the first and last line number, or press function key PF4 as the last number
W	Set the window size. A window can have from 1-16 lines. The default is 16. Specify: <ul style="list-style-type: none"> ◆ The line number and press RETURN ◆ + and the number by which to increase the current window ◆ - and the number by which to decrease the current window

You can retrieve comment data from an external text file and insert it on a comment screen. The text cannot exceed 72 characters per line. You can also write comment data to an external text file. DBA prompts for the file specification of the text file and for the line numbers to position the inserted text. This feature provides advanced comment editing.

Displaying online Help

Wherever you are in DBA, you can access context-related Help. The following table shows different actions you can perform in Help and which keys you use depending on the environment in which you are operating:

Function	Environment	Key
Display context-related Help	VMS and UNIX	function key PF2
Scroll through Help screens	VMS and UNIX	RETURN
Return to the screen from which you accessed Help at any time or at the prompt that signals the end of Help	VMS	CTRL-Z
	UNIX	CTRL-D

VMS

When no more Help screens are available, DBA displays the Topic? Prompt. Press RETURN or CTRL-Z to redisplay the DBA screen.

UNIX

When no more Help screens are available, DBA displays the Subject: prompt. Press CTRL-Z to redisplay the DBA screen.

4

Creating and maintaining a database description

SUPRA Server processes the DBA functions through a series of formatted screens. These screens prompt you to select options from menus and enter data. Your responses must pass certain validation checks before SUPRA DBA accepts them. If you enter an invalid response, SUPRA DBA displays a message and pauses for you to make corrections. If you are unsure how to enter the required input, you can display a series of online Help screens by pressing function key PF2. Online Help text is available at any time during SUPRA DBA processing. For more information about which keys you can use in online Help, see [“Displaying online Help”](#) on page 100.

This chapter describes the screens you use to define database descriptions and their associated data sets, indices, buffers, task and system logs. The screens displayed during database creation are the same as those displayed during database maintenance. However, during database creation, SUPRA Server displays the screens in a predefined sequence. During database maintenance, you select the screens according to the maintenance function you wish to perform. Although presenting the screens in the order they are displayed during database creation, this chapter explains how to select each screen from the preceding menu, as you would if you were maintaining an existing database. Therefore, if you are using a screen in a particular section and are unsure about how to access it, see the menu in the previous section.

After creating your database, turn to [“Defining task log recovery”](#) on page 186 and [“Defining system log recovery”](#) on page 190 for a description of how to define and run task and system log recovery. For a description of the Validate, Compile, Print, and Format functions, see [“Validating, compiling, printing, and formatting a database”](#) on page 229.

The DBA Function Selection menu provides access to the database maintenance and utility functions. Once you select a function, you proceed through the screens required to process that function. The screens displayed depend upon your responses. Enter the number of a function in the "Enter choice no." field.

```
CINCOM SYSTEMS SUPRA DBA-FUNCTION SELECTION FOR THE DBA
```

```
      Select required function :
```

```
      1 : Database descriptions
      2 : Data sets
(VMS only) 3 : Logical views *
      4 : Logical data-items
      5 : Domains
      6 : Validation tables
      7 : Programs
      8 : Users
      9 : Unlock functions
     10 : Administration functions
```

```
Enter choice no.: 1
```



* The word *Reserved* displays next to these functions.

If your access authority is Privileged or DBA/Utilities, you can select any functions available in your environment. However, if your authority is Development Personnel or Read-Only, you can access only the first seven functions in VMS environments and the first two in UNIX environments.

If you press RETURN or function key PF1 before selecting a function, the message "Do you want to exit from the DBA System? (Y,N):" displays.

Database Description functions

Use the Database Description functions to create and maintain the database definitions. When you select function 1, Database descriptions, from the Function Selection for the DBA menu, SUPRA DBA presents the Database Description Function menu shown below. This menu contains the maintenance functions. Enter the number of the desired function in the "Enter choice no." field and press RETURN.

When you select any function other than 6 or 11 (List or Data set functions), SUPRA DBA prompts for the name of the database description. The name you enter must start with a letter and consist of six alphanumeric characters with no embedded spaces. SUPRA DBA carries this name over to subsequent screens during the current session unless you enter another database name or delete the database description.

```
CINCOM SYSTEMS  SUPRA DBA-DATABASE DESCRIPTION FUNCTION

      Function for database descriptions:
1 : Examine
2 : Modify
3 : Create
4 : Delete
5 : Copy
6 : List
7 : Print
8 : Validate
9 : Compile and print
(VMS only) 10 : List logical views for the database *
11 : Data set functions

      Enter choice no.: 3

Create database description name : ORDERS
```



* The word *Reserved* displays next to these functions.

The following are the Database Description functions:

- ◆ **Examine** displays database description details such as the database password, the number of users, comments, the data sets used, the task log, and so on. Use Examine and Modify to display a Database Description Options menu.
- ◆ **Modify** changes the values for an existing database description. You cannot modify a database description if another user is already modifying, validating, printing, or compiling it. Use Examine and Modify to display a Database Description Options menu.
- ◆ **Create** adds a new database description and its details. SUPRA DBA also prompts for data sets, buffers, log files, and so on.

The Create function displays prompts, screens and menus in sequence, leading you through the steps to construct a database description and its associated entities in the following order:

1. Database details
2. Data sets
3. Buffers
4. Indices and secondary keys
5. Task log
6. System log

These screens contain defaults you can modify. Press RETURN to bypass one screen and display the next.

- ◆ **Delete** deletes a database description including details, comments, buffers, data sets, indices, VMS logical views, and task and system logs unless they are also connected to other database descriptions. You cannot delete a database description if another user is modifying, printing, or compiling it. You can delete data sets used only by this database description.

- ◆ **Copy** creates a duplicate database.

VMS

You can specify a new database description name and whether to copy (reconnect) the views.

Select Modify database description to verify that the physical file attributes are correct and to enter new task and system log specifications.

- ◆ **List** lists all available database descriptions. If you select a database from the list, that database is set as the default and you return to the previous menu. If you then select an option for which you must specify a database, you can press function key to accept the default.
- ◆ **Print** generates a listing of the database description and its details. This listing is labeled with the database name plus a .LIS (VMS) or .lis (UNIX) suffix. You can either examine the data from your terminal or route the listing to a printer. You can run Print both online and in batch.
- ◆ **Validate** checks for errors in the database description or associated data sets. You can run Validate both online and in batch.
- ◆ **Compile and Print** compiles and generates a listing of the compiled database description. You can run Compile and Print both online and in batch. SUPRA DBA creates a file called "*dbname*.MOD" (VMS) or "*dbname.mod*" (UNIX) in the current directory where the database is called "*dbname*;" however, you can change this name if you wish. SUPRA DBA must successfully validate the database description before you can execute this function.

- ◆ **VMS List logical views for the database** lists all views connected to the named database description. If you select a view from the list, that view is set as the default and you return to the previous menu. If you then select an option for which you must specify a view, you can press function key PF4 to accept the default.
- ◆ **Data Set Functions** redisplay the Function for Data Sets menu allowing you to maintain associated data sets.

You use the same screens and menus to modify or examine an existing database, selecting them in any order from the Database Description Options menu shown in “[Selecting database description options](#)” on page 108.

If you select Delete, Print, Validate, or Compile and Print, SUPRA DBA performs the requested function and then displays a confirmation message. If the action is unsuccessful, SUPRA DBA displays an error message.

Selecting database description options

SUPRA DBA presents the Database Description Options menu when you select Examine or Modify from the Database Description Function menu. The screens you access are the same for both functions, but Examine prevents any alterations and does not set the database description status to "Needs Validation." To select an option, enter its number in the "Enter choice no." field and press ENTER.

```
CINCOM SYSTEMS SUPRA DBA - DATABASE DESCRIPTION OPTIONS
```

```
Options for database description ORDERS:  
1 : Details  
2 : Comments  
3 : Data sets  
4 : Buffers  
5 : Task log  
6 : System log
```

```
Enter choice no.: 1
```

The following are the Database Description options:

- ◆ **Details** displays the database details (see “[Defining database details](#)” on page 110).
- ◆ **Comments** displays the database comments (see “[Entering database comments](#)” on page 119).
- ◆ **Data Sets** displays a menu from which you select data set details, comments, physical file attributes, the name of the associated buffer, details of any connected indices and secondary keys, record code and data-item information. You can also display a list of the data sets connected to this database (see “[Defining data sets](#)” on page 120 and “[Creating and maintaining indices and secondary keys](#)” on page 161).
- ◆ **Buffers** displays a menu from which you either display the attributes and comments for a specified buffer or list the data sets using the specified buffer (see “[Creating and maintaining buffers](#)” on page 178).
- ◆ **Task Log** displays the task log details and comments (see “[Defining a task log](#)” on page 181 and “[Defining task log recovery](#)” on page 186).
- ◆ **System Log** displays the system log details and comments (see “[Defining a system log](#)” on page 182 and “[Defining system log recovery](#)” on page 190).

Defining database details

Select option 1, Details, from the Database Description Options menu to display the Database Description Maintenance screen shown below. Use this screen to define the database details. SUPRA DBA displays existing or default values and prompts you to specify a field number. When you enter a field number and press RETURN, SUPRA DBA positions the cursor at the beginning of that field and pauses for you to either change the existing value or enter new details.



The Database Description Maintenance screen appears like this in VMS:

```
CINCOM SYSTEMS SUPRA DBA-DATABASE DESCRIPTION MAINTENANCE

 1 : DATABASE-DESCRIPTION-NAME      : ORDERS
 2 : DATABASE-PASSWORD              : CINCOM
 3 : MAX-HELD-RECORDS               : 16
 4 : MAX-TASKS                      : 10
 5 : MAX-UPDATE-TASKS               : 10
 6 : SHADOW-OPTION                  : N
 7 : SINGLE-TASK                    : N
 8 : DATABASE-DESCRIPTION-STATUS    : BEING MODIFIED
 9 : DATE-COMPILED                  :
10 : TIME-COMPILED                  :
11 : ACCESS-METHOD                 : QIO
12 : GLOBAL-SECTION-TYPE            : GROUP
13 : CALLING-MECHANISM              : REFERENCE
14 : CLUSTER/NETWORK-SUPPORT       : N
```

Enter field number to modify a field (or <PF1> to exit):

Refer to the [SUPRA Server PDM UNIX Installation Guide](#), P25-1008 or the [SUPRA Server PDM VMS Installation Guide](#), P25-0147, for information on the minimum and maximum values when designing and modifying your database.



The Database Description Maintenance screen appears like this in UNIX:

```
CINCOM SYSTEMS SUPRA DBA - DATABASE DESCRIPTION MAINTENANCE
```

```

1 : DATABASE-DESCRIPTION-NAME      : ORDERS
2 : DATABASE-PASSWORD              : CINCOM
3 : MAX-HELD-RECORDS               : 16
4 : MAX-TASKS                       : 10
5 : MAX-UPDATE-TASKS               : 10
6 : SHADOW-OPTION                  : N
7 : SINGLE-TASK                     : N
8 : DATABASE-DESCRIPTION-STATUS    : BEING MODIFIED
9 : DATE-COMPILED                  :
10 : TIME-COMPILED                  :
11 : BYTE-ORDER                     : NETWORK
12 : DATABASE-TYPE                  : GROUP
13 : BINARY-ZERO-KEY                : N
14 : NETWORK-SUPPORT                : Y

```

```
Enter field number to modify a field (or <PF1> to exit):
```

DATABASE-DESCRIPTION-NAME

Description *Display.* Displays the database named on the database menu.

DATABASE-PASSWORD

Description *Optional.* Specifies the password that protects the database against unauthorized use of the Format, Recovery, Utility, Expand, and Reset functions.

Format 1–6 alphanumeric characters

Consideration SUPRA DBA displays the password.

MAX-HELD-RECORDS

Description	<i>Optional.</i> Specifies the maximum number of records a single user can hold.
Default	16
Options	0–999

Considerations

- ◆ The total number of records that all users of a database description can hold is MAX-HELD-RECORDS times MAX-UPDATE-TASKS. Thus, you might be able to hold more records than specified in this field. Conversely, if other users are holding more than the number of records specified in this field, you might hold fewer records.
- ◆ If the SINGLE-TASK field is Y, SUPRA DBA sets this field to 0.
- ◆ Some records are held implicitly. For example, deleting the last related record in a list holds multiple records for linkpath maintenance.

MAX-TASKS

Description	<i>Optional.</i> Specifies the maximum number of tasks that may be signed on concurrently in read-only or update modes.
Default	10
Options	1–1000

Considerations

- ◆ If the SINGLE-TASK field is Y, this field is set to 1.
- ◆ This field determines the size of the task table.
- ◆ This field must be large enough to accommodate all programs, both read-only and update.
- ◆ The PDM input parameter MAXTASKS limits the total number of tasks signed on to all databases to 1000.
- ◆ **VMS** Should be at least 2 greater than MAX-UPDATE-TASKS.

MAX-UPDATE-TASKS

Description *Optional.* Specifies the number of tasks that can be concurrently signed on to a database in update mode.

Default 10

Options 0–1000

Considerations

- ◆ This value may not be greater than MAX-TASKS.
- ◆ If the SINGLE-TASK field is Y, this field is set to 1.
- ◆ This field and MAX-HELD-RECORDS determine the size of the record holding table.
- ◆ The PDM input parameter MAX-TASKS limits the total number of tasks signed on to all databases to 1000.

SHADOW-OPTION

Description *Optional.* Specifies how shadow files are used if a read or write fails on a data set or log (I/O error).

Default N

Options

- C Continue processing the file that has not failed. If an error occurs on that file as well, then the PDM returns an IOER status.
- D Deny access to the entire database. Subsequent calls receive a NOTO status.
- F Deny access to the data set or log in error. Subsequent calls receive an IOER status.
- N Do not perform shadow recording regardless of whether a shadow file is defined.
- O Notify and prompt the console operator to handle the error. Suspend all calls to the failed data set or log until the operator has entered a valid response.

SINGLE-TASK

Description	<i>Optional.</i> Indicates whether one or multiple users can use the database concurrently.
Default	N
Options	Y Only one user (single-task) N Multiple users (multitask)

Considerations

- ◆ If single-task only, record holding is bypassed and the task log, if specified, is exclusively for this task.
- ◆ If single-task only, the following fields are overridden with these values:

```
MAX-HELD-RECORDS = 0  
MAX-TASKS = 1  
MAX-UPDATE-TASKS = 1
```
- ◆ Do not specify Y with a task log. Application programs will get a QFUL status when attempting to update because there is no holding table.

DATABASE-DESCRIPTION-STATUS

Description	Displays the status of the database description.
Options	BEING COMPILED BEING MODIFIED NEEDS VALIDATION BEING PRINTED BEING VALIDATED OK (validated and compiled successfully) VALIDATED OK (validated but not compiled)

DATE-COMPILED

Description Displays the date this database was last compiled.

TIME-COMPILED

Description Displays the time this database was last compiled.

ACCESS-METHOD VMS

Description *Optional.* Specifies how the PDM will access the database files.

Default Q

Options Q DM uses QIO for local files, RMS for remote files.

R PDM uses RMS access method for all files.

Considerations

- ◆ Select Q to allow the PDM to choose the optimum access method on a file-by-file basis according to whether the file is remote or local:
 - For remote files the PDM uses RMS access. RMS access is not as fast as QIO, but it is the only access method supported by DECnet.
 - For local files the PDM uses QIO access. QIO access provides optimum performance for local files.
- ◆ The access method affects the number of copies of a buffer used in the following way:
 - QIO access uses the number of copies of a buffer specified through DBA. This is because a number of QIOs can be issued to the same file at the same time.
 - RMS access needs only one buffer; therefore, the PDM dynamically sets the number of buffers to one regardless of the value set in DBA and displays an information message. Maintaining only one buffer ensures that only one asynchronous I/O operation can be performed at one time.

BYTE-ORDER UNIX

Description *Optional.* Specifies the byte ordering of the internal binary-data fields in the database files.

Default N

Options N Internal binary data is held in Network byte order.

H Internal binary data is held in Host byte order.

V Internal binary data is held in VAX byte order.

Considerations

- ◆ Byte order has no effect on user binary data.
- ◆ Specify N if you may need to transfer the database files between many UNIX machines.
- ◆ Specify V if your database files have been transferred from or may be transferred to a VAX/VMS machine.
- ◆ Specify H if neither consideration 2 or 3 apply.

GLOBAL-SECTION-TYPE VMS

DATABASE-TYPE UNIX

Description *Required.* Defines the global sections used to access the database description module.

Default G (GROUP)

Options G (GROUP)

S (SYSTEM)

Considerations

- ◆ Group global sections allow only those processes running within a particular group to access a single copy of the database.
- ◆ System global sections allow processes running throughout the system to access a single copy of the database.

CALLING-MECHANISM VMS

Description *Required.* Specifies the method used to pass parameters to the PDM.

Default R (REFERENCE)

Options R (REFERENCE)

 D (REFERENCE/DESCRIPTOR)

Considerations

- ◆ UNIX On UNIX systems, this field is ignored. All parameters are treated as having been passed by reference.
- ◆ Option R, passing parameters by reference only, is the default, calling mechanism in COBOL.
- ◆ VMS By-reference is also the calling mechanism used by all RDM preprocessed programs whether they are written in COBOL, FORTRAN, or BASIC.
- ◆ Option D, passing parameters by reference and descriptor, passes FORTRAN and BASIC character strings by descriptor and passes all other data types by reference. CSIDAP must then determine whether a parameter is passed by reference or descriptor. Parameter passing by both reference and descriptor can lead to errors because certain by-reference values can be interpreted as string descriptors.

BINARY-ZERO-KEY UNIX

Description *Required.* Determines the empty value of a data record.

Default NO

Options YES Fills empty records with spaces.

 NO Fills empty records with nulls.

Consideration If valid key values for the file include nulls, then set this parameter to YES.

CLUSTER/NETWORK-SUPPORT VMS

- Description** *Required.* Specifies whether to run this copy of SUPRA Server in a clustered environment.
- Default** L (LOCAL)
- Options** C (CLUSTER/NETWORK) Enables cluster/network support, allowing user access to databases throughout a cluster or network.
- L (LOCAL) Disables cluster/network support, restricting database processing to the machine on which the compiled database description is first loaded.

NETWORK-SUPPORT UNIX

- Description** *Required.* Specifies whether or not this database can be accessed by a user on a remote UNIX machine.
- Default** Y
- Options** Y Allows remote access.
- N Does not allow remote access.

Entering database comments

When you select option 2, Comments, from the Database Description Options menu, SUPRA DBA returns the Comments screen. This screen displays either comments for the specified database or the message “No comments written yet.” Select the comment action from the codes displayed on the bottom of the screen (see the [table](#) on page 98 for more information on Comment Action codes). See “[Entering comments](#)” on page 96 for information on how to enter comments.

Defining data sets

You can define data sets either independently or as an integral part of a database description. Therefore, SUPRA DBA provides two Function for Data Sets menus; the one presented depends on the menu from which you select the data sets option. If you choose data sets as a database option, the data set you create is associated with that database. If you choose data sets from the Function Selection for the DBA menu, the data set you create is not associated with any particular database description.

Because both Data Set Function menus and both sets of procedures are similar, this section describes data set maintenance as a database function. If creating a database, the Function for Data Sets menu follows the database comment screen. SUPRA DBA connects any data sets you create to the database. For a primary or related data set, SUPRA DBA then prompts for the buffer used by the data set.

```

CINCOM SYSTEMS      SUPRA DBA - FUNCTION FOR DATA SETS

      Function for data sets:
1 : Examine data set
2 : Modify data set
3 : Create primary data set
4 : Create related data set
(VMS only) 5 : Create RMS data set *
6 : Delete data set
7 : List all data sets
8 : Connect an existing data set
9 : Disconnect an existing data set
10 : List databases using data set
11 : Process database descriptions

      Enter choice no.:

data set :
```



* The word *Reserved* displays next to this function.

Refer to the [SUPRA Server PDM UNIX Installation Guide](#), P25-1008 or the [SUPRA Server PDM VMS Installation Guide](#), P25-0147, for information on the minimum and maximum values when designing and modifying your data sets.

data set

- Description** *Required.* Identifies the data set to be maintained.
- Format** 4 alphanumeric characters. The first character must be alphabetic. No spaces are allowed.

Considerations

- ◆ **Examine Data Set** displays the data set's details including comments, buffer usage, indices and secondary keys, physical file attributes, records, and data-items. It uses the Data Set Details screen (see [“Defining Data Set Details”](#) on page 134), the Physical File Attributes screen (see [“Defining physical file attributes”](#) on page 123), and the Data Set Options menu (see [“Selecting data set options”](#) on page 132).
- ◆ **Modify Data Set** changes the data set's values. It cannot be used if an associated database description is being modified, validated, or compiled. It uses the Data Set Details screen (see [“Defining Data Set Details”](#) on page 134), Physical File Attributes screen (see [“Defining physical file attributes”](#) on page 123), and the Data Set Options menu (see [“Selecting data set options”](#) on page 132).
- ◆ **VMS Create Primary Data Set** adds a primary data set. It uses the Physical File Attributes screen (see [“Defining physical file attributes”](#) on page 123) and the Primary Record Function menu (see [“Defining primary records”](#) on page 136).
- ◆ **Create Related Data Set** adds a related data set. It uses the Physical File Attributes screen (see [“Defining physical file attributes”](#) on page 123) and the Related Record Function menu (see [“Defining related records”](#) on page 145).
- ◆ **VMS Create RMS Data Set** adds an RMS indexed, sequential data set. It uses the Data Set Details screen (see [“Defining Data Set Details”](#) on page 134), the Physical File Attributes screen (see [“Defining physical file attributes”](#) on page 123), and the RMS Record Function menu (see [“Defining RMS records \(VMS\)”](#) on page 152).

- ◆ **Delete Data Set** removes a data set including record codes and VMS logical and physical data-items. You cannot delete a data set connected to any database description except the active database description. This function requires a data set name.
- ◆ **List All Data Sets** lists all existing data sets. If you select a data set from the list, that data set becomes the default and you return to the previous menu. If you then select an option for which you must specify a data set, you can press function key PF4 to accept the default.
- ◆ **Connect to Database Description** (from main menu) or **Connect an Existing Data Set** connects a data set to a database description. This function requires a data set name and a database description name. Any indices connected to the data set are not connected to the database.
- ◆ **Disconnect from Database Description** (from main menu) or **Disconnect an Existing Data Set** removes the association of a data set to a database description. This function requires a data set name and a database description name.



If the data set you specify is connected only to the one database, any indices for that data set are deleted when you remove the connection between data set and database.

- ◆ **List Databases Using Data Set** lists all database descriptions connected to the specified data set.
- ◆ **Process Database Descriptions** (from main menu only) returns the Database Menu from which you can select a database function.

If you select a Delete, Connect, or Disconnect function, SUPRA DBA responds with a message indicating successful completion.

SUPRA DBA prompts for comment processing at various times during data set maintenance. Press RETURN to bypass the comment screen. See “[Entering comments](#)” on page 96 for information about entering comments.

Defining physical file attributes

SUPRA DBA presents the Physical File Attributes screen when you select option 3 or option 4 (Create primary/related data set) from the Function for Data Sets menu described in “[Defining data sets](#)” on page 120. You cannot use a data set that does not belong to a database description to create a physical data file because SUPRA DBA maintains physical file attributes only for database description/data set relationships.

However, you can connect a data set to more than one database and specify more than one physical file. For example, suppose data set REL1 is connected to two databases, ORDERS (used in production) and TESTDB (a copy of ORDERS used to test new programs). You could specify physical file attributes so ORDERS uses the production file and TESTDB uses a smaller sample data file. For additional information, see “[Defining Multiple Physical Databases \(MPDs\)](#)” on page 309.

When you select Physical File Attributes from the Data Set Options menu (see “[Selecting data set options](#)” on page 132), SUPRA DBA prompts for the database name. When creating a database, the screen follows data set details and comments.

This section shows the Physical File Attributes screen for primary and related data sets.



See “[Using RMS files \(VMS\)](#)” on page 149 for the Physical File Specification screen for RMS indexed sequential data sets.

Use the Physical File Attributes screen shown below to specify the physical disk files and corresponding shadow files to hold all or part of a data set. Items 3 and 4 do not appear for primary data sets. A data set can span up to four disk files specified from this screen. If you use shadowing, you must name shadow files for every physical file attribute named. If you respond N to the Shadow-Option of the Database Details screen, SUPRA DBA does not maintain shadow files (see “[Defining database details](#)” on page 110 under the SHADOW OPTION parameter).

```
CINCOM SYSTEMS  FILES FOR DATASET RELS AND DATABASE ORDERS

 1 : TOTAL-LOGICAL-RECORDS           : 0
 2 : LOGICAL-RECORDS-PER-BLOCK      : 10
 3 : ACCESS-MODE                     : UPDATE
 4 : CONTROL-INTERVAL                : 100
 5 : LOAD-LIMIT                      : 80
 6 : FILE-SPEC-1                    : ORDERSDB:RELS.DAT
 7 : SHADOW-FILE-SPEC-1             :
 8 : ALLOCATION-1                    :
 9 : FILE-SPEC-2                    :
10 : SHADOW-FILE-SPEC-2             :
11 : ALLOCATION-2                    :
12 : FILE-SPEC-3                    :
13 : SHADOW-FILE-SPEC-3            :
14 : ALLOCATION-3                    :
15 : FILE-SPEC-4                    :
16 : SHADOW-FILE-SPEC-4            :
17 : ALLOCATION-4                    :

Enter field number to modify a field (or <PF1> to exit):
```

TOTAL-LOGICAL-RECORDS

Description *Optional.* Specifies the number of records the data set can hold.

Options 2–2,147,483,647

Considerations

- ◆ During validation, SUPRA DBA calculates this value and rounds up for the number of blocks allocated. The value is a multiple of the Control Interval field if it is a related data set. If SUPRA DBA changes the value of this field, an appropriate message appears.
- ◆ The minimum value allowed is 2. The value includes internal SUPRA Server records (primary data sets = 1 record, related data sets = 1 record per control interval).

LOGICAL-RECORDS-PER-BLOCK

Description *Optional.* Specifies the number of records in a block.

Default 10

Options 1–32,767

Considerations

- ◆ Minimum value allowed is 1.
- ◆ Specify a large value for related data sets to reduce the number of physical I/Os if many successive actions are performed on the same linkpath.
- ◆ Specify values greater than one to improve primary data performance, particularly when the data set is quite full. The optimum value might even be 10 or more but depends on many factors.
- ◆ During validation, SUPRA DBA calculates this value. If changed by SUPRA DBA, an appropriate message appears.

ACCESS-MODE

Description *Optional.* Specifies the level of access permitted to the data file when accessed from this database.

Default U (UPDATE)

Options U (UPDATE)

 R (READONLY)

Considerations

- ◆ More than one database may access a data file provided only one access is in update mode. Use this option to specify the access allowed through this database.
- ◆ If you use a read-only dbmod to access a physical file which is being actively updated, then you may receive inconsistent results and/or database errors such as ICHN.

CONTROL-INTERVAL

Restriction	Related data sets only.
Description	<i>Optional.</i> Specifies the number of related records held in a PDM internal space-management unit.
Default	100
Options	2–2,147,483,647

Considerations

- ◆ During validation this value is automatically rounded up to a multiple of the Logical-Records-Per-Block field. Refer to the *SUPRA Server PDM System Administration Guide (VMS)*, P25-0130, or the *SUPRA Server PDM System Administration Guide (UNIX)*, P25-0132, for details of how the control interval affects performance.
- ◆ The optimum value for control interval varies according to application. However, consider the following factors when deciding the size of your control interval:
 - The number of records in an average list together with the number of lists.
 - The amount of memory you want to make available to the compiled database-description module.

You may be able to optimize record retrieval by specifying the same value for the control interval as for logical records per block if the average amount of records in a list (an average number for your particular situation) does not exceed logical records per block.

LOAD-LIMIT

Restriction	Related data sets only.
Description	<i>Optional.</i> Specifies the load capacity for a control interval as a percentage.
Default	80
Options	VMS 1–99 UNIX 0–99

Considerations

- ◆ If a control interval exceeds its load limit, it only accepts records in lists which already have been started in that control interval. New lists are started in new control intervals. This feature helps to keep lists of records in one block.
- ◆ If all control intervals are full, new lists are started in any block and a LOAD status is returned when you sign off from the database. Refer to the *SUPRA Server PDM System Administration Guide (VMS)*, P25-0130, or the *SUPRA Server PDM System Administration Guide (UNIX)*, P25-0132, for details of how the load limit affects performance.
- ◆ **UNIX** If the LOAD-LIMIT is set to 0, the behavior of the related data-set space-allocation mechanism is modified. The first record of a list is added in strict round-robin fashion from the first control interval to the last, then back to the first. Records added to existing chains follow the same rules defined for LOAD-LIMIT=1–99. This mechanism can be very effective in evenly loading data sets. Some data sets tend to get front-loaded when the standard LOAD-LIMIT of 1–99 is used.

The LOAD-LIMIT=0 option tends to spread the chains out so that the file is more evenly distributed. A LOAD status will no longer be returned to an application on sign off when a data set is approaching the FULL condition.

FILE-SPEC-1

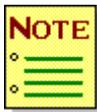
Description *Required.* Specifies the first physical disk file assigned to hold all or part of the data set.

Format 1–44 alphanumeric characters. Must represent a valid VMS file specification or UNIX pathname.

Considerations

- ◆ Specify a single file if one disk provides enough space.
- ◆ **VMS** If you are operating in a networked environment, you must specify the node name for both PDM and RMS files.
- ◆ You can include a logical name in the FILE-SPEC as follows:

```
logical-name:file-name
```



The logical name must be defined before accessing the data set. Its equivalence value must be a valid UNIX path.

SHADOW-FILE-SPEC-1

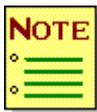
Description *Optional.* Specifies the physical disk file used to shadow File-spec-1.

Format 1–44 alphanumeric characters. Must represent a valid VMS file specification or UNIX pathname.

Considerations

- ◆ The shadow file must be on a different device from the main file.
- ◆ **VMS** If you are operating in a networked environment, you must specify the node name for both PDM and RMS files.
- ◆ You can include a logical name in the FILE-SPEC as follows:

```
logical-name:file-name.
```



The logical name must be defined before accessing the data set. Its equivalence value must be a valid UNIX path.

ALLOCATION-1

Restriction Enter only if multiple file specifications are used.

Description *Optional.* Specifies the relative size of the physical disk file named in File-Spec-1.

Default 1

Options 0–9999

Consideration If you specify more than one file, SUPRA DBA computes relative file size using the ratio of the numbers entered in the allocation field. For example, to store 100 records on two physical files, both Allocation-1 and Allocation-2 could contain the number 1 to store fifty records each (a ratio of 1:1). To store 75 records on the first file and 25 on the second (a ratio of 3:1), enter 3 for Allocation-1 and 1 for Allocation-2.

FILE-SPEC-2, 3, and 4

Description *Optional.* Specifies any remaining physical disk files assigned to hold part of the data set.

Format 1–44 alphanumeric characters. Must represent a valid VMS file specification or UNIX pathname.

Considerations

- ◆ **VMS** If you are operating in a networked environment, you must specify the node name for both PDM and RMS files.
- ◆ **UNIX** You can include a logical name in the FILE-SPEC as follows:

`logical-name:file-name.`



The logical name must be defined before accessing the data set.

SHADOW-FILE-SPEC-2, 3, and 4

Description *Optional.* Specifies the physical disk file used to shadow File-Spec-2, File-Spec-3, or File-Spec-4, if used.

Format 1–44 alphanumeric characters. Must represent a valid VMS file specification or UNIX pathname.

Considerations

- ◆ The shadow file must be on a separate device.
- ◆ **VMS** If you are operating in a networked environment, you must specify the node name for both PDM and RMS files.
- ◆ You can include a logical name in the FILE-SPEC as follows:

logical-name:file-name.

UNIX

The logical name must be defined before accessing the data set. Its equivalence value must be a valid UNIX path.

ALLOCATION-2, 3 and 4

Restriction Enter only if multiple file specifications are used.

Description *Optional.* Specifies the relative size of the physical disk file named in File-Spec-2, File-Spec-3, or File-Spec-4, if used.

Default 0

Options 0–9999

Selecting data set options

Select Examine or Modify Data Set from either the Function for Data Sets menu or the Database description Options menu to display the Data Set Options menu shown below. Use the Data Set Options menu to select a particular type of data set screen without proceeding through the entire series. You specify the data set name with the Examine or Modify function. SUPRA DBA carries the name to the Data Set Options menu. The name determines whether you are processing a primary, related, or VMS RMS data set.

To select a Data Set option, enter its number in the "Enter choice no." field and press RETURN.

```
CINCOM SYSTEMS          SUPRA DBA - DATA SET OPTIONS

      Options for data set CUST
1 : Data set details
2 : Data set comments
3 : Physical file attributes
4 : Buffer to be used
5 : Records and data-items
6 : Index Maintenance
7 : List databases using data set

      Enter choice no.:
```

- ◆ **Data set details** uses the Data Set Details screen containing the various attributes (see “[Defining Data Set Details](#)” on page 134).
- ◆ **Data set comments** uses the data set comment screens (see “[Entering database comments](#)” on page 119).
- ◆ **Physical file attributes** uses the Physical File Attributes screen containing the file and shadow file specifications and allocation (see “[Defining physical file attributes](#)” on page 123).
- ◆ **Buffer to be used** for primary and related data sets returns the name of the buffer. From Modify, you can select different buffers.
- ◆ **Records and data-items** displays the Primary, Related, VMS or RMS. Record Function menu depending on the data set type (see “[Defining primary records](#)” on page 136, “[Defining related records](#)” on page 145, or “[Defining RMS records \(VMS\)](#)” on page 152). These screens present data-item information for all types. They also include coded records for related data sets.
- ◆ **Index maintenance** displays the index maintenance menus through which you define the physical attributes of both indices and secondary-keys and the data-items included in the secondary keys. It also accesses index and secondary-key comment screens (see “[Creating and maintaining indices and secondary keys](#)” on page 161).
- ◆ **List databases using data set** displays a list of databases connected to this data set.

Defining Data Set Details

When you select option 1, Data Set Details, from the Data Set Options menu, SUPRA DBA presents a Data Set Detail screen shown below. Item 4 shown on this screen does not appear for data sets defined as primary VMS or RMS.

```
CINCOM SYSTEMS          DATA SET : REL1
1 : DATA-SET-NAME      : REL1
2 : LOGICAL-RECORD-LENGTH : 0
3 : PRIMARY-OR-RELATED-DATA-SET : RELATED
4 : CONTAINS-CODED-RECORDS : Y
5 : DATA-SET-STATUS    : BEING MODIFIED

Enter field number to modify a field (or <PF1> to exit):
```

DATA-SET-NAME

Description Displays the data set name from the menu.

LOGICAL-RECORD-LENGTH

Description Displays the logical record length of a data set as calculated during validation.

Format Format according to operating system:

VMS The maximum record length is 4096 bytes for datasets. The minimum record length is 21 bytes for primary datasets and 41 bytes for related datasets.

UNIX The maximum record length is 32,768 bytes for datasets. The minimum record length is 21 bytes for primary datasets and 41 bytes for related datasets.

PRIMARY-OR-RELATED-DATA-SET

Description Displays the data set type.

CONTAINS-CODED-RECORDS

Restriction Related data sets only.

Description Specifies if the data set contains coded records.

Options Y

N

Consideration SUPRA DBA determines this value during validation.

DATA-SET-STATUS

Description Displays the current status of the data set.

Options BEING COMPILED

BEING MODIFIED

BEING PRINTED

BEING VALIDATED

OK

Defining primary records

When creating a primary data set, SUPRA DBA displays the Primary Record Function menu (shown below) after the Physical File Attributes screen. You create and process primary data-items in a hierarchy. You can define data-items up to six levels, with the first level being the outer level.

To select Examine, Modify or List, enter the selection number in the "Enter choice no." field and press RETURN.

```
CINCOM SYSTEMS      SUPRA DBA - PRIMARY RECORD FUNCTION

      Function for data set CUST
1 : Examine primary data-items
2 : Modify primary data-items
3 : List primary data-items

      Enter choice no.:  2
```

- ◆ **Examine** lists the data-items for the specified data set in a hierarchy, the first screen containing only those data-items at the outermost level (level 0). To display subdefined data-items, enter the number of the data-item that is subdefined.
- ◆ **Modify** displays the data-items in the same way as option 1, but allows you to add, modify or delete data-items.
- ◆ **List** displays data-items with subdata-items indented. The total number of subdata-items is shown at each level break, and the items are then listed. Listed data-items are preceded by a number indicating the position of the record on the data set.

