



VMS Software

LibMariaDB Version 3.1-0B for VSI OpenVMS

Release Notes

Publication Date: June 2025

Operating Systems: VSI OpenVMS Alpha Version 8.4-2L1 or higher
VSI OpenVMS IA-64 Version 8.4-2L1 or higher
VSI OpenVMS x86-64 Version 9.2-3 or higher

Kit Names: VSI-AXPVMS-LIBMARIADB-V0301-0B-1.PCSI
VSI-I64VMS-LIBMARIADB-V0301-0B-1.PCSI
VSI-X86VMS-LIBMARIADB-V0301-0B-1.PCSI

Table of Contents

1. Introduction	3
2. Acknowledgements	3
3. What's New in This Release	3
4. Requirements	3
5. Recommended Reading	4
6. Installing the Kit	4
6.1. Post-Installation Steps	5
6.2. Privileges and Quotas	5
7. Sample Applications	5
8. What's missing?	6

1. Introduction

Thank you for your interest in this port of the MariaDB C client API to VSI OpenVMS. This version 3.1.0B release of the API for VSI OpenVMS is based on the LGPL libMySQL client library from MySQL 3.23 and PHP's MySQLnd extension.

The MariaDB C client API can be used to connect software applications developed in C/C++ to MariaDB and MySQL databases. This port of the API for OpenVMS also includes a language-agnostic API that can be used to develop applications in other languages such as COBOL, FORTRAN, Pascal, and BASIC. In most cases the client application and database will reside on different hosts and communication between client and database would be via TCP/IP.

This OpenVMS port of the MariaDB client API includes all core functionality; however, some functionality such as support for Connector/C 3.0 functions is not included in this release. It is anticipated that this and other deficiencies will be addressed in future releases.

2. Acknowledgements

VMS Software Inc. would like to acknowledge the work of the MariaDB and PHP teams for the creation and ongoing support of the C client API. VMS Software Inc. would like to thank Neil Rieck for kindly allowing some of his example code for OpenVMS to be included with the kit.

3. What's New in This Release

New features included in this release of the MariaDB client API for VSI OpenVMS include the following:

- Inclusion of a language-agnostic API that can be used to develop applications in other languages such as COBOL, FORTRAN, Pascal, and BASIC.
- SSL/TLS support using OpenSSL 3.0
- Threads support (the API can be used in multi-threaded applications)
- Support for VSI OpenVMS x86-64 V9.2-3 or higher

Additionally, this release addresses an issue on VSI OpenVMS x86-64 where some API functions could not be used with the C++ compiler due to differences in size between C and C++ on x86-64 for the long data type.

For a detailed description of other features and bug fixes included in this release, please read the following release notes: <https://mariadb.com/kb/en/mariadb-connector-c-310-release-notes/>.

4. Requirements

The kit you are receiving has been compiled and built using the operating system and compiler versions listed below. While it is highly likely that you will have no problems installing and using the kit on systems running higher versions of the products listed, we cannot say for sure that you will be so lucky if your system is running older versions.

- OpenVMS 8.4-2L1 or higher (IA-64), OpenVMS V8.4-2L1 or higher (Alpha), OpenVMS 9.2-3 or higher (x86-64)

- VSI TCP/IP
- Your preferred programming language compiler

In addition to the above requirements, it is assumed that the reader has a good knowledge of OpenVMS and of software development in the OpenVMS environment.

5. Recommended Reading

It is recommended that developers read the comprehensive API documentation on the MariaDB web site (<https://mariadb.com/kb/en/mariadb/mariadb-connector-c/>) before using the software.

6. Installing the Kit

The kit is provided as an OpenVMS PCSI kit (VSI-AXPVMS-LIBMARIADB-V0301-0B-1.PCSI, VSI-I64VMS-LIBMARIADB-V0301-0B-1.PCSI, or VSI-X86VMS-LIBMARIADB-V0301-0B-1.PCSI) that can be installed by a suitably privileged user using the following command:

```
$ PRODUCT INSTALL LIBMARIADB
```

