AA-KY66C-TE

VAX Rdb/VMS

Introduction and Master Index

December 1990

This manual provides an overview of VAX Rdb/VMS, a general description of the VAX Rdb/VMS documentation set, a glossary of VAX Rdb/VMS terms, and a master index to the VAX Rdb/VMS manuals.

Revision/Update Information:	This manual is a revision and supersedes previous versions.
Operating System:	VMS
Software Version:	VAX Rdb/VMS Version 4.0

digital equipment corporation
maynard, massachusetts

This document was prepared using VAX DOCUMENT, Version 1.2.

Contents

Tables

Preface

The VAX Rdb/VMS software, often referred to as Rdb/VMS in this manual, is a general purpose database management system based on the relational data model.

Purpose of This Manual

This manual serves four functions. It provides:

1 An overview of Rdb/VMS terms, concepts, and components

2 A brief summary of the purpose and contents of the manuals in the Rdb/VMS documentation set

3 A glossary of Rdb/VMS terms

4 A master index for the Rdb/VMS manuals

Intended Audience

This book is intended for the data processing professional who wants to become acquainted with the components of VAX Rdb/VMS, the relational database system from Digital for VMS systems. You do not need to be an expert with relational databases before reading this book. However, you should be familiar with the VMS operating system and VAX RMS, a record management service, before reading this manual. If you are not familiar with the VMS operating system, refer to the VMS documentation set for more information.

Operating System Information

Information about the versions of the operating system and related software that are compatible with this version of Rdb/VMS is included in the Rdb/VMS media kit and the *VAX Rdb/VMS Installation Guide*.

For information on the compatibility of other software products with this version of Rdb/VMS, refer to the System Support Addendum (SSA) that comes with the Software Product Description (SPD). You can use the SPD/SSA to verify which versions of your operating system are compatible with this version of Rdb/VMS.

Structure

This manual consists of two chapters, a glossary, and a master index.

Chapter 1	Introduces Rdb/VMS terms, concepts, and components.
Chapter 2	Summarizes the purpose and contents of each Rdb/VMS manual.
Glossary	Defines terms used with Rdb/VMS.
Appendix A	Contains a master index for the Rdb/VMS documentation set.

Conventions

In examples, an implied carriage return occurs at the end of each line, unless otherwise noted. You must press the RETURN key at the end of a line of input.

This section explains the conventions used in this manual:

. . .	A vertical ellipsis in an example means that information not directly related to the example has been omitted.
. . .	A horizontal ellipsis in statements or commands means that parts of the statement or command not directly related to the example have been omitted.
e, f, t	Index entries in the printed manual may have a lowercase e, f, or t following the page number; the e, f, or t is a reference to the example, figure, or table, respectively, on that page.
Color	In printed manuals, color in examples shows user input.
$	The dollar sign represents the DIGITAL Command Language prompt. This symbol indicates that the DCL interpreter is ready for input.

References to Products

The Rdb/VMS documentation to which this document belongs often refers to products by their abbreviated names:

- DEC RdbExpert software is referred to as RdbExpert.
- DECdecision software is referred to as DECdecision.
- DECtrace software for VMS is referred to as DECtrace.
- The SQL interface to VAX Rdb/VMS is referred to as SQL. The SQL interface is Digital Equipment Corporation's implementation of the SQL standard ANSI X3.135-1989, ISO 9075:1989, commonly referred to as ANSI/ISO.
- VAX ACMS software is referred to as ACMS.
- VAX BASIC software is referred to as BASIC.
- VAX C software is referred to as C.
- VAX CDD/Plus software is referred to as CDD/Plus, the data dictionary, and the dictionary.
- VAX COBOL software is referred to as COBOL.
- VAX Data Distributor software is referred to as Data Distributor.
- VAX DATATRIEVE software is referred to as DATATRIEVE.
- VAX FORTRAN software is referred to as FORTRAN.
- VAX Pascal software is referred to as Pascal.
- VAX PL/I software is referred to as PL/I.
- VAX RALLY software is referred to as RALLY.
- VAX Rdb/ELN software is referred to as Rdb/ELN.
- VAX Rdb/VMS software is referred to as Rdb/VMS. Version 4.0 of VAX Rdb/VMS software is often referred to as V4.0.
- VAX TDMS software is referred to as TDMS.
- VAX TEAMDATA software is referred to as TEAMDATA.

1 Rdb/VMS Overview

Rdb/VMS is the Digital relational database management product that supports a full set of utilities and data definition and manipulation languages that let you create, query, and maintain your Rdb/VMS databases.

You can use Rdb/VMS for interactive queries or from within application programs. One advantage of a relational database is the ease with which you can retrieve precisely the information you want, even if you must gather the information from data stored in different places. You can access an Rdb/VMS database using any of the interfaces provided in your Rdb/VMS kits. Additionally, there are a number of other Digital products (discussed later in this chapter) through which you can access an Rdb/VMS database.

One of the interfaces supplied with your Rdb/VMS kit is SQL (structured query language), an industry-standard data definition and data manipulation language for relational databases. The SQL interface is Digital Equipment Corporation's implementation of the SQL standard ANSI X3.135-1989, ISO 9075:1989, commonly referred to as ANSI/ISO. Digital recommends that you use SQL to access your Rdb/VMS database.

Using SQL, you can design and create an Rdb/VMS database, load it with data, and read and update both data and data definitions. You can use SQL interactively or in application programs. SQL provides both a module language processor that lets you use SQL with any language that supports the VAX Procedure Calling Standard, whether or not there is a precompiler for that language, and a precompiler that supports VAX Ada, VAX C, VAX COBOL, VAX FORTRAN, VAX Pascal, and VAX PL/I.

When using SQL module language, you put SQL statements in a separate module language file which is called by your application program. SQL and the other Rdb/VMS interfaces are discussed in Section 1.4 and Section 1.5.

For additional introductory information, you can run an online demonstration procedure that introduces Rdb/VMS and shows how to create, use, and maintain a simple database. Run the online demonstration by entering:

```
$ @RDM$DEMO:DEMO
```

This chapter discusses the following topics:

- Relational concepts, Rdb/VMS features, and terminology
- Accessing online help
- Setting up the sample databases
- Accessing Rdb/VMS using SQL
- Accessing Rdb/VMS using other interfaces
- The online program examples for SQL and other interfaces
- Maintaining and tuning Rdb/VMS databases with the Rdb/VMS Management Utility (RMU)
- Rdb/VMS product kits
- Using Rdb/VMS with other Digital products

1.1 Understanding Rdb/VMS Concepts, Features, and Terminology

Before you create an Rdb/VMS database, you need to understand the relational model of database management systems, the Rdb/VMS features available to help you create, access, and maintain your database, and the differences in terminology among the different interfaces. This section discusses these topics.

1.1.1 Understanding Relational Concepts as Implemented in Rdb/VMS

In a relational database, data is organized in tables (or relations). Each **table** is made up of columns and rows. The **columns** are equivalent to data elements, or fields. The **rows** are equivalent to records; they indicate occurrences of column values in the table. If you are familiar with manipulating data in separate data files, you may think of tables as being roughly equivalent to data files. Columns and rows in a table are analogous to fields and records in a data file.

For example, Table 1–1 shows the five columns of the COLLEGES table of the Rdb/VMS sample personnel database, along with the first four rows (record occurrences). AU begins the American University row; BATE begins the Bates College row, and so forth.

Table 1–1 Sample COLLEGES Table

COLLEGE_CODE	COLLEGE_NAME	CITY	STATE	POSTAL_CODE
AU	American University	Washington	DC	20016
BATE	Bates College	Lewiston	ME	04240
BOWD	Bowdoin College	Brunswick	ME	04011
CALT	Cal. Institute of Tech.	Pasadena	CA	91125

Not all tables contain data that is physically stored in the database. A **view** is a logical table whose data is not physically stored in the database. You would access a view just as you would access a table whose data is physically stored. Views provide a way to define database access to fit the needs and access rights of particular users. You can include in your view definition any combination of columns and rows from other tables to tailor the database to suit your needs.

Table 1–2 presents the CURRENT_SALARY view from the sample personnel database. Columns from different tables comprise the columns of the CURRENT_SALARY view. For example, the columns LAST_NAME, FIRST_NAME, and EMPLOYEE_ID are from the EMPLOYEES table; and the SALARY_START and SALARY_AMOUNT columns are from the SALARY_HISTORY table.

Table 1–2 CURRENT_SALARY View

LAST_NAME	FIRST_NAME	EMPLOYEE_ID	SALARY_START	SALARY_AMOUNT
Toliver	Alvin	00164	14-JAN-1983	$51,712.00
Smith	Terry	00165	1-JUL-1982	$11,676.00

Tables and views are defined in the schema for the database. A **schema** contains the data definitions for a database. The data definitions include the tables, views, constraints, indexes, domains, triggers, and, for multifile databases, storage areas and (optionally) storage maps. (In a single-file database, the data definitions and the actual data are in the same file.) The SQL CREATE SCHEMA statement actually creates the database that is based on the specified schema definitions. You can easily modify schema definitions in Rdb/VMS as required by your application needs.

Constraints are conditions that restrict what values can be stored in a table. When you insert and update columns, the constraint checks the values you attempt to insert against the conditions specified by the constraint. For example, a constraint could ensure that the value entered for the SEX column would be either M or F. If a value violates the conditions set by the constraint, SQL generates an error message. You specify constraints in SQL CREATE TABLE and SQL ALTER TABLE statements.

Table-specific constraints are a type of constraint that prevent changes from being made that would violate the referential integrity of the database. For instance, a table-specific constraint could prevent the deletion of a row from the EMPLOYEES table if rows still exist in the JOB_HISTORY table for that employee.

Triggers are used to ensure that if you make a change to a table, appropriate changes will be made in other tables throughout the database. For example, you can define a trigger to specify that before a row is deleted from the EMPLOYEES table, all records for that employee would be deleted from other tables as appropriate. (You could store employee information about to be deleted in an archival database before the triggered action takes place.)

A **domain** can be thought of as a template for one or more table column definitions. The SQL CREATE DOMAIN statement associates a data type, and optionally other characteristics, with a domain name. Subsequent SQL CREATE TABLE statements use the domain definitions in their column definitions. User-defined domains let you standardize information with common characteristics so that the same elements are not defined differently in different tables.

For example, the domain ID_DOM is used for the EMPLOYEE_ID column in several tables and for the SUPERVISOR_ID column in the JOB_HISTORY table; the domain LAST_NAME_DOM is used for the LAST_NAME column in the EMPLOYEES and CANDIDATES tables, and the DATE_DOM domain is used for date columns in the EMPLOYEES and JOB_HISTORY tables. The following example shows some domain definitions and indicates which table uses those domains:

```
SQL> CREATE DOMAIN ID_DOM CHAR(5);
SQL> !
SQL> CREATE DOMAIN LAST_NAME_DOM CHAR(14);
SQL> !
SQL> CREATE DOMAIN FIRST_NAME_DOM CHAR(10);
SQL> !
SQL> CREATE DOMAIN DATE_DOM DATE;
SQL> !
SQL> SHOW TABLE EMPLOYEES;
```

```
Column Name                      Data Type        Domain
-----------                      ---------        ------
EMPLOYEE_ID                      CHAR(5)          ID_DOM
LAST_NAME                        CHAR(14)         LAST_NAME_DOM
FIRST_NAME                       CHAR(10)         FIRST_NAME_DOM
    .
    .
    .
BIRTHDAY                         DATE             DATE_DOM

SQL> SHOW TABLE JOB_HISTORY;
Column Name                      Data Type        Domain
___________                      _________        ______
EMPLOYEE_ID                      CHAR(5)          ID_DOM
    .
    .
    .
JOB_START                        DATE             DATE_DOM
JOB_END                          DATE             DATE_DOM
    .
    .
    .
SUPERVISOR_ID                    CHAR(5)          ID_DOM

SQL> SHOW TABLE CANDIDATES;
Column Name                      Data Type        Domain
_____________________            _________        ______
LAST_NAME                        CHAR(14)         LAST_NAME_DOM
FIRST_NAME                       CHAR(10)         FIRST_NAME_DOM
    .
    .
    .
```

Lists (SQL) or **segmented strings** (RDO) are special data types for the storage and retrieval of unstructured data. Examples of unstructured data include the following formats:

- Large amounts of text
- Long strings of binary input from a data collecting device
- Graphics data

In Rdb/VMS, this data is stored in unstructured bytes and divided into segments no larger than 64K bytes. Except for the length of these segments, Rdb/VMS does not know anything about the type of data contained in these data segments.

A **transaction** is a set of operations on a database that must complete as a unit or not complete at all. A transaction lets you defer the action of making permanent (committing) database changes until you are sure that all statements in a unit have executed successfully and have done what you expected them to do. Because a transaction groups SQL statements into a unit, you can apply (commit) or undo (roll back) the changes made by the statements within the transaction. For example, you can defer committing the changes (making the changes permanent) until you are sure that all update operations have been made correctly.

A **distributed transaction** groups more than one database or more than one database attachment into one transaction, even if the databases are located on different nodes. You apply changes to more than one database within one transaction; Rdb/VMS ensures that transactions involving changes to multiple databases are handled as a unit by using a two-phase commit protocol.

1.1.2 Understanding Rdb/VMS Features

Once you understand the basic structure of an Rdb/VMS database, you need to be aware of the varied features that let you create, use, and maintain the database. Rdb/VMS is a full-feature relational database system that offers:

- Data independence and consistency

  You can remove data definitions from application programs and store them in the database with the data. Because Rdb/VMS reduces the storing of redundant data, it helps ensure that updates do not result in inconsistent data. Rdb/VMS also lets you centralize the management of data definitions, both within the database file and within the data dictionary (VAX CDD/Plus). You can use views to manage data stored in separate tables in the database. Table-specific constraints and triggers can be used to ensure that changed data remains consistent throughout the database.

- Data integrity

  Rdb/VMS maintains the integrity of the database in the event of user errors, hardware or software failures, and concurrent use of the database. A before-image journal (also called recovery-unit journal (RUJ)) contains copies of records before they were updated. Rdb/VMS uses the before-image journal to automatically undo updates to a database when a rollback or system failure occurs. An after-image journal (AIJ) contains copies of rows after they have been updated. You can use the after-image journal to reconstruct a backup database to include the last successfully completed transaction. Integrity constraints ensure that data remains correct even when users try to modify it incorrectly.

- Interactive, multiuser environment

  You can have access to the data at the same time as other users, yet each user can work from a customized view that may include only a subset of the entire database. Rdb/VMS also ensures that your operations on the database do not lead to inaccurate results for other users. Locking mechanisms ensure the maintenance of your data's integrity and consistency. Rdb/VMS uses snapshot files to ensure that you read a consistent view of the data even while other users are updating the database.

- Data security

  Rdb/VMS complies with the C2 class security requirements specified by the National Computer Security Center (NCSC) in the Department of Defense (DoD) Trusted Computer System Evaluation Criteria, (the *Orange Book*). C2 class security requirements include:

  - Discretionary protection ("need-to-know protection")

    The creator of a database object can control the protection on that object.

  - User identification and authentication

    Rdb/VMS allows use of the database by authenticated VMS users.

  - Resource isolation

    This feature protects database objects from interference by non-database processes.

  - Accountability

    Rdb/VMS audits user actions that affect the security of the database.

  Rdb/VMS implements C2 class security through a number of database security mechanisms:

  - Access privilege sets

    You can use an access privilege set to limit database object access to particular users and to specify a list of privileges for those users.

  - Role-oriented privileges

    You can designate users with role-oriented privileges to perform necessary tasks such as a backup operation without compromising the security of the data.

  - Security audit logging

    You can audit events that affect the security of the database, and review the audit records generated by those events, by using facilities provided by RMU.

- Optimized data access

  Rdb/VMS includes a query optimizer that automatically analyzes each query to determine the most efficient method of access to the data. Because of the query optimizer, you do not have to be overly concerned with how you construct your queries.

- Ability to create multifile databases

  You can define multiple files for one database, and you can distribute the individual files across many disks on your VAX system or VAXcluster configuration. You can create either a single-file or multifile Rdb/VMS database. In a single-file database, all user-stored data and internal Rdb/VMS data resides in a single database root file with the default file type RDB. A single-file database also includes a snapshot file, which provides temporary storage for read-only data retrieval and has a default file type SNP. By default, both files are located on the same device and in the same directory. However, these locations can be changed.

  A multifile database includes one or more user-defined storage-area files in which user-stored data from one or more tables or indexes will reside. A multifile database consists of a database root file (file type RDB), one or more storage area files (file type RDA), and one or more snapshot files (file type SNP). One snapshot file is created for each storage area file. These files can be placed on different devices or in different directories.

- Choice of interfaces to the database

  You can access an Rdb/VMS database through various supplied interfaces. These interfaces include:

  - SQL (structured query language)
  - SQL module language
  - SQL precompiler
  - RDO (Relational Database Operator)
  - RDML (Relational Data Manipulation Language)
  - RDBPRE preprocessor

  SQL is used in most examples in the Rdb/VMS documentation set. However, the Rdb/VMS documentation set includes reference manuals for RDO and RDML, as well as a guide to using RDO, RDML, and RDBPRE.

Note *You can use either the SQL or the RDO interface to work with your databases, regardless of which interface was used to create them.*

- Access from systems other than VMS

  You can access an Rdb/VMS database from various platforms, such as VMS, MS-DOS, and ULTRIX, with SQL/Services, a client/server component of Rdb/VMS that provides a callable routine interface similar to dynamic SQL. (**Dynamic SQL** is a set of SQL statements and data structures that allows programs to formulate and execute SQL statements at run time.) Application programs call routines in the SQL/Services Application Programming Interface (API) library. The API library communicates through DECnet with a server process on the VMS system on which the Rdb/VMS database resides.

- Operations involving multiple databases

  You can start a distributed transaction that combines data from more than one database in a single query or update operation. These databases need not be on the same node. Your application can attach to more than one database using authorization identifiers to uniquely identify each database.

  If the distributed transaction makes changes to more than one database, Rdb/VMS uses a two-phase commit protocol to ensure that all changes are either committed or rolled back.

- Ability to use read-only storage areas on a compact disk

  You can store an Rdb/VMS read-only storage area on a compact disk. Compact disk read-only memory (CDROM) software can be used to permit fast and efficient online access to the data. Use of a compact disk enhances online retrieval and conserves storage space.

- Use of indexes for quick data retrieval

  You can locate particular records based on key column values by using an index. Sorted and hashed indexes both allow for quick and efficient data retrieval based on index keys. Sorted indexes can be used to retrieve a range of values for a column; hashed indexes are best used to retrieve only a specific value. Hashed indexes can be used only with multifile databases. Sorted indexes can be used with single-file and multifile databases.

- Language-Sensitive Editor (LSE) templates for SQL statements

  You can use the VAX Language-Sensitive Editor (LSE) templates to code SQL modules for incorporation in application programs. LSE expands the SQL statements to provide syntax for you, letting you simply fill in the blanks in the supplied language template. Templates are not available for RDO, RDBPRE, or RDML.

- Distributed databases

  You can access an Rdb/VMS database on a remote node by supplying the node name with the file specification for the database. You can also use VAX Data Distributor to make complete or partial copies of your Rdb/VMS databases on remote nodes. A replication database would be periodically updated, an extraction database would not.

- Centralized database management

  A single user can handle the responsibility of administering the database. This centralization of database administration tasks helps ensure the consistency of the database. A set of simple RMU commands and SQL or RDO statements make database maintenance and control easy.

- Control over disk space management

  You can place the database root file, data files, snapshot files, journal files, and backup files on different disks to ensure both the availability of space and the preservation of the database in the event of a disk failure.

- Maintenance and performance tools

  Rdb/VMS supplies a variety of maintenance and performance tools for you to use. The Rdb/VMS Management Utility (RMU) lets you perform the following functions:

  - Back up and restore the database either on a database-wide level or by area
  - Open and close the database
  - Unload specific tables and views into files
  - Reload data from these files or from record management services (RMS) files into a specific table
  - Recover the database from the after-image journal file
  - Analyze the efficiency of retrieval through indexes
  - Display database statistics
  - Verify the integrity of the database

- VAXcluster support, including full journaling and recovery

  Rdb/VMS is fully operational in a VAXcluster environment.

