



Mimer SQL

VMS Guide

Version 8.2

Copyright © 2000 Mimer Information Technology AB

Mimer SQL version 8.2 VMS Guide

December, 2000

Copyright © 2000 Mimer Information Technology AB.

Published by Mimer Information Technology AB,
P.O.Box 1713,
SE-751 47 Uppsala, Sweden.
Tel +46(0)18-18 50 00.
Fax +46(0)18-18 51 00.
Internet: <http://www.mimer.com>

Produced by Mimer Information Technology AB, Uppsala, Sweden.

All rights reserved under international copyright conventions.

The contents of this manual may be printed in limited quantities for use at a Mimer SQL installation site. No parts of the manual may be reproduced for sale to a third party.

CONTENTS

1	INTRODUCTION	
1.1	Document objectives.....	1-1
1.2	Terms, definitions and trademarks.....	1-1
2	MIMER SQL INSTALLATION	
2.1	VMS system requirements.....	2-1
2.2	Unpacking the Mimer SQL distribution file.....	2-2
2.2.1	Unpacking a self-extracting VMS ZIP file.....	2-2
2.2.2	Unpacking a VMS saveset from a tape.....	2-3
2.3	The MIMSETUP8 command.....	2-3
2.3.1	MIMSETUP8 syntax.....	2-3
2.3.2	Logical names defined by MIMSETUP8.....	2-5
2.4	Installing the Mimer SQL license key.....	2-5
2.5	Establishing a Mimer SQL database.....	2-6
2.6	Removing a Mimer SQL installation.....	2-6
3	MIMER SQL INSTALLATION COMPONENTS	
3.1	Distributed files.....	3-1
3.1.1	Root directory files.....	3-1
3.1.2	Documentation files (MIMDOC8).....	3-2
3.1.3	Example files (MIMEXAMPLES8).....	3-2
3.1.4	Executable programs (MIMEXE8).....	3-3
3.1.5	Library files (MIMLIB8).....	3-3
3.2	Shared images.....	3-4
4	RUNNING MIMER SQL APPLICATIONS	
4.1	Executing Mimer SQL applications.....	4-1
4.1.1	Using the DCL command RUN.....	4-1
4.1.2	Using the DCL\$PATH logical name.....	4-2
4.1.3	Using VMS command definitions.....	4-2
4.2	Selecting a Mimer SQL installation.....	4-3
5	MANAGING A DATABASE SERVER	
5.1	The MIMCONTROL command.....	5-1
5.1.1	Privileges needed for MIMCONTROL execution.....	5-1
5.1.2	Prerequisites for database server startup.....	5-2
5.2	Automatic database start and stop.....	5-2
5.3	The MIMTCP server.....	5-3
5.3.1	Manually starting a MIMTCP server.....	5-3
6	USING THE MIMER JDBC DRIVER ON VMS	
6.1	Example of using the JDBC driver.....	6-1
A	DATA TYPES USED IN MIMER SQL	
A.1	Compiling applications using floating point data types.....	A-1
A.2	Internal Mimer SQL representation.....	A-1
A.3	External data types supported by Mimer SQL.....	A-2

B	USING MIMER7 APPLICATIONS WITH MIMER8	
B.1	Re-link applications	B-1
B.2	Remap shareable libraries	B-1
B.3	Client-server access	B-2
B.4	Local client/server access	B-2

1 INTRODUCTION

1.1 Document objectives

This publication describes the installation and usage of version 8.2 of the Mimer SQL relational database server under VMS. It gives VMS-specific instructions about installing the software, creating databases and managing database servers.

A working knowledge of system management within the VMS environment is required for the installation of Mimer SQL. Relevant information may be found in the *VMS System Management Guide* (published by Compaq).

Note: Most VMS manuals can be found online at the following Internet address: <http://www.openvms.compaq.com:8000/>.

General familiarity with the concepts and facilities provided by the Mimer SQL system is an advantage.

Other documents that are referred to in this document or that may be of interest when dealing with the tasks described here are: the *Mimer SQL System Management Handbook*, the *Mimer SQL Programmer's Manual*, and the *Mimer SQL Release Notes*.

1.2 Terms, definitions and trademarks

The following terms, trademarks and acronyms are used in this document:

API	Application Programming Interface.
BSQL	Batch SQL, a program used for execution of SQL statements which are read from a command file or entered interactively.
CLD	A CLD (Command Language Definition) file contains definitions for new DCL commands.
Data source	ODBC term for a database.

Databank	“Databank” is the Mimer SQL term for the physical file in which one or more Mimer SQL tables are stored. A databank corresponds to one file in the operating system. A database may contain several databanks.
Database	A database is a collection of databanks, tables, shadows, etc., all defined as objects in the data dictionary. A computer may have several databases operating simultaneously, but no information is shared between them . Each database has a unique name, registered in the SQLHOSTS file.
Database home directory	The directory where the SYSDB databank file is located, also recorded in the SQLHOSTS file.
DCL	Digital Command Language.
DCL\$PATH	A logical name used by DCL to find executable programs. By including the MIMEXE8 directory in the definition of this logical name, all Mimer SQL programs can be started by typing their names directly. Some of the programs also accept UNIX-style command options when started in this fashion.
Dynamic SQL	SQL statements constructed at runtime and passed to the database management system for execution.
Embedded SQL	The term used for SQL statements when they are embedded in a traditional host language.
ESQL	The preprocessor for embedded SQL.
MIMER8	The general term used for version 8 of Mimer SQL.
MIMERxxxx	Symbolic name for the distribution directory tree, unique for each Mimer SQL release, where “xxxxx” stands for the current version number, e.g. “822A”.
ODBC	Microsoft’s Open Database Connectivity, a specification for a database API in the C language, independent of any specific DBMS or operating system.
PSM	Persistent Stored Modules, the term used by ISO/ANSI for Stored Procedures.
Shadow	A Mimer SQL databank may have one or more shadows. If the databank is shadowed, there will be one file for each shadow. A shadow is a copy of the original (master) databank and is continuously updated by Mimer SQL. If the master databank is lost, it is possible to continue operations from the shadow databank without stopping the database server. A databank must have the TRANS or LOG option to be shadowed.

