MySQL 8.0 Reference Manual Including MySQL NDB Cluster 8.0

|  |  |  |
| --- | --- | --- |
| author | liangdu |  |
| email | [liangdu1992@gmail.com](mailto:liangdu1992@gmail.com) |  |
| version | v1.0 |  |
| date | 2021-04-21 |  |

**摘要**

这是MySQL™参考手册。它分别记录了MySQL 8.0至8.0.25，以及基于NDB 8.0至8.0.23-ndb-8.0.23版本的NDB Cluster版本。它可能包括尚未发布的MySQL版本的功能文档。有关哪些版本已经发布的信息，请参见MySQL 8.0发布说明。

MySQL 8.0的特性。本手册描述的功能并不包括在MySQL 8.0的每个版本中；这些功能可能不包括在授权给你的MySQL 8.0版本中。如果你对包括在你的MySQL 8.0版本中的功能有任何疑问，请参阅你的MySQL 8.0许可协议或联系你的Oracle销售代表。

有关详细说明每个版本中的变化的注释，请参见MySQL 8.0版本注释。

有关法律信息，包括许可信息，见前言和法律声明。

有关使用MySQL的帮助，请访问MySQL论坛，在那里你可以与其他MySQL用户讨论你的问题。

文件产生于：2021-01-20（修订：68545）。

通过www.DeepL.com/Translator（免费版）翻译

**前言和法律声明**

本手册是MySQL数据库系统8.0版的参考手册，直至8.0.25版。MySQL 8.0各次要版本之间的差异在本文本中以发布号（8.0.x）为参考进行说明。关于许可证信息，请参见法律声明。

由于MySQL 8.0与以前的版本在功能和其他方面存在许多差异，本手册不打算用于旧版本的MySQL软件。如果你使用的是MySQL软件的早期版本，请参考相应的手册。例如，MySQL 5.7参考手册涵盖了5.7系列的MySQL软件版本。

许可信息-MySQL 8.0。本产品可能包括第三方软件，在许可证下使用。如果你使用的是MySQL 8.0的商业版本，请参阅《MySQL 8.0商业版本许可信息用户手册》以了解许可信息，包括与可能包含在该商业版本中的第三方软件有关的许可信息。如果你使用的是MySQL 8.0的社区版，请参阅《MySQL 8.0社区版许可信息用户手册》以了解许可信息，包括与可能包含在该社区版中的第三方软件有关的许可信息。

许可信息-MySQL NDB Cluster 8.0。如果你使用的是MySQL NDB Cluster 8.0的社区版本，请参阅《MySQL NDB Cluster 8.0社区版本许可信息用户手册》了解许可信息，包括与可能包含在该社区版本中的第三方软件有关的许可信息。

**法律声明**

Copyright © 1997, 2021, Oracle和/或其附属机构。

本软件和相关文档是根据包含使用和披露限制的许可协议提供的，受到知识产权法的保护。除非你的许可协议明确允许或法律允许，否则你不得以任何形式或手段使用、拷贝、复制、翻译、广播、修改、许可、传输、分发、展览、执行、出版或展示任何部分。禁止对本软件进行逆向工程、反汇编或反编译，除非法律要求具有互操作性。

此处包含的信息如有变化，恕不另行通知，也不保证没有错误。如果你发现任何错误，请以书面形式向我们报告。

如果这是交付给美国政府或代表美国政府授权的任何人的软件或相关文件，那么以下通知是适用的。

美国政府终端用户。甲骨文程序（包括任何操作系统、集成软件、任何嵌入、安装或激活在交付的硬件上的程序，以及对这些程序的修改）和甲骨文计算机文档或其他交付给美国政府最终用户或由其访问的甲骨文数据，根据适用的联邦采购条例和机构特定的补充条例，属于 "商业计算机软件 "或 "商业计算机软件文档"。因此，对i)甲骨文程序（包括任何操作系统、集成软件、任何嵌入、安装或激活在交付的硬件上的程序以及这些程序的修改）、ii)甲骨文计算机文档和/或iii)其他甲骨文数据的使用、复制、发行、展示、披露、修改、准备衍生作品和/或改编，都要遵守适用合同中所载的许可规定的权利和限制。管理美国政府使用甲骨文云服务的条款由此类服务的适用合同规定。没有授予美国政府任何其他权利。

该软件或硬件是为在各种信息管理应用中的一般用途而开发的。它不是为任何固有的危险应用而开发或打算使用的，包括可能产生人身伤害风险的应用。如果你在危险的应用中使用本软件或硬件，那么你应负责采取所有适当的防故障、备份、冗余和其他措施来确保其安全使用。甲骨文公司及其附属机构对在危险应用中使用本软件或硬件造成的任何损失不承担任何责任。

Oracle和Java是Oracle和/或其附属机构的注册商标。其他名称可能是其各自所有者的商标。

Intel和Intel Inside是Intel公司的商标或注册商标。所有SPARC商标都是在授权下使用的，是SPARC国际公司的商标或注册商标。AMD、Epyc和AMD标志是Advanced Micro Devices的商标或注册商标。UNIX是The Open Group的注册商标。

本软件或硬件和文档可能提供对第三方内容、产品和服务的访问或相关信息。甲骨文公司及其附属机构对第三方的内容、产品和服务不负责任，并明确拒绝提供任何形式的保证，除非您与甲骨文公司之间的适用协议中另有规定。甲骨文公司及其附属机构不对因您访问或使用第三方内容、产品或服务而产生的任何损失、费用或损害负责，除非您与甲骨文公司之间的适用协议中有规定。

本文档不是根据GPL许可证分发的。使用本文档须遵守以下条款。

你可以为你自己的个人使用而创建本文档的打印副本。只要不以任何方式改变或编辑实际内容，就允许转换为其他格式。您不得以任何形式或在任何媒体上发表或传播本文档，除非您以类似于甲骨文公司传播文档的方式（即在网站上与软件一起以电子方式下载）或在CD-ROM或类似的媒介上传播，但前提是该文档是与软件一起在同一媒介上传播的。任何其他用途，如任何印刷品的传播或在其他出版物中使用本文档的全部或部分内容，都需要事先获得甲骨文公司授权代表的书面同意。甲骨文公司和/或其附属机构保留对本文档的任何和所有权利，而不是上面明确授予的。

**文件的可访问性**

有关甲骨文公司对可访问性的承诺的信息，请访问甲骨文公司可访问性计划网站：https://www.oracle.com/corporate/accessibility/ 。

**获得甲骨文支持**

购买了支持的甲骨文客户可以通过我的甲骨文支持获得电子支持。有关信息，请访问https://www.oracle.com/corporate/accessibility/learning-support.html#support-tab

目录

[第1章 概论 14](#_Toc70523050)

[1.1关于本手册 16](#_Toc70523051)

[1.2 MySQL数据库管理系统概览 21](#_Toc70523052)

[1.2.1 什么是MySQL 21](#_Toc70523053)

[1.2.2 MySQL的主要特性 25](#_Toc70523054)

[1.2.3 MySQL的历史 31](#_Toc70523055)

[1.3 MySQL 8.0中的新内容 32](#_Toc70523056)

[1.4 MySQL 8.0中增加、废弃或删除的服务器和状态变量及选项 113](#_Toc70523057)

[1.5 MySQL的信息来源 155](#_Toc70523058)

[1.6 如何报告缺陷或问题 157](#_Toc70523059)

[1.7 遵守MySQL标准 167](#_Toc70523060)

[1.7.1 MySQL对标准SQL的扩展 169](#_Toc70523061)

[1.7.2 MySQL与标准SQL的区别 175](#_Toc70523062)

[1.7.3 MySQL如何处理约束条件 180](#_Toc70523063)

[1.8 贡献者 184](#_Toc70523064)

[1.8.1 MySQL的贡献者 184](#_Toc70523065)

[1.8.2 文档作者和翻译者 189](#_Toc70523066)

[1.8.3 支持MySQL的软件包 191](#_Toc70523067)

[1.8.4 用于创建MySQL的工具 192](#_Toc70523068)

[1.8.5 MySQL的支持者 193](#_Toc70523069)

[第2章 安装或升级MySQL 194](#_Toc70523070)

[2.1 一般安装指南 197](#_Toc70523071)

[2.1.1 支持的平台 197](#_Toc70523072)

[2.1.2 安装哪个MySQL版本和发行版 197](#_Toc70523073)

[2.1.3 如何获得MySQL 200](#_Toc70523074)

[2.1.4 使用MD5校验和或GnuPG验证软件包的完整性 200](#_Toc70523075)

[2.1.5 安装布局 221](#_Toc70523076)

[2.1.6 特定于编译器的构建特征 221](#_Toc70523077)

[2.2 使用通用二进制文件在Unix/Linux上安装MySQL 222](#_Toc70523078)

[2.3 在Microsoft Windows上安装MySQL 227](#_Toc70523079)

[2.3.1 Microsoft Windows上的MySQL安装布局 232](#_Toc70523080)

[2.3.2 选择一个安装包 233](#_Toc70523081)

[2.3.3 用于Windows的MySQL安装程序 236](#_Toc70523082)

[2.3.4 使用noinstall ZIP档案在Microsoft Windows上安装MySQL 278](#_Toc70523083)

[2.3.5 Windows系统MySQL服务器安装的故障排除 294](#_Toc70523084)

[2.3.6 Windows安装后程序 297](#_Toc70523085)

[2.3.7 Windows平台的限制 301](#_Toc70523086)

[2.4 在macOS上安装MySQL 303](#_Toc70523087)

[2.4.1 在macOS上安装MySQL的一般注意事项 304](#_Toc70523088)

[2.4.2 使用本地软件包在macOS上安装MySQL 306](#_Toc70523089)

[2.4.3 安装和使用MySQL启动守护程序 312](#_Toc70523090)

[2.4.4 安装和使用MySQL首选项面板 318](#_Toc70523091)

[2.5 在Linux上安装MySQL 322](#_Toc70523092)

[2.5.1 使用MySQL Yum资源库在Linux上安装MySQL 324](#_Toc70523093)

[2.5.2 使用MySQL APT资源库在Linux上安装MySQL 331](#_Toc70523094)

[2.5.3 使用MySQL SLES资源库在Linux上安装MySQL 332](#_Toc70523095)

[2.5.4 使用Oracle的RPM包在Linux上安装MySQL 332](#_Toc70523096)

[2.5.5 使用Oracle的Debian软件包在Linux上安装MySQL 340](#_Toc70523097)

[2.5.6 用Docker在Linux上部署MySQL 342](#_Toc70523098)

[2.6 使用Unbreakable Linux Network（ULN）安装MySQL 380](#_Toc70523099)

[2.7 在Solaris上安装MySQL 382](#_Toc70523100)

[2.7.1 使用Solaris PKG在Solaris上安装MySQL 384](#_Toc70523101)

[2.8 在FreeBSD上安装MySQL 386](#_Toc70523102)

[2.9 从源代码安装MySQL 387](#_Toc70523103)

[2.9.1 源码安装方法 388](#_Toc70523104)

[2.9.2 源码安装的先决条件 389](#_Toc70523105)

[2.9.3 源码安装的MySQL布局 392](#_Toc70523106)

[2.9.4 使用标准源发布版安装MySQL 392](#_Toc70523107)

[2.9.5 使用开发源码树安装MySQL 399](#_Toc70523108)

[2.9.6 配置SSL库支持 402](#_Toc70523109)

[2.9.7 MySQL源配置选项 404](#_Toc70523110)

[2.10 安装后的设置和测试 457](#_Toc70523111)

[2.10.1 初始化数据目录 458](#_Toc70523112)

[2.10.2 启动服务器 468](#_Toc70523113)

[2.11 升级MySQL 483](#_Toc70523114)

[2.11.7 使用MySQL Yum仓库升级MySQL 543](#_Toc70523115)

[2.11.12 升级故障排除 550](#_Toc70523116)

[2.12 降级MySQL 557](#_Toc70523117)

[2.13 Perl安装注意事项 557](#_Toc70523118)

[2.13.1 在Unix上安装Perl 557](#_Toc70523119)

[2.13.2 在Windows上安装ActiveState Perl 559](#_Toc70523120)

[2.13.3 使用Perl DBI/DBD接口的问题 560](#_Toc70523121)

[第3章 指南 562](#_Toc70523122)

[第4章 MySQL程序 562](#_Toc70523123)

[第5章 MySQL服务管理 563](#_Toc70523124)

[第6章 安全 563](#_Toc70523125)

[第7章 备份与恢复 563](#_Toc70523126)

[第8章 优化 563](#_Toc70523127)

[第9章 语言结构 563](#_Toc70523128)

[第10章 字符集、对齐方式、统一编码 563](#_Toc70523129)

[第11章 数据类型 563](#_Toc70523130)

[第12章 函数和operator 563](#_Toc70523131)

[第13章 SQL表达式 563](#_Toc70523132)

[第14章 MySQL数据目录 564](#_Toc70523133)

[14.1 数据字典模式 565](#_Toc70523134)

[14.2 移除基于文件的元数据存储 567](#_Toc70523135)

[14.3 词典数据的事务性存储 568](#_Toc70523136)

[14.4 字典对象缓存 569](#_Toc70523137)

[14.5 INFORMATION\_SCHEMA和数据字典集成 570](#_Toc70523138)

[14.6 串行化字典信息（SDI 573](#_Toc70523139)

[14.7 数据字典使用方面的差异 574](#_Toc70523140)

[14.8 数据字典的限制 578](#_Toc70523141)

[第15章 InnoDB存储引擎 579](#_Toc70523142)

[第16章 其他存储引擎 579](#_Toc70523143)

[第17章 复制 579](#_Toc70523144)

[第18章 分组复制 579](#_Toc70523145)

[第19章 MySQL SHELL 580](#_Toc70523146)

[第20章 MySQL文档存储 582](#_Toc70523147)

[第21章 InnoDB 集群 582](#_Toc70523148)

[第22章 InnoDB 副本集 586](#_Toc70523149)

[第23章 MySQL NDB 集群 8.0 589](#_Toc70523150)

[第24章 分区 589](#_Toc70523151)

[第25章 存储对象 589](#_Toc70523152)

[第26章 INFORMATION 表 589](#_Toc70523153)

[第27章 MySQL性能schema 589](#_Toc70523154)

[第28章 MySQL sys schema 589](#_Toc70523155)

[第29章 连接器与API 590](#_Toc70523156)

[29.1 MySQL Connector/C++ 596](#_Toc70523157)

[29.2 MySQL Connector/J 596](#_Toc70523158)

[29.3 MySQL Connector/NET 596](#_Toc70523159)

[29.4 MySQL Connector/ODBC 596](#_Toc70523160)

[29.5 MySQL Connector/Python 597](#_Toc70523161)

[29.6 MySQL Connector/Node.js 597](#_Toc70523162)

[29.7 MySQL C API 597](#_Toc70523163)

[29.8 MySQL的PHP API 597](#_Toc70523164)

[29.9 MySQL的Perl API 597](#_Toc70523165)

[29.10 MySQL Python API 599](#_Toc70523166)

[29.11 MySQL Ruby APIs 599](#_Toc70523167)

[29.11.1 MySQL/Ruby API 600](#_Toc70523168)

[29.11.2 Ruby/MySQL API 600](#_Toc70523169)

[29.12 MySQL Tcl API 600](#_Toc70523170)

[29.13 MySQL Eiffel Wrapper 600](#_Toc70523171)

[第30章 MySQL企业版 601](#_Toc70523172)

[30.1 MySQL企业监控器概述 601](#_Toc70523173)

[30.2 MySQL企业备份概述 604](#_Toc70523174)

[30.3 MySQL企业版安全概述 605](#_Toc70523175)

[30.4 MySQL企业版加密概述 606](#_Toc70523176)

[30.5 MySQL企业审计概述 607](#_Toc70523177)

[30.6 MySQL企业防火墙概述 607](#_Toc70523178)

[30.7 MySQL企业线程池概述 608](#_Toc70523179)

[30.8 MySQL企业版数据屏蔽和取消识别概述 608](#_Toc70523180)

[第31章 MySQL WORKBENCH 610](#_Toc70523181)

[第32章 MySQL OCI 市场 612](#_Toc70523182)

[32.1 在Oracle云基础设施上部署MySQL EE的先决条件 612](#_Toc70523183)

[32.2 在Oracle云基础设施上部署MySQL EE 613](#_Toc70523184)

[32.3 配置网络访问 616](#_Toc70523185)

[32.4 连接 616](#_Toc70523186)

[32.5 维护 617](#_Toc70523187)

[附录A 619](#_Toc70523188)

[附录B 619](#_Toc70523189)

[附录C 619](#_Toc70523190)

[MySQL词汇表 619](#_Toc70523191)

# 概论

MySQL™软件提供了一个非常快速、多线程、多用户和强大的SQL（结构化查询语言）数据库服务器。MySQL服务器的目的是用于关键任务、重负荷的生产系统以及嵌入大规模部署的软件。MySQL是甲骨文公司和/或其附属机构的商标。或其附属公司的商标，未经甲骨文公司明确的书面授权，客户不得使用。其他名称可能是其各自所有者的商标.

MySQL软件是双授权的。用户可以选择将MySQL软件作为开放源码产品在GNU通用公共许可证的条款下使用。用户可以选择根据GNU通用公共许可证（http://www.fsf.org/licenses/）的条款将MySQL软件作为开源产品使用，或者从甲骨文公司购买标准的商业许可证。可以从Oracle购买一个标准的商业许可。参见<http://www.mysql.com/company/legal/> 许可/了解更多关于我们许可政策的信息。

以下列表描述了本手册中一些特别值得关注的章节。

* 关于MySQL数据库服务器功能的讨论，见第1.2.2节 "MySQL的主要特性"。
* 关于MySQL新特性的概述，见第1.3节 "MySQL 8.0的新内容"。关于每个版本的关于每个版本的变化信息，请参见发行说明。
* 关于安装说明，见第2章，安装和升级MySQL。关于升级MySQL的信息，见第2.11节 "升级MySQL"。
* 关于MySQL数据库服务器的教程介绍，见第3章，教程。
* 有关配置和管理MySQL服务器的信息，见第5章，MySQL服务器管理。
* 关于MySQL的安全信息，见第6章，安全。
* 有关设置复制服务器的信息，见第17章，复制。
* 关于MySQL企业版的信息，这是一个具有高级功能和管理工具的商业MySQL版本，见第30章。
* 有关MySQL企业版的信息，请参见第30章，MySQL企业版
* 关于经常被问到的有关MySQL数据库服务器及其功能的一些问题的答案，见附录A，MySQL 8.0常见问题。及其功能的问题，请参见附录A，MySQL 8.0常见问题。
* 有关新功能和错误修复的历史，请参见发行说明。

注意事项：

要报告问题或错误，请使用第1.6节的说明。"如何报告错误或问题"。如果你发现MySQL服务器中的安全漏洞，请立即让我们知道，请发送电子邮件到<secalert\_us@oracle.com>。例外。支持客户应报告所有问题，包括安全漏洞，都应报告给Oracle支持。

## 1.1关于本手册

这是MySQL数据库系统8.0-8.0.26版的参考手册。

MySQL 8.0各次要版本之间的差异在本文中以发布号（8.0.x）为参照进行说明。版本号（8.0.x）。关于许可证信息，请参见法律声明。

本手册不打算用于MySQL软件的旧版本，因为MySQL 8.0与MySQL 8.0之间存在许多功能和其他方面的差异。

由于MySQL 8.0与以前的版本在功能和其他方面有许多不同之处，本手册不打算用于旧版本的MySQL软件。如果你正在使用较早的

如果你使用的是MySQL软件的早期版本，请参考相应的手册。例如，MySQL 5.7参考手册涵盖了5.7系列的MySQL软件版本。

由于本手册作为参考，它并不提供关于SQL或关系数据库概念的一般指导。数据库的概念。它也不教你如何使用你的操作系统或命令行解释器。

MySQL数据库软件正在不断开发中，参考手册也经常更新。也经常更新。该手册的最新版本可以在网上以可搜索的形式获得，网址是https://dev.mysql.com/doc/。其他格式也可在那里找到，包括可下载的HTML和PDF版本。

MySQL的源代码本身包含使用Doxygen编写的内部文档。生成的生成的Doxygen内容可在https://dev.mysql.com/doc/index-other.html。它也可以也可以使用第2.9.10节 "生成MySQL Doxygen文档内容 "中的说明，从MySQL源分发中本地生成这些内容。第2.9.10节，"生成MySQL Doxygen文档内容"。

如果你有关于使用MySQL的问题，请加入MySQL社区Slack，或在我们的论坛上提问；见MySQL论坛上的MySQL社区支持。如果你有关于添加或如果你有关于手册本身的补充或修正的建议，请将它们发送到<http://www.mysql.com/company/contact/>。

**排版和语法惯例**

本手册使用了某些排版惯例

* 这种风格的文本用于SQL语句；数据库、表和列名；程序清单和源代码；以及环境变量。列表和源代码；以及环境变量。例如。"要重新加载授予表，请使用FLUSH PRIVILEGES语句"。
* 这种风格的文本表示你在例子中输入的内容**。**
* 这种风格的文本表示可执行程序和脚本的名称，例子有mysql（MySQL命令行客户端程序）和mysqld（MySQL服务器可执行程序）**。**
* 这种风格的文本用于变量输入，你应该用一个你自己选择的值来代替它。自己选择的值**。**
* 这种风格的文本用于强调**。** 这种风格的文本用于表的标题和传达特别强烈的强调**。这种风格的文本用于表示一个影响程序执行方式的程序选项执行，或提供程序以某种方式运行所需的信息。例如。"--host选项（简称-h）告诉mysql客户端程序它应该连接的MySQL服务器的主机名或IP地址它应该连接到的MySQL服务器的主机名或IP地址"。**
* **文件名和目录名是这样写的。"全局的my.cnf文件位于/etc目录中。"**
* **字符序列是这样写的。"要指定一个通配符，请使用'%'字符"**

当显示的命令是要在一个特定的程**序中执行时，**在命令前面显示的提示符表明要使用哪个命令。命令前显示的提示符表明要使用哪条命令。例如，shell>表示你从登录的shell中执行的命令，root-shell>类似，但应以root身份执行。mysql>表示从mysql客户端程序中执行的语句

shell> **type a shell command here**

root-shell> **type a shell command as *root* here**

mysql> **type a mysql statement here**

在一些地区，不同的系统可能会被区分开来，以表明命令应该在两个不同的环境中执行。在两个不同的环境中执行。例如，在复制工作中，命令可能以源和复制为前缀。

source> **type a mysql command on the replication source here**

replica> **type a mysql command on the replica here**

shell "是你的命令解释器。在Unix中，这通常是一个程序，如sh、csh或bash。在Windows中，相应的程序是command.com或cmd.exe，通常在一个控制台窗口中运行。

当你输入一个例子中的命令或语句时，不要输入例子中的提示符。例子中显示的提示。

数据库、表和列的名称经常必须被替换成语句。为了表示这种，本手册使用db\_name、tbl\_name和col\_name来表示这种替换是必要的。例如，你可能会看到这样的语句。

mysql> **SELECT *col\_name* FROM *db\_name*.*tbl\_name*;**

这意味着，如果你要输入一个类似的语句，你将提供你自己的数据库、表。和列名，也许像这样。

mysql> **SELECT author\_name FROM biblio\_db.author\_list;**

SQL关键字不区分大小写，可以用任何字母大小写来书写。本手册使用大写字母。

在语法描述中，方括号（"["和"]"）表示可选的词或子句。例如，在下面的语句中，IF EXISTS是可选的。

DROP TABLE [IF EXISTS] ***tbl\_name***

当一个语法元素由多个备选方案组成时，这些备选方案由竖条（"|"）分隔。条（"|"）分隔。当可以从一组选项中选择一个成员时，这些选项被列在方括号（"["和"]"）。

TRIM([[BOTH | LEADING | TRAILING] [***remstr***] FROM] ***str***)

当必须从一组选择中选择一个成员时，备选方案被列在大括号内（"{"和"}"）。)

{DESCRIBE | DESC} ***tbl\_name*** [***col\_name*** | ***wild***]

省略号(....)表示省略了语句中的某一节，通常是为了提供更简短的，通常是为更复杂的语法提供一个简短的版本。例如，SELECT ... INTO OUTFILE是SELECT语句的速记形式，它在其他部分之后有一个INTO OUTFILE子句。语句的其他部分之后有一个INTO OUTFILE子句的SELECT语句的简称。

省略号也可以表示语句中前面的句法元素可以重复。在下面的例子中，可以给出多个reset\_option值，在第一个值之后的每一个值都用逗号表示。前面有逗号。

RESET ***reset\_option*** [,***reset\_option***

设置shell变量的命令使用Bourne shell的语法来显示。例如，在Bourne shell中，设置CC环境变量和运行configure命令的顺序是这样的来设置CC环境变量和运行configure命令，在Bourne shell中是这样的语法。

shell> **CC=gcc ./configure**

如果你使用的是 csh 或 tcsh，你必须以某种不同的方式发布命令。

shell> **setenv CC gcc**

shell> **./configure**

**手册撰写说明**

参考手册的源文件是以DocBook XML格式编写的。HTML版本和其他格式自动生成，主要使用DocBook XSL样式表。关于DocBook的信息有关DocBook的信息，请参见http://docbook.org/

本手册最初由David Axmark和Michael "Monty" Widenius编写。它由MySQL文档团队维护。它由MySQL文档团队维护，该团队由Chris Cole、Paul DuBois、Margaret Fisher、Edward，Gilmore, Stefan Hinz, David Moss, Philip Olson, Daniel Price, Daniel So, and Jon Stephens.

## 1.2 MySQL数据库管理系统概览

### 1.2.1 MySQL是什么？

MySQL是最受欢迎的开源SQL数据库管理系统，由甲骨文公司开发、发布和支持。

MySQL网站（http://www.mysql.com/）提供关于MySQL软件的最新信息。

* MySQL是数据库管理系统

数据库是一个结构化的数据集合。数据集可以是任何东西，从简单的购物清单到一个图片库或一个公司网络中的大量信息。要添加、访问和处理存储在计算机数据库中的数据，你需要一个数据库管理系统，例如MySQL服务器这样的数据库管理系统。由于计算机非常善于处理大量的数据，所以数据库管理系统在计算机领域发挥着核心作用。数据库管理系统在计算中发挥着核心作用，作为独立的实用程序，或作为其他应用程序的一部分。

* MySQL数据库是关系型的

关系型数据库将数据存储在不同的表中，而不是将所有的数据放在一个大的存储空间。数据库结构被组织成物理文件，并为速度而优化。逻辑模型逻辑模型，包括数据库、表、视图、行和列等对象，提供了一个灵活的编程环境。你可以设置不同数据之间的关系的规则字段之间的关系，如一对一、一对多、唯一、必需或可选，以及不同表之间的 "指针"。不同表之间的 "指针"。数据库强制执行这些规则，因此，在一个设计良好的数据库中，你的应用程序永远不会看到不一致的、重复的和不完整的数据。

MySQL "的SQL部分代表 "结构化查询语言"。SQL是最常见的用于访问数据库的标准化语言。取决于你的编程环境。你可以直接输入SQL（例如，生成报告），将SQL语句嵌入到用另一种语言编写的代码中，或使用一种特定语言的代码。用另一种语言编写的代码中，或者使用隐藏SQL语法的特定语言API。

SQL是由ANSI/ISO SQL标准定义的。自1986年以来，SQL标准一直在不断发展并有多个版本。在本手册中，"SQL-92 "是指1992年发布的标准。"SQL:1999 "是指1999年发布的标准，而"SQL:2003 "是指当前版本的标准。是指该标准的当前版本。我们使用 "SQL标准 "这个短语来指当前版本的SQL标准在任何时候

* MySQL是开源的

开源意味着任何人都有可能使用和修改该软件。任何人都可以从互联网上下载MySQL软件并使用它，而无需支付任何费用。如果你愿意，你可以研究源代码并根据自己的需要进行修改。MySQL软件使用GPL(GNU General Public License)http://www.fsf.org/licenses/，来定义你在不同情况下可以和不可以做什么在不同情况下可以和不可以做什么。如果你对GPL感到不舒服，或者需要将MySQL代码嵌入到一个商业应用中，你可以将MySQL代码嵌入到一个商业应用中。MySQL代码嵌入商业应用，你可以向我们购买商业许可版本。更多信息见MySQL许可概述(<http://www.mysql.com/company/legal/licensing/> )

* MySQL具有高效、可靠、弹性和易用的特性

如果这是你正在寻找的，你应该试一试。MySQL服务器可以舒适地运行在台式机或笔记本电脑上，与你的其他应用程序、网络服务器等一起运行，只需要很少或不需要注意。如果你把一整台机器用于MySQL，你可以调整设置，以利用所有的内存、CPU功率、以及其他的功能。所有可用的内存、CPU功率和I/O容量。MySQL还可以扩展到联网的机器集群。联网的机器集群。

MySQL服务器最初是为了处理大型数据库而开发的，比现有的解决方案快得多。几年来，它已被成功地用于高要求的生产环境中。几年来一直成功地用于高要求的生产环境。尽管在不断的发展中，MySQL服务器今天提供了丰富和有用的功能集。功能。它的连接性、速度和安全性使MySQL服务器非常适用于访问互联网上的数据库。

* MySQL可以是客户端/服务器架构也可以是嵌入式架构

MySQL数据库软件是一个客户/服务器系统，由一个多线程的SQL服务器组成。支持不同的后端，几个不同的客户端程序和库，管理工具。以及广泛的应用编程接口（API）。

我们还提供MySQL服务器作为一个嵌入式的多线程库，你可以将其链接到你的以获得更小、更快、更易于管理的独立产品。

* 有大量贡献的MySQL软件可用。

MySQL服务器有一套与我们的用户密切合作开发的实用功能。它是很可能你最喜欢的应用程序或语言支持MySQL数据库服务器。

MySQL的官方发音是 "My Ess Que Ell"（不是 "my sequel"），但我们并不介意你发音为 "my sequel "或其他一些本地化的方式。

### 1.2.2 MySQL的主要特性

本节介绍了MySQL数据库软件的一些重要特征。在大多数方面，该路线图适用于MySQL的所有版本。有关在特定系列基础上引入MySQL的特性的信息关于在特定系列基础上引入MySQL的特性，请参见相应的《手册》中的 "简明扼要 "部分。

* MySQL 8.0: Section 1.3, “What Is New in MySQL 8.0”
* MySQL 5.7: What Is New in MySQL 5.7
* MySQL 5.6: What Is New in MySQL 5.6

**内部和可移植性**

* 用C和C++编写。
* 用广泛的不同的编译器进行测试。
* 在许多不同的平台上工作。见https://www.mysql.com/support/supportedplatforms/数据库.html。
* 为了便于移植，使用CMake进行配置。
* 用Purify（一个商业的内存泄漏检测器）以及Valgrind（一个GPL工具）进行测试。(<http://developer.kde.org/~sewardj/>)。
* 使用具有独立模块的多层服务器设计。
* 设计成使用内核线程的完全多线程，如果有多个CPU的话，可以很容易地使用它们。可用。
* 提供事务和非事务存储引擎。
* 使用非常快的B树磁盘表（MyISAM），带有索引压缩。旨在使添加其他存储引擎变得相对容易。如果你想为内部数据库提供一个内部数据库的SQL接口。
* 使用一个非常快的基于线程的内存分配系统。
* 使用优化的嵌套循环连接，执行非常快的连接。
* 实施内存哈希表，作为临时表使用。
* 使用高度优化的类库执行SQL函数，其速度应尽可能快。通常，在查询初始化之后，根本没有内存分配。
* 将服务器作为一个单独的程序提供给客户/服务器网络环境使用，并作为一个可以嵌入（链接）的库。一个可以嵌入（链接）到独立应用程序的库。这样的应用程序可以被用于孤立地使用，或在没有网络的环境中使用。

**数据类型**

* 许多数据类型：1、2、3、4和8字节长的有符号/无符号整数、FLOAT、DOUBLE、CHAR。VARCHAR, BINARY, VARBINARY, TEXT, BLOB, DATE, TIME, DATETIME, TIMESTAMP, YEAR, SET,ENUM，以及OpenGIS空间类型。参见第11章，数据类型。
* 固定长度和可变长度的字符串类型。

**SQL语句和函数**

* 在查询的SELECT列表和WHERE子句中支持全部操作符和函数。举例来说

mysql> **SELECT CONCAT(first\_name, ' ', last\_name)**

-> **FROM citizen**

-> **WHERE income/dependents > 10000 AND age > 30;**

* 完全支持SQL GROUP BY和ORDER BY条款。支持分组函数（COUNT(),AVG(), STD(), SUM(), MAX(), MIN(), 和 GROUP\_CONCAT()
* 支持标准SQL和ODBC的LEFT OUTER JOIN和RIGHT OUTER JOIN。语法。
* 支持标准SQL所要求的对表和列的别名。
* 支持DELETE, INSERT, REPLACE, 和UPDATE，以返回被改变（影响）的行数。改变（影响）的行数，或者通过在连接服务器时设置一个标志来返回匹配的行数。连接到服务器时，通过设置一个标志来返回匹配的行数。
* 支持MySQL特定的SHOW语句，以检索关于数据库、存储引擎、表和索引的信息。检索数据库、存储引擎、表和索引的信息。支持INFORMATION\_SCHEMA数据库，根据标准SQL实现
* EXPLAIN语句，显示优化器如何解决一个查询。
* 函数名与表或列名的独立性。例如，ABS是一个有效的列名。唯一的限制是，对于一个函数调用，在函数名和后面的"("之间不允许有空格。名称和后面的"("之间不能有空格。参见第9.3节，"关键词和保留字"。
* 你可以在同一语句中引用不同数据库的表

**安全**

* **一个非常灵活和安全的权限和密码系统，并能实现基于主机的验证。**
* **当你连接到服务器时，通过对所有密码流量进行加密来保证密码安全。**

**可扩展性和局限性**

* **对大型数据库的支持。我们使用MySQL服务器的数据库包含5000万条记录。我们也知道有些用户使用MySQL服务器，其中有20万个表和大约5,000,000,000行。**
* **每张表最多支持64个索引。每个索引可以由1到16列或部分列组成。InnoDB表的最大索引宽度是767字节或3072字节。参见第15.22节，"InnoDB的限制"。MyISAM表的最大索引宽度是1000字节。参见第16.2节 "MyISAM存储引擎"。索引可以使用CHAR、VARCHAR、BLOB或者TEXT列类型的列的前缀。**

**连接性**

* **客户端可以使用几种协议连接到MySQL服务器**
  + **客户端可以在任何平台上使用TCP/IP套接字进行连接。**
  + **在Windows系统上，如果服务器在启动时启用了named\_pipe系统变量，客户可以使用命名的管道进行连接。如果启动时启用shared\_memory系统变量，Windows服务器也支持共享内存连接。客户端可以通过使用--protocol=memory选项，通过共享内存进行连接。**
  + **在Unix系统上，客户端可以使用Unix域套接字文件进行连接。**
* **MySQL客户端程序可以用许多语言编写。用C语言编写的客户端库可用于用C或C++编写的客户端，或用于任何提供C绑定的语言。**
* **C、C++、Eiffel、Java、Perl、PHP、Python、Ruby和Tcl的API可用，使MySQL客户端可以用许多语言编写。参见第29章，连接器和API。**
* **连接器/ODBC（MyODBC）接口为使用ODBC（开放数据库连接）连接的客户端程序提供MySQL支持。例如，你可以使用MS Access来连接到你的MySQL服务器。客户端可以在Windows或Unix上运行。连接器/ODBC源是可用的。所有的ODBC 2.5功能都支持，还有许多其他功能。参见《MySQL Connector/ODBC开发指南》。**
* **Connector/J接口为使用JDBC连接的Java客户端程序提供MySQL支持。客户端可以在Windows或Unix上运行。Connector/J的源代码是可用的。参见《MySQL Connector/J 5.1开发者指南》。**
* **MySQL Connector/NET使开发人员能够轻松地创建需要与MySQL进行安全、高性能数据连接的.NET应用程序。它实现了所需的ADO.NET接口并集成到ADO.NET感知工具中。开发人员可以使用他们选择的.NET语言构建应用程序。MySQL Connector/NET是一个完全管理的ADO.NET驱动程序，由100%纯C#编写。请参阅MySQL Connector/NET开发者指南。**

**本地化**

* **服务器可以向客户提供多种语言的错误信息。见第10.12节，"设置错误信息语言"。**
* **完全支持几种不同的字符集，包括latin1 (cp1252), german, big5, ujis, 几个Unicode字符集，以及其他。例如，斯堪的纳维亚字符 "å"、"ä "和 "ö "在表和列名中是允许的。**
* **所有数据都保存在所选择的字符集中。**
* **排序和比较是根据默认的字符集和排序方式进行的。当MySQL服务器启动时，可以改变这一点（见第10.3.2节，"服务器字符集和排序方式"）。要看一个非常高级的排序的例子，请看捷克的排序代码。MySQL服务器支持许多不同的字符集，可以在编译时和运行时指定。**
* **服务器时区可以动态改变，个别客户可以指定自己的时区。见第5.1.15节，"MySQL服务器时区支持"。**

**客户端和工具**

* **MySQL包括几个客户端和实用程序。这些程序包括命令行程序，如mysqldump和mysqladmin，以及图形程序，如MySQL Workbench。**
* **MySQL服务器内置了对SQL语句的支持，以检查、优化和修复表。这些语句可以从命令行通过mysqlcheck客户端获得。MySQL还包括myisamchk，这是一个非常快速的命令行工具，用于对MyISAM表执行这些操作。见第4章，MySQL程序。**
* **可以用--help或--?选项调用MySQL程序，以获得在线帮助。**

### **1.2.3 MySQL的历史**

我们一开始打算使用mSQL数据库系统，用我们自己的快速低级（ISAM）例程连接到我们的表。然而，经过一些测试，我们得出结论，mSQL的速度和灵活性都不足以满足我们的需求。这导致了我们的数据库有了一个新的SQL接口，但其API接口几乎与mSQL相同。这个API的设计是为了使为使用mSQL而编写的第三方代码能够容易地移植到MySQL上使用。

MySQL是以联合创始人Monty Widenius的女儿My命名的**。**

MySQL海豚（我们的标志）的名字是 "Sakila"，这是从我们的 "为海豚起名 "比赛中用户建议的大量名字中选出的。获奖的名字是由Ambrose Twebaze提交的，他是来自非洲埃斯瓦蒂尼（原斯威士兰）的开源软件开发者。据Ambrose说，女性的名字Sakila起源于埃斯瓦蒂尼的当地语言SiSwati。萨基拉也是坦桑尼亚阿鲁沙的一个小镇的名字，靠近安布罗斯的原籍国乌干达**。**

## **1.3 MySQL 8.0中的新内容**

本节总结了在MySQL 8.0中增加、废弃和删除的内容。一部分列出了在MySQL 8.0中增加、废弃或删除的MySQL服务器选项和变量；见第1.4节 "MySQL 8.0中增加、废弃或删除的服务器和状态变量及选项**"**

**在MySQL 8.0中增加的功能**

* 数据字典**。 现在，M**ySQL纳入了一个支持事务**的数据字典用于存储有关数据库对象的信息。在以前的MySQL版本中，字典数据被存储在元数据文件和非事务性的表中。更多信息，见第14章，MySQL数据字典。**
* 原子数据定义语句（Atomic DDL）**.原子DDL语句将与DDL操作相关的数据字典更新、存储引擎操作和二进制日志写入合并为一个单一的原子事务。欲了解更多信息，请参见第13.1.1节，"原子数据定义语句支持"。**
* 升级程序**。 以前，在安装新版本的MySQL后，MySQL服务器会在下次启动时自动升级数据字典表，之后DBA要手动调用mysql\_upgrade来升级mysql模式中的系统表，以及其他模式中的对象，如sys模式和用户模式。**

**从MySQL 8.0.16开始，服务器会执行以前由mysql\_upgrade处理的任务。在安装了新的MySQL版本后，服务器现在会在下次启动时自动执行所有必要的升级任务，而不依赖于DBA调用mysql\_upgrade。此外，服务器会更新帮助表的内容（这是mysql\_upgrade没有做到的）。一个新的-upgrade服务器选项提供了对服务器如何执行自动数据字典和服务器升级操作的控制。更多信息，见第2.11.3节 "MySQL升级过程中升级的内容"。**

* 安全和账户管理**。 添加这些增强功能是为了提高安全性，并使DBA在账户管理方面具有更大的灵活性。**
  + **mysql系统数据库中的grant表现在是InnoDB（事务性）表。以前，这些是MyISAM（非事务型）表。grant表存储引擎的变化是对账户管理语句行为的相应变化。以前，一个名为多个用户的账户管理语句（如CREATE USER或DROP USER）可能对一些用户成功，对另一些用户失败。现在，每个语句都是事务性的，要么对所有指定的用户成功，要么在发生任何错误时回滚，没有任何影响。如果语句成功了，就会被写入二进制日志，但如果失败了就不会；在这种情况下，就会发生回滚，不做任何改变。更多信息，请参阅第13.1.1节，"原子数据定义语句支持"。**
  + **一个新的caching\_sha2\_password认证插件已经可用。与sha256\_password插件一样，caching\_sha2\_password实现了SHA-256密码散列，但使用缓存来解决连接时的延迟问题。它还支持更多的传输协议，并且不需要与OpenSSL链接以获得基于RSA密钥对的密码交换功能。参见第6.4.1.2节，"缓存SHA-2可插拔认证"。**

**caching\_sha2\_password和sha256\_password认证插件比mysql\_native\_password插件提供更安全的密码加密，而且caching\_sha2\_password比sha256\_password提供更好的性能。由于caching\_sha2\_password的这些优越的安全和性能特性，它现在是首选的认证插件，也是默认的认证插件，而不是mysql\_native\_password。关于这种默认插件的变化对服务器操作以及服务器与客户端和连接器的兼容性的影响，请参见caching\_sha2\_password作为首选认证插件。**

* + **MySQL现在支持角色，它是命名的权限集合。角色可以被创建和删除。角色可以被授予和撤销权限。角色可以被授予和撤销于用户账户。一个账户的有效适用角色可以从授予该账户的角色中选择，并且可以在该账户的会话中改变。更多信息，见第6.2.10节 "使用角色"。**
  + **MySQL现在包含了用户帐户类别的概念，系统和普通用户根据他们是否有SYSTEM\_USER权限而被区分。见第6.2.11节，"账户类别"。**
  + **以前，除了某些模式外，不可能授予全局性的权限。现在，如果启用了partial\_revokes系统变量，就可以做到这一点。参见第6.2.12节，"使用部分撤销的权限限制"。**
  + **GRANT语句有一个AS user [WITH ROLE]子句，指定了语句执行时要使用的特权上下文的额外信息。这个语法在SQL级别上是可见的，尽管它的主要目的是通过使这些限制出现在二进制日志中，使由部分撤销所施加的授予者特权限制在所有节点上统一复制。见第13.7.1.6节，"GRANT语句"。**
  + **MySQL现在保留了关于密码历史的信息，实现了对以前密码的重复使用的限制。DBA可以要求在一定数量的密码变化或一段时间内，不能从以前的密码中选择新密码。可以在全局以及每个账户的基础上建立密码重复使用策略。**
  + **现在可以要求通过指定要替换的当前密码来验证更改账户密码的尝试。这使DBA能够防止用户在没有证明他们知道当前密码的情况下改变密码。可以在全局和每个账户的基础上建立密码验证策略。**
  + **现在允许账户拥有双重密码，这使得在复杂的多服务器系统中可以无缝地进行分阶段的密码修改，而不需要停机。**
  + **现在，MySQL使管理员能够配置用户账户，从而使由于密码不正确而导致的连续登录失败次数过多会导致账户被临时锁定。每个账户所需的失败次数和锁定时间是可以配置的。**
  + **这些新功能为DBA提供了对密码管理更全面的控制。更多信息请参见第6.2.15节 "密码管理"。**
  + **如果使用OpenSSL进行编译，MySQL现在支持FIPS模式，并且在运行时可以使用OpenSSL库和FIPS对象模块。FIPS模式对加密操作施加了条件，如对可接受的加密算法的限制或对更长的密钥长度的要求。见第6.8节，"FIPS支持"**
  + **现在服务器用于新连接的TLS上下文可以在运行时重新配置。这种能力可能很有用，例如，避免重新启动一个已经运行了很久的MySQL服务器，因为它的SSL证书已经过期。参见服务器端运行时配置和加密连接的监控。**
  + **OpenSSL 1.1.1支持用于加密连接的TLS v1.3协议，MySQL 8.0.16及更高版本也支持TLS v1.3，如果服务器和客户端都使用OpenSSL 1.1.1或更高版本进行编译。参见第6.3.2节，"加密连接TLS协议和密码"。**
  + **MySQL现在将授予命名管道上的客户端的访问控制设置为在Windows上成功通信所需的最小值。较新的MySQL客户端软件可以打开命名管道连接，而无需任何额外的配置。如果旧的客户端软件不能立即升级，新的named\_pipe\_full\_access\_group系统变量可以用来给一个Windows组以必要的权限来打开一个命名的管道连接。完全访问组的成员资格应该是受限制的和临时的。**
* 资源管理。 **MySQL现在支持创建和管理资源组，并允许将服务器内运行的线程分配给特定的组，以便线程根据该组的可用资源执行。组的属性可以控制它的资源，以启用或限制组内线程的资源消耗。DBA可以根据不同的工作负载来修改这些属性。目前，CPU时间是一种可管理的资源，以 "虚拟CPU "的概念为代表，这个术语包括CPU核、超线程、硬件线程等。服务器在启动时确定有多少虚拟CPU可用，具有适当权限的数据库管理员可以将这些CPU与资源组联系起来，并将线程分配给组。欲了解更多信息，请参见第5.1.16节 "资源组"。**
* **表加密管理。 现在可以通过定义和执行加密默认值来全局管理表加密。default\_table\_encryption变量为新创建的模式和一般表空间定义了一个加密默认值。在创建模式时，也可以使用DEFAULT ENCRYPTION子句来定义模式的加密默认值。默认情况下，一个表会继承它所创建的模式或一般表空间的加密。加密默认**值通过启用table\_encryption\_privilege\_check变量来强制执行。当创建或改变具有不同于default\_table\_encryption设置的加密设置的模式或一般表空间时，或者创建或改变具有不同于默认模式加密的加密设置的表时，会发生特权检查。TABLE\_ENCRYPTION\_ADMIN权限允许**在table\_encryption\_privilege\_check被启用时覆盖默认的加密设置。更多信息，请参见为模式和一般表空间定义加密默认值**
* InnoDB的增强功能**。 增加了这些InnoDB的增强功能。**
  + **每次自动增量计数器的值发生变化时，都会将其写入redo log，并在每次检查点时保存到引擎私有的系统表中。这些变化使得当前的最大自动增量计数器的值在服务器重启时是持久的。此外。**
    - **服务器重启不再取消AUTO\_INCREMENT = N表选项的效果。如果你把自动增量计数器初始化为一个特定的值，或者你把自动增量计数器的值改成一个更大的值，新的值会在服务器重新启动时持续存在。**
    - **在ROLLBACK操作之后立即重新启动服务器，不再导致分配给回滚事务的自动增量值的重复使用。**
    - **如果你修改AUTO\_INCREMENT列的值，使其大于当前的最大自动增量值（例如，在UPDATE操作中），新的值会被持久化，随后的INSERT操作会从新的、更大的值开始分配自动增量值。**