- Full online help facilities

  Online help is available through the DIGITAL Command Language (DCL) level as well as through the interactive Rdb/VMS utilities. See Section 1.2 for more information about accessing online help for Rdb/VMS.

1.1.3 Understanding Differences in Relational Terminology

As Section 1.1.2 presents, there are several interfaces to choose from when using Rdb/VMS. In general, terminology for relational database systems varies from system to system and from interface to interface. Thus, the Rdb/VMS interfaces use different terms to mean the same thing. For instance, some terms used by SQL differ from terms used by other interfaces, such as RDO or RDML.

Table 1–3 shows some of the terms used to describe basic relational database concepts. For example, when you use documentation pertaining to SQL, you will encounter the terms table, column, and row. However, when you use documentation pertaining to RDO or RDML, you will encounter the terms relation, field, and record. This manual primarily uses SQL terminology.

Table 1–3 Differences in Relational Terminology

SQL	RDO, RDML	ANSI/ISO SQL STANDARD
Alias	Context variable	Alias
Authorization identifier	Database handle	
Cartesian product	Cross product	Cartesian product
Column	Field	Column
Column select expression	Record selection expression	Column select expression
Consistency level 2	Concurrency	Consistency level 2 (SQL2)[1]
Consistency level 3	Consistency	Consistency level 3 (SQL2)[1]
Context files	N/A[2]	N/A[2]
Domain	Global field	Domain (SQL2)[1]
List	Segmented String	
Parameter	Host language variable	Parameter
Predicate	Conditional expression	Predicate
READ ONLY	READ_ONLY	READ ONLY
READ WRITE	READ_WRITE	READ WRITE
Result table	Record stream	Result table
Row	Record	Row

[1]SQL2 is the industry standard currently being developed.

[2]N/A means that the term is not applicable or not used with the listed product, standard, or system.

(continued on next page)

Table 1–3 (Cont.) Differences in Relational Terminology

SQL	RDO, RDML	ANSI/ISO SQL STANDARD
Storage area	Storage area	N/A[2]
Storage map	Storage map	N/A[2]
Table	Relation	Table

[2]N/A means that the term is not applicable or not used with the listed product, standard, or system.

Just as different terms mean the same thing between RDO and SQL, different statements do the same thing. Table 1–4 presents the SQL data definition and manipulation statements and supplies the RDO equivalents.

Table 1–4 Syntax Differences Between SQL and RDO

SQL Statement	RDO Statement
ALTER DOMAIN	CHANGE FIELD
ALTER INDEX	CHANGE INDEX
ALTER SCHEMA	CHANGE DATABASE
ALTER STORAGE MAP	CHANGE STORAGE MAP
ALTER TABLE	CHANGE RELATION
BEGIN DECLARE	No equivalent
CLOSE	END_STREAM
CLOSE LIST CURSOR	END_SEGMENTED_STRING
COMMENT ON	DESCRIPTION IS
COMMIT	COMMIT
CREATE COLLATING SEQUENCE	DEFINE COLLATING_SEQUENCE
CREATE DOMAIN	DEFINE FIELD
CREATE INDEX	DEFINE INDEX
CREATE SCHEMA	DEFINE DATABASE
CREATE STORAGE AREA clause	DEFINE STORAGE_AREA clause
CREATE STORAGE MAP	DEFINE STORAGE MAP
CREATE TABLE	DEFINE RELATION
CREATE TRIGGER	DEFINE TRIGGER

(continued on next page)

Table 1–4 (Cont.) Syntax Differences Between SQL and RDO

SQL Statement	RDO Statement
CREATE VIEW	DEFINE VIEW
DCL Invoke ($)	DCL Invoke ($)
DECLARE CURSOR	START_STREAM and DECLARE_ STREAM
DECLARE SCHEMA	INVOKE DATABASE
DECLARE STATEMENT	No equivalent
DECLARE TABLE	No equivalent
DECLARE TRANSACTION	START_TRANSACTION
DELETE	ERASE
DESCRIBE	No equivalent
DROP COLLATING SEQUENCE	DELETE COLLATING_SEQUENCE
DROP CONSTRAINT	DELETE CONSTRAINT
DROP DOMAIN	DELETE FIELD
DROP INDEX	DELETE INDEX
DROP PATHNAME	DELETE PATHNAME
DROP SCHEMA	DELETE DATABASE
DROP STORAGE MAP	DELETE STORAGE MAP
DROP TABLE	DELETE RELATION
DROP TRIGGER	DELETE TRIGGER
DROP VIEW	DELETE VIEW
EDIT	EDIT
END DECLARE	No equivalent
Execute @	Execute @
EXECUTE	No equivalent
EXECUTE IMMEDIATE	No equivalent
EXIT	EXIT
EXPORT	EXPORT
FETCH	FETCH
FINISH	FINISH
GRANT	DEFINE PROTECTION
GRANT ANSI style	No equivalent

(continued on next page)

Table 1–4 (Cont.) Syntax Differences Between SQL and RDO

SQL Statement	RDO Statement
HELP	HELP
IMPORT	IMPORT
INCLUDE	No equivalent
INSERT	STORE
INTEGRATE	INTEGRATE DATABASE
OPEN CURSOR	START_STREAM
OPEN INSERT ONLY LIST CURSOR	CREATE_SEGMENTED_STRING
OPEN READ ONLY LIST CURSOR	START_SEGMENTED_STRING
PREPARE	No equivalent
PRINT	PRINT
QUIT	EXIT
RELEASE	No equivalent
REVOKE	DELETE PROTECTION
REVOKE ANSI style	No equivalent
ROLLBACK	ROLLBACK
SELECT	FOR-GET-END_FOR or FOR-PRINT-END_FOR
SET	SET
SET ALL CONSTRAINTS	No equivalent
SET TRANSACTION	START_TRANSACTION
SHOW	SHOW
UPDATE	MODIFY
WHENEVER	ON ERROR
No equivalent	ANALYZE
No equivalent	CLOSE
No equivalent	OPEN
No equivalent	PLACE
Use CREATE TABLE	DEFINE CONSTRAINT

1.2 Accessing Online Help for Rdb/VMS

Complete online help is available for Rdb/VMS. From the DCL level, you can access help for Rdb/VMS, SQL, RDO, RDML, and RMU. For example, to access a general help screen for Rdb/VMS, enter the following command at the DCL prompt:

```
$ HELP RDBVMS
```

(Note the absence of the slash between Rdb and VMS.)

Entering a question mark at the topic prompt will cause DCL to display a list of all topics included under RDBVMS help.

Additionally, more detailed help is available from within the interactive interfaces, SQL and RDO. To access this help, enter the interactive interface of your choice, and then enter help:

```
$ SQL
SQL> HELP
```

Each primary HELP command displays a list of topics on which information is available. At the topic prompt, enter one of the listed subjects to display information, and, in some cases, a list of subtopics.

You may prefer to access a specific topic or subtopic with one simple HELP command. For example, to access the syntax diagram for the SQL CREATE SCHEMA statement, you can enter:

```
SQL> HELP CREATE SCHEMA
```

You can exit help by either pressing the RETURN key until you reach the prompt from which you first entered the HELP command, or by pressing CTRL/Z at any point.

1.3 Setting Up the Sample Database

To help you learn and test Rdb/VMS features, Rdb/VMS provides files to create the various forms of the sample personnel database: PERSONNEL, a single-file database, and MF_PERSONNEL, a multifile database. These forms can be built using either SQL or RDO data definitions. Additionally, you can choose whether or not to store data definitions in the data dictionary.

This section guides you through the process of creating your own copy of the sample Rdb/VMS databases. Many examples in the documentation are based on the sample personnel database. You create copies of the personnel database by executing the DCL command procedure PERSONNEL.COM located in the directory RDM$DEMO. By default, this command procedure builds the single-file PERSONNEL database using SQL data definitions. A prompt appears regarding use of the data dictionary:

```
$ @RDM$DEMO:PERSONNEL

This command file builds the VAX Rdb/VMS sample database, PERSONNEL.
It requires approximately 2900 blocks of disk space.

You may enter the name of a VMS directory in which to create the database
or use your current default directory.  You can also have the database
definitions placed in the common data dictionary if VAX CDD/Plus is
installed on your system.  In this case you must specify a VAX CDD/Plus
directory in which to create the definitions.

NOTE: If a copy of PERSONNEL already exists in the location you
specify, this command procedure will stop.

Do you wish to use CDD/Plus? (Yes or No or Exit) NO

     Enter VMS directory specification for the database files

Directory; default is DISK1:[JONES.TEMP]:

 >> Creating database in specified directory


 Creating domains for the sample database

 Creating tables for the sample database

 Creating views for the sample database

 Adding comments for domain and table definitions

 Storing sample data in several tables. (Other tables to
 be loaded later by separate programs.)
1 row inserted
1 row inserted
1 row inserted
1 row inserted
1 row inserted
1 row inserted

 >> Loading PERSONNEL database from data files

Program: Loading EMPLOYEES
Program: EMPLOYEES Loaded. Normal End-of-Job
Program: Loading JOBS
Program: JOBS Loaded. Normal End-of-Job
Program: Loading DEPARTMENTS
Program: DEPARTMENTS Loaded. Normal End-of-Job
Program: Loading JOB_HISTORY
Program: JOB_HISTORY Loaded. Normal End-of-Job
Program: Loading SALARY_HISTORY
Program: SALARY_HISTORY Loaded. Normal End-of-Job
Program: Loading COLLEGES
Program: COLLEGES Loaded. Normal End-of-Job
Program: Loading DEGREES
Program: DEGREES Loaded. Normal End-of-Job
```

```
 Loading RESUMES table
1 row inserted
1 row inserted
1 row inserted

 Creating remaining sorted indexes for MF_PERSONNEL

 Creating triggers for the sample database

 >> End of command procedure.

 >> Examine the log file DISK1:[JONES.TEMP]PERSONNEL.LOG for the
 >> database definitions and possible errors which occurred during the
 >> database creation.
 >>
 >> The RDM$DEMO directory contains all the files used in building this
 >> sample database.
```

If the command procedure executes successfully, you will find the following files in the directory you specified:

- PERSONNEL.RDB

  This is the file where data and data definitions (metadata) are stored. **Metadata** is data that is used to describe other data. Data definitions are sometimes referred to as metadata.

- PERSONNEL.SNP

  This is the snapshot file that provides read-only access to the database. A snapshot file exists to improve database performance when there are many concurrent database users.

You can create alternative versions of the sample personnel database by using the three available parameters on the PERSONNEL command line. These parameters specify the following choices:

1 Use of either SQL or RDO as the data definition language. (Enter SQL or RDO.)

2 Creation of either a single-file or multifile personnel database. (Enter S or M.)

3 Whether or not to store data definitions in the data dictionary. (Enter CDD or NOCDD.)

To specify the second or third parameters, you must include any previous parameters.

The multifile version of the personnel database is MF_PERSONNEL. To build the MF_PERSONNEL database using SQL data definitions, enter the following command:

```
$ @RDM$DEMO:PERSONNEL SQL M NOCDD
```

This example will create the MF_PERSONNEL database using SQL. Data definitions will not be stored in the data dictionary.

The MF_PERSONNEL database consists of the following files:

- MF_PERSONNEL.RDB

  This is the database root file where data definitions (metadata) are stored.
- Several storage area files (file type RDA)

  Storage area files are where data is stored in a multifile database.
- Several snapshot files (file type SNP); one for each storage area file

  Rdb/VMS uses snapshot files during read-only transactions to ensure each transaction a consistent view of the database, and to improve overall performance when there are many concurrent users of the database.

If you encountered an error while running the command procedure, check the file PERSONNEL.LOG. You may have entered a dictionary or VMS directory specification incorrectly. In this case, you may have produced an incomplete version of the PERSONNEL or MF_PERSONNEL database. To remedy the situation, delete any data dictionary node or database files the command procedure has created. (The command procedure first creates the data dictionary node, then creates the database files, and finally stores data in the database. Unless an error, such as a system failure, occurs when data is being stored, you will have to delete either a dictionary node or database files, but not both.)

You can delete a database by invoking interactive SQL and entering a DROP SCHEMA FILENAME statement that includes a file specification. If you need to delete the database files and a dictionary node, use the PATHNAME clause of the DROP SCHEMA statement. For the VMS file specification and dictionary path name specified in the following examples, you would substitute the VMS file specification and dictionary path name that identify your incomplete copy of the PERSONNEL database:

```
SQL> DROP SCHEMA PATHNAME SYS$COMMON:[CDDPLUS]D32T.FIELDMAN.PERSONNEL;
```

```
SQL> DROP SCHEMA FILENAME DISK02:[FIELDMAN]PERSONNEL;
```

You can then run the PERSONNEL.COM command procedure again with a corrected path name or directory specification.

The PERSONNEL.COM command procedure consists of several smaller command files. These command files are located in the directory RDM$DEMO. You might want to use these files as models in creating your databases. Table 1–5 lists the command files used to create the sample databases with the SQL interface. For information on the command files used with the RDO interface, see the *VAX Rdb/VMS Guide to Using RDO, RDBPRE, and RDML.*

Table 1–5 SQL Files to Create Sample Databases

File Name	Explanation
PERSONNEL.COM	Builds a single-file or multifile version of the sample personnel database; you can use SQL or RDO definitions and you can define the database by file name or data dictionary path name. For the SQL single-file version, the procedure invokes SQL command files to create domains, tables, views, constraints, indexes, and triggers. In addition, it invokes programs to store most of the data. For the SQL multifile version, the procedure invokes SQL command files to create domains, tables, views, constraints, indexes (both sorted and hashed), and to spread tables and indexes across multiple files. In addition, it invokes programs to store most of the data.
BUILDPERS.SQL	Creates domains, tables, views, and constraints, and stores data.
SQL_CREATE_STORAGE.SQL	Creates storage areas for the multifile database.
MF_BUILDPERS.SQL	Creates domains, tables, views, constraints, hashed indexes, storage maps, and one sorted index, and stores data.
INDEXES.SQL	Creates the indexes (sorted) for the single-file database.
MF_INDEXES.SQL	Creates the remaining sorted indexes for the multifile database.
TRIGGERS.SQL	Creates triggers.
LIST.SQL	Loads rows of the RESUMES table.

The definitions of domains, tables, views, constraints, and triggers are the same for the single-file and multifile versions of the sample database. The definitions for the multifile database also include hashed indexes, storage area files, and other structures associated with the multifile implementation.

Note *You can use either the SQL or the RDO interface to work with the resulting databases, regardless of whether the SQL or RDO command files were used to create them.*

There are, however, some differences between the databases of the same name depending on whether you used the SQL-based procedure or the RDO-based procedure to create the database. The significant differences are as follows:

- The SQL-based procedures define default values for certain domains, whereas the RDO procedures define missing values for the corresponding global fields. The *VAX Rdb/VMS Guide to Database Design and Definition* discusses the distinction between a default value and a missing value.
- The SQL-based procedures define constraints to enforce the referential integrity of the database as part of the CREATE TABLE statements. In the RDO-based procedures, such constraints are defined in separate DEFINE CONSTRAINT statements.

1.4 Accessing Rdb/VMS Using SQL

Once you have created the PERSONNEL or MF_PERSONNEL databases, you can practice accessing them using SQL, the primary interface to Rdb/VMS. Digital recommends use of SQL instead of the proprietary RDO, RDML, or RDBPRE interfaces. Using SQL, you can access Rdb/VMS databases in several ways:

- Interactively
- Through programs
  - SQL module language
  - Precompiled SQL
  - Dynamic SQL
- From systems other than VMS (using SQL/Services)

This section presents the various SQL interfaces and discusses the various sample online programs available with Rdb/VMS.

1.4.1 Using Interactive SQL

This section introduces interactive SQL and demonstrates some simple retrieval statements you can use. As you develop application programs, you can invoke interactive SQL to learn the SQL language and to test statements that you plan to include in a program. For database administration, you may prefer the convenience of interactive SQL for creating or changing database definitions and protection. You can also use interactive SQL for queries against the database. While using interactive SQL, you can access online help at any time for information about SQL syntax. See Section 1.2 for more information about accessing online help.

For more detailed information on interactive SQL, see the *VAX Rdb/VMS Guide to Using SQL.*

Note *Interactive SQL can be used by people who are not data processing professionals. However, you might prefer to prepare applications at your site that are specially tailored to meet user needs. Generally, products such as DECdecision, TEAMDATA, or applications developed using RALLY or traditional programming languages, are the recommended relational database interfaces for most end users.*

1.4.1.1 Accessing Interactive SQL To use interactive SQL, you must run the SQL executable image. Define a symbol at the DCL level and include the symbol in your LOGIN.COM file. Define the symbol SQL as a foreign command. This command lets you execute an image by entering only the symbol name. (The command is **foreign** because it is unknown to DCL.) In the following example, the symbol SQL is defined to execute the image SQL$:

```
$ SQL :== $SQL$
```

After you enter the symbol SQL, the SQL prompt (SQL>) indicates that you are at the SQL interface command level and can interactively enter SQL statements.

```
$ SQL
SQL>
```

To exit interactive SQL, press CTRL/Z or type EXIT.

1.4.1.2 Retrieving Data Using SQL SELECT Statements Before you can access the database, you must establish a connection by attaching to it. Enter the following statement to attach to the sample PERSONNEL database (without using data dictionary definitions):

```
SQL> DECLARE SCHEMA FILENAME PERSONNEL;
```

If you are using the multifile MF_PERSONNEL database, specify MF_PERSONNEL rather than PERSONNEL on the command line.

To retrieve data from the database, you can use various combinations of SQL SELECT statements. (The SELECT statement that you use in programs varies slightly from the interactive SELECT statement. See the *VAX Rdb/VMS Guide to Using SQL* for details.) These SELECT statements create result tables. A **result table** is the set of columns and rows that fits the criteria for a specific retrieval statement. For example, the following is a result table produced by a simple SELECT expression that retrieves employee identification (ID) numbers and last names from the EMPLOYEES table:

```
SQL> SELECT EMPLOYEE_ID, LAST_NAME FROM EMPLOYEES;
 EMPLOYEE_ID   LAST_NAME
 00164         Toliver
 00165         Smith
 00166         Dietrich
 00167         Kilpatrick
    .
    .
    .
```

1.4.1.3 Using Indexes for Retrieving Data A particularly efficient way to retrieve data is through the use of an index. A search for data based upon an indexed value is fast and efficient. For example, in the EMPLOYEES table the index is sorted by the EMPLOYEE_ID column. Therefore, use of an EMPLOYEE_ID column value will speed the retrieval of data from the EMPLOYEES table because the query optimizer will choose to use the index to locate the data. The following example demonstrates an SQL SELECT statement that retrieves employee records using the EMPLOYEE_ID column value 00164:

```
SQL> SELECT EMPLOYEE_ID, LAST_NAME, FIRST_NAME, ADDRESS_DATA_1,
cont> CITY, STATE, POSTAL_CODE
cont> FROM EMPLOYEES
cont> WHERE (EMPLOYEE_ID = "00164");
 EMPLOYEE_ID   LAST_NAME       FIRST_NAME   ADDRESS_DATA_1
   CITY                  STATE   POSTAL_CODE
 00164         Toliver         Alvin        146 Parnell Place
   Chocorua              NH      03817

1 row selected
```

If necessary, you can also perform a search on a non-indexed record. Suppose you have an employee's last name, but not his or her employee ID number. You could perform the following search:

```
SQL> SELECT EMPLOYEE_ID, LAST_NAME
cont> FROM EMPLOYEES
cont> WHERE (LAST_NAME = "Toliver");
 EMPLOYEE_ID   LAST_NAME
 00164         Toliver
1 row selected
```

Rdb/VMS finds the information, but does not perform the search as efficiently as when an index has been defined on the column being searched.