SQL	Structured Query Language, standardized language for database manipulation.
SQLHOSTS	A file containing lookup information for all accessible Mimer SQL databases, relative to the current node.
Table	Tables (or relations) hold all the information in a relational database. A table is stored in a databank. It may not be split across databanks, but a databank may contain several tables.

(All other trademarks are the property of their respective holders.)

2 MIMER SQL INSTALLATION

This chapter describes the installation of the Mimer SQL software and the preparations needed to get a database system up and running.

Tasks to be performed during installation of Mimer SQL

- Unpack the distribution file to a directory tree.
- Software setup. (MIMSETUP8)
- Establish the Mimer SQL database(s) that reside on or are to be accessed from the VMS machine.
- Add a MIMSETUP8 command in the SYSTARTUP_VMS.COM file so that the Mimer SQL installation is set up each time the system boots.
- Use the MIMLICENSE application to enter any additional Mimer SQL license keys. A default key, for test and development only, is automatically installed, i.e. its possible to finish the installation without adding a key.

2.1 VMS system requirements

To be able to use MIMER8 on the OpenVMS Alpha, version 7.2-1H1 of the operating system, or later, is needed.

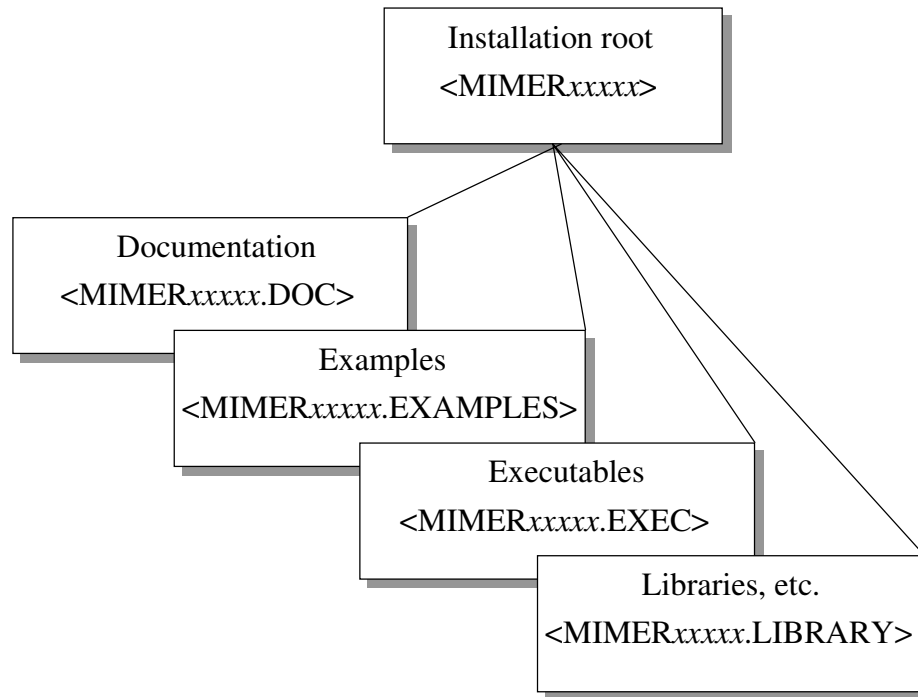
If you are using VMS 7.2-1H1, the following VMS patch must be installed:

- VMS721_UPDATE-V0300

The installed Mimer SQL product requires about 50,000 disk blocks (25 Mb) including on-line documentation.

2.2 Unpacking the Mimer SQL distribution file

The MIMER8 software resides in a directory tree, illustrated in the figure below. The name of the installation root directory in the tree contains the word “MIMER” and the version number of the product, e.g.: “MIMER822A” (generally denoted as MIMER_{xxxxx}).



The directory structure when unpacked and directory contents.

The name of the root directory is unique for each Mimer SQL release, which makes it easier to install new versions without affecting any previous versions of the product.

Note: The three consecutive dots in the “[...]” construction used in the VMS command examples that follow, together with savesets, are an essential part of the command syntax. If they are omitted, all files on the distribution media will be assigned to a single directory and the Mimer SQL installation will fail.

2.2.1 Unpacking a self-extracting VMS ZIP file

Self-extracting ZIP files (created using Info-ZIP) are used when distributing Mimer SQL on the Internet.

Such files look like ordinary executable (.EXE) files. For example, the file containing Mimer SQL version 8.2.2A would be named “MIMER822A.EXE”.