**更多信息，请参见第15.6.1.6节，"InnoDB中的AUTO\_INCREMENT处理"，以及InnoDB AUTO\_INCREMENT计数器初始化。**

* **当遇到索引树损坏时，InnoDB会向redo日志写一个损坏标志，这使得损坏标志的崩溃是安全的。InnoDB还在每个检查点上将内存中的corruption标志数据写到引擎私有的系统表中。在恢复过程中，InnoDB从这两个位置读取corruption标志，并在将内存中的表和索引对象标记为corruption之前合并结果。**
* **InnoDB memcached插件支持多个获取操作（在一个memcached查询中获取多个键值对）和范围查询。参见第15.20.4节，"InnoDB memcached多次获取和范围查询支持"。**
* **一个新的动态变量，innodb\_deadlock\_detect，可以用来禁用死锁检测。在高并发系统中，当许多线程等待同一个锁时，死锁检测会导致速度减慢。有时，当死锁发生时，禁用死锁检测并依靠innodb\_lock\_wait\_timeout设置进行事务回滚可能会更有效。**
* **新的INFORMATION\_SCHEMA.INNODB\_CACHED\_INDEXES表报告了每个索引在InnoDB缓冲池中缓存的索引页数量。**
* **InnoDB临时表现在被创建在共享的临时表空间ibtmp1中。**
* **InnoDB表空间加密功能支持对redo log和Undo logs数据进行加密。参见redo log加密，和Undo logs加密。**
* **InnoDB支持NOWAIT和SKIP LOCKED选项与SELECT ... FOR SHARE和SELECT ... FOR UPDATE锁定读取语句。NOWAIT使语句在请求的行被其他事务锁定时立即返回。SKIP LOCKED从结果集中删除锁定的行。参见用NOWAIT和SKIP LOCKED锁定读并发。**
* **SELECT ... FOR SHARE取代了SELECT ... LOCK IN SHARE MODE，但是为了向后兼容，LOCK IN SHARE MODE仍然可用。这些语句是等价的。然而，FOR UPDATE和FOR SHARE支持NOWAIT、SKIP LOCKED和OF tbl\_name选项。参见章节13.2.10, "SELECT语句"。OF tbl\_name将锁定查询应用于命名的表。**
* **ADD PARTITION, DROP PARTITION, COALESCE PARTITION, REORGANIZE PARTITION, 和REBUILD PARTITION ALTER TABLE选项由本地的就地分区API支持，可以和ALGORITHM={COPY|INPLACE}和LOCK子句一起使用。**
* **带有ALGORITHM=INPLACE的DROP PARTITION会删除存储在分区中的数据，并删除该分区。然而，带有ALGORITHM=COPY或old\_alter\_table=ON的DROP PARTITION会重建分区表，并尝试将数据从被删除的分区转移到另一个具有兼容PARTITION ... VALUES定义。不能移动到另一个分区的数据被删除。**
* **InnoDB存储引擎现在使用MySQL数据字典，而不是它自己的特定存储引擎的数据字典。关于数据字典的信息，见第14章，MySQL数据字典。**
* **mysql系统表和数据字典表现在在MySQL数据目录中名为mysql.ibd的单一InnoDB表空间文件中创建。以前，这些表是在mysql数据库目录下的单个InnoDB表空间文件中创建的。**
* **以下是MySQL 8.0中引入的undo tablespaces变化。**
  + **默认情况下，Undo logs现在驻留在两个undo tablespaces中，这些表空间是在MySQL实例初始化时创建的。Undo logs不再在系统表空间中创建。**
  + **从MySQL 8.0.14开始，可以在运行时使用CREATE UNDO TABLESPACE语法在选定的位置创建额外的undo tablespaces。**

CREATE UNDO TABLESPACE ***tablespace\_name*** ADD DATAFILE '***file\_name***.ibu';

**使用CREATE UNDO TABLESPACE语法创建的Undo表空间可以在运行时使用DROP UNDO TABLESPACE语法删除。**

DROP UNDO TABLESPACE ***tablespace\_name***;

**ALTER UNDO TABLESPACE 语法可以用来标记一个undo tablespaces为活动或不活动。**

ALTER UNDO TABLESPACE ***tablespace\_name*** SET {ACTIVE|INACTIVE};

**在 INFORMATION\_SCHEMA.INNODB\_TABLESPACES 表中增加了显示表空间状态的 STATE 列。一个撤销的表空间在被撤销之前必须处于空的状态。**

* **innodb\_undo\_log\_truncate变量默认是启用的。**
* **innodb\_rollback\_segments变量定义了每个undo tablespaces的回滚段的数量。以前，innodb\_rollback\_segments指定了MySQL实例的回滚段的总数。这一变化增加了可用于并发事务的回滚段的数量。更多的回滚段增加了并发事务为Undo logs使用单独的回滚段的可能性，导致更少的资源争夺**
* **修改了影响缓冲池预冲洗和冲洗行为的变量的默认值**
* **innodb\_max\_dirty\_pages\_pct\_lwm的默认值现在是10。之前的默认值为0，禁用缓冲池预冲洗。当缓冲池中的脏页百分比超过10%时，10的值可以启用预冲洗。启用预冲洗可以提高性能的一致性。**
* **innodb\_max\_dirty\_pages\_pct默认值从75增加到90。InnoDB尝试从缓冲池中刷新数据，使脏页的百分比不超过这个值。增加的默认值允许在缓冲池中有更大比例的脏页。**
* **默认的innodb\_autoinc\_lock\_mode设置现在是2（交错）。交错锁模式允许并行地执行多行插入，这提高了并发性和可扩展性。新的innodb\_autoinc\_lock\_mode默认设置反映了MySQL 5.7中从基于语句的复制到基于行的复制作为默认复制类型的变化。基于语句的复制需要连续的自动增量锁模式（以前的默认值），以确保自动增量值以可预测和可重复的顺序分配给给定的SQL语句序列，而基于行的复制对SQL语句的执行顺序不敏感。更多信息，请参阅InnoDB AUTO\_INCREMENT锁模式。**

**对于使用基于语句复制的系统，新的innodb\_autoinc\_lock\_mode默认设置可能会破坏依赖于顺序自动增量值的应用程序。要恢复以前的默认值，请将innodb\_autoinc\_lock\_mode设置为1。**

* **ALTER TABLESPACE ...支持重命名一个普通表空间。语法来支持。**
* **新的innodb\_dedicated\_server变量，默认是禁用的，可以用来让InnoDB根据服务器上检测到的内存量自动配置以下选项**
* **innodb\_buffer\_pool\_size**
* **innodb\_log\_file\_size**
* **innodb\_flush\_method**

**该选项适用于在专用服务器上运行的MySQL服务器实例。欲了解更多信息，请参见第15.8.12节 "为专用的MySQL服务器启用自动配置"。**

* **新的INFORMATION\_SCHEMA.INNODB\_TABLESPACES\_BRIEF视图为InnoDB表空间提供空间、名称、路径、标志和空间类型数据。**
* **与MySQL捆绑的zlib库版本从1.2.3版提高到1.2.11版。MySQL在zlib库的帮助下实现了压缩。**

**如果你使用InnoDB压缩表，请参阅第2.11.4节 "MySQL 8.0中的变化"，了解相关的升级影响。**

* **除了全局临时表空间和undo tablespaces文件外，所有InnoDB表空间文件中都存在序列化字典信息（SDI）。SDI是表和表空间对象的序列化元数据。SDI数据的存在提供了元数据的冗余性。例如，如果数据字典变得不可用，字典对象元数据可以从表空间文件中提取。SDI提取是使用ibd2sdi工具进行的。SDI数据是以JSON格式存储的。**

**在表空间文件中包含SDI数据会增加表空间文件的大小。一个SDI记录需要一个索引页，默认大小为16KB。然而，SDI数据在存储时被压缩，以减少存储空间的占用。**

* **InnoDB存储引擎现在支持原子DDL，这确保了DDL操作被完全提交或回滚，即使服务器在操作过程中停止了。更多信息，请参见第13.1.1节，"原子数据定义语句支持"。**
* **当服务器处于离线状态时，可以使用innodb\_directories选项将表空间文件移动或恢复到一个新的位置。欲了解更多信息，请参见第15.6.3.6节 "在服务器离线时移动表空间文件"。**
* **实现了以下redo log的优化。**
  + **用户线程现在可以同时向日志缓冲区写入，而无需同步写入。**
  + **用户线程现在可以按照宽松的顺序将脏页添加到冲刷列表中。**
  + **一个专门的日志线程现在负责将日志缓冲区写入系统缓冲区，将系统缓冲区刷新到磁盘，通知用户线程关于写入和刷新的重做，维护放松的刷新列表顺序所需的滞后，以及写入检查点。**
  + **增加了系统变量，用于配置等待冲刷重做的用户线程对自旋延迟的使用。**
    - **innodb\_log\_wait\_for\_flush\_spin\_hwm。定义了最大的平均日志刷新时间，超过这个时间，用户线程在等待刷新的重做时不再旋转。**
    - **innodb\_log\_spin\_cpu\_abs\_lwm: 定义最小的CPU使用量，低于该值时，用户线程在等待冲刷的重做时不再旋转。**
    - **innodb\_log\_spin\_cpu\_pct\_hwm: 定义用户线程在等待刷新的重做时不再旋转的最大CPU使用量。**
    - **innodb\_log\_buffer\_size变量现在是动态的，它允许在服务器运行时调整日志缓冲区的大小。**

**欲了解更多信息，请参见第8.5.4节 "优化InnoDBredo log"。**

* **从MySQL 8.0.12开始，Undo logs支持对大对象（LOB）数据的小型更新，这提高了大小为100字节或更小的LOB更新的性能。以前，LOB更新至少要有一个LOB页面大小，这对于可能只修改几个字节的更新来说是不太理想的。这项改进建立在MySQL 8.0.4中对LOB数据的部分更新的支持之上。**
* **从MySQL 8.0.12开始，ALGORITHM=INSTANT被支持用于以下ALTER TABLE操作。**
  + **增加一个列。这个功能也被称为 "即时添加列"。适用的限制条件。参见章节15.12.1, "在线DDL操作"。**
  + **添加或删除一个虚拟列。**
  + **添加或删除一个列的默认值。**
  + **修改一个ENUM或SET列的定义。**
  + **改变索引类型。**
  + **重命名一个表。**

**支持ALGORITHM=INSTANT的操作只修改数据字典中的元数据。在表上没有元数据锁，表的数据不受影响，使得这些操作是瞬时的。如果没有明确指定，ALGORITHM=INSTANT会被支持它的操作默认使用。如果指定了ALGORITHM=INSTANT但不支持，操作会立即失败并出现错误。**

**关于支持ALGORITHM=INSTANT的操作的更多信息，见第15.12.1节 "在线DDL操作"。**

* **从MySQL 8.0.13开始，TempTable存储引擎支持存储二进制大对象（BLOB）类型的列。这一改进提高了使用包含BLOB数据的临时表的查询性能。以前，包含BLOB数据的临时表被存储在由internal\_tmp\_disk\_storage\_engine定义的磁盘存储引擎中。更多信息，见第8.4.4节 "MySQL中的内部临时表使用"。**
* **从MySQL 8.0.13开始，InnoDB静态数据加密功能支持一般表空间。以前，只有文件-每表的表空间可以被加密。为了支持一般表空间的加密，CREATE TABLESPACE和ALTER TABLESPACE语法被扩展到包括一个ENCRYPTION子句。**

**INFORMATION\_SCHEMA.INNODB\_TABLESPACES表现在包括一个ENCRYPTION列，表明表空间是否被加密。**

**增加了 stage/innodb/alter tablespace (encryption) 性能模式阶段工具，以允许监控一般表空间加密操作。**

* **禁用innodb\_buffer\_pool\_in\_core\_file变量，通过排除InnoDB缓冲池页面来减少核心文件的大小。要使用这个变量，core\_file变量必须被启用，并且操作系统必须支持MADV\_DONTDUMP非POSIX扩展到madvise()，在Linux 3.4及以后版本中支持。更多信息，见第15.8.3.7节 "从核心文件中排除缓冲池页"。**
* **从MySQL 8.0.13开始，用户创建的临时表和由优化器创建的内部临时表被存储在会话临时表空间中，这些临时表空间是从临时表空间池中分配给会话的。当一个会话断开连接时，它的临时表空间被截断并释放回池中。在以前的版本中，临时表是在全局临时表空间（ibtmp1）中创建的，在临时表被丢弃后，它不会将磁盘空间返回给操作系统。**

**innodb\_temp\_tablespaces\_dir变量定义了创建会话临时表空间的位置。默认的位置是数据目录下的#innodb\_temp目录。**

**INNODB\_SESSION\_TEMP\_TABLESPACES表提供关于会话临时表空间的元数据。**

**全局临时表空间（ibtmp1）现在存储对用户创建的临时表所做更改的回滚段。**

* **从MySQL 8.0.14开始，InnoDB支持并行集群索引读取，这可以提高CHECK TABLE性能。这个功能不适用于二级索引扫描。innodb\_parallel\_read\_threads会话变量必须被设置为大于1的值，才能发生并行集群索引读取。默认值是4。 用于执行并行集群索引读取的实际线程数由innodb\_parallel\_read\_threads设置或要扫描的索引子树数决定，以较小者为准。**
* **从8.0.14开始，当innodb\_dedicated\_server变量被启用时，日志文件的大小和数量是根据自动配置的缓冲池大小来配置的。以前，日志文件的大小是根据服务器上检测到的内存量来配置的，而日志文件的数量则不是自动配置的。**
* **从8.0.14开始，CREATE TABLESPACE语句的ADD DATAFILE子句是可选的，这允许没有FILE权限的用户创建表空间。在没有ADD DATAFILE子句的情况下执行的CREATE TABLESPACE语句隐含地创建了一个具有唯一文件名的表空间数据文件。**
* **默认情况下，当TempTable存储引擎占用的内存量超过temptable\_max\_ram变量定义的内存限制时，TempTable存储引擎开始从磁盘分配内存映射的临时文件。从MySQL 8.0.16开始，这种行为由temptable\_use\_mmap变量控制。禁用temptable\_use\_mmap导致TempTable存储引擎使用InnoDB磁盘上的内部临时表而不是内存映射的文件作为其溢出机制。欲了解更多信息，请参见内部临时表存储引擎。**
* **从MySQL 8.0.16开始，InnoDB静态数据加密功能支持对mysql系统表空间进行加密。mysql系统表空间包含mysql系统数据库和MySQL数据字典表。欲了解更多信息，请参见第15.13节 "InnoDB静态数据加密"。**
* **在MySQL 8.0.16中引入的innodb\_spin\_wait\_pause\_multiplier变量为自旋锁轮询的持续时间提供了更大的控制，当线程等待获得一个mutex或rw-lock时，会出现自旋锁轮询的延迟。可以对延迟进行更精细的调整，以考虑不同处理器架构上PAUSE指令持续时间的差异。更多信息请参见第15.8.8节，"配置自旋锁轮询"。**
* **在MySQL 8.0.17中，通过更好地利用读取线程，通过减少并行扫描期间发生的预取活动的读取线程I/O，以及通过支持分区的并行扫描，InnoDB的大数据集并行读取线程性能得到了改善。**

**并行读线程功能由innodb\_parallel\_read\_threads变量控制。现在的最大设置是256，这是所有客户端连接的线程总数。如果达到了线程限制，连接会退回到使用单线程。**

* **在MySQL 8.0.18中引入的innodb\_idle\_flush\_pct变量允许在空闲期间对页面刷新进行限制，这可以帮助延长固态存储设备的寿命。见限制闲置期间的缓冲区刷新。**
* **从MySQL 8.0.19开始，支持为生成直方图统计的目的对InnoDB数据进行有效采样。参见直方图统计分析。**
* **从MySQL 8.0.20开始，双写缓冲区的存储区域驻留在双写文件中。在以前的版本中，该存储区位于系统表空间中。将存储区移出系统表空间，可以减少写延迟，增加吞吐量，并为双写缓冲区页面的放置提供灵活性。以下是为高级双写缓冲区配置引入的系统变量。**
  + **innodb\_doublewrite\_dir**

**定义了双写缓冲区的文件目录。**

* **innodb\_doublewrite\_files**

**定义了双写文件的数量。**

* **innodb\_doublewrite\_pages**

**定义每个线程在批量写入时的最大双写页数。**

* **innodb\_doublewrite\_batch\_size**

**定义在一个批次中要写入的双写页数。**

**更多信息，请参见第15.6.4节，"双写缓冲区"。**

* **在MySQL 8.0.20中改进了争夺意识到的事务调度（CATS）算法，该算法优先考虑正在等待锁的事务。事务调度权重的计算现在完全由一个单独的线程执行，这提高了计算性能和准确性。**

**先入先出（FIFO）算法，也被用于事务调度，被删除。FIFO算法由于CATS算法的增强而变得多余了。以前由FIFO算法执行的事务调度现在由CATS算法执行。**

**在 INFORMATION\_SCHEMA.INNODB\_TRX 表中增加了 TRX\_SCHEDULE\_WEIGHT 列，它允许查询 CATS 算法分配的事务调度权重。**

**增加了以下 INNODB\_METRICS 计数器，用于监控代码级的事务调度事件。**

* **lock\_rec\_release\_attempts**

**释放记录锁的尝试次数。**

* **lock\_rec\_grant\_attempts**

**试图授予记录锁的次数。**

* **lock\_schedule\_refreshes**

**分析等待图以更新事务计划权重的次数。**

**欲了解更多信息，请参见第15.7.6节 "事务调度"。**

* **从MySQL 8.0.21开始，为了提高需要访问表和行资源的锁队的操作的并发性，锁系统突变（lock\_sys->mutex）被分片锁所取代，锁队被分组为表和页锁队分片，每个分片由一个专用突变保护。以前，单一锁系统的mutex保护所有的锁队，这在高并发系统中是一个争论点。新的分片实现允许对锁队列进行更细化的访问。**

**锁系统的mutex（lock\_sys->mutex）被以下分片锁所取代。**

* **一个全局锁（lock\_sys->latches.global\_latch）由64个读写锁对象（rw\_lock\_t）组成。对单个锁队的访问需要一个共享的全局锁和一个锁队分片上的锁。需要访问所有锁队列的操作需要一个独占的全局锁，它锁住所有表和页锁队列分片。**
* **表分片锁（lock\_sys->latches.table\_shards.mutexes），实现为一个512个分片的数组，每个分片专门用于512个表锁队列分片之一。**
* **Page shard latches (lock\_sys->latches.page\_shards.mutexes), 实现为512个mutexes的数组，每个mutex专门用于512个page lock queue shard中的一个。**

**性能模式wait/synch/mutex/innodb/lock\_mutex工具用于监控单一锁系统的mutex，被用于监控新的全局、表分片和页分片锁的工具所取代。**

* **wait/synch/sxlock/innodb/lock\_sys\_global\_rw\_lock**
* **wait/synch/mutex/innodb/lock\_sys\_table\_mutex**
* **wait/synch/mutex/innodb/lock\_sys\_page\_mutex**

**从MySQL 8.0.21开始，使用DATA DIRECTORY子句在数据目录之外创建的表和表分区数据文件被限制在InnoDB已知的目录中。这一变化允许数据库管理员控制表空间数据文件的创建位置，并确保在恢复期间可以找到这些数据文件。**

**一般的和每表文件的表空间数据文件（.ibd文件）不能再在undo tablespaces目录（innodb\_undo\_directory）中创建，除非该目录直接被InnoDB知道。**

**已知的目录是那些由datadir、innodb\_data\_home\_dir和innodb\_directories变量定义的目录。**

**Truncating一个驻留在逐个文件表空间中的InnoDB表时，会放弃现有的表空间并创建一个新的表空间。从MySQL 8.0.21开始，InnoDB在默认位置创建新的表空间，如果当前表空间目录未知，则向错误日志写入警告。要让TRUNCATE TABLE在其当前位置创建表空间，在运行TRUNCATE TABLE之前将目录添加到innodb\_directories设置中。**

* **从MySQL 8.0.21开始，可以使用ALTER INSTANCE {ENABLE|DISABLE}启用和禁用redo log。INNODB REDO\_LOG语法启用和禁用。这个功能是为了将数据加载到一个新的MySQL实例。禁用redo log有助于通过避免redo log的写入来加快数据加载速度。**

**新的INNODB\_REDO\_LOG\_ENABLE权限允许启用和禁用redo log记录。**

**新的Innodb\_redo\_log\_enabled状态变量允许监控redo log的状态。**

**参见禁用redo log。**

* **在启动时，InnoDB根据存储在数据字典中的表空间文件路径验证已知的表空间文件的路径，以防表空间文件被移动到不同的位置。新的innodb\_validate\_tablespace\_paths变量在MySQL 8.0.21中引入，允许禁用表空间路径验证。这个功能是为不移动表空间文件的环境准备的。禁用表空间路径验证可以改善有大量表空间文件的系统的启动时间。**

**更多信息，见第15.6.3.7节 "禁用表空间路径验证"。**

* **从MySQL 8.0.21开始，在支持原子DDL的存储引擎上，CREATE TABLE ... SELECT语句在使用基于行的复制时在二进制日志中被记录为一个事务。以前，它被记录为两个事务，一个用于创建表，另一个用于插入数据。有了这个变化，CREATE TABLE ... SELECT 语句现在对基于行的复制是安全的，并且允许与基于 GTID 的复制一起使用。更多信息，请参见第13.1.1节，"原子数据定义语句支持"。**
* **在一个繁忙的系统上截断一个undo tablespaces可能会影响性能，因为相关的冲洗操作会从缓冲池中删除旧的undo tablespaces页面，并将新的undo tablespaces的初始页面冲洗到磁盘。为了解决这个问题，从MySQL 8.0.21开始，冲刷操作被移除。**

**旧的undo tablespaces页在最近使用最少时被动释放，或者在下一个完整检查点时被删除。新的undo tablespaces的初始页现在被重做记录，而不是在截断操作期间被刷入磁盘，这也提高了undo tablespaces截断操作的持久性。**

**为了防止过多的undo tablespaces截断操作引起的潜在问题，在检查点之间对同一undo tablespaces的截断操作现在被限制在64次。如果超过了这个限制，一个undo tablespaces仍然可以变得不活跃，但直到下一个检查点之后才会被截断。**

**删除了与失效的撤消截断冲刷操作相关的INNODB\_METRICS计数器。被删除的计数器包括：undo\_truncate\_sweep\_count、undo\_truncate\_sweep\_usec、undo\_truncate\_flush\_count以及undo\_truncate\_flush\_usec。**

**参见第15.6.3.4节，"undo tablespaces"。**

* **从MySQL 8.0.22开始，新的innodb\_extend\_and\_initialize变量允许配置InnoDB如何分配空间给Linux上的每表文件和一般表空间。默认情况下，当一个操作需要一个表空间的额外空间时，InnoDB会向表空间分配页面，并向这些页面物理写入NULL。如果经常分配新的页面，这种行为会影响性能。你可以在Linux系统上禁用innodb\_extend\_and\_initialize来避免向新分配的表空间页面物理写入NULL。当innodb\_extend\_and\_initialize被禁用时，空间的分配是使用posix\_fallocate()调用，它保留了空间而没有物理写入NULL。**

**posix\_fallocate()操作不是原子性的，这使得在为表空间文件分配空间和更新文件元数据之间有可能发生故障。这样的故障会使新分配的页面处于未初始化的状态，导致InnoDB试图访问这些页面时出现故障。为了防止这种情况，InnoDB在分配一个新的表空间页之前会写一条redo log记录。如果页面分配操作被中断，在恢复过程中会从redo log记录中重放该操作。**

* **从MySQL 8.0.23开始，InnoDB支持对属于加密表空间的双写文件页进行加密。这些页面是使用相关表空间的加密密钥进行加密的。更多信息，请参见第15.13节，"InnoDB静态数据加密"。**
* **temptable\_max\_mmap变量在MySQL 8.0.23中引入，定义了TempTable存储引擎在开始在磁盘上存储内部临时表数据之前允许从内存映射（MMAP）文件分配的最大内存量。设置为0将禁止从MMAP文件中分配。更多信息，见第8.4.4节 "MySQL中的内部临时表使用"。**
* **AUTOEXTEND\_SIZE选项在MySQL 8.0.23中引入，定义了InnoDB在表空间变满时扩展其大小的数量，使得以较大的增量扩展表空间大小成为可能。AUTOEXTEND\_SIZE选项支持CREATE TABLE、ALTER TABLE、CREATE TABLESPACE和ALTER TABLESPACE语句。更多信息，请参见章节15.6.3.9, "表空间AUTOEXTEND\_SIZE配置"。**

**在 INFORMATION\_SCHEMA.INNODB\_TABLESPACES 表中增加了 AUTOEXTEND\_SIZE 大小列。**

* 字符集支持**。 默认的字符集已经从 latin1 改为 utf8mb4。utf8mb4字符集有几个新的排序，包括utf8mb4\_ja\_0900\_as\_cs，这是MySQL中第一个可用于Unicode的日语特定排序。更多信息，见第10.10.1节 "Unicode字符集"**
* JSON的增强**。 对MySQL的JSON功能进行了以下改进或添加。**
* **增加了->>（内联路径）操作符，它相当于在JSON\_EXTRACT()的结果上调用JSON\_UNQUOTE()。**

**这是MySQL 5.7中引入的列路径操作符->的细化；col->>"$.path "等同于JSON\_UNQUOTE(col->"$.path")。内联路径操作符可以在任何可以使用JSON\_UNQUOTE(JSON\_EXTRACT())的地方使用，如SELECT列列表、WHERE和HAVING子句、ORDER BY和GROUP BY子句。更多信息请参见操作符的描述，以及JSON路径语法。**

* **增加了两个JSON聚合函数JSON\_ARRAYAGG()和JSON\_OBJECTAGG()。JSON\_ARRAYAGG()将一个列或表达式作为其参数，并将其结果聚合为一个单一的JSON数组。表达式可以评估为任何MySQL数据类型；这不一定是一个JSON值。JSON\_OBJECTAGG()接收两个列或表达式，它将其解释为一个键和一个值；它将结果作为一个单一的JSON对象返回。更多信息和例子请参见第12.20节 "聚合函数"。**
* **增加了JSON实用函数JSON\_PRETTY()，它以易于阅读的格式输出现有的JSON值；每个JSON对象成员或数组值被打印在单独的一行，子对象或数组相对于其父对象有2个空格。这个函数也适用于可以被解析为JSON值的字符串。**

**更详细的信息和例子，见第12.18.8节，"JSON实用函数"。**

* **当使用ORDER BY对查询中的JSON值进行排序时，每个值现在由排序键的可变长度部分表示，而不是固定的1K大小的部分。在许多情况下，这可以减少过度使用。例如，一个标量的INT甚至BIGINT值实际上只需要很少的字节，所以这个空间的剩余部分（高达90%或更多）被填充物占用了。这一改变对性能有以下好处。**
* **排序缓冲区的空间现在得到了更有效的利用，因此文件排序不需要像固定长度的排序键那样提前或经常刷新到磁盘。这意味着更多的数据可以在内存中进行排序，避免了不必要的磁盘访问。**
* **较短的键可以比长的键更快地进行比较，在性能上有明显的改善。这对于完全在内存中执行的排序以及需要从磁盘写入和读取的排序都是如此。**
* **在MySQL 8.0.2中增加了对JSON列值的部分就地更新的支持，这比完全删除一个现有的JSON值并在其位置上写入一个新的值更有效，就像以前更新任何JSON列时所做的那样。为了应用这种优化，更新必须使用JSON\_SET(), JSON\_REPLACE(), 或JSON\_REMOVE()。新的元素不能被添加到被更新的JSON文档中；文档中的值不能比更新前占用更多空间。请参阅JSON值的部分更新，以了解对这些要求的详细讨论。**

**JSON文档的部分更新可以被写入二进制日志，比记录完整的JSON文档占用更少的空间。当使用基于语句的复制时，部分更新总是这样被记录下来。为了使其在基于行的复制中工作，你必须首先设置binlog\_row\_value\_options=PARTIAL\_JSON；更多信息请参见这个变量的描述。**

* **增加了JSON实用函数JSON\_STORAGE\_SIZE()和JSON\_STORAGE\_FREE()。JSON\_STORAGE\_SIZE()返回在任何部分更新之前用于JSON文档的二进制表示法的存储空间（见前一项），字节数。JSON\_STORAGE\_FREE()显示JSON类型的表列在使用JSON\_SET()或JSON\_REPLACE()进行部分更新后的剩余空间量；如果新值的二进制表示小于先前的值，则该值大于零。**

**这些函数中的每一个也接受JSON文档的有效字符串表示。对于这样的值，JSON\_STORAGE\_SIZE()返回其转换为JSON文档后的二进制表示所使用的空间。对于一个包含JSON文档的字符串表示的变量，JSON\_STORAGE\_FREE()返回0。如果它的(非空)参数不能被解析为有效的JSON文档，这两个函数都会产生一个错误，如果参数是空的，则产生NULL。**

**更多信息和例子，见第12.18.8节，"JSON实用函数"。JSON\_STORAGE\_SIZE()和JSON\_STORAGE\_FREE()已在MySQL 8.0.2中实现。**

* **在MySQL 8.0.2中增加了对XPath表达式中$[1到5]等范围的支持。在这个版本中还增加了对last关键字和相对寻址的支持，例如$[last]总是选择数组中最后一个（编号最高的）元素，而$[last-1]则是倒数第二的元素。last和使用它的表达式也可以包含在范围定义中。例如，$[last-2 to last-1]返回数组中的最后两个元素，但只有一个。参见搜索和修改JSON值，了解更多信息和例子。**
* **增加了一个JSON合并函数，以符合RFC 7396的要求。JSON\_MERGE\_PATCH()，当用于2个JSON对象时，将它们合并成一个单一的JSON对象，其成员是以下集合的联合体。**
* **第一个对象的每个成员，在第二个对象中没有相同键的成员。**
* **第二个对象中的每个成员，在第一个对象中没有具有相同键的成员，并且其值不是JSON null字面。**
* **每个成员的键都存在于两个对象中，并且其在第二个对象中的值不是JSON null字面。**
* **作为这项工作的一部分，JSON\_MERGE()函数被重新命名为JSON\_MERGE\_PRESERVE()。JSON\_MERGE()在MySQL 8.0中继续被认为是JSON\_MERGE\_PRESERVE()的别名，但现在已被废弃，并将在MySQL的未来版本中被删除。**

**更多信息和例子，见第12.18.4节，"修改JSON值的函数"。**

* **实现了重复键的 "最后一个重复键获胜 "的规范化，与RFC 7159和大多数JavaScript解析器一致。这种行为的一个例子显示在这里，其中只有最右边的成员有键x被保留。**

mysql> SELECT **JSON\_OBJECT('x', '32', 'y', '[true, false]',**

> **'x', '"abc"', 'x', '100') AS Result;**

+------------------------------------+

| Result |

+------------------------------------+

| {"x": "100", "y": "[true, false]"} |

+------------------------------------+

1 row in set (0.00 sec)

**插入MySQL JSON列的值也以这种方式进行规范化，如本例所示。**

mysql> **CREATE TABLE t1 (c1 JSON);**

mysql> **INSERT INTO t1 VALUES ('{"x": 17, "x": "red", "x": [3, 5, 7]}');**

mysql> **SELECT c1 FROM t1;**

+------------------+

| c1 |

+------------------+

| {"x": [3, 5, 7]} |

+------------------+

**这是与以前的MySQL版本不兼容的变化，以前的版本在这种情况下使用 "第一个重复的键获胜 "的算法。**

**参见JSON值的规范化、合并和自动包装，了解更多信息和例子。**

* **在MySQL 8.0.4中增加了JSON\_TABLE()函数。该函数接受JSON数据并将其作为具有指定列的关系表返回。**

**这个函数的语法是JSON\_TABLE(expr, path COLUMNS column\_list) [AS] alias)，其中expr是返回JSON数据的表达式，path是应用于源的JSON路径，column\_list是一个列定义的列表。这里显示了一个例子。**

mysql> ***SELECT \****

-> ***FROM***

-> ***JSON\_TABLE(***

-> ***'[{"a":3,"b":"0"},{"a":"3","b":"1"},{"a":2,"b":1},{"a":0},{"b":[1,2]}]',***

-> ***"$[\*]" COLUMNS(***

-> ***rowid FOR ORDINALITY,***

->

-> ***xa INT EXISTS PATH "$.a",***

-> ***xb INT EXISTS PATH "$.b",***

->

-> ***sa VARCHAR(100) PATH "$.a",***

-> ***sb VARCHAR(100) PATH "$.b",***

->

-> ***ja JSON PATH "$.a",***

-> ***jb JSON PATH "$.b"***

-> ***)***

-> ***) AS jt1;***

+-------+------+------+------+------+------+--------+

| rowid | xa | xb | sa | sb | ja | jb |

+-------+------+------+------+------+------+--------+

| 1 | 1 | 1 | 3 | 0 | 3 | "0" |

| 2 | 1 | 1 | 3 | 1 | "3" | "1" |

| 3 | 1 | 1 | 2 | 1 | 2 | 1 |

| 4 | 1 | 0 | 0 | NULL | 0 | NULL |

| 5 | 0 | 1 | NULL | NULL | NULL | [1, 2] |

+-------+------+------+------+------+------+--------+

**JSON源表达式可以是任何产生有效JSON文档的表达式，包括一个JSON字面，一个表列，或一个返回JSON的函数调用，如JSON\_EXTRACT(t1, data, '$.post.comment') 。更多信息请参见第12.18.6节，"JSON表函数"。**

* 数据类型支持**。 MySQL现在支持在数据类型规范中使用表达式作为默认值。这包括使用表达式作为BLOB、TEXT、GEOMETRY和JSON数据类型的缺省值，这些数据类型以前根本不能被分配缺省值。详情请参见第11.6节 "数据类型默认值"。**
* 优化器。 **增加了这些优化器的改进。**
* **MySQL现在支持不可见索引。一个不可见的索引根本不被优化器使用，但在其他方面被正常维护。索引在默认情况下是可见的。不可见的索引使得有可能测试删除索引对查询性能的影响，而不需要进行破坏性的改变，如果索引被证明是必需的，则必须撤销。参见第8.3.12节，"隐形索引"。**
* **MySQL现在支持降序索引。索引定义中的DESC不再被忽略，而是导致按降序存储键值。以前，索引可以按相反的顺序进行扫描，但在性能上会受到影响。降序索引可以按正向顺序扫描，这更有效率。降序索引也使优化器有可能在最有效的扫描顺序中混合了某些列的升序和其他列的降序时使用多列索引。参见第8.3.13节，"降序索引"。**
* **MySQL现在支持创建索引表达式值而不是列值的功能索引关键部分。功能性关键部分可以对无法以其他方式进行索引的值进行索引，例如JSON值。详情请参见第13.1.15节 "CREATE INDEX语句"。**
* **在MySQL 8.0.14及以后的版本中，由常数字面表达式产生的微不足道的WHERE条件在准备期间被删除，而不是在后来的优化期间被删除。在这个过程中提前删除条件使得有外联的查询有可能简化联接，比如这个。**

SELECT \* FROM t1 LEFT JOIN t2 ON ***condition\_1*** WHERE ***condition\_2*** OR 0 = 1

**现在，优化器在准备过程中发现0 = 1总是假的，这使得OR 0 = 1成为多余的，于是将其删除，留下这个。**

SELECT \* FROM t1 LEFT JOIN t2 ON ***condition\_1*** where ***condition\_2***

**现在，优化器可以把这个查询重写成一个内联，像这样。**

SELECT \* FROM t1 LEFT JOIN t2 WHERE ***condition\_1*** AND ***condition\_2***

**更多信息，见第8.2.1.9节 "外连接优化"。**

* **在MySQL 8.0.16及更高版本中，MySQL可以在优化时使用常数折叠来处理列和常数值之间的比较，其中常数超出范围或处于与列的类型有关的范围边界上，而不是在执行时对每行进行比较。例如，给定一个带有TINYINT UNSIGNED列c的表t，优化器可以重写一个条件，比如WHERE c < 256到WHERE 1（并完全优化条件），或者WHERE c >= 255到WHERE c = 255。**

**参见第8.2.1.14节 "恒定折叠优化"，以了解更多信息。**

* **从MySQL 8.0.16开始，用于IN子查询的半连接优化现在也可以应用于EXISTS子查询。此外，优化器现在对附属于子查询的WHERE条件中的琐碎相关的等价谓词进行装饰，以便它们可以与IN子查询中的表达式类似地处理；这适用于EXISTS和IN子查询。**

**欲了解更多信息，请参阅第8.2.2.1节 "用半连接转换优化IN和EXISTS子查询谓词"。**

* **从MySQL 8.0.17开始，服务器在上下文阶段将任何不完整的SQL谓词（即具有WHERE值形式的谓词，其中值是一个列名或常量表达式，并且没有使用比较运算符）内部改写为WHERE值<>0，因此查询解析器、查询优化器和查询执行器只需要使用完整的谓词。**

**这个变化的一个可见效果是，对于布尔值，EXPLAIN输出现在显示的是true和false，而不是1和0。**

**这个变化的另一个影响是，在SQL布尔上下文中对JSON值的评估会执行与JSON整数0的隐式比较。 考虑在这里创建和填充的表。**

mysql> **CREATE TABLE test (id INT, col JSON);**

mysql> **INSERT INTO test VALUES (1, '{"val":true}'), (2, '{"val":false}');**

**以前，当在SQL布尔上下文中进行比较时，服务器试图将提取的真或假值转换为SQL布尔值，如以下使用IS TRUE的查询所示。**

mysql> **SELECT id, col, col->"$.val" FROM test WHERE col->"$.val" IS TRUE;**

+------+---------------+--------------+

| id | col | col->"$.val" |

+------+---------------+--------------+

| 1 | {"val": true} | true |

+------+---------------+--------------+

**在MySQL 8.0.17及以后的版本中，提取的值与JSON整数0的隐式复制会导致不同的结果。**

mysql> **SELECT id, col, col->"$.val" FROM test WHERE col->"$.val" IS TRUE;**

+------+----------------+--------------+

| id | col | col->"$.val" |

+------+----------------+--------------+

| 1 | {"val": true} | true |

| 2 | {"val": false} | false |

+------+----------------+--------------+

**从MySQL 8.0.21开始，你可以对提取的值使用JSON\_VALUE()，在执行测试之前进行类型转换，如图所示。**

mysql> **SELECT id, col, col->"$.val" FROM test**

-> **WHERE JSON\_VALUE(col, "$.val" RETURNING UNSIGNED) IS TRUE;**

+------+---------------+--------------+

| id | col | col->"$.val" |

+------+---------------+--------------+

| 1 | {"val": true} | true |

+------+---------------+--------------+

**同样从MySQL 8.0.21开始，服务器提供了警告 在SQL布尔上下文中评估一个JSON值会对JSON整数0进行隐式比较；如果这不是你想要的，当以这种方式在SQL布尔上下文中比较提取的值时，考虑用JSON\_VALUE RETURNING将JSON转换为一个SQL数字类型。**

* **在MySQL 8.0.17及以后的版本中，具有NOT IN（子查询）或NOT EXISTS（子查询）的WHERE条件在内部被转换为antijoin。(antijoin返回表的所有记录，对于这些记录，在它被连接的表中没有符合连接条件的记录。) 这就删除了子查询，这可以导致更快的查询执行，因为子查询的表现在在顶层被处理。**

**这与现有的外部连接的IS NULL（不存在）优化类似，并重复使用；见EXPLAIN额外信息。**

* **从MySQL 8.0.21开始，单表UPDATE或DELETE语句现在可以在许多情况下使用半联接转换或子查询的具体化。这适用于这里显示的形式的语句。**
  + **UPDATE t1 SET t1.a=value WHERE t1.a IN (SELECT t2.a FROM t2)**
  + **DELETE FROM t1 WHERE t1.a IN (SELECT t2.a FROM t2)**

**对于满足以下条件的单表UPDATE或DELETE，可以这样做。**

* **UPDATE或DELETE语句使用了一个具有[NOT] IN或[NOT] EXISTS谓词的子查询。**
* **该语句没有ORDER BY子句，也没有LIMIT子句。**

**(UPDATE和DELETE的多表版本不支持ORDER BY或LIMIT)。**

* **目标表不支持先读后写删除（仅与NDB表有关）。**
* **根据子查询中包含的任何提示和optimizer\_switch的值，允许半联接或子查询物化。**

**当半连接优化被用于符合条件的单表DELETE或UPDATE时，这在优化器跟踪中是可见的：对于多表语句，跟踪中有一个join\_optimization对象，而对于单表语句没有。这种转换在EXPLAIN FORMAT=TREE或EXPLAIN ANALYZE的输出中也是可见的；单表语句显示<不可由迭代器执行器执行>，而多表语句报告完整计划。**

**从MySQL 8.0.21开始，使用InnoDB表的多表UPDATE语句支持半一致性读取，用于比REPEATABLE READ弱的事务隔离级别。**

* 改进了哈希连接性能**。 MySQL 8.0.23重新实现了用于哈希连接的哈希表，从而使哈希连接性能得到了一些改进。这项工作包括对一个问题的修复（Bug #31516149，Bug #99933），即只有大约2/3分配给连接缓冲区（join\_buffer\_size）的内存可以被哈希连接实际使用。**

**新的哈希表通常比旧的更快，并且在对齐、键/值以及有许多相等键的情况下使用更少的内存。此外，当哈希表的大小增加时，服务器现在可以释放旧内存。**

* 常见的表表达式**。 MySQL现在支持普通表表达式，包括非递归和递归。普通表表达式可以使用命名的临时结果集，通过允许在SELECT语句和某些其他语句前的WITH子句来实现。更多信息，见第13.2.15节 "WITH（普通表表达式）"。**

**从MySQL 8.0.19开始，递归通用表表达式（CTE）的递归SELECT部分支持一个LIMIT子句。也支持带有OFFSET的LIMIT。参见递归公共表表达式，了解更多信息。**

* 窗口函数**。 MySQL现在支持窗口函数，对于查询的每一条记录，使用与该记录相关的记录进行计算。这些包括诸如RANK()、LAG()和NTILE()等函数。此外，现有的几个聚合函数现在可以作为窗口函数使用（例如，SUM()和AVG()）。更多信息，请参见第12.21节，"窗口函数"。**
* 横向派生表**。 一个派生表现在可以在前面加上LATERAL关键字，以指定它被允许在同一个FROM子句中引用（依赖）前面表的列。横向派生表使某些SQL操作成为可能，这些操作不能用非横向派生表来完成，或者需要较低效率的变通方法。参见章节13.2.11.9, "横向派生表"。**
* 单表DELETE语句中的别名**。 在MySQL 8.0.16及更高版本中，单表DELETE语句支持使用表别名。**
* 支持正则表达式**。 以前，MySQL使用Henry Spencer正则表达式库来支持正则表达式运算符（REGEXP、RLIKE）。正则表达式支持已经使用International Components for Unicode（ICU）重新实现，它提供完整的Unicode支持，并且是多字节安全的。REGEXP\_LIKE()函数以REGEXP和RLIKE操作符的方式执行正则表达式匹配，现在是该函数的同义词。此外，REGEXP\_INSTR()、REGEXP\_REPLACE()和REGEXP\_SUBSTR()函数可分别用于查找匹配位置和执行子串替换和提取。regexp\_stack\_limit和regexp\_time\_limit系统变量提供了对匹配引擎资源消耗的控制。更多信息，请参阅第12.8.2节 "正则表达式"。关于使用正则表达式的应用程序可能会受到实现变化的影响的信息，请参阅正则表达式兼容性考虑。**
* 内部临时表**。 TempTable存储引擎取代了MEMORY存储引擎，成为内存中内部临时表的默认引擎。TempTable存储引擎为VARCHAR和VARBINARY列提供有效的存储。internal\_tmp\_mem\_storage\_engine会话变量定义了内存内部临时表的存储引擎。允许的值是TempTable（默认）和MEMORY。temptable\_max\_ram变量定义了TempTable存储引擎在数据被存储到磁盘之前可以使用的最大内存量。**
* 日志记录**。 错误日志被重写以使用MySQL组件架构。传统的错误日志使用内置组件实现，而使用系统日志的日志则作为一个可加载组件实现。此外，还提供了一个可加载的JSON日志写入器。要控制启用哪些日志组件，请使用log\_error\_services系统变量。更多信息，请参阅第5.4.2节 "错误日志"。**
* 备份锁**。 一种新的备份锁允许在线备份期间的DML，同时防止可能导致不一致的快照的操作。新的备份锁由 LOCK INSTANCE FOR BACKUP 和 UNLOCK INSTANCE 语法支持。使用这些语句需要 BACKUP\_ADMIN 权限。**
* 复制**。 对MySQL Replication进行了以下改进。**
  + **MySQL Replication现在支持使用紧凑的二进制格式对JSON文档的部分更新进行二进制记录，比记录完整的JSON文档节省了日志中的空间。当使用基于语句的日志记录时，这种紧凑的日志记录会自动完成，并且可以通过将新的binlog\_row\_value\_options系统变量设置为PARTIAL\_JSON来启用。更多信息，请参阅JSON值的部分更新，以及binlog\_row\_value\_options的描述。**
* 连接管理。 **MySQL服务器现在允许为管理连接专门配置一个TCP/IP端口。这提供了一个替代方案，即在用于普通连接的网络接口上允许单一的管理连接，即使max\_connections连接已经建立。见第5.1.12.1节，"连接接口"。**