1.4.1.4 Joining Data from Multiple Tables You can combine data from two or more different tables using retrieval statements. Such operations are called joins. **Joins** differ from views in that they operate only for a specific retrieval statement, as opposed to existing as a permanent logical table. The following SQL statement asks for the names and current departments of all employees, and illustrates the use of a join:

```
SQL> SELECT LAST_NAME,
cont>       FIRST_NAME,
cont>       DEPARTMENT_NAME
cont> FROM EMPLOYEES, DEPARTMENTS, JOB_HISTORY
cont> WHERE ((JOB_HISTORY.EMPLOYEE_ID = EMPLOYEES.EMPLOYEE_ID)
cont>        AND
cont>       (DEPARTMENTS.DEPARTMENT_CODE = JOB_HISTORY.DEPARTMENT_CODE)
cont>        AND (JOB_END IS NULL))
cont> ORDER BY LAST_NAME, FIRST_NAME, DEPARTMENT_NAME;
 EMPLOYEES.LAST_NAME   EMPLOYEES.FIRST_NAME   DEPARTMENTS.DEPARTMENT_NAME
 Ames                  Louie                  Northeastern US Sales
 Andriola              Leslie                 Telecommunications Industries
 Babbin                Joseph                 European Sales
 Bartlett              Dean                   Southern U.S. Sales
 Bartlett              Wes                    Corporate Sales
    .
    .
    .
```

In the preceding statement, the columns LAST_NAME, FIRST_NAME, and DEPARTMENT_NAME are retrieved from the EMPLOYEES and DEPARTMENTS tables, respectively. The EMPLOYEES and DEPARTMENTS tables do not share a column. Therefore, the join includes the JOB_HISTORY table to determine the correct department name for each employee. As the SELECT statement indicates, the employee identification number in the JOB_HISTORY table must equal the employee identification number in the EMPLOYEES table, and the DEPARTMENT_CODE column value in the DEPARTMENTS table must equal the DEPARTMENT_CODE column value in the JOB_HISTORY table.

Note that the preceding example qualified table names when the same column was used from more than one table. In SQL, qualifying or identifying the table is necessary only when using multiple tables. To save typing, you can define an alias to qualify the table from which a particular column is selected. (An **alias** is a qualifier for column names that is used in place of the table name in an SQL SELECT statement.) In the following example, the E after EMPLOYEES, the D after DEPARTMENTS, and the JH after JOB_HISTORY are all aliases:

```
SQL> SELECT LAST_NAME,
cont>        FIRST_NAME,
cont>        DEPARTMENT_NAME
cont> FROM EMPLOYEES E, DEPARTMENTS D, JOB_HISTORY JH
cont> WHERE ((JH.EMPLOYEE_ID = E.EMPLOYEE_ID)
cont>        AND
cont>        (D.DEPARTMENT_CODE = JH.DEPARTMENT_CODE)
cont>        AND (JOB_END IS NULL))
cont> ORDER BY LAST_NAME, FIRST_NAME, DEPARTMENT_NAME;
 E.LAST_NAME      E.FIRST_NAME    D.DEPARTMENT_NAME
 Ames             Louie           Northeastern US Sales
 Andriola         Leslie          Telecommunications Industries
 Babbin           Joseph          European Sales
 Bartlett         Dean            Southern U.S. Sales
 Bartlett         Wes             Corporate Sales
   .
   .
   .
```

1.4.1.5 Isolating Unique Data Values It is also possible to isolate unique values for a particular column. You can create a result table that consists of just one row for any given value of a single column (that is, additional rows with the same column value are not returned to you). For example, the EMPLOYEES table contains the column STATE. Suppose you want to list all the states in which employees live, listing each state only once regardless of the number of employees living there. To do so, you would use the DISTINCT clause with the SELECT statement:

```
SQL> SELECT DISTINCT STATE
cont> FROM EMPLOYEES;
 STATE
 CT
 MA
 NH
3 rows selected
```

1.4.2 Using SQL Statements in Programs

Executing SQL statements in programs, rather than from interactive SQL, should be the primary use of SQL for database access at your site. Program access provides a number of advantages, including processing speed. This section describes the ways in which you can use SQL in programs, and outlines the online sample programs available with Rdb/VMS. You can find more detailed information on using SQL statements in programs in the *VAX Rdb/VMS Guide to Using SQL.*

1.4.2.1 Using SQL Module Language You can use the SQL module language from *any* host language that supports the VMS Procedure Calling Standard. The SQL module language allows host language programs to execute SQL statements contained in an SQL module file. The module file includes one or more procedures, each of which contains parameter (variable) declarations and at least one SQL statement. The SQL module procedures can be called from any host language program, including those not supported by the SQL precompiler. A call to a procedure in an SQL module causes the SQL statement in the procedure to be executed.

The SQL module language offers these advantages to precompiled SQL:

- Languages not supported by the SQL precompiler can be used with the SQL module language instead.
- Isolation of all SQL statements in SQL modules improves modularity and avoids the use of two languages in the same source file.
- Programs can work around restrictions of the SQL precompiler by calling SQL modules.
- Programs written in languages for which there is an ANSI/ISO standard can avoid embedding nonconforming code by isolating SQL statements in SQL modules.

For any host language, you can use SQL module language to create a module of SQL procedures. In your host language source file you can then specify calls to these SQL procedures. Process your program by running:

1. The SQL module processor to create an SQL object file
2. The host language compiler to create a host language object file
3. The VMS Linker to create an executable image from the preceding object files
4. The image created by the linker

1.4.2.2 Using Precompiled SQL SQL statements can be embedded directly in certain host language source files. These source files can then be processed by the SQL precompiler. The precompiler converts embedded SQL statements to a form understandable by the host language compiler. The precompiler also invokes the host language compiler and creates an intermediate object file ready for linking.

If your host language is VAX Ada, VAX C, VAX COBOL, VAX FORTRAN, VAX Pascal, or VAX PL/I, you can embed SQL statements directly in a host language source file and process your program by running:

1 The SQL precompiler (which indirectly uses your host language compiler) to create an object module
2 The VMS Linker to create an executable image
3 The executable image created by the linker

1.4.2.3 Using Dynamic SQL **Dynamic SQL** is a set of SQL statements and data structures that allows programs to formulate and process SQL statements at run time. Supported by both the SQL module processor and the SQL precompiler, dynamic SQL offers programs some advantages normally available only in an interactive environment. For example, applications that use dynamic SQL can:

- Enter SQL statements from a terminal
- Accept SQL statements from a front-end program
- Translate statements in another language into SQL
- Act as a generic Rdb/VMS server for applications running on remote systems other than VMS systems

1.4.2.4 Using the SQL Online Program Examples Online SQL source program examples are available for your use in the directory defined by the logical SQL$SAMPLE. These program examples define and query the sample personnel database. You can create an executable form (image) of any source program example for which your system has language support and then run the image to see how the program works. You can also create hardcopy listings of program sources and use them as references when you create your own applications.

The online program examples are presented in several languages and include:

- SQL$ALL_DATATYPES

  Illustrates INSERT and UPDATE statements, variations in techniques of handling null values, and the correspondence between SQL host language data types and their conversions.

- SQL$DIST_TRANS

  Illustrates how to use distributed transactions. This program updates two Rdb/VMS databases, which contain information about the employees in a company's two divisions, and one VAX DBMS database, which contains central information about all employees in the company.

- SQL$DYNAMIC

  Illustrates dynamic SQL. This program consists of two files, SQL$DYNAMIC and SQL$DYNAMIC_DRIVER. The Ada module language version of this program provides a comprehensive example of a general dynamic SQL application. The PL/I version provides a more limited example.

- SQL$INSERT_DEGREES

  Illustrates SQL module language. This program adds information to the database about an employee's college degrees.

- SQL$LOAD_(table-name)

  Loads indicated table from a data file. Specific tables are used to illustrate various techniques. For instance, loading the EMPLOYEES table illustrates how to set a CHAR column to null, based on a column value in the data file; and loading the JOB_HISTORY table illustrates how to set a DATE column to null, based on a column value in the data file.

- SQL$MULTI_STMT_DYN

  Illustrates how to use extended dynamic cursors to process any number of dynamic SQL statements.

- SQL$REPORT

  Writes a salary report with control breaks on the department code and job code columns to print the average salary per job and total salary per department, and writes the grand total of department salary totals.

- SQL$RESUMES

  Illustrates how to insert segmented strings (lists) into the personnel database.

- SQL$TERMINATE

  Illustrates how to update the personnel database when an employee is terminated.

To find out what is available for precompiled SQL programs in a specific language, enter the DIRECTORY command at the DCL prompt as follows. Specify the asterisk wildcard character (*) for the file name and the appropriate language-related file type. These file types are summarized in Table 1–6. For example, to display the programs written in COBOL, enter:

```
$ DIR SQL$SAMPLE:*.SCO
```

Table 1–6 Program File Types

Language	File Type
Ada	SQLADA
C	SC
COBOL	SCO
FORTRAN	SFO
Pascal	SPA
PL/I	SPL

To locate SQL module language programs, enter the following DCL DIRECTORY command:

```
$ DIR SQL$SAMPLE:*.SQLMOD
```

1.4.3 Accessing Rdb/VMS from Systems Other Than VMS Using SQL/Services

Application programs on various types of systems other than the VMS system can access an Rdb/VMS database through SQL/Services, a client/server system that consists of:

- An Application Programming Interface (API) library of callable routines that provides functions similar to the dynamic SQL statements. The data structures and SQL statement syntax accepted by SQL/Services are identical to that accepted by dynamic SQL. In addition, the API library provides routines to establish server connections and to define postprocessing of result tables.
- A server process that is present on all systems running Rdb/VMS Version 3.1 or higher. The server accesses Rdb/VMS databases through dynamic SQL.

The SQL/Services API library communicates with the server process in a manner that is virtually transparent to the application. To do that, SQL/Services uses the same kind of structure as SQL: the SQL Descriptor Area (SQLDA) and the SQL Communications Area (SQLCA). Other than ensuring that DECnet is installed on the client and server systems, no knowledge of networking is required to develop SQL/Services applications.

1.5 Accessing Rdb/VMS Using RDO, RDML, and RDBPRE

Rdb/VMS supplies several other interfaces in addition to SQL. This section discusses the other Rdb/VMS interfaces supplied with your kit and outlines the online program examples located in the RDM$DEMO directory that use those interfaces.

1.5.1 Introducing RDO, RDML, RDBPRE, and Other Interfaces

Other interfaces to Rdb/VMS include:

- RDO/Callable RDO

  The Relational Database Operator (RDO) interface consists of data definition and manipulation statements that you can use either interactively or in application programs. The Callable RDO interface lets you access an Rdb/VMS database from a program written in any host language supported by the VAX Procedure Calling Standard.

Note *Callable RDO uses more resources than either RDBPRE or RDML. If you prefer not to use SQL, and you are programming in BASIC, C, COBOL, FORTRAN, or Pascal, you should use the RDBPRE or RDML interfaces rather than Callable RDO. However, you will have to use Callable RDO to perform certain data definition tasks.*

- RDML

  The Relational Data Manipulation Language (RDML) consists of data manipulation statements that you can embed in C and Pascal application programs. These programs are processed by the RDML preprocessor. After preprocessing, you can submit the resulting source code to the host language compiler.

- RDBPRE

  The RDBPRE preprocessor lets you access an Rdb/VMS database from BASIC, COBOL, or FORTRAN programs. You can embed statements directly in your host language program.

In addition, DATATRIEVE, RALLY, DECdecision, TEAMDATA, and VIDA are products that can be used to access Rdb/VMS databases.

1.5.2 Using the Online Program Examples for RDO, RDML, and RDBPRE

Online sample programs are available for use with the RDBPRE and RDML preprocessors and with Callable RDO. These programs are in the directory specified by the logical name RDM$DEMO, and they are referred to extensively in the language-specific chapters of the *VAX Rdb/VMS Guide to Using RDO, RDBPRE, and RDML*. You can create an executable form (image) of any source program example for which your system has language support and then run the image to see how the program works. You can also create hardcopy listings of program sources and use them as references when you create your own applications.

The exact file names for the programs vary depending on the source language and the interface being used. File names containing the following strings are available:

- SAMPLE

  A program that performs numerous data manipulation operations on the sample PERSONNEL database
- CALL_OTHER

  A module called from the SAMPLE program that passes that program information for preprocessing and returns control to the SAMPLE program
- ERROR

  A module that handles run-time errors for the SAMPLE program
- DEPTFOR.FOR

  A program that performs several data manipulation operations on the PERSONNEL database
- F_DDL_STMNT.FOR

  A program that demonstrates how to perform data definition tasks from an RDBPRE FORTRAN program using Callable RDO

Table 1–7 lists the available programs with their supported languages, and indicates the applicable interface.

Table 1–7 Online Program Examples for RDBPRE and RDML

Language	Program	Interface
BASIC	B_SAMPLE.RBA	RDBPRE
	B_CALL_OTHER.RBA	RDBPRE
	B_CALLABLE_ERROR_HANDLER.BAS	Callable RDO
	B_ERROR_HANDLER.BAS	RDBPRE
COBOL	COB_SAMPLE.RCO	RDBPRE
	COB_CALL_OTHER.RCO	RDBPRE
	COB_CALLABLE_ERROR_HANDLER.RCO	Callable RDO
FORTRAN	F_SAMPLE.RFO[1]	RDBPRE
	F_CALLABLE.FOR	Callable RDO
	F_CALLABLE_ERROR_HANDLER.FOR	Callable RDO
	DEPTFOR.FOR	Callable RDO
	F_DDL_STMNT.FOR	RDBPRE/Callable RDO
C	C_SAMPLE.RC	RDML
	C_CALL_OTHER.RC	RDML
	C_ERROR.RC	RDML
Pascal	P_SAMPLE.RPA	RDML
	P_CALL_OTHER.RPA	RDML
	P_ERROR.RPA	RDML
PL/I	PLI_SAMPLE.PLI	Callable RDO
	PLI_CALL_OTHER.PLI	Callable RDO

[1]Calls many other programs

1.6 Maintaining and Tuning Rdb/VMS Databases with RMU

Rdb/VMS provides the Rdb/VMS Management Utility (RMU) to aid in the administration and maintenance of the database. This utility provides tools to perform the following functions:

- Analyze data storage patterns

  The RMU/ANALYZE command collects and displays information on how data is being stored in the database within both storage areas and logical areas, and describes in detail the index structure or structures for the database.

- Back up and restore the database

  The RMU/BACKUP command creates a full or partial (incremental) backup file of the database or the after-image journal file. The backup operation can be performed on line or off line. The RMU/BACKUP utility is **multithreaded** (that is, multiple tape devices can be used simultaneously) to allow quick and efficient performance. The RMU/RESTORE command restores the database to the condition it was in at the time of the last full or incremental backup operation performed with the RMU/BACKUP command. You can specify a new after-image journal file for the database while restoring it. You can use the RMU/VERIFY command before backing up your database to ensure that the database is not corrupt.

- Open and close the database

  The RMU/OPEN command manually opens the database. The RMU/CLOSE command manually closes the database, letting you perform specific maintenance activities.

- Display the internal contents of various database files

  The RMU/DUMP command displays the contents of the database root file, storage area files, journal files, snapshot files, the backup file, and information about the current users of the database, including all users in a VAXcluster environment.

- Monitor the database

  The RMU/MONITOR command starts or stops the Rdb/VMS database monitor process on your system. You can create a new version of the monitor log file for the database monitor running on your node. An Rdb/VMS monitor process must be running on each node on which you use Rdb/VMS.

- Load and unload tables and views from the database

  The RMU/LOAD command loads specific tables of an Rdb/VMS database from specially structured files or from sequential record management services (VAX RMS) files. The RMU/UNLOAD command extracts the data from tables or views within an Rdb/VMS database into either sequential RMS files or specially structured files. These commands are useful to perform an initial load operation, to restructure specific tables, to create archival databases, or to access data from applications using RMS files.

- Display information about your Rdb/VMS database

  The RMU/SHOW command displays information about the Rdb/VMS system on the node from which you issue the RMU/SHOW command. You can display information about database statistics, database users, the database system being used, and the version of Rdb/VMS running on your system.

- Recover the database

  The RMU/RECOVER command recovers the database from a restored version. This command uses the after-image journal file to roll forward the database to its current state.

- Verify the integrity of the database

  The RMU/VERIFY command checks and verifies the internal integrity of data structures. It can also verify logical areas and indexes. It does not check for invalid data. Use this command before backing up your database.

1.7 Rdb/VMS Product Kits

Different sites require different capabilities from an Rdb/VMS database. For your convenience, Rdb/VMS provides the following three kits:

- **Rdb/VMS Full Development Kit**

  The Rdb/VMS full development kit lets you create and modify databases, create programs that use databases, and run programs that use databases. It lets you access Rdb/VMS databases through SQL/Services client applications. The full development kit is necessary if you will be accessing Rdb/VMS using programs written in standard programming languages (3GLs, that is, third-generation languages).

- **Rdb/VMS Interactive Kit**

  The Rdb/VMS interactive kit lets you use existing Rdb/VMS databases and build new Rdb/VMS databases using the same full-function SQL and RDO interactive interfaces that are supplied with the full development kit. It contains the SQL/Services server component. However, it does not let you preprocess data manipulation language (DML) programs; it does not include any precompilers, so you cannot develop programs.

- **Rdb/VMS Run-Time Kit**

  The Rdb/VMS run-time kit lets you use the Rdb/VMS database built with the full development kit or interactive kit. It contains the SQL/Services server component. However, it does not let you create new databases, modify data definitions of existing databases, or preprocess DML programs. The run-time kit is useful if you use prepared application solutions based on Rdb/VMS, or if you access Rdb/VMS databases from DECdecision, TEAMDATA, DATATRIEVE, or RALLY applications.

1.8 Using Rdb/VMS with Other Digital Products

You can use Rdb/VMS with other Digital products, including the following:

- ACMS

  An application development system that implements and manages transaction processing applications. Use ACMS when you must manage large transaction processing applications whose data is stored in an Rdb/VMS database.

- CDD/Plus

  A central storage facility for data definitions. Use CDD/Plus to centrally store the data definitions your application programs will be using.

- Data Distributor

  A product that gives a database administrator control of the process of distributing database access across remote nodes in a network. Use Data Distributor when your Rdb/VMS database must be distributed across remote nodes.

- DATATRIEVE

  An interactive query language that includes the capability to generate reports and graphics. Use DATATRIEVE as an interface to your Rdb/VMS database. DATATRIEVE is also useful for generating attractive reports based on data in your Rdb/VMS database. You can use DATATRIEVE to get data in a single query from Rdb/VMS, VAX DBMS, and RMS files.

- DECdecision and TEAMDATA

  An information management and decision support system that you can use as an interface to your Rdb/VMS database. Use DECdecision in DECwindows environments, or TEAMDATA in character-cell environments.

- DECtrace

  A product that collects and reports data and performance information from databases on an event basis (as opposed to products that collect on a timer basis). You can have DECtrace collect application workload information, and then have RdbExpert import that information.

- RALLY

  A fourth-generation language (4GL) environment that constructs complete applications quickly. Use RALLY applications as end-user interfaces to your Rdb/VMS database.

- RdbExpert

  A product that you can use to can optimize the physical design of your Rdb/VMS database. Using RdbExpert, you specify information about the application workload, data volume, and system environment of the database. RdbExpert applies its design rules to the schema and to the information you have supplied. It generates several design reports, as well as a command procedure that (with minimal edits) you can run to create a new database with an optimal physical design. This procedure also unloads any existing data and reloads it in the new database.

- TDMS

  A forms package that manages the display of forms and the movement of data to and from the terminal screen. Use TDMS to generate the forms your application programs will use to permit users to enter data interactively.

- VAX DBMS

  The Digital CODASYL-compliant database. VAX DBMS is intended for large, highly complex applications.

- VIDA

  A data access software product. A VIDA database contains data that resides on an IBM system. You can use SQL to retrieve data through VIDA but not to define or update a VIDA database.

2 Rdb/VMS Documentation Directory

This chapter briefly describes the manuals in the Rdb/VMS full development license documentation set. The introductory manual is listed first, followed by more advanced user guides and reference manuals, and finally by the release notes and installation guide.

All manuals other than the *VAX Rdb/VMS SQL Quick Reference Guide* are available on CDROM.

Titles dealing specifically with the RDO or RDML interfaces retain terminology associated with the Digital proprietary RDO interface (for example: relation, record, field). All other titles use terminology associated with the industry-standard SQL interface (for example: table, row, column).

VAX Rdb/VMS Introduction and Master Index (this manual)

> **Audience:** All users
>
> **Content:** Introduces VAX Rdb/VMS, the Digital relational database management system for VMS software environments. Explains major terms and concepts. Includes a glossary, a directory of Rdb/VMS documentation, and a master index that combines entries from all the Rdb/VMS manuals.