To unpack the contents, simply execute the file. The Mimer SQL distribution directory will be created under the current directory.

```
$ SET DEFAULT disk:[000000]      ! Create tree at root level
$ RUN disk:[directory]MIMERxxxxx
```

The auto-extract facility may not always apply the correct file protections to the files it extracts, therefore it is recommended that the following commands be run to ensure that the correct file protections are applied to the files in the tree:

```
$ SET FILE/PROT=(S:RWED,O:RWED,G:RE,W:RE) [MIMERxxxxx...] *.*
$ SET FILE/PROT=(S:RWE,O:RWE,G:RE,W:RE) MIMERxxxxx.DIR
```

2.2.2 Unpacking a VMS saveset from a tape

When distributing the Mimer SQL software for VMS on tape, VMS savesets are primarily used. Unpack it as follows:

```
$ MOUNT/FOREIGN tape:
$ SET DEFAULT disk:[000000]
$ BACKUP tape:/SAVE_SET [...]
$ DISMOUNT tape:
```

2.3 The MIMSETUP8 command

Before the MIMER8 software can be used, certain setup operations must be performed. The Mimer SQL environment, i.e. locations for programs, libraries, data files, documentation, etc., must be defined for users and applications.

The MIMSETUP8 command procedure is found in the Mimer SQL root directory. It defines the logical names needed to run Mimer SQL applications.

Note: The VMS user running MIMSETUP8 may require some of the following privileges: SYSPRV, CMKRNL, SYSNAM (see below for details).

2.3.1 MIMSETUP8 syntax

The syntax for the MIMSETUP8 command is as follows:

```
$ @disk:[MIMERxxxxx]MIMSETUP8 [lnm-table]
```

The parameter *lnm-table* specifies which logical name table to use when defining the logical names required to access a Mimer SQL installation.

Valid values are: SYSTEM, GROUP, JOB, PROCESS.

If the user specifies SYSTEM or GROUP, the following shared images will be installed (see Section 3.2), if they are not installed already:

```
MIMLIB8:MIMDBP8.EXE
MIMLIB8:MIMDB8.EXE
MIMLIB8:MIMDBS.EXE
```

Therefore, a SYSTEM or GROUP level setup **must** be performed at least once for a Mimer SQL installation in order to get these essential shared images installed.

It is generally recommended that MIMSETUP8 is executed to update the SYSTEM logical name table, so that the definitions are available to all users.

In order to define logical names SYSTEM wide, the user must have SYSPRV, CMKRNL and SYSNAM privileges. When logical names are defined in the SYSTEM table, they are defined in executive mode.

To define logical names GROUP wide, the user must have SYSPRV and CMKRNL privileges.

Since SYSTEM or GROUP wide setups have to be re-executed each time the VMS system is booted, it is recommended that the command(s) be entered into the system startup command file (SYS\$MANAGER:SYSTARTUP_VMS.COM).

MIMSETUP8 can be performed at the PROCESS or JOB level by an individual user to set up logical names that may be different to those available from the SYSTEM or GROUP level (no shared image installation is involved in a PROCESS or JOB level setup).

If MIMSETUP8 is used without specifying the *lnm-table* parameter, a **PROCESS** level setup is performed by default.

If the parameter *lnm-table* is preceded by a hyphen ("-"), the MIMSETUP8 command procedure will **remove** the effects of any Mimer SQL setup previously performed for the specified table, including de-installation of shareable images.

MIMSETUP8 examples

- 1) Definition of logical names SYSTEM wide, i.e. all VMS users may access the Mimer SQL installation. Shareable images are installed.

```
$ @SDEPT2: [MIMER822A]MIMSETUP8 SYSTEM
```

- 2) Any user can override the default definition of the logical names. This is useful when a user wishes to test an alternate Mimer SQL installation. No shareable images are installed.

```
$ @SDEPT2: [MIMER822A]MIMSETUP8
```

- 3) A user can remove a Mimer SQL setup which was previously made GROUP wide by running MIMSETUP (MIMROOT8 was defined by MIMSETUP). Shareable images are de-installed.

```
$ @MIMROOT8: [000000]MIMSETUP8 -GROUP
```

2.3.2 Logical names defined by MIMSETUP8

The MIMSETUP8 command procedure defines the logical names listed below:

MIMROOT8	A concealed logical name pointing to the root of the Mimer SQL distribution.
MIMER_SQLHOSTS	Points to the SQLHOSTS file which contains one entry for every accessible Mimer SQL database. Normally this logical name is set to SYSS\$SPECIFIC:[SYSMGR]SQLHOSTS.DAT
MIMDB8	Logical name pointing to the Mimer SQL shareable library. Used when starting Mimer SQL applications.
MIMDBP8	Logical name pointing to the MIMDBP8 image. Used when starting Mimer SQL applications.
MIMDOC8	Points to the directory containing on-line documentation.
MIMEXAMPLES8	Points to the examples directory in the Mimer SQL distribution.
MIMEXE8	Points to the directory containing executable programs in the Mimer SQL distribution.
MIMLIB8	Points to the directory containing application libraries, CLD files, etc.

2.4 Installing the Mimer SQL license key

A license key for test and development is included in the Mimer SQL software distribution and is automatically installed. This means it is possible to set up a complete Mimer SQL environment without adding any additional license keys.

To install a Mimer SQL license key, see Section 3.4.1 of the *Mimer SQL System Management Handbook* for details on how to enter key data using the MIMLICENSE application.

To get the Mimer SQL license key data, the node name must be provided to the Mimer SQL distributor.

Note: The VMS user entering the Mimer license key must have appropriate access to the file SYSS\$SPECIFIC:[SYSMGR]MIMERKEY.DAT.

Make sure that all VMS users authorized to run Mimer SQL have read access to the key file (file protection GROUP:R or WORLD:R, depending on the site organization).

2.5 Establishing a Mimer SQL database

Having installed the Mimer SQL software and any additional Mimer SQL license keys, you can establish the database(s) which are to reside on the VMS machine (local databases).

You can also make any databases residing on other network nodes (remote databases) accessible from the VMS machine.