When you select option 2, Modify, SUPRA DBA displays a screen on which you can change, add, or delete data-items. You can process up to 12 data-items per screen. A number precedes each data-item and SUPRA DBA displays any data-items at the outer level (level 0) and prompts for the data-item number or a data-item command.

```
CINCOM SYSTEMS  SUPRA DBA-DATA-ITEMS IN CUST PRIMARY-DATA
```

```
    4 data-items at the outer level :
```

```
1  CUSTROOT=8
2  CUSTCTRL=20
3  CUSTLKCO=8
4  CUSTNAME   (3 subdata-items)
** End of list **
```

```
Select data-item number or function (List,Modify,Add,Delete,Copy) :
```

Either select the data-item number to perform the maintenance on that item or enter the first letter of the maintenance command you want to perform. The following table describes the maintenance commands:

Command	Code entered	Purpose
LIST	L	Lists the next 12 data-items at the current level or relists from the start.
MODIFY	M	Modifies a data-item. SUPRA DBA prompts you to define the item as a filler (Y or N), change the length (valid values = 1–4096), or add or modify comments.
ADD	A	<p>Adds a new data-item. SUPRA DBA prompts for the 4-character name followed by either the length or function key PF4. If you enter the length, SUPRA DBA prompts for the next data-item name at the same level. If you press function key PF4, SUPRA DBA prompts for the next data-item name, which SUPRA DBA defines as a subdata-item in the hierarchy.</p> <p>If adding data-items to a list, SUPRA DBA prompts for the number of the data-item the new item is to follow. Enter 0 to insert the new data-item before the first data-item on that level.</p>
DELETE	D	Deletes a data-item. SUPRA DBA prompts for the data-item number. VMS If you delete a data-item, SUPRA DBA deletes any associated logical data-items. This affects any logical views using that data-item.
COPY	C	Reserved for future use. Do not use.

If you select a data-item number rather than a maintenance command, SUPRA DBA displays the following:

- ◆ Data-item name
- ◆ Data-item level
- ◆ Type of data (filler or not)
- ◆ The number of subdata-items at the next level
- ◆ Either comments or a remark that no comments have been written
- ◆ A list of the subdata-items at the next level

If you select the Add command when no data-items exist, SUPRA DBA prompts for the first data-item's name and length. SUPRA DBA then displays the full data-item's name and prompts for the next data-item. Add as many data-items as necessary. Press RETURN to display all data-items and the level of those items. If you select the Add command when data-items already exist, the data-items are displayed. SUPRA DBA prompts for the number of the data-item which the first new data-item is to follow. It then prompts for the 4 character variable part of the first data-item name and for the data-item's length. SUPRA DBA then displays the full data-item's name and prompts for the next data-item. Continue entering as many data-items as necessary.

```
CINCOM SYSTEMS SUPRA DBA-DATA-ITEMS IN CUST PRIMARY-DATA
```

```
4 data-items at the outer level :
```

- 1 CUSTROOT=8
- 2 CUSTCTRL=20
- 3 CUSTLKCO=8
- 4 CUSTNAME (3 subdata-items)
** End of list **

```
Select data-item number or function (List,Modify,Add,Delete,Copy) : A
```

```
Add data-item following data-item number: 4
```

```
Four-character data-item name: ADDR
```

```
Data-item length (or hit <PF4> to add subdata-items) : 40
```

After you enter the length of the data-item, DBA prompts you for the 4-character name of another data-item to add. Press RETURN to get back to the data-item-maintenance prompt. Data-items marked filler are inaccessible to user programs. If a filler data-item is subdefined, those subdata-items are accessible unless they are also marked filler.

To subdefine an existing data-item, select the corresponding number. Then perform an Add function to create the subdefinitions. You can subdefine ordinary data-items and CTRL data-items, but not ROOT or linkpaths. The following two screens illustrate the subdefined, filler-data-item CUSTNAME (data-item number 4):

```
CINCOM SYSTEMS SUPRA DBA - DATA-ITEMS IN CUST PRIMARY-DATA
```

```
    5 data-items at the outer level :
```

```
1  CUSTROOT=8
2  CUSTCTRL=20
3  CUSTLKCO=8
4  CUSTNAME   (3 subdata-items)
5  CUSTADDR=40
** End of list **
```

```
Select data-item number or function (List,Modify,Add,Delete,Copy): 4
```

```
CINCOM SYSTEMS SUPRA DBA-DATA-ITEMS IN CUST PRIMARY-DATA

Current data-item : CUSTNAME      Level : 0      Filler : N

    3 subdata-items at level 1 :

    (No comments have been written for this data-item)

1 CUSTFRST=12
2 CUSTINIT=1
3 CUSTLAST=17
** End of list **

Select data-item number or function (List,Modify,Add,Delete,Copy) :
```

If you select the Modify command, SUPRA DBA prompts for the number of the data-item to be modified, and then for the modification you wish to make. Enter a filler indicator (Y or N), the length of the item, and whether you want to modify the comments (Y or N). If you select Y, the existing comments display.

```
CINCOM SYSTEMS SUPRA DBA-DATA-ITEMS IN CUST PRIMARY-DATA
```

```

    5 data-items at the outer level

    1 CUSTROOT=8
    2 CUSTCTRL=20
    3 CUSTLKCO=8
    4 CUSTNAME      (3 subdata-items)
    5 CUSTADDR=30
    ** End of list **

Select data-item number or function (List,Modify,Add,Delete,Copy) : M
Modify data-item number : 5
CUSTADDR to be considered a filler (Y,N) : N
Length of CUSTADDR : 40
Modify comments for CUSTADDR (Y,N) : N

```

After you finish modifying a data-item, SUPRA DBA prompts you for the number of another data-item to modify. Press RETURN to get back to the data-item-maintenance prompt.

If you repeatedly press RETURN, you back up the hierarchy one level at a time until you reach the level where you began adding items. Stop on any of these screens and continue adding data-items, or select another function (Copy, Delete, List, or Modify). Press function key PF1 to return to the Primary, Related VMS or RMS Function menu from which you started.

Consider the following when modifying, adding, or deleting data-items in a primary data set:

- ◆ All data-item names contain 8 characters. The first 4 characters are predefined as the data set name. SUPRA DBA prompts for the last 4 characters and for the length of each data-item.
- ◆ SUPRA DBA creates the first two data-items, ROOT and CTRL. ROOT is display only. Enter the length of CTRL to reflect the size of your control field (the physical key).
- ◆ You cannot delete ROOT and CTRL.
- ◆ Create linkpaths linking a primary data set to its associated related data set on both the primary data set and the related data set. The first 2 characters must be LK; the remaining characters can be any alphanumeric combination. DBA automatically sets the length of each linkpath to 8 characters.
- ◆ The minimum record length for a primary data set, that is, the combined length of all its data-items, is 21 bytes. Remember this when adding, deleting, or modifying the length of data-items.

Defining related records

When creating a related data set, the Related Record Function menu appears after the Physical File Attributes screen. Select Examine, Modify, or List for related data-items, coded or uncoded. You can add or delete a record code and examine or modify record code comments.

Use record codes to improve performance and allow several record types of different formats to be stored on the same physical file. A record code can be any 2 characters (except * and space) that are unique to the data set on which it is used. Each record code contains coded data-items that redefine the last data-item in the base portion of the record.

You can define up to six levels of related data-items unless you use record codes. Since record codes redefine the last data-item in the related record, you can use only five levels. For more information, see [“Design your physical database”](#) on page 46.

```
CINCOM SYSTEMS  SUPRA DBA - RELATED RECORD FUNCTION
```

```
      Function for data set CUIIN
1 : Examine related data-items
2 : Modify related data-items
3 : List related data-items
4 : Examine coded data-items
5 : Modify coded data-items
6 : List coded data-items
7 : Add a new record code
8 : Delete a record code
9 : Examine record code comments
10 : Modify record code comments
```

```
Enter choice no.: 7
```

```
Add record code : HD
```

To select a Related Record function, enter the number in the “Enter choice no.” field and press RETURN. Use functions 1,2, or 3 on the previous screen for uncoded related data-items. The procedure for defining uncoded related data-items is the same as that for defining primary data-items described in “[Defining primary records](#)” on page 136.

Use functions 4–10 on the previous screen for record codes and coded data-items. To get a current list of record codes, select any one of these functions and press function key PF3.

If you select the Add function for a record code, SUPRA DBA prompts for the 2-character name of the record code. When you enter the record code and press RETURN, SUPRA DBA prompts for comments for this record code. You can either enter comments or press RETURN to continue adding data-items.

You add data-items to a record code the same way you add data-items to a data set. “[Defining primary records](#)” on page 136 describes how to add data-items to a primary data set and what the data-item-maintenance command codes mean.

The following two screens illustrate how to add data-items to a record code in a related data set:

```
CINCOM SYSTEMS SUPRA DBA-DATA-ITEMS IN CUIIN CODED-DATA HD 15-Jun-99 16:21
```

```
No data-items defined for this record
```

```
Select data-item number or function (List,Modify,Add,Delete,Copy): A
```

```
Four-character data-item name : ITEM
```

```
Data-item length (or hit <PF4> to add subdata-items) : 10
```

```
Add data-item following data-item CUIINITEM
```

```
Four-character data-item name : DATA
```

```
Data-item length (or hit <PF4> to add subdata-items) : 55
```

```
Add data-item following data-item CUIINDATA
```

```
Four-character data-item name : RETURN
```

Press RETURN to display the data-items you have just defined.

```
CINCOM SYSTEMS SUPRA DBA-DATA-ITEMS IN CUIIN CODED-DATA HD
```

```
2 subdata-items at level 1 :
```

```
1 CUIINITEM=10
```

```
2 CUIINDATA=55
```

```
** End of list **
```

```
Select data-item number or function (List,Modify,Add,Delete,Copy) :
```

In addition to the primary-data-item considerations listed in “**Defining primary records**” on page 136, consider the following when modifying, adding, or deleting data-items in a related data set:

- ◆ If you use record codes, the first data-item must be CODE. SUPRA DBA sets the length of the CODE data-item to 2 bytes.
- ◆ When creating linkpaths, SUPRA DBA prompts for the associated primary-data-set name and the related key data-item. The first 4 characters of a linkpath data-item identify the primary data set to which the related data set is linked.
- ◆ The minimum record length for a related data set (the combined length of all of its data-items) is 41 bytes. Remember this when adding, deleting or modifying the length of data-items.

Using RMS files (VMS)

SUPRA DBA presents the RMS Physical File Specification screen when you select option 5, Create RMS data set, from the Function for Data Sets menu. Use this screen to specify the physical file for an RMS indexed, sequential data set.

```
CINCOM SYSTEMS          RMS PHYSICAL FILE ATTRIBUTES      15-Jun-99  16:01
```

```
 1 : RMS-FILE-SPEC          : ORDERS-CUST.DAT
 2 : RMS-BUCKET-SIZE        : 0
```

```
Enter field number to modify a field (or <PF1> to exit): 1
```

RMS-FILE-SPEC

Description *Required.* Specifies the disk file assigned to the data set.

Format Valid VMS file specification

RMS-BUCKET-SIZE

Description	<i>Optional.</i> Specifies the number of disk pages in each RMS bucket (block).
Default	0
Options	0 or 1–63

Considerations

- ◆ An RMS bucket is equivalent to a SUPRA Server block; the size of both is measured in disk pages. A disk page is 512 bytes. RMS bucket size, like SUPRA Server block size, affects performance.
- ◆ Selecting a value between 1 and 63 specifies the number of 512-byte disk pages in an RMS bucket. For example, a value of 1 gives a bucket size of 512 bytes, 2 gives a bucket size of 1024 bytes, 3 gives 1536 bytes, and so on. Choose your RMS bucket size according to the size of your record and the number of records you wish to hold in one RMS bucket. RMS does not allow records to cross bucket boundaries.
- ◆ The default RMS bucket size of 0 (zero) holds 2 records. Assume, for example, that your records are 1000 bytes long and that you have accepted the default RMS bucket size of 0. This gives you an actual bucket size of 2048 or 4 pages, which is enough for 2 records.
- ◆ If performance is a problem, you can try alternative bucket sizes. However, always check the record length before you change the bucket size. In the above example, increasing the bucket size from 0 to 1 reduces the capacity from 2048 bytes (4 pages) to 1024 bytes (2 pages).
- ◆ As a general rule, choose a bucket size that will hold 6 or 7 records. You can use the VMS EDIT/FDL utility to establish the optimum bucket size. Refer to Digital's *Guide to OpenVMS File Applications* for further information.

Having described the physical RMS file, you may then want to define Recovery Unit Journaling, equivalent to Task Level Recovery on PDM files. To enable Recovery Unit Journaling for your RMS files, mark them using the VMS command SET FILE/RU JOURNAL (file-spec) as follows:

```
$SET FILE/RU_JOURNAL ORDERS-CUST.DAT
```

Then, define the logical name CSI_RMS_RU_ON TRUE before invoking any RMS application that uses a journaled RMS file. If a system or an application failure occurs, all RMS files marked for RU journaling are automatically rolled back to their last successful COMMIT point without any need for user intervention. See “[Defining Recovery Unit Journaling for RMS data sets \(VMS\)](#)” on page 189 for a description of Recovery Unit Journaling.



RMS files marked for Recovery Unit Journaling are inaccessible across a network.



The SUPRA Server RMS Recovery Unit Journaling support option must be activated by Cincom before it can be implemented.

Defining RMS records (VMS)

During RMS indexed, sequential data-set creation, the RMS Record Function menu follows the RMS Physical File Specification screen. Select Examine, Modify, or List for RMS data-items. You can also examine, modify, create, and delete RMS keys. An RMS key name can be any 8-character name. To create RMS keys, you must also provide the first data-item in the key.

To select a function, enter its number in the "Enter Choice No." field and press RETURN. For the HELP facility on RMS keys, select PF one of the RMS key functions (4 through 7) and press function key PF3.

```
CINCOM SYSTEMS      SUPRA DBA - RMS RECORD FUNCTION

      Function for data set RMS1
1 : Examine RMS data-items
2 : Modify RMS data-items
3 : List RMS data-items
4 : Examine RMS key
5 : Modify RMS key
6 : Create RMS key
7 : Delete RMS key

      Enter choice no.: 6

Enter RMS key name:  CUSTNMKY

Enter name of the first data-item in the RMS key:  CTRL
```

If you select Create RMS key, SUPRA DBA prompts for the 8-character name of the RMS key. When you enter the name and press RETURN, SUPRA DBA prompts for the first data-item.

SUPRA DBA then displays the RMS key screen. You can modify the contents of this screen and press RETURN to display the logical data-item screen. When finished, press RETURN to enter comments. The RMS Record Function menu displays after you enter the comments.

```

CINCOM SYSTEMS          RMS KEY : FRD1CUSTNMKY

    The first data-item in FRD1CUSTNMKY is FRD1CTRL

1 : RMS-KEY-NAME          : FRD1CUSTNMKY
2 : RMS-KEY-NUMBER       : 0
3 : RMS-KEY-LENGTH       : 12
4 : RMS-KEY-UNIQUE       : YES
5 : RMS-KEY-CAN-BE-CHANGED : NO

Enter field number to modify a field (or <PF1> to exit):

```

RMS-KEY-NAME

Description Displays the full name of the RMS key (the 4-character data set name followed by the 8-character key name you specified).

RMS-KEY-NUMBER

Description *Optional.* Identifies the key as either a primary or an alternate key.

Default 0

Options 1–254

Considerations

- ◆ A value of 0 identifies a primary key.
- ◆ A value from 1 to 254 identifies an alternate key and indicates the key level.

RMS-KEY-LENGTH

Description *Optional.* Specify the size of the record key.

Default The length of the data-item at which the key starts.

Options 1–255

Considerations

- ◆ The length is measured in bytes.
- ◆ Although the length of the key may be larger or smaller than the length of the physical data-item it represents, it is advisable to make both the same length.

RMS-KEY-UNIQUE

Description *Optional.* Specify whether duplicate values are allowed for this key.

Default Y

Options Y

N

RMS-KEY-CAN-BE-CHANGED

Description	<i>Optional.</i> Allow modification of the key field of an existing record.
Default	N
Options	Y N

DATA-ITEM-SIGN

Description	<i>Required.</i> Specifies if this is a signed data-item.
Default	SIGNED
Options	S SIGNED U UNSIGNED

Considerations

- ◆ Must be SIGNED if the data-item type is P or F.
- ◆ RDM ignores this field if the data-item type is C.
- ◆ Enter either the single character or the complete word. SUPRA DBA displays the complete word.

DATA-ITEM-DECIMAL-PLACES

Description *Optional.* Specifies the number of decimal places. RDM uses this value in conjunction with other details (length, sign, etc.) to describe the data-item in programs.

Default 0

Format 1–5 numeric characters

Consideration The numeric value depends on the values for DATA-ITEM-TYPE and DATA-ITEM-LENGTH according to the information in the following table:

Data-item type	Data-item length	Data-item decimal places
B (Binary)	1	Up to 2 decimal places
	2	Up to 4 decimal places
	4	Up to 9 decimal places
	8	Up to 18 decimal places
C (Character)	1–4096	0
F (Floating Point)	4 or 8	0
K (Kanji)	1–4096	0
L (Leading Numeric)	1–18	Data-item length minus one
N (Numeric)	1–18	Up to 18 decimal places
P (Packed Numeric)	1–10	Maximum 19 decimal places
Z (Zoned Numeric)	1–18	Up to 18 decimal places

Press RETURN or function key PF1 when you finish modifying the RMS Key screen to display the Logical Data-Item screen.

```
CINCOM SYSTEMS          DATA-ITEM : FRD1CTRLNMKY
```

```

1 : DATA-ITEM-NAME           : FRD1CTRL
2 : DATA-ITEM-LENGTH        : 12
3 : DATA-ITEM-USE           : KEY
4 : DATA-ITEM-TYPE          : CHARACTER
5 : DATA-ITEM-SIGN          : SIGNED
6 : DATA-ITEM-DECIMAL-PLACES : 0
7 : DATA-ITEM-SUBDATA-ITEMS : 0
8 : DATA-ITEM-LEVEL         : 0

```

Enter field number to modify a field (or <PF1> to exit):

DATA-ITEM-NAME

Description Displays the physical data-item name.

DATA-ITEM-LENGTH

Description Displays the defined length of the physical data-item.

DATA-ITEM-USE

Description Displays whether this physical data-item is a code, data, key, or linkpath data-item.

DATA-ITEM-TYPE

Description	<i>Optional.</i> Specifies the type of data you wish to maintain in this data-item. RDM uses this information in conjunction with other data-item details to describe the data-item in programs.
Default	CHARACTER
Options	B or BINARY C or CHARACTER K or KANJI L or LEADING SEPARATE NUMERIC N or NUMERIC TRAILING OVERPUNCH P or PACKED DECIMAL F or FLOATING POINT Z or ZONED TRAILING NUMERIC

Considerations

- ◆ Enter either the single character or the complete word. SUPRA DBA displays the complete word.
- ◆ SUPRA DBA validates the data-item type, length, sign, and number of decimal places after you finish modifying them. If they are invalid, you must correct them.
- ◆ The data-item type affects the data-item length and sign as shown in the following table:

Type	Length	Sign	Considerations										
BINARY	1,2,4 or 8	Signed or Unsigned	The available digits vary according to length: <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>Length</th> <th>Digits Available</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>3</td> </tr> <tr> <td>2</td> <td>5</td> </tr> <tr> <td>4</td> <td>10</td> </tr> <tr> <td>8</td> <td>20</td> </tr> </tbody> </table>	Length	Digits Available	1	3	2	5	4	10	8	20
Length	Digits Available												
1	3												
2	5												
4	10												
8	20												
CHARACTER	1–4096	Unsigned											
LEADING SEPARATE NUMERIC	1–18	Signed or Unsigned	If signed, length available is 2–18. The one character required for the sign must be included in the calculation of the field length.										
NUMERIC TRAILING OVERPUNCH	1–18	Signed or Unsigned	If signed, trailing digit is overpunched with the sign.										
PACKED	1–10	Signed	Gives a maximum of 19 digits.										
FLOATING	4 or 8	Signed	Decimal places must be set to 0.										
ZONED TRAILING NUMERIC	1–18	Signed or Unsigned	If signed, the trailing digit is zoned with the sign.										

DBA validates the data-item type, length, sign and number of decimal places validated after you have finished modifying them. If they are invalid, you must correct them.

DATA-ITEM-SUBDATA-ITEMS

Description Displays the number of subdefined data-items owned by this data-item.

DATA-ITEM-LEVEL

Description Displays the level of the data-item.

General considerations

- ◆ SUPRA DBA creates the first data-item, CTRL. Enter the length of CTRL to reflect the size of your control field.
- ◆ CTRL cannot be deleted.
- ◆ Data-items cannot be ROOT or CODE.
- ◆ The first 2 characters cannot be LK since RMS data-items cannot be linkpaths.
- ◆ Data-items are organized in a hierarchy. You can make one of the following responses when data-items are displayed:
- ◆ Enter the number of a data-item to make that data-item the current item and to display its subdata-items.
- ◆ Press RETURN to back up one level in the hierarchy.
- ◆ Press function key PF1 to return the RMS Record Function menu.
- ◆ Press function key PF4 to select the next data-item at the current level and display its subdata-items.

Creating and maintaining indices and secondary keys

An index consists of one or more secondary keys. Each secondary key consists of one or more data-items from a single data set. You can create more than one index for each data set. However, make sure that you define no more than 4 or 5 secondary keys for a single index, and that the combined length of all of the data-items in an index does not exceed 251 bytes in VMS environments, or 255 bytes in UNIX environments. If you define more than 4 or 5 secondary keys in one index, you may experience a drop in performance.

Performance degradation has been noticed at some sites when more than two secondary keys are defined for an index. Some of the considerations for this are cost of time in maintenance routines and performing updates, adds, inserts, and so on. If you notice a significant drop in performance, you may need to restructure your index/indices.

You define indices through a series of DBA screens. During data set creation, DBA prompts you for the index and associated secondary key details after you have defined the data-items in the data set. During data set maintenance, you can add a new index or modify an existing index by selecting option 6, Index maintenance, from the Data Set Options menu to access the following Index menus:

- ◆ Index Maintenance Functions—To define an index and connect/disconnect it from databases and access the Index Maintenance Options menu.
- ◆ Index Maintenance Options—To specify the physical file attributes for the index, enter comments and access the Secondary Key Options menu.
- ◆ Secondary Key Options—To define the physical characteristics of each secondary key, enter comments and access the Data-item Maintenance Options menu.
- ◆ Data-item Maintenance Options—To identify data-items in each secondary key.

The index details you define are the same for both the Create and Modify functions; however, when you create a new index, DBA prompts you to enter details of secondary keys and data-items without using these menus. This section describes how to select the indexing functions from the menus, presenting the screens in the order they appear during index creation. Once you have successfully created an index through DBA, you must take two steps before the PDM will start writing index records:

1. Validate and compile the database to which the index is connected (see “[Validating, compiling, printing, and formatting a database](#)” on page 229).
2. Format and populate the physical index files either by invoking the Index Maintenance utility (CSTUIDX) within DBA (see “[Running CSTUIDX from DBA](#)” on page 265) or from the command level (see “[Running CSTUIDX from the command level](#)” on page 278), or with the POPULATE PDM operator command (refer to the *[SUPRA Server PDM System Administration Guide \(VMS\)](#)*, P25-0130). The populate function automatically activates the index.

UNIX

Index optimization consists of the selection of an index block size appropriate for the index. The index block size is calculated during the dbmod compile step. The index block size is determined by using the largest secondary key defined for the index and the total number of logical records in the data set. If possible, an index block size is selected which will produce a three-level index. Index block sizes will be in the range of 512 bytes to 16K bytes. Additional levels will be added if the block size becomes too large due to a large key length or a large number of logical records in the data set. The block size is calculated only when a new dbmod file is being produced. If the dbmod file exists at compile time, the block size is taken from the old dbmod.

Defining an index

During data set creation, DBA prompts you to define an index after you have defined the data-items in the data set. To modify an existing index, select option 6, Index maintenance, from the Data Set Options menu to display the Index Maintenance Functions menu.

```
CINCOM SYSTEMS      SUPRA DBA - DATA SET OPTIONS

Options for data set BOOK

1 : Data set details
2 : Data set comments
3 : Physical file attributes
4 : Buffer to be used
5 : Records and data-items
6 : Index maintenance
7 : List databases using this data set
Enter choice no.: 6
```

To modify an index, type 2 in the “Enter choice no.” field and enter a 2-character index name in response to the “Index name” prompt.

```
CINCOM SYSTEMS  SUPRA DBA - INDEX MAINTENANCE FUNCTIONS

      Index maintenance functions :

1 : Examine index
2 : Modify index
3 : Create index
4 : Delete index
5 : Connect index to database
6 : Disconnect index from database
7 : List indices for data set
8 : List all indices

      Enter choice no.: 2

Index name: PU
```

Refer to the [SUPRA Server PDM UNIX Installation Guide](#), P25-1008 or the [SUPRA Server PDM VMS Installation Guide](#), P25-0147, for information on the minimum and maximum values when designing and modifying your indexes.

Index name

Description *Optional.* Specifies the index name.

Format 2 alphanumeric characters. No spaces or special characters.

Considerations

- ◆ You must connect each index to at least one database description. If you remove the connection between a database and a data set with an index, you lose the index unless the data set is connected to another database.
- ◆ Press function key PF1 to exit to the Data Set Options menu without creating an index.

If the index name you enter at the Index Name prompt already exists, DBA displays the Index Maintenance Options screen. Select option 1, Attributes, to display the Index Attributes screen.

If the index name you enter at the Index Name prompt is new (if you are creating it), DBA displays the Index Attributes screen.

```
CINCOM SYSTEMS  INDEX-FILE-SP : BOOKIXPU   BOOKDB FILE-SPEC

  1 : INDEX-CORRUPT-ACTION      : OPERATOR
  2 : INDEX-NULL-SORTING        : HIGH
  3 : INDEX-READ-VERIFY         : NO
  4 : INDEX-FILE-SPEC           : DB:BOOKDB-BOOKIXPU.DAT
  5 : INDEX-SHADOW-FILE-SPEC    :
```

Enter field number to modify a field (or <PF1> to exit) :

INDEX-CORRUPT-ACTION

- Description** *Optional.* Specify what action the PDM should take if it detects a corrupt index file.
- Default**
- O Operator **VMS**
 - P Populate **UNIX**
- Options**
- O Operator. **VMS** Marks the index as unavailable and prompts the user at the operator console to either continue without using the index or to perform a dynamic populate. Continues online processing without using the index while prompting at the operator console. (Not available under UNIX.) When SYSOPCOM is set to N, and a database index has its INDEX-CORRUPT-ACTION set to O (Operator), then the PDM server will still prompt the user at the operator console.
 - C Continue. Marks the index as unavailable and continues processing without using the corrupt index file.
 - P Populate. Performs a dynamic populate on the corrupt index file. To prevent tasks from updating the database during the populate, the PDM:
 - **VMS** Signs off each task at its next COMMIT point.
 - **VMS** Unloads the database.
 - **UNIX** Locks the data set associated with the index for the duration of the populate operation. Tasks attempting to use the data set will receive a HELD status after retrying the operation the number of times specified in the RETRIES (the PDM startup parameter). (Refer to the *SUPRA Server PDM System Administration Guide (UNIX)*, P25-0132, for more information on the RETRIES parameter.)

INDEX-NULL-SORTING

- Description** *Optional.* Specify where to sort null values in the collating sequence.
- Default** H High
- Options**
- H High
 - L Low

INDEX-READ-VERIFY

Description *Optional.* Specify if the PDM is to check for a corrupt index when reading an index file.

Default Y

Options N

Y

Considerations

- ◆ If you select NO, the PDM assumes that the records in the data files match those in the index files. This option is useful if you wish to improve performance.
- ◆ If you select YES, the PDM will check that the index is not corrupt when reading records from it. Select YES unless performance is a problem.

INDEX-FILE-SPEC

Description *Required.* Specify a physical disk file assigned to hold the index.

Format A valid VMS file specification or UNIX pathname of up to 44 alphanumeric characters. May include a logical name. For example:

```
DUA3:[SUPRA.DIRECTORY]PRODINDEX.IDX VMS
/prod/datafiles/prodindex.idx UNIX
PRODINDEX
DATAFILES:PRODINDEX.IDX
```

Consideration Index files must be on a device that is local to the machine on which the PDM is executing. Remote access to index files is not supported.

INDEX-SHADOW-FILE-SPEC

Description *Optional.* Specify a physical disk file assigned to hold the shadowed index.

Format A valid VMS file specification of up to 44 alphanumeric characters. May include a logical name. For example:

```
DUA3:[SUPRA.DIRECTORY]PRODINDEX.IDX VMS  
/prod/datafiles/prodindex.idx UNIX  
PRODINDEX-SHADOW  
DATAFILES:PRODINDEX.SHA
```

Consideration Index files must be on a device that is local to the machine on which the PDM is executing. Remote access to index files is not supported.

Defining a Secondary Key

When you finish defining the physical index file's attributes, DBA redisplayes the Index Maintenance Options menu. Select option 3, 4, or 5 to display the Secondary Key name prompt as illustrated:

```
CINCOM SYSTEMS      SUPRA DBA - INDEX MAINTENANCE OPTIONS

                    Index maintenance options :

1 : Attributes
2 : Comments
3 : Create secondary key
4 : Modify secondary key
5 : Examine secondary key
6 : Delete secondary key
7 : List secondary keys for index
8 : List all secondary keys

Enter choice no.: 4

Secondary Key name: NO
```

Secondary Key name

Description *Optional.* Specifies the name of the secondary key.

Format 2 alphanumeric characters. No spaces or special characters.

Consideration **VMS** Because the PDM adds records to each index file as it adds records to the main data file, you should restrict the number of secondary keys per index to four or less. RMS displays a considerable drop in performance with more than 4 or 5 keys per index.

When you enter the 2-character Secondary Key name for the Modify or Examine Secondary Key options, DBA displays the Secondary Key Options menu. The Create Secondary Key option bypasses the Secondary Key Options menu and displays the Secondary Key Attributes screen.

Select option 1, Attributes, to display the Secondary Key Physical Attribute screen.

```
CINCOM SYSTEMS      SUPRA DBA - SECONDARY KEY OPTIONS      15-Jun-99  13:28
```

```
Secondary key options :
```

- 1 : Attributes
- 2 : Comments
- 3 : Data-items

```
Enter choice no.: 1
```

```
CINCOM SYSTEMS      SECONDARY-KEY : BOOKSKNO      15-Jun-99  13:28
```

- 1 : SECONDARY-KEY-NAME :BOOKSKNO
- 2 : SEC-KEY-UNIQUE :YES
- 3 : SEC-KEY-DIRECTION :FORWARD
- 4 : SEC-KEY-POINTER-ORDERING :NO
- 5 : SEC-KEY-POINTER-TYPE :DIRECT
- 6 : SEC-KEY-DATA-TYPE-SORTING :NO
- 7 : SEC-KEY-DUPPLICATES :5

```
Enter field number to modify a field (or <PF1> to exit) :
```

SECONDARY-KEY-NAME

Description Displays the Secondary Key name.

SEC-KEY-UNIQUE

Description *Optional.* Specifies whether the secondary key must be unique.

Default N

Option Y Secondary key unique
N Secondary key nonunique

SEC-KEY-DIRECTION

Description *Optional.* Defines the direction in which the keys are sorted in the file.

Default F

Option F Forward (ascending)
R Reverse (descending)
B Both (ascending and descending)

Considerations

- ◆ **UNIX** All indices may be traversed in ascending or descending sequence.
- ◆ **UNIX** All indices may contain keys of up to 255 bytes in length.
- ◆ **VMS** The direction in which the keys are sorted, together with pointer ordering, affects the number of data-items you can include in the secondary key, as shown in the following table:

Pointer ordering	Sec key direction	Max data-items in key
N	Forward	251
Y	Forward	250
N	Reverse	251
Y	Reverse	250
N	Both	125
Y	Both	125

SEC-KEY-POINTER-ORDERING

Description	<i>Optional.</i> Specifies whether to use pointers to ensure that records with identical keys are retrieved in the order in which they occur in the data set file.
Default	N
Options	Y Ordered N Not ordered

Considerations

- ◆ In a related data set, the pointer is the relative record number (RRN) of the record. If you set pointer ordering to Y, the RRN becomes part of the key and duplicate key values (excluding the RRN) are sorted according to their physical location in the data set file.
- ◆ In a primary data set, the pointer can be either the RRN of the record or the control key. This depends on whether the secondary key's pointer type is direct or indirect as follows:
 - In primary data sets where SEC-KEY-POINTER-TYPE=DIRECT, the RRN becomes part of the key. This means that duplicate key values (excluding the RRN) are sorted according to their physical location in the data set file. Because primary keys are hashed, this order is random.
 - In primary data sets where SEC-KEY-POINTER-TYPE=INDIRECT, the control key becomes part of the secondary key. This means that duplicate key values (excluding the control key) are sorted according to the value in the control key.
 - Note that the RRN or control key is added to the end of the key value and is, therefore, the least significant part of the key during sorting.
- ◆ Without pointer ordering, the PDM retrieves records with identical keys in the order in which they were written to the data set file unless: (1) the data set is a primary file, and records on synonym chains have been moved to free a home location for another record, (2) or the index has been checked and corruption fixed or repopulated.



Pointer ordering affects the number of data-items that you can include in the secondary key as shown in the preceding table.

SEC-KEY-POINTER-TYPE

Description *Optional.* Identifies the pointer type stored with the secondary key to ensure that records with identical keys are retrieved in the order in which they occur in the file.

Default D

Options I Indirect—Using a control key
D Direct—Using relative record number (RRN)

Considerations

- ◆ You can select I, Indirect, for primary data sets only.
- ◆ With pointer ordering set to YES, the PDM retrieves records as shown in the following table:

Pointer type	Related data set's duplicate key retrieval	Primary data set's duplicate key retrieval
Direct	The PDM gets duplicate keys in RRN order (VMS ascending or descending according to setting).	The PDM gets duplicate keys in RRN order (VMS ascending or descending according to SEC-KEY-DIRECTION setting).
Indirect	Invalid	The PDM gets duplicate keys in control key order (VMS ascending or descending according to SEC-KEY-DIRECTION setting).

SEC-KEY-DATA-TYPE-SORTING

Description *Optional.* Specifies if the secondary keys are to be sorted according to data type rather than treating all secondary keys as character strings.

Default N

Options Y Sort secondary keys by data type
N Sort secondary keys as character strings

SEC-KEY-DUPLICATES

Description *Optional.* The percentage of duplicate values you expect for this secondary key.

Default 5

Options 0–99 (up to 99%)

Considerations

- ◆ **VMS** This value represents the percentage of duplicate values expected in the secondary key. For example, if you anticipate a maximum of 100 records with 75 distinct, secondary-key values, enter the value 25 in this field. The value entered has minor influence on performance but can have considerable impact on your file size. Once your index is populated, you can use VMS commands to optimize your index based on the actual data in it.

- ◆ **UNIX** This value is not used.

Including data-items in a secondary key

When creating a new secondary key, DBA displays the Data-items for Secondary Key screen after prompting you to enter comments. When modifying or examining a secondary key, select option 3, Data-items, from the Secondary Key Options menu to specify the data-items you wish to include in the secondary key.

```
CINCOM SYSTEMS      DATA-ITEMS FOR SECONDARY KEY

Data-items connected to Secondary Key (NO)

  1. BOOKPUBL
      *** End of list ***

Add data-item after data-item BOOKPUBL
Enter 4 character data-item name:
```

Enter 4-character data-item name

Description *Required.* Specify a data-item name to include in the secondary key.

Format 4 alphanumeric character name of an existing data-item

Considerations

- ◆ **VMS** The number of data-items you can include in a secondary key depends on the size of each data-item and the number of secondary keys in the index. The maximum available record size for an index is 251 bytes. Therefore, the combined size of all the data-items in all of the secondary keys in an index must not exceed 251 bytes.
- ◆ **UNIX** All keys may be up to 255 bytes in length regardless of how many secondary keys are defined in an index.
- ◆ Press function key PF3 to list available data-items.
- ◆ DBA redisplay the Data-items for Secondary Key screen after each data-item you enter, prompting you to specify another data-item. Press function key PF1 to exit.

Entering the data-item type (UNIX)

After you enter the 4-character data-item name, DBA prompts you for the data type of the data-item. Valid data types are:

- ◆ B Binary
- ◆ C Character (default)
- ◆ F Floating Point
- ◆ K Kanji
- ◆ L Leading separate signed decimal
- ◆ N Unsigned decimal
- ◆ P Packed decimal
- ◆ T Trailing separate signed decimal
- ◆ Z Zoned overpunched signed decimal

If you set SEC-KEY-DATA-TYPE-SORTING=YES on the Secondary Key Physical Attribute screen, it is important to specify the proper data type of each element of the secondary key to maintain the proper sequence of keys in the index.

