

Cincom

SUPRA SERVER PDM

Digest for OS/390 and VSE Systems

P26-9062-70



SUPRA[®] Server PDM Digest for OS/390 and VSE Systems

Publication Number P26-9062-70

© 1989–1994, 1996, 1998–2000, 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This document contains unpublished, confidential, and proprietary information of Cincom.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 Digest for OS/390 and VSE Systems*, P26-9062-70, is dated January 15, 2002. This document supports Release 2.7 of SUPRA Server with IBM PDM support.

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 Street
Cincinnati, OH 45246-3732
U.S.A.



Contents

About this book	ix
Using this document.....	ix
Document organization	ix
Revisions to this manual	x
Conventions	xi
SUPRA Server documentation series	xiv
SUPRA Server overview	17
Three-schema architecture	17
SUPRA Server components.....	19
SUPRA Server with PDM support.....	20
SUPRA Server with PDM support.....	21
Components and tools of SUPRA Server PDM	21
Features of SUPRA Server with PDM support.....	22
Features of SUPRA Server with RDM support	23
SUPRA Server with PDM and RDM support	25
Overview	26
Understanding the relational data structure	28
Ensuring security in SUPRA Server	29
Using multiple file types.....	30

Using the Physical Data Manager	31
Physical Data Manager overview.....	31
Dynamic indexing	33
Types of recovery	33
Using the PDM in OS/390 and VSE environments.....	36
Maintaining the active schema	36
Initializing the PDM	37
Operating in different modes	38
Managing storage space	42
Using utilities for PDM files.....	44
Restarting after a task failure.....	46
Restarting a task after a system failure	46
Recovering and restoring the PDM after a system failure	47
Maintaining the PDM using Interactive Services	48
Using RDM to access relations	49
Designing and building views.....	50
Building view definitions with components.....	52
Validating view columns with domains	53
Adding integrity constraints to base views	54
Creating derived view definitions	55
Accessing data with views	57
Optimizing performance	57
Maintaining current programs	58
Insulating programs from change	59
Creating and testing views with DBAID	60
Obtaining RDM reports and statistics	62
Using Directory Maintenance to define metadata	63
Defining entities and relationships	64
Creating directory entities in a hierarchy.....	69
Establishing entity relationships	70
Maintaining directory security	72
Using Online Directory Maintenance	73
Sign-on screen.....	73
Category menu	74
Command menu	75
Entity data screens	76
Using Batch Directory Maintenance	77
Input statements	77
Batch processing	80
Batch output.....	81
Using the Directory reports	82

Using SPECTRA to retrieve data	83
Overview	83
Selecting an option from the Master menu	85
Splitting screens for dual use	86
Using SPECTRA commands	87
Using the specialized facilities of SPECTRA	88
Customizing reports with built-in functions	88
Calculating statistics	89
Comparing values	90
Controlling the display	91
Using null support	91
Running a SPECTRA process	92
Managing SPECTRA administrative functions	95
Accessing external files	95
Maintaining user profiles	96
Maintaining the personal file system	98
Recovering from a failure	98
Producing user profile reports	98
SUPRA Server documentation synopses	99
PDM/RDM documentation series (OS/390 and VSE)	99
Index	105

About this book

Using this document

This manual introduces SUPRA Server and associated database access, management, and connectivity tools. It is intended to give a SUPRA Server user a detailed overview of capabilities and features.

Document organization

The information in this manual is organized as follows:

Chapter 1—SUPRA Server overview

Describes the SUPRA Server relational database management system and its tools and architecture.

Chapter 2—SUPRA Server with PDM and RDM support

Describes PDM and RDM support in SUPRA Server for high-volume, update-oriented production processing.

Chapter 3—Using the Physical Data Manager

Describes how to use the PDM, which controls the storage and maintenance of data in SUPRA Server PDM databases.

Chapter 4—Using RDM to access relations

Describes how SUPRA Server uses the Relational Data Manager to provide relational access to physical files.

Chapter 5—Using directory maintenance to define metadata in OS/390 and VSE environments

Describes how to use Online or Batch Directory Maintenance to establish, display, modify, or delete entities and relationships on the Directory in IBM environments.

Chapter 6—Using SPECTRA to retrieve data

Describes SPECTRA, a sophisticated report writer and database access tool. This component is not available for UNIX or Alpha environments.

Appendix A—SUPRA Server documentation synopses

Provides general information about the SUPRA Server manual series.

Index

Revisions to this manual

The following changes have been made for this release:

- ◆ This manual replaces the former *SUPRA Server Digest*, P26-9065. The contents of that manual have been divided into separate manuals for SUPRA Server PDM and SUPRA Server SQL.
- ◆ All references to SUPRA Server PDM for IBM Release 2.6 have been changed to 2.7.
- ◆ All references to Windows 95 have been changed to read Windows 95/98.
- ◆ All references to MVS have been changed to OS/390.

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 indicates that four spaces appear between the keywords.	BEGN bbb SERIAL
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.	<u>(WAIT)</u> (NOWAIT)
	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>The example indicates that you must enter ON or OFF when using the MONITOR statement.</p>	<p>MONITOR { ON OFF}</p>
<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>	<p>[<u>WAIT</u>] [<u>NOWAIT</u>]</p> <p><u>STATISTICS</u></p>
Ellipsis points...	<p>Indicate that the preceding item can be repeated.</p> <p>The example indicates that you can enter multiple host variables and associated indicator variables.</p>	<p>INTO :<i>host-variable</i> [:<i>ind-variable</i>],...</p>
SMALL CAPS	<p>Represent a keystroke. Multiple keystrokes are hyphenated.</p>	<p>ALT-TAB</p>

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. The example indicates that 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
OS/390	Information specific to a certain operating system is flagged by a symbol in a shadowed box (OS/390) indicating which operating system is being discussed. Skip any information that does not pertain to your environment.	OS/390 SUPRA Server uses the capabilities of the OS/390 operating system

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, see “[SUPRA Server documentation synopses](#)” on page 99.

Overview

- ◆ *SUPRA Server PDM Digest*, P26-9062

Getting started

- ◆ *SUPRA Server PDM Migration Guide (OS/390 & VSE)*, P26-0550*
- ◆ *SUPRA Server PDM CICS Connector Systems Programming Guide (OS/390 & VSE)*, P26-7452

General use

- ◆ *SUPRA Server PDM Glossary*, P26-0675
- ◆ *SUPRA Server PDM Messages and Codes Reference Manual (RDM/PDM Support for OS/390 & VSE)*, P26-0126

Database administration tasks

- ◆ *SUPRA Server PDM and Directory Administration Guide (OS/390 & VSE)*, P26-2250
- ◆ *SUPRA Server PDM Directory Online User's Guide (OS/390 & VSE)*, P26-1260
- ◆ *SUPRA Server PDM Directory Batch User's Guide (OS/390 & VSE)*, P26-1261
- ◆ *SUPRA Server PDM DBA Utilities User's Guide (OS/390 & VSE)*, P26-6260
- ◆ *SUPRA Server PDM Logging and Recovery (OS/390 & VSE)*, P26-2223
- ◆ *SUPRA Server PDM Tuning Guide (OS/390 & VSE)*, P26-0225
- ◆ *SUPRA Server PDM RDM Administration Guide (OS/390 & VSE)*, P26-8220
- ◆ *SUPRA Server PDM RDM PDM Support Supplement (OS/390 & VSE)*, P26-8221
- ◆ *SUPRA Server PDM RDM VSAM Support Supplement (OS/390 & VSE)*, P26-8222
- ◆ *SUPRA Server PDM Migration Guide (OS/390 & VSE)*, P26-0550*
- ◆ *SUPRA Server PDM Windows Client Support User's Guide*, P26-7500*
- ◆ *SPECTRA Administrator's Guide*, P26-9220

Application programming tasks

- ◆ *SUPRA Server PDM DML Programming Guide (OS/390 & VSE)*, P26-4340
- ◆ *SUPRA Server PDM RDM COBOL Programming Guide (OS/390 & VSE)*, P26-8330
- ◆ *SUPRA Server PDM RDM PL/1 Programming Guide (OS/390 & VSE)*, P26-8331
- ◆ *SUPRA Server PDM Migration Guide (OS/390 & VSE)*, P26-0550*
- ◆ *SUPRA Server PDM Windows Client Support User's Guide*, P26-7500*

Report tasks

- ◆ *SPECTRA User's Guide*, P26-9561



Manuals marked with an asterisk (*) are listed more than once because you use them for multiple tasks.



Educational material is available from your regional Cincom education department.

1

SUPRA Server overview

SUPRA[®] Server is an advanced relational database management system based on a three-schema architecture. SUPRA Server provides tools for information retrieval, report generation, application programming, and database management and control.

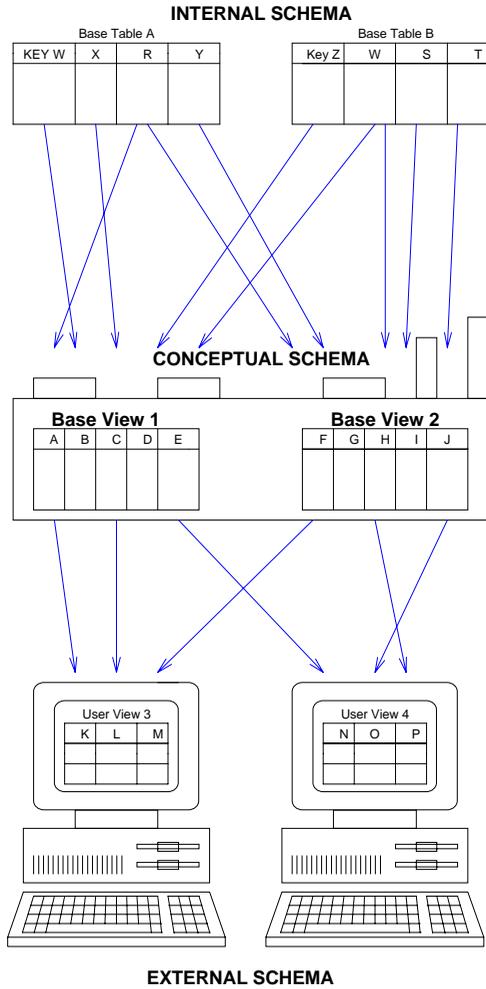
Three-schema architecture

The three-schema architecture implemented by SUPRA Server maintains the table data in three distinct layers:

- ◆ **Internal schema.** Describes the structure of physical files and their fields.
- ◆ **Conceptual schema.** Defines the base logical views that correspond to the physical files. These view definitions specify the implementation of integrity rules within and between views.
- ◆ **External schema.** Describes the logical views whose data is derived from the base views. These derived views describe data for access by applications.

This three-schema architecture means that all data in the database appears to the users as two-dimensional tables. Users can request this data without regard for where or how the data is stored.

The following figure shows the relationships between the three schemas:

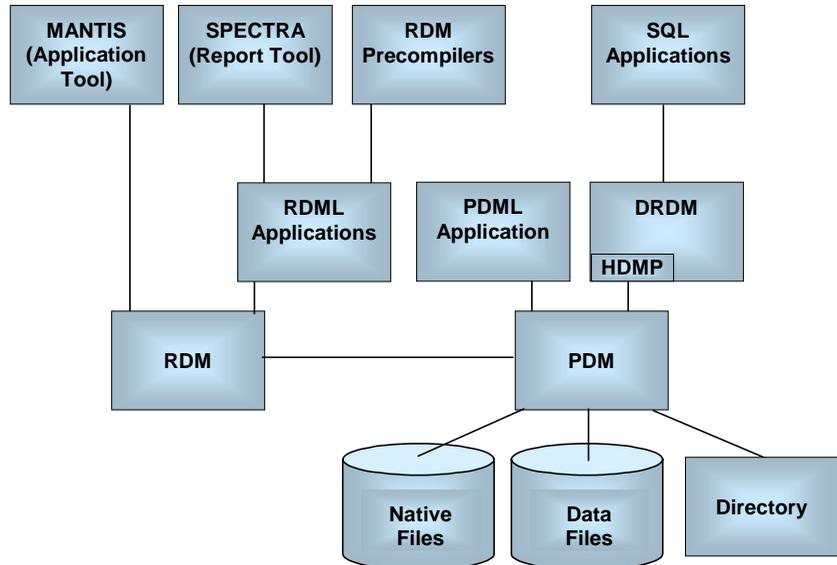


The three-schema architecture of SUPRA Server avoids file-redundancy problems and reduces software maintenance costs.

SUPRA Server components

The following figure shows an overview of SUPRA Server with PDM Server and native file Server support. The key pieces of SUPRA Server include the following:

- ◆ **Physical Data Manager (PDM).** Manages the data structures of the physical files for databases with a PDM Server. PDM files can be accessed through the Relational Data Manager (RDM) or by applications.
- ◆ **Relational Data Manager (RDM).** Processes applications' requests to access PDM data and presents it as though it were arranged in two-dimensional tables.
- ◆ **Directory.** Controls and maintains metadata for PDM databases.



SUPRA Server with PDM support has two options:

- ◆ PDM support
- ◆ PDM/RDM support

SUPRA Server with PDM support

SUPRA Server with PDM support provides access to data through a Physical Data Manager (PDM), which manages the data structures of the physical files that store data. A Relational Data Manager (RDM) extracts data through the PDM and displays the data in two-dimensional tables.

- ◆ **Supported environments.** OS/390, VSE.
- ◆ **How the PDM stores data.** The PDM stores data physically in flat files using either a VSAM or BDAM format.
- ◆ **How you access data.** Use Physical Data Manipulation Language (PDML) to access data. With RDM support, you use Relational Data Manipulation Language (RDML) commands to the RDM, which extracts data from the PDM and displays the data in two-dimensional tables.
- ◆ **Benefits of SUPRA Server with PDM support.** SUPRA Server with PDM support is designed for high-performance, large-scale applications and provides a high degree of control over the storage and maintenance of data.

SUPRA Server with PDM support

The basic element of SUPRA Server with PDM support is the Physical Data Manager (PDM). The PDM maintains data and coordinates database access, processes requests from tools and precompiled programs, and provides backup and recovery tools. The PDM provides a great deal of control over data storage options and is designed for high-volume, online transaction processing.