**MySQL现在对压缩的使用提供了更多控制，以尽量减少通过连接到服务器发送的字节数。以前，一个给定的连接要么是未压缩的，要么使用zlib压缩算法。现在，它也可以使用zstd算法，并为zstd连接选择一个压缩级别。允许的压缩算法可以在服务器端配置，也可以在客户端程序和参与源/复制或组复制的服务器的连接-起源端配置。更多信息，见第4.2.8节，"连接压缩控制"。**

* 配置。 **整个MySQL的主机名的最大允许长度已经从以前的60个字符的限制提高到255个ASCII字符。这适用于，例如，数据字典、mysql系统模式、性能模式、INFORMATION\_SCHEMA和sys模式中与主机名相关的列；CHANGE MASTER TO语句的MASTER\_HOST值；SHOW PROCESSLIST语句输出中的Host列；账户名中的主机名（如在账户管理语句和DEFINER属性中使用的）；以及与主机名相关的命令选项和系统变量**

注意事项。

* **允许的主机名长度的增加会影响在主机名列上有索引的表。例如，mysql系统模式中索引主机名的表现在有一个明确的ROW\_FORMAT属性DYNAMIC，以适应更长的索引值。**
* **一些文件名值的配置设置可能是基于服务器主机名构建的。允许的值受到底层操作系统的限制，它可能不允许文件名长到包括255个字符的主机名。这影响到 general\_log\_file、log\_error、pid\_file、relay\_log 和 slow\_query\_log\_file 系统变量和相应的选项。如果基于主机名的值对操作系统来说太长，必须提供明确的较短的值。**
* **尽管服务器现在支持255个字符的主机名，但使用--ssl-mode=VERIFY\_IDENTITY选项建立的与服务器的连接受到OpenSSL支持的最大主机名长度的限制。主机名匹配与SSL证书的两个字段有关，它们的最大长度如下。通用名称：最大长度为64；主题替代名称：最大长度为RFC#1034。**
* 插件。 **以前，MySQL插件可以用C或C++编写。插件使用的MySQL头文件现在包含C++代码，这意味着插件必须用C++编写，而不是C。**
* C语言API**。 MySQL C API现在支持异步函数，用于与MySQL服务器的非阻塞通信。每个函数都是现有同步函数的异步对应物。如果从服务器连接的读或写必须等待，同步函数就会阻塞。异步函数使应用程序能够检查服务器连接上的工作是否准备好进行。如果不是，应用程序可以在以后再次检查之前执行其他工作。参见C API异步接口。**
* 用于转换的额外目标类型**。 函数CAST()和CONVERT()现在支持转换为DOUBLE、FLOAT和REAL类型。在MySQL 8.0.17中添加。参见第12.11节，"转换函数和操作符"。**
* JSON模式验证**。 MySQL 8.0.17增加了两个函数JSON\_SCHEMA\_VALID()和JSON\_SCHEMA\_VALIDATION\_REPORT()，用于再次验证JSON模式的JSON文档。JSON\_SCHEMA\_VALID()如果文档对模式进行了验证，则返回TRUE(1)，如果没有则返回FALSE(0)。JSON\_SCHEMA\_VALIDATION\_REPORT() 返回一个包含验证结果详细信息的JSON文档。下面的声明适用于这两个函数。**
  + **模式必须符合JSON模式规范的第4稿。**
  + **支持必需的属性。**
  + **不支持外部资源和$ref关键字。**
  + **支持正则表达式模式；无效的模式会被沉默地忽略。**

**更多信息和例子见12.18.7节 "JSON模式验证函数"。**

* 多值索引。 **从MySQL 8.0.17开始，InnoDB支持创建多值索引，这是一个定义在存储数值数组的JSON列上的二级索引，对于一个数据记录可以有多个索引记录。这样的索引使用一个关键部分的定义，如CAST(data->'$.zipcode' AS UNSIGNED ARRAY)。多值索引被MySQL优化器自动用于合适的查询，可以在EXPLAIN的输出中看到。**

**作为这项工作的一部分，MySQL增加了一个新的函数JSON\_OVERLAPS()和一个新的MEMBER OF()操作符，用于处理JSON文档，另外还用一个新的ARRAY关键字扩展了CAST()函数，如以下列表所述。**

* **JSON\_OVERLAPS()比较了两个JSON文档。如果它们包含任何共同的键值对或数组元素，该函数返回TRUE (1)；否则返回FALSE (0)。如果两个值都是标量，该函数执行一个简单的平等测试。如果一个参数是JSON数组，另一个是标量，标量被视为数组元素。因此，JSON\_OVERLAPS()作为JSON\_CONTAINS()的一个补充。**
* **MEMBER OF()测试第一个操作数（标量或JSON文档）是否是作为第二个操作数传递的JSON数组的成员，如果是，返回TRUE (1)，如果不是，返回FALSE (0)。没有对操作数进行类型转换。**
* **CAST(expression AS type ARRAY)允许通过将json\_path的JSON文档中的JSON数组转换为SQL数组来创建一个功能索引。类型指定器仅限于CAST()已经支持的类型，但BINARY除外（不支持）。CAST()的这种用法（以及ARRAY关键字）只被InnoDB支持，并且只用于创建多值索引。**

**关于多值索引的详细信息，包括例子，参见多值索引。第12.18.3节，"搜索JSON值的函数"，提供了关于JSON\_OVERLAPS()和MEMBER OF()的信息，以及使用的例子。**

* 提示表time\_zone**。 从MySQL 8.0.17开始，time\_zone会话变量可以使用SET\_VAR进行提示。**
* redo log归档**。 从MySQL 8.0.17开始，InnoDB支持redo log归档。当备份操作正在进行时，复制redo log记录的备份工具有时可能无法跟上redo log生成的步伐，导致由于这些记录被覆盖而丢失redo log记录。redo log归档功能通过按顺序将redo log记录写入归档文件来解决这个问题。备份工具可以根据需要从归档文件中复制redo log记录，从而避免了潜在的数据丢失。更多信息，请参阅redo log存档。**
* 克隆插件。 **从MySQL 8.0.17开始，MySQL提供一个克隆插件，允许在本地或从远程MySQL服务器实例克隆InnoDB数据。本地克隆操作在MySQL实例运行的同一服务器或节点上存储克隆的数据。远程克隆操作通过网络将克隆的数据从捐赠者的MySQL服务器实例传输到启动克隆操作的接收者服务器或节点上。**

克隆插件支持复制**。除了克隆数据，克隆操作还从捐赠者那里提取和传输复制坐标，并在接收者那里应用这些坐标，这使得使用克隆插件来配置组复制成员和复制。使用克隆插件来配置，比复制大量的事务要快得多，也更有效率。组复制成员也可以被配置为使用克隆插件作为恢复的替代方法，这样成员就会自动选择最有效的方式来检索种子成员的组数据。**

**更多信息，请参阅第5.6.7节 "克隆插件 "和第18.5.3.2节 "分布式恢复的克隆"。**

* 哈希连接优化**。 从MySQL 8.0.18开始，只要连接中的每一对表至少包括一个等价连接条件，并且没有索引适用于任何连接条件，就会使用散列连接。散列连接不需要索引，尽管它可以使用仅适用于单表谓词的索引。散列连接在大多数情况下比块嵌套循环算法更有效。像这里所示的连接可以用这种方式进行优化。**

SELECT \*

FROM t1

JOIN t2

ON t1.c1=t2.c1;

SELECT \*

FROM t1

JOIN t2

ON (t1.c1 = t2.c1 AND t1.c2 < t2.c2)

JOIN t3

ON (t2.c1 = t3.c1)

**哈希连接也可以用于笛卡尔产品--也就是说，当没有指定连接条件时。**

**你可以使用EXPLAIN FORMAT=TREE或EXPLAIN ANALYZE查看散列连接优化何时被用于一个特定的查询。在MySQL 8.0.20和更高版本中，你也可以使用EXPLAIN，省略FORMAT=TREE）。**

**散列连接可用的内存量受 join\_buffer\_size 值的限制。需要超过这一内存的哈希连接在磁盘上执行；磁盘上哈希连接可以使用的磁盘文件的数量由open\_files\_limit限制。**

**从MySQL 8.0.19开始，不再支持MySQL 8.0.18中引入的hash\_join优化器开关（hash\_join=on仍然作为optimizer\_switch值的一部分出现，但设置它不再有任何影响）。HASH\_JOIN和NO\_HASH\_JOIN优化器提示也不再被支持。开关和提示现在都被废弃了；预计它们将在未来的MySQL版本中被删除。在MySQL 8.0.18及以后的版本中，可以使用NO\_BNL优化器开关禁用哈希连接。**

**在MySQL 8.0.20及以后的版本中，在MySQL服务器中不再使用块嵌套循环，并且在以前使用块嵌套循环的任何时候都会采用散列连接，即使查询不包含等价连接条件。这适用于内部非等价连接、半连接、反连接、左外连接和右外连接。优化器开关（optimizer\_switch）系统变量的block\_nested\_loop标志以及BNL和NO\_BNL优化器提示仍然被支持，但今后只控制使用哈希连接。此外，内部和外部连接（包括半连接和反连接）现在都可以采用分批密钥访问（BKA），它可以逐步分配连接缓冲区的内存，这样单个查询就不需要使用大量的资源来解决它们的实际需求。从MySQL 8.0.18开始，只支持内部连接的BKA。**

**MySQL 8.0.20还用迭代器执行器取代了先前版本的MySQL中使用的执行器。这项工作包括替换旧的索引子查询引擎，这些引擎管辖那些没有被优化为半连接的IN查询的WHERE值IN（SELECT column FROM table WHERE ...）形式的查询，以及以前依赖旧执行器的相同形式的查询。**

**更多信息和例子，请参见章节8.2.1.4，"哈希连接优化"。另请参见Batched Key Access Joins。**

* EXPLAIN ANALYZE 语句。 **在MySQL 8.0.18中实现了EXPLAIN语句的一种新形式，即EXPLAIN ANALYZE，为处理查询时使用的每个迭代器提供TREE格式的SELECT语句执行的扩展信息，并使估计成本与查询的实际成本进行比较成为可能。这些信息包括启动成本、总成本、该迭代器返回的行数，以及执行的循环数。**

**在MySQL 8.0.21及以后的版本中，该语句还支持FORMAT=TREE指定器。TREE是唯一支持的格式。**

**参见用EXPLAIN ANALYZE获取信息，以获得更多信息。**

* 查询投射。 **在8.0.18及以后的版本中，MySQL在参数的数据类型和预期数据类型不匹配的表达式和条件中，将投射操作注入查询项目树。这对查询结果或执行速度没有影响，但使执行的查询等同于符合SQL标准的查询，同时保持与MySQL先前版本的向后兼容性。**

**这种隐式转换现在在时间类型（DATE、DATETIME、TIMESTAMP、TIME）和数字类型（SMALLINT、TINYINT、MEDIUMINT、INT/INTEGER、BIGINT；DECIMAL/NUMERIC； FLOAT、DOUBLE、REAL；BIT）之间进行，只要它们使用任何标准数字比较运算符（=、>=、>、<、<=、<>/！=或<=>）来比较。在这种情况下，任何不属于DOUBLE的值都会被铸成一个。现在还对日期或时间值与DATETIME值之间的比较进行了注入，其中参数在必要时被铸成DATETIME。**

**从MySQL 8.0.21开始，当将字符串类型与其他类型进行比较时，也会执行这种转换。被转换的字符串类型包括CHAR、VARCHAR、BINARY、VARBINARY、BLOB、TEXT、ENUM和SET。当将一个字符串类型的值与一个数字类型或YEAR进行比较时，字符串被转换为DOUBLE；如果其他参数的类型不是FLOAT、DOUBLE或REAL，它也被转换为DOUBLE。当一个字符串类型与DATETIME或TIMESTAMP值比较时，字符串被转换为DATETIME；当一个字符串类型与DATE比较时，字符串被转换为DATE。**

**通过查看EXPLAIN ANALYZE、EXPLAIN FORMAT=JSON或如这里所示的EXPLAIN FORMAT=TREE的输出，可以看到何时将转换注入给定查询。**

mysql> **CREATE TABLE d (dt DATETIME, d DATE, t TIME);**

Query OK, 0 rows affected (0.62 sec)

mysql> **CREATE TABLE n (i INT, d DECIMAL, f FLOAT, dc DECIMAL);**

Query OK, 0 rows affected (0.51 sec)

mysql> **CREATE TABLE s (c CHAR(25), vc VARCHAR(25),**

-> **bn BINARY(50), vb VARBINARY(50), b BLOB, t TEXT,**

-> **e ENUM('a', 'b', 'c'), se SET('x' ,'y', 'z'));**

Query OK, 0 rows affected (0.50 sec)

mysql> **EXPLAIN FORMAT=TREE SELECT \* from d JOIN n ON d.dt = n.i\G**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 1. row \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

EXPLAIN: -> Inner hash join *(cast(d.dt as double) = cast(n.i as double))*

(cost=0.70 rows=1)

-> Table scan on n (cost=0.35 rows=1)

-> Hash

-> Table scan on d (cost=0.35 rows=1)

mysql> **EXPLAIN FORMAT=TREE SELECT \* from s JOIN d ON d.dt = s.c\G**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 1. row \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

EXPLAIN: -> Inner hash join *(d.dt = cast(s.c as datetime(6)))* (cost=0.72 rows=1)

-> Table scan on d (cost=0.37 rows=1)

-> Hash

-> Table scan on s (cost=0.35 rows=1)

1 row in set (0.01 sec)

mysql> **EXPLAIN FORMAT=TREE SELECT \* from n JOIN s ON n.d = s.c\G**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 1. row \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

EXPLAIN: -> Inner hash join *(cast(n.d as double) = cast(s.c as double))* (cost=0.70 rows=1)

-> Table scan on s (cost=0.35 rows=1)

-> Hash

-> Table scan on n (cost=0.35 rows=1)

1 row in set (0.00 sec)

**通过执行EXPLAIN [FORMAT=TRADITIONAL]也可以看到这种转换，在这种情况下，在执行EXPLAIN语句后还需要发出SHOW WARNINGS。**

* 对TIMESTAMP和DATETIME的时区支持**。 从MySQL 8.0.19开始，服务器接受一个带有插入日期时间（TIMESTAMP和DATETIME）值的时区偏移。这个偏移量使用的格式与设置time\_zone系统变量时使用的格式相同，只是当偏移量的小时部分小于10时需要一个前导零，并且不允许使用'-00:00'。包含时区偏移的日期时间字段的例子是 "2019-12-11 10:40:30-05:00"，"2003-04-14 03:30:00+10:00"，以及 "2020-01-01 15:35:45+05:30"。**

**在选择日期时间值时，不显示时区偏移。**

**包含时区偏移的日期时间字头可以作为准备好的语句参数值使用。**

**作为这项工作的一部分，用于设置time\_zone系统变量的值现在也被限制在-14:00到+14:00之间，包括在内。(只要MySQL时区表被加载，仍有可能将诸如 "EST"、"Posix/Australia/Brisbane "和 "Europe/Stockholm "这样的名称值分配给这个变量；见填充时区表)。**

**更多信息和例子，见第5.1.15节 "MySQL服务器时区支持"，以及第11.2.2节 "DATE、DATETIME和TIMESTAMP类型"。**

* 关于JSON模式CHECK约束失败的精确信息**。 当使用JSON\_SCHEMA\_VALID()指定一个CHECK约束时，MySQL 8.0.19和更高版本提供了关于这种约束失败的原因的精确信息。**

**有关例子和更多信息，请参见JSON\_SCHEMA\_VALID()和CHECK约束。参见章节13.1.20.6, "CHECK 约束"。**

* 具有ON DUPLICATE KEY UPDATE的行和列别名**。 从MySQL 8.0.19开始，可以使用别名来引用要插入的行，以及可选的列。考虑在一个有列a和b的表t上的以下INSERT语句。**

INSERT INTO t SET a=9,b=5

ON DUPLICATE KEY UPDATE a=VALUES(a)+VALUES(b);

**使用新行的别名new，以及在某些情况下，这个行的列的别名m和n，INSERT语句可以用许多不同的方式重写，这里有一些例子。**

INSERT INTO t SET a=9,b=5 AS new

ON DUPLICATE KEY UPDATE a=new.a+new.b;

INSERT INTO t VALUES(9,5) AS new

ON DUPLICATE KEY UPDATE a=new.a+new.b;

INSERT INTO t SET a=9,b=5 AS new(m,n)

ON DUPLICATE KEY UPDATE a=m+n;

INSERT INTO t VALUES(9,5) AS new(m,n)

ON DUPLICATE KEY UPDATE a=m+n;

**更多信息和例子，请参见章节13.2.6.2, "INSERT ... ON DUPLICATE KEY UPDATE语句"。**

* SQL标准显式表子句和表值构造器**。 根据SQL标准，增加了表值构造器和显式表子句。这些在MySQL 8.0.19中实现，分别为TABLE语句和VALUES语句。**

**TABLE语句的格式是TABLE table\_name，相当于SELECT \* FROM table\_name。它支持ORDER BY和LIMIT子句（后者有可选的OFFSET），但不允许选择单个表列。TABLE可以用在任何可以使用等价的SELECT语句的地方；这包括连接、联合、INSERT ... SELECT、REPLACE、CREATE TABLE ... SELECT语句，以及子查询。例如。**

* **TABLE t1 UNION TABLE t2等同于 SELECT \* FROM t1 UNION SELECT \* FROM t2**
* **CREATE TABLE t2 TABLE t1相当于CREATE TABLE t2 SELECT \* FROM t1**
* **SELECT a FROM t1 WHERE b > ANY (TABLE t2)等同于 SELECT a FROM t1 WHERE b > ANY (SELECT \* FROM t2)。**

**VALUES可以用来为INSERT，REPLACE或SELECT语句提供一个表的值，它由VALUES关键字和一系列用逗号分隔的行构造函数（ROW()）组成。例如，语句INSERT INTO t1 VALUES ROW(1,2,3), ROW(4,5,6), ROW(7,8,9)提供了与MySQL特定的INSERT INTO t1 VALUES (1,2,3), (4,5,6), (7,8,9)相一致的SQL兼容性。你也可以像选择表一样从VALUES表值构造器中选择，记住在这样做的时候你必须提供一个表的别名，并且像使用其他任何东西一样使用这个SELECT；这包括连接、联合和子查询。**

**关于TABLE和VALUES的更多信息，以及它们的使用实例，请参见本文档的以下章节。**

* **第13.2.12节，"TABLE语句"**
* **第13.2.14节，"VALUES语句"**
* **第13.1.20.4节，"CREATE TABLE ... SELECT 语句"**
* **第13.2.6.1节 "INSERT ... SELECT 语句"**
* **第13.2.10.2节 "JOIN条款"**
* **第13.2.11节 "子查询"**
* FORCE INDEX、IGNORE INDEX的优化器提示**。 MySQL 8.0引入了索引级优化器提示，作为第8.9.4节 "索引提示 "中描述的传统索引提示的类似物。这里列出了新的提示，以及它们的FORCE INDEX或IGNORE INDEX的对应物。**
  + **GROUP\_INDEX: 等同于FORCE INDEX FOR GROUP BY**
  + **NO\_GROUP\_INDEX：等同于GROUP BY的忽略索引。**
  + **JOIN\_INDEX: 相当于JOIN的FORCE INDEX**

**NO\_JOIN\_INDEX: 相当于忽略JOIN的INDEX**

* + **ORDER\_INDEX：相当于FORCE INDEX FOR ORDER BY。**

**NO\_ORDER\_INDEX：等同于忽略ORDER BY的INDEX。**

* + **INDEX：与GROUP\_INDEX加JOIN\_INDEX加ORDER\_INDEX相同；相当于FORCE INDEX，没有修饰词。**

**NO\_INDEX：与NO\_GROUP\_INDEX加NO\_JOIN\_INDEX加NO\_ORDER\_INDEX相同；相当于没有修饰符的IGNORE INDEX。**

**例如，下面的两个查询是等价的。**

SELECT a FROM t1 FORCE INDEX (i\_a) FOR JOIN WHERE a=1 AND b=2;

SELECT /\*+ JOIN\_INDEX(t1 i\_a) \*/ a FROM t1 WHERE a=1 AND b=2;

**前面列出的优化器提示遵循与现有的索引级优化器提示相同的语法和用法的基本规则。**

**这些优化器提示旨在取代FORCE INDEX和IGNORE INDEX，我们计划在未来的MySQL版本中弃用这两个提示，并随后从MySQL中删除。它们没有实现与USE INDEX完全等同的单一提示；相反，你可以使用NO\_INDEX、NO\_JOIN\_INDEX、NO\_GROUP\_INDEX或NO\_ORDER\_INDEX中的一个或多个来实现同样的效果。**

**更多信息和使用实例，请参见索引级优化器提示。**

* + JSON\_VALUE()函数**。 MySQL 8.0.21实现了一个新函数JSON\_VALUE()，旨在简化JSON列的索引。在其最基本的形式中，它将一个JSON文档和指向该文档中一个单一值的JSON路径作为参数，以及（可选）允许你用RETURNING关键字指定一个返回类型。JSON\_VALUE(json\_doc, path RETURNING type)等同于此。**

CAST(

JSON\_UNQUOTE( JSON\_EXTRACT(***json\_doc***, ***path***) )

AS ***type***

);

**你也可以指定ON EMPTY、ON ERROR或两个子句，类似于JSON\_TABLE()使用的子句。**

**你可以使用JSON\_VALUE()在JSON列的表达式上创建一个索引，像这样。**

CREATE TABLE t1(

j JSON,

INDEX i1 ( (JSON\_VALUE(j, '$.id' RETURNING UNSIGNED)) )

);

INSERT INTO t1 VALUES ROW('{"id": "123", "name": "shoes", "price": "49.95"}');

**使用这个表达式的查询，如这里所示，可以利用索引。**

SELECT name, price FROM t1

WHERE JSON\_VALUE(j, '$.id' RETURNING UNSIGNED) = 123;

**在很多情况下，这比从JSON列创建一个生成的列，然后在生成的列上创建一个索引要简单。**

**更多信息和例子，请看JSON\_VALUE()的描述。**

* + 账户描述和用户属性**。 MySQL 8.0.21引入了在创建或更新用户账户时设置账户描述和用户属性的能力。账户描述由任意的文本组成，作为参数传递给与CREATE USER或ALTER USER语句一起使用的COMMENT子句。用户属性由JSON对象形式的数据组成，作为ATTRIBUTE子句的参数，用于这两个语句中的一个。该属性可以包含任何有效的JSON对象符号的键值对。在一个CREATE USER或ALTER USER语句中只能使用COMMENT或ATTRIBUTE中的一个。**

**账户描述和用户属性在内部作为一个JSON对象一起存储，评论文本作为以评论为键的元素的值。这个信息可以从INFORMATION\_SCHEMA.USER\_ATTRIBUTES表的ATTRIBUTE列中检索；因为它是JSON格式的，你可以使用MySQL的JSON函数和操作符来解析它的内容（见12.18节，"JSON函数"）。对用户属性的连续变化与它的当前值合并，就像使用JSON\_MERGE\_PATCH()函数时一样。**

**例子。**

mysql> **CREATE USER 'mary'@'localhost' COMMENT 'This is Mary Smith\'s account';**

Query OK, 0 rows affected (0.33 sec)

mysql> **ALTER USER 'mary'@'localhost'**

-≫ **ATTRIBUTE '{"fname":"Mary", "lname":"Smith"}';**

Query OK, 0 rows affected (0.14 sec)

mysql> **ALTER USER 'mary'@'localhost'**

-≫ **ATTRIBUTE '{"email":"mary.smith@example.com"}';**

Query OK, 0 rows affected (0.12 sec)

mysql> **SELECT**

-> **USER,**

-> **HOST,**

-> **ATTRIBUTE->>"$.fname" AS 'First Name',**

-> **ATTRIBUTE->>"$.lname" AS 'Last Name',**

-> **ATTRIBUTE->>"$.email" AS 'Email',**

-> **ATTRIBUTE->>"$.comment" AS 'Comment'**

-> **FROM INFORMATION\_SCHEMA.USER\_ATTRIBUTES**

-> **WHERE USER='mary' AND HOST='localhost'\G**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 1. row \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

USER: mary

HOST: localhost

First Name: Mary

Last Name: Smith

Email: mary.smith@example.com

Comment: This is Mary Smith's account

1 row in set (0.00 sec)

**更多信息和例子，请参见第13.7.1.3节 "CREATE USER语句"，第13.7.1.1节 "ALTER USER语句"，以及第26.3.46节 "INFORMATION\_SCHEMA USER\_ATTRIBUTES表"。**

* + 新的optimizer\_switch标志**。 MySQL 8.0.21为optimizer\_switch系统变量增加了两个新的标志，如以下列表中所述。**
    - **prefer\_ordering\_index标志**

**默认情况下，只要优化器确定有LIMIT子句的任何ORDER BY或GROUP BY查询会导致更快的执行，MySQL就会尝试对其使用有序索引。因为在某些情况下，为这种查询选择不同的优化实际上可能表现得更好，现在可以通过将prefer\_ordering\_index标志设置为关闭来禁用这种优化。**

**这个标志的默认值是开。**

* + subquery\_to\_derived标志

**当这个标志被设置为on时，优化器将符合条件的标量子查询转换为派生表的连接。例如，查询SELECT \* FROM t1 WHERE t1.a > (SELECT COUNT(a) FROM t2) 被改写为SELECT t1.a FROM t1 JOIN ( SELECT COUNT(t2.a) AS c FROM t2 ) AS d WHERE t1.a > d.c。**

**这个优化可以应用于一个子查询，它是SELECT, WHERE, JOIN或HAVING子句的一部分；包含一个或多个聚合函数，但没有GROUP BY子句；不是相关的；并且没有使用任何非确定性函数。**

**该优化也可以应用于表子查询，该查询是IN、NOT IN、EXISTS或NOT EXISTS的参数，并且不包含GROUP BY。例如，查询SELECT \* FROM t1 WHERE t1.b < 0 OR t1.a IN (SELECT t2.a + 1 FROM t2) 被改写为SELECT a, b FROM t1 LEFT JOIN (SELECT DISTINCT 1 AS e1, t2.a AS e2 FROM t2) d ON t1.a + 1 = d.e2 WHERE t1.b < 0 OR d.e1 IS NOT NULL。**

**从MySQL 8.0.24开始，这种优化也可以应用于相关的标量子查询，方法是对它应用一个额外的分组，然后对解除的谓词进行外连接。例如，像SELECT \* FROM t1 WHERE (SELECT a FROM t2 WHERE t2.a=t1.a) > 0这样的查询可以改写为SELECT t1.\* FROM t1 LEFT OUTER JOIN (SELECT a, COUNT(\*) AS ct FROM t2 GROUP BY a) AS derived ON t1. MySQL执行了一个cardinality检查，以确保子查询不会返回多于一条记录（ER\_SUBQUERY\_NO\_1\_ROW）。参见第13.2.11.7节 "相关子查询"，以了解更多信息。**

**这个优化通常是禁用的，因为它在大多数情况下不会产生明显的性能优势；这个标志在默认情况下被设置为关闭。**

**更多信息请参见第8.9.2节，"可切换的优化"。另请参见第8.2.1.19节 "LIMIT查询优化"，第8.2.2.1节 "用半连接转换优化IN和EXISTS子查询谓词"，以及第8.2.2.4节 "用合并或物化优化派生表、视图引用和普通表表达式"。**

* + XML的增强**。 从MySQL 8.0.21开始，LOAD XML语句现在支持要导入的XML中的CDATA部分。**
  + 现在支持对YEAR类型的投递**。 从MySQL 8.0.22开始，服务器允许铸造到YEAR。CAST()和CONVERT()函数都支持个位数、两位数和四位数的YEAR值。对于1位数和2位数的值，允许的范围是0-99。四位数的值必须在1901-2155范围内。YEAR也可以作为JSON\_VALUE()函数的返回类型；这个函数只支持四位数的年份。**

**字符串、时间和日期以及浮点值都可以被转换为YEAR。不支持将GEOMETRY值转换为YEAR。**

**更多信息，包括转换规则，见CONVERT()函数的描述。**

* 检索作为UTC的TIMESTAMP值。 **MySQL 8.0.22及更高版本支持在检索时将TIMESTAMP列值从系统时区转换为UTC DATETIME，使用CAST(value AT TIME ZONE specifier AS DATETIME)，其中指定符是[INTERVAL] '+00:00'或'UTC'之一。如果需要的话，可以指定铸模返回的DATETIME值的精度，最多可达小数点后6位。这个结构不支持ARRAY关键字。**

**也支持使用时区偏移将TIMESTAMP值插入到一个表中。对于CONVERT()或任何其他MySQL函数或结构，不支持使用AT TIME ZONE。**

**有关进一步的信息和例子，请参阅CAST()函数的描述。**

* 倾倒文件的输出同步**。 当通过SELECT INTO DUMPFILE和SELECT INTO OUTFILE语句写入文件时，MySQL 8.0.22及更高版本支持定期同步。这可以通过将select\_into\_disk\_sync系统变量设置为ON来启用；写缓冲区的大小由select\_into\_buffer\_size设置的值决定；默认是131072（217）字节。**

**此外，可以使用select\_into\_disk\_sync\_delay来设置同步到磁盘后的可选延迟；默认是没有延迟（0毫秒）。**

**欲了解更多信息，请参见本项目之前提到的变量描述。**

* 语句的单次准备**。 从MySQL 8.0.22开始，准备好的语句只准备一次，而不是每次执行时准备一次。这是在执行PREPARE时进行的。这对于存储过程中的任何语句也是如此；当存储过程第一次执行时，该语句被准备一次。**

**这个变化的一个结果是，在准备好的语句中使用的动态参数的解决方式也以这里列出的方式改变了。**

* **准备好的语句参数在准备语句时被分配了一个数据类型；该类型在语句的每次后续执行中都持续存在（除非语句被重新准备；见下文）。**

**在准备好的语句中为一个给定的参数或用户变量使用不同的数据类型来执行第一次执行后的语句，可能会导致语句被重新准备；由于这个原因，在重新执行准备好的语句时，最好为一个给定的参数使用相同的数据类型。**

* **为了与SQL标准保持一致，以下采用窗口函数的结构不再被接受。**
* **NTILE(NULL)**
* **NTH\_VALUE(expr, NULL)**
* **LEAD(expr, nn)和LAG(expr, nn)，其中nn是一个负数。**

**这有利于更加符合SQL标准。更多细节请参见各个函数的描述。**

* **现在，在准备好的语句中引用的用户变量，其数据类型在语句准备时就已确定；该类型在语句的每次后续执行中都会持续存在。**
* **在一个存储过程中出现的语句所引用的用户变量现在在第一次执行该语句时确定其数据类型；该类型在随后的任何包含存储过程的调用中持续存在。**
* **当执行形式为SELECT expr1, expr2, ...的准备好的语句时。FROM表ORDER BY ? 时，为参数传递一个整数值N不再导致按选择列表中的第N个表达式对结果进行排序；结果不再被排序，正如对ORDER BY常数的预期。**

**对作为准备好的语句或在存储过程中使用的语句只准备一次，可以提高该语句的性能，因为它否定了重复准备的附加成本。这样做还可以避免准备结构可能的多次回滚，这一直是MySQL中许多问题的来源。**

**更多信息，见第13.5.1节 "PREPARE语句"。**

* RIGHT JOIN作为LEFT JOIN处理**。 从MySQL 8.0.22开始，服务器在内部将RIGHT JOIN的所有实例处理为LEFT JOIN，消除了一些在解析时没有进行完全转换的特殊情况。**
* 派生条件推倒优化。 **MySQL 8.0.22（及更高版本）为具有物化派生表的查询实现了派生条件推倒。对于诸如SELECT \* FROM (SELECT i, j FROM t1) AS dt WHERE i > constant这样的查询，现在在许多情况下可以将外层的WHERE条件下推到派生表，在这种情况下会产生SELECT \* FROM (SELECT i, j FROM t1 WHERE i > constant) AS dt。**

**以前，如果派生表被物化而没有合并，MySQL会物化整个表，然后用WHERE条件限定行。使用派生条件推倒优化将WHERE条件移到子查询中，通常可以减少必须处理的行的数量，这可以减少执行查询所需的时间。**

**当派生表不使用任何聚合或窗口函数时，一个外部的WHERE条件可以直接下推到一个物化派生表。当派生表有一个GROUP BY并且不使用任何窗口函数时，外部WHERE条件可以作为一个HAVING条件被推送到派生表。当派生表使用窗口函数，并且外部WHERE引用窗口函数的PARTITION子句中使用的列时，WHERE条件也可以向下推送。**

**派生条件推倒是默认启用的，如optimizer\_switch系统变量的派生\_条件\_推倒标志所指示。该标志是在MySQL 8.0.22中添加的，默认情况下被设置为打开；要为特定查询禁用优化，可以使用NO\_DERIVED\_CONDITION\_PUSHDOWN优化器提示（也在MySQL 8.0.22中添加）。如果由于派生条件推倒被设置为关闭而导致优化被禁用，你可以使用DERIVED\_CONDITION\_PUSHDOWN为特定查询启用它。**

**派生条件推倒优化不能用于包含UNION或LIMIT子句的派生表。此外，一个本身使用子查询的条件不能被推倒，一个WHERE条件不能被推倒到也是外连接的内表的派生表上。关于其他的信息和例子，请看第8.2.2.5节，"派生条件推倒优化"**

* 对MySQL授予表的非锁定读取。 **从MySQL 8.0.22开始，为了允许在MySQL授权表上进行并发的DML和DDL操作，以前在MySQL授权表上获得行锁的读取操作被作为非锁定读取执行。**

**现在在MySQL授予表上作为非锁定读取执行的操作包括。**

* **通过连接列表和子查询从授予表读取数据的SELECT语句和其他只读语句，包括SELECT ... FOR SHARE语句，使用任何事务隔离级别。**
* **使用任何事务隔离级别，从授予表（通过连接列表或子查询）读取数据但不修改数据的DML操作。**

**有关其他信息，请参见授予表的并发性。**

**在MySQL 8.0中废弃的功能**

以下功能在MySQL 8.0中被废弃，并可能在未来的系列中被删除。在显示替代功能的地方，应更新应用程序以使用它们。

对于使用在MySQL 8.0中被废弃的功能的应用程序，如果在更高的MySQL系列中被删除，当从MySQL 8.0源复制到更高系列的副本时，语句可能会失败，或者可能对源和副本产生不同的影响。为了避免这类问题，应修改使用在8.0中被废弃的功能的应用程序，以避免它们，并尽可能使用替代方案。

* utf8mb3字符集已被废弃。请使用utf8mb4代替。
* 由于caching\_sha2\_password是MySQL 8.0中的默认认证插件，并提供了sha256\_password认证插件的功能超集，因此sha256\_password已被废弃；预计它将在MySQL的未来版本中被移除。使用sha256\_password认证的MySQL账户应迁移到使用caching\_sha2\_password来代替。
* validate\_password插件已被重新实现，以使用组件基础设施。validate\_password的插件形式仍然可用，但现在已被废弃；预计它将在未来的MySQL版本中被删除。使用该插件的MySQL安装应过渡到使用组件。见第6.4.3.3节，"过渡到密码验证组件"
* ALTER TABLESPACE和DROP TABLESPACE语句的ENGINE子句已被废弃。
* PAD\_CHAR\_TO\_FULL\_LENGTH SQL模式已被废弃。
* 对于FLOAT和DOUBLE类型的列（以及任何同义词）的AUTO\_INCREMENT支持已被废弃。考虑从这些列中移除AUTO\_INCREMENT属性，或者将其转换为整数类型。
* 对于FLOAT、DOUBLE和DECIMAL类型的列（以及任何同义词），UNSIGNED属性已被废弃。对于这样的列，可以考虑使用一个简单的CHECK约束来代替。
* FLOAT(M,D)和DOUBLE(M,D)语法用于为FLOAT和DOUBLE类型的列（以及任何同义词）指定数字数量，是一个非标准的MySQL扩展。这种语法已被废弃。
* 对于数字数据类型，ZEROFILL属性已被废弃，整数数据类型的显示宽度属性也被废弃。考虑使用其他方法来产生这些属性的效果。例如，应用程序可以使用LPAD()函数将数字零填充到所需的宽度，或者他们可以将格式化的数字存储在CHAR列。
* 对于字符串数据类型，BINARY属性是一个非标准的MySQL扩展，它是指定列字符集（如果没有指定列字符集，则是表默认字符集）的二进制（\_bin）整理的速记。在MySQL 8.0中，BINARY的这种非标准使用是不明确的，因为utf8mb4字符集有多种\_bin校对，所以BINARY属性已被废弃；预计在MySQL的未来版本中会删除对它的支持。应用程序应调整为使用明确的\_bin校对。
* 使用BINARY来指定数据类型或字符集的做法保持不变。
* 非标准的C风格的&&、||和！运算符分别是标准SQL的AND、OR和NOT运算符的同义词，已被废弃。使用非标准操作符的应用程序应该被调整为使用标准操作符。

注意

除非启用了PIPES\_AS\_CONCAT SQL模式，否则||的使用已被弃用。在这种情况下，||标志着SQL标准的字符串连接操作符）。)

* JSON\_MERGE()函数已被废弃。请使用JSON\_MERGE\_PRESERVE()代替。
* SQL\_CALC\_FOUND\_ROWS查询修饰符和附带的FOUND\_ROWS()函数已被弃用。关于替代策略的信息，请参见FOUND\_ROWS()描述。
* 从MySQL 8.0.13开始，对TABLESPACE = innodb\_file\_per\_table和TABLESPACE = innodb\_temporary子句与CREATE TEMPORARY TABLE的支持已被弃用。
* 对于SELECT语句，从MySQL 8.0.20开始，在FROM后面而不是在SELECT的末尾使用INTO子句已被弃用。最好将INTO放在语句的末尾。
* 对于UNION语句，从MySQL 8.0.20开始，包含INTO的这两个变体被弃用。
* 在一个查询表达式的尾部查询块中，在FROM之前使用INTO。
* 在查询表达式的尾部括号内，使用INTO，无论其相对于FROM的位置如何。参见章节13.2.10.1, "SELECT ... INTO语句 "和章节13.2.10.3, "UNION语句"。
* 从MySQL 8.0.23开始，FLUSH HOSTS被弃用。相反，截断性能模式host\_cache表。

TRUNCATE TABLE performance\_schema.host\_cache;

TRUNCATE TABLE操作需要该表的DROP权限。

* mysql\_upgrade客户端已被废弃，因为其升级mysql系统模式中的系统表和其他模式中的对象的功能已被移到MySQL服务器中。参见第2.11.3节，"MySQL升级过程升级的内容"
* --no-dd-upgrade服务器选项已被弃用。它被--upgrade选项所取代，后者对数据字典和服务器升级行为提供了更精细的控制。
* mysql\_upgrade\_info文件是创建数据目录并用于存储MySQL版本号的，已被废弃；预计它将在MySQL的未来版本中被删除。
* relay\_log\_info\_file系统变量和-master-info-file选项已被弃用。以前，当relay\_log\_info\_repository=FILE和master\_info\_repository=FILE被设置时，这些被用来指定中继日志信息日志和源信息日志的名称，但这些设置已被弃用。为中继日志信息日志和源信息日志使用文件已经被崩溃安全的复制表所取代，在MySQL 8.0中是默认的。
* max\_length\_for\_sort\_data系统变量现在已被废弃，因为优化器的变化使其过时且无影响。
* 这些用于压缩与服务器的连接的传统参数已被废弃。--compress客户端命令行选项；mysql\_options()C API函数的MYSQL\_OPT\_COMPRESS选项；slave\_compressed\_protocol系统变量。关于要使用的参数的信息，见第4.2.8节，"连接压缩控制"。
* 使用MYSQL\_PWD环境变量来指定MySQL密码的做法已被弃用。
* 在INSERT ... ON DUPLICATE KEY UPDATE中使用VALUES()来访问新行的值，从MySQL 8.0.20开始已被废弃。使用新行和列的别名来代替。
* 因为在调用JSON\_TABLE()时，在ON EMPTY之前指定ON ERROR是违反SQL标准的，这种语法现在在MySQL中被废弃。从MySQL 8.0.20开始，只要你试图这样做，服务器就会打印出一个警告。当在单个JSON\_TABLE()调用中指定这两个子句时，确保首先使用ON EMPTY。
* 带有索引前缀的列从未被支持为表的分区键的一部分；以前，在创建、改变或升级分区表时，这些列是被允许的，但被表的分区函数排除在外，而且服务器不会发出发生这种情况的警告。这种允许的行为现在已被废弃，并将在未来的MySQL版本中被删除，在分区键中使用任何这样的列会导致它们发生的CREATE TABLE或ALTER TABLE语句被拒绝。

从MySQL 8.0.21开始，只要使用索引前缀的列被指定为分区键的一部分，就会为每个这样的列产生一个警告。每当CREATE TABLE或ALTER TABLE语句因为提议的分区键中的所有列都有索引前缀而被拒绝时，产生的错误现在提供了拒绝的确切原因。在这两种情况下，这包括在分区函数中使用的列通过采用空的PARTITION BY KEY()子句被隐含地定义为表的主键的情况。

更多信息和例子，请看列索引前缀不支持键分区。

* 从MySQL 8.0.22开始，InnoDB memcached插件已被废弃；预计在未来的MySQL版本中会移除对它的支持。
* 从MySQL 8.0.25开始，temptable\_use\_mmap变量已被废弃；预计在未来的MySQL版本中会移除对它的支持。

**在MySQL 8.0中删除的功能**

以下项目已经过时，并在MySQL 8.0中被删除。在显示替代方案的地方，应用程序应更新以使用它们。

对于使用MySQL 8.0中删除的特性的MySQL 5.7应用程序，当从MySQL 5.7源复制到MySQL 8.0副本时，语句可能会失败，或者可能对源和副本产生不同的影响。为避免此类问题，应修改使用MySQL 8.0中删除的特性的应用程序，以避免它们，并尽可能使用替代方案。

* innodb\_locks\_unsafe\_for\_binlog系统变量被删除。READ COMMITTED隔离级别提供了类似的功能。
* 在MySQL 8.0.0中引入的information\_schema\_stats变量被删除，并在MySQL 8.0.3中被information\_schema\_stats\_expiry取代。
* information\_schema\_stats\_expiry定义了缓存的INFORMATION\_SCHEMA表统计数据的过期设置。更多信息，见第8.2.3节，"优化INFORMATION\_SCHEMA查询"。
* 与过时的InnoDB系统表有关的代码在MySQL 8.0.3中被删除。基于InnoDB系统表的INFORMATION\_SCHEMA视图被数据字典表的内部系统视图取代。受影响的InnoDB INFORMATION\_SCHEMA视图被重新命名。

**Table1.1 Renamed InnoDB Information Schema Views**

|  |  |
| --- | --- |
| **Old Name** | **New Name** |
| **INNODB\_SYS\_COLUMNS** | **INNODB\_COLUMNS** |
| **INNODB\_SYS\_DATAFILES** | **INNODB\_DATAFILES** |
| **INNODB\_SYS\_FIELDS** | **INNODB\_FIELDS** |
| **INNODB\_SYS\_FOREIGN** | **INNODB\_FOREIGN** |
| **INNODB\_SYS\_FOREIGN\_COLS** | **INNODB\_FOREIGN\_COLS** |
| **INNODB\_SYS\_INDEXES** | **INNODB\_INDEXES** |
| **INNODB\_SYS\_TABLES** | **INNODB\_TABLES** |
| **INNODB\_SYS\_TABLESPACES** | **INNODB\_TABLESPACES** |
| **INNODB\_SYS\_TABLESTATS** | **INNODB\_TABLESTATS** |
| **INNODB\_SYS\_VIRTUAL** | **INNODB\_VIRTUAL** |

在升级到MySQL 8.0.3或更高版本后，更新任何引用以前InnoDB INFORMATION\_SCHEMA视图名称的脚本。

* 以下与账户管理有关的功能被删除。
  + 使用GRANT来创建用户。相反，使用CREATE USER。遵循这种做法使得NO\_AUTO\_CREATE\_USER SQL模式对GRANT语句不重要，所以它也被删除了，并且当选项文件中sql\_mode选项的这个值阻止mysqld启动时，现在会有一个错误被写入服务器日志。
  + 使用GRANT来修改权限分配以外的账户属性。这包括认证、SSL和资源限制属性。相反，在创建账户时用CREATE USER建立这些属性，或在之后用ALTER USER修改它们
  + CREATE USER和GRANT的IDENTIFIED BY PASSWORD 'auth\_string' 语法。相反，在CREATE USER和ALTER USER中使用IDENTIFIED WITH auth\_plugin AS 'auth\_string'，其中'auth\_string'的值是与命名的插件兼容的格式。
  + 此外，由于IDENTIFIED BY PASSWORD语法被删除，log\_builtin\_as\_identified\_by\_password系统变量是多余的，被删除。
  + PASSWORD()函数。此外，PASSWORD()的移除意味着SET PASSWORD ... = PASSWORD('auth\_string')语法不再可用。
  + old\_passwords系统变量。
* 查询缓存被移除。移除包括这些项目。
  + FLUSH QUERY CACHE和RESET QUERY CACHE语句。
  + 这些系统变量：query\_cache\_limit, query\_cache\_min\_res\_unit, query\_cache\_size, query\_cache\_type, query\_cache\_wlock\_invalidate。
  + 这些状态变量。Qcache\_free\_blocks, Qcache\_free\_memory, Qcache\_hits, Qcache\_inserts, Qcache\_lowmem\_prunes, Qcache\_not\_cached, Qcache\_queries\_in\_cache, Qcache\_total\_blocks。
  + 这些线程状态：检查缓存查询的权限，检查查询缓存的查询，无效查询缓存条目，发送缓存结果到客户端，将结果存储在查询缓存中，等待查询缓存锁。
  + SQL\_CACHE SELECT修改器。