VAX Rdb/VMS Guide to Using SQL

> **Audience:** All users
>
> **Content:** Describes and illustrates the use of Rdb/VMS features with the SQL industry-standard interface. Shows how to retrieve, modify, and delete data stored in a database. Explains how to use transactions, select expressions, joins, and views. Also explains how to use SQL within programs. You can use this guide as a tutorial for learning the major features of SQL.

VAX Rdb/VMS Guide to Using SQL/Services

Audience: Application programmers

Content: Describes how to develop application programs that use SQL/Services, a client/server software component of Rdb/VMS that allows programs, from various remote computers running the Macintosh, MS-DOS, OS/2, ULTRIX, ULTRIX for RISC, or VMS operating systems to access Rdb/VMS or VIDA databases on a VMS server system.

VAX Rdb/VMS Guide to Distributed Transactions

Audience: Programmers

Content: Describes the two-phase commit protocol and distributed transactions, explains how to start and complete distributed transactions using SQL, RDBPRE, and RDML, and how to recover from unresolved transactions using RMU commands.

VAX Rdb/VMS Guide to Database Design and Definition

Audience: Database designers and administrators

Content: Explains how to design a database and how to set up definitions of database entities. Also guides you from the analysis of your organization's information needs, through logical and physical database design processes, and through the actual creation of the database. Discusses database security and use of the VAX CDD/Plus data dictionary.

VAX Rdb/VMS Guide to Database Maintenance and Performance

Audience: Database administrators and operators

Content: Shows how to use the Rdb/VMS database maintenance utilities to keep the database running and to keep its data consistent. Explains how to perform backup and restore operations, how to handle database journaling, how to evaluate and improve your database performance, and how to optimize the way in which your database uses system resources.

VAX Rdb/VMS Guide to Database Tuning

Audience: Database administrators

Content: Introduces the concept of tuning, and explores how tuning the system, the database, and the application can affect database performance. Outlines a series of steps to follow in identifying, analyzing, isolating, and solving a performance problem, and in monitoring the resulting solution. Includes a set of decision trees that provide an organized approach to solving some common database tuning problems.

VAX Rdb / VMS Guide to Using RDO, RDBPRE, and RDML

Audience: All users

Content: Describes and illustrates the use of Rdb/VMS features with the Relational Database Operator (RDO) Digital-proprietary interface. Shows how to retrieve, modify, and delete data stored in a database. Explains how to use transactions, record selection expressions, relational joins, and views. Also explains how to use RDO, RDML, and Callable RDO within programs. You can use this guide as a tutorial for learning the major features of the RDO interactive utility.

VAX Rdb / VMS SQL Reference Manual

Audience: All users

Content: Contains complete reference information on features of Rdb/VMS that you can specify using the industry-standard SQL interface. Gives syntax and usage for all data definition statements, data manipulation statements, database maintenance utility statements, statements for setting up the interactive environment, and VAX Data Distributor statements.

VAX Rdb / VMS SQL Quick Reference Guide

Audience: All users

Content: Summarizes the information in the *VAX Rdb / VMS SQL Reference Manual.*

VAX Rdb / VMS RDO and RMU Reference Manual

Audience: All users

Content: Contains complete reference information on features of Rdb/VMS that you can specify using the Relational Database Operator (RDO) utility. Describes major Rdb/VMS terms and concepts. Gives syntax and usage for all data definition statements, data manipulation statements, database maintenance utility statements, VAX Data Distributor statements, and statements for setting up the interactive environment. Also includes a chapter on the Rdb/VMS Management Utility (RMU) for administrative and maintenance tasks.

RDML Reference Manual

Audience: Programmers

Content: Describes the syntax and usage of the Relational Data Manipulation Language (RDML), which can be embedded in VAX C or VAX Pascal programs to access Rdb/VMS or Rdb/ELN databases.

VAX Rdb / VMS Release Notes

Audience: All users

Content: Includes information about new features, problems, restrictions, and sometimes other important information about the current Rdb/VMS release. Sometimes contains material that became available too late for inclusion in the other Rdb/VMS manuals.

VAX Rdb / VMS Installation Guide

Audience: System managers

Content: Explains how to install Rdb/VMS and run the Installation Verification Procedure (IVP).

Glossary

This glossary defines terms used in the documentation for VAX Rdb/VMS and related products.

access control entry (ACE)

ACL-style protection. An individual entry on an access control list.

See also access control list (ACL) and ACL-style protection.

access control list (ACL)

ACL-style protection. A list that defines which users can access a schema, table, or view and what operations they can perform. The SQL statements GRANT and REVOKE create, modify, and delete entries in an ACL. Position of an entry on an access control list can affect the privileges allowed.

See also access control entry (ACE), ACL-style protection, and privilege.

access mode

Another term for access mode is lock type.

See lock type and reserving option.

access privilege set (APS)

The generic term used to refer to either ACL-style or ANSI-style protection schemes.

See also ACL-style protection and ANSI-style protection.

access rights

Other terms for access rights are rights or privileges.

ACE

See access control entry (ACE).

ACL

See access control list (ACL).

ACL-style protection

A protection scheme that involves access control lists in which the position of an access control entry is critical.

See also access control entry, access control list, access privilege set, and ANSI-style protection.

ACMS

The VAX ACMS, an application control and management system, is a transaction processing monitor that is layered on VMS and is used to define, run, and control transaction processing applications. You can use ACMS to control existing applications and applications developed with ACMS. It provides a task implementation method that uses high-level definitions to replace complex application code. These definitions reduce the programming time and maintenance costs of traditional application programs that do comparable work. ACMS supports offloaded terminal processing, which allows the execution of ACMS tasks on remote nodes.

See also distributed transaction processing, online transaction processing, and transaction processing.

actual parameter

A parameter declaration in a host language program used in a call statement to a procedure in an SQL module file. Another term for call parameter.

after-image journal (AIJ)

A file that contains copies of records after they have been updated. You can use the after-image journal to reconstruct a saved backup file of a database up to the last successfully completed transaction.

See also before-image journal.

aggregate expression

Another term for function.

AIJ

See after-image journal (AIJ).

alarm

See security alarm.

alias

A qualifier for column names that is used in place of a full table or view name in an SQL SELECT statement. Users can qualify a column name with its table or view name, or with an arbitrary alias they specify in the FROM clause of an SQL statement. SQL requires aliases as qualifiers instead of table or view names in statements that join a table with itself. Another term for alias is context variable.

allocation

The number of pages allotted to each storage area or snapshot file of the database in the SQL CREATE SCHEMA statement.

See also extent, multifile database (MFDB), and storage area.

allow mode

Another term for allow mode is share mode. *See* reserving option.

anchor

A VMS directory that contains all the files that describe a CDO dictionary and directory system.

See also CDD/Plus and Common Dictionary Operator (CDO) utility.

ANSI

American National Standards Institute, the organization that approved a standard, ANSI X3.135-1989, for SQL database languages. That standard specifies the syntax and semantics of interfaces to define and query databases using SQL. ANSI is also developing an extended SQL standard, SQL2, in conjunction with the International Standards Organization (ISO).

See also ISO.

ANSI-style protection

A protection scheme that involves a set of privileges defined at both the system level and the individual level. In contrast to ACL-style protection, the position of the user within the access privilege set does not affect the privileges granted to that user.

See also access privilege set (APS) and ACL-style protection.

API

See Application Programming Interface (API).

Application Control and Management System (ACMS)

See ACMS.

application program

A sequence of instructions and routines, not part of the basic operating system, designed to serve the specific needs of a user. An application program can use a database system, a fourth-generation language like DATATRIEVE, or RMS to access data.

See also run unit.

Application Programming Interface (API)

In SQL/Services, a library of callable routines that are linked into the client executable image.

See also client, server, and SQL/Services.

APS

See access privilege set (APS).

area

See storage area.

area bit map (ABM)

A database page found in storage areas with uniform page format in both single-file and multifile databases. The first page of a logical area is an area bit map. The area bit map can be used to speed up sequential scans of table rows.

See also area inventory page (AIP) and uniform page format.

area inventory page (AIP)

A database page that maintains a queue of database area pointers to logical area bit maps. The area inventory page points to the area bit map for each table.

arithmetic expression

A value expression formed by combining one or more numeric value expressions with arithmetic operators.

array

A data structure that consists of more than one data item, in which all data items have the same data types and are referred to by the same variable name.

ascending order

A sorting order that starts with the lowest value of a sort key and proceeds to the highest value, in accordance with the rules for comparing data items.

See also descending order and sort key.

ASCII

A computer character set and collating sequence. The acronym stands for American Standard Code for Information Interchange. The ASCII character set occupies the first 128 positions of the DEC Multinational Character Set (MCS) that is integral to VAX systems.

ASSOCIATE_STR

A data structure passed to the SQL/Services API routine that allows an application to modify association characteristics.

See also Application Programming Interface (API), association, and SQL/Services.

association

A communications link and related context between the SQL/Services client and database server processes. In this case, the context refers to the data structures that the software creates and uses (on both ends of the link) to keep track of what is happening.

See also client, server, and SQL/Services.

attach (to a database)

The operation by Rdb/VMS that establishes a connection between your program or interactive session and a database. This connection is identified by a database handle that Rdb/VMS assigns as part of establishing the connection. The DECLARE SCHEMA statement in SQL and the INVOKE DATABASE statement in RDO attach to the specified database(s). The term *attachment* is sometimes used instead of the term *attach*.

See also database handle, and detach.

attachment

A connection between a host program or interactive utility and the database it will access. The term *attach* is sometimes used instead of the term *attachment*.

attribute

See field attribute.

audit

See security audit.

audit event

See security event.

audit trail

The list of audit activities (security events) that Rdb/VMS writes to a database security audit journal when the audit feature is enabled for the database by the RMU/SET AUDIT command.

See also security audit, security event and security audit journal.

authorization identifier

In SQL, a name specified in a DECLARE SCHEMA or CREATE SCHEMA statement that identifies the schema in subsequent SQL statements.

See also database handle.

backup operation

An operation that creates a backup copy of the database. A backup database is used to restore the database after a hardware or software failure.

There are two types of backup operations:

- RMU/BACKUP

  Use the RMU/BACKUP command from the DCL level to create a database backup file (default file type RBF) that can be restored using the RMU/RESTORE command.

- EXPORT

  Use the EXPORT statement to migrate a database from one database management system to another, or to restructure a database.

  Migration changes include taking an existing Rdb/VMS database, backing it up with an EXPORT statement, and then using the IMPORT statement to convert the internal data structures so that they are compatible with a newly installed version of Rdb/VMS. Restructuring changes include taking a single-file database, backing it up using the EXPORT statement, and then using the IMPORT statement to define one or more storage areas in a new multifile database.

See also export operation, import operation, and restore operation.

base table

A table that is physically stored in the database.

See also view.

batch processing (Rdb/VMS)

A mode of computer operation in which the SQL or RDO statements and the data that control the actions of the computer are entered by a programmed script rather than by a person at a terminal.

See also interactive processing (Rdb/VMS).

batch-update transaction

A transaction mode that executes without the overhead, or security, of a recovery-unit journal. The batch-update transaction is intended for initial loads of databases. Should *any* error occur during the batch-update transaction, the entire database is permanently corrupted.

See also lock type, reserving option, share mode, and transaction.

before-image journal

A file that contains copies of records before they have been updated. Rdb/VMS uses before-image journaling to automatically remove updates to a database when a transaction is rolled back. Another term for before-image journal is recovery-unit journal (RUJ).

See also after-image journal (AIJ) and idempotent.

before-image journaling

In Rdb/VMS databases, the method of keeping track of changes you make to the database in the before-image journal file. It is this journaling that lets you issue a ROLLBACK statement to remove changes you made to the database.

binary language representation (BLR)

The binary language representation of requests to the database. In other words, the binary form of data manipulation language. A request in this context is a sequence of instructions given to the database system by a host program for compilation and possible execution.

See also data manipulation language (DML), metadata binary language representation (MBLR), and request.

bind

Another term for attach.

bitvector

A structure that is part of the area bit map (ABM) page that contains set bits that indicate the corresponding space area management pages that have entries for a logical area.

block structure

A programming construct that defines the lexical scope of program variables. For example, C is a block structured language where a left brace ({) denotes the beginning of a block and a right brace (}) denotes the end. Blocks are often nested.

BLR

See binary language representation (BLR).

Boolean expression

Another term for predicate or conditional expression.

See also Boolean operator, conditional operator, and predicate.

Boolean operator

A keyword that lets you join two or more predicates to form a complex predicate. Boolean operators negate a predicate (NOT), specify a combination of predicates (AND), or specify a list of alternative predicates (OR). Another term for Boolean operator is logical operator.

broadcast message

A text line that lets you know of a system event, such as a system shutdown or the receipt of mail.

B-tree

A sorted index structure for a specified table (system or user). Specified by using the TYPE IS SORTED clause in the DEFINE INDEX statement.

See also hashed index, index, and sorted index.

built-in function

Another term for function.

cache

The process of storing blocks in memory for future use; used to minimize physical transfer of data between mass storage devices and memory.

Callable RDO

An interpretive call interface that consists of a single external routine that accepts an Rdb/VMS data manipulation language (DML) or data definition language (DDL) statement as a parameter. You can call this routine from any language that adheres to the VAX Procedure Calling Standard. Callable RDO lets a program use Rdb/VMS even if no preprocessor exists for the language.

Callable RDO supports embedded DML statements in VAX BASIC, VAX C, VAX COBOL, VAX FORTRAN, and VAX Pascal source programs. To embed DDL statements or to use dynamic strings in Rdb/VMS DML statements, you must use Callable RDO.

See also Relational Database Operator (RDO).

call interface

A mechanism by which a program accesses a software product.

call parameter

Another term for actual parameter.

candidate key

Any column or set of columns that uniquely identifies the individual rows of a table. Candidate key is similar to a primary key. For example, in a table of employee information, the employee identification number is a candidate key. When more than one primary key exists, the alternatives are called candidate keys. Of the two or more candidate keys, a primary key is selected, with the remaining keys becoming secondary keys.

See also foreign key, primary key, and secondary key.

cardinality

The number of rows in a table or the number of indexed entries in a defined index.

See also index, row, and table.

Cartesian product

For multiple tables that are joined by a select expression in SQL, the Cartesian product is a result table that consists of every possible combination of all the rows and columns of each table. SQL forms the Cartesian product when it evaluates a FROM clause that names more than one table. Another term for Cartesian product is cross product.

See also join.

cascading delete

See cascading update.

cascading update

An update operation performed by a trigger definition that will update appropriate rows from any necessary table. For example, in the sample personnel database, you could have a cascading update specify that whenever an employee leaves the company, the row for that employee will be deleted not only from the EMPLOYEES table, but from the JOB_HISTORY, SALARY_HISTORY, and DEGREES tables as well. Such an operation would be called a cascading delete operation.

See also table-specific constraint and trigger.

CDD/Plus

An abbreviated name of VAX CDD/Plus, a VMS software component through which users can define, organize, and control access to the data and forms definitions required by applications. Users can access an Rdb/VMS database through a database definition they store in CDD/Plus. Rdb/VMS documentation sometimes refers to CDD/Plus simply as the data dictionary.

The data dictionary lets you ensure the integrity of the shared metadata and provides procedures you need to analyze, maintain, manage, and design your business' database; it provides a useful centralized repository for information management.

CDD/Plus supports metadata created in an appropriate format for both the Dictionary Management Utility (DMU) and the Common Dictionary Operator (CDO).

Support for CDD/Plus by Rdb/VMS enables you to:

- Define global column and row definitions in a shareable dictionary
- Directly copy the column or row definition(s) or both from the dictionary to an Rdb/VMS database when you define new Rdb/VMS columns or tables in SQL or RDO
- Receive informational messages about the CDD/Plus column and row definitions that are used by other Rdb/VMS databases
- Integrate shareable dictionary definitions into an Rdb/VMS database (the INTEGRATE SCHEMA FILENAME statement) or integrate database definitions to a CDD/Plus dictionary directory (the INTEGRATE SCHEMA PATHNAME statement)

Rdb/VMS does not require CDD/Plus.

CDO

See Common Dictionary Operator (CDO) utility.

CDROM

Compact disk read-only memory. New type of disk that is used as a distribution medium for software and documentation.

character string

A string of characters (bytes) that is identified by an address and a length.

checksum

A sum of digits or bits used to verify that a number or an operation is valid. For a database page, the sum of all bits on the page.

client

A service requester. In SQL/Services, an application program that uses the Application Programming Interface to request services from the SQL/Services server.

See also Application Programming Interface (API), SQL/Services, and server.

clump

A set of adjacent pages within a storage area when uniform page format is specified. These pages are devoted to specific logical areas which either store rows from a specific table or nodes from one or more sorted indexes on that table by the area's logical ID.

clumplet

A portion of the RDBVMS$TRIGGER_ACTIONS segmented string field that contains a trigger action for a specified trigger. The RDBVMS$TRIGGER_ACTIONS field is part of the RDBVMS$TRIGGERS system relation.

See also trigger.

cluster

See VAXcluster.

clustered index

Hashed indexes whose hash buckets reside on the same page of a storage area file as their associated table rows. Clustering hash buckets with rows may make it possible for one input/output operation to load the hash bucket and rows into memory.

See also hash bucket and hashed index.

CODASYL

An acronym for the Conference on Data Systems Languages, the committee that designed the COBOL language and provided the guidelines used in the development of VAX DBMS, the Digital network database.

CODASYL-compliant

Any database system that conforms to the guidelines set by the Conference on Data Systems Languages. VAX DBMS is CODASYL-compliant.

collating sequence

The sequence in which characters are ordered for sorting, merging, and comparing.

See also National Character Set (NCS) utility.

column

The vertical dimension of a table. A column has a name and a data type, and all values in a column have that same data type. Another terms for column is field.

See also data item and domain.

column select expression

A select expression that specifies a one-column result table.

SQL accepts column select expressions as arguments to IN and quantified predicates, and more generally as value expressions:

- As arguments to IN and quantified predicates, column select expressions specify a collection of values to which SQL compares a value expression. Thus, column select expressions used as arguments to those predicates can return one or more values.
- As a type of value expression, column select expressions specify a single value. Thus, a column select expression used as a value expression should not return more than one value. If it does, SQL evaluates the value expression *based on the first value returned* by the column select expression. For the same reason, a column select expression used as a value expression cannot include GROUP BY or HAVING clauses.

Other terms for column select expression are record selection expression (RSE) and subquery.

See also record stream and select expression.

comment character

A character that begins a line of descriptive text in a program or procedure; it does not affect program or procedure execution. Comment lines begin with a comment character reserved by the language you are using. Some typical comment characters are the exclamation point (!), the asterisk (*), and the letter C. The exclamation point is the comment character used by SQL and the RDO utility. You can also use a double hyphen (--) in interactive SQL and SQL module language. Comment lines end with a carriage return.

commit

To make permanent any changes to the database made during the current transaction. Use the COMMIT statement to commit changes.

See also roll back and transaction.

commit phase

The second phase of the two-phase commit protocol. During the commit phase, the coordinator of the distributed transaction instructs each resource manager to make permanent all changes to the database, that is, to commit the changes. The resource managers commit the changes and the distributed transaction is completed.

See also coordinator, distributed transaction, prepare phase, resource manager, and two-phase commit protocol.

common data dictionary

See CDD/Plus.

Common Dictionary Operator (CDO) utility

The user interface to CDD/Plus. With CDO, you can access any CDO or DMU dictionary to which you have privileges, create or delete field and record definitions, and run all CDO commands.

See also Dictionary Management Utility (DMU).

communication server

In SQL/Services, a multithreaded process on a VMS server system that accepts Application Programming Interface (API) requests, assigns them to execution server processes, and asynchronously returns execution results to client applications. The communication server sets up the process pool during the initializing phase at system startup time, in preparation for handling requests during the working phase.

See also Application Programming Interface (API), client, execution server, initializing phase, process pooling, server, and working phase.

complex predicate

A predicate that combines any number of predicates using the Boolean operators AND, OR, or NOT.

compress

To reduce in size. When data is compressed, free space is eliminated as much as possible.

compressed index

An index in which information in the index node is compressed so that data takes up less space.

See also compress, index.