Creating and maintaining buffers

The Buffer Function menu shown below displays after the Function for Data Sets menu when adding a database description. Alternatively, you can select the Buffer option from the Database Description Options menu. Use the Buffer Function menu to add, change, examine (display), or delete a buffer. You can also display a list of data sets using a buffer. To select a buffer function, enter the number in the “Enter choice no” field and press RETURN.

If you created data sets from the Data Set menu, then you have created the buffer for each data set and you can bypass this screen. If you want to change the buffers allocated to existing data sets, use the Data Sets option from the Function Selection for the DBA menu to modify the data sets so they can use the new buffers.

Select Examine to display the details and any entered comments. The screen prompts for the buffer name. The prompt varies with the function selected. The name entered displays on subsequent buffer screens. The following screen illustrates this by showing the prompt displayed when you select Function 3, Add primary buffer.

```
CINCOM SYSTEMS      SUPRA DBA - BUFFER FUNCTION

      Function for buffers:
1 : Examine buffer
2 : Modify buffer
3 : Add primary buffer
4 : Add related buffer
5 : Delete buffer
6 : List data sets using buffer

      Enter choice no.: 3

Add primary buffer name:
```

Add primary buffer name

Description Specifies the name of the buffer you want to add.

Format 4 alphanumeric characters. The first character must be alphabetic.

Consideration Multiple data sets of the same type can share a buffer. However, primary and related data sets may not share the same buffer. Buffers are unique to a database.

Select Add, Examine, or Modify from the Buffer Function menu to display the Buffer screen. SUPRA DBA gets the buffer name from the menu. When adding a buffer, the screen displays the default values. The only value you can change is the number of copies. After processing the buffer's attributes, a buffer comment screen appears:

```
CINCOM SYSTEMS          BUFFER : ORDERS PRIB
1 : BUFFER-NAME          : PRIB
2 : NUMBER-OF-COPIES-OF-BUFFER : 5
3 : PRIMARY-OR-RELATED-BUFFER  : PRIMARY
4 : BUFFER-SIZE          : 0
```

Enter field number to modify a field (or <PF1> to exit):

BUFFER-NAME

Description Displays the 4-character buffer name specified in response to the preceding prompt.

NUMBER-OF-COPIES-OF-BUFFER

Description *Optional.* Specifies how many copies of this I/O buffer are allocated for use by data sets.

Default 5

Options 1–32,767

Consideration To improve performance, we recommend that you use multiple copies of private buffers. In multitask environments, many tasks compete for a buffer. All buffers are allocated at database load time. The PDM will use the least recently used buffer copy to read in data from a disk.

Refer to the *SUPRA Server PDM System Administration Guide (UNIX)*, P25-0132, or the *SUPRA Server PDM System Administration Guide (VMS)*, P25-0130, for more information on database tuning with buffers.

PRIMARY-OR-RELATED-BUFFER

Description Displays whether this buffer is used by primary or related data sets.

Consideration SUPRA DBA enters this value automatically when you select either the Add primary buffer or Add related buffer function from the Buffer Function menu.

BUFFER-SIZE

Description Displays the size of the buffers in this pool.

Consideration SUPRA DBA calculates this value when the database is validated.

Defining a task log

When creating a database, SUPRA DBA prompts with “There is no TASK-LOG Do you want to create one (Y,N):” after buffer definition. If you respond Y, SUPRA DBA presents the Task Log screen shown below.

To create or modify the task log of an existing database, select option 5, Task log, from the Database Description Options screen. If the database has a task log, SUPRA DBA displays the Action for Task Log menu. The options on this screen are Examine, Modify, and Delete.

After you enter your choice and press RETURN, SUPRA DBA displays the Task Log screen shown below. To change a default value or to modify any of the attributes, enter the appropriate field number after the prompt. After processing the task log’s details, SUPRA DBA presents a comment screen for the task log. See “[Defining task log recovery](#)” on page 186 for more information on how to define task logging for your database.

```

CINCOM SYSTEMS                TASK LOG : ORDERS TASK-LOG

 1 : TASK-LOG-BLOCK-SIZE      : 1
 2 : TASK-LOG-NO-OF-BLOCKS   : 200
 3 : TASK-LOG-FILE-SPEC      :
 4 : TASK-LOG-SHADOW-FILE-SPEC :

Enter field number to modify a field (or <PF1> to exit):

```

Defining a system log

When creating a database, SUPRA DBA prompts with “There is no SYSTEM-LOG Do you want to create one (Y,N):” after task-log definition. If you respond Y, SUPRA DBA presents the System Log screen.

To create or modify the system log of an existing database, select option 6, System log, from the Database Description Options screen. If the database has a system log, SUPRA DBA displays the Action for System Log menu. The options on this screen are Examine, Modify, and Delete.

After you enter your choice and press RETURN, SUPRA DBA displays the System Log screen. To change a default value or to modify any of the attributes, enter the appropriate field number after the prompt. After processing the system log details, SUPRA DBA presents a comment screen for the system log. See “[Defining system log recovery](#)” on page 190 for more information on how to define system logging for your database.

```
CINCOM SYSTEMS          SYSTEM LOG : ORDERS SYSTEM-LOG

1 : SYSTEM-LOG-BLOCK-SIZE      : 1
2 : SYSTEM-LOG-NO-OF-BLOCKS   : 500
3 : FILE-1-FILE-SPEC          :
4 : FILE-2-FILE-SPEC          :
5 : FILE-1-SHADOW-FILE-SPEC   :
6 : FILE-2-SHADOW-FILE-SPEC   :

Enter field number to modify a field (or <PF1> to exit):
```

5

Defining and running recovery

You define the recovery methods and log files for each database through SUPRA DBA. Then, if the system or a task fails, the Physical Data Manager (PDM) uses the methods you specified to recover any lost or damaged data from the recovery log files.

You can specify recovery when creating a new database and when maintaining an existing one. If you are creating a new database, SUPRA DBA prompts you to identify the recovery methods and log files after you specify the buffer details. During database maintenance you must select the appropriate Recovery Function menu from the Database Description Options menu. For additional information on the Database Description menus used for both database creation and database maintenance, see [“Creating and maintaining a database description”](#) on page 101.

SUPRA Server offers three recovery techniques: task logging, system logging, and shadow recording. You may use all three techniques to ensure absolute data integrity, or you may select only one or two techniques. Unless the database is read-only (no database updates are applied), you should specify at least task logging.

Task logging. Use task logging (or task-level recovery) in case of a task or system failure. It ensures the integrity of the database on a task-by-task basis by using commit points. These points allow database updates to be grouped into logical transactions which are either completely applied or not applied at all.

After a program failure, the PDM automatically resets the database to conditions existing at the last successful commit point for a task. The updates for a single-task failure are reversed without affecting any other application program. The PDM does the same for all tasks after a system failure when the first application uses the database.

The recovery is transparent to the end user and programmer and does not require special utilities. As well as being especially useful for online systems, task-level recovery is recommended for batch programs.



Task logging is not available for RMS data sets. Use VMS Recovery Unit Journaling to recover RMS data sets to the last successful commit point. Recovery Unit Journaling is also transparent to the end user.

System logging. Use system logging to recover from a device failure. This technique uses a log file containing all after-images and control functions and a task log containing before-images. If you use system logging, you must use it in connection with task logging. In addition, you must make sure that you regularly dump the system log files and take regular backups. See “[Creating a recovery point](#)” on page 185 for more information on creating a recovery point.

If all or part of the database is physically lost after a failure, you use the recovery facility to apply the after-images held on the system log to the backup copy of the database until the crash point is reached. This technique restores the database to its state just before the failure occurred. Task-level recovery then applies the before-images from the task log, thereby resetting each task that was active at the time of the crash back to its last commit point. This maintains the logical integrity of the data files.

Shadow recording. Use shadow recording as an alternative to system logging for device-failure recovery. Shadow recording duplicates data sets in real-time. For every write operation to the main data set in the database, the PDM issues an asynchronous writer operation to its shadow data set. For additional security, you can shadow the task log and system logs. After a failure, the PDM switches to the data sets still accessible, which eliminates time-consuming backups and downtime.

Although ensuring speedier recovery than system logging, shadow recording increases the time needed for each database update. It is, therefore, inappropriate for use with databases where response time is important or where most database accesses are writes.

Creating a recovery point

A recovery point is a point from which a database may be recovered after changes have been made or after a system or device failure. Situations that require a recovery point include:

** These require a recovery point for both SUPRAD and the user database.*

- ◆ Changing database metadata; for example, changing file size, adding data-items, changing lengths of data-items, deleting data-items, and resetting data set load limits.*
- ◆ Compiling a database description, such as SUPRAD.MOD or MRPDDBO.MOD.
- ◆ Moving your compiled database description to a different location.
- ◆ Reorganizing the data using either DBA utilities or Fast utilities.*
- ◆ Running applications that do not do task logging.
- ◆ Establishing a recovery base after a device failure.

To create a recovery point for a database, follow these steps:

1. Unload the database from the PDM.
2. Dump the system log files, if any.
3. Format the system log files, if any.
4. Back up all database files, including:
 - Compiled database description (dbmod file)
 - Data set files
 - Task log file
 - Index files, if any
 - **VMS** All indices may be traversed in ascending or descending sequence.
 - SUPRA Server RMS files, if any
 - System log dump files, if using system logging



The dumps from the system logs may be used to roll forward from the previous recovery point up to the current recovery point.

Defining task log recovery

You can define task log recovery either during database creation or while maintaining an existing database.

Defining task log recovery during database creation. When creating a database, SUPRA DBA prompts with:

```
There is no TASK-LOG Do you want to create one (Y/N):
```

after buffer definition. If you respond Y, SUPRA DBA presents the Task Log Definition screen shown on the following page.

Defining or modifying the task log of an existing database. To define or modify the task log of an existing database, select option 5, Task log, from the Database Description Options screen. If the database has a task log, SUPRA DBA displays the Action for Task Log menu. The options on this screen are Examine, Modify, and Delete. If the database does not have a task log, SUPRA DBA prompts with:

```
There is no TASK-LOG Do you want to create one (Y/N):
```

After you enter Y and press RETURN, SUPRA DBA displays the Task Log Definition screen shown below. To change a default value or to modify any of the attributes, enter the appropriate field number after the prompt. After processing the task log details, SUPRA DBA presents a comment screen for the task log:

```
CINCOM SYSTEMS                TASK LOG : ORDERS TASK-LOG
```

```
1 : TASK-LOG-BLOCK-SIZE      : 1
2 : TASK-LOG-NO-OF-BLOCKS   : 200
3 : TASK-LOG-FILE-SPEC      :
4 : TASK-LOG-SHADOW-FILE-SPEC :
```

```
Enter field number to modify a field (or <PF1> to exit):
```

TASK-LOG-BLOCK-SIZE

Description *Optional.* Specifies the size of the task log blocks in multiples of 2048 bytes. SUPRA DBA calculates this value during validation using the largest record in the database plus 30 bytes.

Default 1

Options 1–10

Consideration The block size must be at least large enough to contain the largest database record plus 30 bytes of control information.

TASK-LOG-NO-OF-BLOCKS

Description *Optional.* Specifies the total number of logical blocks in the task log.

Default 200

Format 1–9 numeric characters

Considerations

- ◆ The size of a task log block is a multiple of 2048 bytes as specified in the TASK-LOG-BLOCK-SIZE.

- ◆ The minimum and recommended value is:

$\text{TASK-LOG-BLOCK-SIZE} * 2 * ((\text{MAX-TASKS} + 1) + (3 * \text{MAX-UPDATE-TASKS}))$

TASK-LOG-FILE-SPEC

Description *Required.* Specifies the physical file on disk used for task logging.

Format 1–44 alphanumeric characters

Consideration This field must contain a valid file specification or logical name.

TASK-LOG-SHADOW-FILE-SPEC

Restriction If used, the Shadow Option field for the database must be C.

Description *Optional.* Specifies the physical disk file used for the shadow copy of the task log.

Format 1–44 alphanumeric characters. Must represent a valid UNIX pathname or VMS file specification.

Consideration The shadow task log should be on a different device from the main task log.

Defining Recovery Unit Journaling for RMS data sets (VMS)

Recovery Unit (RU) Journaling is a VMS facility that records all updates to RMS data sets in a journal file. RU Journaling for RMS files is equivalent to Task Logging for PDM files.



The SUPRA Server RMS Recovery Unit Journaling Support option must be activated by Cincom before it can be implemented.

To use RU Journaling, take these steps:

1. Install RMS Journaling as described in the *VMS Journaling Installation Guide and Release Notes*.
2. Enable RU Journaling for each RMS data set associated with your database using the VMS SET FILE command as follows:

```
SET FILE/RU-JOURNAL (file-spec)
```

where (file-spec) is the physical file specification for the RMS data set.

3. Define the logical name CSI_RMS_RU_ON to be TRUE as follows:

```
$DEFINE/GR CSI_RMS_RU_ON TRUE
```



You must define this logical name before you invoke any RDM application that accesses a journaled RMS data set. You can put the logical definition in the group or system logical name table to avoid repeating it for each process.

After a program or system failure, RU Journaling automatically recovers each RMS file to the last successful commit point. The RU Journaling does not support network operations. This means that any attempt to access an RMS file marked for RU Journaling from a remote node is rejected.

Defining system log recovery

You select System Log recovery for each database during database definition, specifying file names for the system log files. The PDM then logs after-images and control functions to these system log files. You define system log recovery either during database creation or when maintaining an existing database.

Defining system log recovery during database creation. To create a system log during database creation, SUPRA DBA asks you to specify a system log after you have defined your database details, buffers, and task log with the following prompt:

```
There is no SYSTEM-LOG Do you want to create one (Y/N):
```

If you respond Y, SUPRA DBA presents the System Log Definition screen.

Defining system log recovery for an existing database. To define or modify the system log of an existing database, select option 6, System log, from the Database Description Options screen (see [“Creating and maintaining a database description”](#) on page 101). If the database has a system log, SUPRA DBA displays the Action for System Log menu. The options on this screen are Examine, Modify, and Delete. If the database does not have a system log, SUPRA DBA prompts with:

```
There is no SYSTEM-LOG Do you want to create one (Y/N):
```

After you enter Y and press RETURN, the System Log screen appears. To change a default value or to modify any of the attributes, enter the appropriate field number after the prompt. After processing the system log details, SUPRA DBA presents a comment screen for the system log.



The comment screen appears as follows in VMS:

```

CINCOM SYSTEMS          SYSTEM LOG : ORDERS SYSTEM-LOG      15-Jun 99  15:54

 1 : SYSTEM-LOG-BLOCK-SIZE      : 1
 2 : SYSTEM-LOG-NO-OF-BLOCKS    : 500
 3 : FILE-1-FILE-SPEC          : SYS$USERDISK:[PDM]TESTDB-SLOG1.LOG
 4 : FILE-2-FILE-SPEC          : SYS$USERDISK:[PDM]TESTDB-SLOG2.LOG
 5 : FILE-1-SHADOW-FILE-SPEC    :
 6 : FILE-2-SHADOW-FILE-SPEC    :

```



The comment screen appears as follows in UNIX:

```

CINCOM SYSTEMS          SYSTEM LOG : ORDERS SYSTEM-LOG      15-Jun 99  15:54

 1 : SYSTEM-LOG-BLOCK-SIZE      : 1
 2 : SYSTEM-LOG-NO-OF-BLOCKS    : 500
 3 : FILE-1-FILE-SPEC          : /user/pdm/testdbslog1.log
 4 : FILE-2-FILE-SPEC          : /user/pdm/testdbslog2.log
 5 : FILE-1-SHADOW-FILE-SPEC    :
 6 : FILE-2-SHADOW-FILE-SPEC    :

```

SYSTEM-LOG-BLOCK-SIZE

Description	<i>Optional.</i> Specifies the size of the system log blocks in multiples of 1024 bytes.
Default	1
Options	1-10

SYSTEM-LOG-NO-OF-BLOCKS

Description *Optional.* Specifies the total number of logical blocks in the system log.

Default 500

Options 5–2,147,483,647

Considerations

- ◆ The size of a system log block is a multiple of 1024 bytes as specified in the SYSTEM-LOG-BLOCK-SIZE.
- ◆ You can calculate the minimum value for the number of blocks using the following equation:

$$(\text{MAXTHREADS} * 4) + 1$$



The extra 1 is the system log control block.

- ◆ A low value for the number of blocks (the minimum) means that database processing will be constantly interrupted by system log dumping. It is, therefore, better to specify a high number of blocks.

FILE-1-FILE-SPEC

Description *Required.* Specifies the physical disk file used for the initial system log.

Format 1–44 alphanumeric characters. Must represent a valid VMS file specification or UNIX pathname.

FILE-2-FILE-SPEC

Description *Required.* Specifies the physical disk file used for the secondary system log.

Format 1–44 alphanumeric characters. Must represent a valid VMS file specification or UNIX pathname.

FILE-1-SHADOW-FILE-SPEC

- Restriction** If used, the Shadow Option field for the database must be C (see “[Creating and maintaining a database description](#)” on page 101).
- Description** *Optional.* Specifies the physical disk file used for the shadow copy of the initial system log.
- Format** 1–44 alphanumeric characters. Must represent a valid VMS file specification or UNIX pathname.

Considerations

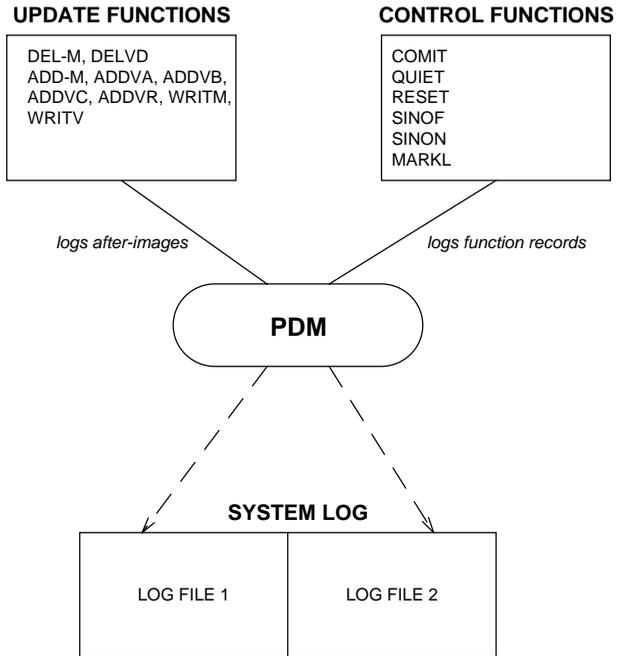
- ◆ If you enter a file spec in this field, you must also enter a file spec in the File-2-Shadow-File-Spec field.
- ◆ The shadow system log should be on a different device from the main system log.

FILE-2-SHADOW-FILE-SPEC

- Restriction** If used, the Shadow Option field for the database must be C (see “[Creating and maintaining a database description](#)” on page 101).
- Description** *Optional.* Specifies the physical disk file used for the shadow copy of the secondary system log.
- Format** 1–44 alphanumeric characters. Must represent a valid VMS file specification or UNIX pathname.
- Consideration** Refer to the considerations under the field “[FILE-1-SHADOW-FILE-SPEC](#)” on page 193.

Maintaining and dumping system logs

A system log consists of two components or log files, which PDM uses in rotation. It logs after-images generated by Update Functions and function records generated by Control functions (see the following figure):



These after-images and control functions gradually fill up the first log file. When the first log file is full, the PDM switches to the second log file. At this point, you should invoke the System Log Dump utility to dump the first log file either to disk (for UNIX environments) or to disk or tape (for VMS environments). The System Log Dump utility is called CSTUDSLF and is usually placed in the directory assigned the logical name SUPRA EXE. Make sure that the logical name CSTUDSLF exists on your system and points to the CSTUDSLF executable image (VMS) or binary (UNIX).

VMS

CSTUDSLF executes automatically by default provided you have set up the logical names CSTUDSLF and DUMPSLF_*dbname* and have created the system log dump input file.

UNIX

CSTUDSLF executes automatically by default provided you have set up the logical name DUMPSLF_*dbname* and have created the system log dump input file.

You can execute CSTUDSLF manually by entering the DUMPSLF operator command online. “[Setting up automatic system log dumping](#)” on page 197 describes how to set up automatic system log dumping and “[Dumping the system log file online](#)” on page 208 describes how to dump the system log file online. The *SUPRA Server PDM System Administration Guide (VMS)*, P25-0130, describes how to define logical names and how to use the DUMPSLF PDM operator command.



Warning: Cincom recommends that you use automatic log dumping. If automatic log dumping is *not* active and the second log file is about 75% full, the PDM sends a message to the operator console prompting you to dump the first log file manually. If you do not dump it at this time, the PDM will either begin signing off users or hang up.

When the second system log file becomes full, the PDM switches back to the first log file. If the first system log file is unavailable because it is still being dumped, the PDM suspends update processing on the database. Update processing resumes when the first log file becomes free.

If the first system log file is unavailable because it is full and the dump program is not active to dump it to tape or disk, the second log file registers "full" at 75%. At this point, the PDM resets and signs off all tasks, using the remaining space in the second system log to log the reset and sign off functions. SUPRA Server displays the following message at any designated operator consoles:

```
CSTI107E Dump required of SLF
component (file spec)
of database (data-base-name) - please invoke manually.
```

prompting you to execute the DUMPSLF operator command.

Signed off tasks first receive a status of ENDT; any further functions receive a NOSO status. Tasks attempting an initial sign-on to the database receive a DSLF status.

Setting up automatic system log dumping

The procedures you use to set up automatic system log dumping vary depending on the environment in which you operate.

Setting up automatic system log dumping under VMS

To set up automatic system log dumping under VMS, you must ensure that the logical name CSTUDSLF exists on your system, create the system log dump input file, and define the logical name DUMPSLF_dbname to point to that file.

Step 1: Ensure that CSTUDSLF exists on your system. Ensure that the logical name CSTUDSLF exists on your system and points to the CSTUDSLF_svc\vl.EXE file. CSTUDSLF exists in the directory assigned the logical name SUPRA_EXE. These logical names should be defined automatically for your environment by the LOGICALS.COM procedure.

Step 2: Create the system log dump input file. The system log dump input file identifies the output location for the dumped log file (either a VMS directory or a tape drive) and specifies the prefix for any output files created by the dump process. The PDM uses the input file to create a parameter file containing details needed by the dump process.

Create an input file using a standard text editor. Each input file is specific to one database, so create one input file for each database for which you enable system logging. Always make sure that the PDM has read access to your system log dump input files.

You specify the log dump input file as follows:

```
{dev:[directory]}  
{device:}
```

output-filename

dev:[directory]

```
{dev:[directory]}  
{device:}
```

Description *Required.* Identifies the location, either disk or tape, to which the PDM will write the dumped system log.

Considerations

- ◆ If the device is a disk, you must also specify the directory—*dev:[directory]*.
- ◆ If the device is a tape, no directory specification is needed— simply a device.
- ◆ If a logical name is used in place of *dev:[directory]*, then be sure that the logical name is in the appropriate logical name table (GROUP, SYSTEM, or special multiple systemwide table) according to the type of PDM in use, and be sure that you place a colon after the logical name.

output-filename

Description *Required.* Defines a name to be used for any output files created by the dump process and defines the name for the dump process itself.

Format 1–13 alphanumeric characters

Considerations

- ◆ The system log dump process will use the *output-filename* to create a maximum of two output files. The files are called:

output-filename_n.LOG

output-filename_n.ERR

where:

output-filename is the 1–13 character file name you specified

n is the number of the system log file being dumped (either 1 or 2). The number is included in the file name in case the dumps of the first and second system log files overlap.

- ◆ The .LOG file contains all messages output from the dump process.
- ◆ The .ERR file contains only error messages.
- ◆ The dump program also sends all messages to the specified operator consoles for the PDM from which it was invoked.
- ◆ The *output-filename* you specify in each input file must be unique because no two processes can share the same name.

dev:[directory]

Description *Required.* Identifies the directory in which you want to create the log file, any error file, and the input parameter file of the dump process.

Format Valid VMS directory specification

Consideration If a logical name is used in place of *dev:[directory]* then be sure that the logical name is in the appropriate logical name table (GROUP, SYSTEM, or special multiple systemwide table) according to the type of PDM in use, and be sure that you place a colon after the logical name.



When you dump a system log file, the PDM checks all previously dumped log files for consistency with the one currently being dumped. Therefore, you must make all system log files available that were dumped since the last system-log format.

Example The following is an example input file:

```
DRB0 : [ PDM . TEST ]  
RDUMP  
DRB0 : [ PDM . LOG ]
```

Step 3: Define the logical name DUMPSLF_dbname. When the PDM invokes the dump program, CSTUDSLF.EXE, it first searches for the logical name DUMPSLF_dbname so that it can read the input file and create the necessary parameters for CSTUDSLF. If this logical name does not exist, or if it points to an invalid file, the dump will not proceed.

Define the logical name DUMPSLF_dbname to point to the input file as follows:

```
DEFINE [ /GR ] [ /SY ] DUMPSLF [xxx_] dbname dev:[dir]filename.ext
```

[/GR]
[/SY]

Description *Optional.* Specifies the table into which the logical name should be placed.

Options /GR Used for a groupwide database
/SY Used for a systemwide database

xxx_

Description *Optional.* Specifies the database prefix.

Format 3 characters followed by an underscore (no space between the underscore and the database name)

dbname

Description *Required.* Specifies the name of the database whose system log file you wish to dump.

Format 6 alphanumeric characters

dev:[dir]

Description *Required.* Identifies the directory specification for the input file.

Format Valid VMS file specification

filename.ext

Description *Required.* Specifies the file name of the dump input file you have created for the specified database.

Format Valid VMS file specification

Example To enable automatic system log dumping for the database TESTDB, create an input file called TESTDB_DUMP.INP and assign it to the logical name DUMPSLF_TESTDB as follows:

```
DEFINE/GR DUMPSLF_TESTDB DRBO:[PDM10.TEST]TESTDB_DUMP.INP
```

What to do if the automatic log dump fails. The automatic dump program may fail for one of the following reasons:

- ◆ You specified an invalid device. For example, you may have specified a device that does not exist or one that is not available, for example, a tape drive that is already allocated to another process.
- ◆ Insufficient system resources are available to initiate the dump program. See the VMS documentation on system resources.

Setting up automatic system log dumping under UNIX

To set up automatic system log dumping under UNIX, you must create the system log dump input file and define the logical name `DUMPSLF dbname`.

Step 1: Create the system log dump input file. The system log dump input file identifies the pathname for the dumped log file and specifies the prefix for any output files created by the dump process.



The PDM uses the input file to create a parameter file containing details needed by the dump process.

Create an input file using a standard text editor. Each input file is specific to one database, so create one input file for each database for which you enable system logging. Always make sure that the PDM has read access to your system log input files.

You specify the log dump input file as follows:

pathname

output-filename

pathname

pathname

Description *Required.* Identifies the directory to which the PDM will write the dumped system log.

Format Valid UNIX pathname

Considerations

- ◆ The PDM will append `csi_dmp_dbmodname.slf` to this path.
- ◆ The pathname must be a full pathname with no embedded environment variables or logical names.

output-filename

Description *Required.* Defines a name to be used for any output files created by the dump process and defines the name for the dump process itself.

Format 1–13 alphanumeric characters

Considerations

- ◆ The system log dump process will use the *output-filename* to create a maximum of two output files. The files are called:

output_filename_n.LOG

output_filename_n.ERR

where:

output-filename is the 1–13 character file name you specified

n is the number of the system log file being dumped (either 1 or 2).

The number is included in the file name in case the dumps of the first and second system log files overlap.

- ◆ The .LOG file contains all messages output from the dump process.
- ◆ The .ERR file contains only error messages.
- ◆ The dump program also sends all messages to the specified operator consoles for the PDM from which it was invoked.
- ◆ The *output-filename* you specify in each input file must be unique because no two processes can share the same name.
- ◆ The PDM appends the *output-filename* to the end of the last pathname in the input file.

pathname

Description *Required.* Identifies the directory in which you want to create the log file and the input parameter file of the dump process.

Format Valid UNIX pathname

Considerations

- ◆ The PDM also creates the input parameter file (see “[Stages in a system log dump](#)” on page 211) in this directory.
- ◆ The PDM appends the output file(s) to this pathname.
- ◆ The pathname must be a full pathname with no embedded environment variables or logical names.



When you dump a system log, all previously dumped system logs must be available so the PDM can check earlier versions for consistency with the version currently being dumped.

Example

The following is an example input file:

```
/data/test
RDUMP
/data/log
```

Step 2: Define the logical name DUMPSLF_dbname. When the PDM invokes the dump program, CSTUDSLF.EXE, the PDM first searches for the logical name DUMPSLF_dbname so that it can read the input file and create the necessary parameters for CSTUDSLF. If this logical name does not exist, or if it points to an invalid file, the dump will not proceed.

Define the logical name DUMPSLF_dbname to point to the input file as follows:

```
csideflog [-g] DUMPSLF_[xxx_]dbname pathname
```

```
[-g]
[-s]
```

Description *Optional.* Specifies either a groupwide or systemwide-logical name.

Options -g Creates a groupwide logical name

-s Creates a systemwide logical name

xxx_

Description *Optional.* Specifies the name of the database prefix.

Format 3 characters followed by an underscore (no space between the underscore and the database name)

dbname

Description *Required.* Specifies the name of the database you wish to dump.

Format 6 alphanumeric characters

pathname

Description *Required.* Specifies the pathname of the input file.

Format Valid UNIX pathname

Example To enable automatic system log dumping for the database TESTDB, create an input file called TESTDB_DUMP.INP and assign to it the logical name DUMPSLF_TESTDB as follows:

```
csideflog -g group DUMPSLF_TESTDB /data/test/TESTDB_DUMP.INP
```

What to do if the automatic log dump fails. The automatic dump program may fail for one of the following reasons:

- ◆ You specified an invalid directory or a directory that does not exist. If this occurs, ensure that the directory you specify is a valid name and that it does exist.
- ◆ Insufficient system resources are available to initiate the dump program. For details on system resources, refer to the *SUPRA Server PDM System Administration Guide (VMS)*, P25-0130, or the *SUPRA Server PDM System Administration Guide (UNIX)*, P25-0132.

Dumping the system log file online

Use the PDM DUMPSLF command to dump the contents of a system log file online. You can use the CSIOPCOM operator interface program to issue the DUMPSLF command. For details on using CSIOPCOM, refer to the *SUPRA Server PDM System Administration Guide (VMS)*, P25-0130, or the *SUPRA Server PDM System Administration Guide (UNIX)*, P25-0132.

Dumping the system log file online under VMS

To dump the system log file online under VMS, you can specify the DUMPSLF command from the CSIOPCOM program (a screen-based user interface) or specify DUMPSLF together with the REPLY command at an operator terminal in response to an operator prompt. Refer to the *SUPRA Server PDM System Administration Guide (VMS)*, P25-0130, for information on using the CSIOPCOM program. The DUMPSLF format is as follows:

```
DUMPSLF [xxx _]database - name [ [uic - group - number] ]
```

xxx _

Description	<i>Optional.</i> Specifies the database prefix.
Format	3 characters followed by an underscore (no space between the underscore and the database name)

database-name

Description	Specifies the database name whose system log files are to be dumped.
Format	6 alphanumeric characters

uic-group-number

Description *Optional.* Specifies the UIC group number that uniquely identifies the database if several databases of the same name are loaded in different groups.

Format 6-digit number

Consideration If you choose to enter a `uic-group-number`, you must code brackets around it. (Do not enter spaces between the brackets and the database name.)

The PDM will not allow you to dump an active system log file— if after-image records are still being logged to it. Use the UNLOAD PDM operator command to sign off these tasks. Refer to the [SUPRA Server PDM System Administration Guide \(VMS\)](#), P25-0130, for descriptions of the PDM operator commands.

Dumping the system log file online under UNIX

You can specify the DUMPSLF command from the csiopcom program (a screen-based user interface) or specify DUMPSLF together with the pdmreply command. Refer to the *SUPRA Server PDM System Administration Guide (UNIX)*, P25-0132, for information on using the csiopcom program. Enter the DUMPSLF command with the pdm reply command at an operator terminal in response to a shell prompt. Either way you use the DUMPSLF command, the DUMPSLF format is as follows:

DUMPSLF [xxx_]database-name[group]

xxx_

Description	<i>Optional.</i> Specifies the database prefix.
Format	3 characters followed by an underscore (no space between the underscore and the database name)

database-name

Description	Specifies the database name whose system log files are to be dumped.
Format	6 alphanumeric characters

group

Description	<i>Optional.</i> Specifies the name or ID uniquely identifying the database if several databases of the same name are loaded in different groups.
--------------------	---

The PDM will not allow you to dump an active system log file— if after-image records are still being logged to it. Use the unload PDM operator command or the disable PDM operator command to sign off these tasks. Refer to the *SUPRA Server PDM System Administration Guide (UNIX)*, P25-0132, for a detailed description of all the PDM operator commands.

Stages in a system log dump

When a dump request is issued either automatically or manually, the PDM performs a series of steps. These steps differ depending on the environment under which you are operating.

Stages in a system log dump under VMS

When you issue a dump request under VMS, the PDM performs the following steps:

Step 1: Check and translate the logical name

DUMPSLF_xxx_database-name. Checks for and translates the logical name `DUMPSLF_xxx_database-name` to find the input file.

Step 2: Open and read the input file. Opens and reads the input file which you created when you set up automatic system log dumping (see [“Setting up automatic system log dumping under VMS”](#) on page 197). It uses the contents of this input file to create a dump parameter file with the name `SL_dbname_uic.INP`,

where:

dbname is the 6-character database name (8–10 characters if a prefix was used)

uic is the 6-digit uic group number

The contents of the parameter file created by the PDM are:

```
OPER=OPERn
MRELAY=x
DATABASE=dbname
OUTPUT=dev:[directory]
FILESPEC=dev:[directory]filename.ext
PDMNAME=pdmname
```

where:

- | | |
|-----------|---|
| OPER= | Identifies the operator console ID for PDM messages. |
| MRELAY= | Specifies whether or not the PDM sends operator messages to a mailbox. |
| DATABASE= | Specifies the 6-character name (4 blank spaces precede the database name if no prefix was used) of the database whose system log is being dumped. |
| OUTPUT= | Identifies the directory or tape specification for the dumped file. |
| FILESPEC= | Identifies the expanded file specification of the system-log-file component to be dumped. Stages in a System Log Dump (VMS). |
| PDMNAME= | Specifies which systemwide PDM will be used if you are using the Multiple Systemwide PDM facility. If you are not using this facility, do not include this parameter in the file. |

Step 3: Create the dump process. Creates the dump process. The dump process then:

- Reads the dump parameter file that was created by the PDM.
- Opens the system log file.
- Reads the device to which the log file will be dumped.
- Checks the previous version of the dump (if any) for consistency with this version.
- Constructs the dump filename in the format:
`CSI_DMP_dbname.SLF`
- Signs on to the PDM using the database name specified in the input file.
- Dumps the system log.
- Closes the dump file and system log.
- Signs off from the database.

Stages in a system log dump under UNIX

When you issue a dump request under UNIX, the PDM performs the following steps:

Step 1: Check and translate the logical name DUMPSLF_ *dbname*.

Checks for and translates the logical name DUMPSLF_ *database-name* to find the input file.

Step 2: Open and read the input file. Opens and reads the input file you created when you set up automatic system log dumping (see “[Setting up automatic system log dumping under UNIX](#)” on page 203). It uses the contents of the input file to create a dump parameter file with the name SLF_ *dbname* _ *gid*.INP,

where:

dbname is the 6-character database name (10 characters if a prefix was used)

gid is the group ID

The contents of the parameter file created by the PDM are:

```
OPER=OPERn
MRELAY=x
DATABASE=dbname
OUTPUT=path name
FILESPEC=path name
```

where:

- | | |
|-----------|---|
| OPER= | Identifies the operator console ID for PDM messages. |
| MRELAY= | Specifies whether or not the PDM sends operator messages to a named pipe. |
| DATABASE= | Specifies the 6-character name (4 blank spaces precede the database name if no prefix was used) of the database whose system log is being dumped. |
| OUTPUT= | Specifies the path for the dumped file. |
| FILESPEC= | Specifies the expanded pathname of the system-log-file component to be dumped. |

Step 3: Create the dump process. After creating the dump process, then the dump process:

- Reads the input file.
- Opens the system log file.
- Reads the device to which the log file will be dumped.
- Checks the previous version of the dump for consistency with this version.
- Constructs the dump filename in the format:
`CSI_DMP_dbname.SLF`
- Signs on to the PDM using the database name specified in the input file.
- Dumps the system log.
- Closes the dump file and system log.
- Signs off from the database.

Running recovery

System log recovery restores the entire database, reapplying all updates and control functions on the dumped system logs to the backup copy of the database. SUPRA Server then uses the task log, if necessary, to recover the restored database back to the last commit point for each task. Use system log recovery following a permanent input/output error on a data set.