这些被废弃的查询缓存项目仍然被废弃，但没有影响；预计它们将在未来的MySQL版本中被移除。

* SQL\_NO\_CACHE SELECT修改器。
* ndb\_cache\_check\_time系统变量。

have\_query\_cache系统变量仍然是废弃的，并且总是有一个NO的值；预计它将在未来的MySQL版本中被删除。

* 数据字典提供了关于数据库对象的信息，因此服务器不再检查数据目录中的目录名来寻找数据库。因此，--ignore-db-dir选项和ignore\_db\_dirs系统变量是多余的，并被删除。
* DDL日志，也被称为元数据日志，已被删除。从MySQL 8.0.3开始，该功能由数据字典innodb\_ddl\_log表处理。参见查看DDL日志。
* tx\_isolation和tx\_read\_only系统变量已被删除。请使用 transaction\_isolation 和 transaction\_read\_only 来代替。
* sync\_frm 系统变量已经被删除， 因为 .frm 文件已经过时。
* 移除了 secure\_auth 系统变量和 --secure-auth 客户端选项。mysql\_options() C API函数的MYSQL\_SECURE\_AUTH选项已被删除。
* 删除了multi\_range\_count系统变量。
* 移除了log\_warnings系统变量和-log-warnings服务器选项。使用 log\_error\_verbosity 系统变量来代替。
* 删除了 sql\_log\_bin 系统变量的全局范围。 sql\_log\_bin 只有会话范围，依赖访问 @@GLOBAL.sql\_log\_bin 的应用程序应该被调整。
* 删除了 metadata\_locks\_cache\_size 和 metadata\_locks\_hash\_instances 系统变量。
* 删除了未使用的 date\_format、datetime\_format、time\_format 和 max\_tmp\_tables 系统变量。
* 删去了这些已经废弃的兼容SQL模式。DB2, maxdb, mssql, mysql323, mysql40, oracle, postgresql, no\_field\_options, no\_key\_options, no\_table\_options。它们不能再被分配给sql\_mode系统变量或作为mysqldump --兼容选项的允许值。
* 移除MAXDB意味着CREATE TABLE或ALTER TABLE的TIMESTAMP数据类型被视为TIMESTAMP，而不再被视为DATETIME。
* 删除了GROUP BY子句中被废弃的ASC或DESC限定符。以前依赖GROUP BY排序的查询可能产生与以前的MySQL版本不同的结果。为了产生一个给定的排序顺序，提供一个ORDER BY子句。
* EXPLAIN语句的EXTENDED和PARTITIONS关键字已被删除。这些关键字是不必要的，因为它们的效果总是被启用。
* 删除了这些与加密有关的项目。
  + ENCODE()和DECODE()函数。
  + ENCRYPT()函数。
  + DES\_ENCRYPT()和DES\_DECRYPT()函数，--des-key-file选项，have\_crypt系统变量，FLUSH语句的DES\_KEY\_FILE选项，以及HAVE\_CRYPT CMake选项。
* **来代替被删除的加密函数。对于ENCRYPT()，考虑使用SHA2()来代替单向散列。对于其他的，考虑使用AES\_ENCRYPT()和AES\_DECRYPT()代替。**
* **在MySQL 5.7中，多个名称下的空间函数被废弃，以使空间函数名称空间更加一致，目标是每个空间函数名称以ST\_开头，如果它执行精确操作，或以MBR开头，如果它执行基于最小边界矩形的操作。在MySQL 8.0中，被废弃的函数被删除，只留下相应的ST\_和MBR函数。**
  + **这些函数被删除，以支持MBR的名称。Contains(), Disjoint(), Equals(), Intersects(), Overlaps(), Within()。**
  + **这些函数被删除，改用ST\_名称。Area(), AsBinary(), AsText(), AsWKB(), AsWKT(), Buffer(), Centroid(), ConvexHull(), Crosses(), Dimension(), Distance(), EndPoint(), Envelope(), ExteriorRing(), GeomCollFromText(), GeomCollFromWKB(), GeomFromText(), GeomFromWKB() 。GeometryCollectionFromText(), GeometryCollectionFromWKB(), GeometryFromText(), GeometryFromWKB(), GeometryN(), GeometryType(), InteriorRingN(), IsClosed(), IsEmpty(), IsSimple(), LineFromText(), LineFromWKB(), LineStringFromText() 。LineStringFromWKB(), MLineFromText(), MLineFromWKB(), MPPointFromText(), MPPointFromWKB(), MPolyFromText(), MPolyFromWKB(), MultiLineStringFromText(), MultiLineStringFromWKB(), MultiPointFromText(), MultiPointFromWKB(), MultiPolygonFromText() 。MultiPolygonFromWKB(), NumGeometries(), NumInteriorRings(), NumPoints(), PointFromText(), PointFromWKB(), PointN(), PolyFromText(), PolyFromWKB(), PolygonFromText(), PolygonFromWKB(), SRID(), StartPoint(), Touches(), X(), Y()。**
  + **移除GLength()，改用ST\_Length()。**
* **第12.17.4节 "从WKB值创建几何值的函数 "中描述的函数以前接受WKB字符串或几何参数。几何参数不再被允许，并产生一个错误。请参阅该节，了解不再使用几何参数的查询指南。**
* **在SQL语句中，解析器不再将 \N 作为NULL的同义词。使用NULL代替。**

**这个变化不影响用LOAD DATA或SELECT ... INTO OUTFILE进行的文本文件导入或导出操作，对于这些操作，NULL继续用 \N 表示。参见第13.2.7节，"LOAD DATA语句"。**

* **PROCEDURE ANALYSE()语法被删除。**
* **删除了客户端的 --ssl 和 --ssl-verify-server-cert 选项。使用 --ssl-mode=REQUIRED 而不是 --ssl=1 或 --enable-ssl。使用 --ssl-mode=DISABLED 来代替 --ssl=0, --skip-ssl, 或 --disable-ssl。使用 --ssl-mode=VERIFY\_IDENTITY 而不是 --ssl-verify-server-cert 选项。(服务器端的--ssl选项仍然可用，但从MySQL 8.0.25开始已被废弃，并将在未来的MySQL版本中被删除)。**

**对于C API，mysql\_options()的MYSQL\_OPT\_SSL\_ENFORCE和MYSQL\_OPT\_SSL\_VERIFY\_SERVER\_CERT选项对应于客户端的--ssl和--ssl-verify-server-cert选项，并被删除。使用MYSQL\_OPT\_SSL\_MODE，选项值为SSL\_MODE\_REQUIRED或SSL\_MODE\_VERIFY\_IDENTITY来代替。**

* **删除了--temp-pool服务器选项。**
* **移除ignore\_builtin\_innodb系统变量。**
* **服务器不再执行将包含特殊字符的前MySQL 5.1数据库名称转换为添加了#mysql50#前缀的5.1格式。由于不再执行这些转换，mysqlcheck的--fix-db-names和--fix-table-names选项、ALTER DATABASE语句的UPGRADE DATA DIRECTORY NAME子句和Com\_alter\_db\_upgrade状态变量被删除。**
* **只支持从一个主要版本升级到另一个主要版本（例如，从5.0到5.1，或从5.1到5.5），所以应该没有什么剩余的需要将旧的5.0数据库名称转换到当前版本的MySQL。作为一种变通方法，在升级到较新的版本之前，将MySQL 5.0安装升级到MySQL 5.1。**
* **mysql\_install\_db程序已从MySQL发行版中删除。数据目录的初始化应该通过调用mysqld的-initialize或-initialize-insecure选项来进行，而不是通过调用mysqld。此外，mysql\_install\_db使用的mysqld的--bootstrap选项被移除，控制mysql\_install\_db安装位置的INSTALL\_SCRIPTDIR CMake选项被移除。**
* **从MySQL服务器中删除了通用分区处理程序。为了支持给定表的分区，用于该表的存储引擎现在必须提供它自己的（"本地"）分区处理程序。从MySQL服务器中删除了--partition和--skip-partition选项，并且在SHOW PLUGINS的输出或INFORMATION\_SCHEMA.PLUGINS表中不再显示与分区相关的条目。**

**目前有两个MySQL存储引擎提供本地分区支持。InnoDB和NDB。其中，只有InnoDB在MySQL 8.0中被支持。在MySQL 8.0中使用任何其他存储引擎创建分区表的尝试都会失败**

**对升级的影响。** 不支持使用InnoDB以外的存储引擎（如MyISAM）将分区表从MySQL 5.7（或更早）直接升级到MySQL 8.0。处理这样的表有两个选择。

* 删除表的分区，使用ALTER TABLE ... 删除分区
* **将表使用的存储引擎改为InnoDB，使用ALTER TABLE ... ENGINE=INNODB。**

**在将服务器升级到MySQL 8.0之前，必须对每个分区的非InnoDB表执行刚才列出的两个操作中的至少一个。否则，这样的表在升级后不能使用。**

由于会导致使用不支持分区的存储引擎的分区表的表创建语句现在以错误（ER\_CHECK\_NOT\_IMPLEMENTED）失败，你必须确保你希望导入到MySQL 8.0服务器的旧版本的MySQL的转储文件（例如由mysqldump编写的）中创建分区表的任何语句不会指定诸如MyISAM这样没有本地分区处理器的存储引擎。你可以通过执行以下任何一项来做到这一点。

* **从CREATE TABLE语句中删除对分区的任何引用，这些语句使用的STORAGE ENGINE选项的值不是InnoDB。**
* **将存储引擎指定为InnoDB，或者允许InnoDB被默认为表的存储引擎。**

**更多信息请参见第24.6.2节，"与存储引擎有关的分区限制"。**

* 在 INFORMATION\_SCHEMA 中不再维护系统和状态变量信息。这些表被移除。global\_variables, session\_variables, global\_status, session\_status。使用相应的性能模式表来代替。参见第27.12.14节 "性能模式系统变量表 "和第27.12.15节 "性能模式状态变量表"。此外， show\_compatibility\_56 系统变量被删除了。在过渡时期，INFORMATION\_SCHEMA表中的系统和状态变量信息被转移到Performance Schema表**中**，因此不再需要它。这些状态变量被移除。Slave\_heartbeat\_period、Slave\_last\_heartbeat、Slave\_received\_heartbeats、Slave\_retried\_transactions、Slave\_running。它们提供的信息可以在Performance Schema表中找到；请看迁移到Performance Schema系统和状态变量表。
* **性能模式的 setup\_timers 表被删除了，performance\_timers 表中的 TICK 行也被删除了。**
* **libmysqld嵌入式服务器库被移除，同时也被移除。**
  + **mysql\_options() MYSQL\_OPT\_GUESS\_CONNECTION、MYSQL\_OPT\_USE\_EMBEDDED\_CONNECTION、MYSQL\_OPT\_USE\_REMOTE\_CONNECTION和MYSQL\_SET\_CLIENT\_IP选项**
  + **mysql\_config --libmysqld-libs, --embedded-libs, 和 --embedded选项。**
  + **CMake的WITH\_EMBEDDED\_SERVER、WITH\_EMBEDDED\_SHARED\_LIBRARY和INSTALL\_SECURE\_FILE\_PRIV\_EMBEDDEDDIR选项。**
  + **mysql --server-arg选项（未记录）。**
  + **mysqltest --embedded-server, --server-arg, and --server-file选项**
  + **mysqltest\_embedded和mysql\_client\_test\_embedded测试程序**
* **mysql\_plugin工具被移除。替代品包括在服务器启动时使用--plugin-load或--plugin-load-add选项加载插件，或者在运行时使用INSTALL PLUGIN语句。**
* 移除 resolveip 工具。可以使用 nslookup、host 或 dig 来代替**。**
* resolve\_stack\_dump工具已被删除。来自官方MySQL构建的堆栈跟踪总是符号化的，因此没有必要使用 resolve\_stack\_dump**。**
* 以下服务器错误代码不被使用，并已被删除。专门测试任何这些错误的应用程序应该被更新。

ER\_BINLOG\_READ\_EVENT\_CHECKSUM\_FAILURE

ER\_BINLOG\_ROW\_RBR\_TO\_SBR

ER\_BINLOG\_ROW\_WRONG\_TABLE\_DEF

ER\_CANT\_ACTIVATE\_LOG

ER\_CANT\_CHANGE\_GTID\_NEXT\_IN\_TRANSACTION

ER\_CANT\_CREATE\_FEDERATED\_TABLE

ER\_CANT\_CREATE\_SROUTINE

ER\_CANT\_DELETE\_FILE

ER\_CANT\_GET\_WD

ER\_CANT\_SET\_GTID\_PURGED\_WHEN\_GTID\_MODE\_IS\_OFF

ER\_CANT\_SET\_WD

ER\_CANT\_WRITE\_LOCK\_LOG\_TABLE

ER\_CREATE\_DB\_WITH\_READ\_LOCK

ER\_CYCLIC\_REFERENCE

ER\_DB\_DROP\_DELETE

ER\_DELAYED\_NOT\_SUPPORTED

ER\_DIFF\_GROUPS\_PROC

ER\_DISK\_FULL

ER\_DROP\_DB\_WITH\_READ\_LOCK

ER\_DROP\_USER

ER\_DUMP\_NOT\_IMPLEMENTED

ER\_ERROR\_DURING\_CHECKPOINT

ER\_ERROR\_ON\_CLOSE

ER\_EVENTS\_DB\_ERROR

ER\_EVENT\_CANNOT\_DELETE

ER\_EVENT\_CANT\_ALTER

ER\_EVENT\_COMPILE\_ERROR

ER\_EVENT\_DATA\_TOO\_LONG

ER\_EVENT\_DROP\_FAILED

ER\_EVENT\_MODIFY\_QUEUE\_ERROR

ER\_EVENT\_NEITHER\_M\_EXPR\_NOR\_M\_AT

ER\_EVENT\_OPEN\_TABLE\_FAILED

ER\_EVENT\_STORE\_FAILED

ER\_EXEC\_STMT\_WITH\_OPEN\_CURSOR

ER\_FAILED\_ROUTINE\_BREAK\_BINLOG

ER\_FLUSH\_MASTER\_BINLOG\_CLOSED

ER\_FORM\_NOT\_FOUND

ER\_FOUND\_GTID\_EVENT\_WHEN\_GTID\_MODE\_IS\_OFF\_\_UNUSED

ER\_FRM\_UNKNOWN\_TYPE

ER\_GOT\_SIGNAL

ER\_GRANT\_PLUGIN\_USER\_EXISTS

ER\_GTID\_MODE\_REQUIRES\_BINLOG

ER\_GTID\_NEXT\_IS\_NOT\_IN\_GTID\_NEXT\_LIST

ER\_HASHCHK

ER\_INDEX\_REBUILD

ER\_INNODB\_NO\_FT\_USES\_PARSER

ER\_LIST\_OF\_FIELDS\_ONLY\_IN\_HASH\_ERROR

ER\_LOAD\_DATA\_INVALID\_COLUMN\_UNUSED

ER\_LOGGING\_PROHIBIT\_CHANGING\_OF

ER\_MALFORMED\_DEFINER

ER\_MASTER\_KEY\_ROTATION\_ERROR\_BY\_SE

ER\_NDB\_CANT\_SWITCH\_BINLOG\_FORMAT

ER\_NEVER\_USED

ER\_NISAMCHK

ER\_NO\_CONST\_EXPR\_IN\_RANGE\_OR\_LIST\_ERROR

ER\_NO\_FILE\_MAPPING

ER\_NO\_GROUP\_FOR\_PROC

ER\_NO\_RAID\_COMPILED

ER\_NO\_SUCH\_KEY\_VALUE

ER\_NO\_SUCH\_PARTITION\_\_UNUSED

ER\_OBSOLETE\_CANNOT\_LOAD\_FROM\_TABLE

ER\_OBSOLETE\_COL\_COUNT\_DOESNT\_MATCH\_CORRUPTED

ER\_ORDER\_WITH\_PROC

ER\_PARTITION\_SUBPARTITION\_ERROR

ER\_PARTITION\_SUBPART\_MIX\_ERROR

ER\_PART\_STATE\_ERROR

ER\_PASSWD\_LENGTH

ER\_QUERY\_ON\_MASTER

ER\_RBR\_NOT\_AVAILABLE

ER\_SKIPPING\_LOGGED\_TRANSACTION

ER\_SLAVE\_CHANNEL\_DELETE

ER\_SLAVE\_MULTIPLE\_CHANNELS\_HOST\_PORT

ER\_SLAVE\_MUST\_STOP

ER\_SLAVE\_WAS\_NOT\_RUNNING

ER\_SLAVE\_WAS\_RUNNING

ER\_SP\_GOTO\_IN\_HNDLR

ER\_SP\_PROC\_TABLE\_CORRUPT

ER\_SQL\_MODE\_NO\_EFFECT

ER\_SR\_INVALID\_CREATION\_CTX

ER\_TABLE\_NEEDS\_UPG\_PART

ER\_TOO\_MUCH\_AUTO\_TIMESTAMP\_COLS

ER\_UNEXPECTED\_EOF

ER\_UNION\_TABLES\_IN\_DIFFERENT\_DIR

ER\_UNSUPPORTED\_BY\_REPLICATION\_THREAD

ER\_UNUSED1

ER\_UNUSED2

ER\_UNUSED3

ER\_UNUSED4

ER\_UNUSED5

ER\_UNUSED6

ER\_VIEW\_SELECT\_DERIVED\_UNUSED

ER\_WRONG\_MAGIC

ER\_WSAS\_FAILED

* 删除了被废弃的 INFORMATION\_SCHEMA INNODB\_LOCKS 和 INNODB\_LOCK\_WAITS 表。使用Performance Schema data\_locks和data\_lock\_waits表代替。

注意

在MySQL 5.7中，INNODB\_LOCKS表中的LOCK\_TABLE列以及sys schema innodb\_lock\_waits和x$innodb\_lock\_waits视图中的locked\_table列包含组合模式/表名值。在MySQL 8.0中，data\_locks表和sys schema视图包含单独的模式名和表名列。参见第28.4.3.9节，"innodb\_lock\_waits和x$innodb\_lock\_waits视图"。

* InnoDB不再支持压缩的临时表。当启用innodb\_strict\_mode时（默认），如果指定了ROW\_FORMAT=COMPRESSED或KEY\_BLOCK\_SIZE，CREATE TEMPORARY TABLE会返回一个错误。如果innodb\_strict\_mode被禁用，会发出警告，并且使用非压缩行格式创建临时表。
* 在MySQL数据目录之外创建表空间数据文件时，InnoDB不再创建.isl文件（InnoDB符号链接文件）。innodb\_directories选项现在支持定位在数据目录之外创建的表空间文件。

随着这一变化，不再支持在服务器离线时通过手动修改.isl文件移动远程表空间。移动远程表空间文件现在被innodb\_directories选项所支持。参见第15.6.3.6节，"在服务器离线时移动表空间文件"。

* 删除了下列 InnoDB 文件格式变量。
  + innodb\_file\_format
  + innodb\_file\_format\_check
  + innodb\_file\_format\_max
  + innodb\_large\_prefix

文件格式变量对于创建与MySQL 5.1中InnoDB早期版本兼容的表是必要的。现在，MySQL 5.1已经达到其产品生命周期的终点，这些选项不再需要了。

FILE\_FORMAT列已从INNODB\_TABLES和INNODB\_TABLESPACES信息模式表中删除。

* 移除了innodb\_support\_xa系统变量，该变量能够支持XA事务中的两阶段提交。InnoDB对XA事务中的两阶段提交的支持总是被启用。
* 删除了对 DTrace 的支持。
* 删除了JSON\_APPEND()函数。使用JSON\_ARRAY\_APPEND()代替。
* 在MySQL 8.0.13中删除了对在共享InnoDB表空间中放置表分区的支持。共享表空间包括InnoDB系统表空间和一般表空间。关于识别共享表空间中的分区并将其移至按表文件的表空间的信息，请参见第2.11.5节 "为升级准备你的安装"。
* 在MySQL 8.0.13中，对在SET以外的语句中设置用户变量的支持已被废弃。这一功能将在MySQL 9.0中删除。
* --ndb perror选项已被删除。请使用ndb\_perror工具来代替。
* 移除innodb\_undo\_logs变量。innodb\_rollback\_segments变量执行相同的功能，应该使用它来代替。
* Innodb\_available\_undo\_logs状态变量被移除。每个表空间的可用回滚段的数量可以使用SHOW VARIABLES LIKE 'innodb\_rollback\_segments'来检索。
* 从MySQL 8.0.14开始，以前废弃的innodb\_undo\_tablespaces变量不再可配置。更多信息，见第15.6.3.4节 "undo tablespaces"。
* 支持ALTER TABLE ... UPGRADE PARTITIONING语句的支持已被删除。
* 从MySQL 8.0.16开始，对internal\_tmp\_disk\_storage\_engine系统变量的支持已被删除；磁盘上的内部临时表现在总是使用InnoDB存储引擎。更多信息请参见磁盘上内部临时表的存储引擎。
* DISABLE\_SHARED CMake 选项未被使用，现已被删除

## **1.4 MySQL 8.0中增加、废弃或删除的服务器和状态变量及选项**

本节列出了服务器变量、状态变量以及在MySQL 8.0中首次添加、被废弃或被删除的选项。

**MySQL 8.0中引入的选项和变量**

下列系统变量、状态变量和服务器选项已在MySQL 8.0中添加。

* Acl\_cache\_items\_count。缓存的特权对象的数量。在MySQL 8.0.0中添加。
* Audit\_log\_current\_size。审计日志文件的当前大小。在MySQL 8.0.11中添加。
* Audit\_log\_event\_max\_drop\_size。最大丢弃的审计事件的大小。在MySQL 8.0.11中添加。
* Audit\_log\_events。处理的审计事件的数量。在MySQL 8.0.11中添加。
* Audit\_log\_events\_filtered。被过滤的审计事件的数量。在MySQL 8.0.11中添加。
* Audit\_log\_events\_lost。被丢弃的审计事件的数量。在MySQL 8.0.11中添加。
* Audit\_log\_events\_written。写入的审计事件的数量。在MySQL 8.0.11中添加。
* Audit\_log\_total\_size。写入的审计事件的综合大小。在MySQL 8.0.11中添加。
* Audit\_log\_write\_waits。写入延迟的审计事件的数量。在MySQL 8.0.11中添加。
* Authentication\_ldap\_sasl\_supported\_methods。为SASL LDAP认证支持的认证方法。在MySQL 8.0.21中添加。
* Caching\_sha2\_password\_rsa\_public\_key：缓存\_sha2\_password认证插件RSA公钥值。在MySQL 8.0.4中添加。
* Com\_alter\_resource\_group。ALTER RESOURCE GROUP语句的计数。在MySQL 8.0.3中添加。
* [**Com\_alter\_user\_default\_role**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\server-administration.html#statvar_Com_xxx):：ALTER USER ..: ALTER USER ... DEFAULT ROLE语句的计数。在MySQL 8.0.0中添加
* Com\_clone。CLONE语句的数量。在MySQL 8.0.2中添加
* Com\_create\_resource\_group: CREATE RESOURCE GROUP语句的计数。在MySQL 8.0.3中添加。
* om\_create\_role: CREATE ROLE语句的计数。在MySQL 8.0.0中添加。
* Com\_drop\_resource\_group: DROP RESOURCE GROUP语句的计数。在MySQL 8.0.3中添加。
* Com\_drop\_role: DROP ROLE语句的计数。在MySQL 8.0.0中添加。
* Com\_grant\_roles: GRANT ROLE语句的计数。在MySQL 8.0.0中添加。
* Com\_install\_component: INSTALL COMPONENT语句的计数。在MySQL 8.0.0中添加。
* Com\_replica\_start。START REPLICA和START SLAVE语句的计数。在MySQL 8.0.22中添加。
* Com\_replica\_stop。STOP REPLICA和STOP SLAVE语句的数量。在MySQL 8.0.22中添加。
* Com\_restart。RESTART语句的计数。在MySQL 8.0.4中添加。
* Com\_revoke\_roles。REVOKE ROLES语句的计数。在MySQL 8.0.0中添加。
* Com\_set\_resource\_group: SET RESOURCE GROUP语句的数量。在MySQL 8.0.3中添加。
* Com\_set\_role: SET ROLE语句的计数。在MySQL 8.0.0中添加。
* [**Com\_show\_replica\_status**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\server-administration.html#statvar_Com_xxx): 。SHOW REPLICA STATUS和SHOW SLAVE STATUS语句的计数。在MySQL 8.0.22中添加。
* Com\_show\_replicas。SHOW REPLICAS和SHOW SLAVE HOSTS语句的计数。在MySQL 8.0.22中添加。
* Com\_uninstall\_component。UINSTALL COMPONENT语句的计数。在MySQL 8.0.0中添加。
* Compression\_algorithm。当前连接的压缩算法。在MySQL 8.0.18中添加。
* Compression\_level（压缩级别）。当前连接的压缩级别。在MySQL 8.0.18中添加。
* Connection\_control\_delay\_generated。服务器延迟了多少次连接请求。在MySQL 8.0.1中添加。
* Current\_tls\_ca。ssl\_ca系统变量的当前值。在MySQL 8.0.16中添加。
* Current\_tls\_capath。ssl\_capath系统变量的当前值。在MySQL 8.0.16中添加。
* [**Current\_tls\_cert**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\server-administration.html#statvar_Current_tls_cert):。ssl\_cert系统变量的当前值。在MySQL 8.0.16中添加。
* [**Current\_tls\_cipher**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\server-administration.html#statvar_Current_tls_cipher):。ssl\_cipher系统变量的当前值。在MySQL 8.0.16中添加。
* Current\_tls\_ciphersuites。tsl\_ciphersuites系统变量的当前值。在MySQL 8.0.16中添加。
* Current\_tls\_crl。ssl\_crl系统变量的当前值。在MySQL 8.0.16中添加。
* Current\_tls\_crlpath。ssl\_crlpath系统变量的当前值。在MySQL 8.0.16中添加。
* Current\_tls\_key。ssl\_key系统变量的当前值。在MySQL 8.0.16中添加。
* [**Current\_tls\_version**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\server-administration.html#statvar_Current_tls_version):。tls\_version系统变量的当前值。在MySQL 8.0.16中添加。
* Error\_log\_buffered\_bytes。在error\_log表中使用的字节数。在MySQL 8.0.22中添加。
* Error\_log\_buffered\_events。在error\_log表中的事件数。在MySQL 8.0.22中添加。
* Error\_log\_expired\_events。从error\_log表中丢弃的事件的数量。在MySQL 8.0.22中添加。
* Error\_log\_latest\_write。最后一次写到error\_log表的时间。在MySQL 8.0.22中添加。
* Firewall\_access\_denied。被MySQL企业防火墙拒绝的语句数量。在MySQL 8.0.11中添加。
* Firewall\_access\_granted。被MySQL企业防火墙接受的语句数量。在MySQL 8.0.11中添加。
* Firewall\_cached\_entries。由MySQL企业防火墙记录的语句数量。在MySQL 8.0.11中添加。
* Innodb\_redo\_log\_enabled。InnoDBredo log状态。在MySQL 8.0.21中添加。
* Innodb\_system\_rows\_deleted。从系统模式表中删除的行数。在MySQL 8.0.19中添加。
* Innodb\_system\_rows\_inserted。插入到系统模式表中的行数。在MySQL 8.0.19中添加。
* Innodb\_system\_rows\_read。从系统模式表中读取的行数。在MySQL 8.0.19中添加。Innodb\_undo\_tablespaces\_active。活动的undo tablespaces的数量。在MySQL 8.0.14中添加。
* Innodb\_undo\_tablespaces\_explicit。用户创建的undo tablespaces的数量。在MySQL 8.0.14中添加。
* Innodb\_undo\_tablespaces\_implicit。由InnoDB创建的undo tablespaces的数量。在MySQL 8.0.14中添加
* Innodb\_undo\_tablespaces\_total：撤消的表空间总数。在MySQL 8.0.14中添加。
* Mysqlx\_bytes\_received\_compressed\_payload。收到的作为压缩消息有效载荷的字节数，在解压前测量。在MySQL 8.0.19中添加。
* Mysqlx\_bytes\_received\_uncompressed\_frame: 作为压缩消息有效载荷收到的字节数，在解压后测量。在MySQL 8.0.19中添加。
* Mysqlx\_bytes\_sent\_compressed\_payload。作为压缩的消息有效载荷发送的字节数，在压缩后测量。在MySQL 8.0.19中添加。
* Mysqlx\_bytes\_sent\_uncompressed\_frame: 作为压缩消息有效载荷发送的字节数，在压缩前测量。在MySQL 8.0.19中添加
* Mysqlx\_compression\_algorithm。此会话的X协议连接使用的压缩算法。在MySQL 8.0.20中添加。
* Mysqlx\_compression\_level: 此会话的X协议连接使用的压缩级别。在MySQL 8.0.20中添加。
* Secondary\_engine\_execution\_count。卸载到二级引擎的查询数量。在MySQL 8.0.13中添加。
* activate\_all\_roles\_on\_login: 是否在连接时激活所有用户角色。在MySQL 8.0.2中添加
* admin-ssl：启用连接加密。在MySQL 8.0.21中添加。
* admin\_address。在管理界面上连接时要绑定的IP地址。在MySQL 8.0.14中添加。
* admin\_port。用于管理界面连接的TCP/IP号码。在MySQL 8.0.14中添加。
* admin\_ssl\_ca。包含受信任的SSL证书颁发机构列表的文件。在MySQL 8.0.21中添加
* admin\_ssl\_capath。包含可信的SSL证书颁发机构证书文件的目录。在MySQL 8.0.21中添加。
* admin\_ssl\_cert。包含X.509证书的文件。在MySQL 8.0.21中添加
* admin\_ssl\_cipher。允许用于连接加密的密码。在MySQL 8.0.21中添加。
* admin\_ssl\_crl。包含证书撤销列表的文件。在MySQL 8.0.21中添加。
* admin\_ssl\_crlpath。包含证书撤销列表文件的目录。在MySQL 8.0.21中添加。
* admin\_ssl\_key。包含X.509密钥的文件。在MySQL 8.0.21中添加
* admin\_tls\_ciphersuites。允许的用于加密连接的TLSv1.3密码套件。在MySQL 8.0.21中添加。
* admin\_tls\_version。允许用于加密连接的TLS协议。在MySQL 8.0.21中添加。
* audit-log。是否激活审计日志插件。在MySQL 8.0.11中添加。
* audit\_log\_buffer\_size。审计日志缓冲区的大小。在MySQL 8.0.11中添加。
* audit\_log\_compression。审计日志文件的压缩方法。在MySQL 8.0.11中添加。
* audit\_log\_connection\_policy。连接相关事件的审计日志策略。在MySQL 8.0.11中添加。
* audit\_log\_current\_session。是否审计当前会话。在MySQL 8.0.11中添加。
* audit\_log\_encryption。审计日志文件的加密方法。在MySQL 8.0.11中添加。
* audit\_log\_exclude\_accounts。不予审计的账户。在MySQL 8.0.11中添加。
* audit\_log\_file。审计日志文件的名称。在MySQL 8.0.11中添加。
* audit\_log\_filter\_id。当前审计日志过滤器的ID。在MySQL 8.0.11中添加。
* audit\_log\_flush。关闭并重新打开审计日志文件。在MySQL 8.0.11中添加。
* audit\_log\_format。审计日志文件格式。在MySQL 8.0.11中添加。
* audit\_log\_include\_accounts。要审计的账户。在MySQL 8.0.11中添加
* audit\_log\_max\_size。对JSON审计日志文件的组合大小的限制。在MySQL 8.0.25中添加。
* audit\_log\_password\_history\_keep\_days。保留存档的审计日志加密密码的天数。在MySQL 8.0.17中添加。
* audit\_log\_policy。审计日志策略。在MySQL 8.0.11中添加。
* audit\_log\_prune\_seconds。审计日志文件被修剪的秒数。在MySQL 8.0.24中添加。
* audit\_log\_read\_buffer\_size。审计日志文件的读取缓冲区大小。在MySQL 8.0.11中添加。
* audit\_log\_rotate\_on\_size。关闭并重新打开这个大小的审计日志文件。在MySQL 8.0.11中添加。
* audit\_log\_statement\_policy。语句相关事件的审计日志策略。在MySQL 8.0.11中添加。
* audit\_log\_strategy。审计日志策略。在MySQL 8.0.11中添加。
* authentication\_ldap\_sasl\_auth\_method\_name：认证方法名称。在MySQL 8.0.11中添加。
* authentication\_ldap\_sasl\_bind\_base\_dn。LDAP服务器的基本区分名称。在MySQL 8.0.11中添加。
* authentication\_ldap\_sasl\_bind\_root\_dn。LDAP服务器根部的识别名称。在MySQL 8.0.11中添加。
* authentication\_ldap\_sasl\_bind\_root\_pwd。LDAP服务器根绑定密码。在MySQL 8.0.11中添加。
* authentication\_ldap\_sasl\_ca\_path。LDAP服务器证书授权文件名。在MySQL 8.0.11中添加。
* authentication\_ldap\_sasl\_group\_search\_attr。LDAP服务器组搜索属性。在MySQL 8.0.11中添加。
* authentication\_ldap\_sasl\_group\_search\_filter。LDAP自定义组搜索过滤器。在MySQL 8.0.11中添加。
* authentication\_ldap\_sasl\_init\_pool\_size。LDAP服务器初始连接池大小。在MySQL 8.0.11中添加。
* authentication\_ldap\_sasl\_log\_status。LDAP服务器的日志级别。在MySQL 8.0.11中添加。
* authentication\_ldap\_sasl\_max\_pool\_size。LDAP服务器的最大连接池大小。在MySQL 8.0.11中添加。
* authentication\_ldap\_sasl\_referral。是否启用LDAP搜索转介。在MySQL 8.0.20中添加。
* authentication\_ldap\_sasl\_server\_host：LDAP服务器主机名或IP地址。在MySQL 8.0.11中添加。
* authentication\_ldap\_sasl\_server\_port。LDAP服务器的端口号。在MySQL 8.0.11中添加。
* authentication\_ldap\_sasl\_tls。是否使用与LDAP服务器的加密连接。在MySQL 8.0.11中添加。
* authentication\_ldap\_sasl\_user\_search\_attr。LDAP服务器的用户搜索属性。在MySQL 8.0.11中添加。
* authentication\_ldap\_simple\_auth\_method\_name：认证方法名称。在MySQL 8.0.11中添加。
* authentication\_ldap\_simple\_bind\_base\_dn。LDAP服务器的基本区分名称。在MySQL 8.0.11中添加。
* authentication\_ldap\_simple\_bind\_root\_dn。LDAP服务器根部的识别名称。在MySQL 8.0.11中添加。
* authentication\_ldap\_simple\_bind\_root\_pwd。LDAP服务器根绑定密码。在MySQL 8.0.11中添加。
* authentication\_ldap\_simple\_ca\_path。LDAP服务器证书授权文件名。在MySQL 8.0.11中添加。
* authentication\_ldap\_simple\_group\_search\_attr。LDAP服务器组搜索属性。在MySQL 8.0.11中添加。
* authentication\_ldap\_simple\_group\_search\_filter。LDAP自定义组搜索过滤器。在MySQL 8.0.11中添加。
* authentication\_ldap\_simple\_init\_pool\_size。LDAP服务器初始连接池大小。在MySQL 8.0.11中添加。
* authentication\_ldap\_simple\_log\_status。LDAP服务器的日志级别。在MySQL 8.0.11中添加。
* authentication\_ldap\_simple\_max\_pool\_size。LDAP服务器的最大连接池大小。在MySQL 8.0.11中添加。
* authentication\_ldap\_simple\_referral。是否启用LDAP搜索转介。在MySQL 8.0.20中添加。
* authentication\_ldap\_simple\_server\_host：LDAP服务器主机名或IP地址。在MySQL 8.0.11中添加。
* authentication\_ldap\_simple\_server\_port。LDAP服务器的端口号。在MySQL 8.0.11中添加。
* authentication\_ldap\_simple\_tls。是否使用与LDAP服务器的加密连接。在MySQL 8.0.11中添加。
* authentication\_ldap\_simple\_user\_search\_attr。LDAP服务器的用户搜索属性。在MySQL 8.0.11中添加。
* authentication\_windows\_log\_level。Windows认证插件的日志级别。在MySQL 8.0.11中添加。
* authentication\_windows\_use\_principal\_name：是否使用Windows认证插件的主体名称。在MySQL 8.0.11中添加
* binlog\_encryption。启用对该服务器上的二进制日志文件和中继日志文件的加密。在MySQL 8.0.14中添加。
* binlog\_expire\_logs\_seconds。在此秒数后清除二进制日志。在MySQL 8.0.1中添加
* binlog\_rotate\_encryption\_master\_key\_at\_startup。在服务器启动时旋转二进制日志主密钥。在MySQL 8.0.14中添加。
* binlog\_row\_metadata。当使用基于行的日志时，是否将所有或仅最小的表相关元数据记录到二进制日志。在MySQL 8.0.1中添加。
* binlog\_row\_value\_options。启用基于行的复制的部分JSON更新的二进制日志。在MySQL 8.0.3中添加。
* binlog\_transaction\_compression。启用二进制日志文件中事务有效载荷的压缩。在MySQL 8.0.20中添加。
* binlog\_transaction\_compression\_level\_zstd：二进制日志文件中事务有效载荷的压缩级别。在MySQL 8.0.20中添加。
* binlog\_transaction\_dependency\_history\_size。为查找最后更新某行的事务而保留的行哈希值的数量。在MySQL 8.0.1中添加。
* binlog\_transaction\_dependency\_tracking。依赖性信息的来源（提交时间戳或事务写入集），据此评估哪些事务可以由副本的多线程应用者并行执行。在MySQL 8.0.1中添加。
* caching\_sha2\_password\_auto\_generate\_rsa\_keys。是否自动生成RSA密钥对文件。在MySQL 8.0.4中添加。
* caching\_sha2\_password\_digest\_rounds。caching\_sha2\_password认证插件的哈希轮数。在MySQL 8.0.24中添
* caching\_sha2\_password\_private\_key\_path。SHA2认证插件的私钥路径名称。在MySQL 8.0.3中添加。
* caching\_sha2\_password\_public\_key\_path。SHA2认证插件的公钥路径名称。在MySQL 8.0.3中添加。
* clone\_autotune\_concurrency。为远程克隆操作启用动态生成线程。在MySQL 8.0.17中添加
* clone\_buffer\_size。定义捐赠者MySQL服务器实例上的中间缓冲区的大小。在MySQL 8.0.17中添加。
* clone\_ddl\_timeout。克隆操作等待备份锁的秒数。在MySQL 8.0.17中添加。
* clone\_donor\_timeout\_after\_network\_failure。网络故障后重新启动克隆操作的允许时间。在MySQL 8.0.24中添加。
* clone\_enable\_compression。在克隆过程中启用网络层的数据压缩。在MySQL 8.0.17中添加。
* clone\_max\_concurrency。用于执行克隆操作的最大并发线程数。在MySQL 8.0.17中添加。
* clone\_max\_data\_bandwidth：用于远程克隆操作的最大数据传输速率，单位为MiB/s。在MySQL 8.0.17中添加。
* clone\_max\_network\_bandwidth：远程克隆操作的最大网络传输速率，单位为每秒MiB。在MySQL 8.0.17中添加。
* clone\_ssl\_ca。指定证书授权（CA）文件的路径。在MySQL 8.0.14中添加。
* clone\_ssl\_cert: 指定公钥证书文件的路径。在MySQL 8.0.14中添加。
* clone\_ssl\_key。指定私钥文件的路径。在MySQL 8.0.14中添加。
* clone\_valid\_donor\_list。定义远程克隆操作的捐赠者主机地址。在MySQL 8.0.17中添加。
* connection\_control\_failed\_connections\_threshold。延迟发生前连续失败的连接尝试。在MySQL 8.0.1中添加。
* connection\_control\_max\_connection\_delay。服务器响应失败连接尝试的最大延迟（毫秒）。在MySQL 8.0.1中添加。
* connection\_control\_min\_connection\_delay。服务器响应失败的连接尝试的最小延迟（毫秒）。在MySQL 8.0.1中添加。
* create\_admin\_listener\_thread。是否为管理界面的连接使用专用监听线程。在MySQL 8.0.14中添加。
* cte\_max\_recursion\_depth。公共表表达式的最大递归深度。在MySQL 8.0.3中添加。
* ddl-rewriter。是否激活ddl\_rewriter插件。在MySQL 8.0.16中添加。
* default\_collation\_for\_utf8mb4：utf8mb4字符集的默认排序。在MySQL 8.0.11中添加。
* default\_table\_encryption。默认的模式和表空间加密设置。在MySQL 8.0.16中添加。
* dragnet.Status。对dragnet.log\_error\_filter\_rules的最近一次分配的结果。在MySQL 8.0.12中添加。
* dragnet.log\_error\_filter\_rules。用于错误日志的过滤规则。在MySQL 8.0.4中添加。
* early-plugin-load。指定在加载强制性内置插件和存储引擎初始化前加载的插件。在MySQL 8.0.0中添加。
* generated\_random\_password\_length。生成密码的最大长度。在MySQL 8.0.18中添加。
* group\_replication\_advertise\_recovery\_endpoints。为分布式恢复提供的连接。在MySQL 8.0.21中添加。
* group\_replication\_autorejoin\_tries。成员为自动重新加入组而进行的尝试次数。在MySQL 8.0.16中添加。
* group\_replication\_clone\_threshold。捐赠者和接受者之间的事务数差距，超过这个差距就会使用远程克隆操作进行状态转移。在MySQL 8.0.17中添加。
* group\_replication\_communication\_debug\_options。组复制组件的调试信息的级别。在MySQL 8.0.3中添加。
* group\_replication\_communication\_max\_message\_size。组复制通信的最大消息大小，较大的消息会被分割。在MySQL 8.0.16中添加。
* group\_replication\_consistency。组提供的事务一致性保证的类型。在MySQL 8.0.14中添加。
* group\_replication\_exit\_state\_action。实例非自愿离开组时的行为方式。在MySQL 8.0.12中添加。
* group\_replication\_flow\_control\_hold\_percent。保持未使用的组配额的百分比。在MySQL 8.0.2中添加。
* group\_replication\_flow\_control\_max\_commit\_quota。组的最大流量控制配额。在MySQL 8.0.2中添加。
* group\_replication\_flow\_control\_member\_quota\_percent。在计算配额时，成员应假定自己可用的配额的百分比。在MySQL 8.0.2中添加。
* group\_replication\_flow\_control\_min\_quota。可分配给每个成员的最低流量控制配额。在MySQL 8.0.2中添加。
* group\_replication\_flow\_control\_min\_recovery\_quota。可分配给每个成员的最低配额，因为另一个组成员正在恢复。在MySQL 8.0.2中添加。
* group\_replication\_flow\_control\_period。定义流量控制迭代之间要等待多少秒。在MySQL 8.0.2中添加。
* group\_replication\_flow\_control\_release\_percent。当流量控制不再需要节制写作者成员时，应如何释放组配额。在MySQL 8.0.2中添加。
* group\_replication\_ip\_allowlist。允许连接到组的主机列表（MySQL 8.0.22及以后版本）。在MySQL 8.0.22中添加。
* group\_replication\_member\_expel\_timeout。怀疑群组成员失败和将其从群组驱逐之间的时间，导致群组成员重新配置。在MySQL 8.0.13中添加。
* group\_replication\_member\_weight。这个成员被选为主要成员的几率。在MySQL 8.0.2中添加。
* group\_replication\_message\_cache\_size。组通信引擎消息缓存的最大内存（XCom）。在MySQL 8.0.16中添加。
* group\_replication\_recovery\_compression\_algorithms。允许的出站恢复连接的压缩算法。在MySQL 8.0.18中添加。
* group\_replication\_recovery\_get\_public\_key。是否接受关于从捐赠者获取公钥的首选项。在MySQL 8.0.4中添加。
* group\_replication\_recovery\_public\_key\_path。用于接受公钥信息。在MySQL 8.0.4中添加。
* group\_replication\_recovery\_tls\_ciphersuites。当TLSv1.3被用于与该实例作为客户端（加入成员）的连接加密时，允许的密码套件。在MySQL 8.0.19中添加。
* group\_replication\_recovery\_tls\_version。允许的TLS协议用于作为客户端（加入成员）的连接加密。在MySQL 8.0.19中添加。
* group\_replication\_recovery\_zstd\_compression\_level。使用zstd压缩的恢复连接的压缩级别。在MySQL 8.0.18中添加。
* group\_replication\_tls\_source。组复制的TLS材料的来源。在MySQL 8.0.21中添加。
* group\_replication\_unreachable\_majority\_timeout。导致少数人离开组的网络分区要等待多长时间。在MySQL 8.0.2中添加。
* histogram\_generation\_max\_mem\_size。用于创建直方图统计的最大内存。在MySQL 8.0.2中添加。
* immediate\_server\_version。作为即时复制源的服务器的MySQL服务器版本号。在MySQL 8.0.14中添加。
* information\_schema\_stats\_expiry。缓存表统计的过期设置。在MySQL 8.0.3中添加。
* innodb\_buffer\_pool\_debug。当缓冲池的大小小于1GB时，允许多个缓冲池实例。在MySQL 8.0.0中添加。
* innodb\_buffer\_pool\_in\_core\_file。控制将缓冲池页面写入核心文件。在MySQL 8.0.14中添加。
* innodb\_checkpoint\_disabled。禁用检查点，以便故意的服务器退出总是启动恢复。在MySQL 8.0.2中添加。
* innodb\_ddl\_log\_crash\_reset\_debug。调试选项，用于重置DDL日志崩溃注入计数器。在MySQL 8.0.3中添加。
* innodb\_deadlock\_detect。启用或禁用死锁检测。在MySQL 8.0.0中添加。
* innodb\_dedicated\_server。启用缓冲池大小、日志文件大小和冲刷方法的自动配置。在MySQL 8.0.3中添加。
* innodb\_directories。定义了在启动时扫描表空间数据文件的目录。在MySQL 8.0.4中添加。
* innodb\_doublewrite\_batch\_size。每批写入的双写页的数量。在MySQL 8.0.20中添加。
* innodb\_doublewrite\_dir。双重写缓冲区文件目录。在MySQL 8.0.20中添加。
* innodb\_doublewrite\_files。双重写文件的数量。在MySQL 8.0.20中添加。
* innodb\_doublewrite\_pages。每个线程的双写页数。在MySQL 8.0.20中添加。
* innodb\_extend\_and\_initialize。控制在Linux上如何分配新的表空间页。在MySQL 8.0.22中添加。
* innodb\_fsync\_threshold。控制InnoDB在创建新文件时调用fsync的频率。在MySQL 8.0.13中添加。
* innodb\_idle\_flush\_pct。限制InnoDB空闲时的I/0操作。在MySQL 8.0.18中添加。
* innodb\_log\_checkpoint\_fuzzy\_now。调试选项，强制InnoDB写入模糊检查点。在MySQL 8.0.13中添加。
* innodb\_log\_spin\_cpu\_abs\_lwm。CPU使用量的最小值，低于此值时，用户线程在等待刷新的重做时不再旋转。在MySQL 8.0.11中添加。
* innodb\_log\_spin\_cpu\_pct\_hwm。用户线程在等待刷新的重做时不再旋转的最大CPU使用率。在MySQL 8.0.11中添加。
* innodb\_log\_wait\_for\_flush\_spin\_hwm。最大的平均日志刷新时间，超过该时间用户线程在等待刷新的重做时不再旋转。在MySQL 8.0.11中添加。
* innodb\_log\_writer\_threads。启用专用的日志写入器线程来写入和冲刷redo log。在MySQL 8.0.22中添加。
* innodb\_parallel\_read\_threads。用于并行索引读取的线程数量。在MySQL 8.0.14中添加。
* innodb\_print\_ddl\_logs。是否将DDL日志打印到错误日志。在MySQL 8.0.3中添加。
* innodb\_redo\_log\_archive\_dirs。标记的redo log存档目录。在MySQL 8.0.17中添加。
* innodb\_redo\_log\_encrypt。控制对加密表空间的redo log数据进行加密。在MySQL 8.0.1中添加。
* innodb\_scan\_directories。定义在InnoDB恢复期间扫描表空间文件的目录。在MySQL 8.0.2中添加。
* innodb\_segment\_reserve\_factor。保留为空页的表空间文件段页的百分比。在MySQL 8.0.25中添加。
* innodb\_spin\_wait\_pause\_multiplier。用于确定自旋等待循环中PAUSE指令数量的乘数值。在MySQL 8.0.16中添加。
* innodb\_stats\_include\_delete\_marked。在计算持久性InnoDB统计数据时，包括有删除标记的记录。在MySQL 8.0.1中添加。
* innodb\_temp\_tablespaces\_dir。会话临时表空间路径。在MySQL 8.0.13中添加。
* innodb\_tmpdir。在线ALTER TABLE操作期间创建的临时表文件的目录位置。在MySQL 8.0.0中添加。
* innodb\_undo\_log\_encrypt。控制对加密表空间的Undo logs数据的加密。在MySQL 8.0.1中添加。
* innodb\_use\_fdatasync。当向操作系统冲刷数据时，InnoDB是否使用fdatasync()而不是fsync()。在MySQL 8.0.25中添加。
* innodb\_validate\_tablespace\_paths。在启动时启用表空间路径验证。在MySQL 8.0.21中添加。
* internal\_tmp\_mem\_storage\_engine：用于内部内存临时表的存储引擎。在MySQL 8.0.2中添加。
* keyring-migration-destination: 钥匙迁移目的地钥匙圈插件。在MySQL 8.0.4中添加。
* keyring-migration-host：连接到运行中的服务器进行密钥迁移的主机名。在MySQL 8.0.4中添加。
* keyring-migration-password：连接到运行中的服务器进行密钥迁移的密码。在MySQL 8.0.4中添加。
* keyring-migration-port。TCP/IP端口号，用于连接到运行中的服务器进行密钥迁移。在MySQL 8.0.4中添加。
* keyring-migration-socket。Unix套接字文件或Windows命名的管道，用于连接到运行中的服务器进行密钥迁移。在MySQL 8.0.4中添加。
* keyring-migration-source。钥匙迁移源钥匙圈插件。在MySQL 8.0.4中添加。
* keyring-migration-to-component。钥匙圈迁移是指从插件到组件。在MySQL 8.0.24中添加。
* keyring-migration-user: 用于连接到运行中的服务器进行密钥迁移的用户名。在MySQL 8.0.4中添加。
* keyring\_aws\_cmk\_id。AWS钥匙圈插件客户主钥匙的ID值。在MySQL 8.0.11中添加。
* keyring\_aws\_conf\_file。AWS钥匙圈插件的配置文件位置。在MySQL 8.0.11中添加。
* keyring\_aws\_data\_file。AWS钥匙圈插件的存储文件位置。在MySQL 8.0.11中添加。
* keyring\_aws\_region。AWS钥匙圈插件区域。在MySQL 8.0.11中添加。
* keyring\_encrypted\_file\_data：keyring\_encrypted\_file插件数据文件。在MySQL 8.0.11中添加。
* keyring\_encrypted\_file\_password：keyring\_encrypted\_file插件的密码。在MySQL 8.0.11中添加。
* keyring\_hashicorp\_auth\_path。HashiCorp Vault AppRole认证路径。在MySQL 8.0.18中添加。
* keyring\_hashicorp\_ca\_path。keyring\_hashicorp CA文件的路径。在MySQL 8.0.18中添加。
* keyring\_hashicorp\_caching：是否启用keyring\_hashicorp缓存。在MySQL 8.0.18中添加。
* keyring\_hashicorp\_commit\_auth\_path：使用中的keyring\_hashicorp\_auth\_path值。在MySQL 8.0.18中添加。
* keyring\_hashicorp\_commit\_ca\_path：使用中的keyring\_hashicorp\_ca\_path值。在MySQL 8.0.18中添加。
* keyring\_hashicorp\_commit\_caching：使用中的keyring\_hashicorp\_caching值。在MySQL 8.0.18中添加。
* keyring\_hashicorp\_commit\_role\_id：使用中的keyring\_hashicorp\_role\_id值。在MySQL 8.0.18中添加。
* keyring\_hashicorp\_commit\_server\_url：使用中的keyring\_hashicorp\_server\_url值。在MySQL 8.0.18中添加。
* keyring\_hashicorp\_commit\_store\_path：使用中的keyring\_hashicorp\_store\_path值。在MySQL 8.0.18中添加。
* keyring\_hashicorp\_role\_id。HashiCorp Vault AppRole认证角色ID。在MySQL 8.0.18中添加。
* keyring\_hashicorp\_secret\_id。HashiCorp Vault AppRole认证秘密ID。在MySQL 8.0.18中添加。
* keyring\_hashicorp\_server\_url。HashiCorp Vault服务器的URL。在MySQL 8.0.18中添加。
* keyring\_hashicorp\_store\_path。HashiCorp Vault的存储路径。在MySQL 8.0.18中添加。
* keyring\_oci\_ca\_certificate。用于同行认证的CA证书文件。在MySQL 8.0.22中添加。
* keyring\_oci\_compartment。OCI隔间OCID。在MySQL 8.0.22中添加。
* keyring\_oci\_encryption\_endpoint。OCI加密服务器端点。在MySQL 8.0.22中添加。
* keyring\_oci\_key\_file。OCI RSA私钥文件。在MySQL 8.0.22中添加。
* keyring\_oci\_key\_fingerprint。OCI RSA私钥文件的指纹。在MySQL 8.0.22中添加。
* keyring\_oci\_management\_endpoint。OCI管理服务器端点。在MySQL 8.0.22中添加。
* keyring\_oci\_master\_key。OCI主密钥OCID。在MySQL 8.0.22中添加。
* keyring\_oci\_secrets\_endpoint。OCI机密服务器的端点。在MySQL 8.0.22中添加。
* keyring\_oci\_tenancy。OCI租户OCID。在MySQL 8.0.22中添加。
* keyring\_oci\_user。OCI用户的OCID。在MySQL 8.0.22中添加。
* keyring\_oci\_vaults\_endpoint。OCI金库服务器端点。在MySQL 8.0.22中添加。
* keyring\_oci\_virtual\_vault。OCI保险库OCID。在MySQL 8.0.22中添加。
* keyring\_okv\_conf\_dir。Oracle Key Vault密钥库插件配置目录。在MySQL 8.0.11中添加。
* keyring\_operations。是否启用了钥匙圈操作。在MySQL 8.0.4中添加。
* lock\_order。是否在运行时启用LOCK\_ORDER工具。在MySQL 8.0.17中添加。
* lock\_order\_debug\_loop。当LOCK\_ORDER工具遇到标记为循环的依赖关系时，是否引起调试断言。在MySQL 8.0.17中添加。
* lock\_order\_debug\_missing\_arc: 当LOCK\_ORDER工具遇到未声明的依赖关系时，是否会引起调试断言。在MySQL 8.0.17中添加。
* lock\_order\_debug\_missing\_key。当LOCK\_ORDER工具遇到没有正确使用Performance Schema工具的对象时，是否会引起调试断言。在MySQL 8.0.17中添加。
* lock\_order\_debug\_missing\_unlock。当LOCK\_ORDER工具遇到仍然持有的锁被破坏时，是否会引起调试断言。在MySQL 8.0.17中添加。
* lock\_order\_dependencies。lock\_order\_dependencies.txt文件的路径。在MySQL 8.0.17中添加。
* lock\_order\_extra\_dependencies。第二个依赖文件的路径。在MySQL 8.0.17中添加。
* lock\_order\_output\_directory。LOCK\_ORDER工具写入日志的目录。在MySQL 8.0.17中添加。
* lock\_order\_print\_txt。是否执行锁序图分析和打印文本报告。在MySQL 8.0.17中添加。
* lock\_order\_trace\_loop。当LOCK\_ORDER工具遇到标记为循环的依赖关系时，是否打印日志文件跟踪。在MySQL 8.0.17中添加。
* lock\_order\_trace\_missing\_arc：当LOCK\_ORDER工具遇到未声明的依赖关系时，是否打印日志文件跟踪。在MySQL 8.0.17中添加。
* lock\_order\_trace\_missing\_key。当LOCK\_ORDER工具遇到没有正确使用Performance Schema工具的对象时，是否打印日志文件跟踪。在MySQL 8.0.17中添加。
* lock\_order\_trace\_missing\_unlock。当LOCK\_ORDER工具遇到仍然持有的锁被破坏时，是否打印日志文件跟踪。在MySQL 8.0.17中添加。
* log\_error\_filter\_rules。用于错误日志的过滤规则。在MySQL 8.0.2中添加。
* log\_error\_services。用于错误日志记录的组件。在MySQL 8.0.2中添加。
* log\_error\_suppression\_list。要抑制的警告/信息错误日志信息。在MySQL 8.0.13中添加。
* log\_slow\_extra。是否向慢速查询日志文件写入额外信息。在MySQL 8.0.14中添加。
* mandatory\_roles。自动授予所有用户的角色。在MySQL 8.0.2中添加。
* mysql\_firewall\_mode。MySQL企业防火墙是否运行。在MySQL 8.0.11中添加。
* mysql\_firewall\_trace。是否启用防火墙跟踪。在MySQL 8.0.11中添加。
* mysqlx：是否初始化了X插件。在MySQL 8.0.11中添加。
* mysqlx\_compression\_algorithms。允许用于X协议连接的压缩算法。在MySQL 8.0.19中添加。
* mysqlx\_deflate\_default\_compression\_level。X协议连接上Deflate算法的默认压缩级别。在MySQL 8.0.20中添加。
* mysqlx\_deflate\_max\_client\_compression\_level。X协议连接上的Deflate算法的最大允许压缩级别。在MySQL 8.0.20中添加。
* mysqlx\_interactive\_timeout。等待交互式客户端超时的秒数。在MySQL 8.0.4中添加。
* mysqlx\_lz4\_default\_compression\_level。X协议连接上LZ4算法的默认压缩级别。在MySQL 8.0.20中添加。
* mysqlx\_lz4\_max\_client\_compression\_level。X协议连接上LZ4算法的最大允许压缩级别。在MySQL 8.0.20中添加。
* mysqlx\_read\_timeout。等待阻塞读操作完成的秒数。在MySQL 8.0.4中添加。
* mysqlx\_wait\_timeout。等待连接活动的秒数。在MySQL 8.0.4中添加。
* mysqlx\_write\_timeout。等待阻塞的写操作完成的秒数。在MySQL 8.0.4中添加。
* mysqlx\_zstd\_default\_compression\_level。X协议连接上zstd算法的默认压缩级别。在MySQL 8.0.20中添加。
* mysqlx\_zstd\_max\_client\_compression\_level。X协议连接上zstd算法的最大允许压缩级别。在MySQL 8.0.20中添加。
* named\_pipe\_full\_access\_group。被授予完全访问命名管道的Windows组的名称。在MySQL 8.0.14中添加。
* no-dd-upgrade。防止在启动时自动升级数据字典表。在MySQL 8.0.4中添加。
* no-monitor。不分叉RESTART所需的监控进程。在MySQL 8.0.12中添加。
* original\_commit\_timestamp。在原始源上提交事务的时间。在MySQL 8.0.1中添加。
* original\_server\_version。事务最初提交的服务器的MySQL服务器版本号。在MySQL 8.0.14中添加。
* partial\_revokes。是否启用了部分撤销。在MySQL 8.0.16中添加。
* password\_history。密码重用前所需的密码修改次数。在MySQL 8.0.3中添加。
* password\_require\_current。更改密码是否需要当前密码验证。在MySQL 8.0.13中添加。
* password\_reuse\_interval。密码重用前需要经过的天数。在MySQL 8.0.3中添加。
* performance\_schema\_max\_digest\_sample\_age。查询重新采样的年龄，以秒为单位。在MySQL 8.0.3中添加。
* performance\_schema\_show\_processlist。选择SHOW PROCESSLIST实现。在MySQL 8.0.22中添加。
* persist\_only\_admin\_x509\_subject。SSL证书X.509主题，可使持久化限制的系统变量持久化。在MySQL 8.0.14中添加。
* persisted\_globals\_load。是否加载持久化的配置设置。在MySQL 8.0.0中添加。
* print\_identified\_with\_as\_hex。对于SHOW CREATE USER，打印包含不可打印字符的十六进制哈希值。在MySQL 8.0.17中添加。
* protocol\_compression\_algorithms。为传入的连接允许的压缩算法。在MySQL 8.0.18中添加。
* regexp\_stack\_limit。正则表达式匹配堆栈大小限制。在MySQL 8.0.4中添加。
* regexp\_time\_limit。正则表达式匹配超时。在MySQL 8.0.4中添加。
* replication\_optimize\_for\_static\_plugin\_config。用于半同步复制的共享锁。在MySQL 8.0.23中添加。
* replication\_sender\_observe\_commit\_only。用于半同步复制的有限回调。在MySQL 8.0.23中添加。
* require\_row\_format。供内部服务器使用。在MySQL 8.0.19中添加。
* resultet\_metadata。服务器是否返回结果集元数据。在MySQL 8.0.3中添加。
* rpl\_read\_size。设置从二进制日志文件和中继日志文件读取的最小数据量（字节）。在MySQL 8.0.11中添加。
* secondary\_engine\_cost\_threshold。优化器成本阈值，用于将查询卸载到辅助引擎。在MySQL 8.0.16中添加。
* select\_into\_buffer\_size。用于OUTFILE或DUMPFILE导出文件的缓冲区的大小；覆盖read\_buffer\_size。在MySQL 8.0.22中添加。
* select\_into\_disk\_sync。在为OUTFILE或DUMPFILE导出文件刷新缓冲区后与存储设备同步数据；OFF禁用同步，是默认值。在MySQL 8.0.22中添加。
* select\_into\_disk\_sync\_delay。当select\_into\_sync\_disk = ON时，设置每次同步OUTFILE或DUMPFILE导出文件缓冲区后的延迟（毫秒），否则没有影响。在MySQL 8.0.22中添加。
* show\_create\_table\_skip\_secondary\_engine。是否从SHOW CREATE TABLE输出中排除SECONDARY ENGINE子句。在MySQL 8.0.18中添加。
* show\_create\_table\_verbosity。是否在SHOW CREATE TABLE中显示ROW\_FORMAT，即使它有默认值。在MySQL 8.0.11中添加。
* sql\_require\_primary\_key。表是否必须有主键。在MySQL 8.0.13中添加。
* ssl\_fips\_mode。是否在服务器端启用FIPS模式。在MySQL 8.0.11中添加。
* syseventlog.设施。用于系统日志消息的设施。在MySQL 8.0.13中添加。
* syseventlog.include\_pid。是否在syslog消息中包括服务器PID。在MySQL 8.0.13中添加。
* syseventlog.tag。syseventlog.tag：syslog消息中服务器标识符的标签。在MySQL 8.0.13中添加。
* table\_encryption\_privilege\_check。启用TABLE\_ENCRYPTION\_ADMIN权限检查。在MySQL 8.0.16中添加。
* temptable\_max\_mmap。TempTable存储引擎可以从内存映射的临时文件分配的最大内存量。在MySQL 8.0.23中添加。
* temptable\_max\_ram。定义在数据存储到磁盘之前，TempTable存储引擎可以占用的最大内存量。在MySQL 8.0.2中添加。
* temptable\_use\_mmap: 定义当达到temptable\_max\_ram阈值时，TempTable存储引擎是否分配内存映射的文件。在MySQL 8.0.16中添加。
* thread\_pool\_algorithm。线程池算法。在MySQL 8.0.11中添加。
* thread\_pool\_high\_priority\_connection。当前会话是否是高优先级。在MySQL 8.0.11中添加。
* thread\_pool\_max\_active\_query\_threads。每个组允许的最大活动查询线程数。在MySQL 8.0.19中添加。
* thread\_pool\_max\_unused\_threads。允许的未使用线程的最大数量。在MySQL 8.0.11中添加。
* thread\_pool\_prio\_kickup\_timer：语句被移到高优先级执行之前的时间。在MySQL 8.0.11中添加。
* thread\_pool\_size。线程池中线程组的数量。在MySQL 8.0.11中添加。
* thread\_pool\_stall\_limit：语句被定义为停滞前的时间。在MySQL 8.0.11中添加。
* tls\_ciphersuites。允许的用于加密连接的TLSv1.3密码套件。在MySQL 8.0.16中添加。
* 升级。控制启动时的自动升级。在MySQL 8.0.16中添加。
* use\_secondary\_engine。是否使用辅助引擎执行查询。在MySQL 8.0.13中添加。
* validate-config。验证服务器配置。在MySQL 8.0.16中添加。
* validate\_password.check\_user\_name：是否根据用户名检查密码。在MySQL 8.0.4中添加。
* validate\_password.dictionary\_file：验证密码的字典文件。在MySQL 8.0.4中添加。
* validate\_password.dictionary\_file\_last\_parsed：字典文件最后被解析的时间。在MySQL 8.0.4中添加。
* validate\_password.dictionary\_file\_words\_count: 词典文件中的字数。在MySQL 8.0.4中添加。
* validate\_password.length: validate\_password要求的密码长度。在MySQL 8.0.4中添加。
* validate\_password.mixed\_case\_count：validate\_password要求的大写/小写字符数。在MySQL 8.0.4中添加。
* validate\_password.number\_count：validate\_password需要的数字字符数量。在MySQL 8.0.4中添加。
* validate\_password.policy：validate\_password密码策略。在MySQL 8.0.4中添加。
* validate\_password.special\_char\_count：validate\_password要求的特殊字符数。在MySQL 8.0.4中添加。
* version\_compile\_zlib。编译的zlib库的版本。在MySQL 8.0.11中添加。
* windowing\_use\_high\_precision。是否以高精确度计算窗口函数。在MySQL 8.0.2中添加