The remaining activities involved in setting up the Mimer SQL installation are described in the *Mimer SQL System Management Handbook* (see below for specific chapter references). The remainder of this guide covers various additional points that relate to the VMS platform.

Refer to Chapter 2 of the *Mimer SQL System Management Handbook* for background information which is useful for understanding the issues and the different components involved in establishing a Mimer SQL database.

Refer to Chapter 3 of the *Mimer SQL System Management Handbook* for details of the actual steps to be followed when establishing local and remote Mimer SQL databases.

Before a database can be accessed, the database server for it must be started on the machine it resides on. Refer to Chapter 4 of the *Mimer SQL System Management Handbook* for details on controlling and managing database servers.

Note: The VMS user establishing Mimer SQL databases on a VMS node must have write access to the file pointed to by the logical name MIMER_SQLHOSTS and should use a text editor that handles logical names correctly.

2.6 Removing a Mimer SQL installation

To remove a Mimer SQL installation, perform the steps listed below. If you plan to remove any Mimer SQL databases (see Section 3.10 of the *Mimer SQL System Management Handbook*), please do this **before** removing the Mimer SQL installation.

- Check that no Mimer SQL applications or database servers are using the installation.
- Run the MIMSETUP8 command procedure to un-install shared images and un-assign logical names (see Section 2.3)

```
$ disk: [MIMERxxxxx]MIMSETUP8 -SYSTEM
```

- Delete the Mimer SQL directory tree. Note that you may have to issue the DELETE command more than once:

```
$ SET DEF disk: [000000]
$ SET PROC/PRIV=BYPASS
$ DELETE [MIMERxxxxx...]*.*.*
$ DELETE [MIMERxxxxx...]*.*.*
$ DELETE [MIMERxxxxx...]*.*.*
$ SET PROC/PRIV=NOBYPASS
```

- If there are no other Mimer SQL installation trees in the system, you may want to delete the SQLHOSTS file and the Mimer SQL license key file:

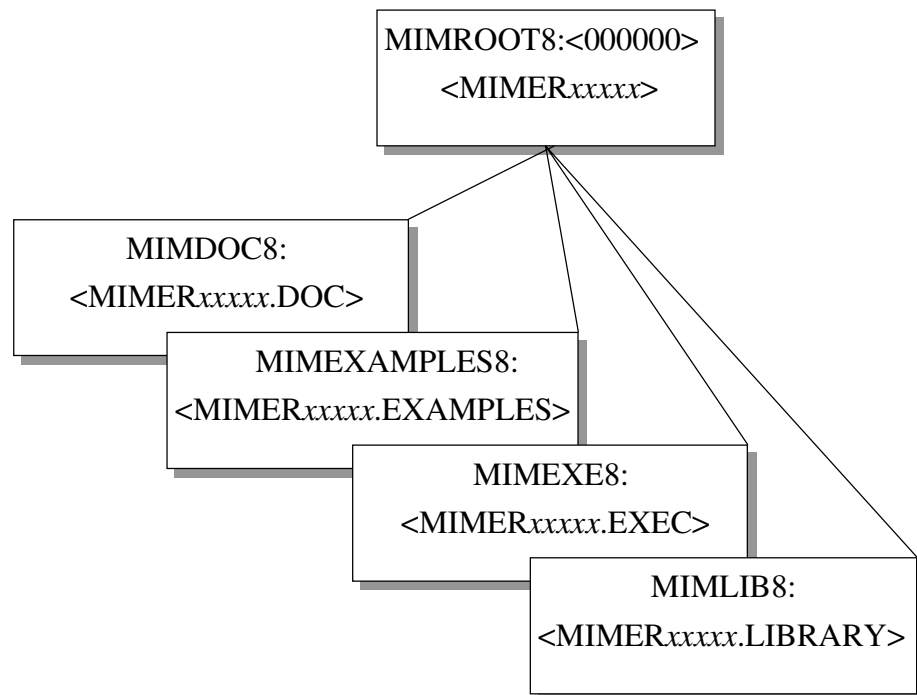
```
$ DELETE SYS$SPECIFIC:[SYSMGR]SQLHOSTS.DAT.*  
$ DELETE SYS$SPECIFIC:[SYSMGR]MIMERKEY.DAT.*
```

- If you have added any Mimer SQL-related commands to the VMS startup file (SYS\$MANAGER:SYSTARTUP_VMS.COM) or the VMS shutdown file (SYS\$MANAGER:SYSHUTDOWN.COM), you should remove those commands.

3 MIMER SQL INSTALLATION COMPONENTS

3.1 Distributed files

The distributed files are all located in a directory structure, illustrated in the figure below. The figure also shows the logical names that are set to point to the different placeholders.



The directory structure and the related logical names.

3.1.1 Root directory files

DEFAULTKEY.MCFG	Default Mimer SQL license key.
MIMSETUP8.COM	Command procedure that defines logical names and installs images, see Section 2.3.
VERSION.DAT	Contains the version number of the Mimer SQL distribution.

3.1.2 Documentation files (MIMDOC8)

RELNOTES.PDF	Mimer SQL Release Notes.
MIMJDBC- <i>n_n</i> .HTML	JDBC usage guide.