To use system log recovery, specify both system logging and task logging in the database description. Unload the dbmod and take a backup of all data files, index files, the task log file, and the dbmod file. Format the system logs. Then if the system fails, perform the following steps:

1. Dump the contents of the remaining system logs (see “[Maintaining and dumping system logs](#)” on page 194). If you attempt to run recovery when data remains on one or both system log files, the recovery program terminates and sends an error message to the operator console asking you to run the system log dump program.
2. Restore the database files, including the index files (if any), the dbmod file, and the task log, but *excluding* the system log, from the backup copy that was created when the system logs were most recently formatted.
3. Run recovery either from DBA or from the command level, either interactively, or in batch (VMS) or as a background process (UNIX).



If you want to recover from tape, ensure that the tape drive is mounted without the /FOREIGN qualifier:

```
$ MOUNT tape-device:label
```

You run system log recovery in one of three ways:

- ◆ Interactively from SUPRA DBA by selecting Recovery from the Administration Functions menu and responding to the prompts.
- ◆ Interactively from the command level by entering a recovery command and responding to the prompts.
- ◆ In batch mode (VMS) or as a background process (UNIX) by submitting a file containing the recovery parameters, or using an on-line command procedure.

Running recovery from SUPRA DBA

To initiate system log recovery from SUPRA DBA, sign on and select function 10, Administration Functions, from the Function Selection for the DBA menu to display the following Administration Functions menu:

```
CINCOM SYSTEMS  SUPRA DBA - ADMINISTRATION FUNCTIONS
```

```
      Select required function:
1 : Format
2 : Utilities
3 : Expand related data set
4 : Reset data set load limit
5 : Recovery
6 : Index facilities
```

```
Enter choice no.: 5
```

Select function 5, Recovery, and respond to the prompts to run system log recovery. The following screens show the system log recovery prompts and example user input:



The online prompts and syntax are the same whether you run system log recovery from SUPRA DBA or from the command level (VMS) or shell (UNIX).



The recovery screen appears as follows in VMS:

```
CINCOM SYSTEMS          SUPRA DBA - RECOVERY          VERSION n.n  15-Jun-99  15:53

Please enter the database name : mrpdbo
Please enter the database password :
Do you wish to recover the database (default is NO)? : YES
Do you wish to print any records (default is NO)? :
Do you require statistics of the run (default is YES)? :
What files do you wish to recover &/or print (default is ALL.)?
: ALL.
Please give input file device/directory
: DKBO:[SUPRA.R22.MSW.DATABASES]
```



The recovery screen appears as follows in VMS:

```
CINCOM SYSTEMS          SUPRA DBA - RECOVERY          VERSION n.n  15-Jun-99  15:53

Please enter the database name : mrpdbo
Please enter the database password :
Do you wish to recover the database (default is NO)? : NO
Do you wish to print any records (default is ALL)? : ALL
Do you require statistics of the run (default is YES)? : YES
What files do you wish to recover &/or print (default is ALL.)?
: ALL.
Please give input file path
: /devel/supral/test
```

Please enter the database name :

Description *Required.* Specifies the name of the database to be recovered.

Format 6 alphanumeric characters

Consideration When specified in a recovery command file, this parameter takes the following format:

DATABASE=xxxxxxx

where xxxxxx is the 6-character database name.

Please enter the database password :

Description *Required* if the database uses a password. Specifies the password assigned to this database.

Format A valid password of 6 alphanumeric characters

Considerations

- ◆ The password you enter does not display on the screen.
- ◆ This word must exactly match the password used by the database description. If you do not enter the correct name in three attempts, the Recovery function terminates.
- ◆ If the database description does not use a password, press RETURN.
- ◆ When specified in a recovery command file, this parameter takes the following format:

```
PASSWORD=xxxxxxx
```

where xxxxxx is the 6-character password.

Do you wish to recover the database (default is NO)? :

Description *Optional*. Confirms that you want system log recovery to proceed; you do not simply want to print system log records or display statistics.

Default N

Options N Do not recover the database

Y Recover the database

Consideration When specified in a recovery command file, this parameter takes the following format:

```
RECOVER=x
```

where x is Y or N.

Do you wish to print any records (default is ALL)? :

Description	<i>Optional.</i> Specifies how much information you want to send to the standard output device.	
Default	ALL	if the response to the preceding recovery prompt was NO
	NO	if the response to the preceding recovery prompt was YES
Options	ALL	Print the first 66 characters of every record
	CONTROL	Print the first 66 characters of COMMIT, MARKL, QUIET, RESET, SINON, and SINOF records
	NO	Print only error messages

Consideration When specified in a recovery command file, this parameter takes the following format:

`PRINT=xxxxxxx`

where xxxxxxx is your chosen Print option.

Do you require statistics of the run (default is YES)? :

Description *Optional.* Indicates whether or not you want a printout of the statistics for the run when it is complete.

Default Y

Options N

 Y

Considerations

- ◆ System log recovery provides the following statistics:
 - The number of records processed.
 - The number of files recovered.
 - The number of records bypassed.
 - The number of control function records (for the following functions: COMMIT, MARKL, QUIET, RESET, SINON, SINOF).
 - The number of records applicable to each file.
- ◆ When specified in a recovery command file, this parameter takes the following format:

STATISTICS=x

where x is Y or N

What files do you wish to recover &/or print (default is ALL.) :

Description	<i>Optional.</i> Specifies data set names that you wish to recover and/or print.
Default	ALL.
Format	One or more 4-character alphanumeric data set names, separated by commas
Options	ALL. Recover all data sets <i>dset1,dset2,dsetn,...END.</i> Recover specified data set names. Specify a maximum of 14 data set names. Terminate the list with END.

Considerations

- ◆ When specified in a recovery command file, this parameter takes the following format:
`FILES=xxxxxx`
where xxxxxx is your chosen option.
- ◆ Terminate the list of data sets with either END. or by omitting the comma (,) separator.
- ◆ For integrity purposes, you should recover all data sets in your database.

VMS Please give input file device/directory :

Description *Required.* Identifies the device and directory (or device alone if the input is on tape) where the dumped system log files are located.

Format A valid VMS device/directory specification up to a maximum of 75 characters.

```
dev:[directory] if the log files are on disk
device:         if the log files are on tape
```

If you are operating in a multimachine environment, you must also specify the node name of the machine as follows:

```
node::dev:[directory] if the log files are on disk
node:device:         if the log files are on tape
```

Considerations

- ◆ You do not need to give the file name of the input file, only the directory specification if stored on disk, or device specification if stored on tape.
- ◆ This specification will be the same as the `OUTPUT=dev:directory` specification used in the system log dump input file. See “[Stages in a system log dump under VMS](#)” on page 211.
- ◆ When specified in a recovery command file, this parameter takes the following format:

```
INPUT=dev:[directory] if the log is on disk
INPUT=device:         if the log is on tape
```

Remember to include the node specification if operating in a multimachine environment:

```
INPUT=node::dev:[directory] or INPUT=node::device:
```

UNIX Please give input file device/directory :

Description *Required.* Identifies the path where the system log input files are located.

Format A valid UNIX path up to a maximum of 64 characters

Considerations

- ◆ You do not need to give the file name of the input file, only the directory path.
- ◆ This specification will be the same as the `OUTPUT= path name` used in the system log dump input file (see “[Stages in a system log dump under UNIX](#)” on page 214).
- ◆ When specified in a recovery shell script, this parameter takes the following format:

`INPUT=directory path`

When you have responded to all of the prompts, the recovery program displays them as you would enter them in a recovery command file and asks you to confirm that you want to proceed with recovery, as on the following screen:

```
Commands (including any defaults are as follows for the database MRPDBO.
  DATABASE=MRPDBO
  PASSWORD=xxxxxxx
  RECOVER=YES
  PRINT=NO                (default value)
  STATISTICS=YES
  FILES=ALL.              (default value)
  INPUT=DKBO:[ SUPRA.R22.MSW.DATABASES]

Do you wish to proceed? : YES
```

Do you wish to proceed ? :

Description *Required.* Asks if you are satisfied with your responses as displayed to you, and whether you wish to proceed with the recovery.

Options Y To proceed with the recovery procedure
N To stop the recovery procedure

Consideration This prompt is displayed only if you run the recovery program interactively. If you submit a command file, as described in “[Running recovery in batch mode \(VMS\) or as a background process \(UNIX\)](#)” on page 227, recovery proceeds automatically once the input file has been located and validated.

Running recovery from the command level (VMS) or at the shell (UNIX)

VMS

You can run system log recovery interactively from the command level. Enter the following RUN command at the command line and respond to the prompts:

```
$RUN CSTURCV
```

UNIX

You can run system log recovery interactively from the shell. Enter the following run command at the shell and respond to the prompts:

```
cturcv
```

The rules for responding to these prompts are the same as for responding to the DBA recovery prompts described in the previous sections.

Running recovery in batch mode (VMS) or as a background process (UNIX)

The following steps demonstrate how to run the recovery program from a command file in batch mode (VMS) or as a background process (UNIX):

1. Use a standard text editor to create a command file containing the input to the recovery program.

VMS command file example. An example of the contents of the RECOVERY.COM file, displayed by issuing the command \$TYPE RECOVERY.COM, is given below. Lines in the command file that begin \$! are comment lines included to help explain the contents of the command file.

```
$
$ RUN CSTURCV
$
$!
$!
$! DATABASE= equates to - Please enter the database name :
$! PASSWORD= equates to - Please enter the database password :
$! RECOVER= equates to - Do you wish to recover the database? :
$! PRINT= equates to - Do you wish to print any of the records? :
$! STATISTICS- equates to - Do you require statistics of the run? :
$! FILES= equates to - What files do you wish to recover &/or print? :
$! INPUT= equates to - Please give input file device/directory
$! specification:
$!
DATABASE=TESTDB
PASSWORD=APRILF
RECOVER=YES
PRINT=NO
STATISTICS=NO
FILES=ALL.
INPUT=MFAL:
$!
$!
```

UNIX **command file example.** An example of the contents of the recovery.com file, displayed by issuing the command `cat recovery.com`, is given below. Lines in the command file that begin # are comment lines included to help explain the contents of the command file:

```
#
#
#
# DATABASE= equates to - Please enter the database name :
# PASSWORD= equates to - Please enter the database password :
# RECOVER= equates to - Do you wish to recover the database? :
# PRINT= equates to - Do you wish to print any of the records? :
# STATISTICS- equates to - Do you require statistics of the run? :
# FILES= equates to - What files do you wish to recover &/or print? :
# INPUT= equates to - Please give input file device/directory
# specification:
#
csturcv << EOF
DATABASE=dbname
PASSWORD=
RECOVER=y
PRINT=no
STATISTICS=y
FILES=ALL.
INPUT=/yourpath
EOF
```

2. Submit the command file to the batch queue as follows:

VMS \$ SUBMIT RECOVERY.COM

(With /NOPRINT, /AFTER=DateTime, or other options.)

UNIX recovery.com >>recovery.log 2>>recovery.log &

(The output of the recovery program will be in a file named recovery.log.)

Or, execute the command file online by typing:

VMS \$ @RECOVERY

UNIX recovery.com

“Running recovery from SUPRA DBA” on page 217 gives the format for the parameters to the recovery program.

6

Validating, compiling, printing, and formatting a database

Before you can use a new database, you must first validate and compile the database description entities contained in the SUPRA Directory. The validate-and-compile process will create a compiled database description file that is used by the PDM. Optionally, you may produce a printed report of the database description by using the Database Description Print function. You select the Validate, Compile and Print functions from the Database Description Function menu or from the command level.

Before you can use the data sets that are defined in a database, you must first format the data sets with the Database Format function. If the data sets have an index, you use the index facilities to create or populate the index. You select the format or index facilities either from the Database Description Function menu, or from the command level.

Before the PDM can use the compiled database description file and before you can run the Format or Index functions, you must define a logical name pointing to the compiled database description file produced by the Compile function, and a logical name to identify a preferred machine list. If you are compiling a new database, you may define these logical names prior to running DBA. Refer to the *SUPRA Server PDM System Administration Guide (VMS)*, P25-0130, or the *SUPRA Server PDM System Administration Guide (UNIX)*, P25-0132, for more information on logical names.

This chapter covers the following topics:

- ◆ Running validate from DBA
- ◆ Running compile and print from DBA
- ◆ Running print from DBA
- ◆ Running batch validate, compile, and print from the command level (VMS) or the shell (UNIX)
- ◆ Running the Format/Reformat function on databases and VMS RMS data sets
- ◆ Formatting, populating, and checking indices

Running Validate from DBA

To run the database Validate function from DBA, select option 8, Validate, from the Database Description Function menu shown below. Any attempt to modify a database during validation receives a "HELD" status.

```
CINCOM SYSTEMS  SUPRA DBA - DATABASE DESCRIPTION FUNCTION

      Function for database descriptions

      1 : Examine
      2 : Modify
      3 : Create
      4 : Delete
      5 : Copy
      6 : List
      7 : Print
      8 : Validate
      9 : Compile and print
(VMS only) 10 : List logical views for the database *
      11 : Data set functions

      Enter choice no : 8

Validate database description name
: ORDERS

Do you want to submit it to batch? (Y/N) : Y
```



* The word *Reserved* displays next to this function.

Validate database description name:

- Description** *Required.* Identifies the database to be validated.
- Format** 6 alphanumeric characters with no embedded spaces. The first character must be alphabetic.

Do you want to submit it to batch? (Y/N) :

Description Specifies whether the validation process is run online or in batch.

Considerations

- ◆ If you respond “N,” the database is validated online. The database validation screen displays the number of errors encountered on the database. At the end of validation, DBA prompts you to modify the database if errors were found, or to compile the database if it validated successfully.
- ◆ If you respond “Y,” DBA prompts for the submit parameters, as shown on the following screen. (**UNIX** User input is lowercase.)

```
CINCOM SYSTEMS SUPRA DBA - DATABASE VALIDATION FOR ORDERS

Name of parameter file
:ORDERS.PAR

Do you want to compile the valid database? (Y/N): Y

Logical database description name (<PF4> will select ORDERS)
:<PF4>

File specification for compiled database (<PF4> will select
order.mod)
:<PF4>

Parameter file created successfully. Do you want to submit it? (Y/N):
Y

Enter submit qualifiers:
```

Name of parameter file

Description Specifies the name of the file to contain the parameters for the batch job.

Default The last parameter file name used, if any.

Format **VMS** Valid file specification

UNIX Full pathname and filename

Considerations

- ◆ If you do not provide a file suffix, the default suffix of .PAR (VMS) or .par (UNIX) is used.
- ◆ The parameter file specification cannot have the suffix .COM (VMS) or .com (UNIX) because a file with this suffix is created during processing.

Do you want to compile the valid database? (Y/N):

Description Indicates whether you want to compile the database if validation is successful.

Logical database description name

Restriction Only displayed if you reply “Y” to the prompt “Do you want to compile the valid database?”.

Description *Optional.* Specifies the name used by application programs to sign on to the compiled database.

Default Database name on the Directory.

Format 6 alphanumeric characters with no embedded spaces. The first character must be alphabetic.

Consideration The database name in the compiled database description may differ from the database name on the Directory if, for example, Multiple Physical Databases (MPDs) are used.

File specification for compiled database

- Restriction** Only displayed if you reply “Y” to the prompt “Do you want to compile the valid database?”.
- Description** Specifies the physical file that contains the compiled database.
- Default** Database name with the suffix “.MOD” (VMS) “.mod” (UNIX)
- Format** **VMS** Valid file specification
UNIX Full pathname and filename
- Consideration** Enter a file specification only if the compiled database description is to be held on a different physical file.

Parameter file created successfully. Do you want to submit it? (Y/N):

Description DBA has stored your responses in a parameter file and prompts to either immediately submit or to save the file for a later submit.

Default Y

Considerations

- ◆ **VMS** If you answer “Y” to submit the parameter file for immediate processing, DBA prompts for the submit qualifiers.
- ◆ **UNIX** If you answer “Y” to submit the parameter file for immediate processing, DBA executes the parameter file as a background process.
- ◆ If you answer “N,” DBA saves the parameter file for later use and returns control to the Database Description Function menu.

Enter submit qualifiers: **VMS**

- Restriction** Displays only if you respond “Y” to the preceding prompt.
- Description** Allows you to specify DCL SUBMIT command qualifiers.
- Default** The system qualifiers for the batch queue
- Format** A string of batch queue qualifiers. Each qualifier must be preceded by a slash (/). Refer to Digital’s *OpenVMS DCL Dictionary* for a detailed description of the SUBMIT command and qualifiers.

Running Compile and print from DBA

Before the PDM can use the compiled database description file produced by the Compile function, you must define: (1) a 6-character logical database name pointing to the compiled database description file and (2) a logical name to identify the machine list. If you are compiling a new database, you may define these logical names prior to running DBA. For procedures on how to define the logical names for the compiled database description and the preferred machine list, refer to the *SUPRA Server PDM System Administration Guide (VMS)*, P25-0130, or the *SUPRA Server PDM System Administration Guide (UNIX)*, P25-0132.

To use the database Compile and print function within DBA, select option 9, Compile and print, from the Database Description Function menu shown below. Any attempt to modify a database during compilation receives a "HELD" status.

```
CINCOM SYSTEMS SUPRA DBA - DATABASE DESCRIPTION FUNCTION
```

```
Function for database descriptions
```

```

1 : Examine
2 : Modify
3 : Create
4 : Delete
5 : Copy
6 : List
7 : Print
8 : Validate
9 : Compile and print
(VMS only) 10 : List logical views for the database *
11 : Data set functions

```

```
Enter choice no : 9
```

UNIX

* The word *Reserved* displays next to this function.

VMS

When you select option 9, the screen appears as follows in VMS:

```
Compile and print database description name(<PF4> will select ORDERS)
:<PF4>

Do you want to submit it to batch? (Y/N):Y

Name of parameter file
<PF4> will select _HSC000$DUA4:[PDM.TEST]ORDERS.PAR;
:<PF4>

Do you want to print database description (Y/N): N

Logical database description name (<PF4> will select ORDERS)
:<PF4>

File specification for compiled database (<PF4> will select
HSC000$DUA4:[PDM.TEST]ORDERS.MOD:)
:<PF4>

Parameter file created successfully. Do you want to submit it? (Y/N): Y

Enter submit qualifiers or <PF4> for a basic submit
:/NOTIFY /NOLOG
```



When you select option 9, the screen appears as follows in UNIX:

```

Compile and print database description name(<PF4> will select orders)
:<PF4>

Do you want to submit it to batch? (Y/N): y

Name of parameter file
<PF4> will select /pdm/test/orders.par;
:<PF4>

Do you want to print database description (Y/N): n

Logical database description name (<PF4> will select orders)
:<PF4>

File specification for compiled database (<PF4> will select
orders.mod)
:<PF4>

Parameter file created successfully. Do you want to submit it? (Y/N): y

```

Compile and print database description name

- Description** *Required.* Identifies the database to be processed.
- Format** 6 alphanumeric characters with no embedded spaces. The first character must be alphabetic.
- Consideration** The database must be valid.

Do you want to submit it to batch?

- Description** Specifies whether to process the database online or in batch.
- Default** Y
- Considerations**
- ◆ If you respond “N,” DBA compiles the database and creates a listing file online. DBA prompts you to specify a logical database name and a file name for the compiled database description.
 - ◆ If you respond “Y,” DBA prompts for the submit parameters.

Name of parameter file

Description Specifies the name of the file to contain the parameters for the batch job.

Default The last parameter file name used, if any.

Format **VMS** Valid file specification

UNIX Full pathname and filename

Considerations

- ◆ DBA uses the default suffix of .PAR (VMS) or .par (UNIX).
- ◆ The parameter file specification cannot have the suffix .COM (VMS) or .com (UNIX) because a file with this suffix is created during processing.

Do you want to print database description? (Y/N)

Description Specifies whether or not to produce a database listing file with the file name database-name.LIS (VMS) or database-name.lis (UNIX).

Logical database description name

Description Specifies the name used by application programs to sign on to the compiled database.

Default Database name on the Directory

Format 6 alphanumeric characters with no embedded spaces. The first character must be alphabetic.

Consideration The database name in the compiled database description may differ from the database name on the Directory if, for example, MPDs are used.

File specification for compiled database

Description Specifies the physical file that contains the compiled database.

Default Database name with the suffix .MOD (VMS) or .mod (UNIX)

Format **VMS** Valid file specification

UNIX Full pathname and filename

Consideration Specify this qualifier only if the compiled database description is to be held on a different physical file from the original database description.

Parameter file created successfully. Do you want to submit it? (Y/N):

Description DBA has stored your responses in a parameter file and prompts to either immediately submit or to save the file for a later submit.

Considerations

- ◆ **VMS** If you answer “Y” to submit the parameter file for immediate processing, DBA prompts you for the SUBMIT qualifiers.
- ◆ **UNIX** If you answer “Y” to submit the parameter file for immediate processing, DBA executes the password or file as a background process.
- ◆ If you answer “N,” DBA saves the parameter file for later use and returns control to the Database Description Function menu.

Enter submit qualifiers **VMS**

Restriction Only displayed if you respond “Y” to the preceding prompt.

Description Allows you to specify the DCL SUBMIT command and qualifiers.

Default The system qualifiers for the batch queue

Format A string of batch queue qualifiers. Each qualifier must be preceded by a slash (/). Refer to the Digital *OpenVMS DCL Dictionary* for a detailed description of the SUBMIT command and qualifiers.

Running Print from DBA

The database Print function generates a listing of a database. Select option 7, Print, from the Database Description Function menu shown below:

```
CINCOM SYSTEMS SUPRA DBA - DATABASE DESCRIPTION FUNCTION

      Function for database descriptions

1 : Examine
2 : Modify
3 : Create
4 : Delete
5 : Copy
6 : List
7 : Print
8 : Validate
9 : Compile and print
(VMS only) 10 : List logical views for the database *
11 : Data set functions

      Enter choice no : 7

Print database description name : ORDERS
```



* The word *Reserved* displays next to this function.

Print database description name :

Description *Required.* Identifies the database to be processed.

Format 6 alphanumeric characters with no embedded spaces. The first character must be alphabetic.

SUPRA DBA returns prompts for the print parameters.

VMS**The screen appears as follows in VMS:**

```
CINCOM SYSTEMS  SUPRA DBA - DATABASE PRINT FOR ORDERS
```

```
Do you want to submit it to batch? (Y/N): Y
```

```
Name of parameter file
```

```
(<PF4> will select DRB0:[TEST]ORDERS.PAR;)
```

```
:BATPRIN.PAR
```

```
Parameter file created successfully. Do you want to submit it?
```

```
(Y/N): Y
```

```
Enter submit qualifiers or <PF4> for a basic submit
```

```
:/AFTER=18:00
```



The screen appears as follows in UNIX:

```
CINCOM SYSTEMS  SUPRA DBA - DATABASE PRINT FOR ORDERS

Do you want to submit it to batch? (Y/N): y

Name of parameter file
(<PF4> will select orders.par)
:batprint par

Parameter file created successfully. Do you want to submit it? (Y/N):
y
```

Do you want to submit it to batch? (Y/N)

Description Specifies whether the database print is run online or in batch (VMS) or as a background process (UNIX).

Considerations

- ◆ If you respond “N,” the database listing is created online. You cannot use the terminal during this time.
- ◆ If you respond “Y,” DBA prompts for the submit parameters as shown on the previous screen.

Name of parameter file

Description	Specifies the name of the file to contain the parameters for the batch job (VMS) or the background process (UNIX).
Default	The last parameter file name used, if any.
Format	VMS Valid file specification UNIX Full pathname and filename

Considerations

- ◆ DBA uses the default suffix of .PAR (VMS) or .par (UNIX).
- ◆ The parameter file specification cannot have the suffix .COM (VMS) or .com (UNIX) because a file with that suffix is created during processing.

Parameter file created successfully. Do you want to submit it? (Y/N):

Description	DBA has stored your responses in a parameter file and prompts to either immediately submit or to save the file for a later submit.
--------------------	--

Considerations

- ◆ **VMS** If you answer “Y” to submit the parameter file for immediate processing, DBA prompts you for the SUBMIT qualifiers.
- ◆ **UNIX** If you answer “Y” to submit the parameter file for immediate processing, DBA executes the parameter file as a background process.
- ◆ If you answer “N,” DBA saves the parameter file for later use and returns control to the Database Description Function menu.

Enter submit qualifiers **VMS**

Restriction	Only displayed if you respond “Y” to the preceding prompt.
Description	Supplies the DCL SUBMIT command and qualifiers.
Default	The system qualifiers for the batch queue
Format	A string of batch queue qualifiers. Each qualifier must be preceded by a slash (/). Refer to Digital's <i>OpenVMS DCL Dictionary</i> for descriptions of the SUBMIT command and qualifiers.

Running Batch Validate, Compile, and Print from the command level (VMS) or at the shell (UNIX)

You can also run the Batch Validate, Compile, and Print facilities from the command level using COMBAT (VMS) or csmcombat (UNIX). This method bypasses the DBA screens; as a result, it is quicker to use. An example combat command file for each environment follows:

Format of the combat command

VMS

```
COMBAT database-name -  
  
/FUNCTION_BATCH=selected-function -  
  
/USERNAME=dba-username -  
  
/PASSWORD=dba-password -  
  
/SIGNON_DB_NAME=logical-database-name -  
  
/OUTPUT=database-description-file-spec -  
  
/NOPRINT
```

UNIX

```
csmcombat DB_NAME=database-name /  
  
FUNCTION_BATCH=selected-function /  
  
USERNAME=dba-username /  
  
PASSWORD=dba-password  
  
SIGNON_DB_NAME=logical-database-name /  
  
OUTPUT=database-description-file-spec /  
  
NOPRINT
```

You can enter the qualifiers to the combat command in any order. In addition, you can abbreviate each qualifier to a minimum unique character string.

VMS

The COMBAT command is described to VMS in a file called COMBAT.CLD. You add this file to the system command table at boot-time, or to your user command table at log in as follows:

```
$ set command supra_coms:csiindex.cld
```

You usually carry out this procedure during the installation of SUPRA Server.

COMBAT VMS

csmcombat UNIX

Description *Required.* The command that initiates Batch Validate, Compile and Print.

Format Must be entered exactly as shown.

database-name VMS

DB_NAME=database-name UNIX

Description *Required.* Specifies the name of the database on the Directory.

Format 6 alphanumeric characters

Considerations

- ◆ The name of the physical file created by Batch Compile consists of the database name with the file extension .MOD (VMS) or .mod (UNIX).
 - ◆ The database must exist on the Directory.
-

/FUNCTION_BATCH=selected-function VMS

FUNCTION_BATCH=selected-function UNIX

Description *Required.* Specifies the function to be performed.

Options VALIDATE [V]

 COMPILE [C]

 PRINT [P]

 ALL [A]

Considerations

- ◆ Minimum abbreviation is /F=*selected-function* (UNIX no slash required).
 - ◆ You may shorten each value to its first character.
-

/USERNAME=dba-username VMS

USERNAME=dba-username UNIX

Description *Required.* Specifies the user name that is to access the Directory.

Format 1–30 alphanumeric characters. Must be a valid DBA user name.

Considerations

- ◆ Minimum abbreviation is /U=dba-username (UNIX no slash required).
- ◆ The specified user name must have at least DEVELOPMENT-PERSONNEL authority to Validate or Compile.
- ◆ A user name with an authority of READ-ONLY may only print the database.

/PASSWORD=dba-password VMS

PASSWORD=dba-password UNIX

Description *Required* unless the specified user name has no password. Specifies the password for the user name which is to access the Directory.

Format 8 alphanumeric characters. Must match the user name.

Considerations

- ◆ Minimum abbreviation is /P=dba-password (UNIX no slash required).
- ◆ If the user name has a password, the password entered must match. If you enter an invalid password, the combat program terminates.
- ◆ The password is displayed on the screen as you type it. You may omit the /PASSWORD qualifier only if the user name has no password.

/SIGNON_DB_NAME=logical-database-name VMS

SIGNON_DB_NAME=logical-database-name UNIX

Restriction Only valid with batch COMPILE or ALL.

Description *Optional.* Specifies the name used by application programs to sign on to the compiled database.

Default Database name on the Directory, specified through the database-name parameter.

Format 6 alphanumeric characters

Consideration The database name in the compiled database description may differ from the database name on the Directory if, for example, MPDs are used.

OUTPUT=database-description-file-spec VMS

OUTPUT=database-description-file-spec UNIX

Restriction Only valid with batch COMPILE or ALL.

Description *Optional.* Specifies the physical file to contain the compiled database description.

Default The database name with a suffix of .MOD (VMS) or .mod (UNIX) in the current directory.

Format VMS Valid file specification

UNIX Full pathname and filename

Consideration Specify this qualifier only if the compiled database description is to be held on a different physical file from the original database description.

/NOPRINT VMS**NOPRINT** UNIX

Restriction	Only valid with batch COMPILE or ALL.
Description	<i>Optional.</i> Does not produce a database listing file.
Default	Produce a database listing file with the file name <i>database-name.LIS</i> (VMS) or <i>database-name.lis</i> (UNIX).

VMS

Running COMBAT without displaying a password. If password security is important, use the following command file to run COMBAT without displaying the user password. Create the command file (called VALCOMPDB.COM in this example) using a standard text editor. Execute the command file from the command level as follows:

```
$ @VALCOMPDB.COM database-name dba-username
```

The prompt "PASSWORD :)" then displays for you to enter your dba-password. The terminal is set to /NOECHO, at this point, so the password does not appear on the screen as you type.

```
$! This command file allows you to run COMBAT interactively
$! without displaying your password
$!
$ SET TER/NOECHO
$ INQUIRE PASSWORD "Password"
$ SET TER/ECHO
$ WRITE SYS$OUTPUT "Invoking COMBAT..."
$ COMBAT 'P1 /USER= 'P2/PASS= 'PASSWORD/FUNC=ALL
```

The following command file, PROCDB.COM, contains the required dba-username and dba-password. To run this command file, you must substitute your own dba-username and dba-password; omit the /PASSWORD qualifier if your user name does not have a password. Execute the command file from the command level as follows:

```
$ @PROCDB.COM
```



You are prompted to enter the initial letter of the function you wish to select; V, C, P, or A (validate, compile, print, or all). Then the prompt "Database :." requests the name of the database to process.

```
$! This command file checks value of the FUNCTION qualifier
$!
      and prompts for the database name
$!
$ IF P2 .EQS. "V" THEN $GOTO INVOKE
$ IF P2 .EQS. "P" THEN $GOTO INVOKE
$ IF P2 .EQS. "C" THEN $GOTO INVOKE
$ IF P2 .EQS. "A" THEN $GOTO INVOKE
$!
$ ASK:
$ INQUIRE P2 " Enter function required, V,C,P or A "
$   IF P2 .EQS. "V" THEN $GOTO INVOKE
$   IF P2 .EQS. "P" THEN $GOTO INVOKE
$   IF P2 .EQS. "C" THEN $GOTO INVOKE
$   IF P2 .EQS. "A" THEN $GOTO INVOKE
$ GOTO ASK
$!
$ INVOKE:
$ IF P1 .EQS. "" THEN
$   INQUIRE P1 " Database "
$ IF P1 .EQS. "" THEN $GOTO INVOKE
$ WRITE SYS$OUTPUT " Invoking COMBAT..."
$ COMBAT 'P1 /USER=USER5/PASS=MOLLIS/FUNC= 'P2
$ EXIT
```

Running Format/Reformat

After you compile a database, you need to allocate disk space for all data set, index, task and system log files. You do this using the Format functions. To format data sets, task and system log files, use the DBA Format function described in this section. You cannot use a database until you have formatted all of the database files.

UNIX

When you use the Format function (as described in “[Running Format from DBA](#)” on page 251), DBA automatically formats all indices as well. No additional index formatting is necessary.

VMS

Format index files using the DBA Index Maintenance utility (CSTUIDX) described in “[Formatting, populating, and checking indices](#)” on page 262.

Run the Format function either by selecting option 1, Format, from the Administrative Functions screen or by running the format program CSTUFMT from the command level.

VMS

The DBA interface to the Format function allows you to format both PDM data sets and RMS data sets. The DCL interface allows you to format only PDM data sets.

UNIX

The DBA interface to the Format function allows you to format PDM data sets.

Running Format from DBA

Access the Format function from DBA by selecting option 1, Format, from the Administration Functions menu.

Before you can run the Format function, you must define: (1) a 6-character, logical database name pointing to the compiled database description file and (2) a logical name to identify the machine list. If you are formatting a new database, you may define these logical names prior to running DBA. For procedures on how to define the logical names for the compiled database description and the preferred machine list, refer to the *SUPRA Server PDM System Administration Guide (VMS)*, P25-0130, or the *SUPRA Server PDM System Administration Guide (UNIX)*, P25-0132.

```
CINCOM SYSTEMS  SUPRA DBA - ADMINISTRATION FUNCTIONS      15-Jun-99  16:32
```

```
      Select required function :
1 : Format
2 : Utilities
3 : Expand related data set
4 : Reset data set load limit
5 : Recovery
6 : Index facilities
```

```
Enter choice no.: 1
```

When you select Format, the Format function signs you off from the Directory and presents the Format Function screen. Enter the choice number and press RETURN.

```
CINCOM SYSTEMS  SUPRA DBA - FORMAT FUNCTION VERSION n.n  15-Jun-99  16:33
```

```
      Select required function :
1 : Format SUPRA database
(VMS only) 2 : Format RMS data set *
```

```
Enter choice no.:
```

UNIX

* This option does not display.

The following are the formatting functions:

- ◆ Format SUPRA database formats (or reformats) a database consisting of SUPRA Server PDM data sets.
- ◆ **VMS** Format RMS data-set formats an RMS data set within a SUPRA Server database.

When you select “Format SUPRA database,” the Format function creates new physical files for both the data set files and the task and system log files. You also use the “Format SUPRA database” function to reformat existing data-set files and task and system logs. To reformat files and task and system logs, use the /RE qualifier when you are prompted for the data set names. You can format some files and reformat others at the same time. (For more information on formatting and reformatting, see the considerations under the “Data sets =” prompt in “[Formatting a SUPRA Server database](#)” on page 253.)

VMS

When you select “Format RMS Data Set,” the Format function creates new physical files for specified RMS data sets.

Formatting a SUPRA Server database

When you select option 1 from the Format Function screen shown on the previous page, the screen below appears. The Format function prompts you to enter the data sets you want to format, reformat, or both. Enter each name and press RETURN, or simply press RETURN to bypass a parameter. Upon completion, the message "FORMAT finished" appears.

Press RETURN to format the next database, or press function key PF1 to display the Format Function menu:

```
CINCOM SYSTEMS SUPRA DBA - FORMAT VERSION  n.n
Name of database to be formatted: ORDERS
Database password:
Data sets =

FORMAT finished. Hit <PF1> to exit, or <return> to FORMAT another database
```

Name of database to be formatted

Description	<i>Required.</i> Specifies the name of the database containing the files to be formatted (the logical database name or the name used to sign on).
Format	6 alphanumeric characters

Database password:

Description	<i>Required</i> if password has been defined. Specifies the password assigned to this database.
Format	6 alphanumeric characters

Considerations

- ◆ If the database description used a password, this name must exactly match that password including any trailing spaces. For example, for FRED, type FRED. If you do not enter the correct password in three attempts, the Format function terminates.
- ◆ If the database description did not use the password parameter, press RETURN.

Data sets =

Description *Required.* Specifies the data sets, task log and/or system log to be formatted.

Considerations

- ◆ **UNIX** DBA automatically formats all indices associated with the selected data set(s).
- ◆ Enter ALL. to format all user data sets in the database including the task log and system log.
- ◆ To format specific data sets, enter each 4-character name after the prompt.
- ◆ To reformat all data sets including the task and system logs, enter ALL./RE.
- ◆ To reformat an existing data set, enter the 4-character name followed by /RE.
- ◆ To format new data sets and reformat existing data sets, follow the rules for each.
- ◆ To create and format a new task log or system log, enter TASKLOG or SYSLOG, respectively.
- ◆ The task log and system log can be reformatted only if all files are reformatted. Otherwise, you must create a new file.

Formatting an RMS data set (VMS)

Select option 2 from the Format Function screen. SUPRA DBA prompts you to enter the names of the database and data sets to be formatted with the following screen. Enter each name and press RETURN. Or if you press RETURN without responding to the prompts, you will return to the previous prompt.

When the formatting finishes, a completion message appears. Press RETURN to format the next database, or press function key PF1 to display the Administration Functions menu.

```
CINCOM SYSTEMS  SUPRA DBA - FORMAT FUNCTION  VERSION 2.2   15-Jun-99  16:35

      Select required function :
      1 : Format SUPRA database
      2 : Format RMS data set
      Enter choice no.: 2

Enter database description name
:      PROJECT

Enter data set name (<PF4> will select DATE)
: <PF4>
```

Enter database description name

- Description** *Required.* Specifies the name of the database containing the RMS files to be formatted.
- Format** 6 alphanumeric characters with no embedded spaces. The first character must be alphabetic.