**在MySQL 8.0中废弃的选项和变量**

下列系统变量、状态变量和选项在MySQL 8.0中已被废弃。

压缩。客户端连接是否在客户端/服务器协议中使用压缩。在MySQL 8.0.18中已废弃。

* admin-ssl：启用连接加密。在MySQL 8.0.25中被弃用。
* expire\_logs\_days。在这么多天后清除二进制日志。在MySQL 8.0.3中被弃用。
* group\_replication\_ip\_whitelist。允许连接到组的主机列表。在MySQL 8.0.22中被废弃。
* have\_openssl：mysqld是否支持SSL连接。在MySQL 8.0.25中被弃用。
* have\_ssl：mysqld是否支持SSL连接。在MySQL 8.0.25中被弃用。
* innodb\_undo\_tablespaces。回滚段划分的表空间文件的数量。在MySQL 8.0.4中被弃用。
* log\_bin\_use\_v1\_row\_events。服务器是否正在使用版本1的二进制日志行事件。在MySQL 8.0.18中被废弃。
* log\_syslog。是否将错误日志写到syslog。在MySQL 8.0.2中已废弃。
* master-info-file。在源的二进制日志中记住源和I/O复制线程的位置和文件名。在MySQL 8.0.18中被弃用。
* master\_info\_repository。是否将包含源信息和源的二进制日志中复制I/O线程位置的连接元数据库写入文件或表。在MySQL 8.0.23中被废弃。
* max\_length\_for\_sort\_data。排序记录中的最大字节数。在MySQL 8.0.20中已废弃。
* no-dd-upgrade。防止在启动时自动升级数据字典表。在MySQL 8.0.16中已废弃。
* relay\_log\_info\_file。应用者元数据存储库的文件名，复制器在其中记录有关中继日志的信息。在MySQL 8.0.18中被弃用。
* relay\_log\_info\_repository。是否将中继日志中的复制SQL线程的位置写入文件或表。在MySQL 8.0.23中已废弃。
* slave\_compressed\_protocol。使用源/复制协议的压缩。在MySQL 8.0.18中已废弃。
* slave\_rows\_search\_algorithms。决定用于复制体更新批处理的搜索算法。从这个列表中任选2或3个。index\_search, table\_scan, hash\_scan。在MySQL 8.0.18中被弃用。
* ssl：启用连接加密。在MySQL 8.0.25中被弃用。
* 符号链接（symbolic-links）。允许MyISAM表的符号链接。在MySQL 8.0.2中被弃用。
* temptable\_use\_mmap。定义当达到temptable\_max\_ram阈值时，TempTable存储引擎是否分配内存映射的文件。在MySQL 8.0.25中被弃用。

**在MySQL 8.0中删除的选项和变量**

以下系统变量、状态变量和选项已在MySQL 8.0中被删除。

* Com\_alter\_db\_upgrade。ALTER DATABASE ...的计数
* UPGRADE DATA DIRECTORY NAME语句。在MySQL 8.0.0中已删除。
* nnodb\_available\_undo\_logs。InnoDB回滚段的总数；与innodb\_rollback\_segments不同，后者显示活动回滚段的数量。在MySQL 8.0.2中被删除。
* Qcache\_free\_blocks。查询缓存中可用的内存块的数量。在MySQL 8.0.3中被删除。
* Qcache\_free\_memory。用于查询缓存的可用内存量。在MySQL 8.0.3中被删除。
* Qcache\_hits。查询缓存的命中率。在MySQL 8.0.3中被删除。
* Qcache\_inserts。查询缓存插入的数量。在MySQL 8.0.3中被删除。
* Qcache\_lowmem\_prunes。由于缓存缺乏可用内存而从查询缓存中删除的查询数量。在MySQL 8.0.3中被删除。
* Qcache\_not\_cached。非缓存查询的数量（不可缓存，或由于query\_cache\_type设置而未缓存）。在MySQL 8.0.3中被删除。
* Qcache\_queries\_in\_cache。在查询缓存中注册的查询数量。在MySQL 8.0.3中被删除。
* Qcache\_total\_blocks。查询缓存中的块的总数。在MySQL 8.0.3中被删除。
* Slave\_heartbeat\_period。复制的心跳间隔，单位是秒。在MySQL 8.0.1中被删除。
* Slave\_last\_heartbeat。显示何时收到最新的心跳信号，格式为TIMESTAMP。在MySQL 8.0.1中被删除。
* Slave\_received\_heartbeats。自上次重置以来复制体收到的心跳次数。在MySQL 8.0.1中被删除。
* Slave\_retried\_transactions。自启动以来，复制的SQL线程重试事务的总次数。在MySQL 8.0.1中被删除。
* Slave\_running。该服务器作为副本的状态（复制I/O线程状态）。在MySQL 8.0.1中被删除。
* bootstrap：由mysql安装脚本使用。在MySQL 8.0.0中被删除。
* date\_format。DATE格式（未使用）。在MySQL 8.0.3中删除。
* datetime\_format。DATETIME/TIMESTAMP格式（未使用）。在MySQL 8.0.3中被删除。
* des-key-file。从给定文件加载用于des\_encrypt()和des\_encrypt的密钥。在MySQL 8.0.3中被删除。
* group\_replication\_allow\_local\_disjoint\_gtids\_join。允许当前服务器加入组，即使它有不存在于组中的事务。在MySQL 8.0.4中被删除。
* have\_crypt。有crypt()系统调用。在MySQL 8.0.3中被删除。
* ignore-db-dir。将目录视为非数据库目录。在MySQL 8.0.0中删除。
* ignore\_builtin\_innodb。忽略内置InnoDB。在MySQL 8.0.3中删除。
* ignore\_db\_dirs。将目录视为非数据库目录。在MySQL 8.0.0中删除。
* innodb\_checksums。启用InnoDB校验和验证。在MySQL 8.0.0中删除。
* innodb\_disable\_resize\_buffer\_pool\_debug。禁用InnoDB缓冲池的大小调整。在MySQL 8.0.0中被删除。
* innodb\_file\_format。新InnoDB表的格式。在MySQL 8.0.0中被移除。
* innodb\_file\_format\_check。InnoDB是否执行文件格式兼容性检查。在MySQL 8.0.0中被删除。
* innodb\_file\_format\_max。共享表空间中的文件格式标签。在MySQL 8.0.0中被删除。
* innodb\_large\_prefix。启用列前缀索引的长键。在MySQL 8.0.0中被移除。
* innodb\_locks\_unsafe\_for\_binlog。强制InnoDB不使用下一个键的锁定。相反，只使用行级锁。在MySQL 8.0.0中被移除。
* innodb\_scan\_directories。定义了在InnoDB恢复期间扫描表空间文件的目录。在MySQL 8.0.4中被删除。
* innodb\_stats\_sample\_pages。为索引分布统计取样的索引页的数量。在MySQL 8.0.0中被删除。
* innodb\_support\_xa。启用InnoDB对XA两阶段提交的支持。在MySQL 8.0.0中被删除。
* innodb\_undo\_logs。InnoDB使用的Undo logs（回滚段）的数量；innodb\_rollback\_segments的别名。在MySQL 8.0.2中被移除。
* internal\_tmp\_disk\_storage\_engine：内部临时表的存储引擎。在MySQL 8.0.16中被删除。
* log-warnings。将一些非关键性的警告写入日志文件。在MySQL 8.0.3中删除。
* log\_builtin\_as\_identified\_by\_password：是否以向后兼容的方式记录CREATE/ALTER USER、GRANT。在MySQL 8.0.11中被删除。
* log\_error\_filter\_rules。用于错误记录的过滤规则。在MySQL 8.0.4中被删除。
* log\_syslog。是否将错误日志写到syslog。在MySQL 8.0.13中删除。
* log\_syslog\_facility。syslog消息的设施。在MySQL 8.0.13中被删除。
* log\_syslog\_include\_pid。是否在syslog消息中包括服务器PID。在MySQL 8.0.13中被删除。
* log\_syslog\_tag。syslog消息中服务器标识符的标签。在MySQL 8.0.13中被删除。
* max\_tmp\_tables。未使用。在MySQL 8.0.3中被删除。
* metadata\_locks\_cache\_size。元数据锁缓存的大小。在MySQL 8.0.13中删除。
* metadata\_locks\_hash\_instances。元数据锁哈希值的数量。在MySQL 8.0.13中被删除。
* multi\_range\_count。在范围选择时一次发送给表处理程序的最大范围数。在MySQL 8.0.3中删除。
* old\_passwords。为PASSWORD()选择密码散列方法。在MySQL 8.0.11中被删除。
* partition（分区）。启用（或禁用）分区支持。在MySQL 8.0.0中被删除。
* query\_cache\_limit：不缓存大于此值的结果。在MySQL 8.0.3中删除。
* query\_cache\_min\_res\_unit。分配结果空间的最小单位大小（最后一个单位在写入所有结果数据后被修剪）。在MySQL 8.0.3中被删除。
* query\_cache\_size。分配用于存储旧查询结果的内存。在MySQL 8.0.3中被删除。
* query\_cache\_type。查询缓存类型。在MySQL 8.0.3中删除。
* query\_cache\_wlock\_invalidate：在LOCK上写的查询缓存中无效的查询。在MySQL 8.0.3中删除。
* secure\_auth。不允许对有旧（4.1前）密码的账户进行认证。在MySQL 8.0.3中删除。
* show\_compatibility\_56：SHOW STATUS/VARIABLES的兼容性。在MySQL 8.0.1中删除。
* skip-partition。不启用用户定义的分区。在MySQL 8.0.0中删除。
* sync\_frm。创建时将.frm同步到磁盘。默认情况下是启用的。在MySQL 8.0.0中删除。
* temp-pool。使用该选项使创建的大多数临时文件使用一组小名称，而不是为每个新文件使用唯一的名称。在MySQL 8.0.1中被删除。
* time\_format。时间格式（未使用）。在MySQL 8.0.3中删除。
* tx\_isolation。默认的事务隔离级别。在MySQL 8.0.3中被删除。
* tx\_read\_only。默认的事务访问模式。在MySQL 8.0.3中被删除

## 1.5 MySQL的信息来源

本节列出了你可能会发现有帮助的其他信息来源，如MySQL网站、邮件列表、用户论坛和Internet Relay Chat。

**MySQL网站**

MySQL文档的主要网站是https://dev.mysql.com/doc/。在线和可下载的文档格式可用于MySQL参考手册、MySQL连接器等。

MySQL开发人员以MySQL服务器博客的形式提供有关新的和即将推出的功能的信息。

**MySQL论坛上的MySQL社区支持**

http://forums.mysql.com 的论坛是一个重要的社区资源。许多论坛都是可用的，分为以下几个一般类别。

* 迁移
* MySQL的使用
* MySQL连接器
* 编程语言
* 工具类
* 第三方应用程序
* 存储引擎
* 编码技术
* SQL标准
* 业务

**MySQL企业版**

Oracle以MySQL企业版的形式提供技术支持。对于那些在关键业务生产应用中依赖MySQL数据库管理系统的组织来说，MySQL企业版是一种商业订阅产品，其中包括。

* MySQL企业服务器
* MySQL企业监控器
* 每月快速更新和季度服务包
* MySQL知识库
* 24x7技术和咨询支持

MySQL企业版有多个层级，让您可以灵活地选择最符合您需求的服务级别。欲了解更多信息，请参阅MySQL企业版。

## 1.6 如何报告缺陷或问题

在发布关于一个问题的错误报告之前，请尝试验证它是否是一个错误，以及它是否已经被报告过。

* 首先在https://dev.mysql.com/doc/ 上搜索MySQL在线手册。我们试图通过经常更新手册以解决新发现的问题来保持手册的更新性。此外，手册所附的发行说明可能特别有用，因为很可能一个较新的版本包含对你问题的解决方案。发行说明可以在刚才给出的手册的位置上找到。
* 如果你得到一个SQL语句的解析错误，请仔细检查你的语法。如果你不能发现其中的问题，那么极有可能是你当前版本的MySQL Server不支持你所使用的语法。如果你使用的是当前版本，而手册没有涵盖你所使用的语法，那么MySQL Server不支持你的语句。

如果手册涵盖了你正在使用的语法，但你有一个较早的MySQL Server版本，你应该检查MySQL的变化历史，看看该语法是何时实现的。在这种情况下，你可以选择升级到较新版本的MySQL Server。

* 关于一些常见问题的解决方案，见B.3节 "问题和常见错误"。
* 在http://bugs.mysql.com/，搜索bug数据库，看看该bug是否已经被报告和修复。
* 你还可以使用http://www.mysql.com/search/，搜索位于MySQL网站的所有网页（包括手册）。

如果你在手册、错误数据库或邮件列表档案中找不到答案，请向你当地的MySQL专家查询。如果你仍然找不到你的问题的答案，请使用以下准则来报告错误。

报告bug的正常方法是访问http://bugs.mysql.com/，这是我们bug数据库的地址。这个数据库是公开的，任何人都可以浏览和搜索。如果你登录到该系统，你可以输入新的报告。

在错误数据库中发布的错误（http://bugs.mysql.com/），如果在某个特定的版本中得到纠正，会在发布说明中注明。

如果你发现MySQL服务器的安全漏洞，请立即通过发送电子邮件到<secalert\_us@oracle.com>让我们知道。例外。支持客户应将所有问题，包括安全漏洞，报告给Oracle支持部门：http://support.oracle.com/。

要与其他用户讨论问题，你可以使用MySQL社区Slack。

写一份好的错误报告需要耐心，但第一次就把它写对了，可以为我们和你自己节省时间。一份好的bug报告，包含bug的完整测试案例，使得我们很可能在下一个版本中修复这个bug。本节将帮助您正确地写出您的报告，这样您就不会浪费您的时间去做那些可能对我们没有什么帮助或根本没有帮助的事情。请仔细阅读本节，并确保在你的报告中包含所有在此描述的信息。

最好在发布之前，你应该使用最新的生产或开发版本的MySQL服务器来测试这个问题。任何人只要在你的测试案例上使用mysql test < script\_file，或者运行你在bug报告中包含的shell或Perl脚本，就应该能够重复这个bug。任何我们能够重复的错误都有很大的机会在下一个MySQL版本中被修复。

如果在bug报告中包括对问题的良好描述，这是最有帮助的。也就是说，举出一个很好的例子，说明你所做的一切导致了这个问题，并准确详细地描述问题本身。最好的报告是那些包括一个完整的例子，显示如何重现该错误或问题。见第5.9节，"调试MySQL"。

请记住，我们有可能对包含太多信息的报告作出回应，但对包含太少信息的报告则不会。人们经常省略事实，因为他们认为他们知道问题的原因，并认为一些细节并不重要。一个好的原则是，如果你对陈述某事有疑问，就陈述它。如果我们必须要求你提供最初报告中缺少的信息，那么在报告中多写几行比等待答案更快、更省事。

错误报告中最常见的错误是：(a)不包括你所使用的MySQL发行版的版本号，以及(b)没有完全描述MySQL服务器所安装的平台（包括平台类型和版本号）。这些都是高度相关的信息，在100个案例中的99个案例中，没有这些信息，错误报告就没有用。我们经常收到这样的问题："为什么这个东西对我不起作用？" 然后我们发现，要求的功能在那个MySQL版本中没有实现，或者报告中描述的错误在较新的MySQL版本中已经被修复。错误往往是与平台有关的。在这种情况下，如果不知道操作系统和平台的版本号，我们几乎不可能修复什么。

如果你从源代码编译MySQL，记得也要提供关于你的编译器的信息，如果它与问题有关。通常人们在编译器中发现bug并认为问题与MySQL有关。大多数编译器一直在开发中，并且一个版本一个版本地变得更好。为了确定你的问题是否与你的编译器有关，我们需要知道你使用的是什么编译器。请注意，每一个编译问题都应被视为一个错误，并相应地报告。

如果一个程序产生了一个错误信息，在你的报告中包括这个信息是非常重要的。如果我们试图从档案中搜索一些东西，最好是报告的错误信息与程序产生的信息完全一致。(甚至字母大小写也要注意。)最好是复制并粘贴整个错误信息到你的报告中。你不应该试图从记忆中重现该信息。

如果你在Connector/ODBC（MyODBC）上遇到问题，请尝试生成一个跟踪文件并与你的报告一起发送。参见如何报告Connector/ODBC的问题或错误。

如果你的报告包括你用mysql命令行工具运行的测试案例的长查询输出行，你可以通过使用--垂直选项或\G语句结束符使输出更可读。本节后面的EXPLAIN SELECT例子演示了\G的使用。

请在你的报告中包括以下信息。

* 你所使用的MySQL分布的版本号（例如，MySQL 5.7.10）。你可以通过执行mysqladmin版本找到你正在运行的版本。mysqladmin程序可以在你的MySQL安装目录下的bin目录中找到。
* 你遇到问题的机器的制造商和型号。
* 操作系统的名称和版本。如果你使用Windows，你通常可以通过双击 "我的电脑 "图标并拉下 "帮助/关于Windows "菜单来获得名称和版本号。对于大多数类似Unix的操作系统，你可以通过执行命令uname -a来获得这些信息。
* 有时，内存的数量（真实和虚拟）是相关的。如果有疑问，请包括这些值。
* 你的MySQL安装中的docs/INFO\_BIN文件的内容。这个文件包含关于MySQL如何配置和编译的信息。
* 如果你使用的是MySQL软件的源码分布，包括你使用的编译器的名称和版本号。如果你有一个二进制分布，包括分布名称。
* 如果问题发生在编译期间，包括确切的错误信息，也包括发生错误的文件中违规代码周围的几行内容。
* 如果mysqld死亡，你也应该报告导致mysqld意外退出的语句。你通常可以通过在启用查询日志的情况下运行mysqld，然后在mysqld退出后查看日志来获得这些信息。见第5.9节，"调试MySQL"。
* 如果一个数据库表与问题有关，在错误报告中包括SHOW CREATE TABLE db\_name.tbl\_name语句的输出。这是一个非常简单的方法来获得数据库中任何表的定义。这些信息有助于我们创建一个与你所经历的情况相匹配的情况。
* 问题发生时有效的SQL模式可能很重要，所以请报告sql\_mode系统变量的值。对于存储过程、存储函数和触发器对象，相关的sql\_mode值是对象被创建时的有效值。对于存储过程或函数，SHOW CREATE PROCEDURE或SHOW CREATE FUNCTION语句会显示相关的SQL模式，或者你可以查询INFORMATION\_SCHEMA获得相关信息。

SELECT ROUTINE\_SCHEMA, ROUTINE\_NAME, SQL\_MODE

FROM INFORMATION\_SCHEMA.ROUTINES;

对于触发器，你可以使用这个语句。

SELECT EVENT\_OBJECT\_SCHEMA, EVENT\_OBJECT\_TABLE, TRIGGER\_NAME, SQL\_MODE

FROM INFORMATION\_SCHEMA.TRIGGERS;

* 对于与性能有关的错误或SELECT语句的问题，你应该总是包括EXPLAIN SELECT ...的输出，以及至少SELECT语句产生的行数。你还应该包括每个涉及到的表的SHOW CREATE TABLE tbl\_name的输出。你提供的关于你的情况的信息越多，就越有可能有人能帮助你。

下面是一个非常好的错误报告的例子。这些语句是使用mysql命令行工具运行的。注意对于那些否则会提供非常长的输出行而难以阅读的语句，使用了\G语句终止符。

mysql> **SHOW VARIABLES;**

mysql> **SHOW COLUMNS FROM ...\G**

***<output from SHOW COLUMNS>***

mysql> **EXPLAIN SELECT ...\G**

***<output from EXPLAIN>***

mysql> **FLUSH STATUS;**

mysql> **SELECT ...;**

***<A short version of the output from SELECT,***

***including the time taken to run the query>***

mysql> **SHOW STATUS;**

***<output from SHOW STATUS>***

* 如果在运行mysqld时出现错误或问题，尝试提供一个再现异常的输入脚本。这个脚本应该包括任何必要的源文件。脚本越能重现你的情况，就越好。如果你能做出一个可重现的测试案例，你应该把它上传到错误报告的附件中。
* 如果你不能提供一个脚本，你至少应该在你的报告中包括mysqladmin variables extended-status processlist的输出，以提供一些关于你的系统是如何执行的信息。
* 如果你不能产生一个只有几行的测试案例，或者测试表太大而不能包括在错误报告中（超过10行），你应该使用mysqldump转储你的表，并创建一个README文件来描述你的问题。使用tar和gzip或zip创建一个你的文件的压缩档案。在你为我们的bug数据库（http://bugs.mysql.com/）发起一个bug报告后，点击bug报告中的文件标签，以获得将存档上传到bug数据库的指示。
* 如果你认为MySQL服务器从一个语句中产生了一个奇怪的结果，不仅要包括结果，还要包括你对这个结果应该是什么的看法，以及描述你的看法的基础的解释。
* 当你提供一个问题的例子时，最好使用你实际情况中存在的表名、变量名等，而不是想出新的名字。问题可能与某个表或变量的名称有关。这些情况也许很少，但安全总比遗憾好。毕竟，对你来说，提供一个使用你的实际情况的例子应该更容易，而且对我们来说，这无论如何是更好的。如果你有不想让别人在错误报告中看到的数据，你可以使用前面描述的文件标签上传。如果这些信息确实是绝密的，而且你甚至不想让我们看到它，请继续使用其他名称提供一个例子，但请将此视为最后的选择。
* 如果可能的话，包括给相关程序的所有选项。例如，指出你在启动mysqld服务器时使用的选项，以及你用于运行任何MySQL客户端程序的选项。诸如mysqld和mysql等程序的选项，以及配置脚本的选项，往往是解决问题的关键，而且非常相关。把它们包括在内永远是个好主意。如果你的问题涉及到用Perl或PHP等语言编写的程序，请包括语言处理器的版本号，以及该程序使用的任何模块的版本。例如，如果你有一个使用DBI和DBD::mysql模块的Perl脚本，包括Perl、DBI和DBD::mysql的版本号。
* 如果你的问题与权限系统有关，请包括mysqladmin reload的输出，以及你在尝试连接时得到的所有错误信息。当你测试你的权限时，你应该执行mysqladmin reload版本并尝试用给你带来麻烦的程序进行连接。
* 如果你有一个错误的补丁，请包括它。但是，如果你没有提供一些必要的信息，例如显示你的补丁所修复的错误的测试案例，就不要认为这个补丁就是我们所需要的，或者我们可以使用它。我们可能会发现你的补丁有问题，或者我们可能根本不理解它。如果是这样，我们就不能使用它。
* 如果我们不能验证该补丁的确切目的，我们也不会使用它。测试案例在这里帮助我们。显示补丁可以处理所有可能发生的情况。如果我们发现一个边界情况（即使是罕见的情况），补丁将无法工作，它可能是无用的。
* 关于错误是什么，为什么会发生，或者它取决于什么的猜测通常是错误的。即使是MySQL团队也不能在不首先使用调试器来确定错误的真正原因的情况下猜测这些东西。
* 在你的错误报告中指出你已经检查了参考手册和邮件存档，以便其他人知道你已经尝试自己解决这个问题。
* 如果你的数据出现损坏，或者你在访问某个特定的表时出现错误，首先用CHECK TABLE检查你的表。如果该语句报告了任何错误。
  + InnoDB崩溃恢复机制在服务器被杀死后重新启动时处理清理工作，所以在典型的操作中，不需要 "修复 "表。如果你遇到InnoDB表的错误，重新启动服务器，看看问题是否持续存在，或者错误是否只影响了内存中的缓存数据。如果数据在磁盘上被破坏，考虑在启用innodb\_force\_recovery选项的情况下重新启动，这样你就可以转储受影响的表。
  + 对于非事务表，尝试用 REPAIR TABLE 或 myisamchk 来修复它们。见第5章，MySQL服务器管理。

如果你运行的是Windows，请使用SHOW VARIABLES LIKE 'lower\_case\_table\_names' 语句来验证lower\_case\_table\_names的值。这个变量影响服务器如何处理数据库和表名的大写字母。它对一个给定值的影响应该如第9.2.3节 "标识符大小写敏感度 "中所描述的。

* 如果你经常得到损坏的表，你应该尝试找出发生这种情况的时间和原因。在这种情况下，MySQL数据目录中的错误日志可能包含一些关于发生了什么的信息。这是名称中带有.err后缀的文件。）见第5.4.2节，"错误日志"。请在你的错误报告中包括该文件的任何相关信息。通常情况下，如果在更新过程中没有任何东西杀死它，mysqld不应该损坏一个表。如果你能找到mysqld死亡的原因，我们就会更容易为你提供问题的修复。参见B.3.1节，"如何确定导致问题的原因"。
* 如果可能的话，下载并安装最新版本的MySQL服务器，并检查它是否解决了你的问题。所有版本的MySQL软件都经过了彻底的测试，应该可以毫无问题地工作。我们相信要使一切尽可能向后兼容，你应该能够毫无困难地切换MySQL版本。见第2.1.2节，"安装哪个MySQL版本和分布"。

## 1.7 遵守MySQL标准

本节描述了MySQL与ANSI/ISO SQL标准的关系。MySQL服务器有许多对SQL标准的扩展，在这里你可以找到它们是什么以及如何使用它们。你还可以找到有关MySQL服务器缺失的功能的信息，以及如何解决一些差异的问题。

自1986年以来，SQL标准一直在发展，存在几个版本。在本手册中，"SQL-92 "是指1992年发布的标准。"SQL:1999"、"SQL:2003"、"SQL:2008 "和 "SQL:2011 "是指在相应年份发布的标准版本，最后一个是最新的版本。我们使用 "SQL标准 "或 "标准SQL "来指任何时候的SQL标准的当前版本。

我们对产品的主要目标之一是继续努力符合SQL标准，但不牺牲速度和可靠性。我们并不害怕增加对SQL的扩展或对非SQL功能的支持，如果这大大增加了MySQL服务器对大部分用户群的可用性。HANDLER接口就是这种策略的一个例子。见第13.2.4节，"HANDLER声明"。

我们继续支持事务型和非事务型数据库，以满足关键任务的24/7使用和大量Web或日志使用。

MySQL服务器最初被设计为在小型计算机系统上处理中等规模的数据库（1千万至1亿行，或每张表约100MB）。今天，MySQL服务器可以处理TB级的数据库。

我们的目标不是实时支持，尽管MySQL的复制能力提供了重要的功能。

MySQL支持ODBC 0至3.51级。

MySQL支持使用NDBCLUSTER存储引擎的高可用性数据库集群。参见第23章，MySQL NDB Cluster 8.0。

我们实现了XML功能，支持W3C XPath标准的大部分内容。参见第12.12节，"XML功能"。

MySQL支持RFC 7159所定义的本地JSON数据类型，并基于ECMAScript标准（ECMA-262）。见第11.5节，"JSON数据类型"。MySQL还实现了SQL:2016标准预发布草案所规定的SQL/JSON函数的一个子集；更多信息见第12.18节，"JSON函数"。

**选择SQL模式**

MySQL服务器可以在不同的SQL模式下运行，并且可以根据sql\_mode系统变量的值为不同的客户端应用这些模式。DBA可以设置全局SQL模式以匹配站点服务器的操作要求，而每个应用程序可以根据自己的要求设置会话SQL模式。

模式影响到MySQL支持的SQL语法和它所执行的数据验证检查。这使得在不同的环境中使用MySQL以及与其他数据库服务器一起使用MySQL更加容易。

关于设置SQL模式的更多信息，见第5.1.11节 "服务器SQL模式"。

**在ANSI模式下运行MySQL**

要在ANSI模式下运行MySQL服务器，用--ANSI选项启动mysqld。在ANSI模式下运行服务器与用下列选项启动它是一样的。

--transaction-isolation=SERIALIZABLE --sql-mode=ANSI

为了在运行时达到同样的效果，执行这两条语句。

SET GLOBAL TRANSACTION ISOLATION LEVEL SERIALIZABLE;

SET GLOBAL sql\_mode = 'ANSI';

你可以看到，将sql\_mode系统变量设置为'ANSI'可以启用所有与ANSI模式有关的SQL模式选项，如下所示。

mysql> **SET GLOBAL sql\_mode='ANSI';**

mysql> **SELECT @@GLOBAL.sql\_mode;**

->'REAL\_AS\_FLOAT,PIPES\_AS\_CONCAT,ANSI\_QUOTES,IGNORE\_SPACE,ANSI'

用--ansi模式运行服务器与将SQL模式设置为'ANSI'不太一样，因为--ansi选项还设置了事务隔离级别。

### 1.7.1 MySQL对标准SQL的扩展

MySQL服务器支持一些你不可能在其他SQL DBMS中找到的扩展。请注意，如果你使用它们，你的代码很可能无法移植到其他SQL服务器。在某些情况下，你可以通过使用以下形式的注释来编写包含MySQL扩展的代码，但仍然可以移植。

/\*! ***MySQL-specific code*** \*/

在这种情况下，MySQL服务器解析并执行注释内的代码，就像它对其他SQL语句一样，但其他SQL服务器应该忽略扩展。例如，MySQL服务器识别以下语句中的STRAIGHT\_JOIN关键字，但其他服务器不应如此。

SELECT /\*! STRAIGHT\_JOIN \*/ col1 FROM table1,table2 WHERE ...