3.1.3 Example files (MIMEXAMPLES8)

BLOBSAMP.EC	Example of a program written in embedded C that stores and retrieves binary data.
CJDATE.SQL	SQL examples.
CREHOTDB.DAT	Contains SQL commands that create an example database, see Section 3.9 of the <i>Mimer SQL System Management Handbook</i> .
DSQL.EC	Embedded C examples that demonstrates the use of dynamic SQL.
DSQL.H	Header file for the dynamic SQL example.
DSQLSAMP.C	Main program for the dynamic SQL example.
EXAMPLE.EC	Very simple embedded C example.
EXAMPLE.ECO	Very simple embedded COBOL example.
EXAMPLE.EFO	Very simple embedded FORTRAN example.
EXAMPLE.JAVA	Java example program using JDBC.
EXAMPLES.SQL	Text file containing examples of SQL commands. This assumes that the example database is loaded, see Section 3.9 of the <i>Mimer SQL System Management Handbook</i> .
FREQCALL.EC	Example of a program written in embedded C that calls a stored procedure.
FREQCALL.ECO	Same as FREQCALL.EC, but written in COBOL.
FREQCALL.EFO	Same as FREQCALL.EC, but written in FORTRAN.
SINGLEDEFS.DAT	Contains a template for database parameters in single-user mode, see Section A.4 of the <i>Mimer SQL System Management Handbook</i> .
SQLHOSTS.DAT	Contains a template for the SQLHOSTS.DAT file. The actual SQLHOSTS file is pointed to by the MIMER_SQLHOSTS logical name (normally SYSS\$MANAGER:SQLHOSTS.DAT). Do not edit this template file.
WAKECALL.EC	Example of a program written in embedded C that calls a stored procedure.

WAKECALL.ECO	Same as WAKECALL.EC, but written in COBOL.
WAKECALL.EFO	Same as WAKECALL.EC, but written in FORTRAN.

3.1.4 Executable programs (MIMEXE8)

BSQL.EXE	Program that executes SQL statements which are entered interactively or read from a command file (described in the <i>Mimer SQL User's Manual</i>).
DBC.EXE	Program that can check if a databank file is internally consistent (described in the <i>Mimer SQL System Management Handbook</i>).
DBOPEN.EXE	Program that opens and restarts all databanks in a database (described in the <i>Mimer SQL System Management Handbook</i>).
DBSERVER.EXE	The database server. Do not start this program directly, use MIMCONTROL to do this.
ESQL.EXE	Pre-processor for embedded SQL.
MIMCONTROL.EXE	The MIMCONTROL command, which is used to control database servers (see Section 5.1).
MIMINFO.EXE	Program that can display status information for a database server.
MIMLICENSE.EXE	Application used to administrate the license key(s).
MIMTCP.EXE	The program executed by the MIMTCP server (see Section 5.3). Do not start this program directly.
SDBGEN.EXE	Program used to create the initial Mimer SQL system databank files in a database (described in the <i>Mimer SQL System Management Handbook</i>).
UTIL.EXE	Utility functions for a database (see the <i>Mimer SQL System Management Handbook</i>).

3.1.5 Library files (MIMLIB8)

LR.OLB	Library with entries for backward compatibility.
LRU.OLB	Library with entries for backward compatibility.
MDR.OLB	Library with entries for backward compatibility.
MIMDB8.EXE	Shareable library image containing the code for the Mimer SQL database client API (see Section 3.2).

MIMDBP8.EXE	Protected shareable library image containing code that performs secure and fast shared memory based communication with local database servers.
MIMDBS.EXE	Shareable library image containing code for running a Mimer SQL database in single-user mode. This library is mapped in dynamically when required.
MIMER.CLD	Command definitions for all the executable programs supplied with the Mimer SQL software.
MIMER.OPT	Options file used for linking Mimer SQL applications.
MIMJDBC- <i>n_n</i> .JAR	JDBC driver.
MIMODBC8.EXE	ODBC driver library.

3.2 Shared images

The Mimer SQL distribution contains a number of shareable images which are located in the MIMLIB8 directory (see Section 3.1.5).

The shared images are installed by the MIMSETUP8 command procedure when defining logical names in the SYSTEM or GROUP name table.

Mimer SQL applications are linked with the shareable library MIMLIB8:MIMDB8.EXE. When the application image is activated, the VMS system locates the correct shareable image by using the logical name MIMDB8 (which MIMSETUP8 has defined as MIMLIB8:MIMDB8). This allows users to run applications with other versions of Mimer SQL by using the MIMSETUP8 procedure, without having to re-link the applications.

If starting an image that is installed with privileges or if the user does not have read (R) access to the image file, VMS will only translate logical names defined in executive mode when activating shareable images. This is a security precaution that VMS takes to avoid activating non-trusted shareable images together with trusted images.

Note: All logical names that MIMSETUP8 defines in the SYSTEM logical name table are defined in executive mode. This means that privileged or protected images will run the Mimer SQL version defined in the SYSTEM table even if there is another Mimer SQL version defined in one of the other tables!

4 RUNNING MIMER SQL APPLICATIONS

This chapter describes how to run applications in the MIMER8 environment.

It covers information that applies to the applications included in the Mimer SQL installation as well as to applications that may have been created to access a Mimer SQL database.

This chapter also describes:

- Defining whether VMS-style or Unix-style command-line flags are accepted by the applications which are supplied as part of the Mimer SQL installation.
- Selecting a Mimer SQL installation - if several Mimer SQL installations reside on the same computer it is essential that users access the correct one.

4.1 Executing Mimer SQL applications

This section describes the various ways an application can be executed under VMS and also describes how to set up the Mimer SQL-supplied programs to use Unix-style or VMS-style command-line flags.

4.1.1 Using the DCL command RUN

The DCL command RUN can be used as follows:

```
$ RUN disk:<directory.app>my_appl.exe
```

It is not possible to supply any flags or other input parameters on the command-line when the RUN command is used.