Components and tools of SUPRA Server PDM

SUPRA Server with PDM support includes the following:

- ◆ **PDM (Physical Data Manager).** A database management system that manages the data structures of the physical files and processes requests for data.
- ◆ **RDM (Relational Data Manager).** An optional database access system that provides relational access to physical files and treats data as though it were arranged in two-dimensional tables.
- ◆ **Directory.** A repository of information about the physical data structures, logical data structures, users, and SUPRA Server implementation options.
- ◆ **SPECTRA.** A relational inquiry-reporting tool that provides the ability to retrieve, manipulate, format, and update data.
- ◆ **CICS connector.** An interface that allows CICS applications to make database calls to SUPRA Server.

SUPRA Server with PDM support provides a variety of optional tools and functions that let you tailor SUPRA Server to your needs and environment. These tools and functions include the following:

- ◆ **DBA utilities.** Tools to organize and maintain data in an efficient, flexible manner. You execute most DBA utilities by coding Utility Command Language (UCL) program input.
- ◆ **MANTIS.** An application programming tool.
- ◆ **Direct VSAM support.** An application that allows SUPRA Server component access to VSAM KSDS/ESDS files through RDM.
- ◆ **Precompilers.** Allow you to embed RDML statements within application programs. SUPRA Server provides precompilers for COBOL and PL/I.

Features of SUPRA Server with PDM support

SUPRA Server with PDM support provides the following features:

- ◆ **Multiple file-type support.** The PDM supports the following types of files:
 - Data files (primary and related, including Directory files) that contain user data and Directory information.
 - Index files that are optional and contain pointers to data file records and secondary key definitions.
 - System files that are optional and are for recording statistics, task level recovery, and system level recovery information.
- ◆ **Active schema (database) maintainability.** You can perform maintenance on entities in an active schema, or database using Directory Maintenance and DBA utilities..
- ◆ **Storage optimization.** The PDM uses experience tables, record blocking, and file buffering to manage and optimize storage space.
- ◆ **Recoverability.** After a failure, you can use task logging and/or system logging to recover the database. Task logging ensures that the PDM saves enough information for all tasks and transactions to perform task level recovery. System logging allows recovery if the task log file is unreadable or unused (on certain environments), or if an updated data file is unreadable and must be reloaded.
- ◆ **Data accessibility.** You can access files in four different ways:
 - Directly by primary key.
 - Directly or sequentially by secondary key.
 - Sequentially (forward or backward).
 - For read-only access to secondary key data.

Features of SUPRA Server with RDM support

SUPRA Server with RDM support provides the following features:

- ◆ **Testability.** The DBAID utility allows you to test RDM views without affecting the directory to ensure they work correctly before putting them into production use. A subset of the DBAID utility is also available to application programmers to test views used in base programs.



CSIDBVER lets you look at PDM data without writing an application.

- ◆ **Data accessibility.** You can access files in four different ways:
 - Directly by primary key.
 - Directly or sequentially by secondary key.
 - Sequentially (forward or backward).
 - For read-only access to secondary key data.
- ◆ **View validation using domains.** A domain is the set of all permissible values for specified fields. You can use domains to indicate whether a value is valid for a given field and whether two fields are logically compatible.
- ◆ **Referential integrity.** Referential integrity ensures that two items representing the same data do not become inconsistent.
- ◆ **Tools for optimizing performance.** You can optimize the performance of the RDM by reducing processing time using Global View Support and View Binding.

2

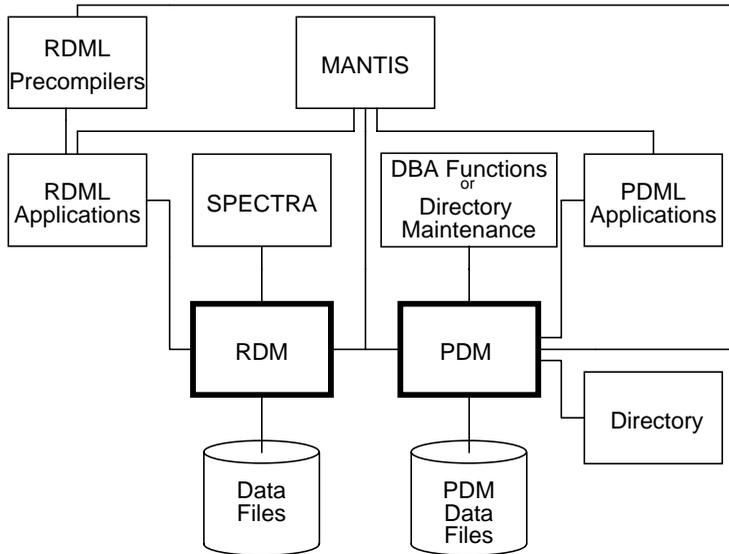
SUPRA Server with PDM and RDM support

SUPRA Server with PDM and RDM support is an advanced relational database management system for high-volume, update-oriented production processing. It contains the following major components:

- ◆ **Physical Data Manager (PDM).** Program that processes requests by RDM and other programs to access native SUPRA Server PDM files (see [“Using the Physical Data Manager”](#) on page 31).
- ◆ **Relational Data Manager (RDM).** Program that processes applications’ requests to access data (PDM, VSAM, or RMS) through logical views (see [“Using RDM to access relations”](#) on page 49).
- ◆ **Directory.** Repository of information about the physical data structures, logical data structures, users, and SUPRA Server implementation options. All requests to the database are handled using information from the Directory (see [“Using Directory Maintenance to define metadata”](#) on page 63).

Overview

The following figure provides an overview of SUPRA Server with PDM support:



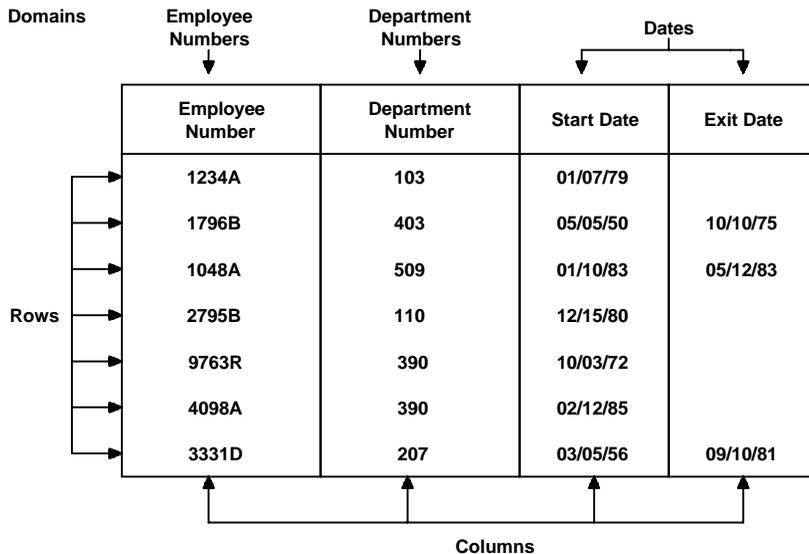
The characteristics and capabilities of SUPRA Server include:

- ◆ SUPRA Server supports a relational data structure enabling you to access data through logical views of relations (tables). These logical views of data have a high degree of independence from the needs of any specific application and from the physical structure of the data files.
- ◆ SUPRA Server has a three-schema architecture that increases the insulation between your applications' views of data and the physical structure of the data files. Changes to file type or structure involve minimal change to logical views and little or no change to application code.
- ◆ SUPRA Server supports several safeguards for the security of your data, protecting it from unauthorized alteration or retrieval.
- ◆ SUPRA Server supports access to a variety of file types. It can access its own native PDM files and native KSDS VSAM files.
- ◆ **OS/390** SUPRA Server uses the capabilities of the OS/390 operating system. SUPRA Server has its own subsystem to coordinate communication between components in different address spaces. SUPRA Server uses extended memory, above the 16 MB line, for certain kinds of work space.
- ◆ The PDM interfaces of SUPRA Server utilize the extended storage capabilities of VSE/ESA and OS/390.
- ◆ SUPRA Server provides for relational and data integrity.
- ◆ SUPRA Server provides for recovery of PDM files.

Understanding the relational data structure

SUPRA Server is a relational database. You can view data as if it were a table consisting of rows and columns. In the following figure, Employee Number, Department Number, Start Date, and Exit Date are column names. The combination of one employee number, one department number, one start date, and one exit date make up a row. All rows for all work histories comprise the Work History relation.

A relation has a primary key (a column or combination of columns whose value(s) uniquely identify each row). The primary key of the Work History relation is the Employee Number column. Each column in a relation must have exactly one domain. The domains in the following figure are Employee Numbers, Department Numbers, and Dates. Each value in a column must belong to that column's domain. A domain is a pool of all permissible values for a given meaning. For example, the Department Numbers domain in the following figure consists of all 3-digit numbers 000–999 that might signify departments.



If your data is organized and viewed as normalized relations, you receive the full benefit of the relational model. Data normalization examines raw data to determine logical relationships between data items and identifies data dependencies. Normalizing data breaks it into unique occurrences and then restructures it into relations. The objective of normalization is to define relations that are independent of process, free of data redundancies, free of data ambiguities, precisely defined, and independent of file structures.

Ensuring security in SUPRA Server

SUPRA Server with PDM support provides database security at the component and the entity levels. You define all users or groups of users on the Directory. The definition includes a nondisplayed user password. The Directory controls who uses your data and how it can be accessed and manipulated. Only authorized users can access Directory Maintenance, DBA Functions, DBAID, SPECTRA, or RDM. Using the relationship commands on the Directory, you can also restrict access to specific views and relations.

Views can also restrict file/relation access. For example, a PDM file opened with shared-update access can be accessible only to a certain user with READ access on a subset of the columns in the relation.

The Security Group and Maintenance Restriction categories on the Directory allow the master database administrator (DBA) to control maintenance to the Directory. The Security Group category holds a Maintenance Restriction entity defining access rules for the related user(s). The Maintenance Restriction category holds rules that permit or deny access to Directory categories, commands, and entities.

Cincom also provides exit points that allow you to insert exit interfaces to external security systems. An exit point is a location in a Cincom product that calls an exit program. The exit program can call and react to the output of an external security system. You may write your own security routines or use another vendor's products, such as ACF II[®], RACF[®], or TOP SECRET[®]. To use an external security system, you must write an exit tailored to your installation's requirements. SUPRA Server components that provide exit points include the PDM, the PDM interfaces, RDM, and the DBA utilities.

Using multiple file types

SUPRA Server with PDM support stores data in multiple user-file types. User files can be:

- ◆ Native PDM files
- ◆ Native KSDS or ESDS VSAM files

The Directory stores detailed, internal-schema information for each file known to SUPRA Server. This information includes:

- ◆ File type
- ◆ Physical access method
- ◆ Data set name and/or ddname
- ◆ Block size or control interval size
- ◆ Length and displacement of the file's key (if applicable)
- ◆ Length and location of the file's secondary key(s) (if applicable)
- ◆ Descriptions of the physical fields in the file's records (names, lengths, displacements, types)

When an application requests data through the RDM, RDM refers to the file information from the Directory. Based on this information, RDM calls the appropriate programs to perform the physical accesses to satisfy the request.

3

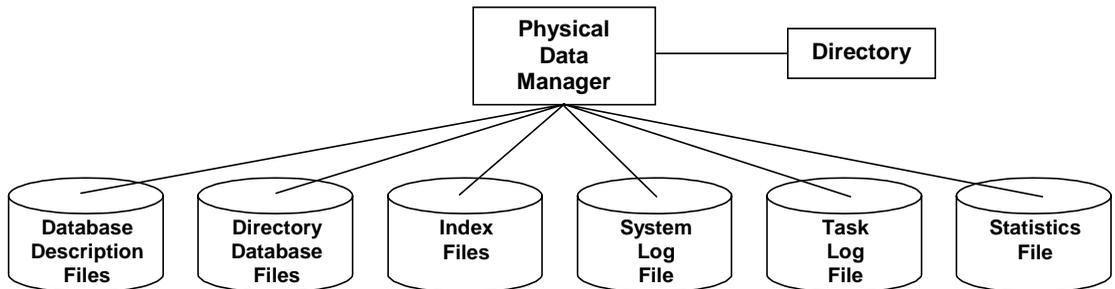
Using the Physical Data Manager

Physical Data Manager overview

The Physical Data Manager (PDM) controls the storage and maintenance of data in SUPRA Server PDM databases. Applications make a request for data to the PDM, which then retrieves the data from the database.

The integration point of this high-performance PDM is the Directory. The Directory stores information about the database and environment managed by the PDM. For more information on the Directory, see [“Using Directory Maintenance to define metadata”](#) on page 63.

The following figure shows the structure of PDM processing and the types of files handled by the PDM:



The PDM supports a complete range of data structure options. These structures allow you to choose the best technique to fit your needs. You may use any of the following:

- ◆ **Hashed.** Provides access to data by hashing of a unique key.
- ◆ **Chained.** Allows the chaining of related data records associated with a unique identifier.
- ◆ **Indexed.** Provides multiple data-access paths.

Hashing and chaining are the default data structures in the PDM. Given a unique identifier for a primary file data record, the PDM processes the identifier's value through a hashing algorithm to determine the location of the physical record and then accesses that record. Given the primary file record, the PDM can find any related file records using the same identifier value. Such records are linked together in chains, and the primary file record points to the first and last related record in the chain.

Dynamic indexing

The PDM uses dynamic indexing to speed the retrieval of data through the use of secondary keys. Dynamic indexing can eliminate intermediate application sorting. You use secondary keys to retrieve data in either forward or reverse order. You can define a secondary key for any data field and get the database records back in order by that data field. If two or more data fields make up a secondary key, the data is returned as if the data fields were a sort key. For example, if you define a secondary key using a product field and a date field, the fields are returned in product and date sequence.

The PDM monitors the activity of applications to ensure that they do not perform unauthorized or invalid functions. If an application tries to perform an invalid function, the PDM terminates the function and backs it out, as if it was partially completed. The following are some examples of invalid functions:

- ◆ Reading or adding a record with an invalid physical key
- ◆ Adding a duplicate primary record
- ◆ Requesting a data set or data item that is not in the database
- ◆ Using an incorrect data-set type
- ◆ Deleting a primary record before all its related data records are deleted

Types of recovery

In addition, the PDM also checks that data sets are opened properly, that they are locked correctly, and that all linkpaths are accurately maintained.

To protect your data from hardware and software failure, the PDM provides task log recovery and system log recovery.