如果你在 ! 字符后添加了一个版本号，只有当MySQL版本大于或等于指定的版本号时，才会执行注释内的语法。以下注释中的KEY\_BLOCK\_SIZE子句仅由MySQL 5.1.10或更高版本的服务器执行。

CREATE TABLE t1(a INT, KEY (a)) /\*!50110 KEY\_BLOCK\_SIZE=1024 \*/;

下面的描述列出了MySQL的扩展，按类别组织。

* 磁盘上的数据组织

MySQL服务器将每个数据库映射到MySQL数据目录下的一个目录，并将数据库中的表映射到数据库目录中的文件名。因此，在文件名区分大小写的操作系统（如大多数Unix系统）上，数据库和表名在MySQL Server中是区分大小写的。见第9.2.3节，"标识符的大小写敏感性"。

* 一般的语言语法
  + 默认情况下，字符串可以用 "以及'"括起来。如果启用了ANSI\_QUOTES SQL模式，字符串只能用'括起来，服务器将用 "括起来的字符串解释为标识符。
  + \ 是字符串中的转义字符。
  + 在SQL语句中，你可以用db\_name.tbl\_name语法访问不同数据库的表。一些SQL服务器提供同样的功能，但称之为用户空间。MySQL服务器不支持表空间，如在这样的语句中使用。CREATE TABLE ralph.my\_table ... IN my\_tablespace。
* SQL语句的语法
  + ANALYZE TABLE, CHECK TABLE, OPTIMIZE TABLE, 和 REPAIR TABLE 语句
  + CREATE DATABASE, DROP DATABASE, 和ALTER DATABASE语句。参见第13.1.12节 "CREATE DATABASE语句"，第13.1.24节 "DROP DATABASE语句"，和第13.1.2节 "ALTER DATABASE语句"。
  + DO语句。
  + EXPLAIN SELECT，获得关于表如何被查询优化器处理的描述。
  + FLUSH和RESET语句。
  + SET语句。参见第13.7.6.1节，"变量分配的SET语法"。
  + SHOW语句。见第13.7.7节，"SHOW语句"。许多MySQL特定的SHOW语句产生的信息可以通过使用SELECT查询INFORMATION\_SCHEMA而以更标准的方式获得。参见第26章， INFORMATION\_SCHEMA表。
  + 使用LOAD DATA。在许多情况下，这种语法与Oracle LOAD DATA兼容。参见章节13.2.7, "LOAD DATA语句"。
  + 使用RENAME TABLE。参见第13.1.36节，"RENAME TABLE语句"。
  + 使用REPLACE而不是DELETE加INSERT。参见第13.2.9节，"REPLACE语句"。
  + 在ALTER TABLE语句中使用CHANGE col\_name, DROP col\_name, 或DROP INDEX, IGNORE或RENAME。在一个ALTER TABLE语句中使用多个ADD、ALTER、DROP或CHANGE子句。参见章节13.1.9, "ALTER TABLE语句"。
  + 在CREATE TABLE语句中使用索引名称、列的前缀索引、以及使用INDEX或KEY。参见章节13.1.20，"CREATE TABLE语句"。
  + 在CREATE TABLE中使用TEMPORARY或IF NOT EXISTS。
  + 在DROP TABLE和DROP DATABASE中使用IF EXISTS。
  + 用一个DROP TABLE语句删除多个表的能力。
  + UPDATE 和 DELETE 语句的 ORDER BY 和 LIMIT 子句。
  + INSERT INTO tbl\_name SET col\_name = ... 语法。
  + INSERT和REPLACE语句的DELAYED子句。
  + INSERT, REPLACE, DELETE, 和UPDATE语句的LOW\_PRIORITY子句。
  + 在SELECT语句中使用INTO OUTFILE或INTO DUMPFILE。参见章节13.2.10, "SELECT语句"。
  + 在SELECT语句中使用STRAIGHT\_JOIN或SQL\_SMALL\_RESULT等选项。
  + 你不需要在GROUP BY子句中命名所有选择的列。这对于一些非常特殊，但相当正常的查询来说，会有更好的性能。参见第12.20节，"聚合函数"。
  + 你可以用GROUP BY指定ASC和DESC，而不仅仅是用ORDER BY。
  + 能够在语句中用:=赋值操作符设置变量。参见第9.4节，"用户定义的变量"。

**数据类型**

* MEDIUMINT, SET, 和ENUM数据类型，以及各种BLOB和TEXT数据类型。
* AUTO\_INCREMENT、BINARY、NULL、UNSIGNED和ZEROFILL数据类型属性。

**函数和操作符**

* 为了使从其他SQL环境迁移过来的用户更容易，MySQL服务器支持许多函数的别名。例如，所有字符串函数都支持标准SQL语法和ODBC语法。
* MySQL服务器理解||和&&运算符是指逻辑OR和AND，就像在C编程语言中一样。在MySQL Server中，||和OR是同义词，正如&&和AND一样。由于这种漂亮的语法，MySQL Server不支持用于字符串连接的标准SQL ||操作符；使用CONCAT()代替。因为CONCAT()可以接受任何数量的参数，所以很容易将||操作符的使用转换为MySQL Server。
* 使用COUNT(DISTINCT value\_list)，其中value\_list有多于一个元素。
* 字符串比较默认是不区分大小写的，排序由当前字符集的排序决定，默认是utf8mb4。要执行大小写敏感的比较，你应该用BINARY属性声明你的列，或者使用BINARY转换，这将导致使用底层字符代码值而不是词法排序来进行比较。
* %操作符是MOD()的同义词。也就是说，N % M等同于MOD(N,M)。％是为C语言程序员和与PostgreSQL兼容而支持的
* =, <>, <=, <, >=, >, <<, >>, <=>, AND, OR, 或 LIKE 操作符可以在 SELECT 语句中的输出列列表（在 FROM 的左边）的表达式中使用。例如
* mysql> **SELECT col1=1 AND col2=2 FROM my\_table;**
* LAST\_INSERT\_ID()函数返回最近的AUTO\_INCREMENT值。参见12.16节 "信息函数"。
* 在数字值上允许使用LIKE。
* REGEXP和NOT REGEXP扩展正则表达式运算符。
* CONCAT()或CHAR()有一个参数或两个以上的参数。在MySQL服务器中，这些函数可以接受可变数量的参数）。
* BIT\_COUNT()、CASE、ELT()、FROM\_DAYS()、FORMAT()、IF()、MD5()、PERIOD\_ADD()、PERIOD\_DIFF()、TO\_DAYS()和WEEKDAY()函数。
* 使用TRIM()来修剪子字符串。标准SQL只支持删除单个字符。
* GROUP BY函数STD(), BIT\_OR(), BIT\_AND(), BIT\_XOR(), 和GROUP\_CONCAT()。参见第12.20节，"聚合函数"。

### 1.7.2 MySQL与标准SQL的区别

我们努力使MySQL服务器遵循ANSI SQL标准和ODBC SQL标准，但MySQL服务器在某些情况下执行的操作不同。

* 在MySQL和标准SQL的权限系统之间有几个区别。例如，在MySQL中，当你删除一个表时，该表的权限不会自动撤销。你必须明确发出REVOKE语句来撤销一个表的权限。更多信息，请参见第13.7.1.8节，"REVOKE语句"。
* CAST()函数不支持向REAL或BIGINT的转换。参见章节12.11, "转换函数和操作符"。

#### 1.7.2.1 SELECT INTO TABLE的区别

MySQL服务器不支持SELECT ... INTO TABLE Sybase SQL扩展。相反，MySQL服务器支持INSERT INTO ... SELECT标准SQL语法，这基本上是同样的事情。见第13.2.6.1节，"INSERT ... SELECT语句"。比如说

INSERT INTO tbl\_temp2 (fld\_id)

SELECT tbl\_temp1.fld\_order\_id

FROM tbl\_temp1 WHERE tbl\_temp1.fld\_order\_id > 100;

另外，你可以使用SELECT ... INTO OUTFILE或CREATE TABLE ... SELECT。

你可以使用SELECT ... INTO和用户定义的变量。同样的语法也可以在存储程序中使用游标和局部变量。参见章节13.2.10.1, "SELECT ... INTO语句"。

另外，你可以使用SELECT ... INTO OUTFILE或CREATE TABLE ... SELECT。

你可以使用SELECT ... INTO和用户定义的变量。同样的语法也可以在存储程序中使用游标和局部变量。参见章节13.2.10.1, "SELECT ... INTO语句"。

#### 1.7.2.2 UPDATE的区别

如果你在表达式中访问要更新的表中的列，UPDATE使用该列的当前值。下面语句中的第二个赋值将col2设置为当前（更新）的col1值，而不是原来的col1值。其结果是col1和col2具有相同的值。这种行为与标准SQL不同。

UPDATE t1 SET col1 = col1 + 1, col2 = col1;

#### 1.7.2.3 FOREIGN KEY约束的差异

MySQL对外键约束的实现在以下关键方面与SQL标准不同。

* 如果在父表中有几条具有相同引用键值的记录，InnoDB会执行外键检查，就像其他具有相同键值的父行不存在一样。例如，如果你定义了一个RESTRICT类型的约束，并且有一条子记录有几条父记录，InnoDB不允许删除任何一条父记录
* 如果ON UPDATE CASCADE或者ON UPDATE SET NULL循环更新它之前在同一级联中更新过的表，它的行为就像RESTRICT。这意味着你不能使用自我参照的ON UPDATE CASCADE或ON UPDATE SET NULL操作。这是为了防止级联更新导致的无限循环。另一方面，一个自我参照的ON DELETE SET NULL是可能的，正如一个自我参照的ON DELETE CASCADE一样。级联操作的嵌套深度不能超过15层。
* 在插入、删除或更新许多行的SQL语句中，外键约束（如唯一约束）被逐行检查。当执行外键检查时，InnoDB对它必须检查的子记录或父记录设置共享行级锁。MySQL立即检查外键约束；检查不会被推迟到事务提交。根据SQL标准，默认行为应该是延迟检查。也就是说，约束只有在整个SQL语句被处理后才被检查。这意味着不可能删除一个使用外键引用自己的行。
* 没有一个存储引擎，包括InnoDB，能够识别或执行参考完整性约束定义中使用的MATCH子句。使用显式的MATCH子句不会产生指定的效果，而且会导致ON DELETE和ON UPDATE子句被忽略。应该避免指定MATCH。
* SQL标准中的MATCH子句控制了当与被引用表中的主键比较时如何处理复合（多列）外键中的NULL值。MySQL基本上实现了MATCH SIMPLE定义的语义，它允许一个外键全部或部分为空。在这种情况下，包含这种外键的（子表）行可以被插入，即使它不匹配被引用的（父）表中的任何行。(可以使用触发器来实现其他语义)。
* 由于性能的原因，MySQL要求被引用的列要有索引。然而，MySQL并不强制要求被引用的列是UNIQUE或被声明为NOT NULL。
* 引用非UNIQUE键的FOREIGN KEY约束不是标准SQL，而是InnoDB的扩展。另一方面，NDB存储引擎要求在任何作为外键引用的列上有一个明确的唯一键（或主键）。
* 在UPDATE或DELETE CASCADE等操作中，对非唯一键或包含NULL值的键的外键引用的处理并没有很好的定义。建议你使用只引用UNIQUE（包括PRIMARY）和NOT NULL键的外键。
* MySQL解析但忽略了 "内联REFERENCES规范"（如SQL标准中所定义），其中引用被定义为列规范的一部分。MySQL只接受作为单独的FOREIGN KEY规范的一部分而指定的REFERENCES条款。对于不支持外键的存储引擎（如MyISAM），MySQL服务器解析并忽略外键规范。

关于外键约束的信息，请参见第13.1.20.5节，"FOREIGN KEY约束"。

#### 1.7.2.4 '--'作为注释的开始

标准SQL使用C语法/\*这是一个注释\*/进行注释，MySQL服务器也支持这种语法。MySQL还支持该语法的扩展，使MySQL特定的SQL能够嵌入注释中，如第9.7节 "注释 "中所述。

标准SQL使用"--"作为开始-注释序列。MySQL服务器使用#作为开始注释字符。MySQL服务器还支持 -- 注释样式的一个变体。也就是说， -- 开始注释序列后面必须有一个空格（或一个控制字符，如换行）。需要空格是为了防止自动生成的SQL查询出现问题，这些查询使用诸如下面的结构，在这里我们自动插入付款的值。

UPDATE account SET credit=credit-payment

考虑一下如果payment有一个负值，如-1，会发生什么。

UPDATE account SET credit=credit--1

credit--1在SQL中是一个有效的表达式，但是--被解释为注释的开始，表达式的一部分被丢弃了。其结果是一条具有完全不同含义的语句。

UPDATE account SET credit=credit

该语句根本没有产生任何值的变化。这说明，允许注释以--开头会产生严重的后果。

使用我们的实现要求在--后面有一个空格，以便在MySQL服务器中被识别为一个开始--注释序列。因此，credit--1是可以安全使用的。

另一个安全的特点是，mysql命令行客户端会忽略以--开头的行。

### 1.7.3 MySQL如何处理约束条件

MySQL使你既能处理允许回滚的事务性表，又能处理不允许回滚的非事务性表。正因为如此，约束处理在MySQL中与其他DBMS中有些不同。我们必须处理这样的情况：你在一个非事务表中插入或更新了大量的行，当发生错误时，这些变化不能回滚。

基本的理念是，MySQL服务器在解析要执行的语句时，会尝试对任何可以检测到的东西产生错误，并尝试从执行语句时发生的任何错误中恢复。我们在大多数情况下都会这样做，但还不是所有的情况。

当错误发生时，MySQL的选择是在中间停止语句，或者尽可能地从问题中恢复并继续。默认情况下，服务器遵循后一种路线。这意味着，例如，服务器可能会将无效的值胁迫到最接近的有效值。

有几个SQL模式的选项可以为处理坏的数据值以及在发生错误时是继续执行语句还是中止执行提供更大的控制。使用这些选项，你可以将MySQL服务器配置为以更传统的方式行事，与其他拒绝不正当输入的DBMS一样。SQL模式可以在服务器启动时全局设置，以影响所有客户端。单个客户端可以在运行时设置SQL模式，这使得每个客户端可以选择最适合其要求的行为。参见第5.1.11节，"服务器SQL模式"。

下面几节描述了MySQL服务器如何处理不同类型的约束。

#### 1.7.3.1 PRIMARY KEY和UNIQUE索引约束

通常，对于违反主键、唯一键或外键约束的数据更改语句（如INSERT或UPDATE），会发生错误。如果你使用的是InnoDB这样的事务性存储引擎，MySQL会自动回滚该语句。如果你使用非事务存储引擎，MySQL会在发生错误的行处停止处理语句，并留下未处理的任何剩余行。

MySQL支持INSERT、UPDATE等的IGNORE关键字。如果你使用它，MySQL将忽略主键或唯一键的违反，并继续处理下一行。参见你正在使用的语句的章节（第13.2.6节，"INSERT语句"，第13.2.13节，"UPDATE语句"，等等）。

你可以用mysql\_info()C API函数获得关于实际插入或更新的行数的信息。你也可以使用SHOW WARNINGS语句。参见mysql\_info()，以及第13.7.7.42节 "SHOW WARNINGS语句"。

InnoDB和NDB表支持外键。参见章节1.7.3.2, "FOREIGN KEY约束条件"。

#### 1.7.3.2 FOREIGN KEY限制条件

外键让你在不同的表中交叉引用相关的数据，而外键约束有助于保持这种分散的数据的一致性。

MySQL在CREATE TABLE和ALTER TABLE语句中支持ON UPDATE和ON DELETE外键引用。可用的引用动作是RESTRICT、CASCADE、SET NULL和NO ACTION（默认）。

SET DEFAULT也被MySQL服务器支持，但目前被InnoDB拒绝为无效。由于MySQL不支持延迟约束检查，NO ACTION被视为RESTRICT。关于MySQL支持的外键的确切语法，见第13.1.20.5节，"FOREIGN KEY约束"。

允许使用MATCH FULL、MATCH PARTIAL和MATCH SIMPLE，但应避免使用它们，因为它们导致MySQL服务器忽略同一语句中使用的任何ON DELETE或ON UPDATE子句。MATCH选项在MySQL中没有任何其他效果，它实际上是全时执行MATCH SIMPLE语义的。

MySQL要求外键列是有索引的；如果你创建一个有外键约束的表，但在一个给定的列上没有索引，就会创建一个索引。

你可以从INFORMATION\_SCHEMA.KEY\_COLUMN\_USAGE表中获得关于外键的信息。这里显示了一个针对这个表的查询的例子。

mysql> **SELECT TABLE\_SCHEMA, TABLE\_NAME, COLUMN\_NAME, CONSTRAINT\_NAME**

> **FROM INFORMATION\_SCHEMA.KEY\_COLUMN\_USAGE**

> **WHERE REFERENCED\_TABLE\_SCHEMA IS NOT NULL;**

+--------------+---------------+-------------+-----------------+

| TABLE\_SCHEMA | TABLE\_NAME | COLUMN\_NAME | CONSTRAINT\_NAME |

+--------------+---------------+-------------+-----------------+

| fk1 | myuser | myuser\_id | f |

| fk1 | product\_order | customer\_id | f2 |

| fk1 | product\_order | product\_id | f1 |

+--------------+---------------+-------------+-----------------+

3 rows in set (0.01 sec)

关于InnoDB表的外键信息也可以在INFORMATION\_SCHEMA数据库的INNODB\_FOREIGN和INNODB\_FOREIGN\_COLS表中找到。

InnoDB和NDB表支持外键。

#### 1.7.3.3 无效数据的强制约束

默认情况下，MySQL 8.0拒绝无效或不适当的数据值，并中止出现这些数据的语句。可以通过禁用严格的SQL模式（见第5.1.11节，"服务器SQL模式"）来改变这种行为，使其对无效值更加宽容，从而使服务器强制它们成为数据输入的有效值，但不建议这样做。

旧版本的MySQL默认采用宽容的行为；关于这种行为的描述，见无效数据的约束。

#### 1.7.3.4 ENUM和SET约束条件

ENUM和SET列提供了一种有效的方法来定义只能包含一组给定值的列。参见第11.3.5节 "ENUM类型 "和第11.3.6节 "SET类型"。

除非禁用严格模式（不推荐使用，但请参见第5.1.11节 "服务器SQL模式"），否则ENUM或SET列的定义对输入该列的值起到了约束作用。对于不满足这些条件的值会发生错误。

* ENUM值必须是列定义中列出的值之一，或者是其内部的数字等价物。该值不能是错误值（即0或空字符串）。对于一个定义为ENUM('a','b','c')的列，诸如'','d',或'ax'的值是无效的，会被拒绝。
* 一个SET值必须是空字符串或仅由列定义中列出的值组成，并以逗号分隔。对于一个定义为SET('a','b','c')的列，诸如'd'或'a,b,c,d'的值是无效的，会被拒绝。

如果你使用INSERT IGNORE或UPDATE IGNORE，无效值的错误可以在严格模式下被抑制。在这种情况下，会产生一个警告而不是一个错误。对于ENUM，该值被插入作为错误成员（0）。对于SET来说，除了删除任何无效的子串之外，值会按照给定的方式插入。例如，'a,x,b,y'的结果是一个'a,b'的值。

## **1.8 贡献者**

以下各节列出了帮助使MySQL成为今天这个样子的开发者、贡献者和支持者**。**

### **1.8.1 MySQL的贡献者**

尽管甲骨文公司和/或其附属机构拥有MySQL服务器和MySQL手册的所有版权，但我们希望承认那些对MySQL发行版作出了这样或那样贡献的人。贡献者被列在这里，顺序有点随机**。**

* Gianmassimo Vigazzola <qwerg@mbox.vol.it> 或 <qwerg@tin.it>

**最初移植到Win32/NT**

* **Per Eric Olsson**

**对于动态记录格式的建设性批评和实际测试。**

* **Irena Pancirov <irena@mail.yacc.it>**

**使用Borland编译器的Win32移植。 mysqlshutdown.exe和mysqlwatch.exe。**

* **David J. Hughes**

**为了努力做一个共享软件的SQL数据库。在TcX，即MySQL AB的前身，我们开始使用mSQL，但发现它不能满足我们的目的，所以我们改为为我们**的应用程序构建器Unireg写了一个SQ**L接口。mysqladmin和mysql client是主要受mSQL对应程序影响的程序。我们花了很多精力使MySQL的语法成为mSQL的超集。API的许多想法都是从mSQL借来的，以使免费的mSQL程序很容易移植到MySQL API上。MySQL软件不包含任何来自mSQL的代码。发行版中的两个文件（client/insert\_test.c和client/select\_test.c）是基于mSQL发行版中相应的（非版权）文件，但作为例子被修改，显示了将代码从mSQL转换到MySQL服务器所需的修改。(mSQL的版权归David J. Hughes所有。)**

* **Patrick Lynch**

**为帮助我们获得http://www.mysql.com/。**

* **Fred Lindberg**

**为设置qmail来处理MySQL邮件列表，以及为我们在管理MySQL邮件列表中得到的难以置信的帮助。**

* **Igor Romanenko <igor@frog.kiev.ua>**

**mysqldump（以前是msqldump，但由Monty移植和增强）。**

* **Yuri Dario**

**为了保持和扩展MySQL OS/2的移植。**

* **Tim Bunce**

**mysqlhotcopy的作者。**

* **Zarko Mocnik <zarko.mocnik@dem.si>**

**为斯洛文尼亚语进行排序。**

* **"TAMITO" <tommy@valley.ne.jp>**

**\_MB字符集宏以及ujis和sjis字符集。**

* **Joshua Chamas <joshua@chamas.com>**

**并发插入、扩展日期语法、在NT上调试的基础，以及在MySQL邮件列表上的回答。**

* **Yves Carlier <Yves.Carlier@rug.ac.be>**

**mysqlaccess，一个显示用户访问权限的程序。**

* **Rhys Jones <rhys@wales.com> (And GWE Technologies Limited)**

**用于早期的JDBC驱动之一。**

* **朱晓坤博士 <X.Zhu@brad.ac.uk>**

**进一步开发早期的JDBC驱动之一和其他与MySQL有关的Java工具。**

* **James Cooper <pixel@organic.com>**

**因为在他的网站上建立了一个可搜索的邮件列表档案。**

* **Rick Mehalick <Rick\_Mehalick@i-o.com>**

**用于xmysql，一个用于MySQL服务器的图形化X客户端。**

* **Doug Sisk <sisk@wix.com>**

**为Red Hat Linux提供MySQL的RPM包。**

* **Diemand Alexander V. <axeld@vial.ethz.ch>**

**为Red Hat Linux-Alpha提供MySQL的RPM包。**

* **Antoni Pamies Olive <toni@readysoft.es>**

**为英特尔和SPARC提供大量MySQL客户端的RPM版本。**

* **Jay Bloodworth <jay@pathways.sde.state.sc.us>**

**为MySQL 3.21提供了RPM版本。**

* **David Sacerdote <davids@secnet.com>**

**关于安全检查DNS主机名的想法。**

* **Wei-Jou Chen <jou@nematic.ieo.nctu.edu.tw>**

**对中文(BIG5)字符的一些支持。**

* **Wei He <hewei@mail.ied.ac.cn>**

**为中文(GBK)字符集提供了很多功能。**

* **Jan Pazdziora <adelton@fi.muni.cz>**

**捷克语的排序顺序。**

* **Zeev Suraski <bourbon@netvision.net.il>**

**FROM\_UNIXTIME()时间格式化，ENCRYPT()函数，以及bison顾问。活跃的邮件列表成员。**

* **Luuk de Boer <luuk@wxs.nl>**

**移植（和扩展）了基准套件到 DBI/DBD。对crash-me和运行基准有很大帮助。一些新的日期函数。mysql\_setpermission脚本。**

* **Alexis Mikhailov <root@medinf.chuvashia.su>**

**用户定义的函数（UDFs）；CREATE FUNCTION和DROP FUNCTION。**

* **Andreas F. Bobak <bobak@relog.ch>**

**AGGREGATE扩展到用户定义的函数。**

**Ross Wakelin <R.Wakelin@march.co.uk>**

**帮助设置InstallShield for MySQL-Win32。**

**Jethro Wright III <jetman@li.net>**

**libmysql.dll库。**

* **James Pereria <jpereira@iafrica.com>**

**Mysqlmanager，一个用于管理MySQL服务器的Win32 GUI工具。**

* **Curt Sampson <cjs@portal.ca>**

**将MIT-pthreads移植到NetBSD/Alpha和NetBSD 1.3/i386。**

* **Martin Ramsch <m.ramsch@computer.org>**

**MySQL教程中的例子。**

* **Steve Harvey**

**用于使mysqlaccess更加安全。**

**持久系统私人有限公司的Konark IA-64中心**

**对MySQL服务器的Win64端口的帮助。**

* **Albert Chin-A-Young.**

**为Tru64配置更新，大文件支持和更好的TCP包装器支持。**

* **John Birrell**

**对OS/2的pthread\_mutex()进行仿真。**

* **Benjamin Pflugmann**

**扩展MERGE表以处理INSERTS。是MySQL邮件列表中的活跃成员**

* **Jocelyn Fournier**

**优秀的发现和报告无数的bug（特别是在MySQL 4.1的子查询代码中）。**

### **1.8.2 文档作者和翻译者**

**以下人员帮助我们编写了MySQL文档和翻译了MySQL中的文档或错误信息。**

* **Kim Aldale**

**帮助将Monty和David的早期尝试改写成英文。**

* **Michael J. Miller Jr. <mke@terrapin.turbolift.com>**

**为第一本MySQL手册。以及为FAQ（很久以前变成了MySQL手册）做了大量的拼写/语言修正。**

* **严彩玲**

**2000年初将MySQL参考手册翻译成简体中文的第一人，Big5和香港的编码版本就是基于此。**

* **Jay Flaherty <fty@mediapulse.com>**

**手册中Perl DBI/DBD部分的大部头。**

* **Paul Southworth <pauls@etext.org>, Ray Loyzaga <yar@cs.su.oz.au>**

**对《参考手册》进行校对。**

* **Therrien Gilbert <gilbert@ican.net>, Jean-Marc Pouyot <jmp@scalaire.fr>**

**法语错误信息。**

* **Petr Snajdr, <snajdr@pvt.net>**

**捷克语错误信息。**

* **Jaroslaw Lewandowski <jotel@itnet.com.pl>**

**波兰语错误信息。**

* **Miguel Angel Fernandez Roiz**

**西班牙的错误信息。**

* **Roy-Magne Mo <rmo@www.hivolda.no>**

**挪威的错误信息和MySQL 3.21.xx的测试。**

* **Timur I. Bakeyev <root@timur.tatarstan.ru>**

**俄罗斯的错误信息。**

* **<brenno@dewinter.com> & Filippo Grassilli <phil@hyppo.com>**

**意大利语错误信息。**

* **Dirk Munzinger <dirk@trinity.saar.de>**

**德国的错误信息。**

* **Billik Stefan <billik@sun.uniag.sk>**

**斯洛伐克语的错误信息。**

* **Stefan Saroiu <tzoompy@cs.washington.edu>**

**罗马尼亚语的错误信息。**

* **Peter Feher**

**匈牙利语错误信息。**

* **Roberto M. Serqueira**

**葡萄牙语错误信息。**

* **Carsten H. Pedersen**

**丹麦语错误信息。**

* **Arjen Lentz**

**荷兰语错误信息，完成了早期的部分翻译（还包括一致性和拼写方面的工作）**

### **1.8.3 支持MySQL的软件包**

以下是一些最**重要的**API/软件包/应用程序的创建者/维护者的列表，很多人都在使用MySQL

我们不能在这里列出每一个可能的包，因为这样的列表就很难维护。对于其他软件包，请参考软件门户http://solutions.mysql.com/software/

* **Tim Bunce, Alligator Descartes**

**对于DBD（Perl）接口。**

* **Andreas Koenig <a.koenig@mind.de>**

**用于MySQL服务器的Perl接口。**

* **Jochen Wiedmann <wiedmann@neckar-alb.de>**

**用于维护Perl DBD::mysql模块。**

* **Eugene Chan <eugene@acenet.com.sg>**

**用于为MySQL服务器移植PHP。**

* **Georg Richter**

**负责MySQL 4.1的测试和bug查找。新的PHP 5.0 mysqli扩展（API）用于MySQL 4.1及以上版本。**

* **Giovanni Maruzzelli <maruzz@matrice.it>**

**用于移植iODBC（Unix ODBC）。**

* **Xavier Leroy <Xavier.Leroy@inria.fr>**

**LinuxThreads（由Linux上的MySQL服务器使用）的作者。**

### **1.8.4 用于创建MySQL的工具**

**下面是我们用于创建MySQL的一些工具的列表。我们用它来表达我们对创造这些工具的人的感谢，因为没有这些工具，我们不可能使MySQL成为今天这样。**

* **自由软件基金会**

**我们从他们那里得到了优秀的编译器（gcc）、优秀的调试器（gdb）和libc库（我们从它那里借用了strto.c来使一些代码在Linux中工作）。**

* **自由软件基金会和XEmacs开发团队**

**为了一个真正伟大的编辑器/环境。**

* **朱利安-苏厄德**

**valgrind的作者，一个优秀的内存检查工具，帮助我们发现了很多原本在MySQL中很难发现的bug。**

* **Dorothea Lütkehaus 和 Andreas Zeller**

**DDD（数据显示调试器）的作者，它是gdb的一个优秀的图形化前端）。**

### 1.8.5 MySQL的支持者

尽管Oracle公司和/或其附属机构拥有MySQL服务器和MySQL手册的所有版权，但我们希望承认以下公司，它们帮助我们资助了MySQL服务器的开发，例如为我们开发一个新的功能支付报酬，或为我们开发MySQL服务器提供硬件。

* VA Linux / Andover.net

资助了复制。

* NuSphere

编辑MySQL手册。

* Stork设计工作室

1998-2000年间使用的MySQL网站。

* 英特尔

为Windows和Linux平台的开发做出了贡献。

* 康柏公司

为在Linux/Alpha平台上的开发做出了贡献

* SWSoft

在嵌入式mysqld版本上的开发。

* FutureQuest

--skip-show-database选项

# 安装或升级MySQL

**章节目录**

本章描述了如何获得和安装MySQL。下面是该程序的摘要，后面的章节提供了细节。如果你计划将现有版本的MySQL升级到较新的版本，而不是首次安装MySQL，请参见第2.11节 "升级MySQL"，以了解有关升级程序和升级前应考虑的问题的信息。

如果你对从另一个数据库系统迁移到MySQL感兴趣，请参阅A.8节 "MySQL 8.0常见问题。迁移"，其中包含对一些有关迁移问题的常见问题的回答。

MySQL的安装通常遵循这里概述的步骤。

1、确定MySQL是否在你的平台上运行并被支持。

请注意，并非所有平台都同样适合运行MySQL，而且并非所有已知MySQL可以运行的平台都得到甲骨文公司的正式支持。关于那些正式支持的平台的信息，请参见MySQL网站上的<https://www.mysql.com/support/supportedplatforms/database.html>。

2、选择要安装的版本。

有几个版本的MySQL可用，而且大多数都有几种发布格式。你可以从包含二进制（预编译）程序或源代码的预包装发行版中进行选择。如果有疑问，请使用二进制发行版。甲骨文公司还为那些想看最新发展和测试新代码的人提供了对MySQL源代码的访问。要确定你应该使用哪个版本和发行版的类型，请参见第2.1.2节，"安装哪个MySQL版本和发行版"。、

3、下载你要安装的发行版。

有关说明，见第2.1.3节，"如何获得MySQL"。要验证发行版的完整性，请使用第2.1.4节 "使用MD5校验和或GnuPG验证包的完整性 "中的说明。

4、安装该发行版。

要从二进制发行版安装MySQL，请使用第2.2节 "使用通用二进制文件在Unix/Linux上安装MySQL "中的说明。或者，使用《安全部署指南》，该指南提供了部署MySQL企业版服务器通用二进制发行版的程序，该发行版具有管理MySQL安装安全的功能。

要从源发布版或当前开发源树中安装MySQL，请使用第2.9节 "从源安装MySQL "中的说明。

5、执行任何必要的安装后设置。

在安装完MySQL后，请参阅第2.10节 "安装后的设置和测试"，了解有关确保MySQL服务器正常工作的信息。还请参考第2.10.4节 "确保初始MySQL账户的安全 "中提供的信息。本节描述了如何保护初始MySQL根用户账户，该账户在你分配密码之前没有密码。无论你使用二进制或源码发行版安装MySQL，本节都适用。

6、如果你想运行MySQL的基准脚本，必须有对MySQL的Perl支持。见第2.13节，"Perl安装说明"。

在不同的平台和环境中安装MySQL的说明可按平台来进行。

* Unix、Linux、FreeBSD

关于在大多数Linux和Unix平台上使用通用二进制文件（例如，.tar.gz包）安装MySQL的说明，见第2.2节，"在Unix/Linux上使用通用二进制文件安装MySQL"。

关于完全从源代码发行版或源代码库构建MySQL的信息，见第2.9节，"从源代码安装MySQL"

关于安装、配置和从源码构建的具体平台帮助，见相应的平台部分。

* + Linux，包括关于发行版特定方法的说明，见第2.5节，"在Linux上安装MySQL"。
  + IBM AIX，见第2.7节，"在Solaris上安装MySQL"。
  + FreeBSD，见第2.8节，"在FreeBSD上安装MySQL"。
* Windows系统

关于使用MySQL安装程序或Zipped二进制文件在Microsoft Windows上安装MySQL的说明，见第2.3节，"在Microsoft Windows上安装MySQL"。

关于使用Microsoft Visual Studio从源代码构建MySQL的细节和说明，见第2.9节，"从源代码安装MySQL"。

* macOS

关于在macOS上的安装，包括使用二进制包和本地PKG格式，见第2.4节，"在macOS上安装MySQL"。

关于利用macOS启动守护程序来自动启动和停止MySQL的信息，见第2.4.3节，"安装和使用MySQL启动守护程序"。

关于MySQL首选项窗格的信息，见第2.4.4节，"安装和使用MySQL首选项窗格"。

## 2.1 一般安装指南

紧接着的章节包含了选择、下载和验证你的发行版的必要信息。本章后面几节的说明描述了如何安装你选择的发行版。对于二进制发行版，请参见第2.2节 "在Unix/Linux上使用通用二进制文件安装MySQL "的说明，或你的平台的相应章节（如果有）。要从源码构建MySQL，请使用第2.9节 "从源码安装MySQL "中的说明。

### 2.1.1 支持的平台

MySQL平台的支持随着时间的推移而变化，请参考https://www.mysql.com/support/supportedplatforms/database.html，以了解最新的更新。

### 2.1.2 安装哪个MySQL版本和发行版

当准备安装MySQL时，要决定使用哪种版本和发布格式（二进制或源代码）。

首先，决定是安装开发版本还是通用版本（GA）。开发版本有最新的功能，但不建议用于生产。GA版本，也被称为生产或稳定版本，是为生产使用准备的。我们建议使用最新的GA版本。

MySQL 8.0中的命名方案使用由三个数字和一个可选的后缀组成的版本名称（例如，mysql-8.0.1-dmr）。版本名称中的数字解释如下。

* 第一个数字（8）是主要版本号。
* 第二个数字(0)是次要版本号。主版本号和次版本号加在一起，构成了版本系列号。该系列号描述了稳定的功能集。
* 第三个数字（1）是版本系列中的版本号。每一个新的错误修复版本都会递增。在大多数情况下，一个系列中的最新版本是最好的选择。

版本名称也可以包括一个后缀，以表示该版本的稳定性水平。一个系列中的版本通过一组后缀来表示稳定性水平的提高。可能的后缀是。

* dmr表示一个开发里程碑版本（DMR）。MySQL开发使用一个里程碑模型，每个里程碑都引入了一小部分经过彻底测试的特性。从一个里程碑到下一个里程碑，根据试用这些早期版本的社区成员提供的反馈，功能界面可能会发生变化，甚至可能被删除。里程碑版本中的功能可以被认为是生产前的质量。
* rc表示候选发布版（RC）。候选版本被认为是稳定的，已经通过了MySQL的所有内部测试。在RC版本中仍可能引入新的功能，但重点转向修复错误，以稳定该系列中早期引入的功能。
* 没有后缀的版本表示是通用版本（GA）或生产版本。GA版本是稳定的，已经成功通过了早期的发布阶段，并且被认为是可靠的，没有严重的错误，适合在生产系统中使用。

一个系列内的开发从DMR版本开始，接着是RC版本，最后达到GA状态的版本。

在选择了要安装的MySQL版本后，决定为你的操作系统安装哪种发布格式。对于大多数使用情况，二进制发行版是正确的选择。二进制发行版可用于许多平台的本地格式，如Linux的RPM包或macOS的DMG包。分发版也可以用更通用的格式，如Zip档案或压缩的tar文件。在Windows上，你可以使用MySQL安装程序来安装二进制发行版。

在某些情况下，从源码发行版安装MySQL可能更可取。

* 你想在某个明确的位置安装MySQL。标准的二进制发行版可以在任何安装位置运行，但你可能需要更多的灵活性来把MySQL组件放在你想要的地方。
* 你想用可能不包括在标准二进制发行版中的功能来配置mysqld。下面是一个最常见的额外选项的列表，用于确保功能的可用性。
  + -DWITH\_LIBWRAP=1用于支持TCP包装器。
  + -DWITH\_ZLIB={system|bundled}，用于依赖压缩的功能。
  + -DWITH\_DEBUG=1用于调试支持

有关其他信息，请参见第2.9.7节，"MySQL源配置选项"。

* 你想配置mysqld，但不包括标准二进制发行版中的某些功能。
* 你想阅读或修改组成MySQL的C和C++代码。为此目的，获得一个源码发行版。
* 源码发行版比二进制发行版包含更多的测试和例子。

### 2.1.3 如何获得MySQL

请查看我们的下载页面https://dev.mysql.com/downloads/，了解有关当前版本的MySQL的信息和下载说明。

对于使用Yum作为其软件包管理系统的基于RPM的Linux平台，可以使用MySQL Yum Repository来安装MySQL。详情见第2.5.1节，"使用MySQL Yum仓库在Linux上安装MySQL"。

对于基于Debian的Linux平台，可以使用MySQL APT资源库来安装MySQL。详情请参阅第2.5.2节 "使用MySQL APT资源库在Linux上安装MySQL"。

对于SUSE Linux Enterprise Server（SLES）平台，可以使用MySQL SLES资源库安装MySQL。详情见第2.5.3节 "使用MySQL SLES资源库在Linux上安装MySQL"。

要获得最新的开发源码，见第2.9.5节 "使用开发源码树安装MySQL"。

### 2.1.4 使用MD5校验和或GnuPG验证软件包的完整性

在下载了适合你的需求的MySQL包后，在试图安装它之前，要确保它是完整的，没有被篡改过。有三种完整性检查的手段。

* MD5校验和
* 使用GNUPG（GNU Privacy Guard）的加密签名
* 对于RPM包，内置的RPM完整性验证机制

下面的章节描述了如何使用这些方法。

如果你发现MD5校验和或GPG签名不匹配，首先尝试再下载一次相应的包，也许是从另一个镜像站点。

#### 2.1.4.1 验证MD5校验和

在你下载了一个MySQL包之后，你应该确保它的MD5校验和与MySQL下载页面上提供的校验和一致。每个包都有一个单独的校验和，你可以对照你下载的包来验证。正确的MD5校验和在每个MySQL产品的下载页面上列出；你应该将它与你下载的文件（产品）的MD5校验和进行比较。

每个操作系统和设置都提供自己版本的工具来检查MD5校验和。通常情况下，该命令被命名为md5sum，或者它可能被命名为md5，一些操作系统根本就不提供它。在Linux上，它是GNU Text Utilities软件包的一部分，该软件包可用于各种不同的平台。你也可以从 http://www.gnu.org/software/textutils/ 下载源代码。如果你已经安装了OpenSSL，你可以使用openssl md5 package\_name命令来代替。md5命令行工具的Windows实现可以从http://www.fourmilab.ch/md5/。winMd5Sum是一个图形化的MD5检查工具，可以从http://www.nullriver.com/index/products/winmd5sum。我们的Windows系统示例假定名称为md5.exe。

Linux和Microsoft Windows的例子。

shell> **md5sum mysql-standard-8.0.26-linux-i686.tar.gz**

aaab65abbec64d5e907dcd41b8699945 mysql-standard-8.0.26-linux-i686.tar.gz

shell> **md5.exe mysql-installer-community-8.0.26.msi**

aaab65abbec64d5e907dcd41b8699945 mysql-installer-community-8.0.26.msi

你应该验证所产生的校验和（十六进制数字的字符串）与紧接在相应软件包下面的下载页面上显示的校验和相匹配。

注意

*确保验证归档文件（例如，.zip、.tar.gz或.msi文件）的校验和，而不是归档文件中包含的文件。换句话说，在提取文件内容之前要先验证该文件。*

#### 2.1.4.2 使用GnuPG进行签名检查

另一种验证包的完整性和真实性的方法是使用加密签名。这比使用MD5校验和更可靠，但需要更多工作。

我们用GnuPG（GNU Privacy Guard）来签署MySQL可下载包。GnuPG是Phil Zimmermann著名的Pretty Good Privacy（PGP）的一个开源替代方案。大多数Linux发行版都默认安装了GnuPG。否则，请参阅 http://www.gnupg.org/ 了解更多关于 GnuPG 的信息以及如何获得和安装它。

为了验证特定软件包的签名，你首先需要获得一份我们公开的GPG构建密钥的副本，你可以从http://pgp.mit.edu/ 下载。你想获得的密钥名为mysql-build@oss.oracle.com。另外，你也可以直接从以下文本中复制和粘贴密钥。