COMPUTED BY columns

Virtual columns that appear in a table or view definition, but not in the physical row. Because the value of a COMPUTED BY column is computed as part of a statement, it occupies no space in the row.

concurrency

The simultaneous use of a database by more than one user.

See also consistency.

conditional expression

Another term for predicate or Boolean expression.

See also conditional operator and predicate.

conditional operator

A keyword that specifies how you want to compare value expressions in a predicate. SQL uses these conditional operators:

=	(equal to)
<>	(not equal to)
<	(less than)
<=	(less than or equal to)
>	(greater than)
>=	(greater than or equal to)
BETWEEN	
CONTAINING	
EXISTS	
IN	

IS

LIKE

STARTING
WITH

UNIQUE

Another term for conditional operator is relational operator.

configuration file

In SQL/Services, a file that the communication server reads at system startup time to set up the process pool. The SQLSRV$CONFIG.DAT file in the SYS$STARTUP system directory contains one or more definitions, each consisting of a set of parameters. Each definition describes the characteristic of a subpool within the process pool. An initial version of the file is stored on the system during installation. The system manager can modify the file at any time to better suit application requirements.

See also communication server, process pooling, and subpool.

consistency

The level to which a database system guarantees that records being read by a user cannot be changed simultaneously by other users.

See also concurrency.

constant

Another term for literal.

constraint

A condition that restricts the values that can be stored in a table. When you insert and update column values, the constraint checks the values against the conditions specified by the constraint. If a value violates the constraint, an error message is generated and the insert or update statement fails. You specify constraints in SQL CREATE and ALTER TABLE statements. Optionally, you supply a name for the constraints following the SQL DIAGNOSTIC keyword.

context file

A special SQL command procedure that contains SQL-specific declarations such as DECLARE SCHEMA and DECLARE TRANSACTION statements. SQL applies the declarations in a context file to the precompilation and execution of a host language program.

A context file is a way to manage schema and transaction contexts for the program without coding these declarations in source code. Context files help limit SQL statements in programs to generic SQL syntax. They let the program be more easily transported for use with a variety of database products. The SQL precompiler accepts a context file specification as an optional argument when it processes a host language module.

context structure

A data structure in SQL that contains a distributed transaction identifier (TID).

See also distributed transaction identifier (TID).

context variable

Another term for alias.

control break

Partitions between groups of rows with the same value in an intermediate result table. Specify control breaks with the GROUP BY clause.

See also intermediate result table.

coordinator

A component of a distributed transaction. The coordinator is the transaction manager on the node where the application started a distributed transaction. The coordinator orchestrates the distributed transaction.

See also DECdtm services, distributed transaction, resource manager, transaction manager, and two-phase commit protocol.

correlated reference

Another term for outer reference.

correlated subquery

A subquery (or column select expression) that contains an outer reference.

correlation name

Another term for alias.

cross operation

Another term for join.

cross product

Another term for Cartesian product.

cursor

A result table defined by the select expression in an SQL DECLARE CURSOR statement. Unlike other result tables, a cursor can exist throughout execution of more than one statement. Host language programs require cursors because programs must perform operations one row at a time and therefore may execute statements more than once to process an entire result table.

See also list cursor, result table, and table cursor.

database

A collection of interrelated data on one or more mass storage devices. The collection is organized to facilitate efficient and accurate inquiry and update. In a database, more than one user can access the data at the same time. Data integrity and security are provided by the database. Specifically, database refers to the data definitions, user data, and the database system files associated with a schema.

See also CODASYL, hierarchical database, network database, relational database, and schema.

Database Control System (DBCS)

The component that, together with the VMS operating system, provides run-time control of database processing.

database execution server

In SQL/Services, one of two types of execution servers (database and generic execution servers) that execute Application Programming Interface (API) requests for the communication server. Database servers are already attached to an Rdb/VMS database at system startup time and provide access to Rdb/VMS databases in a specific user environment. Having servers already attached to a database eliminates the need for each API request to make a separate database attach.

See also Application Programming Interface (API), communication server, execution server, generic execution server, and server.

database handle

Another term for authorization identifier. RDO and RDML use *database handle* instead of the term authorization identifier.

See also attach and detach.

database key

See dbkey.

database management system (DBMS)

A system for creating, maintaining, and accessing a collection of interrelated data records that can be processed by one or more applications without regard to physical storage. Data is described independently of application programs, providing ease in application development, data security, and data visibility.

Note that the abbreviation "DBMS" is often used in general technical literature and in the trade press to mean "database management system." Do not confuse this generic use of the term DBMS with references to the specific Digital software product VAX DBMS, a CODASYL-compliant database management system.

See also CODASYL, database, hierarchical database, network database, and relational database.

database name

A name that a person, program, or product must supply to identify and access a database.

database page

The structure used to store and locate data in an Rdb/VMS database. Rdb/VMS database pages consist of 1 or more disk blocks of 512 bytes each. The default size of a database page is 2 blocks, or 1024 bytes.

database root file

In a single-file database, the database root file contains data, metadata, and system information. In a multifile database, this file contains only system information. The metadata is stored in the RDB$SYSTEM storage area. Data is stored in the RDB$SYSTEM or other storage area files. The database root file has the file type RDB.

See also storage area.

database tuning

The process of adjusting system or database parameters to attain optimum resource use for database applications without using unnecessary or excess central processing unit, memory, or input/output capacity.

See also process pooling, query optimizer, SPAM page, and threshold.

data compression

A technique that reduces data to its smallest possible size to optimize data storage. Enabled by default, data compression can be disabled by specifying the CREATE STORAGE MAP . . . DISABLE COMPRESSION statement or the ALTER STORAGE MAP . . . DISABLE COMPRESSION statement for a given table.

data definition language (DDL)

A set of statements that lets you define the structure and characteristics of stored data. You use data definition language to describe databases characteristics, storage area files, columns, tables, views, indexes, and constraints. Both the SQL and RDO interactive interfaces include Rdb/VMS data definition language.

See also CDD/Plus and metadata binary language representation (MBLR).

Data Distributor

A Digital product that lets you make complete or partial copies of databases. These database copies can be made available to users on the source node or on remote nodes. Data Distributor can copy the databases of three relational database systems: Rdb/ELN, Rdb/VMS, and VIDA. Database copies are in Rdb/VMS format.

See also extraction, Rdb/ELN, Rdb/VMS, replication, source database, source node, target database, and target node.

data element

Another term for data item.

data item

The smallest unit of data that you can retrieve from a row. A data item occupies a single column in a row. Another term for data item is data element.

See also column and domain.

data manipulation language (DML)

A set of statements that lets you store, retrieve, modify, and erase data from a database. There are several methods of manipulating data using SQL:

- Use a high-level language program to execute the SQL statements in an SQL module file.
- Embed the data manipulation statements in a high-level language, such as Ada, BASIC, C, COBOL, FORTRAN, Pascal, or PL/I. SQL has a precompiler for Ada, C, COBOL, FORTRAN, Pascal, and PL/I.

- Issue the data manipulation statements interactively.

The Rdb/VMS data manipulation languages (RDO, SQL) can be used to access a VIDA database as well as an Rdb/VMS database.

See also binary language representation (BLR).

data table

See table.

DATATRIEVE

A VAX query language and data management tool for manipulating, storing, and modifying records from RMS data files, VAX DBMS, Rdb/VMS, and Rdb/ELN databases. DATATRIEVE also generates reports and graphs from data stored in RMS files and VAX DBMS, Rdb/VMS, VIDA, and Rdb/ELN databases. SQL provides DATATRIEVE support clauses in the CREATE DOMAIN, CREATE TABLE, and CREATE VIEW statements, and in the ALTER DOMAIN and ALTER TABLE statements.

DATATRIEVE is callable from a variety of high-level languages.

data type

The characteristic of columns and host language variables that controls how Rdb/VMS interprets and stores values.

DBCS

See Database Control System (DBCS).

dbkey

A unique value that points to specific table rows in a database, identifying the address of the table row. The Database Control System (DBCS) assigns the value when a record is stored in the database. Application programs can use the DBKEY keyword in SQL statements to refer to the database key for a table row. SQL statements that retrieve rows by specifying their dbkeys offer faster access and reduced locking over indexed or sequential searches. Another term for dbkey is database key.

See also scope.

DBMS

See database management system.

DBR

The name of the process that performs database recovery. It is created by the Rdb/VMS monitor in case of:

- A (hard) executable image termination, where the usual exit mode is bypassed; this can occur when the user terminates an executable image by entering CTRL/Y and then invokes the DCL command STOP or any other command that calls a new image.
- A VMS system failure
- A VAX node exit in a VAXcluster environment

DCL

DIGITAL Command Language. A set of commands that provides an operating system interface for users. When users log in to a VMS system, they typically enter the DCL environment. In this environment, they can create files and directories and use a variety of programs and utilities to get their work done.

Interactive SQL and RDO let users issue DCL commands with the DCL Invoke ($) statement.

DCL command procedure

A sequence of DIGITAL Command Language (DCL) commands stored in a file; sometimes referred to as a DCL procedure.

See also RDO command procedure and SQL command procedure.

DDL

See data definition language (DDL).

deadlock

A situation in which two or more processes request the same set of resources and there is no method for resolving the conflict other than to reject all but one of the processes. For example, if process A has row 1 locked and requests row 2 while process B has row 2 locked and is requesting row 1, a deadlock occurs between the two processes.

DECdtm services

The VMS utility that coordinates distributed transactions by providing the underlying operating system support for distributed transactions. DECdtm services supports the two-phase commit protocol, and guarantees consistent execution of the distributed processing of applications on the VMS operating system.

See also coordinator, distributed transaction, resource manager, transaction manager, and two-phase commit protocol.

DECnet

The Digital software facility that implements the DIGITAL Network Architecture (DNA) to let a user access information on a remote computer through telecommunications lines. A DECnet–VAX network enables a VMS system to function as a network node and, for instance, allows programs running on one node to remotely access an Rdb/VMS database on another node in the same DECnet–VAX network.

DECtrace

A product that collects and reports data and performance information from databases on an event basis (as opposed to products that collect on a timer basis). You can have DECtrace collect application workload information, and then have RdbExpert import that information.

See also event and RdbExpert.

default

A value that is assumed unless or until you specifically indicate another value.

default dictionary directory

The CDD/Plus directory assigned to you when you invoke an image that uses the CDD/Plus dictionary. This directory becomes the first directory listed within path names. You can define a directory as the default by assigning a path name to the logical name CDD$DEFAULT. If you do not, the default directory is the equivalent of the CDD$COMPATIBILITY logical name. By default, CDD$COMPATIBILITY is defined as SYS$COMMON:[CDDPLUS].

See also anchor, dictionary directory, and path name.

default directory

The directory from which the VMS system reads and to which it writes all files that you create unless you explicitly name a directory.

See also directory.

default schema

The database declared without an explicit authorization identifier in the DECLARE SCHEMA statement. Table and view names in SQL statements that refer to the default schema do not have to be qualified by an authorization identifier.

In SQL module language, the authorization identifier specified in the module header is the one that designates the default schema. In precompiled SQL programs and in interactive SQL, the special authorization identifier RDB$DBHANDLE designates the default schema. In all environments, omitting an explicit authorization identifier is the same as specifying the authorization identifier that designates the default schema.

If a user or program does not issue an explicit DECLARE SCHEMA statement, SQL attempts to declare a default database using the file specification assigned to the logical name SQL$DATABASE.

See also authorization identifier and schema.

default value

The data value that is stored in the database if an insert or update operation on a row specifies no data value for that column. In SQL, you can define a default value for a domain or for a column. A default value using the SQL interface is not the same as the missing value that you can define using the RDO interface.

See also missing value.

deferred snapshots

A process by which update transactions write to the snapshot files only when a read-only transaction is in progress. Deferred snapshots eliminate the unnecessary writing of updated records to the snapshot file.

See also snapshot.

degree (of a table)

The number of columns in a table definition.

descending order

A sorting order that starts with the highest value of a key and proceeds to the lowest value, in accordance with the rules for comparing data items.

See also ascending order and sort key.

descriptor

A data structure used for passing argument types, addresses, and other information between SQL and host language programs. SQL module procedures can specify that parameters be passed by descriptor. Also, SQL uses a descriptor called the SQL Descriptor Area (SQLDA) to communicate with programs about dynamically executed SQL statements (however, note that the SQLDA is not a standard VMS descriptor).

See also SQLDA.

detach (from a database)

The operation by Rdb/VMS that closes the database stream to a database and releases resources associated with the terminated connection. The FINISH statement in both SQL and RDO detaches from the database (in addition to performing other operations).

See also attach.

dictionary

In the most general sense: an overall hierarchical storage facility that includes dictionary directories, subdictionaries, and objects. Dictionary facilities are also called repositories. In the Rdb/VMS documentation, dictionary refers to the CDD/Plus dictionary product.

As a keyword used with the SET and SHOW statements, DICTIONARY has the more limited meaning of the current location within the CDD/Plus dictionary used by the invoked database.

See also CDD/Plus.

dictionary directory

The structure for organizing data descriptions stored in the CDD/Plus dictionary. Dictionary directories are similar in function to VMS directories. They “own” other dictionary directories or dictionary objects.

See also default dictionary directory.

Dictionary Management Utility (DMU)

The utility used with VAX CDD (prior to CDD/Plus) to create and maintain the dictionary directory hierarchy and its associated access control and history lists. DMU is still available with CDD/Plus, but only to work with dictionary definitions that existed before CDD/Plus was installed on your system. To share definitions among more than one Rdb/VMS database, the metadata must reside in the CDO-format dictionary.

See also Common Dictionary Operator (CDO) utility.

dictionary object

A data definition stored in the CDD/Plus dictionary. Examples of objects include:

- ACMS definitions
- CDD/Plus record definitions
- DATATRIEVE domains, records, procedures, plots, and tables
- VAX DBMS schemas, areas, sets, and records
- Rdb/VMS database entry
- TDMS forms, requests, and request library definitions

Data Distributor does not store any definitions for a target database in a CDD/Plus dictionary; however, you can use the INTEGRATE statement to copy the database definitions of extractions and replications to a CDD/Plus dictionary yourself.

See also CDD/Plus, extraction, replication, and target database.

DIGITAL Command Language (DCL)

See DCL.

directory

A file that catalogs a set of files stored on disk or tape. The directory includes the name, type, and version number of each file in the set.

See also default directory and dictionary directory.

DISTINCT clause

Used in a select expression to find unique values for a column and to eliminate repeating records.

distributed transaction

A transaction that groups more than one database or more than one database attachment together into one transaction even if the databases are located on different nodes. Rdb/VMS uses the two-phase commit protocol of DECdtm services to guarantee that if one operation in a transaction cannot be completed, none of the operations is completed.

See also coordinator, DECdtm services, resource manager, transaction manager, and two-phase commit protocol.

distributed transaction identifier (TID)

An identifier generated by DECdtm services to keep track of a distributed transaction and of the resource managers and transaction managers that are involved in a particular distributed transaction.

See also coordinator, distributed transaction, resource manager, and transaction manager.

distributed transaction processing

The processing of ACMS tasks on two separate nodes, a front end for forms processing and a back end for processing against the database. An ACMS user or task submitter logged in to an ACMS system on one node can select tasks in an application on an ACMS system on another node. ACMS uses the DECnet network to communicate transparently between nodes without rewriting applications or tasks, and can distribute processing between nodes in a VAXcluster, a local area network, a wide area network, or any combination of the three.

See also ACMS and transaction processing.

DML

See data manipulation language (DML).

DMU

See Dictionary Management Utility (DMU). *See also* Common Dictionary Operator (CDO) utility.

domain

The set of values that a table column can have. The CREATE DOMAIN statement specifies the set of values by associating a data type with a domain name. CREATE TABLE statements can use the domain in column definitions.

See also column and data item.

dynamic cursor

A cursor used in dynamic SQL programs. You explicitly specify the cursor name at compile time; however, you do not explicitly specify the SELECT statement. Instead, you specify the name of a prepared statement. In other words, the cursor name is known at compile time, but the SELECT statement is not known until run time.

See also cursor, dynamic SQL, and extended dynamic cursor.

dynamic interface

Another term for dynamic SQL.

dynamic SQL

A set of special SQL statements (PREPARE, DESCRIBE, EXECUTE, EXECUTE IMMEDIATE, and RELEASE) and data structures (SQLCA and SQLDA) that let programs accept or generate SQL statements for which no executable form exists until the program runs. Unlike statements embedded in programs or contained in SQL module files, such dynamically executed SQL statements are not part of any source code but are created while the program runs. Dynamic SQL is useful when programs cannot predict before they execute the type of SQL statement they will need to process.

To dynamically execute statements, programs either embed the special dynamic SQL statements in host language source files for precompilation or call SQL module files that contain procedures with the special statements.

See also client, dynamically executable statement, executable statement, nonexecutable statement, server, and SQL/Services.

dynamically executable statement

An SQL statement that can be formulated and executed at run time.

See also dynamic SQL, executable statement, and non executable statement.

edit string

A character or group of characters that controls how DATATRIEVE displays data in a column or domain.

embedded DML

DML statements that are embedded directly in a host language program.

embedded RDO

RDO statements that are embedded directly in a host language program rather than being issued in the interactive RDO utility. Programs that contain embedded RDO must be processed by the preprocessor to convert the embedded RDO statements to a form that is understandable by the host language compiler.

embedded select

Another term for singleton select.

embedded SQL

SQL statements that are embedded in a host language program rather than in an SQL module language procedure or issued in the interactive SQL utility. Programs that contain embedded SQL statements must be processed by the SQL precompiler before compilation and execution to convert the embedded SQL statements to a form that is understandable by the host language compiler.

equijoin

A join operation that matches values in a column from one table with those in a corresponding column in another table.

event

An occurrence of some activity within a facility. DECtrace tracks two types of events: duration and point. **Duration events** have logical beginning and ending points. **Point events** occur instantaneously. DECtrace allows you to collect and report on data based on a set of predefined Rdb/VMS events.

See also DECtrace and security event.

executable image

An image that can be run in a process. When run, an executable image is read from a file for execution in a process.

executable statement

In host language programs, SQL statements that undergo processing during precompilation or module compilation, but do not actually execute until the program runs. In interactive SQL, statements whose operation is processed immediately by SQL.

See also dynamic SQL, dynamically executable statement, and nonexecutable statement.

execution server

A process that executes Application Programming Interface (API) requests for the communication server. There are two classes of execution servers that the communication server can create: generic or database. By default, the communication server uses generic execution servers when a request does not explicitly name a database execution server.

See also communication server, database execution server, generic execution server, and server.

export operation

Makes a copy of a database in an intermediate form. Use an import operation to rebuild an Rdb/VMS database from the intermediate (RBR) file created by the export operation.

See also backup operation, import operation, and restore operation.

expression

Another term for value expression.

extended dynamic cursor

A cursor used in dynamic SQL programs. You supply parameters for the cursor name and the SELECT statement. In other words, the cursor name and select statement are not known until run time. Extended dynamic cursors let you use one set of cursor-related statements to process any number of dynamically generated statements.

See also cursor, dynamic cursor, and dynamic SQL.

extent

The number of pages by which a storage area file or a snapshot file is extended each time its space limit has been reached. The SQL CREATE SCHEMA statement specifies the initial number of pages for each storage area.

See also multifile database (MFDB), snapshot, and storage area.

extraction

In Data Distributor, the process by which complete or partial copies of a source database are transferred to a directory. Data Distributor can transfer a database to a directory on the same node as the source database or to directories on remote nodes. The source database for a Data Distributor extraction can be any of these database systems: Rdb/VMS, Rdb/ELN, and VIDA. Database copies are in Rdb/VMS format. The target database for an extraction process is not periodically updated

See also Data Distributor, replication, source database, and source node.

field

Another term for column or domain.

See also data item.

field attribute

In RDO, a condition or characteristic of a field in a record.

file

A collection of related records treated as a unit; often referred to by a logical name.

file name

The name you choose to identify a file. The file name can have from 1 to 39 characters selected from the letters A through Z, the numbers 0 through 9, and the underscore (_) or the dollar sign ($). When you name files, you can use any names that are meaningful to you.

file specification

A name that uniquely identifies a file. A full file specification identifies the node, device, directory name, file name, file type, and version number under which a file is stored.

file type

The part of a file specification that describes the nature or class of file. The file type follows a period after the file name and consists of 1 to 39 characters. The VMS operating system and VAX software products recognize many default file types used for special purposes.