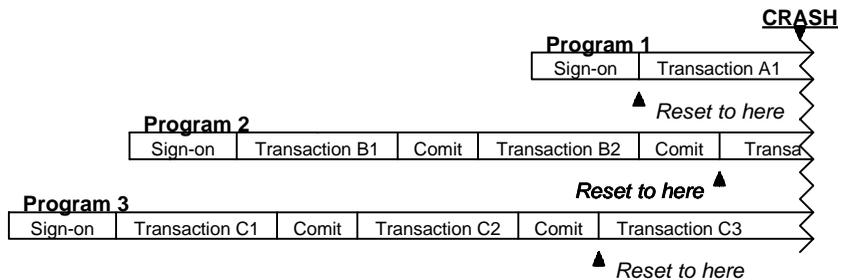
Logging permits recovery, restoration, or resetting of a failed PDM file at a point in the processing cycle where the data is assumed valid. The PDM can record data and function images, and can control information during the processing cycle. Using task logging and task level recovery provides logical integrity and enables restoration of individual tasks and transactions. System logging provides physical database recovery.

Task log recovery

Task log recovery records each successful transaction on a task log. You specify task logging when you define each database on the Directory. To make best use of task log recovery, the application programmer divides each program into logical transactions and inserts a COMMIT statement at the end of each transaction. When the program issues a COMMIT statement, PDM performs three actions:

- ◆ Writes all updates to the database
- ◆ Releases all records held for update
- ◆ Resets the task log file for that task

The following figure illustrates three programs executing at the same time. At the end of each logical transaction is a COMMIT statement. When a program fails, the PDM automatically rolls back processing to the last commit point. The only exception is when a program, as in program 1, fails before the first COMMIT statement. Then the PDM rolls the transaction back to the sign-on.

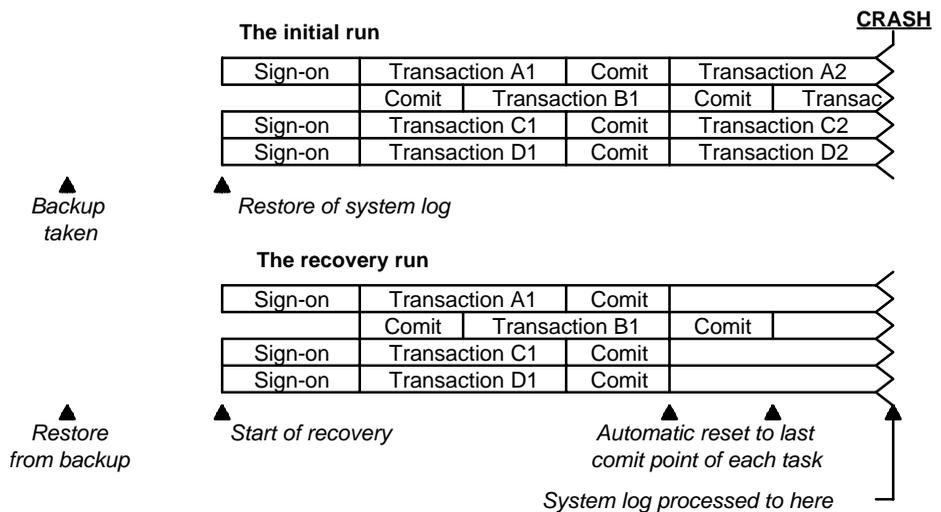


If the system fails, (because of a power failure), the first application to sign on to the database causes the PDM to automatically recover each failed task. In the next figure, the PDM automatically reverses the database updates for Transactions A1, C52, and D3. In other words, the PDM restores the database to its most recent consistent state.

System log recovery

System log recovery logs all completed control functions and images of updated records to a system log. After a failure, system log recovery applies the previously successful updates to a backup copy.

To recover from a system log after a failure, you must first restore the database from a backup copy. Then run system log recovery either online (from DBA command level) or in batch to recover all tasks to the time of failure. Automatic task log recovery then sets all transactions back to their last secure point. The following figure shows the system log applied to a backup database and the subsequent task log recovery.



Using the PDM in OS/390 and VSE environments

Many aspects of the PDM, such as data structures, maintaining data integrity and logging, operate the same regardless of the operating system under which they are running; however, some aspects of the PDM, such as the following, are OS/390 and VSE operating-system dependent:

- ◆ File security
- ◆ Initiating the PDM
- ◆ Operating modes
- ◆ Storage space management
- ◆ File buffering
- ◆ Utilities
- ◆ Restarting a task after system failure
- ◆ Recovering and restoring the PDM after a failure
- ◆ Maintaining the PDM

Maintaining the active schema

The DBA can use the SPECIAL FUNCTION command for a schema to enable maintenance to entities qualified by the active schema. When using active schema maintenance, you cannot delete or rename a consistent file, but you can add a new file definition to the Directory using Directory Maintenance. Then, using the UTILITY OPEN command, the PDM will dynamically obtain the new file definition and allocate the file.

To change an inconsistent file, you must make the required changes to make the file consistent. Then, use the UTILITY CLOSE command and reopen the file. You can perform all maintenance functions on other entities.

Initializing the PDM

During installation, the PDM uses the bootstrap environment to open and read the Directory files. This bootstrap environment consists of a set of independent load modules containing:

- ◆ One supplied bootstrap schema
- ◆ One or more bootstrap environment descriptions that you define
- ◆ One validation module for the bootstrap schema and environment description(s)
- ◆ Logging and recovery information in the event of a system failure

You use three bootstrap utilities to define and maintain the bootstrap environment:

- ◆ Modify Schema utility
- ◆ Create Environment Description utility
- ◆ Create VALMOD utility

After installation, the PDM uses the CSIPARM file to control the system in various operating environments. This file is used during system initialization to determine:

- ◆ Which user schema and environment description(s) to use
- ◆ Which bootstrap Directory schema and environment description(s) to use
- ◆ Operating system's page size
- ◆ Which PDM and interface to use

Operating in different modes

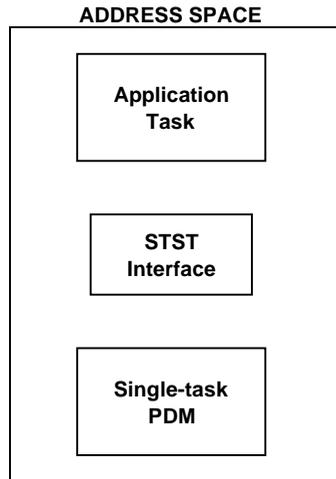
The PDM can use any of these operating modes:

- ◆ Single-task
- ◆ Central
- ◆ Attached
- ◆ Attached central

Using the single-task operating mode

In single-task mode, only one application task is active. This task has exclusive access to the single-task PDM within its same address space through the STST interface. A single-task PDM exists only while the task exists. You initialize the PDM by issuing a SINON command.

The following figure illustrates the single-task operating mode:



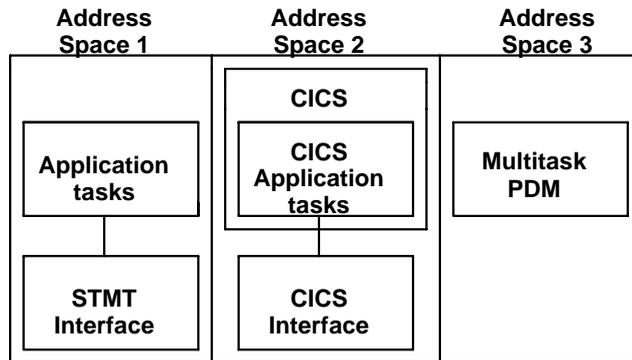
Using the central operating mode

In central mode, the multitask PDM executes and initializes in its own address space, separate from the applications that communicate with it. The following applications can access the same copy of the multitask PDM through the interface (shown in the figure below):

- ◆ Zero, one, or many batch applications
- ◆ Zero, one, or many CICS applications, each with one or more tasks

You must bring up the central PDM before you start any batch applications and before you connect CICS with the PDM.

The following figure illustrates the central operating mode:

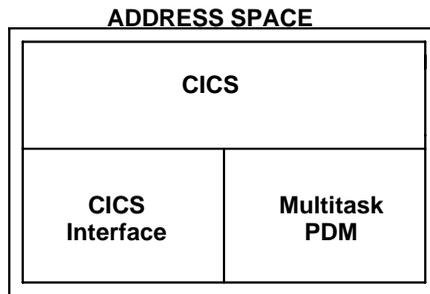


Using the attached operating mode

In attached mode, the application task and a multitask PDM reside in the same address space, but are different operating system tasks. There are two ways to run the tasks in attached mode:

- ◆ Use the Attacher program to attach a PDM and an application as subtasks.
- ◆ Run CICS with the ATTACH parameter in the CSIPARM file. In CICS environments, CICS can attach the PDM as a subtask.

The following figure illustrates the attached operating mode:

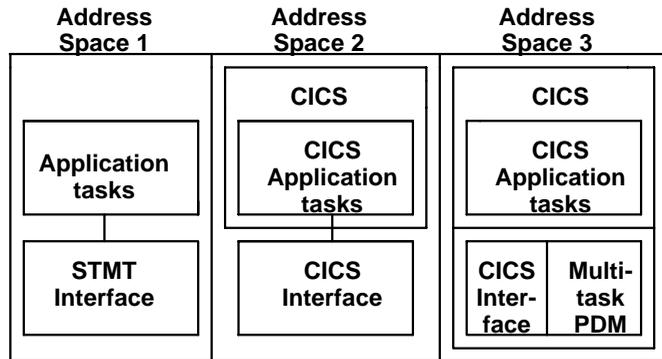


Using the attached central operating mode

In attached central mode, the multitask PDM executes and initializes in the same address space with the initial application task(s). Each additional application task resides in a different address space from the one containing the multitask PDM. All cross-address-space communications between the multitask PDM and an application task use Cross Memory Services.

To run in the attached central operating mode, run CICS with the ATTACH parameter in the CSIPARM file. The PDM is automatically attached and initialized when you issue the CONNECT command.

The following figure illustrates the attached central operating mode:



Managing storage space

The PDM uses the following to manage and optimize storage space:

- ◆ **Experience table.** The PDM uses a file experience table (or bitmap) to track which blocks on a user file are known to be full. When searching the file for free space to add a record, the PDM does not search the full blocks, and, thereby, conserves CPU time and I/O operations.
- ◆ **Record blocking.** A block is a series of contiguous records. You determine the size of a block based on performance and tuning factors. You define the block size on the Directory by entering values for the blocking parameters when adding a file.

When determining the blocking factor, consider the following:

- Larger blocks mean larger and fewer buffers.
- Larger blocks mean more information per track.
- Larger blocks mean less I/O.
- Larger blocks mean slower data transfer, which should be avoided in densely loaded multiprogramming systems.

- ◆ **File buffering.** All files require memory areas to contain physical records (blocks) obtained (read) from or transferred (written) to them. The PDM allows files in the system to share memory areas (buffer pools) or to have unique areas. Shared allocation optimizes storage space use.

You can specify the number of buffers in a pool for direct access use by changing the buffer pool description on the Directory.

Shared buffers minimize the use of storage space; however, keep the following considerations in mind when determining the allocation of files to a buffer pool:

- Buffers being used by PDM files may not be shared by a task log file, system log file, statistics file, or index files.
- The PDM does not allocate any buffers for KSDS VSAM files. Buffering is provided by VSAM.
- The optional Buffer Cache Facility improves PDM performance by using user-defined pools of cache blocks to buffer PDM BDAM files above the 16 MB line, reducing physical reads to files. It also allows you to free the PDM region from certain memory restrictions for users with large numbers of files by reclaiming much of the buffer memory below the 16 MB line. You cannot cache VSAM or BSAM files.
- The Buffer Cache Facility is defined as a user exit in the user environment description. The facility reads an input file containing control information and allocates storage for the cache. During read processing, if the PDM determines that a file block does not exist in a buffer, the PDM looks next to the cache. If found, the block is moved to a buffer. If it is not found, a standard I/O request is issued. Once a physical read is completed, the facility moves a copy of the block from the buffer to the cache. During PDM write processing, the facility updates data in the cache.

Using utilities for PDM files

The PDM file utilities allow you to organize and maintain data in an efficient, flexible manner. Execute the utility functions by coding Utility Command Language (UCL) program input and submitting with a single JCL stream. The following list summarizes the functions of the PDM file utilities:

- ◆ **Expand.** Expands the number of records in an existing BDAM- or ESDS-related file.
- ◆ **Depopulate.** Deletes secondary keys.
- ◆ **File Statistics.** Reports physical and logical characteristics of a PDM file. This allows you to monitor file growth and predict expansion needs. You may also monitor performance by using synonym rates and chain-migration statistics.
- ◆ **Format.** Formats a PDM file, a task log file, and a system log file to fill all records with spaces. The Format function also builds the file control records for primary, related, index, and task log files.
- ◆ **Load.** Adds data records from a sequential file to a primary or related PDM file(s). The sequential input may have been created by the Unload function or by a program that you write.
- ◆ **Log Print.** Prints out a formatted report of the system log file.
- ◆ **Modify.** Allows you to modify specific elements in records in any PDM file and linkpaths in a primary or related file.
- ◆ **Print.** Prints all or selected records from a PDM file.
- ◆ **Recover.** Recovers one or more PDM files, based on information in the system log file, if a software or hardware failure occurs.
- ◆ **Restore.** Restores one or more data files, based on information in the system log file, if a software or hardware failure occurs. The Restore function applies after images to the database until the restore point is encountered.

- ◆ **Reorganize.** Rebuilds the secondary key structure.
- ◆ **Review.** Examines the specified files to see if they are locked, then prints an appropriate message.
- ◆ **Sorted-Populate.** Sorts files and then constructs an index making secondary keys available for use.
- ◆ **Unload.** Extracts all or selected records from a PDM file and writes them to a sequential output file. The output file records are built in a format that is compatible with the Load function.
- ◆ **Unlock.** Resets locks in a PDM file that did not go through the normal PDM close logic due to an abend or system failure.
- ◆ **Version 2 Unload.** Extracts records from SUPRA Server converted and native PDM file(s) or Directory files at high speed and writes them to a sequential output medium appropriate for the Version 2 Load utility.
- ◆ **Version 2 Load.** Formats SUPRA Server converted or native PDM files and writes data records at high speed from a sequential medium (the output from a Version 2 Unload function) to the files.
- ◆ **Version 2 Insert Linkpath.** Inserts linkpath data into SUPRA Server converted and native PDM files or Directory files without reloading primary files. The input for this utility must be the output of the Version 2 Load utility.
- ◆ **PDM Termination.** Allows you to terminate a central PDM.
- ◆ **Execution Statistics.** Prints the contents of the statistics file, which is produced when the PDM calculates and accumulates specific system statistics for the RSTAT command (read statistics) or when the PDM is initialized or terminated.