-----BEGIN PGP PUBLIC KEY BLOCK-----

Version: GnuPG v1

mQGiBD4+owwRBAC14GIfUfCyEDSIePvEW3SAFUdJBtoQHH/nJKZyQT7h9bPlUWC3

RODjQReyCITRrdwyrKUGku2FmeVGwn2u2WmDMNABLnpprWPkBdCk96+OmSLN9brZ

fw2vOUgCmYv2hW0hyDHuvYlQA/BThQoADgj8AW6/0Lo7V1W9/8VuHP0gQwCgvzV3

BqOxRznNCRCRxAuAuVztHRcEAJooQK1+iSiunZMYD1WufeXfshc57S/+yeJkegNW

hxwR9pRWVArNYJdDRT+rf2RUe3vpquKNQU/hnEIUHJRQqYHo8gTxvxXNQc7fJYLV

K2HtkrPbP72vwsEKMYhhr0eKCbtLGfls9krjJ6sBgACyP/Vb7hiPwxh6rDZ7ITnE

kYpXBACmWpP8NJTkamEnPCia2ZoOHODANwpUkP43I7jsDmgtobZX9qnrAXw+uNDI

QJEXM6FSbi0LLtZciNlYsafwAPEOMDKpMqAK6IyisNtPvaLd8lH0bPAnWqcyefep

rv0sxxqUEMcM3o7wwgfN83POkDasDbs3pjwPhxvhz6//62zQJ7Q2TXlTUUwgUmVs

ZWFzZSBFbmdpbmVlcmluZyA8bXlzcWwtYnVpbGRAb3NzLm9yYWNsZS5jb20+iGwE

ExECACwCGyMCHgECF4ACGQEGCwkIBwMCBhUKCQgCAwUWAgMBAAUCXEBY+wUJI87e

5AAKCRCMcY07UHLh9RZPAJ9uvm0zlzfCN+DHxHVaoFLFjdVYTQCfborsC9tmEZYa

whhogjeBkZkorbyIaQQTEQIAKQIbIwYLCQgHAwIEFQIIAwQWAgMBAh4BAheAAhkB

BQJTAdRmBQkaZsvLAAoJEIxxjTtQcuH1X4MAoKNLWAbCBUj96637kv6Xa/fJuX5m

AJwPtmgDfjUe2iuhXdTrFEPT19SB6ohmBBMRAgAmAhsjBgsJCAcDAgQVAggDBBYC

AwECHgECF4AFAk53PioFCRP7AhUACgkQjHGNO1By4fUmzACeJdfqgc9gWTUhgmcM

AOmG4RjwuxcAoKfM+U8yMOGELi+TRif7MtKEms6piGkEExECACkCGyMGCwkIBwMC

BBUCCAMEFgIDAQIeAQIXgAIZAQUCUZSROgUJFTchqgAKCRCMcY07UHLh9YtAAJ9X

rA/ymlmozPZn+A9ls8/uwMcTsQCfaQMNq1dNkhH2kyByc3Rx9/W2xfqJARwEEAEC

AAYFAlAS6+UACgkQ8aIC+GoXHivrWwf/dtLk/x+NC2VMDlg+vOeM0qgG1IlhXZfi

NsEisvvGaz4m8fSFRGe+1bvvfDoKRhxiGXU48RusjixzvBb6KTMuY6JpOVfz9Dj3

H9spYriHa+i6rYySXZIpOhfLiMnTy7NH2OvYCyNzSS/ciIUACIfH/2NH8zNT5CNF

1uPNRs7HsHzzz7pOlTjtTWiF4cq/Ij6Z6CNrmdj+SiMvjYN9u6sdEKGtoNtpycgD

5HGKR+I7Nd/7v56yhaUe4FpuvsNXig86K9tI6MUFS8CUyy7Hj3kVBZOUWVBM053k

nGdALSygQr50DA3jMGKVl4ZnHje2RVWRmFTr5YWoRTMxUSQPMLpBNIkBHAQQAQIA

BgUCU1B+vQAKCRAohbcD0zcc8dWwCACWXXWDXIcAWRUw+j3ph8dr9u3SItljn3wB

c7clpclKWPuLvTz7lGgzlVB0s8hH4xgkSA+zLzl6u56mpUzskFl7f1I3Ac9GGpM4

0M5vmmR9hwlD1HdZtGfbD+wkjlqgitNLoRcGdRf/+U7x09GhSS7Bf339sunIX6sM

gXSC4L32D3zDjF5icGdb0kj+3lCrRmp853dGyA3ff9yUiBkxcKNawpi7Vz3D2ddU

pOF3BP+8NKPg4P2+srKgkFbd4HidcISQCt3rY4vaTkEkLKg0nNA6U4r0YgOa7wIT

SsxFlntMMzaRg53QtK0+YkH0KuZR3GY8B7pi+tlgycyVR7mIFo7riQEcBBABCAAG

BQJWgVd0AAoJEEZu4b/gk4UKk9MH/Rnt7EccPjSJC5CrB2AU5LY2Dsr+PePI2ubP

WsEdG82qSjjGpbhIH8LSg/PzQoGHiFWMmmZWJktRT+dcgLbs3b2VwCNAwCE8jOHd

UkQhEowgomdNvHiBHKHjP4/lF68KOPiO/2mxYYkmpM7BWf3kB57DJ5CTi3/JLoN7

zF40qIs/p09ePvnwStpglbbtUn7XPO+1/Ee8VHzimABom52PkQIuxNiVUzLVn3bS

Wqrd5ecuqLk6yzjPXd2XhDHWC9Twpl68GePru6EzQtusi0m6S/sHgEXqh/IxrFZV

JlljF75JvosZq5zeulr0i6kOij+Y1p6MFffihITZ1gTmk+CLvK2JASIEEAECAAwF

Ak53QS4FAwASdQAACgkQlxC4m8pXrXwJ8Qf/be/UO9mqfoc2sMyhwMpN4/fdBWwf

LkA12FXQDOQMvwH9HsmEjnfUgYKXschZRi+DuHXe1P7l8G2aQLubhBsQf9ejKvRF

TzuWMQkdIq+6Koulxv6ofkCcv3d1xtO2W7nb5yxcpVBPrRfGFGebJvZa58DymCNg

yGtAU6AOz4veavNmI2+GIDQsY66+tYDvZ+CxwzdYu+HDV9HmrJfc6deM0mnBn7SR

jqzxJPgoTQhihTav6q/R5/2p5NvQ/H84OgS6GjosfGc2duUDzCP/kheMRKfzuyKC

OHQPtJuIj8++gfpHtEU7IDUX1So3c9n0PdpeBvclsDbpRnCNxQWU4mBot4kBIgQQ

AQIADAUCToi2GQUDABJ1AAAKCRCXELibyletfLZAB/9oRqx+NC98UQD/wlxCRytz

vi/MuPnbgQUPLHEap10tvEi33S/H/xDR/tcGofY4cjAvo5skZXXeWq93Av7PACUb

zkg0X0eSr2oL6wy66xfov72AwSuX+iUK68qtKaLqRLitM02y8aNRV/ggKvt7UMvG

mOvs5yLaYlobyvGaFC2ClfkNOt2MlVnQZCmnYBCwOktPGkExiu2yZMifcYGxQcpH

KVFG59KeF2cM2d4xYM8HJqkSGGW306LFVSyeRwG+wbttgLpD5bM/T2b3fF/J35ra

CSMLZearRTq8aygPl+XM7MM2eR946aw6jmOsgNBErbvvIdQj6LudAZj+8imcXV2K

iQEiBBABAgAMBQJOmdnRBQMAEnUAAAoJEJcQuJvKV618AvIIAIEF1ZJ+Ry7WOdKF

5oeQ/ynaYUigzN92fW/9zB8yuQlngkFJGidYMbci1tR1siziIVJFusR3ZonqAPGK

/SUta9Y6KWLhmc7c5UnEHklq/NfdMZ2WVSIykXlctqw0sbb+z1ecEd4G8u9j5ill

MO1B36rQayYAPoeXLX8dY4VyFLVGaQ00rWQBYFZrpw16ATWbWGJP332NSfCk4zZq

6kXEW07q0st3YBgAAGdNQyEeZCa4d4pBRSX6189Kjg6GDnIcaiOF6HO6PLr9fRlL

r5ObCgU+G9gEhfiVwDEV9E+7/Bq2pYZ9whhkBqWQzdpXTNTM24uaEhE01EPO5zeC

O214q6mJASIEEAECAAwFAk6rpgEFAwASdQAACgkQlxC4m8pXrXzAhwf/f9O99z16

3Y5FZVIxexyqXQ/Mct9uKHuXEVnRFYbA49dQLD4S73N+zN7gn9jFeQcBo4w8qVUV

94U/ta/VbLkdtNREyplPM4XY8YE5Wfd9bfyg3q1PbEiVjk995sBF+2+To99YYKst

gXPqjlH0jUfEyDmexOj+hsp8Rc63kvkIx36VBa4ONRYFefGAhKDMigL2YAhc1UkG

tkGTuLmlCGwIV6lviDZD3RJf5375VFnaHv7eXfwQxCwE+BxG3CURrjfxjaxMTmMP

yAG2rhDp5oTUEvqDYNbko5UxYOmrSjvF4FzXwqerElXJUkUzSh0pp7RxHB/1lCxD

s7D1F1hlgFQuNIkBIgQQAQIADAUCTrzZHAUDABJ1AAAKCRCXELibyletfMUpB/4s

07dREULIBnA1D6qr3fHsQJNZqbAuyDlvgGGLWzoyEDs+1JMFFlaa+EeLIo1386GU

2DammDC23p3IB79uQhJeD2Z1TcVg4cA64SfF/CHca5coeRSrdAiudzU/cgLGtXIP

/OaFamXgdMxAhloLFbSHPCZkyb00phVa8+xeIVDrK1HByZsNIXy/SSK8U26S2PVZ

2o14fWvKbJ1Aga8N6DuWY/D8P2mi3RAbiuZgfzkmKL5idH/wSKfnFKdTgJzssdCc

1jZEGVk5rFYcWOrJARHeP/tsnb/UxKBEsNtO7e3N2e/rLVnEykVIO066hz7xZK/V

NBSpx3k3qj4XPK41IHy2iQEiBBABAgAMBQJOzqO8BQMAEnUAAAoJEJcQuJvKV618

2twH/0IzjXLxN45nvIfEjC75a+i9ZSLlqR8lsHL4GpEScFKI0a0lT4IVAIY2RKG+

MAs2eHm0UfKuwGs5jluRZ9RqKrc61sY0XQV9/7znY9Db16ghX04JjknOKs/fPi87

rvKkB/QxJWS8qbb/erRmW+cPNjbRxTFPS5JIwFWHA16ieFEpvdAgKV6nfvJVTq1r

jPDcnIA9CJN2SmUFx9Qx3SRc6ITbam1hjFnY6sCh6AUhxLI2f1mq1xH9PqEy42Um

68prRqTyJ7Iox1g/UDDkeeUcAg7T1viTz7uXpS3Wrq4zzo4yOpaJfLDR3pI5g2Zk

SNGTMo6aySE4OABt8i1Pc1Pm6AmJASIEEAECAAwFAk7yPFYFAwASdQAACgkQlxC4

m8pXrXzXiAf9FrXe0lgcPM+tYOWMLhv5gXJi2VUBaLxpyRXm/kJcmxInKq1GCd3y

D4/FLHNu3ZcCz/uklPAbZXWI0O6ewq0LWsRtklmJjWiedH+hGyaTv95VklojRIBd

8nBaJ6M98rljMBHTFwWvjQFVf4FLRJQZqHlvjcCkq2Dd9BWJpGXvr/gpKkmMJYNK

/ftfZRcChb35NI19WRpOhj9u808OPcqKVvZBcPwFGV5cEBzmAC94J7JcD8+S8Ik8

iUJMQGGL3QcmZOBozovh86hj7KTSEBHlLXl832z89H1hLeuLbnXoGLv3zeUFSxkv

1h35LhZLqIMDQRXLuUzxGHMBpLhPyGWRJ4kBIgQQAQIADAUCTwQJFwUDABJ1AAAK

CRCXELibyletfABvB/9Cy69cjOqLGywITs3Cpg//40jmdhSAVxilJivP6J5bubFH

DJlVTx541Dv5h4hTG2BQuueQ4q1VCpSGW+rHcdhPyvmZGRz1rxdQQGh1Dv0Bod2c

3PJVSYPSrRSwCZJkJHOtVRBdjK4mkZb5aFTza+Tor9kxzj4FcXVd4KAS+hHQHYHc

Ar8tt2eOLzqdEFTULeGiSoNn+PVzvzdfhndphK+8F2jfQ2UKuc01O7k0Yn9xZVx0

OG6fE1gStzLv7C5amWLRd8+xh+MN0G8MgNglpBoExsEMMlPBYSUHa6lxpdMNMuib

rIyVncE9X8QOhImt8K0sNn/EdbuldJNGYbDLt7O4iQEiBBABAgAMBQJPFdTcBQMA

EnUAAAoJEJcQuJvKV6184owH+wZ/uLpezXnSxigeH1sig72QEXMrNd5DVHCJdig3

bo+K5YmmN710/m5z+63XKUEWpd6/knajObgckThzWftNeK1SSFQGPmoYZP9EZnSU

7L+/dSUpExbj842G5LYagrCyMGtlxRywWEmbi72TKS/JOK0jLiOdvVy+PHrZSu0D

TVQ7cJh1BmPsbz7zzxjmcI5l+7B7K7RHZHq45nDLoIabwDacj7BXvBK0Ajqz4QyJ

GQUjXC7q+88I+ptPvOXlE5nI/NbiCJOMI6d/bWN1KwYrC80fZuFaznfQFcPyUaDw

yRaun+K3kEji2wXecq+yMmLUEp01TKsUeOL50HD6hHH07W+JASIEEAECAAwFAk85

bQsFAwASdQAACgkQlxC4m8pXrXwKPQgAlkbUsTr7nkq+haOk0jKpaHWEbRMEGMrB

I3F7E+RDO6V/8y4Jtn04EYDc8GgZMBah+mOgeINq3y8jRMYV5jVtZXv2MWYFUcjM

kVBKeqhi/pGEjmUdmdt3DlPv3Z+fMTMRmAocI981iY/go8PVPg/+nrR6cFK2xxnO

R8TacikJBFeSfkkORg1tDzjjYv1B5ZIEkpplepl5ahJBBq7cpYhTdY6Yk0Sz0J8w

EdffLSaNxrRuWLrRhWzZU7p9bFzfb/7OHc21dJnB7wKv5VvtgE+jiQw9tOKaf5hc

SgRYuF6heu+B25gc5Uu88lo409mZ7oxQ6hDCn7JHvzh0rhmSN+Kid4kBIgQQAQIA

DAUCT0qQrQUDABJ1AAAKCRCXELibyletfC9UB/4o2ggJYM0CLxEpP0GU8UKOh3+/

zm1DN7Qe4kY2iCtF1plKHQaTgt5FlgRCFaiXcVv7WzGz/FnmxonR1leLl+kfRlwy

PPnoI/AWPCy/NO4Cl5KnjsSmsdDUpObwZ4KYsdilZR7ViJu2swdAIgnXBUwrlRJR

7CK4TAKrTeonRgVSrVx8Vt//8/cYj73CLq8oY/KK0iHiQrSwo44uyhdiFIAssjyX

n6/2E+w0zgvPexNSNNROHQ8pjbq+NTY6GwKIGsaej3UTRwQ7psvKXz8y7xdzmOAr

/khGvxB5gjkx02pimjeia8v66aH6rbnojJMAovNUS4EHdHnulv4rovC8Kf9iiQEi

BBABAgAMBQJPVdsaBQMAEnUAAAoJEJcQuJvKV618vVEIALFXPBzcAO1SnQarBLzy

YMVZZumPvSXKnUHAO+6kjApXPJ+qFRdUaSNshZxVKY9Zryblu4ol/fLUTt0CliSD

IxD6L4GXEm4VYYCl4lPO3bVsJnGITLFwQGHM27EmjVoTiD8Ch7kPq2EXr3dMRgzj

pdz+6aHGSUfOdLTPXufDvW83bEWGaRVuTJKw+wIrcuRqQ+ucWJgJGwcE4zeHjZad

Jx1XUm1X+BbI73uiQussyjhhQVVNU7QEdrjyuscaZ/H38wjUwNbylxDPB4I8quC1

knQ0wSHr7gKpM+E9nhiS14poRqU18u78/sJ2MUPXnQA6533IC238/LP8JgqB+BiQ

BTSJASIEEAECAAwFAk9ng3cFAwASdQAACgkQlxC4m8pXrXxQRAf/UZlkkpFJj1om

9hIRz7gS+l7YvTaKSzpo+TBcx3C7aqKJpir6TlMK9cb9HGTHo2Xp1N3FtQL72NvO

6CcJpBURbvSyb4i0hrm/YcbUC4Y3eajWhkRS3iVfGNFbc/rHthViz0r6Y5lhXX16

aVkDv5CIFWaF3BiUK0FnHrZiy4FPacUXCwEjv3uf8MpxV5oEmo8Vs1h4TL3obyUz

qrImFrEMYE/12lkE8iR5KWCaF8eFyl56HL3PPl90JMQBXzhwsFoWCPuwjfM5w6sW

Ll//zynwxtlJ9CRz9c2vK6aJ8DRu3OfBKN1iiEcNEynksDnNXErn5xXKz3p5pYdq

e9BLzUQCDYkBIgQQAQIADAUCT3inRgUDABJ1AAAKCRCXELibyletfGMKCADJ97qk

geBntQ+tZtKSFyXznAugYQmbzJld8U6eGSQnQkM40Vd62UZLdA8MjlWKS8y4A4L2

0cI14zs5tKG9Q72BxQOw5xkxlLASw1/8WeYEbw7ZA+sPG//q9v3kIkru3sv64mMA

enZtxsykexRGyCumxLjzlAcL1drWJGUYE2Kl6uzQS7jb+3PNBloQvz6nb3YRZ+Cg

Ly9D41SIK+fpnV8r4iqhu7r4LmAQ7Q1DF9aoGaYvn2+xLGyWHxJAUet4xkMNOLp6

k9RF1nbNe4I/sqeCB25CZhCTEvHdjSGTD2yJR5jfoWkwO9w8DZG1Q9WrWqki4hSB

l0cmcvO34pC1SJYziQEiBBABAgAMBQJPinQFBQMAEnUAAAoJEJcQuJvKV618CFEI

AJp5BbcV7+JBMRSvkoUcAWDoJSP2ug9zGw5FB8J90PDefKWCKs5Tjayf2TvM5ntq

5DE9SGaXbloIwa74FoZlgqlhMZ4AtY9Br+oyPJ5S844wpAmWMFc6NnEPFaHQkQ+b

dJYpRVNd9lzagJP261P3S+S9T2UeHVdOJBgWIq9Mbs4lnZzWsnZfQ4Lsz0aPqe48

tkU8hw+nflby994qIwNOlk/u+I/lJbNz5zDY91oscXTRl2jV1qBgKYwwCXxyB3j9

fyVpRl+7QnqbTWcCICVFL+uuYpP0HjdoKNqhzEguAUQQLOB9msPTXfa2hG+32ZYg

5pzI5V7GCHq0KO6u5Ctj3TGJASIEEAECAAwFAk+cQEEFAwASdQAACgkQlxC4m8pX

rXzi7AgAx8wJzNdD7UlgdKmrAK//YqH7arSssb33Xf45sVHDpUVA454DXeBrZpi+

zEuo03o5BhAuf38cwfbkV6jN1mC2N0FZfpy4v7RxHKLYr7tr6r+DRn1L1giX5ybx

CgY0fLAxkwscWUKGKABWxkz9b/beEXaO2rMt+7DBUdpAOP5FNRQ8WLRWBcMGQiaT

S4YcNDAiNkrSP8CMLQP+04hQjahxwCgBnksylciqz3Y5/MreybNnTOrdjVDsF0Oe

t0uLOiWXUZV1FfaGIdb/oBQLg+e1B74p5+q3aF8YI97qAZpPa1qiQzWIDX8LX9QX

EFyZ3mvqzGrxkFoocXleNPgWT8fRuokBIgQQAQIADAUCT64N/QUDABJ1AAAKCRCX

ELibyletfDOGCACKfcjQlSxrWlEUrYYZpoBP7DE+YdlIGumt5l6vBmxmt/5OEhqr

+dWwuoiyC5tm9CvJbuZup8anWfFzTTJmPRPsmE4z7Ek+3CNMVM2wIynsLOt1pRFK

4/5RNjRLbwI6EtoCQfpLcZJ//SB56sK4DoFKH28Ok4cplESPnoMqA3QafdSEA/FL

qvZV/iPgtTz7vjQkMgrXAIUM4fvKe3iXkAExGXtmgdXHVFoKmHrxJ2DTSvM7/19z

jGJeu2MhIKHyqEmCk6hLjxyCE5pAH59KlbAQOP1bS28xlRskBApm2wN+LOZWzC62

HhEReQ50inCGuuubK0PqUQnyYc+lUFxrFpcliQEiBBABAgAMBQJPv9lVBQMAEnUA

AAoJEJcQuJvKV618AzgH/iRFFCi4qjvoqji1fi7yNPZVOMMO2H13Ks+AfcjRtHuV

aa30u50ND7TH+XQe6yerTapLh3aAm/sNP99aTxIuwRSlyKEoDs93+XVSgRqPBgbF

/vxv0ykok3p6L9DxFO/w5cL8JrBhMZoJrEkIBFkwN8tWlcXPRFQvcdBYv3M3DTZU

qY+UHnOxHvSzsl+LJ0S9Xcd9C5bvYfabmYJvG5eRS3pj1L/y3a6yw6hvY+JtnQAk

t05TdeHMIgQH/zb8V9wxDzmE0un8LyoC2Jx5TpikQsJSejwK6b3coxVBlngku6+C

qDAimObZLw6H9xYYIK0FoJs7j5bQZEwUO7OLBgjcMOqJASIEEAECAAwFAk/Rpc8F

AwASdQAACgkQlxC4m8pXrXw49Qf/TdNbun2htQ+cRWarszOx8BLEiW/x6PVyUQpZ

nV/0qvhKzlJUjM9hQPcA0AsOjhqtCN6Cy8KXbK/TvPm9D/Nk6HWwD1PomzrJVFk2

ywGFIuTR+lluKSp7mzm5ym0wJs5cPq731Im31RUQU8ndjLrq9YOf5FVL8NqmcOAU

4E8d68BbmVCQC5MMr0901FKwKznShfpy7VYN25/BASj8dhnynBYQErqToOJB6Cnd

JhdTlbfR4SirqAYZZg3XeqGhByytEHE1x7FMWWFYhdNtsnAVhYBbWqAzBs8lF9Jd

Mhaf0VQU/4z10gVrRtXLR/ixrCi+P4cM/fOQkqd6pwqWkaXt6okBIgQQAQIADAUC

T+NxIAUDABJ1AAAKCRCXELibyletfFBBCAC6+0TUJDcNaqOxOG1KViY6KYg9NCL8

pwNK+RKNK/N1V+WGJQH7qDMwRoOn3yogrHax4xIeOWiILrvHK0O6drS1DjsymIhR

Sm2XbE/8pYmEbuJ9vHh3b/FTChmSAO7dDjSKdWD3dvaY8lSsuDDqPdTX8FzOfrXC

M22C/YPg7oUG2A5svE1b+yismP4KmVNWAepEuPZcnEMPFgop3haHg9X2+mj/btDB

Yr6p9kAgIY17nigtNTNjtI0dMLu43aIzedCYHqOlNHiB049jkJs54fMGBjF9qPtc

m0k44xyKd1/JXWMdNUmtwKsChAXJS3YOciMgIx6tqYUTndrP4I6q1rfriQEiBBAB

AgAMBQJP9T1VBQMAEnUAAAoJEJcQuJvKV618J9wIAI1lId9SMbEHF6PKXRe154lE

pap5imMU/lGTj+9ZcXmlf8o2PoMMmb3/E1k+EZUaeSBoOmjS8C2gwd5XFwRrlwAD

RlK/pG5XsL4h5wmN2fj1ororrJXvqH427PLRQK9yzdwG4+9HTBOxjoS8qZT9plyK

AJZzAydAMqyseRHgNo0vMwlgrs4ojo+GcFGQHrF3IaUjvVfUPOmIj7afopFdIZmI

GaSF0TXBzqcZ1chFv/eTBcIuIKRvlaDee5FgV7+nLH2nKOARCLvV/+8uDi2zbr83

Ip5x2tD3XuUZ0ZWxD0AQWcrLdmGb4lkxbGxvCtsaJHaLXWQ2m760RjIUcwVMEBKJ

ASIEEAECAAwFAlAGYWsFAwASdQAACgkQlxC4m8pXrXwyVAgAvuvEl6yuGkniWOlv

uHEusUv/+2GCBg6qV+IEpVtbTCCgiFjYR5GasSp1gpZ5r4BocOlbGdjdJGHTpyK8

xD1i+6qZWUYhNRg2POXUVzcNEl2hhouwPLOifcmTwAKU76TEv3L5STviL3hWgUR2

yEUZ3Ut0IGVV6uPER9jpR3qd6O3PeuFkwf+NaGTye4jioLAy3aYwtZCUXzvYmNLP

90K4y+5yauZteLmNeq26miKC/NQu4snNFClPbGRjHD1ex9KDiAMttOgN4WEq7srT

rYgtT531WY4deHpNgoPlHPuAfC0H+S6YWuMbgfcb6dV+Rrd8Ij6zM3B/PcjmsYUf

OPdPtIkBIgQQAQIADAUCUBgtfQUDABJ1AAAKCRCXELibyletfAm3CACQlw21Lfeg

d8RmIITsfnFG/sfM3MvZcjVfEAtsY3fTK9NiyU0B3yX0PU3ei37qEW+50BzqiStf

5VhNvLfbZR+yPou7o2MAP31mq3Uc6grpTV64BRIkCmRWg40WMjNI1hv7AN/0atgj

ATYQXgnEw7mfFb0XZtMTD6cmrz/A9nTPVgZDxzopOMgCCC1ZK4Vpq9FKdCYUaHpX

3sqnDf+gpVIHkTCMgWLYQOeX5Nl+fgnq6JppaQ3ySZRUDr+uFUs0uvDRvI/cn+ur

ri92wdDnczjFumKvz/cLJAg5TG2Jv1Jx3wecALsVqQ3gL7f7vr1OMaqhI5FEBqdN

29L9cZe/ZmkriQEiBBIBCgAMBQJVoNxyBYMHhh+AAAoJEEoz7NUmyPxLD1EH/2eh

7a4+8A1lPLy2L9xcNt2bifLfFP2pEjcG6ulBoMKpHvuTCgtX6ZPdHpM7uUOje/F1

CCN0IPB533U1NIoWIKndwNUJjughtoRM+caMUdYyc4kQm29Se6hMPDfyswXE5Bwe

PmoOm4xWPVOH/cVN04zyLuxdlQZNQF/nJg6PMsz4w5z+K6NGGm24NEPcc72iv+6R

Uc/ry/7v5cVu4hO5+r104mmNV5yLecQF13cHy2JlngIHXPSlxTZbeJX7qqxE7TQh

5nviSPgdk89oB5jFSx4g1efXiwtLlP7lbDlxHduomyQuH9yqmPZMbkJt9uZDc8Zz

MYsDDwlc7BIe5bGKfjqJAhwEEAECAAYFAlSanFIACgkQdzHqU52lcqLdvg//cAEP

qdN5VTKWEoDFjDS4I6t8+0KzdDWDacVFwKJ8RAo1M2SklDxnIvnzysZd2VHp5Pq7

i4LYCZo5lDkertQ6LwaQxc4X6myKY4LTA652ObFqsSfgh9kW+aJBBAyeahPQ8CDD

+Yl23+MY5wTsj4qt7KffNzy78vLbYnVnvRQ3/CboVix0SRzg0I3Oi7n3B0lihvXy

5goy9ikjzZevejMEfjfeRCgoryy9j5RvHH9PF3fJVtUtHCS4f+kxLmbQJ1XqNDVD

hlFzjz8oUzz/8YXy3im5MY7Zuq4P4wWiI7rkIFMjTYSpz/evxkVlkR74qOngT2pY

VHLyJkqwh56i0aXcjMZiuu2cymUt2LB9IsaMyWBNJjXr2doRGMAfjuR5ZaittmML

yZwix9mWVk7tkwlIxmT/IW6Np0qMhDZcWYqPRpf7+MqY3ZYMK4552b8aDMjhXrnO

OwLsz+UI4bZa1r9dguIWIt2C2b5C1RQ9AsQBPwg7h5P+HhRuFAuDKK+vgV8FRuzR

JeKkFqwB4y0Nv7BzKbFKmP+V+/krRv+/Dyz9Bz/jyAQgw02u1tPupH9BGhlRyluN

yCJFTSNj7G+OLU0/l4XNph5OOC7sy+AMZcsL/gsT/TXCizRcCuApNTPDaenACpbv

g8OoIzmNWhh4LXbAUHCKmY//hEw9PvTZA1xKHgyJAhwEEgECAAYFAlJYsKQACgkQ

oirk60MpxUV2XQ//b2/uvThkkbeOegusDC4AZfjnL/V3mgk4iYy4AC9hum0R9oNl

XDR51P1TEw9mC1btHj+7m7Iq1a5ke5wIC7ENZiilr0yPqeWgL5+LC98dz/L85hqA

wIoGeOfMhrlaVbAZEj4yQTAJDA35vZHVsQmp87il0m+fZX04OBLXBzw86EoAAZ7Q

EoH4qFcT9k1T363tvNnIm3mEvkQ5WjE1R9uchJa1g7hdlNQlVkjFmPZrJK9fl4z5

6Dto89Po4Sge48jDH0pias4HATYHsxW819nz5jZzGcxLnFRRR5iITVZi9qzsHP7N

bUh3qxuWCHS9xziXpOcSZY848xXw63Y5jDJfpzupzu/KHj6CzXYJUEEqp9MluoGb

/BCCEPzdZ0ovyxFutM/BRcc6DvE6sTDF/UES21ROqfuwtJ6qJYWX+lBIgyCJvj4o

RdbzxUleePuzqCzmwrIXtoOKW0Rlj4SCeF9yCwUMBTGW5/nCLmN4dwf1KW2RP2Eg

4ERbuUy7QnwRP5UCl+0ISZJyYUISfg8fmPIdQsetUK9Cj+Q5jpB2GXwELXWnIK6h

K/6jXp+EGEXSqdIE53vAFe7LwfHiP/D5M71D2h62sdIOmUm3lm7xMOnM5tKlBiV+

4jJSUmriCT62zo710+6iLGqmUUYlEll6Ppvo8yuanXkYRCFJpSSP7VP0bBqIZgQT

EQIAJgUCTnc9dgIbIwUJEPPzpwYLCQgHAwIEFQIIAwQWAgMBAh4BAheAAAoJEIxx

jTtQcuH1Ut4AoIKjhdf70899d+7JFq3LD7zeeyI0AJ9Z+YyE1HZSnzYi73brScil

bIV6sbQ7TXlTUUwgUGFja2FnZSBzaWduaW5nIGtleSAod3d3Lm15c3FsLmNvbSkg

PGJ1aWxkQG15c3FsLmNvbT6IbwQwEQIALwUCTnc9rSgdIGJ1aWxkQG15c3FsLmNv

bSB3aWxsIHN0b3Agd29ya2luZyBzb29uAAoJEIxxjTtQcuH1tT0An3EMrSjEkUv2

9OX05JkLiVfQr0DPAJwKtL1ycnLPv15pGMvSzav8JyWN3IhlBBMRAgAdBQJHrJS0

BQkNMFioBQsHCgMEAxUDAgMWAgECF4AAEgkQjHGNO1By4fUHZUdQRwABAa6SAJ9/

PgZQSPNeQ6LvVVzCALEBJOBt7QCffgs+vWP18JutdZc7XiawgAN9vmmITAQTEQIA

DAUCPj6j0QWDCWYAuwAKCRBJUOEqsnKR8iThAJ9ZsR4o37dNGyl77nEqP6RAlJqa

YgCeNTPTEVY+VXHR/yjfyo0bVurRxT2ITAQTEQIADAUCPkKCAwWDCWIiiQAKCRC2

9c1NxrokP5aRAKCIaaegaMyiPKenmmm8xeTJSR+fKQCgrv0TqHyvCRINmi6LPucx

GKwfy7KIRgQQEQIABgUCP6zjrwAKCRCvxSNIeIN0D/aWAKDbUiEgwwAFNh2n8gGJ

Sw/8lAuISgCdHMzLAS26NDP8T2iejsfUOR5sNriIRgQQEQIABgUCP7RDdwAKCRCF

lq+rMHNOZsbDAJ0WoPV+tWILtZG3wYqg5LuHM03faQCeKuVvCmdPtro06xDzeeTX

VrZ14+GIRgQQEQIABgUCQ1uz6gAKCRCL2C5vMLlLXH90AJ0QsqhdAqTAk3SBnO2w

zuSOwiDIUwCdFExsdDtXf1cL3Q4ilo+OTdrTW2CIRgQTEQIABgUCRPEzJgAKCRD2

ScT0YJNTDApxAKCJtqT9LCHFYfWKNGGBgKjka0zi9wCcCG3MvnvBzDUqDVebudUZ

61Sont+ITAQQEQIADAUCQYHLAQWDBiLZiwAKCRAYWdAfZ3uh7EKNAJwPywk0Nz+Z

Lybw4YNQ7H1UxZycaQCePVhY4P5CHGjeYj9SX2gQCE2SNx+ITAQQEQIADAUCQYHL

NAWDBiLZWAAKCRCBwvfr4hO2kiIjAJ0VU1VQHzF7yYVeg+bh31nng9OOkwCeJI8D

9mx8neg4wspqvgXRA8+t2saITAQQEQIADAUCQYHLYgWDBiLZKgAKCRBrcOzZXcP0

cwmqAJsFjOvkY9c5eA/zyMrOZ1uPB6pd4QCdGyzgbYb/eoPu6FMvVI9PVIeNZReI

TAQQEQIADAUCQdCTJAWDBdQRaAAKCRB9JcoKwSmnwmJVAKCG9a+Q+qjCzDzDtZKx

5NzDW1+W+QCeL68seX8OoiXLQuRlifmPMrV2m9+ITAQQEQIADAUCQitbugWDBXlI

0gAKCRDmG6SJFeu5q/MTAKCTMvlCQtLKlzD0sYdwVLHXJrRUvgCffmdeS6aDpwIn

U0/yvYjg1xlYiuqITAQSEQIADAUCQCpZOgWDB3pLUgAKCRA8oR80lPr4YSZcAJwP

4DncDk4YzvDvnRbXW6SriJn1yQCdEy+d0CqfdhM7HGUs+PZQ9mJKBKqITAQSEQIA

DAUCQD36ugWDB2ap0gAKCRDy11xj45xlnLLfAKC0NzCVqrbTDRw25cUss14RRoUV

PACeLpEc3zSahJUB0NNGTNlpwlTczlCITAQSEQIADAUCQQ4KhAWDBpaaCAAKCRA5

yiv0PWqKX/zdAJ4hNn3AijtcAyMLrLhlZQvib551mwCgw6FEhGLjZ+as0W681luc

wZ6PzW+ITAQSEQIADAUCQoClNAWDBSP/WAAKCRAEDcCFfIOfqOMkAJwPUDhS1eTz

gnXclDKgf353LbjvXgCeLCWyyj/2d0gIk6SqzaPl2UcWrqiITAQTEQIADAUCPk1N

hAWDCVdXCAAKCRAtu3a/rdTJMwUMAKCVPkbk1Up/kyPrlsVKU/Nv3bOTZACfW5za

HX38jDCuxsjIr/084n4kw/uITAQTEQIADAUCQdeAdgWDBc0kFgAKCRBm79vIzYL9

Pj+8AJ9d7rvGJIcHzTCSYVnaStv6jP+AEACeNHa5yltqieRBCCcLcacGqYK81omI

TAQTEQIADAUCQhiBDgWDBYwjfgAKCRB2wQMcojFuoaDuAJ9CLYdysef7IsW42UfW

hI6HjxkzSgCfeEpXS4hEmmGicdpRiJQ/W21aB0GIZQQTEQIAHQULBwoDBAMVAwID

FgIBAheABQJLcC/KBQkQ8/OnABIHZUdQRwABAQkQjHGNO1By4fWw2wCeJilgEarL

8eEyfDdYTyRdqE45HkoAnjFSZY8Zg/iXeErHI0r04BRukNVgiHsEMBECADsFAkJ3

NfU0HQBPb3BzLi4uIHNob3VsZCBoYXZlIGJlZW4gbG9jYWwhIEknbSAqc28qIHN0

dXBpZC4uLgAKCRA5yiv0PWqKX+9HAJ0WjTx/rqgouK4QCrOV/2IOU+jMQQCfYSC8

JgsIIeN8aiyuStTdYrk0VWCIjwQwEQIATwUCRW8Av0gdAFNob3VsZCBoYXZlIGJl

ZW4gYSBsb2NhbCBzaWduYXR1cmUsIG9yIHNvbWV0aGluZyAtIFdURiB3YXMgSSB0

aGlua2luZz8ACgkQOcor9D1qil+g+wCfcFWoo5qUl4XTE9K8tH3Q+xGWeYYAnjii

KxjtOXc0ls+BlqXxbfZ9uqBsiQIiBBABAgAMBQJBgcuFBYMGItkHAAoJEKrj5s5m

oURoqC8QAIISudocbJRhrTAROOPoMsReyp46Jdp3iL1oFDGcPfkZSBwWh8L+cJjh

dycIwwSeZ1D2h9S5Tc4EnoE0khsS6wBpuAuih5s//coRqIIiLKEdhTmNqulkCH5m

imCzc5zXWZDW0hpLr2InGsZMuh2QCwAkB4RTBM+r18cUXMLV4YHKyjIVaDhsiPP/

MKUj6rJNsUDmDq1GiJdOjySjtCFjYADlQYSD7zcd1vpqQLThnZBESvEoCqumEfOP

xemNU6xAB0CL+pUpB40pE6Un6Krr5h6yZxYZ/N5vzt0Y3B5UUMkgYDSpjbulNvaU

TFiOxEU3gJvXc1+h0BsxM7FwBZnuMA8LEA+UdQb76YcyuFBcROhmcEUTiducLu84

E2BZ2NSBdymRQKSinhvXsEWlH6Txm1gtJLynYsvPi4B4JxKbb+awnFPusL8W+gfz

jbygeKdyqzYgKj3M79R3geaY7Q75Kxl1UogiOKcbI5VZvg47OQCWeeERnejqEAdx

EQiwGA/ARhVOP/1l0LQA7jg2P1xTtrBqqC2ufDB+v+jhXaCXxstKSW1lTbv/b0d6

454UaOUV7RisN39pE2zFvJvY7bwfiwbUJVmYLm4rWJAEOJLIDtDRtt2h8JahDObm

3CWkpadjw57S5v1c/mn+xV9yTgVx5YUfC/788L1HNKXfeVDq8zbAiQIiBBMBAgAM

BQJCnwocBYMFBZpwAAoJENjCCglaJFfPIT4P/25zvPp8ixqV85igs3rRqMBtBsj+

5EoEW6DJnlGhoi26yf1nasC2frVasWG7i4JIm0U3WfLZERGDjR/nqlOCEqsP5gS3

43N7r4UpDkBsYh0WxH/ZtST5llFK3zd7XgtxvqKL98l/OSgijH2W2SJ9DGpjtO+T

iegq7igtJzw7Vax9z/LQH2xhRQKZR9yernwMSYaJ72i9SyWbK3k0+e95fGnlR5pF

zlGq320rYHgD7v9yoQ2t1klsAxK6e3b7Z+RiJG6cAU8o8F0kGxjWzF4v8D1op7S+

IoRdB0Bap01ko0KLyt3+g4/33/2UxsW50BtfqcvYNJvU4bZns1YSqAgDOOanBhg8

Ip5XPlDxH6J/3997n5JNj/nk5ojfd8nYfe/5TjflWNiput6tZ7frEki1wl6pTNbv

V9C1eLUJMSXfDZyHtUXmiP9DKNpsucCUeBKWRKLqnsHLkLYydsIeUJ8+ciKc+EWh

FxEY+Ml72cXAaz5BuW9L8KHNzZZfez/ZJabiARQpFfjOwAnmhzJ9r++TEKRLEr96

taUI9/8nVPvT6LnBpcM38Td6dJ639YvuH3ilAqmPPw50YvglIEe4BUYD5r52Seqc

8XQowouGOuBX4vs7zgWFuYA/s9ebfGaIw+uJd/56Xl9ll6q5CghqB/yt1EceFEnF

CAjQc2SeRo6qzx22iEYEEBECAAYFAkSAbycACgkQCywYeUxD5vWDcACfQsVk/XGi

ITFyFVQ3IR/3Wt7zqBMAoNhso/cX8VUfs2BzxPvvGS3y+5Q9iEYEEBECAAYFAkUw

ntcACgkQOI4l6LNBlYkyFgCbBcw5gIii0RTDJsdNiuJDcu/NPqEAniSq9iTaLjgF

HZbaizUU8arsVCB5iEYEEBECAAYFAkWho2sACgkQu9u2hBuwKr6bjwCfa7ZK6O+X

mT08Sysg4DEoZnK4L9UAoLWgHuYg35wbZYx+ZUTh98diGU/miF0EExECAB0FAj4+

owwFCQlmAYAFCwcKAwQDFQMCAxYCAQIXgAAKCRCMcY07UHLh9XGOAJ4pVME15/DG

rUDohtGv2z8a7yv4AgCeKIp0jWUWE525QocBWms7ezxd6syIXQQTEQIAHQUCR6yU

zwUJDTBYqAULBwoDBAMVAwIDFgIBAheAAAoJEIxxjTtQcuH1dCoAoLC6RtsD9K3N

7NOxcp3PYOzH2oqzAKCFHn0jSqxk7E8by3sh+Ay8yVv0BYhdBBMRAgAdBQsHCgME

AxUDAgMWAgECF4AFAkequSEFCQ0ufRUACgkQjHGNO1By4fUdtwCfRNcueXikBMy7

tE2BbfwEyTLBTFAAnifQGbkmcARVS7nqauGhe1ED/vdgiF0EExECAB0FCwcKAwQD

FQMCAxYCAQIXgAUCS3AuZQUJEPPyWQAKCRCMcY07UHLh9aA+AKCHDkOBKBrGb8tO

g9BIub3LFhMvHQCeIOOot1hHHUlsTIXAUrD8+ubIeZaJARwEEgECAAYFAkvCIgMA

CgkQ3PTrHsNvDi8eQgf/dSx0R9Klozz8iK79w00NOsdoJY0Na0NTFmTbqHg30XJo

G62cXYgc3+TJnd+pYhYi5gyBixF/L8k/kPVPzX9W0YfwChZDsfTw0iDVmGxOswiN

jzSo0lhWq86/nEL30Khl9AhCC1XFNRw8WZYq9Z1qUXHHJ2rDARaedvpKHOjzRY0N

dx6R2zNyHDx2mlfCQ9wDchWEuJdAv0uHrQ0HV9+xq7lW/Q3L/V5AuU0tiowyAbBL

PPYrB6x9vt2ZcXS7BOy8SfQ1i8W2QDQ/Toork4YwBiv6WCW/ociy7paAoPOWV/Nf

2S6hDispeecbk7wqpbUj5klDmwrlgB/jmoAXWEnbsYkBIgQQAQIADAUCSSpooAUD

ABJ1AAAKCRCXELibyletfFOMCACpP+OVZ7lH/cNY+373c4FnSI0/S5PXS0ABgdd4

BFWRFWKrWBeXBGc8sZfHOzVEwkzV96iyHbpddeAOAkEA4OVPW1MMFCmlHxi2s9/N

JrSrTPVfQOH5fR9hn7Hbpq/ETw0IoX1FKo7vndMnHZnFEnI+PDXLcdMYQgljYzhT

xER4vYY0UKu8ekSshUy4zOX7XSJxwqPUvps8qs/TvojIF+vDJvgFYHVkgvS+shp8

Oh/exg9vKETBlgU87Jgsqn/SN2LrR/Jhl0aLd0G0iQ+/wHmVYdQUMFaCZwk/BKNa

XPzmGZEUZ3RNbYa19Mo7hcE3js76nh5YMxFvxbTggVu4kdFkiQEiBBABAgAMBQJK

M06IBQMAEnUAAAoJEJcQuJvKV618F4gH/innejIHffGMk8jYix4ZZT7pW6ApyoI+

N9Iy85H4L+8rVQrtcTHyq0VkcN3wPSwtfZszUF/0qP6P8sLJNJ1BtrHxLORYjJPm

gveeyHPzA2oJl6imqWUTiW822fyjY/azwhvZFzxmvbFJ+r5N/Z57+Ia4t9LTSqTN

HzMUYaXKDaAqzZeK7P0E6XUaaeygbjWjBLQ1O0ezozAy+Kk/gXApmDCGFuHSFe7Z

mgtFcbXLM2XFQpMUooETD2R8MUsd+xnQsff/k6pQOLxi+jUEsWSr/iqmvlk6gZ4D

pemBjuhcXYlxJYjUaX9Zmn5s+ofF4GFxRqXoY7l9Z+tCM9AX37lm6S+JASIEEAEC

AAwFAkpEcgoFAwASdQAACgkQlxC4m8pXrXz2mgf/RQkpmMM+5r8znx2TpRAGHi5w

ktvdFxlvPaOBWE28NDwTrpcoMqo9kzAiuvEQjVNihbP21wR3kvnQ84rTAH0mlC2I

uyybggpqwzOUl+Wi0o+vk8ZA0A0dStWRN8uqneCsd1XnqDe1rvqC4/9yY223tLmA

kPvz54ka2vX9GdJ3kxMWewhrVQSLCktQpygU0dujGTDqJtnk0WcBhVF9T87lv3W2

eGdPielzHU5trXezmGFj21d56G5ZFK8co7RrTt4qdznt80glh1BTGmhLlzjMPLTe

dcMusm3D1QB9ITogcG94ghSf9tEKmmRJ6OnnWM5Kn9KcL63E5oj2/lY9H54wSYkB

IgQQAQIADAUCSlY+RwUDABJ1AAAKCRCXELibyletfOOQB/0dyJBiBjgf+8d3yNID