Some of the Mimer SQL-supplied programs allow parameters and options to be supplied via logical names, e.g. MIMER_DATABASE to supply a database name and MIMER_MODE to define the database access mode (see documentation for the programs in the *Mimer SQL System Management Handbook* for specific details).

4.1.2 Using the DCL\$PATH logical name

The DCL\$PATH logical name defines a list of directories in which the VMS operating system will look when trying to locate the executable for a specified program name.

In order for the programs supplied by Mimer SQL to be run by specifying the program name followed by the **Unix-style** command line flags and parameters, the MIMEXE8 directory must be included in the directory list defined in DCL\$PATH.

If there are other directories containing executables for programs that are to be run this way, those directories must also be included in the directory list defined in DCL\$PATH.

For example, the following DCL\$PATH definition:

```
$ DEFINE DCL$PATH MIMEXE8, disk:<directory.app>
```

will allow the programs supplied by Mimer SQL to be run by specifying the program name followed by the Unix-style command line flags and parameters. It will also allow all programs found in the *app* sub-directory of *directory* on *disk:* to be run by specifying the program name.

Example using Unix-style command-line flags:

```
$ bsq1 -s mydb
```

4.1.3 Using VMS command definitions

It is possible to set up the programs supplied by Mimer SQL so that they may be run by specifying the program name followed by the **VMS-style** command line flags and parameters.

This is done by defining the Mimer SQL programs as DCL command verbs.

Issuing the following VMS command will define all the Mimer SQL-supplied programs as DCL command verbs:

```
$ SET COMMAND MIMLIB8:MIMER
```

Example using VMS-style command-line flags:

```
$ BSQL/SINGLE MYDB
```

Note: If a Mimer SQL-supplied program is set up as a DCL command verb, the Unix-style command-line flags and parameters **cannot** be used, even if MIMEXE8 is included in DCL\$PATH.

It is possible to un-define a DCL command by issuing the following command:

```
$ SET COMMAND/DELETE=command-name
```

Care should be taken when using this DCL command, because any DCL command verb can be un-defined.

4.2 Selecting a Mimer SQL installation

A host computer may have several versions of the Mimer SQL database system installed simultaneously. Access through a specific version is done through logical names (MIMEXE8, etc.). Since the major version number (currently 8) is included in the logical names, Mimer SQL version 7 and version 8 can run concurrently without any disruption.

If several versions of MIMER8 are installed, the MIMSETUP8 command procedure can be used to specify exactly which version a program should work with. Normally, a system wide definition of the logical names is made. However, a user may specify another Mimer SQL version by running the MIMSETUP8 command procedure and specifying a GROUP, JOB or PROCESS logical name definition (see Section 2.3 for details on MIMSETUP8).

Note: When starting a database server, any JOB or PROCESS logical names will **not** be inherited by the database server process. The database server will use the Mimer SQL version specified in the GROUP or SYSTEM logical name tables.

5 MANAGING A DATABASE SERVER

This chapter contains information relevant to the administration of a database server under VMS.

For general information on managing database servers, refer to Chapter 4 of the *Mimer SQL System Management Handbook*.

This chapter also describes:

- Using the MIMCONTROL command under VMS.
- The addition of commands to the SYSTARTUP_VMS and SYSHUTDWN files to automatically start or stop database servers when the system comes up or goes down.

5.1 The MIMCONTROL command

Database servers on VMS are controlled by using the MIMCONTROL command (refer to Section 4.2.1 of the *Mimer SQL System Management Handbook* for details).

Note: The MIMCONTROL program **cannot** be started by using the DCL command RUN.

5.1.1 Privileges needed for MIMCONTROL execution

A user running the MIMCONTROL command must have either:

SETPRV privilege

or

CMKRNL, CMEXEC, SHMEM, SYSPRV, WORLD, TMPMBX, OPER, NETMBX, PSWAPM, DETACH, ALTPRI, PRMGBL, SYSGBL, SYSLCK and SYSNAM privileges.

5.1.2 Prerequisites for database server startup

In order to successfully start a database server, the following conditions must be fulfilled:

- The system databank (SYSDB) must have been created.
- There must be an entry for the database in the local section of the SQLHOSTS file.
- The ProcName of the MULTIDEFS file must not specify a process name prefix that is identical to that of another running multi-user system.
- The MIMSETUP8 command procedure must have defined the logical names to be SYSTEM-wide or GROUP-wide.
- There must not be any logical names in the JOB or PROCESS tables that override the SYSTEM or GROUP definitions.
- The shareable image in the file named MIMLIB8:MIMDBP8.EXE must be properly installed.
- There must not be any other node in a cluster which has started the same database server.
- The database must not be in use in single-user access mode at the time the database server is started.
- The file SYS\$MANAGER:MIMERKEY.DAT must contain a valid Mimer SQL license key.

5.2 Automatic database start and stop

If you want the database server to start automatically whenever the system is booted, you should edit the SYS\$MANAGER:SYSTARTUP_VMS.COM file and add the relevant commands at the end. The following example sets up a Mimer SQL installation (MIMSETUP8) and then starts two database servers:

```
$ @SDEPT2 : [MIMER822A]MIMSETUP8 SYSTEM
$ SET COMMAND MIMLIB8:MIMER
$ MIMCONTROL/START PRODUCTION
$ MIMCONTROL/START INVENTORY
```

If you want to perform a controlled shutdown of the database server whenever the VMS system is shut down, you should edit the SYS\$MANAGER:SYSHUTDWN.COM file and add the relevant commands at the end. The following example stops two database servers:

```
$ SET COMMAND MIMLIB8:MIMER
$ MIMCONTROL/STOP PRODUCTION
$ MIMCONTROL/STOP INVENTORY
```

5.3 The MIMTCP server

In MIMER8, a MIMTCP server listening to a specific port (usually port 1360) will be started whenever a database server is started. The TCP/IP port number this server will listen to is specified in the TCPPort parameter in the MULTIDEFS.DAT file. If several database servers specify the same port number, they will share the same MIMTCP server.