Restarting after a task failure

A task failure may occur in one or more tasks; for example, one CICS task, all of CICS, the PDM itself, or a single batch task. For a task failure, the PDM recovers and restores your database.

When the PDM is using task level recovery and is running under CICS, CICS will indicate to the PDM when a single task fails. The PDM recovers the PDM files to the last commit point. If the CICS step fails, or if a batch task fails, the PDM recognizes the failure and recovers the PDM files to the last commit point for all tasks associated with the interface that failed. In either case, the PDM frees all PDM resources held by the task and maintains both physical and logical database integrity.

Restarting a task after a system failure

A system failure will abend any task running under CICS. To recover CICS resources for all active tasks after a system failure, perform an emergency restart. CICS resources are recovered to the last syncpoint for each task. PDM resources are recovered to the last commit point for each task.

Signed-on tasks which are not committed are recovered to the point of sign-on and are then signed off. All other tasks remain active.

Recovering and restoring the PDM after a system failure

If the PDM is forced to terminate abnormally for any reason, you should normally run the PDM job again. The PDM will warm start, automatically using the task log file to recover its data files to the last commit point of each active task. If the task log file or your data files are physically damaged, you must run the Recover or Restore function of the PDM utilities.

You run the Recover and Restore functions of the PDM utilities on the system log file after a failure occurs. The Recover function begins at the end of the file and applies before images until it reaches a commit point (or quiet if no task logging). The Restore function starts at the beginning of the file and applies after images. If you plan to restore files, you must back up your PDM data files. If you plan to use the Recover function only, the backup is not necessary.

If you are running under CICS and CICS abends, the PDM automatically recovers SUPRA Server resources when it detects the CICS abend. If the PDM is in attached operating mode, the PDM recovers when it is restarted. If you emergency restart CICS, you must reconnect the CICS connector to the PDM. If you selected the Reconnect option during CICS initialization, this step is automatic. Otherwise, you use the CICS CONNECT operator control command.

Maintaining the PDM using Interactive Services

To help you maintain the PDM, you can use the Interactive Services programs. The Interactive Services Main menu provides the following options:

- ◆ **Physical File Services.** Allows you to see the physical file's data set name, type, mode, access method, block size, logical record length, control interval size, and so on. You can also display file statistics that the PDM gathers for the file you select.

- ◆ **PDM Services.** Allows you the following options:
 - PDM System Information shows the PDM's name, type, and operating mode.
 - PDM System Statistics displays execution statistics.
 - Active Environment Information shows information about the environment the PDM is running in.
 - Task Management allows you to display information about tasks and interfaces; allows you to purge an inactive task, or purge a task whose interface is no longer connected.

4

Using RDM to access relations

SUPRA Server with PDM support provides relational access to physical files. The Relational Data Manager (RDM) treats data as though it were arranged in two-dimensional tables (relations) with rows and columns. Logical views define tables for access by RDM.

User applications issue commands in the SUPRA Server Relational Data Manipulation Language (RDML) to RDM. RDM then formulates and issues the appropriate call to the PDM.

With RDM, you can access database information as you view it without concern for its physical location or structure. You can change and restructure your physical database without rewriting or recompiling application programs. RDM handles database navigation, data integrity, data security, and data validation for application programmers and end users.

With RDM, you can access your data files in different ways:

- ◆ Directly by primary key
- ◆ Sequentially (forward or backward)
- ◆ Directly or sequentially by secondary key

Designing and building views

Logical views define relations (tables) consisting of rows (tuples, logical records) and columns (attributes) for access by RDM. Each RDML command must specify a view. In general, a single RDML verb accesses a single row in a relation.

There are two types of views:

- ◆ **Base views.** These access physical files. Base views are part of the conceptual schema.
- ◆ **Derived views.** These access other views. Derived views are part of the external schema. User views are a subset of derived views.

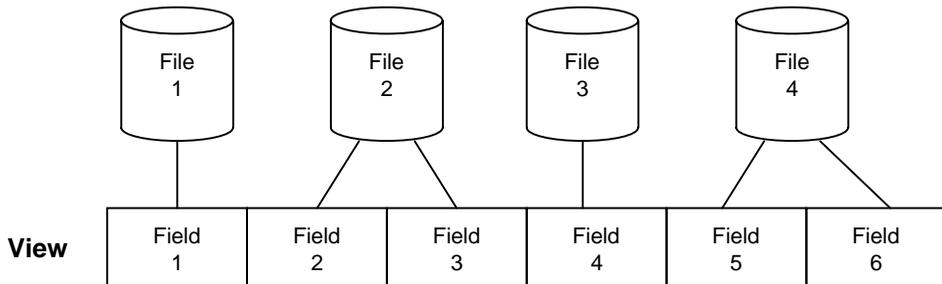
RDM base views can access files of different types, including Cincom's Physical Data Manager (PDM) files and native KSDS VSAM files. The DBA defines views on the Directory for the data in these files. A single view can access files of different types.



A program should not access the same files through both the PDM and the RDM.

Refer to the *SUPRA Server PDM RDM PDM Support Supplement (OS/390 & VSE)*, P26-8221, and the *SUPRA Server PDM RDM VSAM Support Supplement (OS/390 & VSE)*, P26-8222, to explain how to create base views for access to physical files.

A single view can draw fields from several different physical files as shown in the following figure:



A derived view can draw fields from one or several base views. For example, in the following figure, the EMPLOYEE-INFORMATION base view contains 7 fields. If you only need a portion of that data for particular views, you can define user views specifying only the fields you need. The EMPLOYEE-MAIL-ADDRESS derived view uses 4 fields from the base view, and the EMPLOYEE-PAY-RATE derived view uses three. Thus, you retrieve only the data you need for a particular task.

**EMPLOYEE-
INFORMATION
View**

Employee Number.	Employee Name	Employee Address	Employee Exemption	Employee Hourly Rate	Employee Social Security Number	Employee Department Number
------------------	---------------	------------------	--------------------	----------------------	---------------------------------	----------------------------

**EMPLOYEE-
MAIL-ADDRESS
User View**

Employee Number	Employee Name	Employee Department Number	Employee Address
-----------------	---------------	----------------------------	------------------

**EMPLOYEE-
PAY-RATE
User View**

Employee Number	Employee Social Security Number	Employee Hourly Rate
-----------------	---------------------------------	----------------------

If your applications use derived views rather than base views, you get both the benefit of the three-schema architecture and the insulation from changes to the physical structure.

You can create views with the help of the following software:

- ◆ Online DBAID
- ◆ Batch DBAID
- ◆ Online Directory Maintenance
- ◆ Batch Directory Maintenance

All views are stored on the Directory. You can use Directory Maintenance, DBA Functions, or DBAID to assign your views to specific users or groups of users. You can use the Directory reports to show the definition of a view, who has access to a view, and which programs use the view.

Views can be used with DBAID, MANTIS, SPECTRA, COBOL, PL/I, and FORTRAN applications.

Building view definitions with components

A view definition consists of column definitions and access definitions. The column-definition portion of the view precedes the access-definition portion.

When RDM retrieves a row (logical record) using the view, RDM returns one value (or null) for each defined column (except for a constant column, one with the qualifier CONST or UNIQUE CONST). The physical order of the values in the row after retrieval is the same as the order of the column definitions in the view. The chronological order in which the values are retrieved for the row is determined by the access definitions.

You must provide one or more access definitions as part of your view definition. An access definition provides RDM with directions for accessing the desired file(s) or view(s). (Base views access files, derived views access views.)

The following is an example of a base-view definition. Each line except the last defines a column. The last line is the access definition.

```
View name:  BRANCH
KEY  BRANCH-NUMBER
      BRANCH-NAME
      BRANCH-ADDRESS
      BRANCH-CITY
      BRANCH-STATE
      BRANCH-ZIP-CODE
REQ  BRANCH-REGION
      BRANCH-DELIVERY-ROUTE
      BRANCH-SALE-ROUTE
      BRANCH-STAFF-QUOTA
ACCESS BRANCH WHERE BRANCH-NUMBER = BRANCH-NUMBER ALLOW ALL
```

Validating view columns with domains

A domain is the set of all permissible values for a column. Domains are important because they let the business rules be defined and enforced without program maintenance. To create domains on the Directory, a DBA can use Directory Maintenance. The DBA creates domains using a list of permitted values or by specifying a combination of data-field size, data type, and data-value range. Each domain must be identified by a unique name.

Domains provide two types of validation:

- ◆ They indicate whether a value is valid for a given field
- ◆ They indicate whether fields are logically compatible

For example, if a customer number is a 5-digit decimal integer, its field should be assigned to a domain defined as the set of all 5-digit decimal integers. RDM can then reject values of the wrong length and type. If an employee number is a 5-digit decimal integer, its field should also be assigned to a domain defined as the set of all 5-digit decimal integers; however, these two fields should not be assigned to the same domain, because they are not logically comparable. It makes no sense to search for an employee with the same number as a customer.

When RDM processes a request to add or update a row, it verifies that the new value is valid by checking the domain for the column. If the value is invalid, RDM returns an error status to the application.

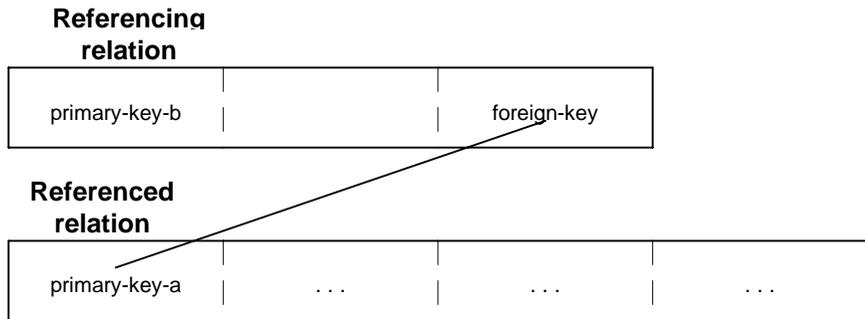
When a view definition indicates that a column in one relation represents the same data as a column in another relation (the foreign key in a referencing relation and the primary key in the referenced relation), RDM verifies that the fields for those columns belong to the same domain.

Adding integrity constraints to base views

If a combination of column(s) in one relation represents the same data as the primary key in a second relation, then:

- ◆ The column(s) in the first relation comprise a foreign key.
- ◆ The first relation is a referencing, or source, relation.
- ◆ The second relation is a referenced, or target, relation.

The following figure shows how one relation references another with a foreign key:



You must identify any foreign keys in your base views so that RDM can maintain referential integrity. Referential integrity ensures that two items representing the same data do not become inconsistent.

RDM supports the following referential integrity rules:

- ◆ A foreign key value must exist in the referenced relation as a primary key. In other words, a primary key value must exist for each foreign key value in a referencing relation.
- ◆ Null values can be allowed for a foreign key.

RDM checks for referential integrity in two ways:

- ◆ **Foreign key value integrity.** When inserting or updating a row that contains a foreign key, the foreign key value must either point to a valid primary key in the referenced relation or be null. Otherwise, RDM will not perform the insert or update.
- ◆ **Deletion integrity.** RDM will not delete a row in a referenced relation if the row's primary key is referenced by one or more foreign keys. All foreign keys with that primary key's value must be deleted or set to null before RDM can delete the referenced row.

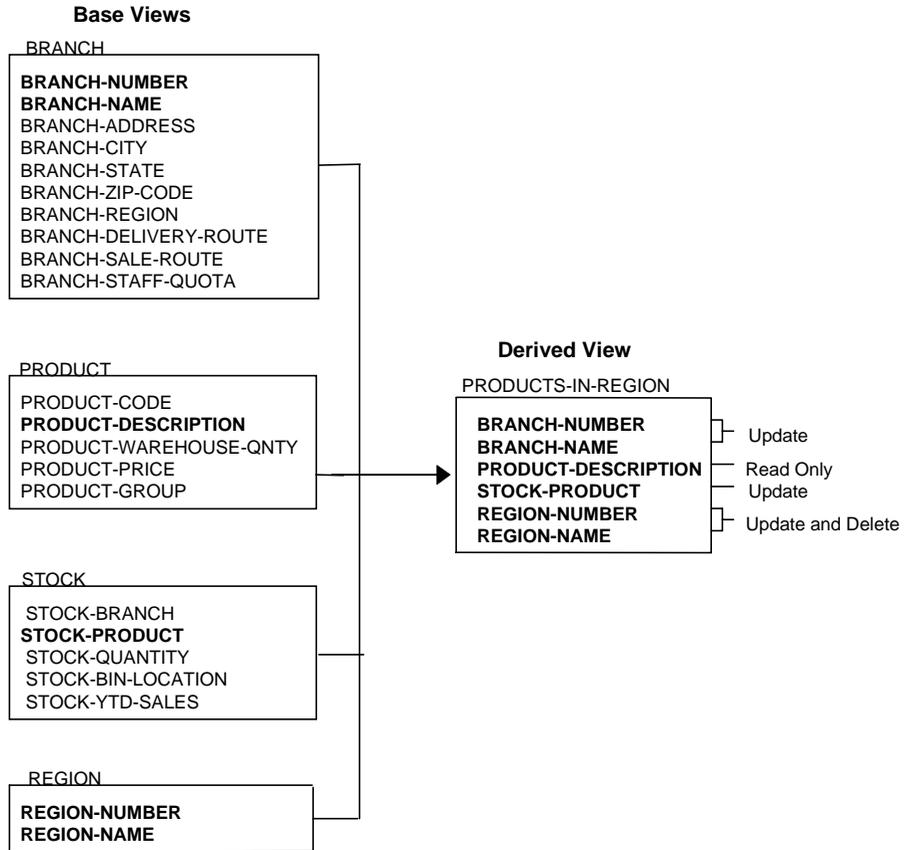
Creating derived view definitions

Derived views access other views instead of accessing physical files. An ACCESS clause specifies a view name instead of a physical file name.