In an Rdb/VMS database, the possible default file types are as follows:

- RDB

  The database root file, where information about the characteristics and physical structure of the entire database is maintained. There is one database root file per Rdb/VMS database. In a single-file database, the database root file also contains all the user's data.

- RDA

  One or more storage area files, where the user's data is located in a multifile database. System relations are stored in the default storage area. The RDA file type applies only to multifile databases.

- SNP

  One snapshot (SNP) file for each storage area (RDA) file in a multifile database when snapshots are enabled for the database. In a single-file database, there is a single snapshot file.

- RUJ

  One recovery-unit journal file created for each user who starts a read/write transaction. The recovery-unit journal file resides, by default, in the user's SYS$LOGIN directory. The RUJ file is deleted after the user detaches from the database.

- AIJ

  One after-image journal file per database if after-image journaling is enabled.

- RBF

  A backup file of the database created by the RMU/BACKUP command.

- RBR

  A backup file of the database created by the RDO EXPORT statement.

- RRD

  A file containing data definitions for a table or view created by the RMU/UNLOAD command.

- UNL

  A file containing data created by the RMU/UNLOAD command.

See also multifile database (MFDB).

foreign key

A column in a table that does not uniquely identify rows in that table, but is used as a link to matching columns in other tables.

See also candidate key, primary key, and secondary key.

formal parameter

A parameter declaration in an SQL module procedure that corresponds to an actual parameter in a calling program. The SQL statement in the module procedure uses the formal parameter name to refer indirectly to the actual parameter named in the host language call to the module procedure.

See also actual parameter.

free space

The portion of a database page that is not filled with data.

See also database page.

full path name

A name that uniquely identifies a dictionary directory, subdictionary, or object in the CDD/Plus hierarchy. The full path name is a concatenation of the given names of directories and objects, beginning with the anchor name, ending with the given name of the object or directory you want to specify, and including the given names of the intermediate subdictionaries and directories. The names of the directories and objects are separated by periods.

SYS$COMMON:[CDDPLUS]CORP.ACCOUNTING.PERSONNEL is a full path name that uniquely identifies the object PERSONNEL, an entry for an Rdb/VMS database definition.

See also logical path name and relative path name.

function

A keyword that calculates a single value based on all the values in a column of a result table or group. SQL uses these functions:

COUNT	Number of rows in a result table
SUM	Sum of the values in a column
AVG	Average of the values in a column
MAX	Largest value in a column
MIN	Smallest value in a column

Other terms for function are aggregate expression, built-in function, and statistical expression.

generic execution server

In SQL/Services, one of two types of execution servers (database and generic servers) that exeucte Application Programming Interface (API) requests for the communication server. Generic execution servers are assigned by default to execute API requests. Unlike database execution servers, generic servers are not pre-attached to an Rdb/VMS or VIDA database at system startup time. Generic servers, however, require no application or server system modification, as database execution servers require.

See also Application Programming Interface (API), communication server, execution server, database execution server, and server.

global aggregate

An expression that uses field values from one relation to group records from another. A function is then used to calculate a value for the group. For example, you can group salary records in a SALARY_HISTORY relation according to the DEPARTMENT_CODE field in the DEPARTMENT relation. Then you can use the AVERAGE function to find the average salary for each department.

global transaction

See distributed transaction.

handle

See database handle.

hash bucket

A data structure that maintains information about an index key, and a list of internal pointers (or dbkeys) to the page and row that contain the value of the index key.

Another term for hash bucket is hashed index node.

hashed index

In a hashed index, the index key value is converted mathematically to a relative page number in the storage area of a particular table. On that page is a hash bucket or hashed index node that contains pointers (or dbkeys) that point to where the row is actually stored. To find a row using the hashed index, the database system searches the hash bucket, finds the appropriate dbkey, and fetches the table row. Hashed indexes are more effective for random, direct access when the query supplies the entire index key. Hashed index structures are created when a row is stored and a hashed index has been defined for that table.

A hashed index is defined in Rdb/VMS by specifying the TYPE IS HASHED clause within the CREATE INDEX statement. Hashed indexes are available only in a multifile database.

See also hashing, hash bucket, index, multifile database (MFDB), multisegmented index, and sorted index.

hashed index node

Another term for hash bucket.

hashing

The conversion of a column's primary key value (for example, an EMPLOYEE_ID of 00167) into a database page number on which the row will be stored; subsequent retrieval operations that specify the key column value use the same hashing algorithm and can locate the row directly. Hashing provides fast retrieval for data that contains a unique key value.

See also hashed index.

hierarchical database

A type of database that organizes the relationships between record types as a tree structure (usually depicted "upside-down," with branches growing downward and out). A hierarchical database stores related records on the same branch of the tree to make data retrieval efficient.

See also database, network database, relational database, and schema.

host language program

All the host language source files that go together to make up an entire program.

host language source file

A file containing host language source code. Such files can use SQL statements in one of the following ways:

- By using host language calls to procedures contained in an SQL module file compiled by the SQL module processor, independently of the host language source file.
- By embedding SQL statements directly in the file. The statements are flagged by special delimiters. Such host language source files must be processed by the SQL precompiler before the host language statements in them are compiled.

host language variable

A variable declared in a program that the program can refer to in an embedded SQL statement.

See also parameter.

host structure

A host language variable that corresponds to a group construct of several host language variables. Once a program declares a host structure, it can refer to the host structure in some SQL statements instead of listing the host language variables that comprise it.

host variable

See host language variable.

idempotent

A state of being in which database recovery using before-image journal (RUJ) files should succeed following one or more system failures as if the recovery were done completely and successfully the first time.

See also before-image journal and recovery.

import operation

Creates an Rdb/VMS database from an intermediate RBR file. You use the import operation with an export operation to make changes to Rdb/VMS databases that cannot be made any other way.

See also backup operation, export operation, and restore operation.

index

A structure within a file or database that lets you quickly locate particular records based on key column values.

In Rdb/VMS, you can use any column or combination of columns from a row as an index key. You can also define more than one index for a given table. Rdb/VMS has two types of indexes, sorted and hashed. Hashed indexes are available only in a multifile database.

See also cardinality, hashed index, multisegmented index, and sorted index.

index fill factor

A parameter that controls the initial fullness percentage of each index node. Generally, the performance of query-intensive applications benefit from index structures with large, full nodes, while the performance of update-intensive applications can benefit from index structures with small, partially filled nodes.

See also index, index node, and sorted index.

index key

A column of a row in an indexed file or database that determines the order of search and retrieval. You can use any column or combination of columns from a row as an index key. You can also define more than one index for a given table.

See also hashed index, index, and sorted index.

index node

A sorted index data structure that contains key values and pointers (dbkeys) to rows in the database, as well as to other nodes and rows in this sorted index structure. You can control the size of a node in a sorted index by using the NODE SIZE IS clause within the SQL ALTER INDEX, SQL CREATE INDEX, and IMPORT statements.

See also hashed index, index, and sorted index.

indicator array

Another term for indicator structure.

indicator parameter

A parameter whose value indicates whether its associated main parameter has been assigned a null value, or whether or not the text string passed from a database has been truncated.

See also main parameter.

indicator structure

A one-dimensional array of host language variables declared with the signed word data type that programs use as indicator variables for host structures. Indicator structures are the only way to specify indicator parameters for host structures.

indicator vector

Another term for indicator structure.

initializing phase

The period beginning at SQL/Services startup time in which the communication server reads the configuration file and creates the process pool of execution server processes. The initializating phase ends when the communication server receives its first message request from a client application at the outset of the working phase.

See also communication server, configuration file, execution server, process pooling, SQL/Services, and working phase.

input parameter

Another term for parameter marker.

Installation Verification Procedure (IVP)

A test procedure that determines whether or not a software product has been installed correctly.

integrity

The correctness of information in an Rdb/VMS database. There are three general types of integrity control:

- Integrity constraints make sure that database information remains correct when users try to modify it incorrectly.
- Concurrency control lets only one user at a time update a file while allowing many users simultaneous access to the database.
- Recovery restores a database to the state it was in before a system failure.

interactive processing (Rdb/VMS)

A mode of computer operation in which the SQL or RDO statements and the data that control the actions of the computer are entered by a person at a terminal.

See also batch processing (Rdb/VMS).

interactive SELECT

Another term for the SELECT statement.

intermediate result table

A temporary result table created by SQL as it evaluates the clauses of a select expression. After each clause, SQL logically produces an intermediate result table that is used to evaluate the next clause.

interpretive call interface

See Callable RDO.

interval

The number of data pages between space area management (SPAM) pages.

See also SPAM page and threshold.

ISO

International Standards Organization. ISO maintains an SQL standard, ISO 9075:1989, and is developing an extended SQL standard, SQL2, in conjunction with ANSI.

IVP

See Installation Verification Procedure (IVP).

join

An operation in which Rdb/VMS retrieves data from more than one table based on matching column values. To process a join operation, Rdb/VMS first forms the Cartesian product of the tables.

See also Cartesian product, equijoin, and reflexive join.

journal file

A file that contains all records modified by a run unit or transaction. The journal file allows reconstruction of the database in the event of corruption due to system or program failures.

See also after-image journal (AIJ), before-image journal, and before-image journaling.

journaling

The process of recording, on a recoverable resource, information about operations on a database. The type of information recorded depends on the type of journal being created.

See also after-image journal (AIJ), before-image journal, and before-image journaling.

key

A column in a row that you use to locate one or more specific rows. Using keys, Rdb/VMS can locate rows in the table directly, without searching sequentially. Identifying keys increases the speed of some database operations.

See also candidate key, foreign key, hashed index, index key, key value, primary key, and sorted index.

key value

The values supplied in a data manipulation language operation to identify a specific row for access.

keyword

A word reserved for use in certain specified syntax formats, usually in a command or a statement. In Rdb/VMS syntax diagrams, keywords are shown in capital letters *and* are underlined.

License Management Facility (LMF)

A software facility that automatically checks for licenses prior to installation of Digital software.

See also Product Authorization Key (PAK) and Service Update PAK (SUP).

line index

A dynamic section of a database page that acts as a directory to data on the page by indexing page offsets of individual data segments.

See also database page.

linker

A program that creates an executable program, called an image, from one or more object modules produced by a language compiler or assembler. Programs must be linked before they can be executed.

list

A large data object with a segmented internal structure, used for storage and retrieval of unstructured data such as graphics data, large amounts of text, or long strings of binary data. The entire list appears as the value of a column. A list has the LIST OF BYTE VARYING or LIST OF VARBYTE data type. In RDO, a list is called a segmented string and has the SEGMENTED STRING data type.

See also list cursor, LIST OF BYTE VARYING data type, RDB$LENGTH, and RDB$VALUE.

list cursor

A result table defined by the SELECT expression in an SQL DECLARE LIST CURSOR statement in order to scan the individual elements of an SQL list. A list cursor enables programs to perform operations on each element of a list. Each element of a list is stored as the value of an SQL column. In order to reference elements within a row, a list cursor must reference a table cursor because the table cursor provides the row context. A list cursor is used to find the value or length of the segments of unstructured data, such as graphics data, large amounts of text, or long strings of binary data in the LIST OF BYTE VARYING data type. There are two types of list cursors: read-only and insert-only. See the sample program SQL$RESUMES for more information on how to request segment length in SQL programs.

In RDO and RDML, segmented strings are retrieved by creating a record stream that finds the contents of the value expressions RDB$LENGTH and RDB$VALUE.

See also list, LIST OF BYTE VARYING data type, positioned insert, RDB$LENGTH, RDB$VALUE, record stream, result table, and table cursor.

LIST OF BYTE VARYING data type

A special data type for the storage and retrieval of unstructured data, such as graphics data, large amounts of text, or long strings of binary data. List data type is the SQL term; the comparable RDO term is segmented string data type.

See also list cursor, RDB$LENGTH, and RDB$VALUE.

literal

A value expression that directly specifies a value. Literals can be numeric, character string, or date.

Another term for literal is constant.

LMF

See License Management Facility (LMF).

locking

A mechanism for protecting transactions from interference by other concurrently executing transactions. The mechanism that controls the allocation and deallocation of a resource, such as a record or a process. Rdb/VMS allows locks on individual rows and on all tables in one or multiple storage area files. The RESERVING clause of the SET TRANSACTION or DECLARE TRANSACTION statements can limit the extent of the locking to specific portions of the database.

See also lock type, reserving option, and share mode.

lock type

Part of the reserving option in a SET TRANSACTION, DECLARE TRANSACTION, or START_TRANSACTION statement. The reserving option, if specified, consists of a share mode and a lock type.

The possible share modes are exclusive, protected, and shared. The possible lock types are read and write. For example, a SET_TRANSACTION statement might reserve a table, specifying a reserving option of PROTECTED WRITE. Another term for lock type is access mode.

See also locking, reserving option, and share mode.

logical area

An internal partitioning of a storage area with uniform page format. Logical areas are used for the storage of tables and indexes. You can think of a logical area as a set of pages with common characteristics. There will be different logical areas within each storage area. Also, if a table is partitioned over multiple storage areas, it will have one logical area per physical storage area. This is most noticeable with the dbkey, which uses the logical area portion to denote which partition contains the record.

logical name

A user-specified name for any portion or all of a file specification. Logical name assignments are maintained in logical name tables for each process, each group, and the system.

See also the RDM$ . . . and RDMS$ . . . entries for descriptions of some logical names used by Rdb/VMS.

logical operator

Another term for Boolean operator.

logical path name

A logical name that uniquely identifies a dictionary directory, subdictionary, or object in the CDD/Plus dictionary. The logical path name is a name you define for a full or relative path name.

macro

In certain programming languages, such as C, a command, series of commands, or text string that is assigned to a key or word, similar to a DCL logical name.

main parameter

A parameter that contains the value to be used in an SQL statement.

See also indicator parameter.

map

See storage map.

MBLR

See metadata binary language representation (MBLR).

message file

A file that contains a table of message symbols and their associated text.

message protocol

The internal message-packet protocol used in SQL/Services client/server network communications.

metadata

Data that is used to describe other data. Data definitions are sometimes referred to as metadata. Examples of metadata include schema, table, and column definitions.

See also data definition language (DDL).

metadata binary language representation (MBLR)

The binary language representation of metadata manipulation. MBLR specifies the actions that the database must perform to change the metadata. In other words, the binary form of data definition language.

See also binary language representation (BLR), data definition language (DDL), and metadata.

MFDB

See multifile database (MFDB).

missing value

In RDO syntax, a special value retrieved when a column of a row in a table is null. Missing values signify in displays and to programs that there is no value stored.

SQL does not use missing values. However, you can specify default values. If you do not specify a default value, NULL is displayed in interactive SQL when it encounters null values. In programs, SQL requires the use of indicator parameters to handle the possibility of null values. Default values in SQL are not the same as missing values in RDO.

See also default value and null value.

mixed page format

A storage area format that allows rows from more than one table to reside on or near a particular page of the storage area file.

See also interval, SPAM page, threshold, and uniform page format.

module file

Another term for SQL module file.

module language

Another term for SQL module language.

module procedure

Another term for an SQL module procedure.

module processor

Another term for SQL module processor.

multifile database (MFDB)

A database in which the database root file (default file type RDB) contains only metadata and system information; all data is stored in one or more storage area files. The files in a multifile database can be on the same disk or on multiple disks. (Careful placement of files on different disk drives can reduce input/output contention and substantially improve application performance.)

See also allocation, extent, file type, hashed index, storage area, and storage map.

multisegmented index

An index with a key comprised of more than one column.

See also hashed index, index, and sorted index.

multithreaded

In SQL/Services, a characteristic of a server process that enables it to handle multiple Application Programming Interface (API) requests simultaneously. The communication server can process multiple API requests simultaneously because incoming requests can be processed before the execution of previous requests is complete.

National Character Set (NCS) utility

A utility that provides a way to define collating sequences, register them as definition modules in an NCS library, and make them locally accessible to programmers. NCS collating sequences are selective subsets of the DEC Multinational Character Set, and are frequently used in international applications. For example, a Spanish collating sequence would resolve sorting characters you might encounter when processing strings from the Spanish language.

The SQL and RDO interfaces to Rdb/VMS use the NCS collating sequences to specify collating sequences and languages.

See also collating sequence.

NCS

See National Character Set (NCS) utility.

network database

A database model that establishes relationships between records using sets. A single record can participate in any number of sets, so you can relate it to any other record in the database, not just those above and below it in a hierarchy.

Network databases are also called CODASYL databases. VAX DBMS is a network database.

See also CODASYL, database, hierarchical database, relational database, and schema.

nonexecutable statement

In host language programs, statements that SQL processes completely when it precompiles a program (DECLARE statements, INCLUDE and WHENEVER) or compiles an SQL module (DECLARE statements only). In interactive SQL, statements for which the operation controlled by the statement does not occur until SQL encounters an executable statement (DECLARE TRANSACTION and DECLARE CURSOR).

See also executable statement.

normalization

The process that reduces a database structure to its simplest form and eliminates data redundancy. Normalization physically separates related concepts in the database into separate tables or rows. A data item is stored only once and requires only one update operation to change it.

null value

The absence of a value. If a particular column of a row in a table is null, that means there is no value stored. In RDO syntax, a null value is called a missing value.

numeric data type

A characteristic assigned to a column that indicates column values are to be considered numbers rather than text.

object

A passive repository of information to be protected. In Rdb/VMS, an object is a database or schema, table, view, or column.

See also dictionary object and security.

OLTP

See online transaction processing (OLTP).

online transaction processing (OLTP)

A technique for organizing multiuser, high-volume, online applications that provides control over user access and updates of data.

See also ACMS and transaction processing.

optimizer

See query optimizer.

outer query

The top-level select expression.

See also column select expression.

outer reference

A reference in a subquery to a table specified in an outer query that contains the column select expression (or subquery).

output parameter

In dynamic SQL, another term for a select list item.

See also select list.

page header

A fixed-length section at the beginning of the database page that contains page and storage area information.

See also database page.

PAK

See Product Authorization Key (PAK) and Service Update PAK (SUP).

parameter

A variable declared in a host language program that is associated with an SQL statement. The meaning of parameter encompasses host language variables named directly in embedded SQL statements, actual and formal parameters in programs that use SQL module language, and parameter markers in the statement string of a PREPARE statement.

See also actual parameter, formal parameter, host language variable, indicator parameter, main parameter, and parameter marker.

parameter marker

A question mark (?) in the statement string of a PREPARE statement. In dynamic SQL, the question marks in the statement string of a PREPARE statement serve the same purpose as host language variables in an SQL statement embedded in a program (or as formal parameters in a SQL statement that is part of an SQL module procedure). SQL writes information about the number and data types of any parameter markers to the SQLDA when it executes a DESCRIBE . . . MARKERS statement. The program uses the information to declare host language variables corresponding to the markers and puts values in the variables that SQL uses when it dynamically executes the prepared statement.

parameter specification

Another term for parameter.

partial path name

See relative path name.

path name

In the data dictionary, a name that begins with the user's CDD/Plus anchor and ends with the given name of a dictionary directory or object. The path name includes names of intervening dictionary directories, for example:

SYS$COMMON:[CDDPLUS]PERSONNEL.DEPARTMENTS

You can have full, logical, and relative path names.

See also full path name, logical path name, and relative path name.

placeholder

Used in VAX Language-Sensitive Editor (LSE) templates, placeholders represent the SQL syntax you need to define SQL elements and data dictionary objects, and to use SQL. When you expand a placeholder, LSE provides the required syntax or indicates optional elements. You can expand a placeholder into:

- The required SQL syntax elements that are appropriate for that context
- Optional elements
- Tokens that represent appropriate keywords or information to be supplied
- Other placeholders

PLACEMENT VIA index option

A row placement option in which Rdb/VMS uses the index value in determining the database page on which to store rows. The PLACEMENT VIA index option is declared in the storage map definition.

See also hashed index, sorted index, and storage map.

plan file

Another term for context file.

platform

The combination of computer hardware and operating system software.

See also server.

pointer

A place marker that identifies a row's address in a storage area.