When a client connects to the TCP/IP port, the MIMTCP server will accept the connection. The client specifies the database to which a connection is to be established and the MIMTCP server will hand over the connection to the appropriate database server. All further communication between the client and the database server is then done directly without involving the MIMTCP server.

Whenever a MIMTCP server starts, it will define the system logical name "MIMTCP_xxxx" (where xxxx is the port number) to be the PID of the MIMTCP server process. This makes it easy to find the MIMTCP server process that is listening to a particular TCP/IP port.

There is no explicit command for stopping MIMTCP servers as there is generally no need to do this. The MIMTCP process does not have to be stopped or restarted when database servers are stopped or (re)started.

To manually stop the MIMTCP server process listening to port 1360 (for example), execute the following commands:

```
$ SHOW LOG MIMTCP_1360      ! Find PID of process
$ STOP/ID=xxxxxxxxx        ! Delete the process found
```

5.3.1 Manually starting a MIMTCP server

It is normally not necessary to start a MIMTCP server manually because the server process is started automatically when a database server is started.

It is possible to start several MIMTCP servers listening on several ports. This will make version 8 servers accessible from all ports.

A user starting a MIMTCP server must have either:

SETPRV privilege

or

SYSPRV, TMPMBX, OPER, NETMBX and SYSNAM privileges.

To manually start a MIMTCP server process listening to port 1377 (for example), execute the following command:

```
$ RUN/PRIV=(SYSPRV,TMPMBX,OPER,NETMBX,SYSNAM)/DETACH/INPUT=1377 -
$_ /OUTPUT=MIMROOT8:[000000]MIMTCP.LOG MIMEXE8:MIMTCP
```

The /INPUT parameter is used to specify the port number on which the server process listens.

The /OUTPUT parameter is used to specify the server process log file. Note that several server processes can share the same log file.

6 USING THE MIMER JDBC DRIVER ON VMS

The Mimer SQL distribution includes a JDBC driver. This driver allows Java programs running on VMS to access any Mimer SQL database server running at least V8.2. The JDBC driver is a "type 4" driver which means that it is written entirely in Java, and can be moved to any platform supporting Java.

The Mimer JDBC driver resides in MIMLIB8:MIMJDBC-*n_n*.JAR.

For more information about Java and JDBC, please see:

- SYSS\$COMMON:<SYSHLP.JAVA.RELEASE_NOTES>JDK118_VMS_RELEASE_NOTES.HTML
Information about using Java on VMS.
- MIMDOC8:MIMJDBC-*n_n*.HTML
Information about the Mimer JDBC driver.
- <http://java.sun.com/products/jdbc>
JDBC technology information from Sun.

6.1 Example of using the JDBC driver

First the VMS Java environment needs to be set up. The following commands define the Java commands:

```
$ DEASSIGN JAVA$USE_DCL
$ @SYS$MANAGER:JAVA$SETUP
```

To use the Mimer JDBC driver, the Java environment must be able to find it. The logical name CLASSPATH is used for this purpose. This logical name contains a list of directories and Java archives (.ZIP and .JAR files).

Please note that the CLASSPATH logical name specifies the file names using a UNIX syntax. It is also possible to use the JAVA\$CLASSPATH logical name which uses standard VMS file specifications. Please read the VMS Java release notes for further details.

The following example first determines the exact version of the file to be used, and then sets the CLASSPATH logical name accordingly. Since the equivalence string becomes rather long, and must be enclosed in quotes, a DCL string is constructed.

```
$ DIR MIMLIB8:MIMJDBC*.JAR

Directory MIMROOT8:<LIBRARY>

MIMJDBC-1_3.JAR;1

Total of 1 file.
$ SHOW LOG CLASSPATH
"CLASSPATH" = "/sys$common/java/lib/JDK118_CLASSES.ZIP:."
(LNM$PROCESS_TABLE)
$ CLASSPATH=F$TRNLNM("CLASSPATH")+":/MIMLIB8/MIMJDBC-1_3.JAR"
$ DEFINE CLASSPATH "'CLASSPATH'"
```

The Mimer JDBC driver should now be accessible. Please note that the driver contains a main() function so it is possible to execute it as a program for testing purposes. We first use the "-version" switch to verify that the Java environment can locate and use the Mimer JDBC driver. Note that quotes must be used since Java package names are case sensitive.

```
$ JAVA "com.mimer.jdbc.Driver" -version
Mimer JDBC driver version 1.3
```

Since the driver seems to work we now use the "-ping" switch to test that the driver can make a connection with a Mimer V8.2 database server. Please read the JDBC driver guide for an explanation of the syntax of the connection URL.

```
$ JAVA "com.mimer.jdbc.Driver" -ping -
$_ "jdbc:mimer://SYSADM:PASSWORD@mynode/mydatabase"
Database connection established.
getDatabaseProductName(): MIMER/DB
getDatabaseProductVersion(): 08.02.0001 MIMER/DB 8.2.04
```

```
Ping tests:
 0      2 ms
 1      2 ms
 2      1 ms
 3      1 ms
 4      1 ms
 5      1 ms
 6      0 ms
 7      2 ms
 8      1 ms
 9      1 ms
avg     1 ms      min      0 ms      max      2 ms
```