Enter data set name

- Description** *Required.* Specifies the data set to be formatted.
- Format** 4 alphanumeric characters

Running format from the command level (VMS) or at the shell (UNIX)

VMS

To run Format at the command level, type:

```
$ RUN CSTUFMT
```

UNIX

To run Format at the shell, type:

```
cstufmt
```

The system then prompts you for the database name, the database password, and the data sets. [“Formatting a SUPRA Server database”](#) on page 253 describes the parameters for these prompts.

Running format in batch (VMS) or as a background process (UNIX)

VMS To format a database in batch, create a command file containing the RUN command and the format parameters and submit it to the batch queue.

```

$RUN CSTUFMT
  DBMOD = database-name
  [PASSWORD]

  [ FILES = { ALL. [/RE]
             dataset [/RE][, dataset [/RE]... ] } ]
  [TASKLOG]
  [SYSLOG]
/

```

UNIX To format a database as a background process, create a shell script containing the cstufmt command and the format parameters and execute the script.

```

cstufmt << EOF
  DBMOD = database - name
  [PASSWORD]

  [ FILES = { ALL. [/RE]
             dataset [/RE][, dataset [/RE]... ] } ]
  [TASKLOG]
  [SYSLOG]
EOF

```

RUN CSTUFMT VMS
cstufmt << EOF UNIX

Description *Required.* Initiates the format program.

Format Enter exactly as shown

DBMOD=*database-name*

Description *Required.* Specifies the database containing the data sets to be formatted.

Format 6 alphanumeric character database name

Consideration If the database is in use, use the UNLOAD PDM operator command to unload it before you format the data sets.

PASSWORD

Description *Required* if the database has a password. Specifies the password that protects the database.

Format 1–6 alphanumeric characters

Consideration Leave a blank line between the DBMOD= specification and the FILES= specification if the database does not have a password.

FILES = $\left\{ \begin{array}{l} \text{ALL. [/RE]} \\ \text{dataset [/RE][, dataset [/RE]...]} \end{array} \right\}$

Description *Optional.* Specifies the data sets to be formatted.

Format FILES = Enter exactly as shown. Specifies what you want to format.

ALL. Enter exactly as shown to format all data sets and the task and system logs that are connected to the specified database.

dataset One or more 4-character data-set names, with each name separated by a comma.

/RE Enter exactly as shown to reformat specified data set.

TASKLOG

Description *Optional.* Specifies that the task log should be formatted.

Consideration You cannot use the /RE REFORMAT qualifier when you specify either TASKLOG or SYSLOG.

SYSLOG

Description *Optional.* Specifies that the system log should be formatted.

Consideration You cannot use the /RE REFORMAT qualifier when you specify either TASKLOG or SYSLOG.

/ VMS

EOF UNIX

Description *Required.* Terminates the format specification.

Example command files

Below is an example format command file for each environment:

◆ VMS example 1

```
$! Command file to reformat some data sets in QADBD1
$! and format others.
$ RUN CSTUFMT
DBMOD=QADBD1
FILES=CUST/RE,ADDR/RE,CSOR,CSPA
TASKLOG
SYSLOG
/
```

◆ VMS example 2

To format and/or reformat many but not all of the data sets from a database, you must split FILES= into more than one command line in the command procedure file.



There are restrictions on the length of a command line. See the DEC manual for the length you are allowed to use on your machine.

Below is an example command procedure:

```
$! Command file to reformat some data sets in MPRDDBO
$!
$RUN CSTUFMT
DBMOD=MPRDDBO
PASS
FILES=SCRM/RE,OLTH/RE,ORGM/RE,RQMS/RE,RQIM
FILES=BATH,BATM,RQMT/RE,TEXT,USER
/
```

◆ UNIX example

```
#
# UNIX Shell Script command file to reformat data sets in
# DBNAME
# and format others
#
cstufmt << EOF
DBMOD=DBNAME
FILES=CUST/RE,ADDR/RE,CSOR,CSPA
TASKLOG
SYSLOG
EOF
```

Formatting, populating, and checking indices

Once you have defined an index, you need to format, populate, and activate the index file(s) before you can use them. You format and populate index file(s) using the Index Maintenance utility, CSTUIDX. Populate automatically activates the index; however, you can activate the index manually using the ACTIVATE PDM operator command described in the *SUPRA Server PDM System Administration Guide (VMS)*, P25-0130.

Before you can format and populate an index, you must have performed these steps:

1. Validate and compile the database. (See “Running Validate from DBA” on page 231, “Running Compile and print from DBA” on page 235, and “Running Batch Validate, Compile, and Print from the command level (VMS) or at the shell (UNIX)” on page 244.)
2. Define a 6-character logical name pointing to the compiled database description file. (Refer to the *SUPRA Server PDM System Administration Guide (VMS)*, P25-0130.)
3. Define a preferred machine list `dbname_CSI_PDM_MACS mac1[,mac2...]`. (Refer to the *SUPRA Server PDM System Administration Guide (VMS)*, P25-0130.)
4. Format the database (see “Running Format/Reformat” on page 250).



When you format the database, DBA automatically formats the indices.

You can run the index format program from DBA or the command level. Whether you invoke the program from DBA or at the command level, you are prompted for the same information. In addition, you can use the following function and control keys from DBA and at the command level when formatting an index:

- ◆ **function key PF2 (Help).** Press function key PF2 to display online Help at any prompt. The help is context-sensitive.

VMS

You can display help on other related topics by pressing ? and RETURN at the Topic? prompt as you can with standard VMS help.

UNIX

You can display help on the related topics by entering the topic and pressing RETURN at the Topic? prompt as you can with csihelp.

- ◆ **function key PF3 (options).** Press function key PF3 at the Dataset, Index and Function prompts to display a menu of valid options for the prompt. For example, at the Dataset name prompt, function key PF3 displays a list of data sets. You can select an option from the menu instead of typing it in at the prompt.
- ◆ **function key PF4 (parameters).** Press function key PF4 to modify the current program parameters. SUPRA DBA displays the current parameters and prompts you to enter:
 - A maximum number of errors permitted during an index check
 - The name of the log file to which all error, warning, system, and information messages are written. (If you have already specified a log file, that name is displayed instead of the prompt.)

- ◆ **MINUS (cancel).** Press the MINUS (-) key on the keypad to terminate the current prompt and return control to the previous prompt. It works in the same way as pressing function key PF1 from other DBA screens. If you are at the top level (where the database name prompt is being displayed) the program terminates as if you had pressed **VMS** CTRL-Z or **UNIX** CTRL-D. Selecting the Cancel option from the pop-up menus (displayed by pressing function key PF3) has the same result.
- ◆ **VMS** CTRL-Z (quit). **UNIX** CTRL-D (quit). Press the Quit key to terminate index processing. If you invoked CSTUIDX by selecting option 6, Index facilities, from the Administration Functions menu, the Quit key redisplay this menu. If you invoked CSTUIDX from DCL, the Quit key returns you to the command level. Selecting the Exit option from the pop-up menus (displayed by pressing function key PF3) has the same result.

Running CSTUIDX from DBA

Access the Index Format function from DBA by selecting option 6, Index facilities, from the Administration Functions menu. This displays the copyright screen as shown in the following screen illustration:



Before you can run the Index Format function, you must define: (1) a 6-character logical database name pointing to the compiled database description file and (2) a logical name to identify the machine list. If you are formatting a new database, you may define these logical names prior to running DBA. For procedures on how to define the logical names for the compiled database description and the preferred machine list, the [SUPRA Server PDM System Administration Guide \(VMS\)](#), P25-0130, or the [SUPRA Server PDM System Administration Guide \(UNIX\)](#), P25-0132.

```

+-----+
|  CINCOM SYSTEMS  INDEX MAINTENANCE UTILITY RELEASE n.n  |
+-----+
|                                     Data Entry -----+
| Name of database :                                     |
+-----+
|                                     Information & Warnings -----+
|
|                                     +-----+
|                                     |                                     |
|                                     | (C) Cincom Systems, Inc. 1995.   |
|                                     | All Rights Reserved.             |
|                                     |                                     |
|                                     +-----+
| Use of this software is governed by a license               |
| agreement. This software contains confidential               |
| and proprietary information of Cincom Systems,               |
| Inc. which is protected by copyright, trade                  |
| secret, and trademark law.                                   |
|                                     +-----+
|                                     |                                     |
|                                     +-----+
+-----+
| PF2 = Help  PF3 = Options  PF4 = Parameters  MINUS = Cancel  CTRL/Z = Exit  |
+-----+

```


Name of database:

Description *Required.* Name of the database containing the data set whose index you wish to format.

Format 6 alphanumeric characters with no embedded spaces. The first character must be alphabetic.

Considerations

- ◆ The database must already exist on the Directory.
- ◆ If the database is in use, unload it using the UNLOAD PDM operator command before formatting, populating, or checking the index files.

After you enter the database name, press RETURN to display the password prompt. If there is a database password, enter it and press RETURN to display the data set name prompt. If there is no database password, simply press RETURN to display the data set name prompt.

At this point, you may either enter the information at the prompts or make selections from a series of pop-up menus. If you know the name of the index or indices and the options and functions available, you might find it quicker to enter the information at the prompts. However, if you don't know the name of the index (or indices) and the functions and options available, press function key PF3 (options) to display a series of pop-up menus.

Password

Description Specifies the password defined for the database.

Format 6 alphanumeric character database password

Consideration If the database does not have a password defined for it, press ENTER at the Password prompt.

Processing an index by entering information at the prompts

To format, populate, or check an index by responding to the prompts, perform the following steps:

Step 1: Enter the data set name at the prompt as shown in the following screen illustration:

```
CINCOM SYSTEMS  INDEX MAINTENANCE UTILITY RELEASE n.n

Name of dataset :

-----
----- System Messages -----

PF2 = Help  PF3 = Options  PF4 = Parameters  MINUS = Cancel  CTRL/Z = Exit
```

Name of data set:

Description	<i>Required.</i> Enter the name of the data set(s) containing the indices to be formatted.
Format	<p>ALL.[/RE] Format, populate, or check all of the index files for all of the data sets connected to the specified database.</p> <p><i>dset[/RE][,dset[/RE]...]</i> Format, populate, or check one or more index files for one or more 4-character data set names. Separate each data-set name with a comma.</p>

Considerations

- ◆ If you specify a list, the Index function will always process all of the indices for these data sets. It will not display the indices prompt to allow you to select indices. If you specify only one data set, you can specify one or more indices.
- ◆ Enter /RE after ALL. to reformat all of the index files for all of the data sets.
- ◆ Enter /RE after each data set name to reformat all of the index files for the specified data set.
- ◆ **VMS** If you do not specify /RE, the Index Format function will create a new index file with a higher version number than the old index file.
- ◆ **UNIX** If you do not specify /RE, the index format function will create the new index on top of the existing one without removing the file first.
- ◆ Press function key PF3 to list a pop-up menu of available options as shown in the [screen illustration](#) on page 274.

Step 2: Enter the index name(s) at the Name of index prompt:

```

CINCOM SYSTEMS  INDEX MAINTENANCE UTILITY RELEASE n.n

Name of index :

-----
----- System Messages -----
-----

PF2 = Help  PF3 = Options  PF4 = Parameters  MINUS = Cancel  CTRL/Z = Exit
    
```

Name of index

Description	<i>Required.</i> Enter the name(s) of the index or indices you wish to process.	
Format	ALL.[/RE]	Process all indices connected to the specified data set(s).
	<i>nn</i> [/RE],[<i>nn</i> [/RE]...]	2 alphanumeric character index name. Separate each index name with a comma. No special characters are required.

Considerations

- ◆ Enter /RE after ALL. to reformat all of the index files for the specified data sets.
- ◆ Enter /RE after an index name to reformat the index file. The /RE qualifier is ignored if you subsequently select the Index Check function.
- ◆ The indices you specify must be connected to the specified data set(s).
- ◆ Press function key PF3 to display a pop-up menu of available options as shown in the [screen illustration](#) on page 274.

Step 3: Enter the name of the function you want to perform using the parameters following this screen illustration:

```
CINCOM SYSTEMS  INDEX MAINTENANCE UTILITY RELEASE 2.2

Function :

-----
----- System Messages -----

PF2 = Help  PF3 = Options  PF4 = Parameters  MINUS = Cancel  CTRL/Z = Exit
```

Function:

Description *Required.* Enter the name of the function you wish to perform.

Options F (FORMAT) Creates and formats (or reformats) the physical files for one or more of the indices defined for a SUPRA Server database.

VMS

The index files are of a fixed, allocated size. This size may vary according to the number of secondary keys defined for each index and may exceed the size of the physical data set file for which the index is defined.

UNIX

The index files are dynamic and expand as necessary to store additional keys. Deleted space is reused first before expanding the file size.

P (POPULATE) Formats (or reformats), populates, and activates one or more index files, reading records from the data file and writing corresponding index records to the index file. By formatting a new index file before writing the index records, this function minimizes the risk of inconsistencies between index files and their data files. If a shadow index is defined, this function formats (or reformats) a new shadow index file before writing the index records to it.

If population of the main index file fails, then both the main and the shadow index files are marked as invalid. If the population succeeds for the main index file, but fails for the shadow file, only the shadow file is marked as invalid and PDM index processing can proceed on the main index file only.

C (CHECK) Reads, corrects, and activates indices ensuring that:

- ◆ Each record on a data set has a corresponding record on the index file. On error, CHECK connects the index to the data set.
- ◆ No data inconsistency exists between index file and data file— the data in the secondary key matches that in the data file. On error, CHECK deletes the incorrect index entries and adds new entries.
- ◆ No data exists on the index for which there is no record on the data file. On error, CHECK deletes any excess index records.

Use CHECK to update and activate indices that have been deactivated for any length of time before reactivating them. CHECK will add any missing records and avoid data inconsistencies.

Press function key PF4, Parameters, to specify how many errors can occur before the check is terminated. If you set max errors to -1, CHECK will continue processing until all of the records in the data file have been checked and, if necessary, updated.



U (UNLOCK) Unlocks an index.

Use unlock with caution. Normally, an index file will be locked and inaccessible to the PDM only if there is a problem such as a system failure. During Warm Start Recovery, the PDM uses the lock to determine if the index needs to be populated. If so, the index is populated automatically during Warm Start Recovery.

Considerations

- ◆ You need enter only the first character. Any subsequent character stroke transmits your response.
- ◆ Press function key PF3 to display a pop-up menu of available options as shown on the [screen illustration](#) on page 274.
- ◆ If you did not specify the /RE qualifier, the Index Format function automatically formats and populates new index file(s).

Processing an index using a series of pop-up menus

Step 1: Press function key PF3 to display the Datasets pop-up menu shown in the following screen illustration. Use the information following the figure to make a selection from this menu.

```

CINCOM SYSTEMS  INDEX MAINTENANCE UTILITY RELEASE  n.n

Name of dataset :      +--Datasets--+
                       Exit          +
                       Cancel        +
                       ALL.          +
                       PLM1         +
                       PCHK         +
                       VCHM         +
                       GLJM         +
                       CHKM         +
----- S | GLEN          +-----
----- S | VLIT         +-----
                       PYMT         +
                       +-----+

```

PF2 = Help PF3 = Options PF4 = Parameters MINUS = Cancel CTRL/Z = Exit

The Index Maintenance utility positions the cursor in the first character position of the first function. Use the TAB or arrow keys to move the cursor and press RETURN to select a function. Alternatively, press function key PF1 and then the up arrow to select the first option, or press function key PF1 and then the down arrow to select the last option (RETURN not needed).

The following is a description of the options that display in the Datasets pop-up menu:

- ◆ **Exit** To exit from the Index facilities
- ◆ **Cancel** To remove the pop-up menu
- ◆ **ALL.** To process all data sets connected to the specified data set (these indices are listed below the ALL. option)
- ◆ **4-character data set name** To process that data set

Step 2: Press function key PF3 to display the Indices pop-up menu shown in the following screen illustration. Use the information that follows to make a selection.



After you make a selection of ALL. or a 2-character index name, the system displays the Functions pop-up menu.

```

CINCOM SYSTEMS  INDEX MAINTENANCE UTILITY RELEASE n.n

Name of index :          +--Indices--+
                        |  Exit    |
                        |  Cancel   |
                        |  ALL.     |
                        |  01       |
                        +-----+

-----
----- System Messages -----

PF2 = Help  PF3 = Options  PF4 = Parameters  MINUS = Cancel  CTRL/Z = Exit
    
```

The Index Maintenance utility positions the cursor in the first character position of the first function. Use the TAB or arrow keys to move the cursor, and press RETURN to select a function. Alternatively, press function key PF1 and then the up arrow to select the first option, or press function key PF1 and then the down arrow to select the last option (RETURN not needed).

The following is a description of the options that display in the indices pop-up menu:

- ◆ **Exit** To exit from the Index facilities
- ◆ **Cancel** To remove the pop-up menu
- ◆ **ALL.** To process all indices connected to the specified data set (these indices are listed below the ALL. option)
- ◆ **2-character index name** To process that index

Step 3: Make a selection from the Functions menu shown in the following screen illustration. Use the information following the screen to make a selection.



After you make a selection of ALL. or a 2-character index name, the system displays the Options pop-up menu.

```

CINCOM SYSTEMS  INDEX MAINTENANCE UTILITY RELEASE  n.n

Name of index :
+--Indices--+
|  Exit      | +
|  Cancel    | |
|  ALL.      | |
|  01        | |
+-----+-----+
+-----+-----+
|  Exit      | |
|  Cancel    | |
|  Format     | |
|  Format & Populate  | +-
|  Check     | |
+-----+-----+
----- System |

```

PF2 = Help PF3 = Options PF4 = Parameters MINUS = Cancel CTRL/Z = Exit

The Index Maintenance utility positions the cursor in the first character position of the first function. Use the TAB or arrow keys to move the cursor and press RETURN to select a function. Alternatively, press function key PF1 and then the up arrow to select the first option, or press function key PF1 and then the down arrow to select the last option (RETURN key not needed).

The following is a description of the options that display in the pop-up menu.

- ◆ **Exit** To exit from the Index facilities.
- ◆ **Cancel** To remove the pop-up menu.
- ◆ **Format** To display an options screen to format new index files, or
- ◆ **Format and Populate** To display an options screen to allow you to format and populate existing index files.
- ◆ **Check** To read, correct and activate indices.

For more information on the last three options, see [Options: under "Function:"](#) on page 272.

Running CSTUIDX from the command level

Before running CSTUIDX to format, populate, or check an index online, enter:

```
VMS $SET COMMAND SUPRA_COMS:CSIINDEX.CLD
```

This command adds the commands for CSTUIDX to your process command table using the Command Definition Utility.

Then, to run the procedure, enter:

```
VMS $RUN CSTUIDX
```

```
UNIX cstuidx
```

and respond to the prompts. The prompts are the same as when you run the index format program from DBA (see “[Running CSTUIDX from DBA](#)” on page 265).

Running CSTUIDX in the batch environment

The following two sections discuss running CSTUIDX in the batch VMS version and in the batch UNIX version.

Running CSTUIDX in the batch VMS version

The batch VMS version of CSTUIDX accepts up to three parameters, enabling you to specify an output file, force the utility to accept batch input lines, and specify a certain number of allowable errors before terminating an index check.

To enable the utility to accept these options, the VMS task must first execute the command :

```
$ set command supra_exe:csiindex.cld
```

Then to format, populate, or check an index online, enter:

```
$ csiindex
```

followed by any combination of the following options :

- ◆ `/output=filename` Sends the output from the index utility session to the file name specified.
- ◆ `/batch` Forces the index utility to accept batch input lines. Batch input lines are expected by default during any non-interactive session, but it is advisable to use this option during non-interactive runs of the index utility.
- ◆ `/max_errors=n` Terminates an index check when $n+1$ errors occur. The default value is 0. Specify -1 to permit an unlimited number of errors during an index check. (When you run the index utility from within DBA, or via the `$run cstuidx` command, n is 0.)

When you've entered `$set command...` during the VMS session, the VMS task may enter the `$csiindex...` command as many times as needed. However, when the task signs off VMS, it must reinvoke `$set command...` before using `$csiindex...` in that VMS session.

A sample session that implements these features would include

```
$set command supra_exe:csiindex_22.cld
```

sometime in the session before the Index utility is invoked.

Then, the Index utility may be invoked interactively with the command:

```
$ csiindex/max=-1/out=csiindex.out
```

This command displays the normal interactive index screen. Or, the Index utility may then be run via a command procedure that contains the following lines:

```
$ csiindex/batch/max=-1/out=work_area:csiindex.out
DBMOD=MRPDBO
PASSWORD
FILES=PART
INDICES=AA
FUNCTION=C
```

The following command procedure may also be submitted as a batch job and processed overnight. (Note that you can specify as many indices as will fit on a line for each data set.)

```
$ csiindex/batch/max=-1/out=work_area:csiindex.out
DBMOD=MRPDBO
PASSWORD
FILES=PART
INDICES=AA, BB, CC, DD
FUNCTION=P
```

Running CSTUIDX in the batch UNIX version

The CSTUIDX utility may be executed in batch mode by incorporating it into a script. When running in this mode, the input commands have a different format than when the utility is running interactively. The following syntax must be used when running the utility in batch mode (redirecting input from a file containing interactive input data will not work properly).

The keywords are:

- ◆ DBMOD = *dddddd*
- ◆ PASSWORD = *ppp*
- ◆ FILES = *ffff*
- ◆ INDICES = *ii*
- ◆ FUNCTION = *x*

where:

dddddd is the dbmod name

ppp is the password or blank for no password

ffff is the 4-character dataset name or ALL.

ii is the 2-character index name or ALL.

x is *f* for format, *p* for populate, *c* for check, or *u* for unlock

Examples

1. If you want to perform the same function on all indices in all files, the executable file will be:

```
cstuidx << eof
DBMOD=mrpdbo
                                     (blank line or password)

FILES=all.
FUNCTION=p                             (f, p, c, u)
eof
```

2. If you want to perform the same function on all indices in one file, the executable file will be:

```
cstuidx << eof
DBMOD=mrpdbo
                                     (blank line or password)

FILES=cust
INDICES=all.
FUNCTION=p                             (f, p, c, u)
eof
```

3. If you want to perform different functions on selected files, the executable file will be:

```
cstuidx << eof
DBMOD=mrpdbo
                                     (blank line or password)

FILES=cust
INDICES=c1
FUNCTION=p                             (f, p, c, u)
INDICES=c2
FUNCTION=u                             (f, p, c, u)
FILES=addr
FUNCTION=c                             (f, p, c, u)
eof
```

7

Using the Administration functions

The database Administration functions allow you to control the running of your databases. These facilities include:

- ◆ Defining users
- ◆ Using the Unlock function
- ◆ Expanding data sets
- ◆ Resetting data set load limits
- ◆ **VMS** Controlling programs

Access the first two facilities (Users and Unlock functions) from the Function Selection for the DBA menu. Access the second two facilities (Expand related data set and Reset data set load limit) by selecting option 10, Administration functions, from the Function Selection for the DBA menu. When you select option 10, SUPRA DBA displays the Administration Functions menu. To select a function, enter its number in the "Enter choice no." field and press RETURN.

```
CINCOM SYSTEMS SUPRA DBA-FUNCTION SELECTION FOR THE DBA
```

```
      Select required function :
```

- 1 : Database descriptions
- 2 : Data sets
- (VMS only) 3 : Logical views *
- 4 : Logical data-items
- 5 : Domains
- 6 : Validation tables
- 7 : Programs
- 8 : Users
- 9 : Unlock functions
- 10 : Administration functions

```
Enter choice no.: 10
```



* The word *Reserved* displays next to this function.

```
CINCOM SYSTEMS SUPRA DBA - ADMINISTRATION FUNCTIONS
```

```
      Select required function :
```

- 1 : Format
- 2 : Utilities
- 3 : Expand related data set
- 4 : Reset data set load limit
- 5 : Recovery
- 6 : Index facilities

```
Enter choice no.:
```

Defining users

Select option 8, Users, from the Function Selection for the DBA menu to display the User Maintenance menu. This function adds, changes, deletes and displays user information. You can access this function only if your access authority is Privileged or DBA/Utilities.

To process a User function, enter its number in the “Enter choice no:” field and press RETURN.

```
CINCOM SYSTEMS      SUPRA DBA - USER MAINTENANCE

      1 : Examine details and comments
      2 : Modify details and comments
      3 : Create a new user
      4 : Delete an existing user
      5 : List users

Enter choice no.: 3

Create user name : SENIOR
```

After you select a function, DBA prompts for the name of the user unless you select List. The first word of the prompt varies with the selection. See “[Using SUPRA DBA](#)” on page 61 for a description of the generic DBA functions such as create, examine, and list.

Create user name

- Description** *Required.* Specifies the name of the user to be maintained.
- Format** 1–30 alphanumeric characters and hyphens. The first character must be alphabetic. Embedded blanks are not allowed. Use a hyphen as a connector.
- Consideration** Certain user names have special significance: DATABASE-DESCRIPTIONS, DATA-DICTIONARY, PUBLIC and UTILITIES. These user names are described in the *SUPRA Server PDM System Administration Guide (VMS)*, P25-0130, or the *SUPRA Server PDM System Administration Guide (UNIX)*, P25-0132.

The following are the available User functions:

- ◆ **Examine details and comments.** Displays the user definition and comments created for that user.
- ◆ **Modify details and comments.** Changes the values of the user definition and comments. You cannot change a user name; you must delete it and recreate it.
- ◆ **Create a new user.** Defines a new user to SUPRA Server.
- ◆ **Delete an existing user.** Removes a user definition from SUPRA Server. You cannot delete the user name you used to sign on. DBA responds with a message indicating successful completion.
- ◆ **List users.** Displays a sequential list of all existing user names. You can select a name from the list to display the definition for that user.

Entering user details

If you select Examine, Modify, or Create from the User Maintenance menu, the following User Definition screen is displayed:

```
CINCOM SYSTEMS                USER : SENIOR

1 : USER-ACCESS-AUTHORITY      : DEVELOPMENT PERSONNEL
2 : DIRECTORY-PASSWORD         :
```

Enter field number (or <PF1> to exit) :

USER-ACCESS-AUTHORITY

Description	<i>Required.</i> Specifies the type of access available to this user.
Default	DEVELOPMENT PERSONNEL
Options	Privileged (PRIV) Access to all users and functions.
	DBA/Utilities (DA) Access to all users and functions except Privileged.
	Development Access to database, data set. Read-only access to logical data item. VMS Read-only access to view and program functions.
	Read-only (READ) Read-only access to databases and data sets.
	RDM User (RDM) VMS No access to DBA functions. Defined for use by RDM programs.

Considerations

- ◆ You can enter either whole words or abbreviations.
- ◆ **VMS** Users of all authorities can access views in RDM programs if they are connected to those views.

DIRECTORY-PASSWORD

Description	<i>Optional.</i> Specifies the password assigned to this user for signing on to the system.
Format	1–7 alphanumeric and any printable characters
Consideration	If a password is assigned, the user must enter the password during sign-on to access any component of SUPRA Server.

Entering user comments

After entering the user maintenance information, DBA prompts for comments and displays any existing comments. If you perform maintenance on the comment screen, DBA redisplay the upgraded comments for your review.

Unlocking entities

DBA locks database descriptions on the Directory during database processing. If processing is interrupted for any reason, the database description remains locked. Use the Unlock function to release such database descriptions for use. The Unlock function is available only to users with Privileged or DBA/Utilities access authority.

The Unlock function releases the following entities:

- ◆ All databases and data sets
- ◆ Specific databases and data sets
- ◆ A database and its associated data sets
- ◆ **VMS** A specific view or program
- ◆ A specific domain or validation table



Warning: Do not attempt to unlock any entity currently used by another function because inconsistencies could result.

When you select option 9, Unlock, from the Function Selection for the DBA menu, DBA displays the Unlock Functions menu. Enter the function number in the "Enter choice no." field and press RETURN. If you select a specific database, data set, VMS view, or VMS program function, DBA requests the entity's name. When you press RETURN, DBA responds with a confirmation and statistics message showing the number of databases, data sets, VMS views, programs, domains, or validation tables unlocked.

```

CINCOM SYSTEMS  SUPRA DBA - UNLOCK FUNCTIONS

      Unlock :
      1 : All database descriptions
      2 : All data sets
      3 : One database description
      4 : One data set
      5 : A database and its data set
(VMS only) 6 : A logical view *
(VMS only) 7 : A program *
      8 : A domain
      9 : A validation table

      Enter choice no.: 3

      Database description name
      : TESTDB
      1 database description unlocked successfully      <return>
    
```



* The word *Reserved* displays next to these functions.

Use option 1 or 3 to unlock database descriptions. Their status is changed back to the initial status, as follows:

Status before lock	Initial status
OK, VALIDATED OK or NEEDS VALIDATION	None
BEING COMPILED	VALIDATED OK
BEING MODIFIED or NEEDS VALIDATION	BEING VALIDATED

Use option 5 if a failure occurs while printing, compiling, or validating a database description.

Expanding a related data set

Select option 3, Expand related data set, from the Administration Function menu to increase the physical size of a related data set. Use DBA utilities or CHANGEDB to increase the size of a primary data set.

After the function is successfully completed, Expand modifies the Directory definition to reflect the changes and performs an automatic populate if the database is connected to an index. Expand also updates the compiled database description file with the new values for the total logical records and file-specification allocations.



UNIX The Expand utility will create a backup of the data set before it is modified. This backup will have a .bak extension to the data set being modified (TABV.SDB.bak). The backup file will reside in the same directory as the data set. If you have limited space and you have a current backup of the data set, you may request that a backup not take place. You can accomplish this by setting the logical name CSI_BAK to NO by executing the following command prior to executing the Expand utility:

```
csideflog -p CSI_BAK NO
```

The default for CSI_BAK is YES. If the function is set at NO, no backup or .bak file will be established, or old backup updated, until the function is manually returned to the default of YES.

When you select Expand, DBA returns the following prompts for you to select the data set to expand:

VMS: Sample responses to the expand prompts

```
Enter data set name: REBR

Enter database name: TESTDB

Database password:

Enter file spec for compiled database description: DEMODB
(<PF4> will select TESTDB.MOD)
:TESTDIR:TESTDB.MOD.

*Enter file spec-1 for data set: REBR
(<PF4> will select : REBR.DAT)
:TESTDIR:REBR.DAT

*Enter shadow file spec-1 for data set: REBR
(<PF4> will select: REBR.SHD)
:TESTDIR:REBR.SHD

Data set = REBR

File spec          = DUA0:[USER25.TEST]REBR.DAT;
Shadow file spec=DUA0:[USER25.TEST]REBR.SHD;

Total logical records      =      120
Records in above file spec =      100
Control interval size     =        20

Enter new number of total logical records: 136

NUMBER OF RECORDS ADDED =        20

Do you wish to EXPAND the data set (Y,N): Y
```

UNIX: Sample responses to the expand prompts

Enter data set name: **rebr**

Enter database name: **testdb**

Database password:

Enter file spec for compiled database description: demodb
(<PF4> will select: testdb.mod)

: **/yourpath/testdb.mod**

Enter file spec-1 for data set: rebr
(<PF4> will select: rebr.shd)

: **/yourpath/rebr.dat**

Enter shadow file spec-1 for data set: rebr
(<PF4> will select: rebr.shd)

: **/yourpath/rebr.shd**

Data set = rebr

File spec = /yourpath/rebr.dat

Shadow file spec = /yourpath/rebr.shd

Total logical records = 120

Records in above file spec = 100

Control interval size = 20

Enter new number of total logical records: **136**

NUMBER OF RECORDS ADDED = 20

Do you wish to add a new extent to the dataset (Y, N) : Y

Enter file spec-3 for data set: rebr

: **rebr3.QAR**

Do you wish to EXPAND the data set (Y,N): **Y**

Enter data set name

Description *Required.* Specifies the name of the data set to expand.

Format 4 alphanumeric characters

Consideration Expand can be run only on a related data set.

Enter database name

Description *Required.* Specifies the name of a database associated with the data set to be expanded.

Format 6 alphanumeric characters with no embedded spaces. The first character must be alphabetic.

Consideration The database must be defined on the Directory, and it must be connected to the entered data set.

Database password

Description *Required* if a password has been defined for this database.

Format 6 alphanumeric characters

Considerations

- ◆ If the database description uses a password, this name must exactly match that password. DBA gives you three attempts to enter the password. If the password does not match, Expand terminates and DBA displays the Unlock Functions menu.
 - ◆ If the database description does not use the password parameter, press RETURN.
-

Enter file spec for the compiled database description

Description *Required.* Specifies the name of a physical file containing the database description.

Default The 6-character database name with the suffix .MOD VMS or .mod UNIX.

Format 1–63 alphanumeric characters. Must be a valid VMS file specification or a full UNIX pathname and filename.

Enter file spec-n for data set

- Description** *Required.* Specifies the name of a physical file containing the data set.
- Default** The data set file specification or full UNIX pathname and filename defined on the Directory.
- Format** 1–63 alphanumeric characters. Must be a valid VMS file specification or full UNIX pathname and filename.
- Consideration** If the data set has more than one file-spec, DBA prompts you for each one. The Expand function expands only the last file-spec you define.

Enter shadow file spec-n for data set

- Description** *Required.* Specifies the name of a physical file containing the shadow data set.
- Default** The shadow data set file specification defined on the Directory.
- Format** 1–63 alphanumeric characters. Must be a valid VMS file specification or full UNIX pathname and filename.

Considerations

- ◆ DBA displays this prompt only if the data set is shadowed.
- ◆ If the shadow data set has more than one file specification, DBA prompts you for each one. The Expand function expands only the last file specification you define.

Enter new number of total logical records

Description *Required.* Specifies the new number of records.

Options 2–2,147,473,647

Considerations

- ◆ The value must be larger than the current number of total logical records.
- ◆ The value entered is rounded up, if necessary, to generate a number of complete control intervals.

General considerations

- ◆ You can expand the data set only if the Directory definition of the associated database corresponds to the compiled database description. The database description must have a status of “COMPILED OK.”
- ◆ If the data set definition includes more than one file specification, you can expand only the last file specification.
- ◆ If the data set is connected to more than one database, other databases need not be recompiled if they use a different physical file.
- ◆ You cannot run Expand against an active database.
- ◆ You cannot expand the Directory data sets.

UNIX Do you wish to add a new extent to the dataset (Y, N) :

Description *Required.* Specifies whether you want to add a new extent to the dataset or increase the size of the current extent.

Options Y/N

Considerations

- ◆ When you select N the “Enter file spec for dataset” question is not asked.
- ◆ If there are already four extents defined to the dataset, the “Enter file spec for dataset” question is not asked.
- ◆ When you select Y and specify a new file specification, all of the new records will be added to the new extent.

Enter file spec-*n* for dataset: xxxx

:

Description *Required.* Specifies the Unix file name of the new extent to be created.

Default None

Format Up to 63 alphanumeric characters. Must be a valid UNIX path and file name. File name may include a logical name for the path in the form LOGICAL-NAME:*file name*.

Consideration DBA displays this prompt only if you have answered Y to the question “Do you wish to add a new extent to the dataset.”

Resetting a data set load limit

Select option 4, Reset data set load limit, from the Administration Function menu to alter the load limit for a related data set. Reset updates each control record in a data set to reflect the new load limit. After the function successfully completes, DBA modifies the Directory definition to reflect the changes and updates the compiled database description file with the new value for the cylinder load limit.

When you select Reset from the Function Selection for the DBA menu, DBA returns the following prompts:



DBA displays each prompt after you answer the previous one; it does not display all prompts at the same time.

```
Enter data set name: RELB
Enter database name: DEMODB
Database password:
Enter file spec for compiled database description: DEMODB
(<PF4> will select DEMODB.MOD)
<PF4>
Enter file spec-n for data set: RELS
(<PF4> will select : DEMODIR:RELS1.FIL)
:<PF4>
Enter shadow file spec-n for data set: RELS
(<PF4> will select: DEMODIR:RELS1.SHD)
:<PF4>
```

```
Data set = RELS
*Data set file spec = DUA0:[USER25.DBA]RELS1.FIL;
  *Data set shadow file spec = DUA0:[USER25.DBA]RELS1.SHD;
    Data set load limit =      60
    Enter new load limit:      70
    Do you wish to RESET the data set's load limit (Y,N) : Y
```



* Use a full pathname and filename in the Data set file spec and Data set shadow file spec fields.

Enter data set name

Description *Required.* Specifies the name of the data set to be reset.

Format 4 alphanumeric characters

Considerations

- ◆ You can only run Reset on a related data set.
- ◆ Press RETURN at this stage to return to the Function Selection for the DBA menu.