For example, the following derived view PRODUCTS-IN-REGION draws data from the base views BRANCH, PRODUCT, STOCK, and REGION to list all the products in stock in a region.

```
View name:  PRODUCTS-IN-REGION
KEY  REGION-NUMBER
      REGION-NAME
KEY  BRANCH-NUMBER
      BRANCH-NAME
      STOCK-PRODUCT
      PRODUCT-DESCRIPTION
ACCESS REGION WHERE REGION-NUMBER = REGION-NUMBER ALLOW UPDATE
      DELETE
ACCESS BRANCH WHERE BRANCH-REGION = REGION-NUMBER ALLOW ALL
ACCESS STOCK WHERE STOCK-BRANCH = BRANCH-NUMBER ALLOW ALL
ACCESS PRODUCT WHERE PRODUCT-CODE = STOCK-PRODUCT
      AND STOCK-PRODUCT = STOCK-PRODUCT
```

The following figure illustrates the base views, the derived view, and the Update options specified for the derived view:



A derived view can have more restrictive access than its source views, but not less restrictive access. For example, if the base view allows only read access, the derived view cannot allow updates to a field from the base view, even if the derived view specifies ALLOW UPDATE.

When defining a derived view, you need not enter the access definitions for the integrity constraint. You need not rewrite a derived view if the physical file for the BRANCH relation is broken apart into different files or put into another file with a different name.

Accessing data with views

Application programmers access a view through program logic. With RDM, these programs are insulated from the physical environment, including the Physical Data Manager (PDM), data structures, and access and navigation strategies. A programmer uses the Relational Data Manipulation Language (RDML) precompiler to compile source statements into executable code. RDM refers to information from the Directory to retrieve the information required for the view and to access files of any supported type. Supported file types include native SUPRA Server Physical Data Manager (PDM) files and IBM native KSDS VSAM files.

The programmer can write validation user exits to access other types of files. Programmers can write applications using VMS Basic, FORTRAN, COBOL, PL/I, or MANTIS and can access views on the Directory using simple RDML instructions within their programs. Programmers can manipulate data in a program using the four RDML commands: GET, INSERT, UPDATE, and DELETE.

End users can use views online through SPECTRA, a query facility with update capabilities. See [“Using SPECTRA to retrieve data”](#) on page 83.

Optimizing performance

You can optimize the performance of RDM in several ways:

- ◆ **View binding.** This allows you to store an open version of a view on the Directory. Binding reduces processing time for view requests that cause a view to be opened.
- ◆ **Global View support.** This allows you to have certain views open when the database starts. This saves RDM the processing overhead of opening views when the application program first accesses them.
- ◆ **Storing the RDM program in shared memory.** (LPA for OS/390, SVA for VSE.) This allows all applications to share the same copy of the RDM program. This saves application memory space and tends to reduce the paging rate.

You can put some work memory, the RDM program, and global views into extended memory (above the 16 MB line).

Maintaining current programs

RDM uses several checks to ensure that your program is current and that any views it uses are the same as other separately compiled modules in the application program. When an application program issues an RDML command, RDM checks to see if the view, as defined in the Directory, is still correct for this program's use. The checking includes column existence, column length, type and number of decimal places. If the view is not correct, RDM returns an error status; then the program must be recompiled.

Application systems are often composed of several separately compiled programs which depend on a common definition of data terms. These programs call each other to perform special tasks. RDM checks on each RDML call to ensure that the definition of the view is the same for each program. If a program or subroutine is compiled with the same view as another program or subroutine and the view definition does not match, RDM generates an error message. The column (attribute) list generated by the RDML precompiler at precompile time contains the data used to perform this error checking.

Insulating programs from change

RDM insulates application programs from changes to the physical database. For example, moving a column (attribute) from one relation to another has no impact on application programs. In most cases, you can change a view design without affecting the application programs. For example, adding a new column to a view does not affect a program.

Changes to relations include changing a relation type. Usually, RDM insulates application programs from these types of changes. However, the view must be modified and rebound, if previously bound, and reglobalized, if previously globalized, or both.

Physical changes (changing the characteristics of a column in a view) may require changes to the program logic and recompilation. Not every program using a view needs to be recompiled; only recompile the programs that use the column in their user view. The view must also be modified and rebound.

Adding or deleting indexes, secondary keys, or linkpaths may change the behavior of unbound, unglobalized views. RDM selects the access strategy at view open time, and the addition of secondary keys or indexes may alter this selection. If you want a bound or global view to take advantage of a newly defined index or secondary key, you must first rebind the bound view or reglobalize global views or do both.

Creating and testing views with DBAID

It is important to define and test your views to ensure they work correctly before putting them into production use. The DBAID utility allows you to do the following:

- ◆ Define a new view without affecting the Directory.
- ◆ Open the view.
- ◆ Issue RDML statements.
- ◆ Examine the results.

You can then change the view if necessary, reorder the view for efficiency, or try different navigation methods until the view works the way you want it to.

If the Directory is available for update, you can instruct DBAID to save your view onto the Directory with the SAVE command. As soon as the SAVE is processed, the view is available to application programs unless a bound version of the view exists. You can bind a view using the DBAID command BIND. You can also take existing views from the Directory, change them for new requirements, and test them to verify that they still work, without affecting the view stored on the Directory or the Global View file.

A subset of the DBA commands is available to application programmers. This subset allows application programmers to use the DBAID utility when constructing programs using views. Programmers can use the DBAID utility to see how the views work and to determine how to design the application based on this information.

Access to the DBAID commands is controlled by the name used for sign-on. If the user is the DBA, as defined on the Directory, then all of the commands are recognized by DBAID. However, if the signed-on user is not the DBA, only certain commands are available. The application programmer cannot define new views or edit existing views. The programmer can access only those views related to the user ID in the Directory file or the Global View file and can access only the data available through those views.

You can use DBAID in a batch or online environment. DBAID provides the following types of commands:

- ◆ **System.** Display information about the currently executing DBAID utility. These include commands to display current users, to display active views, and to save a view created through DBAID.
- ◆ **Editing.** Change existing view definitions and create new views for testing.
- ◆ **RDML.** Test a defined view against the database.
- ◆ **View Display.** Display the fields and descriptions for an open view.
- ◆ **Statistics.** Gather and print statistics.

Obtaining RDM reports and statistics

The RDM reports provide information on views and the application programs that use them. You can report on all views for all users or select reports on specific views or users. RDM provides a Report utility that generates the following reports that you can run separately or simultaneously:

- ◆ **DBA Report.** Helps you track the views you have defined on the Directory. This report also includes a list of the users who can use the view.
- ◆ **Programmer's Report.** Provides all the needed information about a view.
- ◆ **End-user Report.** Provides an abbreviated report intended for the nontechnical end-user. It shows the users related to the view, the sequence number of each column, and the column names and descriptions.
- ◆ **Impact of Change Report.** Shows any changes you make to relations or base views that can impact derived views, application programmers, or end users.
- ◆ **Views and Programs Report.** Provides a list of programs used by each view and the time and date of the last successful update.

In addition to these reports, RDM reports and prints statistics on a task basis. Statistics show the number of RDML requests made by the task and the number of PDM requests made by RDM when processing RDML requests. These statistics can be used as a tool to test the efficiency of your views.

5

Using Directory Maintenance to define metadata

The Directory provides the central source of control for your PDM user and system files. The Directory provides a database interface that defines operational requirements and separates the user's logical requests from the physical requirements of the database. The Directory contains information to:

- ◆ Establish and maintain security and privacy measures for both Directory files and PDM files.
- ◆ Generate logical and physical entities from relational data.
- ◆ Define and describe any new, reorganized, or restructured PDM files and their contents.
- ◆ Provide dictionary capabilities and data dictionary information on Directory reports.
- ◆ Define secondary key entities needed for automatic file indexes.
- ◆ Maintain relationships and structures between entities in various categories.
- ◆ Define the operating requirements and physical characteristics of PDM files.
- ◆ Perform specific maintenance processing on the categories of information contained on the Directory files.
- ◆ Define and describe relations.
- ◆ Define and describe views.

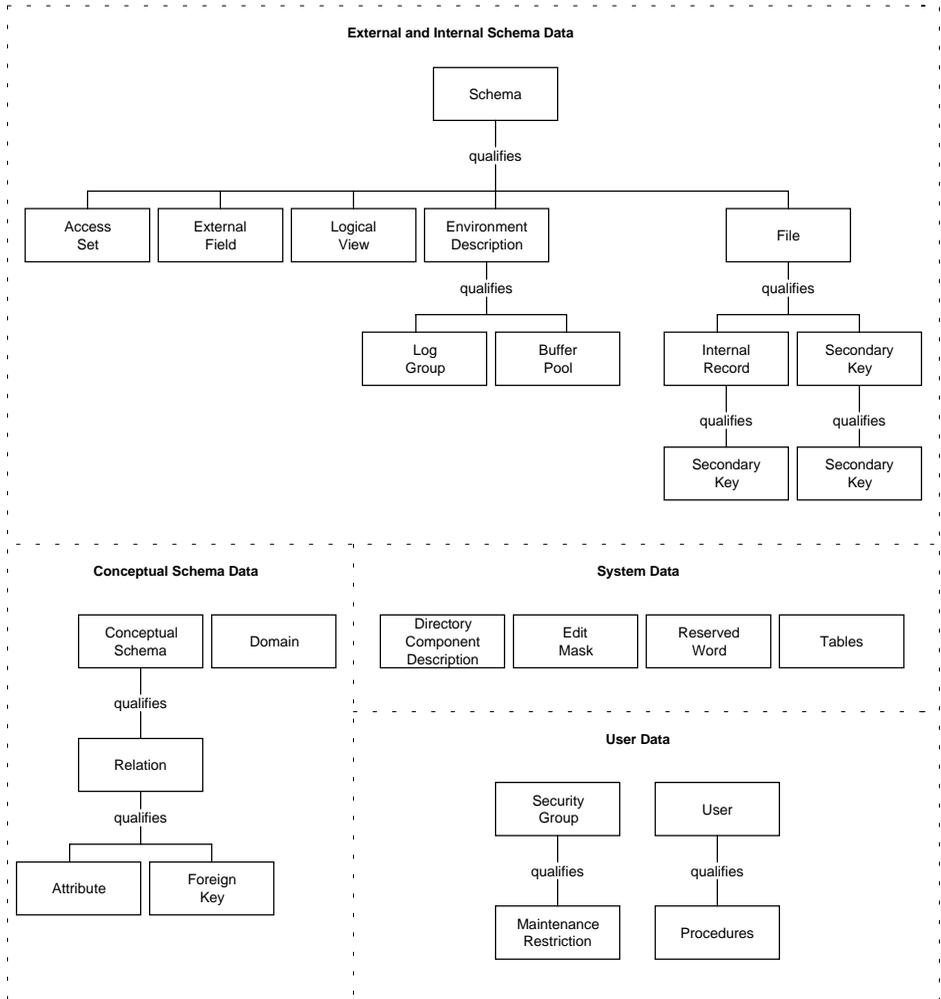
You use Online or Batch Directory Maintenance to establish, display, modify, or delete entities and relationships on the Directory.

Defining entities and relationships

The Directory maintains the entities and their relationships within five types of data:

- ◆ Conceptual schema data
- ◆ External schema data
- ◆ Internal schema data
- ◆ System data
- ◆ User data

The Directory maintains conceptual schema, external schema, and internal schema data entities in a hierarchy. The following figure shows the entities maintained by the Directory and the hierarchy of these entities:



Conceptual schema data includes the following entities:

- ◆ **Attribute.** Defines the characteristics of associated domains and relations.
- ◆ **Conceptual schema.** A collection of relations that define the database in general, relational terms.
- ◆ **Domain.** A set representing an idea and containing all valid values of an attribute. It specifies the function of the entity and provides validation options.
- ◆ **Foreign key.** Identifies the attribute or combination of attributes in one relation that are defined across the same domains as the attribute(s) that compose the primary key in the same or another relation.
- ◆ **Relation.** Defines a two-dimensional table represented by rows (tuples) and columns (attributes). A row is a collection of columns. Each row in a relation contains the same columns but differing values. One column or set of columns in the relation is defined as the primary key and uniquely identifies each row.

External schema data includes the following entities:

- ◆ **Access set.** Defines navigational information needed by RDM to access data.
- ◆ **External field.** Describes the external appearance of a data element along with a physical field name on a PDM file for use with RDM.
- ◆ **Logical view.** Contains information on a particular view of the database and indicates whether RDM can use this view.

Internal schema data includes the following entities:

- ◆ **Buffer pool.** Describes the file I/O buffers used by the PDM.
- ◆ **Environment description.** Contains execution-time information that describes the operating environment to the PDM.
- ◆ **File.** Describes the physical characteristics of your PDM files.
- ◆ **Internal record.** Defines the physical record layout, including the base portion and the redefined portion of a record.
- ◆ **Key code.** Identifies a Physical Field or a combination of physical fields as a secondary key for an internal record.
- ◆ **Log group.** Connects one or more system log data sets to form a logical system log.
- ◆ **Physical field.** Describes the physical characteristics of a database element, including storage and format information.
- ◆ **Schema.** Defines the global view of your database. The global view includes the logical (external) and physical (internal) definitions.
- ◆ **Secondary key.** Used to provide indexed access to a PDM file. A secondary key identifies a key code or combination of key codes used as the key for indexing.

System data includes the following entities:

- ◆ **Directory component description.** Provides execution options for Directory Maintenance.
- ◆ **Edit mask.** Contains the format mask used to edit the physical representation of data into a format suitable for displaying or printing.
- ◆ **Reserved word.** Defines a word with a predefined meaning to the system. This word cannot be used in any other capacity.
- ◆ **Tables.** Defines table values used to convert or validate field values.

User data includes the following entities:

- ◆ **Maintenance restriction.** Defines a set of rules that permits or denies access to Directory entities.
- ◆ **Procedure.** Defines retrieval procedures used by RDM.
- ◆ **Security group.** Contains a collection of maintenance restrictions that enforces access rules for the related user.
- ◆ **User.** Contains information about each user of Directory Maintenance, SPECTRA, and RDM.