Finally, the JDBC example program is compiled and executed. You should copy the example program to a private directory and edit it in order to set the connection URL string, database username and passwords.

```
$ SET DEF [SOMEWHERE.PRIVATE]
$ COPY MIMEXAMPLES8:EXAMPLE.JAVA []
$ ! Edit the example. Alter the URL and username/password
$ EDIT EXAMPLE.JAVA
$ JAVAC EXAMPLE.JAVA
$ JAVA "Example"
```

A DATA TYPES USED IN MIMER SQL

A.1 Compiling applications using floating point data types

The floating point data types supported on Alpha/VMS are F_FLOAT (4 bytes) and G_FLOAT (8 bytes). The default compiler option, /G_FLOAT, should be used when compiling programs that are to be linked with Mimer SQL libraries.

Mimer SQL does not currently support D_FLOAT or the use of the two IEEE floats, S_FLOAT and T_FLOAT. (H_FLOAT is also not supported by Mimer SQL on Alpha/VMS since that data type is not supported natively by the Alpha architecture).

A.2 Internal Mimer SQL representation

Mimer SQL uses only two data representations internally: numeric and character. These two types are used to store all data in the database.

The numeric data type stores numbers in packed BCD format with a maximum precision of 45 digits. Every number stored has an exponent with a range of -999 to +999.

The character data type uses the ISO 8859-1 standard character set, also known as Latin 1. This standard specifies graphic characters and the representation of each character. It is a proper extension of the DEC multinational character set, with a few exceptions.

On most platforms, including VMS, the ISO 8859-1 character set is used by Mimer SQL to transfer data to and from the application, i.e. the same character set as is used internally.

Apart from VT100 and VT200, which only support the DEC multinational character set, the VT300 series terminals support both DEC multinational and ISO 8859-1. Since the two character sets are quite similar it is recommended that any VMS site using VT300 terminals uses the ISO character set.

The ISO 8859-1 character set is presented in Appendix B of the *Mimer SQL Reference Manual*.

A.3 External data types supported by Mimer SQL

Any value stored in the database may be read into host language variables as described in Appendix A of the *Mimer SQL Programmer's Manual*. Mimer SQL will perform all the necessary conversions and will signal an error if the value to be converted is not compatible with the destination type.

B USING MIMER7 APPLICATIONS WITH MIMER8

General database upgrade information is provided in the *Mimer SQL Release Notes*. This appendix describes the specific task of using a MIMER7 application with a MIMER8 database server.

If a MIMER7 application is to be used with a MIMER8 database server, there are four main approaches to establishing communication:

- Re-link the application with the MIMDB8 library.
- Use the sharable images distributed with MIMER8.
- Connect to the database using the client/server interface, i.e. from a MIMER7 client node to a MIMER8 server node.
- Connect to the database using the client/server interface locally, i.e. the MIMER7 client and MIMER8 server are on the same node.

Please note that the MIMER7 application needs a MIMER7 license (located in MIMKEY7). The MIMER8 database server needs a MIMER8 license located in SYSSPECIFIC:[SYSMGR]MIMERKEY.DAT. A single machine will need both licenses if it runs MIMER7 applications and a MIMER8 database server.

B.1 Re-link applications

MIMER7 applications can be re-linked for use in a MIMER8 environment, accessing a MIMER8 database server. This method is recommended for applications running on the same machine as the database server.

In this case the MIMER.OPT file distributed with MIMER8 should be used, instead of the one distributed with MIMER7. Other MIMER7 libraries that may be used by the application, such as SH (Screen Handler), FM (Forms Manager), etc., should be linked in as before.

B.2 Remap shareable libraries

A MIMER7 application can access a MIMER8 database server by mapping in the MIMER8 sharable image instead of the MIMER7 one. This method is recommended for Mimer version 7 tools, such as PG or QL.

This can be achieved by defining the following logical names, after executing MIMSETUP7:

```
$ DEFINE MIMDB7M MIMDB8
$ DEFINE MIMDB7S MIMDB8
```

B.3 Client-server access

MIMER7 applications and tools can access a MIMER8 database server through the client/server interface without any modification to the application. This method is recommended when the version 8 database server is on a remote node in a network.

The entry created for a MIMER8 database server in the SQLHOSTS file is described in Appendix B of the *Mimer SQL System Management Handbook*.

In MIMER8, the SERVICE field of the SQLHOSTS file should specify the specific port (TCP/IP) or network object (DECNET) to which the database server listens. If the NODE field is set to the name of the current node and the SERVICE field specifies a local database server, the connection will be established to the local server, using the client/server interface.

B.4 Local client/server access

It is possible to use the client/server interface locally, i.e. using a remote database definition which actually points to a local database server.

To achieve this, the file MIMLIB7:SQLHOSTS.DAT must be updated in a specific way using a 6th parameter in the remote definition, as shown in the following example (current node is "STARTREK", i.e. where the MIMER8 database "M8SERVER" runs):

```
--
-- =====
DEFAULT:
--
-- Database
-----
SINGLE
-- =====
LOCAL:
--
-- Database          Path
-----
SINGLE              SYS$DISK: []
M8SERVER           DISK: [DIRECTORY]
-- =====
REMOTE:
--
-- Database          Node          Protocol Interface Service
-----
example_database   server_nodename  ''         ''         1360
M8ACCESS          STARTREK        TCP        ''         1360          M8SERVER
```

To access the "M8SERVER" database, the MIMER7 application must connect to the database "M8ACCESS".