Enter database name

- Description** *Required.* Specifies the name of a database associated with the data set to be reset.
- Format** 6 alphanumeric characters with no embedded spaces. The first character must be alphabetic.
- Consideration** The database must be defined on the Directory and it must be connected to the entered data set.
-

Database password

- Description** *Required* if a password is defined for this database.
- Format** 6 alphanumeric characters
- Considerations**
- ◆ If the database description uses a password, this name must match that password. DBA gives you three attempts to enter the password. If the password does not match, Reset terminates and DBA displays the Unlock Functions menu.
 - ◆ If the database description does not use the password parameter, press RETURN.
-

Enter file spec for the compiled database description

- Description** *Required.* Specifies the name of a physical file containing the database description.
- Default** The 6-character database name with the suffix .MOD (VMS) or .mod (UNIX).
- Format** 1–63 alphanumeric characters. Valid VMS file specification or full UNIX pathname and filename.

Enter file spec-n for data set

- Description** *Required.* Specifies the name of a physical file containing the data set.
- Default** The data set file specification or the full UNIX pathname and filename defined on the Directory.
- Format** 1–63 alphanumeric characters. Valid VMS file specification or full UNIX pathname and filename.
- Consideration** If the data set has more than one file specification, DBA prompts you for each one.

Enter shadow file spec-n for data set

- Description** *Required.* Specifies the name of a physical file containing the shadow data set.
- Default** The shadow data set file specification defined on the Directory.
- Format** 1–63 alphanumeric characters. Valid VMS file specification or full UNIX pathname and filename.
- Considerations**
- ◆ DBA displays this prompt only if the data set is shadowed.
 - ◆ If the shadow data set has more than one file specification, DBA prompts for each one.

Enter new load limit

Description *Required.* Specifies the new data set load limit.

Options **VMS** 1–99

UNIX 0–99

Considerations

- ◆ **UNIX** If the LOAD-LIMIT is set to 0, the behavior of the related data set space-allocation mechanism is modified. The first record of a list is added in strict round-robin fashion from the first control interval to the last, and then back to the first.
- ◆ Records added to existing chains follow the same rules defined for LOAD-LIMIT=1–99. This mechanism can be very effective in evenly loading data sets. Some data sets tend to get front-loaded when the standard LOAD-LIMIT of 1–99 is used. The LOAD-LIMIT=0 option tends to spread the chains out so that the file is more evenly distributed.
- ◆ There will no longer be a LOAD status returned to an application on sign off when a data set is approaching the FULL condition.

General considerations

- ◆ If the data set is connected to more than one database, other databases need not be recompiled if they use a different physical file.
- ◆ You cannot run Reset against an active database.
- ◆ You can alter the data set load limit only if the Directory definition of the associated database corresponds to the compiled description.
- ◆ You cannot use Reset to alter the load limit of the Directory data sets.
- ◆ DBA updates any shadow data set as above.

Controlling programs (VMS)

Use option 7, Programs, from the Function Selection for the DBA menu to identify the views used by an application program and to perform restricted maintenance on application programs described on the Directory.

When you run an RDM preprocessor, SUPRA Server enrolls the specified program onto the Directory, giving its name, date and time of preprocessing, its language, and a list of the views it uses. This is the only way to add program information to the Directory. Program enrollment details on the Directory (except comments) cannot be modified. Refer to the *SUPRA Server PDM System Administration Guide (VMS)*, P25-0130, for further information about program enrollment.

Enter the number of the function in the "Enter choice no." field and press RETURN. To list the programs in the Directory, select any program function and when prompted for the program name, press function key PF3 to display a menu.

```
CINCOM SYSTEMS          SUPRA DBA - PROGRAM FUNCTION

      Functions for programs
1 : Details
2 : Logical views used
3 : Comments
4 : Delete

Enter choice no.:
```

The following are the program functions:

- ◆ **Details** Examines the attributes of the program. Read-only access.
- ◆ **Logical Views Used** Displays all base and derived views the program uses. Read-only access.
- ◆ **Comments** Displays comments associated with the program. Read-only users have read-only access; other users can modify comments.
- ◆ **Delete** Deletes a program from the Directory. This function disconnects associated views but does not delete them. In other words, you can still run the program provided the views it uses are up-to-date and available to the user name specified. SUPRA DBA does not display this option to Read-only users. Other users can delete information about any program.

Examining the details of a program (VMS)

When you select Details from the Program Function menu, SUPRA DBA prompts for the program name. Enter the name and press RETURN. SUPRA DBA returns the program name, status, enrollment date, enrollment time, modification level, and language (BASIC, COBOL, or FORTRAN).

The first time a program is preprocessed and automatically enrolled on the Directory, SUPRA DBA sets the modification level to 0. Each subsequent time the program is preprocessed and automatically enrolled, the modification level is incremented by 1. (The following screens use ACCOUNTS-RECEIVABLE as an example.)

```
CINCOM SYSTEMS      SUPRA DBA - PROGRAM FUNCTION
```

```
      Functions for programs
1 : Details
2 : Logical views used
3 : Comments
4 : Delete
```

```
Enter choice no.: 1
```

```
Examine Details of program name
: ACCOUNTS-RECEIVABLE
```

```
CINCOM SYSTEMS      PROGRAM : ACCOUNTS-RECEIVABLE
```

```
1 : PROGRAM-NAME           : ACCOUNTS-RECEIVABLE
2 : PROGRAM-STATUS        : O.K.
3 : PROGRAM-ENROLLMENT-DATE : 01-JUN-95
4 : PROGRAM-ENROLLMENT-TIME : 10:40:23:00
5 : PROGRAM-MODIFICATION-LEVEL : 1
6 : LANGUAGE              : COBOL
```

```
Enter field number (or <PF1> to exit) :
```

Displaying the views used by a program (VMS)

When you select Logical views used from the Program Function menu, SUPRA DBA prompts for the name of the program:

```
CINCOM SYSTEMS          SUPRA DBA - PROGRAM FUNCTION

      Functions for programs
1 : Details
2 : Logical views used
3 : Comments
4 : Delete

      Enter choice no.: 2

List logical views used by program name
(<PF4> will select ACCOUNTS-RECEIVABLE)
:<PF4>
```

After you enter the name, SUPRA DBA displays the names of the views used by the program:

```
CINCOM SYSTEMS  ACCOUNTS-RECEIVABLE

Logical views used by the program :
1 : CUSTOMER-ORDERS
2 : ITEMS-SHIPPED
3 : SHIPPING DATES
4 : CUSTOMER-BILLING
5 : BILLING-DATES
6 : PAST-DUE-ACCOUNTS
```

Maintaining comments (VMS)

When you select Comments from the Program Function menu, SUPRA DBA prompts for the name of the program. Enter the name and press RETURN:

```

CINCOM SYSTEMS      SUPRA DBA - PROGRAM FUNCTION

      Functions for programs
1 : Details
2 : Logical views used
3 : Comments
4 : Delete

      Enter choice no.: 3

Comments for program name
(<PF4> will select ACCOUNTS-RECEIVABLE)
:<PF4>

```

SUPRA DBA returns a list of the comments associated with the program. Any comments you have added remain if you reenroll the program using an RDM preprocessor. You lose the comments if you delete the program information from the Directory.

```

CINCOM SYSTEMS      Comments for ACCOUNTS-RECEIVABLE

1 ACCOUNTS-RECEIVABLE is used for weekly updates

action : A,C,D,F,L,M,N,O,P,R or W - Hit <PF2> for explanation
:

```

Deleting program information from the directory (VMS)

When you select Delete from the Program Function menu, SUPRA DBA prompts for the name of the program to be deleted from the SUPRA Directory. After you enter the name, SUPRA DBA displays a confirmation message before deleting the program. Check the name and press Y to delete or N to cancel.

```
CINCOM SYSTEMS          SUPRA DBA - PROGRAM FUNCTION

                          Functions for programs
                          1 : Details
                          2 : Logical views used
                          3 : Comments
                          4 : Delete

                          Enter choice no.: 4

Delete name
: INVENTORY-CONTROL-4
Confirm deletion of INVENTORY-CONTROL-4 (Y or N) : Y
```

8

Defining Multiple Physical Databases (MPDs)

The Multiple Physical Database (MPD) facility allows you to define databases that share the same logical structure but that are implemented with different physical database files. You might use the MPD facility for the following reasons:

- ◆ To have a small version of your production database containing peculiar or “extreme” data for testing purposes. Keeping smaller physical files saves valuable disk space, and using separate test data enables you to thoroughly test programs before putting them into production.
- ◆ To manage several remote sites from a central SUPRA Directory. The central SUPRA Directory contains a separate physical description with different file sizes or file specifications for each remote site.

MPD is particularly useful where your corporation has one site at which central software development is carried out for distribution to other sites. MPD improves control of the distribution of software and data because it centralizes the information on one SUPRA Directory.

You implement an MPD in three steps:

1. Copy a valid, compiled database description using Copy (option 5 on the Database Description Function Menu). (Note that the Copy function does not copy the task log file specifications or the system log file specifications.) You now have two databases with different names. The databases are identical apart from task and system logs.
2. Modify the copied database as follows:
 - a. Change the physical file specifications for the main and shadow data sets.
 - b. Change the database details if required.
 - c. Define new task log specifications.
 - d. Define new system log specifications.
3. Validate and compile the database. You may need to change the logical database description name to that used by application programs to sign on to the database. (See “[Validating, compiling, printing, and formatting a database](#)” on page 229 for details of how to change the logical database name when compiling a database.)

At this point, you have two copies of the same logical database description. Both databases share the logical structure and logical database name, but they are held on two different, compiled database description files.

For example, your original database description is ORDERS, the logical sign-on name used by programs is ORDERS, and the compiled database description file is ORDERS.MOD. Your copied database description is TESTDB, the logical sign-on name used by programs is ORDERS, and the compiled database description file is TESTDB.MOD. Both database descriptions share the same logical sign-on name, ORDERS, so programs do not need to change to access either database description. To make your programs access the TESTDB database instead of ORDERS, redefine the logical name ORDERS to point to the TESTDB.MOD file instead of ORDERS.MOD; for example:

```
VMS $DEFINE/GR ORDERS dev:[directory]TESTDB.MOD
```

```
UNIX csideflog -g ORDERS TESTDB.MOD
```

Copying a database description

Select option 5, Copy, from the Database Description Function menu as shown below to copy an entire database description:

```
CINCOM SYSTEMS SUPRA DBA-DATABASE DESCRIPTION FUNCTION

      Function for database descriptions:
1 : Examine
2 : Modify
3 : Create
4 : Delete
5 : Copy
6 : List
7 : Print
8 : Validate
9 : Compile and print
(VMS only) 10 : List logical views for the database *
11 : Data set functions
      Enter choice no.: 5

Copy database description name
: ORDERS
To database description name
: TESTDB
: Do you want to copy all the logical views? (Y,N) : Y VMS
```



* The word *Reserved* displays next to this function.

Copy database description name

Description *Required.* Identifies the database to be copied.

Format 6 alphanumeric characters with no embedded spaces. The first character must be alphabetic.

Considerations

- ◆ The database must exist on the Directory.
- ◆ You should always copy a valid database, although the database does not need to be compiled before it is copied.
- ◆ Press function key PF3 to display a list of available databases.

To database description name

Description *Required.* Identifies the database to be created.

Format 6 alphanumeric characters with no embedded spaces. The first character must be alphabetic.

Considerations

- ◆ The database must not exist on the Directory.
- ◆ Press function key PF3 to display a list of database names that are already in use.

Do you want to copy all the logical views? (Y,N)

Description *Required.* Specifies whether you want to link all of the views associated with the original database to the new database.

Options Y
 N

Consideration The new database uses the views of the original database without copying the text of the view. Therefore, if you change the text of a view, both databases will use the changed view.

Modifying a copied database description

Select option 2, Modify, from the Database Description Function menu to display the Database Description Options menu as shown below.

```
CINCOM SYSTEMS SUPRA DBA - DATABASE DESCRIPTION OPTIONS

      Options for database description TESTDB:
1 : Details
2 : Comments
3 : Data sets
4 : Buffers
5 : Task log
6 : System log

Enter choice no.:
```

The following is a description of the options that display in the pop-up menu.

- ◆ **Details** To change the database details.
- ◆ **Data sets** To change the physical file specifications.
- ◆ **Task log** To define a new task log file specification.
- ◆ **System log** To define a new system log file specification if required.

For additional information on these database description options, see [“Creating and maintaining a database description”](#) on page 101 and [“Defining and running recovery”](#) on page 183.

Validating and compiling a copied database

Once you modify the physical file attributes of the copied database, select option 8, Validate, from the Database Description Function menu. When validation is complete, compile the database. During database compilation, SUPRA DBA prompts for the logical database description name. Remember to specify the logical database description name used by application programs to sign on to the database. SUPRA DBA provides a default logical database name, which is the same as the database name on the Directory.

See “[Validating, compiling, printing, and formatting a database](#)” on page 229 for additional information on both the Validate and Compile functions.

9

Defining logical data-items, domains, validation tables, and views

After defining the physical database, you create views to access and manipulate the data held on that database. You create two types of views: base and derived. Base views make up the conceptual schema and access the physical database description directly. Derived views make up the external schema and access data only through other views.

Base views contain a list of logical data-items and an access definition describing how to navigate the physical database to obtain those data-items. A logical data-item is an alternative name for a physical data-item in the database description and is usually more meaningful than a physical data-item name. For example, the physical key into the customer data set might be CUSTCTRL. A more meaningful logical data-item name for CUSTCTRL would be CUSTOMER-NAME. You can assign any number of logical data-item names to one physical data-item, and each logical data-item name corresponds to only one physical data-item. A logical data-item can be used in many views and, when included in a view, is referred to as a column.

This chapter describes how to create logical data-items and associated domains, and how to use the DBA interface to EDIT/EDT to define views. This chapter does not describe the rules for defining base and derived views, nor how to design column and access definitions. Refer to the *SUPRA Server PDM RDM Administration Guide (VMS)*, P25-8220, for a description of the syntax of base and derived views.

Defining logical data-items

Select option 4 from the DBA Function Selection menu to display the Logical Data-item Function menu. Enter the number of the desired function in the “Enter choice no.” field and press RETURN.

```
CINCOM SYSTEMS  SUPRA DBA - LOGICAL DATA-ITEM FUNCTION

      Functions for logical data-items
1 : Examine
2 : Modify
3 : Create
4 : Delete
5 : List logical data-items for data set

      Enter choice no.:
```

The following functions are available from the menu:

- ◆ **Examine** Displays attributes for the associated physical data-item and any comments for the logical data-item.
- ◆ **Modify** Changes attributes for the associated physical data-item or comments.
- ◆ **Create** Adds a new logical data-item and identifies its associated physical data-item.
- ◆ **Delete** Deletes a logical data-item. The associated physical data-item is disconnected from this logical data-item but is not deleted.
- ◆ **List logical data-items for data set** Lists all logical data-items for a data set. This function requires a data set name.

Examining or modifying a logical data-item

When you select Examine or Modify from the Logical Data-item Function menu, SUPRA DBA prompts for the name of the logical data-item to be displayed. The menu containing the options then appears. If you select Modify, you can change the physical data-item details or any comments. Examine provides read-only access. Enter the name and press RETURN. All examples are shown with a logical data-item named CUSTOMER-NAME.

```
CINCOM SYSTEMS      SUPRA DBA - CUSTOMER-NAME

      Examine
1 : Physical data-items attributes
2 : Association with a data-item
3 : Comments

      Enter choice no.: 1
```

Select option 1, Physical data-items attributes, to display the Attribute screen that identifies the physical data-item associated with this logical data-item.

```
CINCOM SYSTEMS      DATA-ITEM : CUSTNAME

      LOGICAL DATA-ITEM : CUSTOMER-NAME

1 : DATA-ITEM-NAME           :CUSTNAME
2 : DATA-ITEM-LENGTH         :20
3 : DATA-ITEM-USE            :KEY
4 : DATA-ITEM-TYPE           :CHARACTER
5 : DATA-ITEM-SIGN           :SIGNED
6 : DATA-ITEM-DECIMAL-PLACES :0

      Enter field number to modify a field (or <PF1> to exit) :
```

If you select option 2, Association with a data-item, SUPRA DBA identifies the physical data-item associated with this logical data-item.

```
CINCOM SYSTEMS      SUPRA DBA - CUSTOMER-NAME

      Examine
1 : Physical data-items attributes
2 : Association with a data-item
3 : Comments

      Enter choice no.: 2

CUSTOMER-NAME connected to CUSTNAME                                <PF1>
```

If you select option 3, Comments, SUPRA DBA returns any comments written about this logical data-item.

```
CINCOM SYSTEMS      Comments for CUSTOMER-NAME

1 The logical data-item, CUSTOMER-NAME identifies all customers
2 used in the ORDERS database. This logical data-item is
3 connected to several views.

Action : C,L,N,O,P or W - Hit <PF2> for explanation
```

Creating a logical data-item

When you select Create from the Logical Data-item Function menu, SUPRA DBA prompts for the names of the associated data set, the physical data-item and the logical data-item. Enter the requested names and press RETURN. Press function key PF3 for a menu of available data sets.

```
CINCOM SYSTEMS  SUPRA DBA - CREATE LOGICAL DATA-ITEM

Enter physical data-item information

Data set name : CUST

Four-character data-item name : NAME

Create logical data-item name : CUSTOMER-NAME
```

Data set name

Description *Required.* Specifies the name of the data set containing the associated physical data-item.

Format 4 alphanumeric character name of existing data set on the Directory

Four-character data-item name

Description *Required.* Specifies the physical data-item associated with this logical data-item.

Format 4 alphabetic characters

Considerations

- ◆ The first 4 characters of the name are the same as the data set name. You enter only the last 4 characters.
 - ◆ A physical data-item can be associated with more than one logical data-item.
 - ◆ The data-item definition must already be on the Directory.
-

Create logical data-item name

Description *Required.* Specifies the name of the logical data-item being created.

Format 1–26 alphanumeric characters and hyphens. The first character must be alphabetic. The last character cannot be a hyphen. Hyphens cannot be consecutive characters.

Considerations

- ◆ A logical data-item can be associated with only one physical data-item.
- ◆ Only attempt to create a new logical data-item for a database that has been successfully validated.

After you enter the naming data for the new logical data-item, SUPRA DBA returns the details for the associated physical data-item. The screen displays the physical data-item name and any default values. (The following screen uses CUSTNAME as an example.)

```

CINCOM SYSTEMS                DATA-ITEM : CUSTNAME

                                LOGICAL DATA-ITEM: CUSTOMER-NAME

1 : DATA-ITEM-NAME   : CUSTNAME
2 : DATA-ITEM-LENGTH : 20
3 : DATA-ITEM-USE    : DATA
4 : DATA-ITEM-TYPE   : CHARACTER
5 : DATA-ITEM-SIGN   : SIGNED
6 : DATA-ITEM-DECIMAL-PLACES : 0
7 : DATA-ITEM-SUBDATA-ITEMS : 0
8 : DATA-ITEM-LEVEL  : 0

Enter field number to modify a field (or <PF1> to exit) :

```

DATA-ITEM-NAME

Description Displays the physical data-item named on the previous screen.

DATA-ITEM-LENGTH

Description Displays the defined length of the physical data-item.

DATA-ITEM-USE

Description Displays whether this physical data-item is a code, data, a key, or a linkpath.

DATA-ITEM-TYPE

Description	<i>Optional.</i> Specifies the type of data you wish to maintain in this data-item. RDM uses this information in conjunction with other data-item details to describe the data-item in programs.
Default	CHARACTER
Options	B BINARY C CHARACTER K KANJI L LEADING SEPARATE NUMERIC N NUMERIC TRAILING OVERPUNCH P PACKED DECIMAL F FLOATING POINT Z ZONED TRAILING NUMERIC

Considerations

- ◆ Enter either the single character or the complete word. SUPRA DBA displays the complete word.
- ◆ SUPRA DBA validates the data-item's type, length, sign and number of decimal places after you finish modifying them. If they are invalid, you must correct them.
- ◆ The data-item's type affects the data-item's length as shown in the following table:

Type	Length	Sign	Considerations										
BINARY	1,2,4, or 8	Signed or Unsigned	The available digits vary according to length: <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>Length</th> <th>Digits Available</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>3</td> </tr> <tr> <td>2</td> <td>5</td> </tr> <tr> <td>4</td> <td>9</td> </tr> <tr> <td>8</td> <td>20</td> </tr> </tbody> </table>	Length	Digits Available	1	3	2	5	4	9	8	20
Length	Digits Available												
1	3												
2	5												
4	9												
8	20												
CHARACTER	1–4096	Unsigned											
LEADING SEPARATE NUMERIC	1–18	Signed or Unsigned	If signed, length available is 2–18. The 1 character required for the sign must be included in the calculation of the field length.										
NUMERIC TRAILING OVERPUNCH	1–18	Signed or Unsigned	If signed, trailing digit is overpunched with the sign.										
PACKED	1–9	Signed	Gives a maximum of 19 digits.										
FLOATING	4 or 8	Signed	Decimal places must be set to 0.										
ZONED TRAILING NUMERIC	1–18	Signed	If signed, the trailing digit is zoned with the sign.										

DATA-ITEM-SIGN

Description *Required.* Specifies if this is a signed data-item.

Default SIGNED

Options S SIGNED

 U UNSIGNED

Considerations

- ◆ Must be SIGNED if type is P or F.
- ◆ RDM ignores this value if the data-item type is C.
- ◆ Enter either the single character or the complete word. SUPRA DBA displays the complete word.
- ◆ If you change this value for one logical data-item, the value changes for any other logical data-items associated with this physical data-item.

DATA-ITEM-DECIMAL-PLACES

Description *Optional.* Specifies the number of decimal places. RDM uses this value in conjunction with other details (length, sign) to describe the data-item in programs.

Default 0

Format 1–5 numeric characters

Consideration The numeric value depends on the values for DATA-ITEM-TYPE and DATA-ITEM-LENGTH according to the following table:

Data-item type	Data-item length	Data-item decimals
B, Binary	1	Up to 2 decimal places
	2	Up to 4 decimal places
	4	Up to 9 decimal places
	8	Up to 18 decimal places
C, Character	1–4096	0
F, Floating Point	4 or 8	0
K, Kanji	1–4096	0
L, Leading Numeric	1–18	Data-item length minus one
N, Numeric	1–18	Up to 18 decimal places
P, Packed Numeric	1–9	Maximum 19 decimal places
Z, Zoned Numeric	1–18	Up to 18 decimal places

DBA validates the data-item's type, length, sign and number of decimal places validated after you have finished modifying them. If they are invalid, you must correct them.

DATA-ITEM-SUBDATA-ITEMS

Description Displays the number of subdefined data-items owned by this data-item.

DATA-ITEM-LEVEL

Description Displays the level of the data-item.

The following table shows COBOL, FORTRAN, and BASIC descriptions for data-items:

Signed/ unsigned	Type	Decimal length	Places	COBOL description	FORTRAN description	BASIC description
	Char	1–9999	0	PIC X(L) ¹	CHARACTER* L ¹	STRINGX= L ¹
S	Binary	1	0	PIC X	Not Valid	Not Valid
U	Binary	1	0	PIC X	Not Valid	Not Valid
S	Binary	2	0	PIC S9(4) COMP	INTEGER*2	SHORT
U	Binary	2	0	PIC 9(4) COMP	INTEGER*2	SHORT
S	Binary	4	0	PIC S9(9) COMP	INTEGER*4	LONG
U	Binary	4	0	PIC 9(9) COMP	INTEGER*4	LONG
S	Binary	8	0	PIC S9(16) COMP	Not Valid	Not Valid
U	Binary	8	0	PIC 9(16) COMP	Not Valid	Not Valid

¹ L is the length of the data-item.

Signed/ unsigned	Type	Decimal length	Places	COBOL description	FORTTRAN description	BASIC description
S	Binary	2	1–4	PIC S9(B) ³ V9(D) ² COMP	Not Valid	Not Valid
U	Binary	2	1–4	PIC 9(CB) ³ V9(D) ² COMP	Not Valid	Not Valid
S	Binary	4	1–9	PIC S9(B) ³ V9(D) ² COMP	Not Valid	Not Valid
U	Binary	4	1–9	PIC 9(B) ³ V9(D) ² COMP	Not Valid	Not Valid
S	Binary	8	1–18	PIC S9(B) ³ V9(D) ² COMP	Not Valid	Not Valid
U	Binary	8	1–18	PIC 9(B) ³ V9(D) ² COMP	Not Valid	Not Valid
S	Numeric	1-18	0	PIC S9(B) ³	Not Valid	Not Valid
U	Numeric	1-18	0	PIC 9(B) ³	Not Valid	Not Valid
S	Numeric	1-18	1–L ¹	PIC S9(B) ³ V9(D) ²	Not Valid	Not Valid

¹ L is the length of the data-item.

² D is the number of the decimal places after the decimal.

³ For COBOL, B is the total number of digits before the decimal point.

For FORTRAN and BASIC, B is the total number of digits before and after the decimal.

Signed/ unsigned	Type	Decimal length	Places	COBOL descrip- tion	FORTTRAN description	BASIC description
U	Numeric	1–18	1– L ¹	PIC 9(B) ³ V9(D) ²	Not Valid	Not Valid
S	Packed Decimal	1–9	0	PIC 9(L) ¹ COMP-3	Not Valid	DECIMAL
S	Packed Decimal	1–9	1–($2L$ – 1) ¹	PIC 9(B) ³ V9(D) ² COMP-3	Not Valid	DECIMAL
S	Floating Point	4	0	COMP-1	REAL	SINGLE
S	Floating Point	8	0	COMP-2	DOUBLE PRECISION	DOUBLE

¹ L is the length of the data-item.

² D is the number of the decimal places after the decimal.

³ For COBOL, B is the total number of digits before the decimal point.

For FORTRAN and BASIC, B is the total number of digits before and after the decimal.

Connecting a domain to a data-item

During the Modify or Create functions, the Domain for Data-item screen shown below follows the logical data-item Comments screen. If you select Examine from the Logical Data-item Function menu, the name of any connected domain is displayed after the Comments screen.

```
CINCOM SYSTEMS  SUPRA DBA - DOMAIN FOR DATA-ITEM

      For CUSTOMER-NAME
1 : Connect domain
2 : Disconnect domain
3 : Display domain connected

      Enter choice no : 1

Enter domain name :
```

Enter domain name :

Description *Required.* Specifies the name of the domain you wish to connect to the logical data-item.

Format 1–30 alphanumeric character name of existing domain on the Directory.

Consideration The domain details such as length, data type, and so on, must exactly match those of the data-item to which you are connecting it. Any details that the domain and the data-item have in common are compared. If there is a mismatch, the two are not related and an error message is displayed.

Deleting a logical data-item

When you select Delete from the Logical Data-item Function menu, SUPRA DBA prompts for the name of the logical data-item. Enter the requested name and press RETURN. SUPRA DBA responds with a confirmation message.

Enter Y to delete the logical data-item or N to cancel the request. If you enter Y, a message indicating successful deletion displays. If unsuccessful, a message indicating the reason appears. Correct the error and retry the deletion. (The following screen uses CUSTOMER-NUMBER as an example.)

```
CINCOM SYSTEMS  SUPRA DBA - LOGICAL DATA-ITEM FUNCTION

      Functions for logical data-items
1 : Examine
2 : Modify
3 : Create
4 : Delete
5 : List logical data-items for data set

      Enter choice no.: 4

Delete logical data-item name : CUSTOMER-NUMBER

Confirm deletion of logical data-item CUSTOMER-NUMBER
(Y or N) : Y

CUSTOMER-NUMBER deleted successfully                                <PF1>
```

Defining domains

Domains are associated with data-items in the internal schema. The internal schema describes the physical record layout of your data, the location and structure of the physical files that hold the records, and the physical access method. A domain defines the format of data in detail enabling you to carry out comprehensive validation checks on user entered data.

A domain can define the following:

- ◆ Domain function (area, date, money, string, weight (not supported by RDM))
- ◆ Unit type (acre, feet/inches, square meter, cubic foot (not supported by RDM))
- ◆ Domain format (character, floating point, binary)
- ◆ Domain length
- ◆ Number of decimal places
- ◆ Signed or unsigned
- ◆ Null value
- ◆ Default value
- ◆ Pool of valid values
- ◆ Maximum/minimum range of values
- ◆ User-defined validation exit

You create and maintain domains through the Domain Function menu. One domain can be connected to one or more data-items, but a data-item can be connected to only one domain. Once you have created a domain, you can connect it to a data-item either through the Domain Functions menu or VMS through the Logical Data-item Function menu. After you have connected a domain to a data-item, you cannot change any characteristics of the physical data-item without first disconnecting the domain. If you change the characteristics of the domain, these will be transferred to any connected data-items.

A validation table contains a pool of acceptable values for a domain. All data-items connected to a domain that specifies a validation table become subject to that validation table. To make this connection, you specify a domain-validation-type of TABLE and supply the name of the validation table containing the pool of valid values in response to a prompt.

For example, suppose a company uses nine suppliers for a particular part. You can create a validation table containing only those suppliers. You then create a domain for the logical data-item SUPPLIER-NAME and connect it to the validation table containing the list of suppliers. RDM will reject any values entered for SUPPLIER-NAME that do not match those in the validation table. If the supplier is in the table, your order is processed.

Creating, examining, or modifying a domain

Select option 5, Domains, from the Function Selection for the DBA screen to display the Domain Function menu.

```
CINCOM SYSTEMS      SUPRA DBA - DOMAIN FUNCTION

                    Functions for domains :
                    1 : Examine
                    2 : Modify
                    3 : Create
                    4 : Delete
                    5 : List data-items connected
                    6 : Connect domain to data-item
                    7 : Disconnect domain from data-item

                    Enter choice no :
```

Options 1, 2, and 3 from this menu display the following screens in the same sequence:

- ◆ First screen: Domain details
- ◆ Second screen: Domain comments
- ◆ Third screen: Validation table used by domain

Defining and maintaining domain details

The Domain Details screen allows you to define and maintain the contents of a domain. Options 1, 2, and 3 from the Domain Function menu display this screen after first prompting you to specify the domain name.

```

CINCOM SYSTEMS                DOMAIN : MONTH

 1 : DOMAIN-NAME                :MONTH
 2 : DOMAIN-FUNCTION            :STRING
 3 : DOMAIN-UNIT                :N/A
 4 : DOMAIN-FORMAT              :NUMERIC
 5 : DOMAIN-LENGTH              :2
 6 : DOMAIN-DECIMALS            :0
 7 : DOMAIN-SIGNED              :SIGNED
 8 : DOMAIN-NULLS-ALLOWED      :NO
 9 : DOMAIN-NULL-VALUE         :
10 : DOMAIN-DEFAULT-VALUE      :
11 : DOMAIN-RETRIEVAL-VALIDATION :YES
12 : DOMAIN-VALIDATION-TYPE     :RANGE
13 : DOMAIN-MINIMUM-VALUE      :01
14 : DOMAIN-MAXIMUM-VALUE      :12
15 : DOMAIN-VALIDATION-EXIT-NAME :
16 : DOMAIN-STATUS              :O.K.

Enter field number
(or <PF1> to exit) :
    
```

DOMAIN-NAME

- Description** *Required.* The name of the domain, specified in response to a prompt on the Domain Function menu.
- Format** 1–30 alphanumeric characters

DOMAIN-FUNCTION

Description The category of data described by this domain.

Default STRING

Options A AREA
DA DATE
DI DISTANCE
M MONEY
N NUMBER
P PRESSURE
S STRING
TE TEMPERATURE
TI TIME
VE VELOCITY
VO VOLUME
W WEIGHT

Consideration You need enter only the abbreviation.

DOMAIN-UNIT

Description The unit in which the category of data is measured.

Options See “[Format of entries for DOMAIN-UNIT](#)” on page 383.

Considerations

- ◆ Must be a valid unit for the DOMAIN-FUNCTION specified.
- ◆ DOMAIN-UNIT must be blank if DOMAIN-FUNCTION is NUMBER or STRING.
- ◆ You need enter only the abbreviation.

DOMAIN-FORMAT

Description	The data type.
Default	CHARACTER
Options	B BINARY C CHARACTER F FLOATING POINT K KANJI L LEADING NUMERIC N NUMERIC P PACKED NUMERIC Z ZONED NUMERIC

Considerations

- ◆ Make sure the format is consistent with the function. For example, you would be unlikely to store a velocity as a character or money as floating point.
- ◆ The domain format (the data type) affects the DOMAIN-LENGTH field.
- ◆ You need enter only the abbreviation.

DOMAIN-LENGTH

Description Stored length of any data-item connected to this domain.

Options 1–4096

Considerations

- ◆ Depends on the value for DOMAIN-FORMAT as follows:

DOMAIN-FORMAT	DOMAIN-LENGTH
B	1, 2, 4, or 8
C	1–4096
F	4 or 8
K	1–4096
L	1–18
N	1–18
P	1–10
Z	1–18

- ◆ The DOMAIN-LENGTH must exactly match the length of any data-items to which it is connected. If the lengths are incompatible, DBA displays an error message when you try to connect the domain to the data-item and does not allow you to make the connection.
- ◆ If DOMAIN-FORMAT is “C”, the DOMAIN-LENGTH may be 0 but will take on the length of the first data-item connected to it. For any other DOMAIN-FORMAT, you cannot define a DOMAIN-LENGTH of 0.

DOMAIN-DECIMALS

- Description** *Required*, but may remain at 0 if value for DOMAIN-FORMAT is BINARY, NUMERIC, or PACKED NUMERIC. Indicates the number of digits after the decimal point.
- Default** 0
- Options** Depends on the values for DOMAIN-FORMAT and DOMAIN-LENGTH as follows:

DOMAIN-FORMAT	DOMAIN-LENGTH	DOMAIN-DECIMALS
B, Binary	1	Up to 2 decimal places
	2	Up to 4 decimal places
	4	Up to 9 decimal places
	8	Up to 18 decimal places
C, Character	1–4096	0
F, Floating Point	4 or 8	0
K, Kanji	1–4096	0
L, Leading Numeric	1–18	Up to 17 decimal places
N, Numeric	1–18	Up to 18 decimal places
P, Packed Numeric	1–10	Maximum 19 decimal places
Z, Zoned Numeric	1–18	Up to 18 decimal places

DOMAIN-SIGNED

- Description** *Required* if the DOMAIN-FORMAT is B, N, or P.
- Default** SIGNED
- Options** S SIGNED
U UNSIGNED

Considerations

- ◆ Must match the sign of the data-item to which the domain is connected.
- ◆ Must be SIGNED if the domain format is BINARY.

DOMAIN-NULLS-ALLOWED

Description *Optional.* Specifies whether the data-item may contain a null value.

Default N

Options Y Null values allowed

 N No null values allowed

Considerations

- ◆ A null value indicates that a field is empty.
- ◆ If you allow null values, you must specify the value that represents a null in the DOMAIN-NULL-VALUE field.
- ◆ If the DOMAIN-FORMAT is floating point, you must enter a valid value.
- ◆ If the DOMAIN-FORMAT is anything other than floating point, DBA displays a WARNING if you enter an invalid value, but still allows you to proceed.

DOMAIN-NULL-VALUE

Description *Required* if DOMAIN-NULLS-ALLOWED is Y.

Format Must be a valid value for the data-items associated with this domain.

Considerations

- ◆ Usually null values are represented with blanks; however, you can define any other character as representing a null value.
- ◆ **VMS** RDM gives null numeric and packed data-items a null value of blanks even if you do not specify a null value.
- ◆ **VMS** An application program inserts a null value by changing the ASI for the data-item to "N." A DBAID user inserts a null value by entering the keyword "NULL."

DOMAIN-DEFAULT-VALUE

Description *Optional.* Specifies a default.

Format Must be a valid value for the data-items associated with this domain.

Considerations

- ◆ The default value is used only if the column to which it applies is omitted from a derived view or user view. The default is not used if the column is simply left blank by the user; the user may have left the column blank intentionally (an address-line which is not required for a very short address).
- ◆ If the DOMAIN-FORMAT is floating point, you must enter a valid value.
- ◆ If the DOMAIN-FORMAT is anything other than floating point, DBA displays a warning if you enter an invalid value but still allows you to proceed.

DOMAIN-RETRIEVAL-VALIDATION

Description *Required.* Specifies whether RDM will perform validation during GET.

Default NO

Options Y

N

Consideration If this field is Y, you must further specify the type of validation through the next four fields.

DOMAIN-VALIDATION-TYPE

Description *Required.* If DOMAIN-RETRIEVAL-VALIDATION was “Y.”

Default NONE

Options R RANGE

T TABLE

E EXIT

N NONE

Considerations

- ◆ If you specify RANGE, you must enter the minimum and maximum permitted values in fields 13 and 14.
- ◆ If you specify TABLE, DBA prompts you to enter the name of the validation table after you have completed this screen.
- ◆ If you specify EXIT, you must supply the name of the validation exit in field 15.