You use Directory Maintenance commands to establish, display, modify, or delete entities or relationships on the Directory. Each command has a 2-character code. The following are examples of the commands and their functions:

- ◆ **CHECK (CK).** Verifies consistency of your entity descriptions.
- ◆ **RELATE (RL).** Establishes relationships between entities.
- ◆ **SPECIAL FUNCTION (SF).** Permits or denies active schema maintenance.
- ◆ **STRUCTURE DISPLAY (SD).** Displays or prints the names of entities that are subordinate to or related to the specified entity.
- ◆ **UTILITIES (UT).** Performs functions on PDM files, including populating secondary keys on index files.

Creating directory entities in a hierarchy

As shown in the figure under “[Defining entities and relationships](#)” on page 64, the Directory maintains conceptual schema, internal schema, and external schema data entities in a hierarchy.

Before you can perform maintenance on an entity, you must enter each qualifying entity, or the “naming data.” The naming data shows how a given entity is qualified (made unique and, therefore, addressable) and how the Directory categories are arranged in a hierarchy.

When you perform online maintenance on an entity, Directory Maintenance displays the Enter Naming Data prompt on the Command menu to identify the information you must enter. When you perform batch maintenance, you use Naming Data Transactions to enter the naming data. In both cases, once you have specified the naming data entities, those names are carried over through subsequent entries until changed.

A conceptual schema is the primary entity for conceptual schema data. All conceptual schema entities are ultimately qualified by the conceptual schema. The conceptual schema name qualifies relations, and relations qualify attributes, primary keys, and foreign keys. You cannot add an attribute or foreign key unless the relation to which it belongs already exists. Only a domain entity is not qualified by a conceptual schema and can exist by itself.

A schema is the primary entity for external schema and internal schema data entities. All of these entities are ultimately qualified by the schema, with other levels of qualification for some entities.

You cannot perform maintenance on an entity until you have established all qualifiers. For example, you cannot add a physical field to the Directory unless the schema, file, and internal record to which it belongs already exist.

Establishing entity relationships

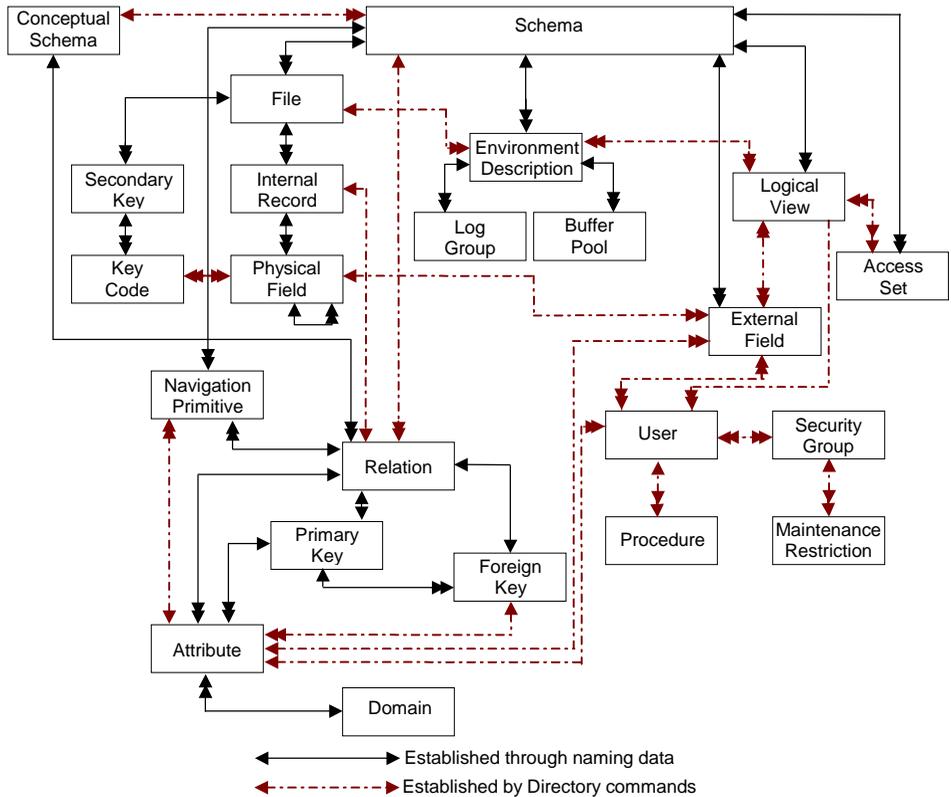
Directory Maintenance establishes two types of relationships:

- ◆ **Required.** Established through naming data.
- ◆ **Optional.** Established using Directory commands.

Some relationships are qualification relationships created automatically when entities are added. These qualification relationships are automatically erased when certain entities are deleted.

Other relationships are maintainable. You may use the RELATE (RL) and REMOVE (RM) commands to maintain these relationships. Some relationships have data associated with them. This is called relationship data. You use the RELATIONSHIP DISPLAY (RD) and the RELATIONSHIP CHANGE (RC) commands to maintain this relationship.

The following figure shows the relationships and how they are established. Single and multiple arrows indicate one-to-many relationships. For example, a schema may contain multiple files, but a file can be defined only once in a given schema.



Maintaining directory security

The Directory maintains three entities that allow you to control access:

- ◆ **User.** Enables you to define information about each user of Directory Maintenance, SPECTRA, and RDM. A user must be defined in the Directory to be permitted access to the system. A user must also be related to logical views that may be used in the user's RDM applications. You can define a user as a DBA to allow the DBA to access additional information not available to other users.
- ◆ **Maintenance Restriction.** Defines a set of rules that restrict access to Directory entities through Batch or Online Directory Maintenance.
- ◆ **Security Group.** Contains a collection of maintenance restrictions that enforce defined rules on the related user.

By defining maintenance restrictions and security groups, the DBA can "permit" or "deny" a user or group of users access through Directory Maintenance to any entity or category of entities on the Directory or use of any Directory command.

Using Online Directory Maintenance

You use Online Directory Maintenance to maintain your entities and relationships. During an online maintenance session, you use four types of screens to maintain data:

- ◆ Sign-on screen
- ◆ Category menu
- ◆ Command menu
- ◆ Entity data screen

These screens prompt you for the data that you need to enter.

Sign-on screen

The first screen displayed during an online session is the Sign-on screen. You must enter a valid user ID and password before you are allowed to perform Directory Maintenance functions.

```
TIS/XA DIRECTORY MAINTENANCE

*****
**                SIGN-ON                **
*****

DBA ID:
PASSWORD:
```

Category menu

When your user ID and password are validated, Directory Maintenance displays the Category menu.

```
TIS/XA DIRECTORY MAINTENANCE

ENTER CATEGORY:

AS  ACCESS SET                LV  LOGICAL VIEW
AT  ATTRIBUTE                 MR  MAINTENANCE RESTRICTION
BP  BUFFER POOL              PF  PHYSICAL FIELD
CS  CONCEPTUAL SCHEMA      PR  PROCEDURE
DC  DIRECTORY COMPONENT DSC  QC  QUERY COMPONENT DESC
DM  DOMAIN                   RE  RELATION
ED  ENVIRONMENT DESCRIPTION  RW  RESERVED WORD
EM  EDIT MASK                SC  SCHEMA
FI  FILE                     SG  SECURITY GROUP
FK  FOREIGN KEY              SK  SECONDARY KEY
IR  INTERNAL RECORD          TA  TABLES
KC  KEY CODE                 US  USER
LG  LOG GROUP                XF  EXTERNAL FIELD

ACTIVE SC:                    ACTIVE ED:                    MAINTENANCE PERMITTED:
```

This screen shows the Directory entity categories and category codes. Your active schema and environment description will be displayed. Y or N indicates whether you can perform maintenance on entities qualified by the displayed active schema.

Command menu

After you select a category, Directory Maintenance displays a Command menu that lists the available commands and command codes and the required entity-naming data for the selected category.

This example shows the Command menu for the file category:

```

PHYFLD: COMMANDS          TIS/XA DIRECTORY MAINTENANCE

AD  ADD                   RD  REL DSPLY
CG  CHANGE                RL  RELATE
CK  CHECK                 RM  REMOVE
CO  COPY                  RN  RENAME
DE  DELETE                SD  STRU DSPLY
DI  DISPLAY               SE  SHORT EDIT
LE  LONG EDIT             ST  SHORT TEXT
LT  LONG TEXT             UT  UTILITIES
RC  REL CHANGE

ENTER SELECTION CODE:          SUBCATEGORY CODE:

ENTER NAMING DATA:
SCHEMA:          DEMOSCHM
FILE:           REGION
INTERNAL RECORD:
PHYSICAL FIELD:

ACTIVE SC:          ACTIVE ED:          MAINTENANCE PERMITTED:

```

The top line of the screen displays the selected category, the center portion of the screen displays the commands you can use, and the bottom portion identifies the required naming data. Messages are displayed on the last line of the screen.

Entity data screens

After you enter valid naming data on the Command menu, Directory Maintenance displays an entity data screen for those functions that require entity data. This example shows the Entity Data screen for the file category:

```
FILE: ADD                TIS/XA DIRECTORY MAINTENANCE
SCHEMA: DEMOSCHM        FILE: REGION                1 OF 3
LAST UPDATE  09.54.23 01/01/1993 V: 0001  USER: THOMAS
DDNAME:                REGION
DATA SET NAME:         USER.DATASET.NAME
FILE TYPE:             PRIMARY
FILE ACCESS METHOD:     BDAM
FILE DEVICE TYPE:     3360
FILE DEVICE ASSIGNMENT:
LOGICAL RECORD LENGTH: 100
TOTAL LOGICAL RECORDS: 6000
TOTAL TRACKS:         0
RECORDS PER BLOCK/CI: 0
BLOCKS PER TRACK:     3
VSAM CONTROL INTERVAL: 0
TOTAL VSAM CNTRL INTVL: 0
COMMAND:
```

The top of the screen displays the category and command being processed and the naming data for the entity qualifiers. The next line in the top portion displays audit information from the previous update. This information includes the time and date of the last update, the number of changes made to this entity, and the ID of the last person to update this entity. The center portion of the screen displays the data for this entity. You may enter values for any field, or you may use any of the displayed default values. A command field located at the bottom of the screen is used for paging if you are using a set of two or more screens, or for error message specification. The last line on the screen is used for messages.

Using Batch Directory Maintenance

You can use Batch Directory Maintenance when you have large volumes of input that would be time consuming to enter online, to maintain an active schema used during normal operation hours, to do required maintenance, or to process functions that require a batch program to generate input.

Input statements

For a Batch Directory Maintenance job, you can use three types of input statements to maintain data:

- ◆ Run Option Definition
- ◆ Comment
- ◆ Command

The input format for all types of statements is one or more 80-character fixed-length records. Positions 73–80 are reserved for optional sequence numbers and are ignored by the input processor unless you select the Sequence Number Check option.

Run Option Definition statements

Run Option Definition statements define any special options you want in effect while processing your transactions. These statements contain a plus sign (+) in the first position followed by a keyword indicating the option. You can submit Run Option Definition statements at the beginning of each run. Some run options remain in effect for the entire run; you can change others between commands. Some possible options are:

- ◆ **Continue on error.** Continues processing the next command after an error has been detected.
- ◆ **Eject.** Starts a new page with each transaction or prints continuously.
- ◆ **Null character.** Specifies the character to be used to set the field to blank or zero.
- ◆ **Print suppress.** Records only images of input statements entered or complete data.
- ◆ **Sequence number.** Verifies sequence numbers on input statements.
- ◆ **Syntax check.** Provides syntax validation without updating the PDM Directory.

Comment statements

Comment statements can contain any information you want to enter. Comment statements contain an asterisk (*) in the first position followed by the applicable comment in positions 2–72. You can include these statements with Command statements or Run Option Definition statements. However, they must be positioned before or after all records for a given command or Run Option Definition statements because a Comment statement terminates the input for a transaction.

Command statements

Command statements perform the maintenance functions on the Directory. A Command statement contains a 2-character command code in the first two positions, followed by a 2-character category code in positions 4 and 5. The entity name and attributes begin in the seventh position and continue over as many records as needed. If you use multiple records for a function, only the first record contains the command and category codes. Positions 1–6 must be blank on all subsequent records for that function.

For example, once you have specified the naming data, adding a foreign key entity requires two input statements. The first input statement contains the AD command code, the FK category code, and the foreign key name. The second input statement contains the name of the relation containing the primary key, the foreign key type, and whether the rows should be clustered or chained.

Input Statement

Pos. 1-2	Pos. 4-5	Pos. 7-36	
-------------	-------------	--------------	--

AD FK CUSTOMER-NAME

Input Statement

	Pos. 7-36	Pos. 38	Pos. 59	Pos. 61	
--	--------------	---------	------------	------------	--

ORDERS

ASSIGNED

Y

N

Batch processing

Your batch input statements must pass the same validation and execution routines as online input. You first submit a sign-on statement that contains the user ID and password. Directory Maintenance validates this transaction before you can perform any maintenance processing.

Directory Maintenance then processes the category and command statements. You must include statements that qualify the entity to be processed within the hierarchy; these statements are called Naming Data Transactions. Once you enter this naming data, the names are saved and carried over to subsequent transactions until you change them. These names that are saved are referred to as *sticky fields*. Directory Maintenance validates names at the start of processing for each command.

Directory Maintenance processes input records in the sequence in which they are entered. All records, including comments, are printed. If an input record contains a category and entity name but no command, Directory Maintenance validates the category, and the name becomes a sticky field. If a valid command is specified, command processing is initiated. If the category, command, or naming data does not pass validation, Directory Maintenance prints error messages and continues processing with the next transaction.

After initial validation, the parameter values you enter are processed. If you do not specify a value for a field, Directory Maintenance uses a default value, if one is defined for the field, or an existing value. Directory Maintenance validates and prints parameter values and, if any errors are detected, prints error messages. When all records have been processed, Directory Maintenance prints statistics for the run and terminates.

Batch Directory Maintenance output is designed to closely resemble Online Directory Maintenance output.

Batch output

At the end of a Batch Directory Maintenance run, the system prints out a report of the completed or attempted processing. The printout lists all valid input records and includes input record numbers. If Directory Maintenance detects input parameter errors, the printout shows a short error message to the right of the data and a long error message at the end of the transaction.

The following statistics are printed at the end of the job:

- ◆ Number of statements read
- ◆ Number of commands processed
- ◆ Number of commands executed
- ◆ Number of errors
- ◆ Maximum error code

You can submit List Control options between command statements to control the content and format of batch output. The Print Suppress option determines whether the output will include only the input-record-image list and any error or completion messages, or that information plus any associated data such as all attribute values for the function requested. The Eject option determines if each logical transaction will begin on a new page or if printing will be continuous or contiguous.