See also dbkey.

pointer variable

A variable that provides indirect access to storage by storing the address of data instead of directly storing data in the variable. To take full advantage of dynamic SQL, host languages must support pointer variables.

positioned insert

A special form of the SQL INSERT statement used for putting a row into a table cursor. You must use a positioned insert to prepare for storing data values in a list. When you store the row data, the positioned insert places the cursor on the table row where the list is stored, so that you can use subsequent INSERT statements to insert data into the individual list elements.

See also list cursor.

precompiled SQL

Another term for embedded SQL.

precompiler

Another term for SQL precompiler.

predicate

A condition that SQL evaluates as either true, false, or unknown. Predicates compare value expressions or result tables with different conditional operators. In SQL statements, predicates follow the WHERE or HAVING keywords. In RDO, a predicate is called a conditional expression. Other terms for predicate are Boolean expression, conditional expression, and search condition.

See also conditional operator.

prepare phase

The first phase of the two-phase commit protocol. During the prepare phase, the coordinator of the distributed transaction asks each resource manager involved in the distributed transaction whether or not it is prepared to commit the changes to the databases. If the coordinator receives yes responses from *all* the resource managers, the coordinator instructs the participants in the distributed transaction to enter the commit phase.

See also commit phase, coordinator, distributed transaction, resource manager, and two-phase commit protocol.

prepared statement

An SQL statement that has been processed with the PREPARE statement and that can be dynamically executed.

preprocesser

Another term for precompiler.

primary key

A column in a table whose value uniquely identifies its row in the table. It cannot be a null value.

See also candidate key, foreign key, and secondary key.

print list

One or more value expressions (including the names of elementary and group fields) whose values you want Rdb/VMS to display.

privilege

The ability to access a file or other resource for a certain purpose.

See also access privilege set (APS), ACL-style protection, and ANSI-style protection.

procedure

A general-purpose routine, entered by means of a call instruction, that uses an argument list passed by a calling program and only local variables for data storage. A procedure is entered from and returns control to the calling program. An example of a procedure is an SQL module procedure.

An SQL module procedure contains a series of SQL statements. These statements can be executed with the execute (@) command.

See also SQL module procedure.

procedure parameter

Another term for formal parameter.

process

An environment from which a VMS user can issue commands. These commands can be issued either in interactive or noninteractive mode. A user automatically creates an interactive process when he or she logs in to a VMS system. The user can then create subordinate processes, such as batch jobs, that execute independently of the interactive process. A user authorization file (UAF), usually maintained by a system manager, specifies characteristics that are associated with any process created by that user. Among those characteristics are device and VMS directory defaults, and a set of privileges and resource quotas that define what the process can use on the system.

process pooling

In SQL/Services, the feature that allows Application Programming Interface (API) requests on the client system to be executed on the server system by the execution server processes that comprise the process pool. The communication server starts a predefined set of execution server processes during the initializing phase at SQL/Services system startup time. The number of execution server processes created for the process pool depends on the number of definitions specified in the configuration file. Process pooling decreases API request response time and reduces overall system resource utilization.

See also Application Programming Interface (API), communication server, configuration file, and execution server.

Product Authorization Key (PAK)

A unique paper key provided to customers with the necessary information needed to register and use a layered product. Rdb/VMS requires a key for successful installation and use.

See also License Management Facility (LMF) and Service Update PAK (SUP).

project operation

See reduction operation.

qualifier

A portion of a command string that modifies a command verb or command parameter. A qualifier follows the command verb or parameter to which it applies and has the following format:

/qualifier[=option]

query header

A substitute column header that you define to replace the field name when DATATRIEVE displays values from a field. For example, you might want to define the query header "Status" to appear at the top of the column of values from the field EMPLOYEE_STATUS. You can specify a query header with the RDO CHANGE FIELD or DEFINE FIELD statement.

query name

A synonym you give to a DATATRIEVE field name in order to make input easier to type and remember. For example, to make it easier to write DATATRIEVE statements about the field SECTION_NUMBER, you can define the query name NUM and substitute it for the full field name. You can specify a query name with the RDO CHANGE FIELD or DEFINE FIELD statement.

query optimizer

The software component of Rdb/VMS that evaluates each query and determines the most efficient means of accessing the data requested.

query specification

Another term for select expression.

quiet point

A time when a run unit is not accessing any database areas. Quiet points occur between transactions.

See also run unit and transaction.

RdbExpert

A product you can use to can optimize the physical design of your Rdb/VMS database. Using RdbExpert, you specify information about the application workload, data volume, and system environment of the database. RdbExpert applies its design rules to the schema and to the information you have supplied. It generates several design reports, as well as a command procedure that (with minimal edits) you can run to create a new database with an optimal physical design. This procedure also unloads any existing data and reloads it in the new database.

Rdb/ELN

A Digital relational database management product designed for real-time applications on systems running in the VAXELN run-time application environment. Rdb/ELN uses the relational model of database organization.

Rdb/VMS

A relational database management system. The SQL interface is a component of Rdb/VMS that allows users to define and query an Rdb/VMS database using standard SQL syntax.

See also SQL.

Rdb/VMS Management Utility (RMU)

A DCL-level Rdb/VMS utility that allows database administrators to:

- Analyze how disk space is being used by the database
- Convert a previous version of the database to the current version's format
- Display the contents of Rdb/VMS database files
- Control the Rdb/VMS monitor process
- Display information about current Rdb/VMS database users and database activity statistics
- Back up and restore Rdb/VMS databases
- Truncate the after-image journal (AIJ) file and create a secondary, backup copy of the AIJ
- Load and unload database files
- Verify the integrity of Rdb/VMS databases
- Open and close an Rdb/VMS database

RDB$LENGTH

A special Rdb/VMS value expression for the length in bytes of a segment of an RDO segmented string. RDB$LENGTH can be referenced as LENGTH in the RDML precompiler.

See also list, list cursor, LIST OF BYTE VARYING data type, and RDB$VALUE.

RDB$VALUE

A special Rdb/VMS value expression for the value stored in a segment of an RDO segmented string. RDB$VALUE can be referenced as VALUE in the RDML precompiler.

See also list, list cursor, LIST OF BYTE VARYING data type, and RDB$LENGTH.

RDM$BIND_BUFFERS

A logical name Rdb/VMS recognizes that lets you specify a NUMBER OF BUFFERS to be used at run time that is different from the defined default. This can be a very powerful tool for tuning specific applications. For example:

```
$ DEFINE RDM$BIND_BUFFERS 100
```

RDM$BUGCHECK_DIR

A logical name Rdb/VMS recognizes that lets you redirect the location of bugcheck files from the SYS$LOGIN directory to another location. This can be useful when SYS$LOGIN does not have enough space for bugcheck files. For example:

```
$ DEFINE/SYSTEM/EXEC RDM$BUGCHECK_DIR DISK12:[BUGCHECK_DIR]
```

RDML

See Relational Data Manipulation Language (RDML).

RDMS$BIND_SEGMENTED_STRING_BUFFER

A logical name Rdb/VMS recognizes that may let you increase the efficiency of applications that manipulate segmented strings by increasing the buffer space for segmented strings. The default value is 10,000 bytes. For example:

```
$ DEFINE RDMS$BIND_SEGMENTED_STRING_BUFFER 20000
```

An adequate buffer size is needed to store large segmented strings (using segmented string storage maps) in storage areas other than the default RDB$SYSTEM storage area.

See also list.

RDMS$BIND_SORT_WORKFILES

A logical name that specifies how many work files the VMS Sort utility (SORT) is to use if work files are required. The default is 2 (the SORT default) and the maximum is 10.

The work files can be individually controlled by the SORTWORKn logical names (where *n* is from 0 to 9).

RDMS$BIND_WORK_FILE

A logical name Rdb/VMS recognizes that lets you redirect the location of the temporary tables that Rdb/VMS sometimes creates for use in matching operations to a disk structure other than SYS$LOGIN (the default). For example:

```
$ ! Assign the work area to another disk with read/write access:
$ DEFINE RDMS$BIND_WORK_FILE WORK$DISK:[RDB.WORK]
$ DIRECTORY/PROTECTION RDMS$BIND_WORK_FILE

Directory WORK$DISK:[RDB.WORK]

LATENT.FIL;1            (RWED,RWED,RWED,RWED)

Total of 1 file.
```

RDMS$BIND_WORK_VM

A logical name Rdb/VMS recognizes that may let you reduce the overhead of disk input/output on matching operations when temporary tables are used to perform this operation. By redefining this logical, you can specify the amount of virtual memory (VM) that will be allocated to your process for use in matching operations. Once the allocation is exhausted, additional data values will be written to a temporary file on disk (SYS$LOGIN, if RDMS$BIND_WORK_FILE is undefined). The default VM is 10,000 bytes. For example:

```
$ DEFINE RDMS$BIND_WORK_VM 20000
```

RDMS$KEEP_PREP_FILES

A logical name Rdb/VMS recognizes that lets you prevent the RDBPRE preprocessor from deleting the intermediate (file type MAR) and language files. This can be helpful when you are trying to debug an RDBPRE program and need to see the language files to do so. For example:

```
$ DEFINE RDMS$KEEP_PREP_FILES YES
```

RDMS$RUJ

A logical name Rdb/VMS recognizes that lets you locate the recovery-unit journal (RUJ) file on a different disk and directory from the default (SYS$LOGIN). This can be helpful in reducing contention in that directory. For example:

```
$ DEFINE RDMS$RUJ USERS1:[CLIENTS.JOURNAL]
```

RDO

See Relational Database Operator (RDO).

RDO command procedure

A sequence of Rdb/VMS statements stored in a text file; sometimes referred to as an RDO procedure.

See also DCL command procedure and SQL command procedure.

record

Another term for row.

record locking

See row locking.

Record Management Services (RMS)

A set of VMS operating system procedures that programs can call to process files and records within files. RMS lets programs issue GET and PUT requests at the record level (record input/output) as well as read and write blocks (block input/output). RMS is an integral part of the VMS system software and is used by high-level languages, such as VAX COBOL and VAX BASIC, to implement their input and output statements.

record selection expression (RSE)

In RDO and RDML, a set of conditions that individual records (rows) of a relation (table) must meet before being included in a record stream (result table).

See also column select expression, record stream, result table, select expression, and select operation.

record stream

In RDO and RDML, a temporary group of related records (rows), formed by a record selection expression (RSE), from a relation (table). Streams are used in an application program or with RDO to retrieve one record at a time for manipulation. In SQL terminology, streams correspond to result tables that consist of entire rows from their source tables. Another term for record stream is stream.

See also record selection expression (RSE).

recovery

The process of restoring a database to a known condition after a system or program failure.

See also after-image journal (AIJ), before-image journal, idempotent, journal file, journaling, and transaction.

recovery-unit journal (RUJ)

See before-image journal.

reduction operation

In RDO, an operation that finds the unique values for a field or group of fields and eliminates repeated records. You use the REDUCED TO clause to perform the operation. The REDUCED TO clause is the RDO equivalent to the SQL DISTINCT clause. Another term for reduction operation is project operation.

referential integrity

A database state where every foreign key value matches some value for its associated primary key.

See also table-specific constraint and trigger.

reflexive join

An operation that joins a relation to itself.

See also Cartesian product.

relation

Another term for table. The term *relation* is used in RDO and RDML documentation.

relation-specific constraint

See table-specific constraint.

relational database

A database model that represents data as a set of independent tables. Within a table, data is organized in columns and rows, with no more than one data item occupying each intersection. Relationships between tables depend on values within the tables. In RDO and RDML documentation, tables are called relations, columns are called fields, and rows are called records.

See also database, hierarchical database, network database, and schema.

Relational Database Operator (RDO)

An interactive utility for maintaining Rdb/VMS databases, creating and modifying definitions of database elements, and storing and manipulating data. The RDO utility can also be used to access a VIDA database and to perform Data Distributor operations. The SQL interface to Rdb/VMS provides largely parallel features to RDO.

See also Callable RDO.

Relational Data Manipulation Language (RDML)

A data manipulation language for VAX C and VAX Pascal programs that access Rdb/VMS and Rdb/ELN databases. Programs written in RDML are processed by the RDML preprocessor, which converts the RDML statements into a series of calls to the database. Following successful preprocessing, the programmer can submit the resulting source code to the host language compiler. The SQL interface to Rdb/VMS provides largely parallel features to RDML.

relational join operation

See join.

relational operator

Another term for conditional operator.

relative path name

The shortened form of a dictionary path name. It includes only the parts of the path name that follow the default CDD/Plus directory name. You can use either the full path name or the relative path name to refer to directories, subdictionaries, and objects in a CDD/Plus dictionary.

remote server

The part of Rdb/VMS that lets you access data on other computers on your DECnet network. If, for example, you are using the computer VACKS1 and you type DECLARE SCHEMA FILENAME "TRIXIE::DISK1:[TOP] PERSONNEL", Rdb/VMS instructs the remote server to log in to an account on TRIXIE, which then processes your statements.

replication

In Data Distributor, the process by which a copy of a complete or partial database is first transferred from a source database on the source node to a target database either on the source node or on a remote node. Then, the database in the target directory is periodically updated when any changes are made to that portion of the source database that was originally copied to the target database. The source replications database must be an Rdb/VMS database. Note that the target databases for replications are periodically updated, while those for extractions are not.

See also Data Distributor, extraction, source database, source node, target database, and target node.

request

An optional clause of certain data manipulation language (DML) statements. The request clause lets you specify the transaction or request to be affected by the DML statement. A request can include a transaction handle, a request handle, or both.

See also data manipulation language (DML), request handle, and transaction handle.

request handle

A variable that uniquely identifies a request.

reserving option

The sharing and locking characteristics that Rdb/VMS applies to row access operations during a transaction. The reserving option consists of a share mode (exclusive, protected, or shared) and a lock type (read or write). The reserving option is specified or defaulted when the transaction is started. Another term for share mode is allow mode. Another term for lock type is access mode.

See also lock type and share mode.

resource manager

A component of a distributed transaction. The resource manager is a database management system, such as Rdb/VMS. The resource manager is responsible for maintaining and recovering its own resources. From the perspective of the application, the resource manager is a single attachment to an Rdb/VMS database. In addition, other database products that support the two-phase commit protocol can be resource managers.

See also coordinator, DECdtm services, distributed transaction, transaction manager, and two-phase commit protocol.

restore operation

An operation that rebuilds a database from a saved backup file or copy. Usually, you restore a saved backup file or copy after a hardware or software failure.

There are two types of restore operations in Rdb/VMS:

- RMU/RESTORE

  Use the RMU/RESTORE command from DCL level to restore a database backup file (default file type RBF) created by the RMU/BACKUP command.

- IMPORT

  Use the IMPORT statement in SQL to migrate a database from one database management system to another, or to restructure a database.

  Migration changes include taking an existing Rdb/VMS database, backing it up, and then using the IMPORT statement to convert the internal data structures so that they are compatible with a newly installed version of Rdb/VMS. Restructuring changes include taking a single-file database, backing it up using the EXPORT statement, and then using the IMPORT statement to define one or more storage areas in a new multifile database.

See also backup operation, export operation, and import operation.

restrict

See select operation.

result table

A temporary table of values derived from columns and rows of one or more tables that meet conditions specified by a select expression.

RMS

See Record Management Services (RMS).

RMU

See Rdb/VMS Management Utility (RMU).

roll back

To cancel any changes to the database made during the current transaction. Use the ROLLBACK statement to roll back changes.

See also commit and transaction.

rollforward

The process of using an after-image journal to restore a database to a known state. This process replaces updates to the database that were lost because a system, program, or disk failure required the installation of backup media.

See also recovery.

root file

See database root file.

row

The horizontal dimension of a table composed of a set of columns that contain one data item each.

row locking

A process by which a database management system reserves a row or set of rows for use by one user. This process can limit or prevent access to those rows by other users. Row locking helps guarantee the consistency of data.

RSE

See record selection expression (RSE).

RUJ

Recovery-unit journal.

See before-image journal.

run unit

An execution of a single program that accesses a database.

See also quiet point and transaction.

scalar expression

Another term for value expression.

scale factor

The power of 10 to multiply by when storing a value with a fixed number of decimal places in an exact numeric data type. For example, the scale factor for a dollar amount such as $999.99 is two.

schedule definition

In Data Distributor, a definition that specifies when a particular transfer definition is executed by the transfer monitor. You can also define the frequency of transfer if you need to transfer records or updates to existing records on, for example, a daily or weekly basis. You create schedule definitions using either the CREATE SCHEDULE statement of interactive SQL or the DEFINE SCHEDULE statement of interactive RDO.

See also Data Distributor, Relational Database Operator (RDO), transfer database, transfer definition, and transfer monitor.

schema

The data definitions that comprise a database created with SQL data definition statements. The SQL CREATE SCHEMA statement lets you specify in a single SQL statement all data and privilege definitions for a new schema (you can also add definitions to the schema later).

See also database, hierarchical database, network database, and relational database.

schema element

Any CREATE statement or a GRANT statement embedded within a CREATE SCHEMA statement.

schema name

A name that users or programs supply to identify a schema and the database system files associated with it.

The schema name is a VMS file specification or a data dictionary path name.

scientific notation

A way of expressing a very large or very small number as a constant multiplied by the appropriate power of 10. For example:

```
.000000009     .9E-8 (9 times 10 to the power of -8)
9000000.       .9E 7 (9 times 10 to the power of 7)
```

scope

The scope of a transaction refers to the statements beginning with the executable statement that starts the transaction and ending with the COMMIT or ROLLBACK statement that completes the transaction.

The scope of a dbkey refers to how long the database system guarantees that a particular row's dbkey will point to that row only and not be reused even if the row is deleted. Scope is also used as a standard term from programming languages that refers to the part of a program where particular declarations can be seen.

See also dbkey and transaction.

search condition

Another term for predicate.

secondary key

Any key other than the primary key.

See also candidate key, foreign key, and primary key.

security

The protection of the information stored in a database against unauthorized reading, writing, or deleting.

See also access privilege set (APS), access control list (ACL), ACL-style protection, ANSI-style protection, object, privilege, and subject.

security alarm

One-time notifications of Rdb/VMS security events that are sent to all terminals enabled as security operators. Security alarms are triggered by the occurrence of an event previously designated as worthy of the alarm because of its security implications.

See also security audit, security event, and security operator terminal.

security audit

In Rdb/VMS, the record or recorded history of Rdb/VMS security events that are written to the security audit journal file. The writing of security audits are triggered by the occurrence of an event previously designated as worthy of the audit because of its security implications. Security audit is also referred to as audit trail.

See also security event and security audit journal.

security audit journal

The record of audit activities (security events) that is created by Rdb/VMS when the audit feature is enabled for the database by the RMU/SET AUDIT command. The security audit journal is sometimes referred to as the audit journal, audit trail, or security journal.

See also security alarm, security audit, and security event.

security event

Some operation (in SQL, RDO, or RMU) that affects the security of the objects in the database. A security event is sometimes referred to as an event or as an audit event.

See also security audit and security audit journal.

security manager

The person or persons responsible for protecting the security of the database. This role is sometimes performed by the same person who functions as the database administrator. It requires the same skills as the DBA, but includes additional privilege (the SECURITY privilege) as well as knowledge of the security features provided with an Rdb/VMS database.

See also privilege and security.

security operator terminal

A class of terminal that has been enabled to receive security alarms. Normally, such a terminal is a hardcopy terminal in a protected room. The output provides a log of security-related events and details that identify the source of the event.

See also security alarm and security event.

segmented string

The RDO term for a list.

See also list, RDB$LENGTH, and RDB$VALUE.

segmented string data type

A special RDO data type designed to handle large pieces of unstructured data, such as graphics data, large amounts of text, or long strings of binary data, with a segmented internal structure. The maximum size of a string segment is 64K bytes.

See also list, RDB$LENGTH, and RDB$VALUE.

segmented string identifier

A number that points to the location of the segmented string or list. This number, rather than the contents of the list, is stored in the table column when you create a table that contains a list. In SQL, you must use cursors to retrieve or store the contents of a list. In RDO and RDML, you must use a FOR loop to retrieve or store the values for RDB$LENGTH and RDB$VALUE.

See also list.

select expression

The fundamental element in SQL syntax that is the basis of the SELECT, DECLARE CURSOR, CREATE VIEW, and INSERT statements. Select expressions specify a result table derived from some combination of the tables or views identified in the FROM clause of the expression.

select list

A list of column names and value expressions in a select expression. The select list specifies the columns in the result table for the select expression.

select operation

An operation that chooses from tables those rows that satisfy a conditional expression. For example, if you want to display employee names with salaries greater than $20,000, a selection operation eliminates employee rows with salaries less than or equal to $20,000 from the output.