The installation will then proceed as follows (output may differ slightly from that shown):

```
Performing product kit validation of signed kits ...
```

```
The following product has been selected:
```

```
    VSI X86VMS LIBMARIADB V3.1-0B          Layered Product
```

```
Do you want to continue? [YES]
```

```
Configuration phase starting ...
```

```
You will be asked to choose options, if any, for each selected product and
for any products that may be installed to satisfy software dependency
requirements.
```

```
Configuring VSI X86VMS LIBMARIADB V3.1-0B
```

```
    VMS Software Inc. & the MariaDB team
```

```
* This product does not have any configuration options.
```

```
Execution phase starting ...
```

```
The following product will be installed to destination:
```

```
    VSI X86VMS LIBMARIADB V3.1-0B          DISK$X86SYS:[VMS$COMMON.]
```

```
Portion done: 0%...90%...100%
```

```
The following product has been installed:
```

```
    VSI X86VMS LIBMARIADB V3.1-0B          Layered Product
```

```
VSI X86VMS LIBMARIADB V3.1-0B
```

```
    Post-installation tasks are required.
```

To start LibMariaDB at system boot time, add the following lines to SYS\$MANAGER:SYSTARTUP_VMS.COM:

```
$ file := SYS$STARTUP:LIBMARIADB$STARTUP.COM
$ if f$search("''file'") .nes. "" then @'file'
```

To stop LibMariaDB at system shutdown, add the following lines to SYS\$MANAGER:SYSHUTDOWN.COM:

```
$ file := SYS$STARTUP:LIBMARIADB$SHUTDOWN.COM
$ if f$search("''file'") .nes. "" then @'file'
```

6.1. Post-Installation Steps

After the installation has successfully completed, include the commands displayed at the end of the installation procedure into SYSTARTUP_VMS.COM to ensure that the logical names required in order to use the software are defined system-wide at start-up.

In addition to the logical name LIBMARIADB\$ROOT (which points to the root of the LibMariaDB installation tree), the logical name LIBMARIADB\$SHR is also defined. This logical name points to the shareable image LIBMARIADB\$ROOT:[LIB]LIBMARIADB\$SHR.EXE, which can be linked with application code. Alternatively, it is possible to statically link application code with the object library LIBMARIADB\$ROOT:[LIB]LIBMARIADB.OLB. See LIBMARIADB\$ROOT:[EXAMPLES]EXAMPLES.COM for examples of both linking methods.

6.2. Privileges and Quotas

Generally speaking, there are no special quota or privilege requirements for applications developed using LibMariaDB, although a high BYTLM is recommended. The following quotas should be more than adequate for most purposes:

Maxjobs:	0	Fillm:	256	Bytln:	128000
Maxacctjobs:	0	Shrfillm:	0	Pbytln:	0
Maxdetach:	0	BIOLm:	150	JTquota:	4096
Prclm:	50	DIOlm:	150	WSdef:	4096
Prio:	4	ASTlm:	300	WSquo:	8192
Queprio:	4	TQElm:	100	WSextent:	16384
CPU:	(none)	Enqlm:	4000	Pgflquo:	256000

7. Sample Applications

The directory LIBMARIADB\$ROOT:[EXAMPLES] contains a number of example programs that illustrate basic usage of the API with C, COBOL, and FORTRAN. These examples can be compiled and linked using the provided build procedure (EXAMPLES.COM).

Some of the C examples are adapted with permission from example code created by Neil Rieck.

Additional example code may be found on the MariaDB web site (<https://mariadb.com/kb/en/mariadb/mariadb-connector-c/>).

From a development perspective, it should be noted that symbols in the shareable image and object library for the C API are mixed-case, and application developers wishing to utilize the C API must therefore use the C/C++ compiler option **/NAMES=(AS_IS,SHORTENED)** or include in their code appropriate `#pragma` directives to ensure that symbols are correctly resolved when

linking. Developers will also need to include in their code one or more of the header files found in `LIBMARIADB$ROOT:[INCLUDE]`. All language-agnostic API function and constant names are uppercase.

It should also be noted, that the C API can be used only with 32-bit pointers. This consideration is particularly important when using the API with C++ on VSI OpenVMS x86-64 for which 64-bit pointers are used by default.

8. What's missing?

As noted previously, this port of the MariaDB C client API for OpenVMS does not include some functionality. Specifically, Connector/C 3.0 functionality is not currently provided; it is hoped that these matters will be addressed in future releases of the software.

Additionally, the following limitations should be noted:

- Conversion of double precision values to strings with prepared SQL statements is not supported in this release.
- The language-agnostic API is incomplete and does not currently implement all functionality supported by the C API.