Using the Directory reports

The DBA can generate Directory reports to monitor Directory performance and structure. These reports can serve as a dictionary by defining each entity (and its use and relationships) on the Directory.

The Directory reports provide information for every entity maintained on the Directory. When requesting Directory reports, you use the name shown in parentheses below. You may choose a full report, or you can select or suppress particular categories or entities.

- ◆ **DBA User Reference Report (DIRUSERP).** Lists user descriptions and includes the logical views, relations, security groups, procedures, and associated information available to each user.
- ◆ **Directory Conceptual Schema Structure Report (DIRCSSTP).** Produces three subreports for related schemas, related relations, and all domains in the Directory.
- ◆ **Directory External Fields Where Used Report (DIRXFWUP).** Lists external fields within a schema along with their physical fields, internal records, and files.
- ◆ **Directory Full Definition Report (DIRDEFNP).** Generates subreports for most entities on the Directory. Each subreport lists the entities within a specified category and their descriptions.
- ◆ **Directory Full Text Description Report (DIRTEXTP).** Lists entity names and their qualifiers along with the associated short/long text.
- ◆ **Directory Name Cross Reference Report (DIRXREFP).** Lists each entity name, its category, qualifiers, date and time of the last update, and the user ID for the last maintenance session for that entity.
- ◆ **Directory Name Report (DIRNAMEP).** Lists all entity names and provides the qualifier's definition number, the definition number of each entity, and the category of each entity.
- ◆ **Directory Physical Structure Report (DIRSTRUP).** Provides the name, definition number, qualifier definition number, and linkpath information for each entity within the Directory.
- ◆ **Directory Schema Structure Report (DIRSCSTP).** Produces four subreports describing the full explosion of each schema's structure.
- ◆ **Directory Security Group Report (DIRSECGRP).** Produces two subreports that describe security group-to-user relationships.

6

Using SPECTRA to retrieve data

SPECTRA is a sophisticated report-writer and database-access-retrieval tool. It is an optional component of SUPRA Server with SQL support, and it is a standard component of SUPRA Server with PDM support.

Overview

With SPECTRA, you can use simple English commands to select and produce reports on the data contained in SUPRA Server tables and views. You can create complex reports including customized titles, statistics, totals, and grand totals. SPECTRA is a fully interactive system with online documentation and Help facilities.

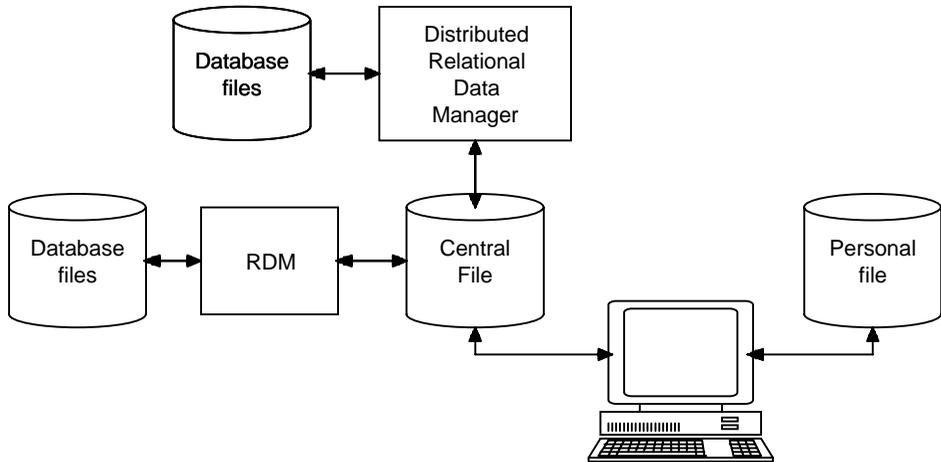
When accessing table data, SUPRA Server controls database navigation, security, and integrity. SUPRA Server rejects any updates that are not allowed, and SPECTRA returns an explanation to the user. SPECTRA also ensures that any data requirements are met by verifying the data requests you enter before processing them.

SPECTRA accesses SUPRA Server tables and views as its central files. SPECTRA can also access VSAM files as a source of central files.

With SPECTRA, you can create personal files containing data copied from central files or entered from other sources. Your personal files are not available to other users unless you share the files.

You use SPECTRA commands to create a set of instructions (a process) to perform tasks. You can write processes to request information, perform calculations, enter data, create reports, update personal files, and so on. You can store processes for repeated use. You can also copy and modify an existing process to create a new process for similar tasks.

You can use SPECTRA in batch mode. Any command available with online SPECTRA is available in batch. You can perform all SPECTRA functions, such as creating and running processes, listing files, issuing HELP commands, and creating and printing personal files. OS/390, VSE and CICS users can also submit a process to run in batch mode while using online SPECTRA. You can write and test your processes online and then use the SUBMIT control command to submit the process to run in batch mode. The following figure shows the relationship between central files, personal files, and processes:



Selecting an option from the Master menu

You begin a SPECTRA session by selecting one of the following options from the Master menu:

- ◆ **Central files.** SPECTRA lists any central files you are authorized to access. You may select one of the listed files and use SPECTRA commands to:
 - Browse and edit the central file (edit is only available for RDM views).
 - Copy central file data to a personal file.
- ◆ **Personal files.** SPECTRA lists any files you have created. You can use SPECTRA commands to:
 - Create, delete, rename, or copy a file.
 - Change the file layout and field descriptions.
 - Browse and edit file contents.
 - Move or copy data between files.
 - Add or delete fields in a file.
- ◆ **Processes.** SPECTRA lists your defined processes. You can use SPECTRA commands to:
 - Create, delete, change, or rename a process.
 - Select an existing process to perform a task using process commands.
- ◆ **User's guide.** SPECTRA displays the table of contents for the online user's guide.

Splitting screens for dual use

You can divide a SPECTRA screen into two sections (or windows). Each window has its own command line, and you may perform two operations simultaneously. For example, you can edit using one window and display a file process or the Help menu in the other window.

After you unsplit the screen, you can switch between windows by entering EXCHANGE. You may then perform operations on the displayed screen without affecting the other window—both windows remain available.

Using SPECTRA commands

SPECTRA provides two types of easy-to-use commands to browse data and to define your reports:

- ◆ **Control commands.** Use to perform immediate actions at the terminal:
 - Browse and select central and personal file data
 - Copy central file data into a personal file
 - Dynamically update personal file data
 - Print files, processes, and help text
 - Copy, delete, and rename personal files and processes
 - Split the screen and display two windows
 - Exchange one screen for another
 - List names of central and/or personal files or processes
 - Print selected data
 - Sort personal files
 - Back up and restore personal files and processes
- ◆ **Process commands.** Use to create a set of instructions that perform tasks such as:
 - Creating reports
 - Entering data
 - Updating personal file or RDM data
 - Performing calculations
 - Retrieving information

Using the specialized facilities of SPECTRA

SPECTRA offers the following facilities to help you manipulate your data and provide meaningful reports:

- ◆ Built-in functions
- ◆ Arithmetic operators
- ◆ Statistics
- ◆ Comparison symbols and operators
- ◆ Display control
- ◆ Null support

Customizing reports with built-in functions

SPECTRA has several built-in functions that you can use in a process to customize titles and footers and to control paging. The following functions are available:

- ◆ **DATE.** Prints the current date.
- ◆ **TIME.** Prints the current time.
- ◆ **WEEKDAY.** Prints the current day of the week.
- ◆ **LINECOUNT.** Counts the number of lines.
- ◆ **PAGECOUNT.** Counts the number of pages.

Calculating statistics

SPECTRA provides the following statistical functions:

- ◆ **ANY.** Returns YES if any of a set of records meets a stated condition; returns NO if all records fail to meet the condition.
- ◆ **AVERAGE.** Computes the average value of a series of numeric values.
- ◆ **COUNT.** Counts the number of times a specific value occurs.
- ◆ **EVERY.** Returns YES if every record in a set of records meets a stated condition; returns NO if any records fail to meet the condition.
- ◆ **MAX.** Prints the maximum value for a field in a set of records.
- ◆ **MIN.** Prints the minimum value for a field in a set of records.
- ◆ **TOTAL.** Adds a series of values you select in a set of records.

Comparing values

SPECTRA allows you to select data using one or more of the following comparison symbols or operators:

Symbol	Definition
=	Selects a value that is equal to the specified value.
<	Selects the values less than the specified amount.
>	Selects the values greater than the specified amount.
<>	Selects all values other than the specified value.
NOT=	Selects values not equal to the specified value.
<=	Selects values less than or equal to the specified value.
NOT>	Selects values not greater than the specified values.
>=	Selects values greater than or equal to the specified value.
NOT<	Selects values not less than the specified value.
AND	Selects values based on two different comparisons which must both be true.
BETWEEN	Selects values that fall between two specified values.
CONTAINS	Searches a textual value for the specified pattern.
ENDS	Searches for fields ending with a specified value.
MAYBE	Selects values that are null or that fall within the selection criteria.
OR	Selects values based on two different comparisons. If either comparison is true, the value is selected.
SAME	Selects rows with the same values from two or more files.
STARTS	Searches for fields beginning with a specified value.

Controlling the display

You can use the following special characters in a process to control the displayed data:

Character	Definition
&	Joins (concatenates/connects) text values together.
TRIM	Removes blanks from text values.
@	Extracts parts (substrings) of textual information.
:	Controls the output display. You can specify the column width and/or supply an edit mask for the column.

Using null support

SPECTRA supports null values in both personal files and central files. SPECTRA displays null values as blanks. SPECTRA tracks whether a field value is missing and whether it is blank or zero.

You can use the NULL keyword as follows:

- ◆ In Edit and Browse mode to select database on a null value.
- ◆ In processes to retrieve data from central and personal files based on whether a field contains a null value. You can also update, delete, and insert null values into SPECTRA personal files.

Running a SPECTRA process

The following screens show how to run an existing SPECTRA process. The screens show a process that creates a report on each manifest number. After signing on to SPECTRA, you select the function you want to perform from the Master menu. SPECTRA prompts you through all subsequent steps.

The following screen shows the Master menu. You enter a 3 to view a list of processes.

```
Welcome to SPECTRA>
==> 3

SELECT ONE OF THE TOPICS BELOW. TYPE THE NUMBER AND PRESS ENTER.

New users should first read the User's Guide.

For assistance, type Help at the command line (==>) and press ENTER.

  1 CENTRAL FILES      Lists the central files available to you.
  2 PERSONAL FILES    Lists your personal files.
  3 PROCESSES         Lists processes available to you.
  4 USER'S GUIDE      Provides a complete guide to SPECTRA.

(c) Cincom Systems, Inc. 1992
All Rights Reserved

1=HELP 2=TOP 3=END 4=EX 5=SPLIT 6=INPUT 7=P 8=NEXT 9=MARK 10=GET 11=MOVE 12=PUT
```

On the screen that lists the available processes, you can enter an abbreviated version of the SHOW command to display the process EXP-TOTAL-1.

```

Ready
==> sh 33
ITEM ... PROCESS NAME                ITEM ... PROCESS NAME .....
 1 $INPUT                             21 EXP-REPT-6
 2 EXP-ARITH-1                         22 EXP-REPT-7
 3 EXP-ARITH-2                         23 EXP-SELCT-1
 4 EXP-ARITH-3                         24 EXP-SELCT-2
 5 EXP-ARITH-4                         25 EXP-SELCT-3
 6 EXP-JOIN-1                          26 EXP-SORT-1
 7 EXP-JOIN-2                          27 EXP-SORT-2
 8 EXP-JOIN-3                          28 EXP-SORT-3
 9 EXP-NULL-1                          29 EXP-SORT-4
10 EXP-NULL-2                          30 EXP-TERM-1
11 EXP-NULL-3                          31 EXP-TERM-2
12 EXP-RDML-1                          32 EXP-TERM-3
13 EXP-RDML-2                          33 EXP-TOTAL-1
14 EXP-RDML-3                          34 EXP-TOTAL-2
15 EXP-REPT-1                          35 EXP-TOTAL-3
16 EXP-REPT-1A                         36 EXP-WRITE-1
17 EXP-RDML-2                          37 EXP-WRITE-2
18 EXP-RDML-3                          38 EXP-WRITE-3
19 EXP-REPT-2                          39 EXP-WRITE-4
20 EXP-REPT-5                          40 X-AND1

1=HELP 2=TOP 3=END 4=EX 5=SPLIT 6=INPUT 7=P 8=NEXT 9=MARK 10=GET 11=MOVE 12=PUT

```

SPECTRA displays the process. Notice that this process performs a join, selecting data from two SQL tables. You enter GO to execute the process.

```
Process EXP-TOTAL-1
==> GO

.....TEXT.....|Ax
*EXP-TOTAL-1
GET AMOUNT = QUANTITY * PRICE
  FROM A=SQL>CDB:MANIFEST B=SQL>CDB:MANIFEST_ITEM
  WHERE A.MANIFEST_NO = B.MANIFEST_NO
REPORT GROUPED BY A.MANIFEST_NO
TITLE 'MANIFEST' A.MANIFEST_NO ';'
  LEFT 'ORDER ' ORDER_NO
  RIGHT 'DATE ' ENTRY_DATE : 'MM/DD/YY'
PRINT
  INVENTORY_ITEM_NO AS 'ITEM_NO'
  QUANTITY PRICE : 'FZZ,ZZ9.99' AMOUNT : 'FZZ,ZZ9.99'
WHEN A.MANIFEST_NO CHANGES
  (PRINT COUNT INVENTORY_ITEM_NO TOTAL AMOUNT : 'FZZ,ZZZ,ZZ9.99' PAGE)

1=HELP 2=TOP 3=END 4=EX 5=SPLIT 6=INPUT 7=P 8=NEXT 9=MARK 10=GET 11=MOVE 12=PUT
```

SPECTRA displays the resulting report. Notice that it provides a count of the item numbers and a total of the amount.

```
APRIL 1ST, 1993 12:30:59 PAGE 1
ORDER 2001131      MANIFEST 10      DATE 06/01/91

ITEM_NO  QUANTITY  PRICE      AMOUNT
-----  -
1001121   175        $222.25   $38,893.75
1001183   450        $162.00   $72,900.00
2
                                * COUNT *
TOTAL AMOUNT: $111,793.75
Press enter for next page
==>
```

Managing SPECTRA administrative functions

The DBA is responsible for handling SPECTRA administrative functions. These functions include:

- ◆ Accessing external files
- ◆ Maintaining user profiles
- ◆ Maintaining the Personal File System
- ◆ Recovering from failure
- ◆ Producing User Profile reports

Accessing external files

The SPECTRA INPUT, OUTPUT, IMPORT, EXPORT, IMPORTALL, and EXPORTALL commands allow SPECTRA to access external files.