See also record selection expression (RSE).

SELECT statement

A select expression with an optional ORDER BY clause.

sequential file

An RMS file whose records appear in the order in which they were originally written. A sequential file does not have an index.

server

A service provider. In SQL/Services, the server is a process present on all VMS systems running Rdb/VMS V3.1 (or higher) that provides access to databases with application programs running on various platforms.

See also Application Programming Interface (API), client, dynamic SQL, platform, and SQL/Services.

Service Update PAK (SUP)

A PAK for customers who already have service contracts with Digital.

See also License Management Facility (LMF) and Product Authorization Key (PAK).

shadowing

For table rows, a storage strategy to optimize input/output operations that is used when a parent table has a much smaller cardinality than the related child table. This storage strategy uses a hashed index placement index to place parent rows with its hash buckets in one storage area along with the hash buckets of the second table. When the rows of the child table are placed in the second storage area, they are clustered (shadowed) in the vicinity of where the hash buckets in the first storage area are located. When sufficient space is allocated for storing the child rows and the child rows are stored in a uniform storage area, then free space will be available to maintain the clustering effect of the child rows and result in a minimum of two input/output operations to retrieve a child row. Shadowing serves to physically group rows together to minimize input/output operations and optimize query strategies.

See also hash bucket.

share mode

The degree of sharing that Rdb/VMS will permit when another user attempts to access a table that you have reserved for a transaction. The share mode is part of the reserving option, which is specified or defaulted for each table at the start of the transaction. The possible share modes are exclusive, protected, and shared. Another term for share mode is allow mode.

See also lock type and reserving option.

single-file database

Databases that have a combined root and storage area file. By default, a CREATE SCHEMA statement creates a single-file database. The absence of a CREATE STORAGE AREA clause in a CREATE SCHEMA statement is what makes the created database single file.

singleton select

A SELECT statement that must specify a result table no larger than one row. A singleton select includes an additional clause, INTO, to assign the values in the row to host language variables in a program. SQL allows singleton select statements only within a host language program. Another term for singleton select is embedded select.

snapshot

A consistent view of the data within the database at a selected time. Initiated in an Rdb/VMS database when a user starts a read-only transaction.

See also deferred snapshots.

sort key

The columns or value expressions in an ORDER BY clause according to which SQL sorts an intermediate result table. The first column or value expression in the ORDER BY clause is the major sort key. Subsequent columns or value expressions are minor sort keys. SQL groups all rows with the same value for a sort key together.

See also ascending order and descending order.

sorted index

A tree structure (B-tree) of nodes that the database system navigates by reading nodes on progressively lower-level branches of the tree until it finds the record that contains the location (or dbkey) of a particular row.

See also B-tree, dbkey, hashed index, index, key, key value, and multisegmented index.

SORTWORKn (where n = 0 to 9)

Logical names VMS recognizes that let you increase the efficiency of Rdb/VMS sort operations, which use the VMS Sort utility (SORT), by assigning the location of the temporary sort work files to different disks. These assignments are made by using the logical names SORTWORK0, SORTWORK1, and so forth. Normally, SORT places work files in the user's SYS$SCRATCH directory. By default, SYS$SCRATCH is the same device and directory as the SYS$LOGIN location.

Example:

```
$ DEFINE SORTWORK0 $222$DUA10:[SORT.TEMP]
$ DEFINE SORTWORK1 $222$DUA11:[SORT.TEMP]
```

source database

In Data Distributor, a single, centralized database that is intended to support the information requirements of users throughout a network.

See also Data Distributor, extraction, replication, source node, target database, and target node.

source node

In Data Distributor and Rdb/VMS, the node in a network on which the source database resides.

See also Data Distributor, extraction, replication, source database, and target node.

space area management page

See SPAM page.

SPAM page

A special database page (common to both uniform page format and mixed page format storage areas) that stores the fullness thresholds of each data page in the storage area, for a specified interval of pages. The space area management (SPAM) page and also contains a list of clumps and the logical areas to which the clumps belong.

See also clump, interval, mixed page format, threshold, and uniform page format.

spooling

The way in which output to slow devices is placed into queues on faster devices to wait for transmission to the slower devices.

SQL$DATABASE

The logical name SQL uses to declare the default schema. If a user or program does not issue an explicit DECLARE SCHEMA statement, SQL attempts to declare a default schema using the file specification assigned to SQL$DATABASE.

SQL

Structured query language. A data definition and data manipulation language for relational databases. Variations of SQL are offered by most major vendors for their relational database products. The SQL interface to Rdb/VMS provides a user interface to other database products. ANSI Standard X3.135-1989 specifies the syntax and semantics approved for SQL by the American National Standards Institute.

SQL2

An extension to the SQL standard under development by the American National Standards Institute and the International Standards Organization.

SQLCA

SQL Communications Area. A host structure SQL uses to provide information about the execution of SQL statements to application programs. The SQLCA shows whether or not a statement was successful, and for some errors, the particular error when a statement is not successful.

SQLCODE

An integer parameter whose value indicates the error status returned by the most recently executed SQL statement. A positive value indicates a warning, a negative value an error, and zero indicates successful execution. The SQLCA contains an SQLCODE field.

SQLDA

SQL Descriptor Area. A collection of host language variables used only in dynamic SQL programs. The SQLDA provides information about dynamic SQL statements (the number and data type of a prepared statement's parameter markers and select list items) to the program, and information about memory allocated by the program (the addresses of host language variables that correspond to parameter markers and select list items) to SQL.

See also descriptor.

SQL command procedure

A sequence of SQL statements stored in a text file; sometimes referred to as an SQL procedure.

See also DCL command procedure and RDO command procedure.

SQL module file

The file containing SQL module language elements to be compiled by the SQL module processor. An SQL module file includes one or more procedures that contain an SQL statement. The procedure can be called from a host language program.

SQL module language

Special keywords and syntax that allow procedures that contain SQL statements to be called from host languages that are not supported by the SQL precompiler.

SQL module procedure

A construct within an SQL module file named in host language calls to the module. SQL module procedures contain formal parameter declarations and an SQL statement that executes when the procedure is called by a host language program.

SQL module processor

The component of SQL that compiles SQL module files. The SQL module processor produces object (OBJ) files that can be linked with object files produced by host language compilers.

SQL precompiler

The SQL utility that converts embedded SQL statements in host language programs to a form understandable by the host language compiler. The precompiler also invokes the host language compiler and creates an intermediate object file ready for linking.

SQL/Services

A client/server component of Rdb/VMS that allows application programs running on various types of computers to access databases that use the dynamic interface to SQL.

See also client, dynamic SQL, server, and SQL.

static SQL

Precompiled SQL statements that do not use dynamic SQL.

statistical expression

Another term for function.

storage area

A subdivision of the database, named in the CREATE STORAGE AREA clause within a CREATE SCHEMA statement. This clause also determines the physical characteristics of the storage area or areas. Subsequent CREATE STORAGE MAP statements associate the storage area with particular tables in the schema. Only multifile Rdb/VMS databases contain storage areas as physical VMS files separate from the database root file (default file type RDB). Each storage area has a default file type of RDA, and each storage area has a corresponding snapshot file with a default file type of SNP.

See also allocation, extent, multifile database (MFDB), and storage map.

storage map

A definition that associates a table with one or more storage areas in a multifile database. The CREATE STORAGE MAP statement specifies a storage map that controls which rows of a table are stored in which storage areas.

When you create a storage map, you specify the table name and the storage area or areas in which rows from the table are to be stored; you can also specify whether or not an index will be used to choose the location for storing the row and whether or not data compression will be enabled when rows are stored in the table.

See also multifile database (MFDB) and storage area.

stream

Another term for record stream.

structure

In the *VAX Rdb/VMS Guide to Using SQL* and *VAX Rdb/VMS SQL Reference Manual*, refers to a parameter that contains other fields. (Unlike the case of repeating items that make up one dimension in an array, the basic elements in a structure do not need to be identically defined.) In some host languages, the term *record* and the term *structure* refer to the same declaration. In other languages, such as FORTRAN, you declare structures and records separately. If the language you are using declares records separately from structures, you specify record names in precompiled SQL statements. In this case, the fields listed or implied by the record declaration constitute the structure to SQL.

subdirectory

A list of files that is grouped one or more levels below the top-level or main VMS directory.

subject

An active entity that gains access to information on behalf of people. In Rdb/VMS, a subject is an attachment to the database. An attachment can be either an interactive user or an application program.

See also object and security.

subpool

In SQL/Services, a set of either generic or database execution server processes. Multiple subpools comprise the process pool. Definitions in the configuration file specify the characteristics of a subpool. The communication server creates one subpool for every definition in the configuration file.

See also communication server, configuration file, database execution server, execution server, and generic execution server.

subquery

Another term for column select expression.

subselect

Another term for select expression.

substring

A value expression made up of a character string that has a specified start position and an optionally specified length. The characters in the substring are numbered beginning with the number 1.

See also value expression.

system manager

A VMS user responsible for the overall operation of an ACMS or VMS system. Duties of the system manager include authorizing all users of the system, setting access requirements for all system resources, and running all procedures necessary to ensure the correct and timely operation of the system.

system relation

A relation (table) that holds information (such as table or column names) needed for the operation of a database management system.

Systems Network Architecture (SNA)

A description of network structure, configuration and operation control, and protocols (rules) developed by IBM.

table

A set of data elements that has a horizontal dimension (row) and a vertical dimension (column) in a relational database system. A table has a specified number of columns but can have any number of rows. Rows stored in a table are structurally equivalent to records from sequential files in that they must not contain repeating columns. Another term for table is relation.

See also cardinality.

table cursor

A result table defined by the SELECT expression in an SQL DECLARE CURSOR statement in order to scan the individual rows of a table. A table cursor enables programs to perform operations on each row of a table. It can also be positioned on a row that contains a list, so that a list cursor can access the individual elements of that list. There are three types of table cursors: read-only, insert-only, and update.

See also cursor, list cursor, and result table.

table-specific constraint

A constraint that indicates that if a certain update action is taken on one table, appropriate update actions will be taken on other specified tables as well. Another term for table-specific constraint is relation-specific constraint.

See also cascading update, referential integrity, and trigger.

target database

In Data Distributor, a database into which Data Distributor copies data. A target database can be created by an extraction or replication. You can create a target database on the same node as the source database or on a node at a remote site. The target database is referred to as either an extraction database or replication database, depending on the process that created it.

See also Data Distributor, extraction, replication, source database, source node, and target node.

target node

In Data Distributor, the receiving node for data transferred from the source database on the source node. A copy of a database can be transferred to a directory on the source node or to a directory on a remote node.

See also Data Distributor, extraction, replication, source database, source node, and target database.

TDMS

A Digital product that uses forms to collect and display information on the terminal. TDMS stands for Terminal Data Management System. TDMS provides data independence by allowing data used in an application to be separated from the application program. ACMS multiple-step tasks use TDMS services to manage terminal input and output.

Terminal Data Management System (TDMS)

See TDMS.

threshold

A value that indicates the fullness of a data page. Rdb/VMS checks the space area management (SPAM) pages to determine the fullness thresholds of data pages.

See also interval and SPAM page.

TID

See distributed transaction identifier (TID).

TP

See transaction processing.

transaction

A group of statements whose changes can be made permanent or undone only as a unit. A transaction ends with a COMMIT or ROLLBACK statement. If it ends with a COMMIT statement, all the changes made to the database by the statements are made permanent. If the transaction ends with ROLLBACK, none of the statements takes effect.

See also commit, quiet point, recovery, and roll back.

transaction handle

A host language variable that specifies the name of a transaction so that it can be uniquely identified.

transaction manager

A component of a distributed transaction. The transaction manager coordinates the actions of the resource managers that are located on the same node as the transaction manager. DECdtm services is a transaction manager.

See also coordinator, DECdtm services, distributed transaction, resource manager, and two-phase commit protocol.

transaction processing

An environment that addresses large, corporate-level applications that support many users for critical business functions. In an OLTP application, there are usually many users simultaneously performing predefined (query and update) functions to a collection of shared data, generally a database. Results are usually expected immediately (real time). In contrast to standard transaction processing, when an OLTP transaction is completed, all data is fully updated during the request and output stages. OLTP applications can be used for tasks such as order processing, banking, accounting, or inventory control.

See also ACMS and online transaction processing (OLTP).

transfer database

In Data Distributor, the storage location of all transfer definitions, schedule definitions, and record selection criteria. The transfer database is created by the transfer monitor.

See also Data Distributor, schedule definition, transfer definition, and transfer monitor.

transfer definition

In Data Distributor, a definition that uses record selection expressions to identify subsets of the source database that the transfer monitor can copy to one or more target databases. You create transfer definitions with SQL or RDO.

See also Data Distributor, record selection expression (RSE), source database, target database, and transfer monitor.

transfer monitor

The program that oversees all of the functions provided by Data Distributor. In general, the transfer monitor regulates the execution of data transfers from all source databases to target databases (directories) that are located either on the same node as the source database or on remote nodes.

See also Data Distributor, extraction, replication, and transfer definition.

trigger

A statement that causes certain updating actions to be performed before or after an update of a table in order to maintain the referential integrity of the database. For example, a trigger would specify that when an employee's record is deleted from the EMPLOYEES table, that same employee's record will be automatically deleted from other specified tables within the database.

See also cascading update, table-specific constraint, and referential integrity.

truth table

A method of showing how predicates combined with the Boolean operators AND, OR, and NOT, are evaluated.

tuple

Relational database terminology for a record or row. In SQL, the term row is used. In RDO and RDML, the term record is used.

two-phase commit protocol

A protocol provided by DECdtm services and the VMS utility. The two-phase commit protocol coordinates the activity of components of a distributed transaction to ensure that every required operation is completed before the distributed transaction is made permanent.

See also commit phase, coordinator, DECdtm services, distributed transaction, prepare phase, resource manager, and transaction manager.

UIC

See user identification code (UIC).

uniform page format

The default storage area page format. A format that consists of groups of *n* pages, called clumps, where *n* equals the buffer size divided by the page size. A set of clumps forms a logical area.

See also clump, logical area, and mixed page format.

UNION operation

The merger of the values of columns in one table with the values of columns in other tables.

unit of recovery

Another term for transaction.

usage mode

See reserving option.

user identification code (UIC)

A code that uniquely identifies a VMS user by a group number or name and a member number or name. The UIC is enclosed in brackets ([ ]). A UIC can be in numeric or alphanumeric format.

user name

A designation assigned to a VMS user to identify that user. Also, the name a terminal user types to log in to VMS and ACMS, or for VIDA, to log in to an IBM system.

validation

The process of checking data on entry to ensure that it meets previously established requirements.

value expression

A symbol or string of symbols used to calculate a value. If you use a value expression in a statement, SQL or RDO calculates the value associated with the expression and uses that value when executing the statement. Types of value expressions include literals, arithmetic expressions, functions, and column select expressions.

variable

A symbol declared in a program that stands for a memory storage location. A variable has a name and a data type.

VAXcluster

A highly integrated organization of VMS systems that communicate over a high-speed communications path. VAXcluster systems have all the functions of single-node VMS systems, plus the ability to share CPU resources, queues, and disk storage. Like a single-node system, the VAXcluster organization provides a single security and management environment. Member nodes can share the same operating environment or serve specialized needs.

vector

A one-dimensional array in a host language program. SQL can refer to vectors only as indicator structures.

VIDA

A database and internetworking facility that gives VMS users read-only access to data managed by an IBM database system.

view

A logical table whose data is not physically stored. Views refer to rows stored in other tables. You define a view to:

- Access a subset of the columns stored in a row
- Access a set of columns stored in different rows
- Avoid creating a redundant copy of data that is already stored

VMS

Virtual Memory System. The major operating system for Digital VAX processors.

VMS user

A person or account authorized by a VMS system manager to access a VMS system. The user is assigned a user name, a password, a user identification code (UIC), a default VMS directory, a default command language, quotas, limits, and privileges.

wildcard character

A symbol, such as the asterisk (*) or percent sign (%), that you use in place of all or part of a file specification.

working phase

The period beginning when the communication server receives its first message request from the client system and ending when the SQL/Services system fails or is manually shut down. During the working phase, the communication server accepts incoming client message requests, dispatches them to execution server processes for processing, receives in return execution results, and sends results back to the waiting client application.

See also client, communication server, execution server, initializing phase, and SQL/Services.

A
Rdb/VMS Master Index

The Rdb/VMS master index uses the abbreviations in Table A–1 for the manuals in the Rdb/VMS documentation set.

Table A–1 Abbreviations for the Manuals Included in the Master Index

Abbreviation	Manual Title
INTRO	VAX Rdb/VMS Introduction and Master Index
GUSQL	VAX Rdb/VMS Guide to Using SQL
DESIGN	VAX Rdb/VMS Guide to Database Design and Definition
MAINT	VAX Rdb/VMS Guide to Database Maintenance and Performance
GURRR	VAX Rdb/VMS Guide to Using RDO, RDBPRE, and RDML
SQLSRV	VAX Rdb/VMS Guide to Using SQL/Services
DIST_TRANS	VAX Rdb/VMS Guide to Distributed Transactions
TUNING	VAX Rdb/VMS Guide to Database Tuning
SQLRM	VAX Rdb/VMS SQL Reference Manual
RDORM	VAX Rdb/VMS RDO and RMU Reference Manual
RDMLRM	RDML Reference Manual

Index

How to Order Additional Documentation

Technical Support

If you need help deciding which documentation best meets your needs, call 800-343-4040 before placing your electronic, telephone, or direct mail order.

Electronic Orders

To place an order at the Electronic Store, dial 800-DEC-DEMO (800-332-3366) using a 1200- or 2400-baud modem. If you need assistance using the Electronic Store, call 800-DIGITAL (800-344-4825).

Telephone and Direct Mail Orders

Your Location	Call	Contact
Continental USA, Alaska, or Hawaii	800-DIGITAL	Digital Equipment Corporation P.O. Box CS2008 Nashua, New Hampshire 03061
Puerto Rico	809-754-7575	Local Digital subsidiary
Canada	800-267-6215	Digital Equipment of Canada Attn: DECdirect Operations KAO2/2 P.O. Box 13000 100 Herzberg Road Kanata, Ontario, Canada K2K 2A6
International	————	Local Digital subsidiary or approved distributor
Internal[1]	————	USASSB Order Processing - WMO/E15 *or* U.S. Area Software Supply Business Digital Equipment Corporation Westminster, Massachusetts 01473

[1]For internal orders, you must submit an Internal Software Order Form (EN-01740-07).

Reader's Comments

VAX Rdb/VMS
Introduction and Master Index
AA–KY66C–TE

Please use this postage-paid form to comment on this manual. If you require a written reply to a software problem and are eligible to receive one under Software Performance Report (SPR) service, submit your comments on an SPR form.

Thank you for your assistance.

I rate this manual's:	Excellent	Good	Fair	Poor
Accuracy (software works as manual says)	☐	☐	☐	☐
Completeness (enough information)	☐	☐	☐	☐
Clarity (easy to understand)	☐	☐	☐	☐
Organization (structure of subject matter)	☐	☐	☐	☐
Figures (useful)	☐	☐	☐	☐
Examples (useful)	☐	☐	☐	☐
Index (ability to find topic)	☐	☐	☐	☐
Page layout (easy to find information)	☐	☐	☐	☐

I would like to see more/less ______________________________

____

What I like best about this manual is ______________________________

____

What I like least about this manual is ______________________________

____

I found the following errors in this manual:
Page Description

______ ______________________________

______ ______________________________

______ ______________________________

Additional comments or suggestions to improve this manual:

____

____

____

I am using **Version** ______ of the software this manual describes.
Name/Title ______________________ Dept. ____________

Company ______________________ Date ____________

Mailing Address ______________________________

______________________ Phone ____________

Do Not Tear – Fold Here and Tape

digital

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 33 MAYNARD MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

DIGITAL EQUIPMENT CORPORATION
CORPORATE USER PUBLICATIONS
MRO1–3/L12
P.O. BOX 1001
MARLBOROUGH, MA 01752–9840

Do Not Tear – Fold Here and Tape