DOMAIN-MINIMUM-VALUE

Description *Required.* If DOMAIN-VALIDATION-TYPE is RANGE. Specifies the minimum acceptable value.

Format Must be a valid value for the data-items associated with this domain.

Considerations

- ◆ If the DOMAIN-FORMAT is floating point, you must enter a valid value.
- ◆ If the DOMAIN-FORMAT is anything other than floating point, DBA displays a warning if you enter an invalid value but still allows you to proceed.

DOMAIN-MAXIMUM-VALUE

Description *Required.* If DOMAIN-VALIDATION-TYPE is RANGE. Specifies the maximum acceptable value.

Format Must be a valid value for the data-items associated with this domain.

Considerations

- ◆ If the DOMAIN-FORMAT is floating point, you must enter a valid value.
- ◆ If the DOMAIN-FORMAT is anything other than floating point, DBA displays a warning if you enter an invalid value but still allows you to proceed.
- ◆ This value must be greater than DOMAIN-MINIMUM-VALUE.

DOMAIN-VALIDATION-EXIT-NAME

Description *Required.* If DOMAIN-VALIDATION-TYPE is EXIT.

Format 8 alphanumeric characters

Consideration Specify the name of the entry point into the validation exit module, not the VMS file name.

DOMAIN-STATUS

Description Displays the current status of the domain.

Options O.K.

BEING MODIFIED

General considerations

- ◆ If DBA changes the length of a data-item during database validation, it attempts to disconnect any connected domain in order to prevent inconsistencies. If the domain is successfully disconnected, DBA displays a warning message. If DBA cannot disconnect a domain, it displays an error message and database validation is unsuccessful.
- ◆ Do not connect data-items to domains until after you validate the database to which they are related. The length of physical data-items is not maintained dynamically during data-item maintenance. Therefore, the length of data-items that have subdata-items may not be correct until after validation.
- ◆ Certain attributes of a data-item connected to a domain cannot be modified without first disconnecting the domain. If you select a data-item, which has a domain from any of the following screens:
 - Modify primary data-items
 - Modify related data-items
 - Modify coded data-items
 - Modify RMS data-items

you cannot modify the length. The prompts for “filler” and “comments” are still displayed, but data-item length and logical data-item type are inaccessible until you disconnect the domain. Selecting a data-item by number will display the subdata-item screen, as usual, but it will not allow you to ADD or DELETE a subdata-item because that would affect the length.

- ◆ During RMS key maintenance, the key detail screen is followed by the data-item details screen. However, if the data-item is connected to a domain, then this screen is set to read-only and the following message displayed:

```
Note that (data-item) is used by domain (domain)
```
- ◆ You can use the RDM Domain Usage report (refer to the *SUPRA Server PDM RDM Administration Guide (VMS)*, P25-8220) to keep track of the connections between domains and physical data-items.

Entering domain comments

Once you have specified the domain details, DBA displays the domain Comments screen. You can enter comments on this screen using the procedures described in “Using SUPRA DBA” on page 61.

Deleting a domain

Select option 4 from the Domain Function menu to delete a domain. You are prompted for the name of the domain to be deleted and then for confirmation of the delete. All associated data-items and validation tables are disconnected but not deleted.

Connecting a domain to a validation table

If you enter “TABLE” in the DOMAIN-VALIDATION-TYPE field, DBA displays different prompts according to the task you are performing. If you are examining a domain that is connected to a validation table, DBA displays the name of the validation table after the domain Comments screen. If you are creating a new domain and enter “TABLE” in the DOMAIN-VALIDATION-TYPE field, DBA prompts you to specify the name of the table as follows:

Enter validation table name:

If you are modifying a domain and the DOMAIN-VALIDATION-TYPE is TABLE, DBA displays these prompts after the domain Comments screen:

```
CINCOM SYSTEMS   SUPRA DBA - DOMAIN USES VALIDATION TABLE   15-Jun 99 13:23
```

```
Validation table for domain (domain-name)
is : (validation-table-name)
```

```
Do you wish to change it ? (Y/N) :
```

```
Enter validation table name :
```

**Validation table for domain (*domain-name*)
is :(*validation-table-name*)**

Description Displays the name of the validation table and the domain to which it is connected, if any.

Do you wish to change it ? (Y/N) :

Description *Optional.* Indicates whether you wish to change the connection between domain and validation table.

Options Y Connect a different validation table to the specified domain.
N Leave the connection as it is.

Considerations

- ◆ If you enter “Y,” you are prompted to specify the name of an alternative validation table.
- ◆ If you press function key PF3 without entering a validation table name, the previous connection is used if there is one. If there is no previous connection between the domain and a validation table, pressing function key PF1 sets DOMAIN-VALIDATION-TYPE to N, no validation.

Enter validation table name :

Description *Required.* If your answer to the previous prompt was “Y.” Connects a validation table to the specified domain.

Format 1–30 alphanumeric characters

Considerations

- ◆ Press function key PF3 to display a list of available validation tables.
- ◆ If you enter a validation table name that does not exist, DBA displays an error message and prompts you to enter a different validation table.
- ◆ If you press function key PF1 without entering a validation table name, the previous connection is used if there is one. If there is no previous connection between the domain and a validation table, pressing function key PF1 sets DOMAIN-VALIDATION-TYPE to N, no validation.

Connecting a domain to a data-item

A domain can exist in isolation or connected to one or more data-items. You can make the connection either through the Logical Data-item Function menu or through the Domain Function menu. The end result is the same except that if you make the connection from the Domain Function menu, the data-item details will be overwritten with the domain details, if the two differ. If you connect from the Logical Data-item Function menu, you must make sure both the data-item details and the domain details match exactly; if they differ, the connection will not be made.

Select option 6 to connect a domain to a data-item. DBA first prompts you to enter the name of the domain you wish to connect; then enter the physical name of the data-item to which you wish to connect it.

```
CINCOM SYSTEMS      SUPRA DBA - DOMAIN FUNCTION

      Functions for domains :
1 : Examine
2 : Modify
3 : Create
4 : Delete
5 : List data-items connected
6 : Connect domain to data-item
7 : Disconnect domain from data-item

      Enter choice no : 6

Enter domain name : BRANCH-NUMBERS

Enter physical data-item information

Data set name : BRAN

Four-character data-item name : CTRL
```

Enter domain name :

Description *Required.* Specifies the name of the domain you wish to connect.

Format 1–30 alphanumeric character name of an existing domain

Consideration If the domain length is 0, it is set to the length of the data-item to which it is being connected.

Enter physical data-item information**Data set name:**

Description *Required.* Specifies the name of the data set containing the physical data-item that you wish to connect to the domain.

Format 4 alphanumeric character name of existing data set on the Directory

Four-character data-item name:

Description *Required.* Specifies the name of the physical data-item you wish to connect to the domain.

Format 4 alphanumeric character name of existing data-item on the Directory

Consideration The data-item must *not* be connected to any other domain. If any such connection already exists, the domain is not connected and an error message is displayed.

General considerations

- ◆ Any details that the domain and the data-item have in common are compared. If the lengths do not match, the two are not related and an error message is displayed. If any other details do not match, a warning message indicates that the domain details will overwrite the data-item details and a confirmation prompt is displayed.
- ◆ Because the domain and data-item have several details in common, there are constraints on the modification of these details. The affected details are:
 - LENGTH
 - TYPE
 - DECIMAL PLACES
 - SIGN

These are the constraints:

- When you connect a data-item to a domain, the length of the data-item and domain must match; otherwise, DBA does not allow the connection.
- When you connect a data-item to a domain through the Domain Function menu, and the values for TYPE, DECIMAL PLACES or SIGN do not match, DBA transfers the values used in the domain to the data-item and displays a warning. You are prompted to confirm the connection between the data-item and domain.
- When you modify a domain, the new values for the details cascade into any connected data-items. Again, DBA displays a warning and offers you the chance to confirm. This is particularly important if you change the data-item length because you must then recompile the physical database and unload and reload the changed data set(s) for the data-item(s).
- When you modify a data-item, you cannot change the length or any other details unless you first disconnect any domain.

Defining validation tables

Select option 6, Validation tables, from the Function Selection for the DBA menu to display the Validation Table Function menu shown below:

```
CINCOM SYSTEMS      SUPRA DBA - VALIDATION TABLE FUNCTION

      Functions for validation tables :
1 : Examine
2 : Modify
3 : Create
4 : Delete
5 : List domains using validation table

      Enter choice no :
```

Options 1, 2, and 3 from this menu display the following screens in the same sequence:

- ◆ First screen: Validation table text (see “[Entering validation table text](#)” on page 350).
- ◆ Second screen: Validation table comments (see “[Entering validation table comments](#)” on page 351).

Entering validation table text

The Validation Table screen shown below allows you to define and maintain the text of a validation table. DBA displays this screen in response to options 1, 2, or 3 from the Validation Table Function menu, after first prompting you to specify the validation table name.

```
CINCOM SYSTEMS          VALIDATION TABLE for TITLES

 1  SAG
 2  DBAG
 3  Planning Guide
 4  Digest
 5  Programming Guide
 6  Utilities Guide
 7  Messages and Codes
 8  Migration Guide
 9  RDM Guide
10  DBAID Reference Card
11  SPECTRA Reference Card
12  SPECTRA User's Guide
13  Tutorial
14  Glossary
15  Master Index

action : C,L,N,O,P or W - Hit <PF2> for explanation

:
```

You enter values into the Validation Table as you would enter comments on a standard DBA comment screen. “Using SUPRA DBA” on page 61 describes these procedures; alternatively, press function key PF2 to display Help text.

Each entry in the validation table must match the format of the data-item or data-items to which it corresponds. To optimize performance, always enter the most popular values first in the table.

Entering validation table comments

Once you specify the validation table text, DBA displays the validation table Comments screen. You can enter comments on this screen using the procedures described in “[Entering comments](#)” on page 96.

Deleting a validation table

Select option 4 from the Validation Table Function menu to delete a validation table. You are first prompted for the name of the validation table and then to confirm the delete. DBA displays an error message if the validation table you specified is still connected to one or more domains and does not proceed with the delete. You must return to the Domain Function menu and disconnect the domain from the validation table. To do this, either change the DOMAIN-VALIDATION-TYPE to something other than T (TABLE) or connect a different validation table.

Defining views (VMS)

You define your base views after you create, validate, compile and format your database description, and after you have formatted, populated and activated any indices and created all the logical data-item names you need to access the database through RDM. See “[Creating and maintaining a database description](#)” on page 101 for procedures for creating a database and see “[Validating, compiling, printing, and formatting a database](#)” on page 229 for procedures to validate, compile, and format a database.

This section describes how to use the SUPRA DBA interface EDIT/EDT to create and manipulate view text. It does not explain the view syntax. Refer to the [SUPRA Server PDM RDM Administration Guide \(VMS\)](#), P25-8220, for the rules and considerations involved in designing base and derived views.

To invoke the EDIT/EDT interface, you must first select the Logical views function from the Function Selection for the DBA menu (this displays the Logical View Function menu). When you select the Examine, Modify or Create function from the Logical View Function menu, SUPRA DBA invokes EDIT/EDT and places the cursor in an EDT buffer called =MAIN. If you select the Create function, this buffer is empty awaiting the text of the new view. With either the Examine or Modify function, the buffer contains both the column definition and the access definition of the existing view.

The following are the Logical View Maintenance functions:

- ◆ **Examine** Displays the view details such as the logical data-items within the view, the navigation definition and comments.
- ◆ **Modify** Changes the logical data-items within the view, the navigation definition or comments.
- ◆ **Create** Adds a new view and specifies the logical data-items, view details, and comments to be included (see “[Creating and maintaining views through DBA \(VMS\)](#)” on page 355).
- ◆ **Delete** Deletes a view including attributes and comments (see “[Deleting a view \(VMS\)](#)” on page 369). The associated logical data-items are disconnected but not deleted.
- ◆ **Connect to database description** Connects a view to a database description. This function requires a view name and a database description name.
- ◆ **Disconnect from database description** Disconnects a view from a database description. This function requires a view name and a database description name.
- ◆ **List database descriptions using view** Lists all database descriptions using a view. This function requires a view name.
- ◆ **List all logical views** Lists all views on the Directory.
- ◆ **User Authorization** Specifies which users can access the view (see “[Defining user authorization \(VMS\)](#)” on page 370).

Before you select Logical View Maintenance from within SUPRA DBA, you must have access to an EDT startup file with the logical name EDTINI and to the command file DBAEDT.EDT. Access to the EDT startup file and the command file DBAEDT.EDT depends on the following:

- ◆ The logical name EDTINI might already be assigned to a file containing commands to customize your EDT environment. However, you will not necessarily have an EDTINI file since EDT can operate without it.
- ◆ The command file DBAEDT.EDT is provided by Cincom and contains commands to customize the user-environment to suit the needs of view maintenance. You MUST have access to this file.

How you access the DBAEDT.EDT file depends upon whether you have an EDTINI file.

- ◆ If you have an EDTINI file, then you must either assign the logical name DBAEDT to DBAEDT.EDT or place DBAEDT.EDT in the VMS directory from which you run DBA. If you use the logical name DBAEDT, you may give the file any name you want. If you rely on the file being present in your default VMS directory, the file must be called DBAEDT.EDT.
- ◆ If you have no EDTINI file, then you must assign the logical name EDTINI to DBAEDT.EDT. The logical name DBAEDT will not then be necessary.
- ◆ If you have both an EDTINI file and a DBAEDT file, the DBAEDT file is executed immediately after the EDTINI file. In this way, you can maintain your usual editing environment while adding the special commands needed to use EDT for view maintenance.

Creating and maintaining views through DBA (VMS)

The prompts displayed when you create a view are the same as those displayed when you modify a view. The following description explains how to modify an existing view. Follow the same procedure to create a view. When you create a view, you are presented with an empty buffer in which to enter the view text.

To modify a view, select option 3, Logical Views, from the Function Selection for the DBA menu to display the Logical View Function menu as shown below. Select option 2, Modify, and respond to the prompts.

```
CINCOM SYSTEMS      SUPRA DBA - LOGICAL VIEW FUNCTION

      Functions for logical views
1 : Examine
2 : Modify
3 : Create
4 : Delete
5 : Connect to database description
6 : Disconnect from database description
7 : List database descriptions using view
8 : List all logical views
9 : User authorization

      Enter choice no.: 2

Modify logical view name :
(<PF4> will select CUSTOMER) : CUSTOMER-REFERENCE

Database to which logical view refers : CUSTDB
```

Modify logical view name :

Description *Required.* Identifies the view to be modified or created.

Format 1–30 alphanumeric characters and hyphens. The first character must be alphabetic. The last character must not be a hyphen. Hyphens cannot be consecutive characters.

Consideration To modify a view, the view must already exist on the Directory.

Database to which logical view refers :

Description *Required.* Identifies the database to which this view is connected.

Format 6 alphanumeric characters with no embedded spaces. The first character must be alphabetic.

Consideration The database must exist on the Directory in a validated, compiled, and formatted state with formatted task and system logs, if defined. DBA view maintenance signs on to RDM before allowing you to create or modify a view. This ensures that only valid, opened views are saved on the Directory. You cannot sign on to RDM unless a compiled database description file is available with formatted task and system logs where defined.

The following RDM message is displayed to indicate that you have specified a valid view name and database name, and that EDT has been invoked successfully.

```
FSI * VSI: = MSG: SUCCESSFUL COMPLETION- LEVEL 05 <PF1/2>
```

Press function key PF1 to display the EDT buffer containing the view to be maintained.

```
KEY CUSTOMER-NO
CUSTOMER-CR-CODE
CUSTOMER-CR-LIM
REQ CUSTOMER-BRANCH = CUSTOMER-BRANCH = CUSTOMER=NO
CUSTOMER-ADDR
CUSTOMER-NAME
CUSTOMER-CLASS
ACCESS CUST WHERE CUSTOMER-NO = CUSTOMER-NO ALLOW ALL
* To verify that CUSTOMER-BRANCH contains a valid branch on INSERT
* and update
ACCESS BRAN ONCE WHERE BRANCH-NO = CUSTOMER-BRANCH
[EOB]
```

Using EDT to edit views (VMS)

You use the same procedures when you edit a view through DBA as when you edit a file using EDT. EDT function-key sequences are tied to the same functions with five exceptions. These exceptions are defined in the DBAEDT.EDT file and will apply only if you have access to the DBAEDT file as described in “Defining views (VMS)” on page 352. If you have customized the DBAEDT file in any way, then the following descriptions will apply only in part:

- ◆ The screen width is restricted to 72 characters, and automatic wrap-around is enabled to prevent you from entering lines that exceed the maximum permitted length for view text.
- ◆ Function key PF1 followed by S attempts to open the view and, if successful, saves it on the Directory.
- ◆ Function key PF1 followed by B attempts to open and save the view and, if successful, binds the view.
- ◆ Function key PF1 followed by L prompts you to enter the name of a data set and copies all of the logical data-item names associated with that data set into an EDT buffer of the same name (=PROD for the PROD data set, =CUST for the CUST data set).
- ◆ Function key PF2 does not display EDT Help text. Press function key PF1 followed by keypad 7 to display the “command:” prompt and type HELP to access the general DBAID Help text.



HELP displays general help; HELP (topic) displays help on a particular topic.

Throughout the following description, you can replace the keystroke sequence function key PF1 followed by keypad 7 with the keystroke sequence CTRL-Z. Both allow you to issue line-mode commands. CTRL-Z sets line mode (displaying the asterisk prompt “*”); you can issue a series of commands and must type C and press RETURN to get back to screen mode when you are finished. Function key PF1 followed by keypad 7 sets command mode (displaying the prompt “command:”) without leaving screen mode. You can then issue one command at a time, returning automatically to screen mode after the command has executed. To return to screen mode without executing a command, type C and press ENTER.

You must press ENTER to transmit all commands issued from command mode (function key PF1 and keypad 7). Press RETURN to transmit all commands issued from line mode (CTRL-Z).

Displaying line numbers (VMS)

When you edit a view using EDT, no line numbers display; however, RDM assigns line numbers to the view. If RDM returns a validation error when you save the view, the error message displays the line number that caused the error. To review the line that caused the problem, press function key PF1 followed by keypad 7 and type the number of the line you wish to see.

Using EDT buffers (VMS)

When you initiate the EDT session, the session sets the default buffer =MAIN and loads a copy of the view text into this buffer, displaying it on your screen. EDT creates one other buffer, =PASTE, to hold text removed using the EDT APPEND, CUT and PASTE commands. To change buffers, press function key PF1 followed by keypad 7 to switch to line mode and then type:

```
Command: =(buffer-name)
```

where (buffer-name) specifies the buffer you want to display. This buffer may already exist, for instance =PASTE, or it may be a new buffer. When you execute this command, EDT displays the contents of the buffer you specify.

To list all of the buffers available during your current EDT session, press function key PF1 followed by keypad 7 and type SHOW BUFFER. The buffer with the equal symbol to its left is the current buffer. EDT buffer usage is described in detail in Digital's *VMS EDT Reference Manual*.

The following are a few hints for using the buffering facilities when editing views through DBA:

- ◆ **Using function key PF1 followed by L.** Function key PF1 followed by L prompts you to enter a data set name as follows:

```
Logical data-items for data set :
```

Type the 4-character data set name and press RETURN. EDT then lists the logical data-items connected to that data set in a buffer and gives the buffer the same name as the data set you specified (e.g., =PROD). EDT sets this new buffer. To return to the buffer containing the view text, press function key PF1 followed by keypad 7 and type =MAIN.

- ◆ **Using COPY.** You can copy the logical data-item names into your main buffer using the EDT command:

```
COPY =buffer-name-1 range TO =buffer-name-2 line-no
```

If you do not specify a target buffer (=buffer-name-2), EDT copies the contents of =buffer-name-1 to your current buffer. Therefore, if you are looking at the contents of =MAIN (your view text) and you wish to copy the first three logical data-items from the buffer =PROD, enter:

```
COPY =PROD 1 thru 3
```

EDT places the first three lines of =PROD at the beginning of =MAIN, your current buffer.

If you wish to specify precisely where to place the copied lines in =MAIN, you must specify both buffer-name-2 and line-no. For example:

```
COPY =PROD 6 THRU 11 TO =MAIN 8
```

This command copies lines 6 to 11, inclusive, from the buffer PROD to before line 8 in =MAIN.

Creating text files (VMS)

You can create VMS text files from within EDT and include these files during a subsequent EDT session. This feature is particularly useful if you wish to use the text of one view as the basis for another view.

Using WRITE or EXIT to create a VMS text file. To create a VMS text file of the view you are currently editing, press function key PF1 followed by keypad 7 to get to command mode; then type either:

```
WRITE (vms-file-specification)
```

or

```
EXIT (vms-file-specification)
```

This writes the contents of your current buffer to a VMS text file. These commands, in combination with the INCLUDE command, perform the same function as the DBAID COPY command allowing you to use the text from one view in another view.



If the (*vms-file-specification*) exactly matches the view name, the effect is the same as pressing function key PF1 followed by S.

Using INCLUDE to copy the contents of a VMS text file. You can use the EDT INCLUDE command and copy the contents of VMS text files into an EDT buffer. You can then use all or part of those files in your view definition.

You issue the INCLUDE command from line mode. Press function key PF1 followed by keypad 7 to switch to line mode. The format of the INCLUDE command is as follows:

```
INCLUDE vms-file-specification [=buffer] [line-no]
```

where:

vms-file-specification is the name of the external file you wish to copy into your buffer

[=buffer] [line-no] is the name of the buffer and position to which you want to copy the text. Both =buffer and line-no are optional. If you do not specify a target buffer, EDT copies the external file into your current buffer.

This example illustrates how to copy an external file into a named buffer:

```
INCLUDE [CLOUGH.DB]DATA-ITEMS.LIS =LIST
```

copies the contents of the VMS file DATA-ITEMS.LIS into a buffer called =LIST. If this buffer does not already exist, EDT creates it.

You can use the INCLUDE command to copy the text from one view into another view. You do this in five steps:

1. Select Modify from the Logical View Function menu and specify the name of the view you wish to copy.
2. Use either the WRITE command or the EXIT command to create a text file containing the text of that view.
3. Exit EDT (described in “[Exiting view maintenance \(VMS\)](#)” on page 364) without saving any changes.
4. Select Create from the Logical View Function menu.
5. Use the INCLUDE command to copy the text of the first view into your current buffer.

You can then edit the new view as you wish. You can also use the above procedure to copy view text into existing views.

Saving views (VMS)

When you finish editing your view, press function key PF1 followed by S to save it on the Directory. Before saving the view, RDM attempts to open it. Any error messages displayed at this point are the same as those that might be issued by a DBAID OPEN or a DBAID SAVE. If RDM can open the view, it displays the following message:

```
FSI: * VSI: = MSG: nnn BYTES USED IN OPENING VIEW <PF1>
```

At this point, press function key PF1. If RDM can save the view, it displays the following message:

```
SAVED VIEW (view-name)
```

After the open and save operation is finished, whether it is successful or not, you are returned to the EDT session.

An alternative method is to press function key PF1 followed by keypad 7 and type WRITE *view-name*. Provided you type the *view-name* exactly as it is stored on the Directory (with no suffix and with all the right characters), this key sequence will perform the same function as function key PF1 followed by S, returning you to EDT when finished. If you accidentally type a name that does not match the view name, EDT will create a file of that name in your current VMS directory.

You can also press function key PF1 followed by keypad 7 and then type EXIT (with no parameters) to save your view on the Directory. However, this method exits you from EDT and returns to the Logical View Function menu when the view save is completed successfully.

If you attempt to save a view that was previously bound, the following prompt is displayed:

```
VIEW IS BOUND. DO YOU WISH TO REBIND IT?
```

Enter "Y" to rebind the view. Any other response will only save the view.

To exit the EDT session after you save the view, press function key PF1 followed by keypad 7 and type QUIT. You return to the Logical View Function Menu.

Binding views (VMS)

You can bind a view through DBA by pressing function key PF1 followed by B. This causes RDM to attempt to open, save, and bind the view you are currently editing. The view bind takes place only if the previous two operations were successful. If RDM successfully binds the view, it displays the following message:

```
VIEW BINDING SUCCESSFUL
```

Exiting view maintenance (VMS)

You always return to the EDT editor after you save or bind the view, whether the save or bind was successful or unsuccessful. To exit EDT, press function key PF1 followed by keypad 7 and type QUIT. If you do not save the view text before you quit, it remains the same as it was when you began the EDT session.

Printing views (VMS)

You can print the following two types of information about views:

- ◆ The user view itself
- ◆ Reports containing details about the view (available in VAX environments only)

Printing a user view. You can print a user view from DBA and from DBAID.

To print a user view from DBA, perform the following:

1. Select option 3, Logical views, from the Function Selection for the DBA menu.
2. Select option 1, Examine, from the Logical View Function screen as shown below:

```

CINCOM SYSTEMS      SUPRA DBA - LOGICAL VIEW FUNCTION

                    Functions for logical views

1 : Examine
2 : Modify
3 : Create
4 : Delete
5 : Connect to database description
6 : Disconnect from database description
7 : List database descriptions using view
8 : List all logical views
9 : User authorization

Enter choice no.: 1

Examine logical view name : CUSTOMER

```

3. Printing a User View Enter the name of the view you want to print at the prompt. (This displays the view on-screen.)
4. Enter CTRL-Z to get the * prompt.
5. Type EXIT and press RETURN. (This creates a file called *view-name.EDT*.)
6. Enter PRINT *view-name.EDT* at the DCL prompt.

To print a user view using DBAID, perform the following:

1. Type \$DBAID *dbmod-name* to invoke DBAID.
2. Sign on with your user name and password.
3. Type LIST *view-name* at the prompt. (This makes the text of the view known to DBAID.)
4. Type EDT * at the prompt.
5. Enter CTRL-Z to get the * prompt.
6. Type EXIT and press RETURN. (This creates a file called *view-name.EDT*.)
7. Sign off from DBAID by typing BYE.
8. Enter PRINT *view-name.EDT*.

Printing reports containing details about a view (VAX environments only). Use the information in Appendix A of the *SUPRA Server PDM RDM Administration Guide (VMS)*, P25-8220, to print various reports about a view. These reports provide information such as details about data-items in a view, domain usage and logical data-item cross-references.

Customizing your EDT environment (VMS)

The following example shows the contents of DBAEDT.EDT as supplied by Cincom. The characters CTRL-Z indicate you should press the control key and Z together. If you type the file on a VT1 series terminal, this key sequence is represented as an information block; on a VT2 series terminal, it is represented as a backward-facing question mark.

```
SET SCREEN 72
SET WRAP 72
SET HELP SUPRA-HELP
DEFINE KEY GOLD B AS "XLATE-BIND(CTRL Z)"
DEFINE KEY GOLD S AS "XLATE-SAVE(CTRL Z)."
DEFINE KEY GOLD L AS "XLATE? 'Logical data-items for data set
: '(CTRL Z)."
SET MODE CHANGE
```

SET SCREEN 72 sets the screen width to 72 characters.

SET WRAP 72 sets the automatic wrap-around facility to 72 characters. The maximum view line length is 76 characters, but 4 characters are reserved for the line number and space before the text. The line number and space are not displayed during an EDT session. EDT allows you to define a view containing lines that are shorter or longer than the 72-character limit but when you attempt to save the view, DBA pads shorter lines with spaces and truncates longer lines. The screen width and automatic wrap-around are set to 72 to remind you of the maximum line length. Both are, however, optional.

SET HELP SUPRA-HELP invokes the DBAID help library instead of the EDT help library. This means that function key PF2 displays the message "No help available for this key"; you must press the space bar to return to the editing screen. To display the DBAID help during view maintenance, press function key PF1 followed by keypad 7 and type HELP or HELP (topic). You can change the help library, or omit it altogether to invoke the EDT standard help. Which help library (DBAID or EDT) you prefer depends upon whether you are more familiar with how to define views or how to use EDT.

DEFINE KEY GOLD B AS “XLATE-BIND(CTRL-Z).” sets the key sequence function key PF1 followed by B to save and bind the view you are currently editing. You can redefine the key sequence but do not remove the key definition. For example, to redefine the BIND function to the keys function key PF1 followed by V, you enter:

```
DEFINE KEY GOLD W AS "XLATE-BIND(CTRL Z)."
```

DEFINE KEY GOLD S AS “XLATE-SAVE(CTRL-Z).” sets the key sequence function key PF1 followed by S to save the view you are currently editing without binding it. You can redefine the key sequence but do not remove the key definition. For example, to redefine the SAVE function to the keys function key PF1 followed by W you enter:

```
DEFINE KEY GOLD V AS "XLATE-SAVE(CTRL Z)."
```

DEFINE KEY GOLD L AS “XLATE? 'Logical data-items for data set :'(CTRL-Z).” sets the key sequence function key PF1 followed by L to prompt for the name of a data set, and then display a list of the logical data-items available in that data set. This list is displayed in another EDT buffer, given the same name as the data set, for example, =CUST. You can redefine the key sequence as described above, or change the prompt. For example, to redefine the key sequence to function key PF1 followed by Q and change the prompt to “Enter data set name :”, you enter:

```
DEFINE KEY GOLD Q AS "XLATE? 'Enter data set name :'(CTRL Z)."
```

You can change only the characters between the single quotes.

Deleting a view (VMS)

When you select Delete from the Logical View Function menu, SUPRA DBA prompts for the view name. After you enter the name, SUPRA DBA displays a message for confirmation before deleting the view.

Press Y to delete the view or N to cancel the request. If you delete a view, the logical data-items it used remain on the system for use in other views. This example deletes the CUSTOMER-REFERENCE view.

```
CINCOM SYSTEMS      SUPRA DBA - LOGICAL VIEW FUNCTION

      Functions for logical views
1 : Examine
2 : Modify
3 : Create
4 : Delete
5 : Connect to database description
6 : Disconnect from database description
7 : List database descriptions using view
8 : List all logical views
9 : User authorization

      Enter choice no.: 4

Delete logical view name :
(<PF4> will select CUSTOMER) : CUSTOMER-REFERENCE

Confirm deletion of CUSTOMER-REFERENCE (Y or N) : Y
```

Defining user authorization (VMS)

Before a user can access a view, the user name must be related to that view. When you select User Authorization from the Logical View Function menu, SUPRA DBA prompts you for the name of the view you are maintaining; then it presents a list of the current users for that view. The user who created the view is included on this list.

To create the relationship between a user and a view, type A (Add) after the prompt and enter the new names. The user names must already be defined on the system (see “[Defining users](#)” on page 285). To remove a user-to-view relationship, type D (Delete) after the prompt. SUPRA Server then requests the number of the user whose relationship to the view is to be deleted.

```
CINCOM SYSTEMS                CUSTOMER-ORDERS

Users of the view
  1 : SUPRA-DBA
  2 : DICK-G
  3 : STEPH-S
  4 :

Specify function (Allow, Disallow), <PF4> to list or <PF1> to exit:
```

A

Minimum and maximum values

VMS environments

The following sections list the minimum and maximum values for SUPRA Server PDM support in VMS environments.

Database descriptions

Maximum concurrent tasks	1–1000 (Default=10)
Maximum update tasks	0–1000 (Default=10)
Access methods	QIO or RMS (Default – QIO)
Maximum number of data sets (files)	1024 per database
File types supported	SUPRA Server primary and related, and RMS
Cluster/network support	Single- or multi-machine environment (Default, single-machine (local))
Maximum number of databases on a SUPRA Server Directory	Unlimited
Recovery methods	Shadow logging Task logging System logging Recovery Unit Journaling for RMS data sets

SUPRA Server physical data manager

Maximum concurrent PDM tasks	1–1000 (Default=50)
Maximum overlapping threads per database	1–100 (Default=3)
Maximum number of PDM operator terminals	Unlimited
Maximum size of PDM message buffer	32767 (Default=4096)
Maximum number of loaded databases	Unlimited

SUPRA Server primary files

Total number of logical records	2–2,147,483,647 (Default calculated during validation)
Logical records per block	1–32,767 (Default=10)
Maximum primary record length	4096 bytes
Minimum primary record length	21 bytes
Maximum number of data items (fields)	4088
Maximum data item (field) length	4088 bytes
Maximum length of the control key	255 bytes
Maximum number of primary keys	1
Maximum number of linkpaths	510
Number of copies of a buffer	1–32767 (Default=5)
Maximum buffer size	32256

SUPRA Server related files

Total number of logical records	2–2,147,483,647 (Default calculated during validation)
Logical records per block	1–32,767 (Default=10)
Maximum record length	4096 bytes
Minimum record length	41 bytes
Maximum number of data items (fields)	4088
Maximum data item (field) length	4088 bytes
Maximum number of coded records	36 squared
Maximum length of the related key	255 bytes
Maximum number of related keys	454
Minimum number of related keys	1
Maximum number of linkpaths	511
Minimum number of linkpaths	1
Number of copies of a buffer	1–32767 (Default=5)
Maximum buffer size	32556

SUPRA Server index files

Maximum number of indexes per database	Unlimited
Maximum number of secondary keys per index	80
Maximum number of data items per secondary	251

RMS files

Maximum RMS data item length	4096 bytes
Minimum RMS record length	1 byte
Maximum RMS record length	16,362 bytes
Maximum number of data items	16,362 bytes
Maximum length of the RMS key	255 bytes
Maximum number of RMS primary keys	1
Maximum number of secondary keys (sort, search, or alternate keys)	254
RMS bucket size	0 or 1–63 disk pages (Calculated by SUPRA Server if 0)
Buffering techniques	Supplied by VMS

Validation of use-entered data

Maximum number of domains	Unlimited (but only one domain per logical data item)
Maximum length of null value	32 bytes
Maximum length of range value	32 bytes
Maximum length of table value	72 bytes
Maximum number of validation tables	Unlimited (but only one validation table per domain)
Maximum number of values in a validation table	Unlimited
Maximum number of RDM validation exits	Unlimited (but only one per domain)

Views

Maximum length of a view name	30 characters
Maximum length of a logical data-item name	26 characters
Maximum logical-view length	200 lines
Maximum number of logical keys	9
Maximum length of column name	26 characters
Maximum column length	4088 characters
Maximum number of columns	9999 columns
Maximum number of columns per row	9999 columns
Maximum length of constant	24 characters
Maximum number of record codes per data set accessed by a view	10
Maximum number of views	Unlimited
Maximum number of views opened concurrently	Unlimited
Maximum number of bound views	Unlimited
Maximum number of global views	32,767 per user
Maximum number of derived views	Unlimited

SPECTRA query language

Direct access to data on a SUPRA Server database	Available through SPECTRA's Central Files
Database update capability	Available, but may be restricted by the DBA
Maximum number of SPECTRA personal files	99,999
Maximum length of a personal file field	256 bytes
Maximum number of SPECTRA processes	99,999
Maximum number of lines per SPECTRA process	No limit
Total records processed per session	Unlimited, but may be restricted by the DBA
Total sort file size	Unlimited, but may be restricted by the DBA
Number of SPECTRA file blocks per user	Unlimited, but may be restricted by the DBA
Online HELP facility	Direct access, indexed, and context-sensitive help screens
Nonstandard report formats	Yes, user-definable
Batch support	Yes
Spawn facility	Yes
Keypad customization	User-definable single- and multiple-keystroke environments

Supported data types

Integer	1 byte, 2 byte, 4 byte (longword), 8 byte (quadword)
Floating point	4 byte and 8 byte
Numeric/string	Numeric (trailing overpunch), 1–18 digit Leading numeric, 1–18 digit Packed numeric, 1–10 bytes Zoned numeric, 1–18 digit Character (4096)

Supported data access techniques

Direct access	Yes
Hashed	Yes
ISAM (Indexed Sequential Access Method)	Yes
Sequential	Yes
B tree	No
Indexed	No

UNIX environments

The following sections list the minimum and maximum values for SUPRA Server PDM support in UNIX environments.