The INPUT and OUTPUT commands allow you to use an external file within a process for reporting. You use the INPUT command to read data from an external file into a SPECTRA process. You use the OUTPUT command to write data from a SPECTRA process to an external file. INPUT and OUTPUT describe the file layout including the length of each field.

The EXPORT, IMPORT, EXPORTALL, and IMPORTALL commands allow you to copy personal files or processes to or from an external file. You use these commands to back up your data or to share data with other SPECTRA users.

Maintaining user profiles

SPECTRA user profiles provide an additional layer of security for SUPRA Server users. To use SPECTRA with SUPRA, you must be defined both as a SUPRA Server user and as a SPECTRA user. The SPECTRA user definition is maintained in the user profile.

The DBA defines the SPECTRA user profiles, which control feature availability for each user. SPECTRA stores user profiles on the USER_PROFILES personal file, which can only be accessed by the DBA.

Each user profile provides flags which the DBA activates or deactivates. A user profile also provides the user name, language, date and time default formats, a default printer, several user-defined fields, and statistics and accounting information.

The user flags specify whether a user can:

- ◆ Read central files during EDIT
- ◆ Read central files from a process
- ◆ Change central file contents during EDIT
- ◆ Change central file contents with a process
- ◆ Create personal files
- ◆ Create a process
- ◆ Execute a process
- ◆ Execute a process online
- ◆ Show the contents of a process
- ◆ Access external files
- ◆ Import files

The statistics and accounting information for a user includes the following:

- ◆ Maximum number of calls allowed to central, personal, external, or sort files for batch processing. If the maximum limit is reached, SPECTRA stops executing.
- ◆ Maximum number of seconds SPECTRA should run a request (moving data or running a process) before SPECTRA interrupts.
- ◆ Maximum number of calls allowed to central, personal, external, or sort files for online processing. If the limit is reached, SPECTRA interrupts.
- ◆ How many records can be processed after the initial online interrupt message before another message is issued.
- ◆ How much disk space is available for personal files and processes.
- ◆ How many blocks can be processed for sort files before a process is canceled.
- ◆ Current count of personal file blocks used.
- ◆ Number of records processed from central files, personal files, external files, and sort files.
- ◆ Total amount of time, in seconds, the user has been signed on to SPECTRA.

Maintaining the personal file system

SPECTRA personal files exist in a single RMS file referred to as the PFS file. The contents of each block in the PFS file are managed by the SPECTRA personal file system. The DBA is responsible for maintaining this hierarchy of files. Refer to the *SPECTRA Administration Guide*, P26-9220, for additional information on maintaining this file system.

Recovering from a failure

SPECTRA manages a buffer pool that is written to the PFS file at various commit points, and SPECTRA guarantees the PFS file to be intact to the most recent commit point. Refer to the *SPECTRA Administration Guide*, P26-9220, for additional information on using the PFS file and other means for recovery.

Producing user profile reports

SPECTRA provides four processes you can use to produce reports about user profiles. You can use these processes as they are, or you can copy and modify them to use as a base for your own reporting. The four reports are:

- ◆ **Users.** Reports on all of the SPECTRA users.
- ◆ **Statistics.** Reports on the user statistics.
- ◆ **Blocks.** Reports on the percentage of the allocated blocks a user has used.
- ◆ **Authorization.** Reports on each user's authorization.



SUPRA Server documentation synopses

The following synopses provide general information about the SUPRA Server PDM manual series. Some of the SUPRA Server tools are optional; therefore, you may not have all the manuals listed. The documents are listed by publication number within each series.

PDM/RDM documentation series (OS/390 and VSE)

The following are synopses for the manuals included in the PDM/RDM documentation series for OS/390 and VSE environments:

- | | |
|-----------------|--|
| P26-0126 | <i>SUPRA Server PDM Messages and Codes Reference Manual (PDM/RDM Support for OS/390 & VSE)</i> |
| Audience | General |
| Synopsis | Explains the messages issued and codes returned by SUPRA Server. Prescribes the action to take in response to a message or code in OS/390 or VSE environments. |
| P26-0225 | <i>SUPRA Server PDM Tuning Guide (OS/390 & VSE)</i> |
| Audience | Database administrators or systems programmers |
| Synopsis | Explains how to tune your SUPRA Server system for optimum performance. |
| P26-0550 | <i>SUPRA Server PDM Migration Guide (OS/390 & VSE)</i> |
| Audience | Database administrators |
| Synopsis | Explains how to migrate your database to the current version of SUPRA Server from a previous version, or from certain versions of TIS or TOTAL. |

- P26-0675** ***SUPRA Server Glossary***
- Audience** General
- Synopsis** Alphabetically lists SUPRA Server terms with their definitions.
-
- P26-1260** ***SUPRA Server PDM Directory Online User's Guide (OS/390 & VSE)***
- Audience** Database administrators
- Synopsis** Describes the screens, commands, and options involved in maintaining the SUPRA Server Directory with Online Directory Maintenance. This manual is a companion to the *SUPRA Server PDM and Directory Administration Guide (OS/390 & VSE)*, P26-2250.
-
- P26-1261** ***SUPRA Server PDM Directory Batch User's Guide (OS/390 & VSE)***
- Audience** Database administrators
- Synopsis** Describes the commands and options involved in maintaining the SUPRA Server Directory with Batch Directory Maintenance. This manual is a companion to the *SUPRA Server PDM and Directory Administration Guide (OS/390 & VSE)*, P26-2250.
-
- P26-2223** ***SUPRA Server PDM Logging and Recovery Guide (OS/390 & VSE)***
- Audience** Database administrators
- Synopsis** Explains how to use SUPRA Server logging. Explains how to recover or restore SUPRA Server files or tasks from the log files. Covers the following topics:
- ◆ Creating the PDM task log file
 - ◆ Creating PDM system log file groups
 - ◆ Running SUPRA Server with logging
 - ◆ Recovering SUPRA Server CICS applications

- P26-2250** ***SUPRA Server PDM and Directory Administration Guide (OS/390 & VSE)***
- Audience** Database administrators
- Synopsis** Provides reference and task information needed to administer the SUPRA Server PDM and the SUPRA Server Directory. It covers the following topics:
- ◆ Overview of the PDM and Directory
 - ◆ Defining the database
 - ◆ Modifying the bootstrap environment
 - ◆ Adding entities to the user environment
 - ◆ Operating modes
 - ◆ Logging
 - ◆ Tuning for optimum performance
 - ◆ CSIPARM file parameters
 - ◆ Initializing, operating, and terminating the PDM
 - ◆ Expanding the Directory
 - ◆ Using the TIS/XA Software Selection Facility
 - ◆ Defining security
 - ◆ Generating Directory reports
 - ◆ Copying Directory information from one set of Directory files to another
 - ◆ Maintaining the PDM with Interactive Services
 - ◆ Using TIS/XA subsystem commands
 - ◆ Descriptions of important macros
 - ◆ Sample environment descriptions
- P26-4340** ***SUPRA Server PDM DML Programming Guide (OS/390 & VSE)***
- Audience** Database administrators
- Synopsis** Provides reference information on certain physical Data Manipulation Language (DML) commands for controlling the SUPRA Server PDM.

P26-6260	<i>SUPRA Server PDM DBA Utilities User's Guide (OS/390 & VSE)</i>
Audience	Database administrators or systems programmers
Synopsis	<p>Describes the utilities for maintaining your SUPRA PDM database files. Describes utilities for accomplishing the following:</p> <ul style="list-style-type: none">◆ Formatting files◆ Unloading, reloading, or expanding files◆ Populating, depopulating, or reorganizing an index file for a secondary key◆ Recovering or restoring files◆ Reviewing the lock status of files◆ Unlocking files◆ Printing reports on the contents of log files or statistics files◆ Printing or modifying the contents of files
P26-7452	<i>SUPRA Server PDM CICS Connector Systems Programming Guide (OS/390 & VSE)</i>
Audience	Systems programmers
Synopsis	Explains how to use the IBM CICS macros to create the CICS tables needed to run SUPRA Server under CICS.
P26-8220	<i>SUPRA Server PDM RDM Administration Guide (OS/390 & VSE)</i>
Audience	Database administrators
Synopsis	<p>Explains how to administer the SUPRA Server RDM. Covers the following topics:</p> <ul style="list-style-type: none">◆ Managing and maintaining the RDM◆ Designing and constructing views◆ Retrieving and updating data◆ Maintaining referential integrity◆ Using the DBAID utility◆ Creating RDM reports◆ Using RDM user exits

- P26-8221** ***SUPRA Server PDM RDM PDM Support Supplement (OS/390 & VSE)***
- Audience** Database administrators
- Synopsis** Provides reference information needed to access SUPRA Server PDM files through SUPRA Server RDM base views. Provides examples. This manual is a supplement to the *SUPRA Server PDM RDM Administration Guide (OS/390 & VSE)*, P26-8220.
- P26-8222** ***SUPRA Server PDM RDM VSAM Support Supplement (OS/390 & VSE)***
- Audience** Database administrators
- Synopsis** Provides reference information needed to access native Key-Sequenced Data Sets (KSDS) VSAM files through SUPRA Server RDM base views. Provides examples. This manual is a supplement to the *SUPRA Server PDM RDM Administration Guide (OS/390 & VSE)*, P26-8220.
- P26-8330** ***SUPRA Server PDM RDM COBOL Programming Guide (OS/390 & VSE)***
- Audience** Applications programmers
- Synopsis** Explains how to issue requests to the SUPRA RDM from a COBOL program. Provides examples.
- P26-8331** ***SUPRA Server PDM RDM PL/I Programming Guide (OS/390 & VSE)***
- Audience** Applications programmers
- Synopsis** Explains how to issue requests to the SUPRA Server RDM from a PL/I program. Provides examples.
- P26-9062** ***SUPRA Server Digest***
- Audience** General
- Synopsis** Introduces and provides an overview of SUPRA Server.

P26-9220 ***SPECTRA Administration Guide***

Audience Database administrators or systems programmers

Synopsis Provides administration information on SPECTRA. SPECTRA is an information retrieval tool that provides interactive and batch query and report functions. This manual describes:

- ◆ SPECTRA installation
- ◆ The SPECTRA options, options module, and options macro
- ◆ Control of user profiles
- ◆ Use of external files
- ◆ Use of the personal file system
- ◆ Recovery of data

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

Audience Advanced end user

Synopsis Introduces SPECTRA and explains how to use it. SPECTRA is an information retrieval tool that provides interactive and batch query and report functions. This manual describes:

- ◆ Displaying files
- ◆ Creating and deleting personal files
- ◆ Modifying data in a file
- ◆ Creating and maintaining file layouts
- ◆ Maintaining processes
- ◆ Designing reports
- ◆ Using external files
- ◆ Importing and exporting personal files and processes
- ◆ Using command syntax

Index

A

- active schema, maintaining 36
- attached central operating mode, PDM 41
- attached operating mode, PDM 40

B

- batch Directory Maintenance
 - batch output 81
 - batch processing 80
 - command statements 79
 - comment statements 78
 - input statements 77
 - online 73
 - Run Option Definition statements 78
- buffer Cache Facility 43
- buffering PDM BDAM files 43

C

- category menu, Online Directory Maintenance 74
- central operating mode, PDM 39
- chained data 32
- command menu, Online Directory Maintenance 75
- COMMIT statement 34
- creating
 - program with DBAID 60

D

- data
 - dynamic indexing 33
 - normalization 28
 - storage 42
 - types, PDM 32
- DBAID
 - commands 61
 - using to create and test programs 60
- derived view 56

- directory
 - entities, creating in a hierarchy 69
 - general 19
- directory security, maintaining 72
- domains
 - RDM 53
- dynamic indexing 33

E

- entities
 - creating in a hierarchy 69
 - defining using Directory Maintenance 64
- entity relationships, establishing 70
- exit point 29
- experience table 42

F

- file
 - buffering 43
 - experience table 42
 - types, under RDM 30
 - utilities, PDM 44

H

- hashed data 32

I

- indexed data 32
- initializing the PDM
 - IBM environment 37
- interactive Services 48

L

- logging
 - PDM 33

M

- maintaining
 - active schema 36
 - directory security 72
 - PDM 48
 - RDM/PDM current programs 58

- metadata
 - using Directory Maintenance to define 63
- modes, operating, PDM 38

O

- online directory maintenance 73
- operating modes 38
- optimizing performance, RDM 57

P

- PDM
 - files 30
 - general 19, 31
 - initializing
 - IBM environments 37
 - logging 33
 - maintaining 48
- PDM Kernel
 - general 49
 - support, file types 30
- PDM Server
 - support, security 29
- programs
 - insulating from change under RDM 59
 - maintaining under RDM 58

R

- RDM/PDM
 - accessing data with views 57
 - creating and testing programs with DBAID 60
 - designing and building views 50
 - domains 53
 - general 19
 - insulating programs from change 59
 - maintaining current programs 58
 - optimizing performance 57
 - relational data structure 27, 28
 - reports and statistics 62
 - support
 - general 25
 - three-schema architecture 17
- record blocking, PDM 42

- recovering and restoring after system failure
 - IBM 47
 - PDM 33
- recovering and restoring files
 - IBM environments 47
 - PDM 33
- relational data structure 28
- relationships, defining using Directory Maintenance 64
- reports
 - RDM 62
- restarting, after failure, IBM environments 46
- retrieving data
 - using dynamic indexing 33

S

- security, RDM 29
- single-task operating mode, PDM 38
- SPECTRA
 - accessing external files 95
 - calculating statistics 89
 - commands 87
 - customizing reports 88
 - facilities 88
 - general 83
 - Master Menu options 85
 - processes 92
 - reports 98
 - splitting screens 86
 - user profiles 96
- sticky field 80
- storing
 - data 42

T

- Task failure 46
- Testing, programs with DBAID 60
- three-schema architecture 17

V

- View
 - accessing RDM data 57
 - designing and building, RDM 50