Database descriptions

Maximum concurrent tasks	1–1000 (Default=10)
Maximum update tasks	0–1000 (Default=10)
Access methods	UNIX
Maximum number of data sets (files)	1024 per database
File types supported	SUPRA Server primary and related files
Network support	TCP/IP
Maximum number of databases on a SUPRA Server Directory	Unlimited
Recovery methods	Shadow logging Task logging System logging
Maximum number of buffer pools per database	1000

SUPRA Server physical data manager

Maximum concurrent PDM tasks	1–1000 (Default=50)
Maximum overlapping threads per database	1–100 (Default=3)
Maximum number of PDM operator terminals	Unlimited
Maximum size of PDM message buffer	32767 (Default=4096)
Maximum number of loaded databases	Unlimited
Maximum retries on held records	100 (Default=5)
Maximum time interval between retries	1000 seconds (Default=5)

SUPRA Server primary files

Total number of logical records	2–2147483647 (Default calculated during validation)
Logical records per block	1–32767 (Default=10)
Maximum primary record length	32766 bytes
Minimum primary record length	21 bytes
Maximum number of data items (fields)	4088
Maximum data item (field) length	4088 bytes
Maximum length of the control key	255 bytes
Maximum number of primary keys	1
Maximum number of linkpaths	510
Number of copies of a buffer	1–32767 (Default=5)
Maximum buffer size	32256

SUPRA Server related files

Total number of logical records	2–2147483647 (Default calculated during validation)
Logical records per block	1–32767 (Default=10)
Maximum record length	32766 bytes
Minimum record length	41 bytes
Maximum number of data items (fields)	4088
Maximum data item (field) length	4088 bytes
Maximum number of coded records	36 squared
Maximum length of the related key	255 bytes
Maximum number of related keys	454
Minimum number of related keys	1
Maximum number of linkpaths	511
Minimum number of linkpaths	1
Number of copies of a buffer	1–32767 (Default=5)
Maximum buffer size	32556

SUPRA Server index files

Maximum number of indexes per database	Unlimited
Maximum number of secondary keys per index	80
Maximum number of data items per secondary key	251
Maximum key size	255

Supported data-access techniques

Direct access	Yes
Hashed	Yes
ISAM (Indexed Sequential Access Method)	No
Sequential	Yes
B tree	Yes
Indexed	Yes
Linked	Yes

B

Format of entries for DOMAIN-UNIT

This appendix lists the valid entries for the DOMAIN-UNIT. The following information is provided for each entry:

- ◆ **Input-length.** The minimum number of characters you need to enter for the abbreviation.
- ◆ **Abbreviation.** The abbreviation that you can enter.
- ◆ **Term-value.** The full name of the unit, as displayed when entered.



RDM does not currently support DOMAIN-UNIT validation.

See [“Defining and maintaining domain details”](#) on page 334 for more information on maintaining domain details.

Input-length	Abbreviation	Term-value
0		N/A
0		N/A
0		N/A
2	A	ARE
4	AP-D	AP-DRAM
4	AP-G	AP-GRAIN
4	AP-L	AP-POUND
4	AP-O	AP-OUNCE
4	AP-P	AP-POUND
4	AP-S	SCRUPLE

Input-length	Abbreviation	Term-value
2	AR	ARE
4	AV-D	DRAM
4	AV-L	POUND
4	AV-O	OUNCE
4	BI-B	BI-BUSHEL
7	BI-FL-D	BI-FLUIDDRAM
7	BI-FL-O	BI-FLUIDOUNCE
9	BI-FLUIDO	BI-FLUIDOUNCE
9	BI-FLUIDR	BI-FLUIDDRAM
5	BI-GA	BI-GALLON
5	BI-GI	BI-GILL
4	BI-M	BI-MINIM
5	BI-PE	BI-PECK
5	BI-PI	BI-PINT
5	BI-PK	BI-PECK
5	BI-PT	BI-PINT
4	BI-Q	BI-QUART
2	BU	BUSHEL
2	C	CELSIUS
2	CA	CENTARE
2	CC	CUBIC-CENTIMETER
3	CEL	CELSIUS
5	CENTA	CENTARE
9	CENTIGRAD	CELSIUS
9	CENTIGRAM	CENTIGRAM
6	CENTIL	CENTILITER
13	CENTIMETER	CENTIMETER
13	CENTIMETER**2	SQUARE-CENTIMETER
13	CENTIMETERS-P	CENTIMETERS-PER-SECOND
2	CG	CENTIGRAM
2	CL	CENTILITER
3	CM	CENTIMETER

Input-length	Abbreviation	Term-value
3	CM*	SQUARE-CENTIMETER
3	CMP	CENTIMETERS-PER-SECOND
4	CU-CE	CUBIC-CENTIMETER
4	CU-CM	CUBIC-CENTIMETER
4	CU-CU	CUBIC-CUBIT
3	CU-F	CUBIC-FOOT
3	CU-I	CUBIC-INCH
3	CU-M	CUBIC-METER
3	CU-Y	CUBIC-YARD
8	CUBIC-CE	CUBIC-CENTIMETER
8	CUBIC-CM	CUBIC-CENTIMETER
8	CUBIC-CU	CUBIC-CUBIT
7	CUBIC-F	CUBIC-FOOT
7	CUBIC-I	CUBIC-INCH
7	CUBIC-M	CUBIC-METER
7	CUBIC-Y	CUBIC-YARD
6	CUBIT	CUBIT
6	CUBIT*	SQUARE-CUBIT
3	DAK	DEKAGRAM
3	DAL	DEKALITER
3	DAM	DECAMETER
3	DAS	DEKASTERE
3	DAY	DAYS
3	DD/	EUROPEAN
4	DECA	DECAMETER
5	DECIG	DECIGRAM
5	DECIL	DECILITER
5	DECIM	DECIMETER
5	DECIS	DECISTERE
5	DEKAG	DEKAGRAM
5	DEKAL	DEKALITER
5	DEKAS	DEKASTERE

Input-length	Abbreviation	Term-value
2	DG	DECIGRAM
2	DI	DINARS
2	DL	DECILITER
3	DM	DECIMETER
0	DMY	EUROPEAN
2	DO	DOLLARS
3	DR	DRAM
3	DRA	DRAM
5	DRY-P	DRY-PINT
5	DRY-Q	DRY-QUART
2	DS	DECISTERE
2	DW	PENNYWEIGHT
1	E	EUROPEAN
2	F	FAHRENHEIT
2	FA	FAHRENHEIT
2	FE	FEET-PER-SECOND
4	FL-D	FLUIDDRAM
4	FL-O	FLUIDOUNCE
6	FLUIDO	FLUIDOUNCE
6	FLUIDR	FLUIDDRAM
5	FOOT	FOOT
5	FOOT*	SQUARE-FOOT
4	FORM	FORMATTED
4	FORT	FORTNIGHT
2	FP	FEET-PER-SECOND
2	FR	FRANCS
3	FT	FOOT
3	FT*	SQUARE-FOOT
2	FU	FURLONG
2	G	GRAM
3	G/C	GRAMS-PER-SQ-CENTIMETER
3	G/S	GRAMS-PER-SQ-CENTIMETER

Input-length	Abbreviation	Term-value
2	GA	GALLON
2	GI	GILL
3	GM	GRAM
4	GM/C	GRAMS-PER-SQ-CENTIMETER
4	GM/S	GRAMS-PER-SQ-CENTIMETER
3	GR	GRAIN
4	GRAI	GRAIN
5	GRAM	GRAM
5	GRAMS	GRAMS-PER-SQ-CENTIMETER
3	GRE	GREGORIAN
2	HA	HECTARE
5	HECTA	HECTARE
6	HECTOG	HECTOGRAM
6	HECTOL	HECTOLITER
6	HECTOM	HECTOMETER
2	HG	HECTOGRAM
2	HL	HECTOLITER
2	HM	HECTOMETER
2	HO	HOURS
2	HR	HOURS
3	IN	INCH
3	IN*	SQUARE-INCH
5	INCH	INCH
7	INCH**2	SQUARE-INCH
8	INCHES-P	INCHES-PER-SECOND
2	IP	INCHES-PER-SECOND
1	J	JULIAN
2	K	KELVIN
2	KE	KELVIN
3	KG	KILOGRAM
3	KG/	KILOGRAMS-PER-CENTARE
9	KILOGRAM	KILOGRAM

Input-length	Abbreviation	Term-value
11	KILOGRAMS-P	KILOGRAMS-PER-CENTARE
5	KILOL	KILOLITER
10	KILOMETER	KILOMETER
10	KILOMETER*	SQUARE-KILOMETER
12	KILOMETERS-P	KILOMETERS-PER-HOUR
2	KL	KILOLITER
3	KM	KILOMETER
3	KM*	SQUARE-KILOMETER
2	KN	KNOT
2	KP	KILOMETERS-PER-HOUR
2	L	LITER
3	LB	POUND
5	LBS/F	POUNDS-PER-SQUARE-FOOT
5	LBS/I	POUNDS-PER-SQUARE-INCH
8	LBS/SQ-F	POUNDS-PER-SQUARE-FOOT
8	LBS/SQ-I	POUNDS-PER-SQUARE-INCH
8	LIQUID-P	PINT
8	LIQUID-Q	QUART
3	LIR	LIRE
3	LIT	LITER
6	LONG-C	LONG-HUNDREDWEIGHT
6	LONG-H	LONG-HUNDREDWEIGHT
6	LONG-T	LONG-TON
2	M	METER
2	MA	MARKS
2	MD	AMERICAN
6	METER	METER
8	METERS-P	METERS-PER-SECOND
4	METR	METRIC-TON
2	MG	MILLIGRAM
3	MI	MILE

Input-length	Abbreviation	Term-value
3	MI*	SQUARE-MILE
5	MILE	MILE
5	MILE*	SQUARE-MILE
7	MILES-P	MILES-PER-HOUR
6	MILLIG	MILLIGRAM
6	MILLIL	MILLILITER
11	MILLIMETER	MILLIMETER
13	MILLIMETERS-P	MILLIMETERS-PER-SECOND
4	MIN	MINIM
4	MINI	MINIM
4	MINS	MINUTES
4	MINU	MINUTES
2	ML	MILLILITER
3	MM	MILLIMETER
3	MM/	AMERICAN
3	MMP	MILLIMETERS-PER-SECOND
2	MO	MONTHS
3	MPH	MILES-PER-HOUR
3	MPS	METERS-PER-SECOND
2	MT	METRIC-TON
2	MY	MYRIAMETER
2	OU	OUNCE
2	OZ	OUNCE
3	PEC	PECK
3	PEN	PENNYWEIGHT
3	PES	PESOS
2	PI	PINT
2	PK	PECK
6	POUND	POUND
7	POUNDS	POUNDS

Input-length	Abbreviation	Term-value
19	POUNDS-PER-SQUARE-F	POUNDS-PER-SQUARE-FOOT
19	POUNDS-PER-SQUARE-I	POUNDS-PER-SQUARE-INCH
2	PT	PINT
2	PW	PENNYWEIGHT
2	Q	QUINTAL
2	QT	QUART
3	QUA	QUART
3	QUI	QUINTAL
3	RD	ROD
3	RD*	SQUARE-ROD
4	ROD	ROD
4	ROD*	SQUARE-ROD
2	S	STARE
2	SC	SCRUPLE
2	SE	SECONDS
7	SHORT-C	SHORT-HUNDREDWEIGHT
7	SHORT-H	SHORT-HUNDREDWEIGHT
7	SHORT-T	TON
5	SQ-CE	SQUARE-CENTIMETER
5	SQ-CM	SQUARE-CENTIMETER
5	SQ-CU	SQUARE-CUBIT
4	SQ-F	SQUARE-FOOT
4	SQ-I	SQUARE-INCH
4	SQ-K	SQUARE-KILOMETER
4	SQ-M	SQUARE-MILE
4	SQ-R	SQUARE-ROD
9	SQUARE-CE	SQUARE-CENTIMETER
9	SQUARE-CU	SQUARE-CUBIT
8	SQUARE-F	SQUARE-FOOT

Input-length	Abbreviation	Term-value
8	SQUARE-I	SQUARE-INCH
8	SQUARE-K	SQUARE-KILOMETER
8	SQUARE-M	SQUARE-MILE
8	SQUARE-R	SQUARE-ROD
3	STA	STARE
3	STO	STONE
3	T-G	TROY-GRAIN
3	T-L	TROY-POUND
3	T-O	TROY-OUNCE
2	TO	TON
6	TROY-G	TROY-GRAIN
6	TROY-O	TROY-OUNCE
6	TROY-P	TROY-POUND
4	US-B	BUSHEL
8	US-DRY-P	DRY-PINT
8	US-DRY-Q	DRY-QUART
7	US-FL-D	FLUIDDRAM
7	US-FL-O	FLUIDOUNCE
9	US-FLUIDO	FLUIDOUNCE
9	US-FLUIDR	FLUIDDRAM
5	US-GA	GALLON
5	US-GI	GILL
11	US-LIQUID-P	PINT
11	US-LIQUID-Q	QUART
4	US-M	MINIM
5	US-PE	PECK
5	US-PI	PINT
5	US-PK	PECK
5	US-PT	PINT
4	US-Q	QUART

Input-length	Abbreviation	Term-value
2	WE	WEEKS
2	WK	WEEKS
5	YARD	YARD
7	YARDS-P	YARDS-PER-SECOND
3	YD	YARD
4	YDPS	YARDS-PER-SECOND
3	YEARS	YEARS
3	YEN	YEN
3	YRS	YEARS

Index

A

- abbreviations, for domain units of measurement 383
- access authority, for SUPRA DBA
 - defining 288
 - described 103
- access method, for database files 115
- access mode, defining for a data set 126
- access, required for defining views 354
- accessing SUPRA DBA 66
- action codes, for entering comments 97
- adding
 - primary data items
 - considerations for 144
 - how to 139
 - programs to the SUPRA directory 303
 - related data items 146
 - relationships between users and views 370
- altering a data set load limit 298
- attributes
 - defined 18
 - defining for physical file 123
 - displaying for physical data item 317
 - modifying for physical data item 317
- authority. *See* access authority
- authorization, defining for views 370
- automatic log dumping 195, 197

B

- base views 315
- BASIC description for data items 325
- batch
 - formatting databases and data sets in 258
 - running database validation, compile and print in 244
- binding views 364
- block size, defining for
 - system log 191
 - task log 187
- bucket size, defining for RMS data sets 150
- buffers
 - changing 359
 - defining 178
 - data set type 180
 - naming conventions 179
 - number of copies 180
 - size 180
 - EDT =MAIN
 - described 352
 - displaying 357
 - EDT =PASTE 359
 - effects of PDM access methods on 115
 - listing 360
- byte order, defining 116

C

- calculating
 - record length 57
 - space for
 - database 57
 - primary data sets 57
 - related data sets 57
- calling mechanism, defining 117
- changing a data set load limit 298
- changing buffers 359
- cluster environment, specifying for
 - multiple shadow files 131
 - one shadow file 129
- cluster support, defining 118
- COBOL description for data items 325

- coded records
 - linkpaths in 52
 - physical keys in 52
- coded related records,
 - characteristics 50
- COMBAT command
 - format of 244
 - running without a password 248
- command files, for your EDT environment (DBAEDT.EDT) 354
- commands
 - COMBAT. *See* COMBAT command
 - csmcombat 244
 - data item maintenance 138
 - difference between examine and modify 108
 - EDT EXIT 361
 - EDT INCLUDE 362
 - EDT WRITE 361
 - for entering comments 97
 - signing on to SUPRA DBA 66
- comment action codes 97
- comment screen, bypassing
 - during data set definition 122
- comments
 - entering for
 - database descriptions 119
 - domains 344
 - entities 96
 - logical data items 318
 - programs 307
 - users 288
 - validation tables 351
 - rules for entering 97
 - screen example 97
- compiling a copied database 314
- compiling and printing a database
 - from command level 244
 - from DBA 235
- connecting entities 91
 - domains to data items 346
 - prerequisite for 343
 - restrictions on 348
 - rules for 332
 - tracking connections 343
 - domains to validation tables 344
- connections between entities
 - creating 79
 - deleting 85
 - examining 71
 - listing 89
 - modifying 75
- control field, specifying size of 144
- Control functions 194
- control interval
 - defining for a data set 127
 - specifying the load limit for 128
- converting data
 - first normal form 33
 - second normal form 34
 - third normal form 35
 - unnormalized form 32
- converting data, to third normal form
 - optimizing 37
- copying a database description 311
- corrupt index file
 - checking for 167
 - specify how PDM should handle 166
- creating *See also* defining entities
 - connected entities and relationships 79
 - described 77
 - linkpaths, naming
 - conventions for 144
 - RMS keys 152
- csmcombat command 244
- CSTUDSLF logical name
 - described 195
 - location of 197
- CSTUIDX (index format program)
 - running from command level 278
 - running from DBA 265
 - running in the batch VMS version 280
- customizing the EDT environment 367

D

data

- analyzing for database design
 - 31
- converting to first normal form
 - 33
- converting to second normal form
 - 34
- converting to third normal form
 - 35
- converting to unnormalized form
 - 32
- optimizing third normal form
 - 37
- data item length
 - database validation 343
 - restriction on modifying 343
- data items *See also* logical data items, physical data items, primary data items, related data items, and RMS data items
 - automatically created by SUPRA DBA 144
 - deleting 144
 - filler 140
 - including in secondary keys 175
 - naming conventions 144
 - subdefining 141
- data relationships *See also* relation and relationships
 - defined 18
 - many-to-many 37
- data sets *See also* primary data sets, related data sets, and RMS data sets
 - bypassing the comment screen 122
 - connecting to more than one database 123
 - defining details 134
 - defining physical attributes for
 - access mode 126
 - allocation for physical disk file 130, 131
 - control interval 127
 - load limit 128
 - physical disk file for shadow files 129, 131
 - physical disk file name 129, 130
 - record number 125
 - records per block 125
 - naming conventions 121
 - selecting options 132
 - specifying for database format 255
 - unlocking 289
- data type of domains
 - defining 336
 - effects of on domain length 337
- data views. *See* views
- database description
 - entering comments for 119
 - functions 105
 - options 109
- database design
 - analyzing your data 31
 - analyzing your organization 27
 - creating physical design 53
 - optimizing space usage 57
 - optimizing the design 60
- database details, defining
 - access method 115
 - byte order 116
 - calling mechanism 117
 - cluster support for 118
 - database status 114
 - maximum held records 112
 - maximum signed on tasks 112
 - maximum update tasks signed on 113
 - network support 118
 - password 111
 - shadow files 113
 - single tasking 114
- database name
 - rules for 104
 - specifying for view definition 356
- database status
 - after system log dump 196
 - as a result of locking 290
- database validation,
 - disconnection of domains during 343

databases

- automatic population of 291
 - compiling a copied database 314
 - compiling and printing
 - from command level 244
 - from DBA 235
 - copying 311
 - deleting 83
 - formatting
 - from command level 257
 - from DBA 253
 - in batch 258
 - modifying a copied database 313
 - printing from DBA 240
 - security for deleting 82
 - unlocking 289
 - validating
 - from command level 244
 - from DBA 231
 - validating a copied database 314
- DBA. *See* SUPRA DBA
- DBA/Utilities access authority 288
- DBAEDT logical name 354
- DBAEDT.EDT command file
 - customizing 367
 - described 354
- DBAID, using to print views 366
- decimal places
 - defining for domains 338
 - specifying for RMS data items 156
- default value, defining for domains 340
- defining *See also* creating
- buffers
 - data set type 180
 - naming conventions 179
 - number of copies 180
 - size 180
 - data items
 - RMS 157

data sets

- access mode 126
 - bypassing the comment screen 122
 - control interval 127
 - load limit 128
 - naming conventions 121
 - physical disk file for shadow files 129, 131
 - physical disk file name 129, 130
 - physical disk file size 130, 131
 - physical file attributes 123
 - record number 125
 - records per block 125
- database details
- access method 115
 - byte order 116
 - calling mechanism 117
 - cluster support 118
 - database status 114
 - maximum held records 112
 - maximum signed on tasks 112
 - maximum update tasks
 - signed on 113
 - network support 118
 - password 111
 - shadow files 113
 - single tasking 114
- domains
- allowing null values 339
 - consideration for 343
 - data type 336
 - decimals 338
 - default value 340
 - function 335
 - length 337
 - naming conventions 334
 - sign 338
 - status 342
 - unit of measurement 335
- indices
- corrupt index check 167
 - how PDM should handle
 - corrupt index 166
 - null sorting 166
 - physical disk file 167
 - physical disk file for shadow file 168

- defining (*cont.*)
 - changing associated physical details 321
 - connecting to domains 329
 - naming conventions for 320
- record codes 145
- records
 - primary 136
 - related 145
- Recovery Unit Journaling 189
- secondary keys
 - duplicate percentage 174
 - including a data item 175
 - physical attributes for 170
 - pointer type 173
 - pointers 172
 - sorting by data type 173
 - sorting direction 171
 - uniqueness 171
- system logs
 - block size 191
 - number of blocks 192
 - physical file 192
 - physical file for shadow copy 193
- task logs
 - block size 187
 - number of blocks 188
 - physical file 188
 - physical file for shadow copy 188
- users
 - comments for 288
 - details 287
 - password for 288
 - user name 286
- validation tables
 - entering comments 351
 - entering text 350
- views
 - defining view name and connected database 355
 - entering view text using EDT 358
 - creating VMS text files for 361
 - displaying line numbers 359
 - function keys available 358
 - using EDT buffers 359
 - functions available 353
- deleting
 - considerations for 83
 - data items
 - primary 138
 - restrictions on 144
 - databases 83
 - domains 344
 - entities 81
 - entities in general
 - connected entities and relationships 85
 - described 83
 - logical data items 330
 - physical files 84
 - relationships between users and views 370
 - validation tables 351
 - views 369
- derived views 315
- designing your database
 - analyzing your data 31
 - analyzing your organization 27
 - converting logical design to SUPRA Server database 53
 - creating the physical design 46
 - how-to flowchart 26
 - optimizing 60
 - optimizing space for 57
 - timing processing cycles 58
- development access authority 288
- device, defining for system log
 - dump input file 198
- direct record retrieval 173
- directory, defining for system log
 - dump input file 200
- disconnecting entities
 - described 94
 - domains 332
- displaying
 - entities
 - connected entities and relationships 89
 - described 87
 - menus with list function 87
 - help 100

- domains
 - and data item modification 343
 - and database validation 343
 - connecting to
 - data items 346, 348
 - validation tables 344
 - defining
 - allowing null values 339
 - considerations for 343
 - data type 336
 - decimals 338
 - default value 340
 - function 335
 - length 337
 - name 334
 - sign 338
 - status 342
 - unit of measurement 335
 - deleting 344
 - described 331
 - entering comments for 344
 - naming 329
 - unit of measurement
 - valid entries 383
 - unlocking 289
- double linkage 56
- dumping system log file
 - automatically
 - UNIX 203
 - VMS 197
 - online
 - UNIX 210
 - VMS 208
- DUMPSLF logical name
 - defining for automatic system
 - log dump
 - UNIX 206
 - VMS 201
 - defining for online system log
 - dump
 - UNIX 210
 - VMS 208
 - described 195
- duplicate key retrieval 173
- duplicates, specifying for
 - secondary keys 174

E

- EDIT/EDT interface
 - displaying line numbers 359
 - exiting EDT 364
 - function keys available 358
 - invoking 352
 - using EDT buffers 359
 - using to define or edit views 358
- editing views using EDIT/EDT
 - creating VMS text files for 361
 - displaying line numbers 359
 - displaying view text 357
 - function keys available 358
 - using EDT buffers 359
- EDT
 - buffers
 - =MAIN 352, 357
 - =PASTE 359
 - commands
 - EXIT 361
 - INCLUDE 362
 - WRITE 361
 - environment, customizing 367
 - function keys 358
- EDTINI 354
- enrolling programs in the SUPRA directory 303
- entering
 - comments
 - for database descriptions 119
 - for domains 344
 - for validation tables 351
 - how to 96
 - rules 97
 - text for validation tables 350
- entities
 - connecting
 - described 91
 - creating
 - connected entities and relationships 79
 - defined 18
 - deleting
 - connected entities and relationships 85
 - disconnecting 94
 - entering comments for 96

- entities (*cont.*)
 - examining
 - connected entities and relationships 71
 - menus that allow the examine function 69
 - listing
 - connected entities and relationships 89
 - menus that allow the list function 87
 - modifying
 - connected entities and relationships 75
 - menus that allow the modify function 73
 - unlocking 289
 - errors, how to resolve during
 - database validation, compilation, or print 290
 - examining entities
 - connected entities and relationships 71
 - domains 333
 - how to 69
 - logical data items
 - comments 318
 - displaying associated physical data item 318
 - displaying associated physical data item attributes 317
 - menus with examine function 69
 - primary data items 136
 - EXIT, EDT command 361
 - exiting the EDIT/EDT interface 364
 - expanding a related data set 291, 296
- F**
- failure
 - of system log dump
 - UNIX 207
 - VMS 202
 - recovering from program failure 183
 - specifying shadow files for 113
 - filler data items 140
 - first normal form 33
 - format, defining for domains 336
 - formatting
 - databases
 - from command level 257
 - from DBA 253
 - in batch 258
 - indices
 - described 262
 - from command level 278
 - from DBA 265
 - function keys for 263
 - in the batch VMS version 280
 - preliminary procedures 262
 - RMS data sets
 - from command level 257
 - in batch 258
 - FORTRAN description for data items 325
 - function keys
 - described 65
 - for binding views 364
 - for EDT 358
 - for index formatting 263
 - for saving views 363
 - rules for using 65
- G**
- getting Help 20
- H**
- Help key 20
 - Help library
 - specifying in your DBAEDT.EDT file 367
 - UNIX 20
 - VMS 20
 - Help, displaying 100
 - host byte order 116

I

INCLUDE, EDT command 362
 indices
 defining
 corrupt check 167
 null sorting 166
 PDM action if corrupt 166
 physical disk file 167
 shadow file specification 168
 formatting and populating
 described 262
 from command level 278
 from DBA 265
 function keys for 263
 in the batch VMS version 280
 preliminary procedures 262
 naming conventions for 164
 indirect record retrieval 173
 input file, for system log dumping
 UNIX 203
 VMS 197
 internal schema 331
 invoking the EDIT/EDT interface
 352

K

key, defining for RMS records
 154

L

length
 defining for
 data items, restrictions on
 modifying 343
 domain units of measurement
 383
 domains 337
 RMS data items
 considerations for 159
 RMS key 154
 linking, a relation to itself 56
 linkpaths
 defined 48
 function in coded records 52
 naming conventions for 144,
 148

listing entities
 connected entities and
 relationships 89
 described 87
 menus with list function 87
 load limit
 changing for a related data set
 302
 defining 128
 local access for index files 167
 logical data items
 connecting to a domain 329,
 346
 creating 319
 deleting 330
 described 315
 examining
 associated physical data item
 318
 associated physical data item
 attributes 317
 comments for 318
 functions available for 316
 modifying associated physical
 details 321
 naming conventions for 320
 logical database design
 converting to SUPRA Server
 database 53
 creating 40
 logical name, EDTINI 354
 logical views. *See* views

M

main menu 21
 maintaining *See also* modifying
 programs
 defining details for 305
 deleting from SUPRA
 directory 308
 displaying views used by 306
 entering comments for 307
 modification level 305
 many-to-many data relationship
 37
 maximum
 records held 112
 tasks signed on 112
 values for a domain 342
 menu hierarchy 64

- menus
 - with disconnect function 94
 - with examine function 69
 - with list function 87
- minimum value, defining for a domain 341
- minimum/maximum values, SUPRA Server, PDM support
 - UNIX 378
 - VMS 371
- modification level, of programs 305
- modifying
 - copied databases 313
 - data items
 - primary 142, 144
 - related 148
 - restrictions for 348
 - domains
 - allowing null values 339
 - changing the domain name 334
 - data type 336
 - decimals 338
 - default value 340
 - function 335
 - length 337
 - sign 338
 - status 342
 - unit of measurement 335
 - entities
 - connected entities and relationships 75
 - menus with modify function 73
 - logical data items
 - associated physical data item
 - attributes 317
 - comments 318
 - displaying associated physical data item 318
 - views 352
 - modifying view name and connected database 355
 - modifying view text using EDT 358
 - creating VMS text files for 361
 - displaying line numbers 359
 - function keys available 358
 - using EDT buffers 359

- MPD. See Multiple Physical Database (MPD) facility
- Multiple Physical Database (MPD) facility 309

N

- naming conventions, for
 - buffers 179
 - data items 144
 - domains 329
 - indices 164
 - logical data items 320
 - secondary keys 170
 - users 286
 - validation tables 345
 - views 356
- network byte order 116
- network support, defining 118
- normalizing data 31

O

- optimizing
 - database design 60
 - database space 57
 - record retrieval 127
- output file, defining for system
 - log dump
 - UNIX 204
 - VMS 199

P

- parameter file, defining for
 - database compilation 238
 - database printing 243
 - database validation 233
- passing parameters to PDM 117
- passwords
 - defining for database 111
 - defining for users 288
 - withholding display of database password 248

PDM

- access method to your database 115
- processing during a system log dump
 - in general 196
 - UNIX 214
 - VMS 211
- reset after failure 183
- specifying corrupt index check for 167
- specifying number of related records held in 127
- performance, improving
 - in multitasking environments 180
 - using logical records per block 125
- physical data items
 - association with a logical data item 318
 - modifying attributes of 317
 - restrictions on changing 332
- Physical Data Manager (PDM) values for SUPRA Server
 - UNIX 378
 - VMS 371
- physical disk file, defining for
 - compiled database 234, 239
 - data set during an expand 295
 - data set shadow files 129
 - data sets 129
 - index shadow files 168
 - indices 167
 - system logs 192
 - task log 188
- physical files
 - defining attributes for 123
 - deleting 84
- physical key, function of in coded records 52
- pointers
 - effects on record retrieval 173
 - specifying
 - for secondary keys 172
 - type 173
 - using with
 - primary data sets 172
 - related data sets 172

- populating indices
 - described 262
 - from command level 278
 - from DBA 265
 - function keys for 263
 - in the batch VMS version 280
 - preliminary procedures 262
- primary data items
 - adding 139, 144
 - deleting 138
 - examining 136
 - listing 136, 139
 - modifying 144
- primary data sets
 - and pointer ordering 172
 - buffer consideration 179
 - calculating space for 57
 - defined 46
 - duplicate key retrieval 173
 - format 48
 - minimum length 144
- primary records, defining 136
- printing
 - databases
 - from command level 244
 - from DBA 240
 - views
 - using DBA 365
 - using DBAID 366
- privileged access authority 288
- programs
 - allowing space for 112
 - defining details 305
 - deleting from SUPRA directory 308
 - enrolling in the SUPRA directory 303
 - entering comments for 307
 - logical views used by 306
 - modification level of 305
 - performing maintenance for 303
 - unlocking 289

Q

- QFUL status 114

R

RDM domain usage report 343
 RDM user access authority 288

read failure, specifying shadow files for 113

read-only access authority 288

recompiling a database 348

record codes, defining 145

record length

calculating 57

minimum for

primary data sets 144

related data sets 148

record retrieval

and pointer ordering 173

optimizing 127

records

defining for data set 125

printing during system log

recovery 220

specifying maximum held 112

records per block, defining for a data set 125

Recovery Unit Journaling, defining 189

recovery. *See* Recovery Unit Journaling, shadow

recording, system log

recovery, and task log

recovery

reference calling mechanism, defining 117

related data items

adding 146, 148

CODE data item 148

modifying 148

related data sets

and pointer ordering 172

buffer consideration 179

calculating space for 57

defined 48

defining control interval for 127

duplicate key retrieval 173

example 55

expanding 291, 296

resetting load limit

considerations for 302

procedures for 298

related records, defining 145

relation

linking to another relation 55

linking to itself 56

relationships

creating for users and views 370

types of connections you can

create 79

delete 85

examine 71

list 89

modify 75

Relative Record Number (RRN), and pointer ordering 172

remote access, specifying for your database 118

reports, printing view reports 366

resetting a data set load limit

considerations for 302

how to 298

restrictions, for connecting

domains to data items 348

RMS data items

considerations for type, length and sign 159

defining 157

data item level 160

length 157

number of decimal places 156

sign 155

sub data items 160

type 158

use 157

RMS data sets

defining 149

defining Recovery Unit Journaling for 189

described 46

formatting 256, 258

RMS key

allowing duplicate values for 154

allowing modification of 155

naming conventions 153

specifying length 154

specifying number 153

RMS records

- creating RMS key 152
- defining 152

ROOT data item 144

RRN. See Relative Recrod

- Number (RRN), and pointer ordering

S

saving views 363

screen hierarchy 64

second normal form 34

secondary keys

- defining 169
- naming conventions 170
- physical attributes for 170
- pointers 172
- sorting direction 171
- uniqueness 171

Secondary keys

- defining
- duplicate percentage 174
- pointer type 173
- sorting by data type 173
- including data items in 175
- naming conventions for 176

security, provided with the delete

- function 82

shadow files

- defining for
- read or write failure 113
- system log 193
- defining physical disk for 129
- specifying for an expanded data set 295

Shadow files

- defining for
- task log 188

sign

- considerations for 159
- defining for domains 338
- specifying for RMS data items 155

signing on, to SUPRA DBA 66

sign-on screen 66

single task environment, and maximum held records 112

size, of physical disk files 130, 131

sorting

- direction, specifying for secondary keys 171
- null values 166

Sorting

- records by data type 173

space

- calculating for
- primary data sets 57
- related data sets 57
- optimizing for your database 57

starting

- EDIT/EDT interface 352
- SUPRA DBA 66

statistics, specifying for system

- log recovery 221

status of database

- as result of locking 290
- displaying 114

Status of database

- after system log dump 196

status, defining for domains 342

structure data set 56

subdata items, defining for

- primary data items 141
- RMS data items 160

SUPRA DBA

- access authority 103
- defined 17, 18
- function keys 65
- Help key 20
- main menu 21
- screen hierarchy 64
- signing on to 66
- terminal types supported 20

SUPRA directory, enrolling

- programs in 303

SUPRA Server

- PDM support
- UNIX minimum/maximum values 378
- VMS minimum/maximum values 371

SUPRA Server components

- in UNIX environments 20
- in VMS environments 19

syntax, for
 domain names 334
 DUMPSLF logical name 201,
 206
 system log dump input file 198,
 203
 validation table names 345
 view names 356
 system log
 defining
 block size 191
 number of blocks 192
 physical file 192
 physical file for shadow copy
 193
 described 194
 system log dump
 automatic log dump
 defining directory 200, 203
 defining input file 197, 203
 defining logical name
 DUMPSLF 201, 206
 defining output file 199, 204
 example DUMPSLF definition
 202, 206
 example input file 200, 205
 input file, creating 198, 203
 failure of 202, 207
 online log dump 208, 210
 stages of 211, 214
 system log recovery
 defining 190
 described 216
 running
 from command level 226
 from DBA 217

T

task log recovery, defining
 block size 187
 minimum block size 187
 number of blocks 188
 physical file 188
 physical file for shadow copy
 188
 task log, considerations 114
 task logging, description 183
 tasking, specifying single or multi
 114
 tasks, specifying maximum 112
 terminal types supported 20

text, entering for validation tables
 350
 third normal form 35, 37
 type of data item, defining 159

U

unique, specifying for secondary
 keys 171
 unit of measurement, for
 domains
 defining 335
 valid values for 383
 UNIX
 Help library 20
 SUPRA Server components 20
 system log dump
 automatic 203
 online 210
 stages of 214
 terminal types supported by
 DBA 20
 UNIX environment,
 minimum/maximum values
 for SUPRA Server (PDM)
 database descriptions 378
 index files 380
 primary files 379
 related files 380
 supported data access
 techniques 381
 SUPRA Server Physical Data
 Manager 379
 unlocking entities 289
 unnormalized form 32
 user authorization, defining for
 views 370
 users, defining 285
 comments for 288
 details 287
 password for 288
 user name 286
 using the EDIT/EDT interface to
 edit views 358
 creating VMS text files for 361
 customizing the EDT
 environment 367
 displaying line numbers 359
 displaying view text 357
 exiting EDT 364
 function keys available 358
 using EDT buffers 359

V

- validating databases
 - considerations for 343
 - from command level 244
 - from DBA 231
 - validating a copied database 314
- validation tables
 - connecting to a domain 344
 - defining 349
 - deleting 351
 - described 332
 - entering table comments 351
 - entering table text 350
 - naming 345
 - unlocking 289
- values
 - defining for a domain 341
 - for domain unit of measurement 383
- VAX byte order 116
- views
 - authorization for using 354
 - base views 315
 - binding 364
 - defined 28
 - defining 352
 - authorization for 370
 - entering view text using EDT 358, 359, 361
 - functions available 353
 - specifying name and connected database 356
 - deleting 369
 - derived views 315
 - displaying those used by programs 306
 - editing using EDT 358
 - printing
 - using DBA 365
 - using DBAID 366
 - saving 363
 - unlocking 289

VMS

- creating text files for views 361
- expanding a related data set 291
- Help library 20
- Recovery Unit Journaling 189
- SUPRA Server components 19
- system log dump
 - automatic 197
 - online 208
 - stages of 211
- terminal types supported by DBA 20

VMS environment

- minimum/maximum values for SUPRA Server (PDM)
 - database descriptions 371
 - index files 373
 - primary files 372
 - related files 373
 - RMS files 374
 - SPECTRA Query Language 376
- supported data types 377
- supported data-access techniques 377
- SUPRA Server Physical Data Manager 372
- validation of user-entered data 374
- views 375

W

- write failure, specifying shadow files for 113
- WRITE, EDT command 361