pDktLhZYw8crIjPBVdOgX12xaUYBTGcQITRVHSggzffDA5BQXeUuWhpL4QB0uz1c

EPPwSMiWiXlBtwF5q6RVf3PZGJ9fmFuTkPRO7SruZeVDo9WP8HjbQtOLukYf566e

grzAYR9p74UgWftpDtmrqrRTobiuvsFBxosbeRCvEQCrN0n+p5D9hCVB88tUPHnO

WA4mlduAFZDxQWTApKQ92frHiBqy+M1JFezz2OM3fYN+Dqo/Cb7ZwOAA/2dbwS7o

y4sXEHbfWonjskgPQwFYB23tsFUuM4uZwVEbJg+bveglDsDStbDlfgArXSL/0+ak

lFcHiQEiBBABAgAMBQJKaAqEBQMAEnUAAAoJEJcQuJvKV618rH0H/iCciD4U6YZN

JBj0GN7/Xt851t9FWocmcaC+qtuXnkFhplXkxZVOCU4VBMs4GBoqfIvagbBTyfV4

Di+W8Uxr+/1jiu3l/HvoFxwdwNkGG6zNBhWSjdwQpGwPvh5ryV1OfLX/mgQgdDmx

vqz5+kFDUj4m7uLaeuU2j1T0lR4zU0yAsbt7J3hwfqJCXHOc9bm5nvJwMrSm+sdC

TP5HjUlwHr9mTe8xuZvj6sO/w0P4AqIMxjC9W7pT9q0ofG2KSTwt7wFbh05sbG4U

QYOJe4+Soh3+KjAa1c0cvmIh4cKX9qfCWwhhdeNfh1A9VTHhnl5zTv/UjvnQtjhl

H/Fq1eBSKcSJASIEEAECAAwFAkp5LgoFAwASdQAACgkQlxC4m8pXrXwY6wgAg3f8

76L3qDZTYlFAWs3pXBl8GsUr1DEkTlEDZMZKDM3wPmhaWBR1hMA3y6p3aaCUyJIJ

BEneXzgyU9uqCxXpC78d5qc3xs/Jd/SswzNYuvuzLYOw5wN5L31SLmQTQ8KqE0uo

RynBmtDCQ4M2UKifSnv+0+3mPh85LVAS481GNpL+VVfCYtKesWNu40+98Yg6L9NG

WwRTfsQbcdokZo44Jz7Y7f81ObC4r/X1DgPj2+d4AU/plzDcdrbINOyprs+7340e

cnaGO4Lsgd19b1CvcgJgltRquu3kRvd+Ero2RYpDv6GVK8Ea0Lto4+b/Ae8cLXAh

QnaWQCEWmw+AU4Jbz4kBIgQQAQIADAUCSo5fvQUDABJ1AAAKCRCXELibyletfA08

B/9w8yJdc8K+k07U30wR/RUg3Yb2lBDygmy091mVsyB0RGixBDXEPOXBqGKAXiV1

QSMAXM2VKRsuKahY2HFkPbyhZtjbdTa7Pr/bSnPvRhAh9GNWvvRg2Kp3qXDdjv9x

ywEghKVxcEIVXtNRvpbqRoKmHzIExvUQck5DM1VwfREeYIoxgs4035WADhVMdngQ

S2Gt8P2WaU/p8EZhFGg6X8KtOlD68zGboaJe0hj2VDc+Jc+KdjRfE3fW5IToid/o

DkUaIW6tB3WkXb0g6D/2hrEJbX3headChHKSB8eQdOR9bcCJDhhU8csd501qmrhC

ctmvlpeWQZdIQdk6sABPWeeCiQEiBBABAgAMBQJKoBJHBQMAEnUAAAoJEJcQuJvK

V618Ml8H/1D88/g/p9fSVor4Wu5WlMbg8zEAik3BIxQruEFWda6nART6M9E7e+P1

++UHZsWYs6l9ROpWxRLG1Yy9jLec2Y3nUtb20m65p+IVeKR2a9PHW35WZDV9dOYP

GZabKkO1clLeWLVgp9LRjZ+AeRG+ljHqsULXro1dwewLTB/gg9I2vgNv6dKxyKak

nM/GrqZLATAq2KoaE/u/6lzRFZIzZnLtjZh8X7+nS+V8v9IiY4ntrpkrbvFk30U6

WJp79oBIWwnW/84RbxutRoEwSar/TLwVRkcZyRXeJTapbnLGnQ/lDO1o1d7+Vbjd

q/Sg/cKHHf7NthCwkQNsCnHL0f51gZCJASIEEAECAAwFAkqoEAAFAwASdQAACgkQ

lxC4m8pXrXwE/Af/XD4R/A5R6Ir/nCvKwCTKJmalajssuAcLEa2pMnFZYO/8rzLO

+Gp8p0qFH9C4LFwA0NvR5q6X/swuROf4zxljSvNcdlQVaAfJ2ZDEgJ5GXzsPplrv

SAI9jS3LL7fSWDZgKuUe0a4qx7A0NgyGMUYGhP+QlRFa8vWEBI9fANd/0mMqAeBV

qQyOH0X1FiW1Ca2Jn4NKfuMy9GEvRddVIbB1LvoNVtXPNzeeKMyNb9Jdx1MFWssy

COBP2DayJKTmjvqPEc/YOjOowoN5sJ/jn4mVSTvvlTooLiReSs6GSCAjMVxN7eYS

/Oyq6Iu1JDcJvmB8N2WixAZtAVgF8OA7CWXKVYkBIgQQAQIADAUCSrnHiQUDABJ1

AAAKCRCXELibyletfPChB/9uECti1dZeNuFsd0/RuGyRUVlrrhJE6WCcOrLO9par

rPbewbKBmjSzB0MygJXGvcC06mPNuquJ7/WpxKsFmfg4vJBPlADFKtgRUy9BLzjC

eotWchPHFBVW9ftPbaQViSUu7d89NLjDDM5xrh80puDIApxoQLDoIrh3T1kpZx56

jSWv0gelFUMbXAzmqkJSyL4Xdh1aqzgUbREd7Xf2ICzuh0sV6V7c/AwWtjWEGEsA

HZaiQDywZwbC18GwrMLiAzGWb/AScFDQRCZKJDjL+Ql8YT6z+ZMVr8gb7CIU5PKY

dhiIf2UVTQwLAoW7lNRCQQAqcGjK3IMIz7SO/yk4HmVUiQEiBBABAgAMBQJK3gjG

BQMAEnUAAAoJEJcQuJvKV618jkEH+wb0Zv9z7xQgpLMowVuBFQVu8/z7P5ASumyB

PUO3+0JVxSHBhlCKQK7n11m1fhuGt2fCxXhSU6LzXj36rsKRY53lGZ9QhvqFUtQH

3Xb2IQLIJC4UKjG2jSSCdcuA/x98bwp2v7O03rn7ndCS16CwXnRV3geQoNipRKMS

DajKPpZv1RiZm8pMKqEb8WSw352xWoOcxuffjlsOEwvJ85SEGCAZ9tmIlkZOc7Ai

QONDvii9b8AYhQ60RIQC0HP2ASSmK0V92VeFPxHmAygdDQgZNVtbVxgnnt7oTNEu

VRXNY+z4OfBArp7R+cTsvijDRZY4kML1n22hUybwoxUEvjqZV2+JASIEEAECAAwF

AkrvOlQFAwASdQAACgkQlxC4m8pXrXxrPAgArXiNgZirNuBhfNCXlkzkCHLx5wnV

e4SmTpbWzTwWw7+qk7d4l9hlWtdImISORINzo7f4ShSUzJX2GciNaXhaHRo7+y5O

Zbu82jQb09aQQj/nibKYuqxqUrobTEm+DuYz3JUQZm2PsPcHLS8mX9cxvrJUncPG

nXEV0DRaq71SGWDprtkvBbp6i38aY3sIhYgz8wM5m1szKDtjywmBYcFehIdozt9z

hm7wZshzRWQX1+Rf/pIsnk+OzBIa34crSemTnacbV/B7278z2XAyziPNFuqz0xu+

iltOmYmayfNWAmumuw9NcuwWMlth6Mc2HLrpo0ZBheJ6iuDMPsHnwqdB/4kBIgQQ

AQIADAUCSwBd2gUDABJ1AAAKCRCXELibyletfP6tB/4m1w0BtlkJgtS6E+B/ns14

z4A4PGors+n+MYm05qzvi+EnDF/sytCmVcKeimrtvDcfoDtKAFFvJjcYXfnJdGWm

Pu0SJMRL5KKCirAKwZmU/saxOgoB5QLNw+DHPteJ3w9GmWlGxIqG1r15WC5duzBC

y3FsnjJYG3jaLnHOO9yXXb5h0kUTORfUKdvAr1gxF2KoatZWqGoaPPnHoqb88rjt

zk8I7gDqoXnzh8wLxa0ZYvfTC/McxdWTrwXLft+krmMQ18iIZEne2hvVLNJVuluU

oiWLeHA8iNCQ4W4WTdLc1mCnCjGTMX/MN41uLH0C9Ka4R6wEaqj4lPDk1B/1TV+Q

iQEiBBABAgAMBQJLEYGrBQMAEnUAAAoJEJcQuJvKV618naIH/2t9aH5mBTKBN6fU

qhrf79vIsjtI/QNS5qisBISZMX3/1/0Gu6WnxkPSfdCUJMWCjMcnVj7KU2wxTHHG

VpAStd9r2afUNxRyqZwzwyytktuZok0XngAEDYDDBS3ssu2R4uWLCsC2ysXEqO/5

tI5YrTWJZrfeIphTaYP5hxrMujvqy3kEwKKbiMz91cDeiLS+YCBcalj5n/1dMYf7

8U8C6ieurxAg/L8h6x25VM4Ilx4MmG2T8QGtkkUXd+Fd/KYWmf0LE5LLPknf0Hhw

oVslPXeinp4FsHK/5wzviv4YZpzuTqs9NlKcMsa4IuuPOB0FDf0pn+OFQbEg9QwY

2gCozK+JASIEEAECAAwFAksjTdQFAwASdQAACgkQlxC4m8pXrXwlogf/XBGbXRVX

LMaRN4SczOjwT3/tUCriTkb3v+zKjRG90zFhYAccjn7w+7jKQicjq6quQG1EH2X4

/Su6ps1lDLqGHHhiJW3ZhxQScLZmhdAYsh2qG4GP/UW3QjXG7c61t+H3olvWg2cr

wqCxxFZAgkAAkr9xcHWFZJEQeXoob6cCZObaUnHSANdmC6s5lUxXYa2bmL7Q3UB4

4KCzDvAfbPZKJOw9k0qb3lc11zx+vGdyZFbm4R0+3LPp/vT0b3GlSbbF9lU1GOXh

VaphrgFFa76dmjfHCkPplXAkK1VSIU/aPGAefduTFMdlSZpdMtJ5AULjGcszBDlR

pLlPxvqVa0ZpgIkBIgQQAQIADAUCSycmkgUDABJ1AAAKCRCXELibyletfHlNCACp

1YespiHfQt2alcscE5zgfETEHHic8Ai6pNkU9HT4TeWcFHEDe5QqfYcpjLrQvBXS

kSvxEittbyRdv+e+j5Z+HyHjiG8nAQBL6qy9eHqQE4+d7gYs6DTk7sG9ZMYphREb

ltzD+F4hVCQdLT8LNr0eVFN7ehqECScDaCG8/Qyti+l/0M902/Yn+mz0ilOiUdWJ

9x6LPaIINtb1gsYDEylLjwGIZmI0r5Kh9wYoV4vnNezFbxO1uRiW0B7iaPjIEsbt

OOKp7wx2aX+DM3N9F3BtaIY8XnzcnomNm83SNsgmgrZljpQltUnNqIhNM8DupQ+I

WOV5gtl6pTC7CgeVTVyRiQEiBBABAgAMBQJLOGXuBQMAEnUAAAoJEJcQuJvKV618

ll4IAKJ9mm4jb0c8fe9+uDI8eCJRbzNbVXm8zWzpA8GUtQAakwxoKv332QP1Wa1P

odni/e3EMhsSREOZJJv79YqGxGRBTE9Kb/VjM34nas4XSnXKW28XWhKyIw+XwQAi

nY2swFHh+83Htr/mwTdJfS2aEYl2zboBvd/JZCdhOGU2GH737S/3uEczoKkfVQ/w

OTM8X1xWwlYWqx23k/DsGcuDs9lA2g7Mx7DSqBtVjaTkn9h0zATzXLDkmP4SAUVj

cZ83WDpFre5WnizZjdXlBMM5OCexp5WpmzyHLTnaBFK4jEmnsk5C2Rnoyp8Ivz6g

Ecg1tRbEXijRw++d2TFYlJwLKtiJASIEEAECAAwFAktKMicFAwASdQAACgkQlxC4

m8pXrXxqHQgAuYY5scKrh0m/GS9EYnyC9494lOlO6iytU0CpE6oBC31M3hfX/Dbj

UbcS5szZNU+2CPYo4ujQLZ7suN7+tTjG6pZFfMevajT9+jsL+NPMF8RLdLOVYmbl

TmSQGNO+XGEYaKYH5oZIeIW5AKCgi2ozkdFlBBLAx7Kqo/FyybhkURFEcvEyVmgf

3KLV7IIiX/fYLfoCMCJ/Lcm9/llSFB1n8Nvg66Xd533DKoHjueD3jyaNAVlo2mq/

sIAv++kntvOiB3GDK5pfwHZ78WWiCpsWZpE5gzAnzJ1Y0WEigRo0PVLu3cLO0jLG

23d+H/CbfZ8rkajHJeCDQF7YVmP0t0nYpYkBIgQQAQIADAUCS1v+ZgUDABJ1AAAK

CRCXELibyletfNS/CACqt2TkB86mjqM+cJ74+dWBvJ2aFuURuxzm95i9Q/W/hU08

2iMbC3+0k2oD8CrTOe61P+3oRyLjv/UEDUNzLncNe2YsA9JeV+4hvPwH5Vp3Om13

089fCKZUbqslXNKkHiWYU+zAaZJXEuGRmRz0HbQIeAMOWF4oa226uo1e4ws1Jhc+

F3E/ApCRyFBqBUdL05hapQLditYpsBjIdiBGpjzidMLE2wX2W4ZpAdN0U6BIyIqR

mTPjbSkvzS9kSWFmfhQgnBDKEYJpVZgE1sN52rYC1sDeGeiuKxlzjVov9MMhYMWa

Zo3R5o3F2iIM/BK6FbC252lf/Mhu3ICuXujNBZNYiQEiBBABAgAMBQJLbSH4BQMA

EnUAAAoJEJcQuJvKV618kd0IAJLLwDH6gvgAlBFklQJXqQxUdcSOOVMAWtlHgWOy

ozjgomZZBkRL8dtCDr9YBMcj5czcQ3qpmLJdppXhKB+kJV2iUXfDMSFXwJ4wLfIs

8FNnXw8H5U01oBkGH/Ku6ngL9Vwt+MjYHtCWkw9QueUKZnDudX9qIzLAIt+mwSTu

A6+fY4VWIg40AA0v3exaQM55YR/UhlKunpGG9o8Qkq77dMEbTMpOmBoLbOMRB3Dd

MAvVU6G2l6Pcb7KobVCuOBnb6batXARV/G8sw+nzfJ16fr/KobZT2A6m+Jrqk4dl

F14ljLbz16O5JGUPAryN2G2ddBdSAy7dtFSVhWWiWC9n88q5Ag0EPj6jHRAIAO/h

iX8WzHWOMLJT54x/axeDdqn1rBDf5cWmaCWHN2ujNNlgpx5emoU9v7QStsNUCOGB

bXkeO4Ar7YG+jtSR33zqNh3y5kQ0YkY3dQ0wh6nsl+wh4XIIY/3TUZVtmdJeUBRH

JlfVNFYad2hX1guFI37Ny1PoZAFsxO82g+XB/Se8r/+sbmVcONdcdIeFKrE3FjLt

IjNQcxC6l9Q2Oy8KDxG/zvUZG3+H5i3tdRMyGgmuD6gEV0GXOHYUopzLeit1+Aa0

bCk36Mwbu+BeOw/CJW3+b0mB27hOaf9aCA855IP6fJFvtxcblq8nHIqhU3Dc9tec

sl9/S1xZ5S8ylG/xeRsAAwUH/i8KqmvAhq0X7DgCcYputwh37cuZlHOa1Ep07JRm

BCDgkdQXkGrsj2Wzw7Aw/TGdWWkmn2pxb8BRui5cfcZFO7c6vryi6FpJuLucX975

+eVY50ndWkPXkJ1HF4i+HJwRqE2zliN/RHMs4LJcwXQvvjD43EE3AO6eiVFbD+qA

AdxUFoOeLblKNBHPG7DPG9xL+Ni5rkE+TXShxsB7F0z7ZdJJZOG0JODmox7IstQT

GoaU9u41oyZTIiXPiFidJoIZCh7fdurP8pn3X+R5HUNXMr7M+ba8lSNxce/F3kmH

0L7rsKqdh9d/aVxhJINJ+inVDnrXWVoXu9GBjT8Nco1iU9SIVAQYEQIADAUCTnc9

7QUJE/sBuAASB2VHUEcAAQEJEIxxjTtQcuH1FJsAmwWK9vmwRJ/y9gTnJ8PWf0BV

roUTAKClYAhZuX2nUNwH4vlEJQHDqYa5yQ==

=ghXk

-----END PGP PUBLIC KEY BLOCK-----

**要将构建的密钥导入你的个人公共 GPG 密钥圈，使用 gpg --import。例如，如果你把钥匙保存在一个名为 mysql\_pubkey.asc 的文件中，导入命令看起来像这样。**

shell> **gpg --import mysql\_pubkey.asc**

gpg: key 5072E1F5: public key "MySQL Release Engineering

<mysql-build@oss.oracle.com>" imported

gpg: Total number processed: 1

gpg: imported: 1

gpg: no ultimately trusted keys found

**你也可以使用公钥 ID 5072E1F5 从公钥服务器下载钥匙。**

shell> gpg --recv-keys 5072E1F5

gpg: requesting key 5072E1F5 from hkp server keys.gnupg.net

gpg: key 5072E1F5: "MySQL Release Engineering <mysql-build@oss.oracle.com>"

1 new user ID

gpg: key 5072E1F5: "MySQL Release Engineering <mysql-build@oss.oracle.com>"

53 new signatures

gpg: no ultimately trusted keys found

gpg: Total number processed: 1

gpg: new user IDs: 1

gpg: new signatures: 53

**如果你想把密钥导入你的RPM配置以验证RPM安装包，你应该可以直接导入密钥。**

shell> **rpm --import mysql\_pubkey.asc**

**如果你遇到问题或需要RPM的具体信息，请参阅第2.1.4.4节，"使用RPM进行签名检查"。**

**在你下载并导入公共构建密钥后，下载你想要的MySQL包和相应的签名，这也可以从下载页面获得。签名文件的名称与分发文件相同，扩展名为.asc，如下表的例子所示。**

**表2.1 源文件的MySQL包和签名文件**

|  |  |
| --- | --- |
| **File Type** | **File Name** |
| Distribution file | **mysql-standard-8.0.26-linux-i686.tar.gz** |
| Signature file | **mysql-standard-8.0.26-linux-i686.tar.gz.asc** |

**确保两个文件都存储在同一个目录下，然后运行下面的命令来验证发行文件的签名。**

shell> **gpg --verify *package\_name*.asc**

**如果下载的软件包是有效的，你应该看到一个类似这样的 Good signature 消息。**

shell> **gpg --verify mysql-standard-8.0.26-linux-i686.tar.gz.asc**

gpg: Signature made Tue 01 Feb 2011 02:38:30 AM CST using DSA key ID 5072E1F5

gpg: Good signature from "MySQL Release Engineering <mysql-build@oss.oracle.com>"

**良好的签名信息表明，与我们网站上列出的签名相比，文件签名是有效的。但你也可能看到警告，像这样。**

shell> **gpg --verify mysql-standard-8.0.26-linux-i686.tar.gz.asc**

gpg: Signature made Wed 23 Jan 2013 02:25:45 AM PST using DSA key ID 5072E1F5

gpg: checking the trustdb

gpg: no ultimately trusted keys found

gpg: Good signature from "MySQL Release Engineering <mysql-build@oss.oracle.com>"

gpg: WARNING: This key is not certified with a trusted signature!

gpg: There is no indication that the signature belongs to the owner.

Primary key fingerprint: A4A9 4068 76FC BD3C 4567 70C8 8C71 8D3B 5072 E1F5

**这很正常，因为它们取决于你的设置和配置。下面是对这些警告的解释。**

* **gpg: 没有找到最终受信任的钥匙。这意味着特定的密钥没有被你或你的信任网络 "最终信任"，这对于验证文件签名的目的来说是可以的。**
* **警告：这把钥匙没有经过可信签名的认证! 没有迹象表明该签名是属于所有者的。这指的是你的信任程度，你认为你拥有我们真正的公钥。这是一个个人决定。理想情况下，MySQL开发人员会亲自交给你钥匙，但更常见的是你下载了它。下载的东西被篡改了吗？可能没有，但这个决定取决于你。设置一个信任网是信任他们的一种方法。**

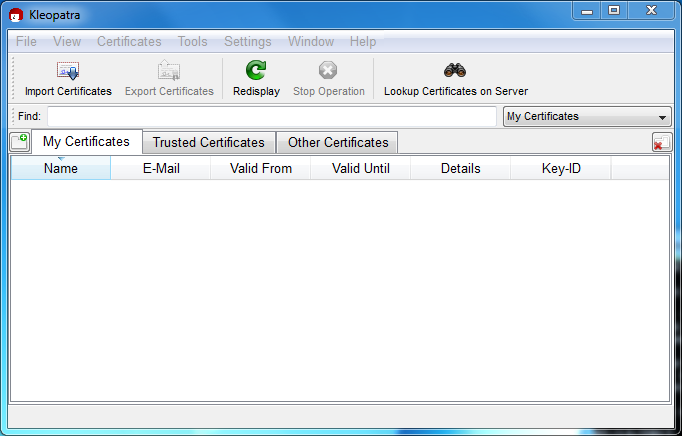
**关于如何使用公钥的更多信息，请参见GPG文档。**

#### **2.1.4.3 使用 Gpg4win for Windows 进行签名检查**

**第2.1.4.2节 "使用GnuPG进行签名检查 "描述了如何使用GPG验证MySQL下载。该指南也适用于Microsoft Windows，但另一个选择是使用Gpg4win这样的GUI工具。你可以使用一个不同的工具，但我们的例子是基于Gpg4win的，并利用其捆绑的Kleopatra GUI。**

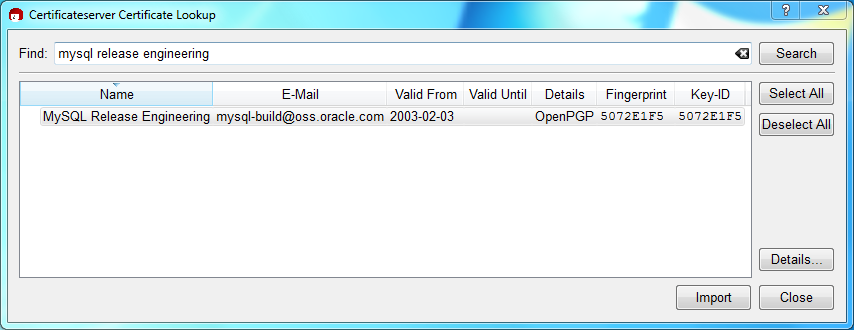
**下载并安装 Gpg4win，然后加载 Kleopatra。对话框看起来应该类似于。**

**图2.1 Kleopatra：初始屏幕**



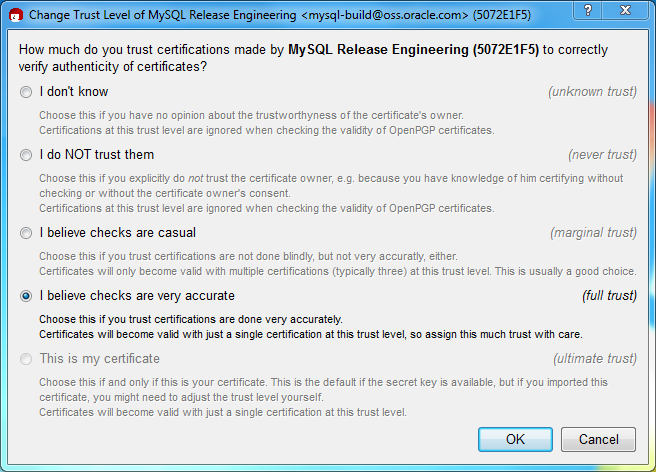
**接下来，添加MySQL发布工程证书。通过点击 "文件"、"服务器上的证书查询 "来完成此操作。在搜索框中输入 "Mysql Release Engineering"，然后按搜索。**

**图2.2 Kleopatra: 在服务器上查找证书向导。找到一个证书**



**选择 "MySQL Release Engineering "证书。指纹和密钥ID必须是 "5072E1F5"，或者选择 "详细信息... "来确认证书是否有效。现在，通过点击导入来导入它。当显示导入对话框时，选择 "好"，该证书现在应该被列在导入的证书标签下。**

**接下来，为我们的证书配置信任级别。选择我们的证书，然后从主菜单中选择证书，改变所有者信任....。我们建议为我们的证书选择 "我相信检查是非常准确的"，否则你可能无法验证我们的签名。选择 "我相信检查是非常准确的 "来启用 "完全信任"，然后按确定。**



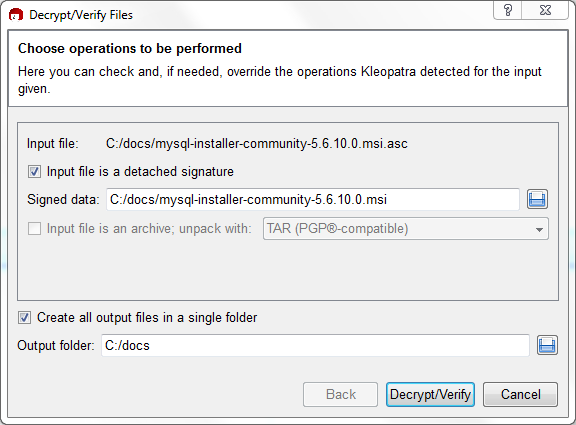
**接下来，验证下载的MySQL包文件。这需要打包的文件和签名的文件。签名文件的名称必须与打包的文件相同，但有一个附加的.asc扩展名，如下表的例子所示。签名在每个MySQL产品的下载页面上都有链接。你必须用这个签名来创建.asc文件。**

**Table 2.2 MySQL Package and Signature Files for MySQL Installer for Microsoft Windows**

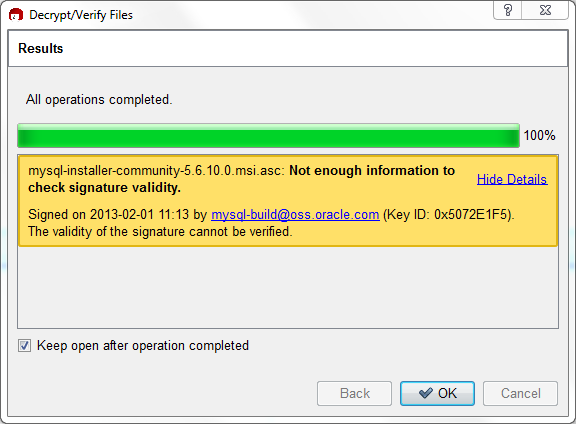
|  |  |
| --- | --- |
| **File Type** | **File Name** |
| Distribution file | **mysql-installer-community-8.0.26.msi** |
| Signature file | **mysql-installer-community-8.0.26.msi.as** |

确保两个文件都存储在同一目录下，然后运行以下命令来验证分发文件的签名。可以将签名（.asc）文件拖入Kleopatra，或者从文件，解密/验证文件...加载对话框，然后选择.msi或.asc文件。

图2.4 Kleopatra: 解密和验证文件对话框

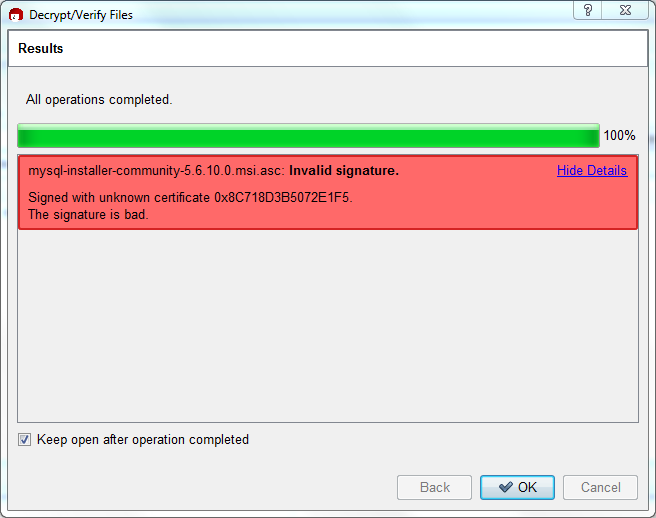


点击解密/验证来检查文件。两种最常见的结果如下图所示；虽然黄色警告看起来有问题，但下面意味着文件检查成功通过。你现在可以运行这个安装程序了



看到红色的The signature is bad错误意味着该文件是无效的。如果你看到这个错误，请不要执行MSI文件。

**Figure 2.6 Kleopatra: the Decrypt and Verify Results Dialog: Bad**



第2.1.4.2节，"使用GnuPG进行签名检查 "一节解释了为什么你没有看到绿色的良好签名结果。

#### 2.1.4.4 使用RPM进行签名检查

对于RPM包，没有单独的签名。RPM包有一个内置的GPG签名和MD5校验码。你可以通过运行以下命令来验证一个包。

shell> **rpm --checksig *package\_name*.rpm**

例子。

shell> **rpm --checksig MySQL-server-8.0.26-0.linux\_glibc2.5.i386.rpm**

MySQL-server-8.0.26-0.linux\_glibc2.5.i386.rpm: md5 gpg OK

注意

*如果你使用RPM 4.1并且它抱怨（GPG）不确定（MISSING KEYS: GPG#5072e1f5），即使你已经将MySQL的公共构建密钥导入你自己的GPG密钥圈，你需要先将该密钥导入RPM密钥圈。RPM 4.1不再使用你的个人GPG钥匙圈（或GPG本身）。相反，RPM维护一个单独的钥匙圈，因为它是一个全系统的应用程序，而一个用户的GPG公钥圈是一个用户特定的文件。要导入MySQL公钥到RPM的钥匙圈，首先要获得钥匙，然后使用rpm --import来导入钥匙。例如。*

shell> gpg --export -a 5072e1f5 > 5072e1f5.asc

shell> rpm --import 5072e1f5.asc

另外，rpm也支持直接从URL加载密钥，你可以使用这个手册页面。

shell> **rpm --import https://dev.mysql.com/doc/refman/8.0/en/checking-gpg-signature.html**

如果你需要获得MySQL公钥，请参阅第2.1.4.2节，"使用GnuPG进行签名检查"。

### 2.1.5 安装布局

不同的安装类型（例如，本地包、二进制压缩包和源压缩包）的安装布局是不同的，这可能导致在管理不同的系统或使用不同的安装源时出现混乱。各自的布局在相应的安装类型或平台章节中给出，如下所述。请注意，来自 Oracle 以外的供应商的安装布局可能与这些布局不同。

### 2.1.6 特定于编译器的构建特征

在某些情况下，用于构建MySQL的编译器会影响到可使用的特性。本节中的说明适用于由Oracle公司提供的或你自己从源码编译的二进制发行版。

icc（Intel C++编译器）构建

用icc构建的服务器有这些特点。

* 不包括SSL支持。

## 2.2 使用通用二进制文件在Unix/Linux上安装MySQL

甲骨文公司提供了一套MySQL的二进制发行版。这些包括为一些平台提供的压缩tar文件（扩展名为.tar.xz的文件）形式的通用二进制发布，以及为选定平台提供的特定平台包格式的二进制文件。

本节涵盖了在Unix/Linux平台上从压缩的tar文件二进制分布中安装MySQL。对于关注MySQL安全特性的Linux-通用二进制分布的安装说明，请参阅《安全部署指南》。对于其他平台特定的二进制包格式，请参阅本手册中其他平台特定的章节。例如，对于Windows发行版，见第2.3节 "在Microsoft Windows上安装MySQL"。关于如何获得不同分发格式的MySQL，见第2.1.3节 "如何获得MySQL"。

MySQL压缩的tar文件二进制发行版的名称是mysql-VERSION-OS.tar.xz，其中VERSION是一个数字（例如，8.0.26），OS表示该发行版所针对的操作系统类型（例如，pc-linux-i686或winx64）。

还有一个 "最小安装 "版本的MySQL压缩tar文件，用于Linux通用二进制发行版，它的名字是mysql-VERSION-OS-GLIBCVER-ARCH-minimal.tar.xz。最小安装发行版不包括调试二进制文件，并且剥离了调试符号，使得它比普通二进制发行版小得多。如果你选择安装最小安装版本，记得在后面的说明中调整文件名格式的差异。

警告

* 如果你以前使用你的操作系统本地软件包管理系统（如Yum或APT）安装MySQL，你可能会遇到使用本地二进制安装的问题。确保你以前的MySQL安装已被完全删除（使用你的软件包管理系统），并且任何额外的文件，如你的数据文件的旧版本，也已被删除。你还应该检查配置文件，如/etc/my.cnf或/etc/mysql目录并删除它们。

关于用官方MySQL包替换第三方包的信息，请参见相关的APT指南或Yum指南。

* MySQL有一个对libaio库的依赖性。如果本地没有安装这个库，数据目录初始化和随后的服务器启动步骤会失败。如果有必要，使用适当的软件包管理器安装它。例如，在基于Yum的系统上。

shell> ***yum search libaio*** # search for info

shell> ***yum install libaio*** # install library

或者，在基于 APT 的系统上。

shell> ***apt-cache search libaio*** # search for info

shell> ***apt-get install libaio1*** # install library

* Oracle Linux 8 / Red Hat 8 (EL8)。这些平台默认不安装文件/lib64/libtinfo.so.5，这是MySQL客户端bin/mysql对软件包mysql-VERSION-el7-x86\_64.tar.gz和mysql-VERSION-linux-glibc2.12-x86\_64.tar.xz的要求。为了解决这个问题，安装ncurses-compat-libs包。

shell> ***yum install ncurses-compat-libs***

要安装一个压缩的tar文件二进制分布，在你选择的安装位置（通常是/usr/local/mysql）解压。这将创建下表中所示的目录。

表2.3 通用Unix/Linux二进制包的MySQL安装布局

**Table 2.3 MySQL Installation Layout for Generic Unix/Linux Binary Package**

|  |  |
| --- | --- |
| **Directory** | **Contents of Directory** |
| bin | [**mysqld**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\programs.html#mysqld) server, client and utility programs |
| docs | MySQL manual in Info format |
| man | Unix manual pages |
| include | Include (header) files |
| lib | Libraries |
| share | Error messages, dictionary, and SQL for database installation |
| **support-files** | Miscellaneous support files |

mysqld二进制文件的调试版本可以作为mysqld-debug使用。要从源码发布中编译你自己的调试版本的MySQL，使用适当的配置选项来启用调试支持。参见第2.9节，"从源码安装MySQL"。

要安装和使用一个MySQL二进制发行版，命令序列看起来像这样。

shell> **groupadd mysql**

shell> **useradd -r -g mysql -s /bin/false mysql**

shell> **cd /usr/local**

shell> **tar xvf */path/to/mysql-VERSION-OS*.tar.xz**

shell> **ln -s *full-path-to-mysql-VERSION-OS* mysql**

shell> **cd mysql**

shell> **mkdir mysql-files**

shell> **chown mysql:mysql mysql-files**

shell> **chmod 750 mysql-files**

shell> **bin/mysqld --initialize --user=mysql**

shell> **bin/mysql\_ssl\_rsa\_setup**

shell> **bin/mysqld\_safe --user=mysql &**

# Next command is optional

shell> **cp support-files/mysql.server /etc/init.d/mysql.server**

注意

这个过程假设你有系统的root（管理员）权限。或者，你可以使用sudo(Linux)或pfexec(Solaris)命令给每个命令加上前缀。

mysql-files目录提供了一个方便的位置，可以作为secure\_file\_priv系统变量的值，它将导入和导出操作限制在一个特定的目录中。参见第5.1.8节，"服务器系统变量"。

下面是前面关于安装二进制发行版的更详细版本的描述。

创建一个mysql用户和组

如果你的系统还没有一个用于运行mysqld的用户和组，你可能需要创建它们。下面的命令添加了mysql组和mysql用户。你可能想把用户和组称为其他东西而不是mysql。如果是这样，在下面的指令中替换适当的名称。在不同版本的Unix/Linux上，useradd和groupadd的语法可能略有不同，或者它们可能有不同的名字，如adduser和addgroup。

shell> **groupadd mysql**

shell> **useradd -r -g mysql -s /bin/false mysql**

注意

因为用户只是出于所有权的目的，而不是登录的目的，所以useradd命令使用-r和-s /bin/false选项来创建一个对你的服务器主机没有登录权限的用户。如果你的useradd不支持这些选项，请省略它们。

获取并解压发行版

选择你想解压发行版的目录，并把位置改到它那里。这里的例子是在 /usr/local 下解压发行版。因此，说明中假设你有权限在 /usr/local 中创建文件和目录。如果该目录受到保护，你必须以root身份进行安装。

shell> **cd /usr/local**

使用第2.1.3节 "如何获得MySQL "中的说明获得一个分发文件。对于一个给定的版本，所有平台的二进制分发文件都是从相同的MySQL源分发文件构建的。

解压发行版，这将创建安装目录。如果tar有z选项支持，它可以解压和解压发行版。

shell> **tar xvf */path/to/mysql-VERSION-OS*.tar.xz**

tar命令创建了一个名为mysql-VERSION-OS的目录。

要从压缩的tar文件二进制发行版中安装MySQL，你的系统必须有GNU XZ Utils来解压缩发行版，并有合理的tar来解包。

注意

*在MySQL Server 8.0.12中，压缩算法从Gzip变为XZ；通用二进制文件的扩展名也从.tar.gz变为.tar.xz。*

众所周知，GNU tar是可以工作的。一些操作系统提供的标准tar不能解开MySQL发布中的长文件名。你应该下载并安装GNU tar，或者如果有的话，使用预装版本的

如果你的 tar 不支持 xz 格式，那么就用 xz 命令来解压发行版，用 tar 来解压。用下面的替代命令代替前面的tar命令来解压和提取发行版。

shell> **xz -dc */path/to/mysql-VERSION-OS*.tar.xz | tar x**

接下来，创建一个符号链接到由 tar 创建的安装目录。

shell> **ln -s *full-path-to-mysql-VERSION-OS* mysql**

ln命令为安装目录建立一个符号链接。这使你能够更容易地把它称为/usr/local/mysql。为了避免你在使用MySQL时总是要输入客户程序的路径名称，你可以将/usr/local/mysql/bin目录添加到你的PATH变量中。

shell> **export PATH=$PATH:/usr/local/mysql/bin**

执行安装后的设置

安装过程的其余部分涉及设置分发所有权和访问权限、初始化数据目录、启动MySQL服务器和设置配置文件。有关说明，见第2.10节 "安装后设置和测试"。

## 2.3 在Microsoft Windows上安装MySQL

/

重要的是

MySQL 8.0服务器需要Microsoft Visual C++ 2015 Redistributable Package才能在Windows平台上运行。用户在安装服务器之前，应确保该软件包已经安装在系统上。该软件包可在微软下载中心获得。此外，MySQL调试二进制文件需要安装Visual Studio 2015。

MySQL仅适用于微软Windows 64位操作系统。关于支持的Windows平台信息，见https://www.mysql.com/support/supportedplatforms/database.html。

在Microsoft Windows上安装MySQL有不同的方法。

**MySQL安装程序方法**

最简单和推荐的方法是下载MySQL安装器（用于Windows），让它安装和配置特定版本的MySQL服务器，方法如下。

1、从https://dev.mysql.com/downloads/installer/ 下载MySQL安装器并执行它。

注意

与标准的MySQL安装程序不同，较小的网络社区版本不捆绑任何MySQL应用程序，而是只下载你选择安装的MySQL产品。

2、确定初始安装MySQL产品时要使用的设置类型。例如。

* 开发者默认。提供一种设置类型，包括所选版本的MySQL Server和与MySQL开发有关的其他MySQL工具，如MySQL工作台。
* 仅限服务器。提供所选版本的MySQL Server的设置，而没有其他产品。
* 自定义。使你能够选择任何版本的MySQL服务器和其他MySQL产品。

3、安装服务器实例（和产品），然后按照屏幕上的说明开始进行服务器配置。关于每个单独步骤的更多信息，请参见第2.3.3.3.1节，"用MySQL安装程序配置MySQL服务器"。

现在MySQL已经安装完毕。如果你将MySQL配置为一个服务，那么每次重启系统时，Windows都会自动启动MySQL服务器。此外，这个过程还在本地主机上安装了MySQL安装程序，你以后可以用它来升级或重新配置MySQL服务器。

注意

如果你在系统上安装了MySQL Workbench，可以考虑用它来检查你新的MySQL服务器连接。默认情况下，该程序在安装MySQL后自动启动。

**其他安装信息**

可以将MySQL作为一个标准的应用程序或作为一个Windows服务运行。通过使用服务，你可以通过标准的Windows服务管理工具监测和控制服务器的运行。欲了解更多信息，请参见第2.3.4.8节 "以Windows服务方式启动MySQL"。

为了适应RESTART语句，当作为服务或独立运行时，MySQL服务器会分叉，以使一个监控进程监督服务器进程。在这种情况下，有两个mysqld进程。如果不需要RESTART能力，可以用--no-monitor选项启动服务器。参见第13.7.8.8节，"RESTART语句"。

一般来说，你应该用一个有管理员权限的账户在Windows上安装MySQL。否则，你可能会在某些操作上遇到问题，如编辑PATH环境变量或访问服务控制管理器。安装时，MySQL不需要使用具有管理员权限的用户来执行。

关于在Windows平台上使用MySQL的限制清单，见第2.3.7节 "Windows平台限制"。

除了MySQL服务器包外，你可能需要或想要额外的组件来将MySQL用于你的应用程序或开发环境。这些包括，但不限于。

* 要使用ODBC连接到MySQL服务器，你必须有一个Connector/ODBC驱动程序。欲了解更多信息，包括安装和配置说明，请参阅《MySQL Connector/ODBC开发指南》。

注意事项

MySQL安装程序为你安装和配置了Connector/ODBC。

* 要在.NET应用程序中使用MySQL服务器，你必须有Connector/NET驱动程序。欲了解更多信息，包括安装和配置说明，请参见《MySQL Connector/NET开发人员指南》。

注意

MySQL安装程序为你安装和配置MySQL Connector/NET。

用于Windows的MySQL发行版可以从https://dev.mysql.com/downloads/。参见第2.1.3节 "如何获得MySQL"。

用于Windows的MySQL有几种发布格式，详见这里。一般来说，你应该使用MySQL安装程序。它比老的MSI包含更多的功能和MySQL产品，比压缩文件更简单易用，而且你不需要额外的工具就能让MySQL启动和运行。MySQL安装程序自动安装MySQL服务器和其他MySQL产品，创建一个选项文件，启动服务器，并使你能够创建默认用户账户。关于选择软件包的更多信息，见第2.3.2节 "选择安装包"。

* 一个MySQL安装程序发行版包括MySQL服务器和其他MySQL产品，包括MySQL工作台，以及MySQL for Visual Studio。MySQL Installer也可以用来在未来升级这些产品（见https://dev.mysql.com/doc/mysql-compat-matrix/en/）。

关于使用MySQL安装器安装MySQL的说明，请参见第2.3.3节 "Windows的MySQL安装器"。

* 标准的二进制发布（以压缩文件的形式打包）包含所有必要的文件，你可以解压到你选择的位置。该包包含完整的Windows MSI安装包中的所有文件，但不包括安装程序。

关于使用压缩文件安装MySQL的说明，见第2.3.4节，"使用noinstall ZIP档案在Microsoft Windows上安装MySQL"。

* 源码分发格式包含使用Visual Studio编译器系统构建可执行文件的所有代码和支持文件。

有关在Windows上从源码构建MySQL的说明，见第2.9节 "从源码安装MySQL"。

**Windows上的MySQL考虑因素**

* 大表支持

如果你需要大小超过4GB的表，在NTFS或更新的文件系统上安装MySQL。当你创建表时，不要忘记使用MAX\_ROWS和AVG\_ROW\_LENGTH。见第13.1.20节，"CREATE TABLE语句"。

* MySQL和病毒检查软件

病毒扫描软件，如Norton/Symantec Anti-Virus，在包含MySQL数据和临时表的目录上，可能会导致问题，既影响MySQL的性能，又使病毒扫描软件误认为文件内容包含垃圾邮件。这是由于病毒扫描软件使用的指纹机制，以及MySQL快速更新不同文件的方式，可能被识别为潜在的安全风险。

在安装MySQL服务器后，建议你禁用用于存储MySQL表数据的主目录（datadir）的病毒扫描。病毒扫描软件中通常有一个系统，可以使特定的目录被忽略。

此外，默认情况下，MySQL在标准的Windows临时目录中创建临时文件。为了防止临时文件也被扫描，为MySQL临时文件配置一个单独的临时目录，并将该目录添加到病毒扫描排除列表中。要做到这一点，请在你的my.ini配置文件中为tmpdir参数增加一个配置选项。更多信息，见第2.3.4.2节 "创建选项文件"。

### 2.3.1 Microsoft Windows上的MySQL安装布局

对于Windows上的MySQL 8.0，对于用MySQL安装程序进行的安装，默认的安装目录是C:\Program Files\MySQL\MySQL Server 8.0。如果你使用ZIP存档方法来安装MySQL，你可能更喜欢安装在C:\mysql。然而，子目录的布局仍然是一样的。

所有的文件都位于这个父目录内，使用下表中所示的结构。

**Table 2.4 Default MySQL Installation Layout for Microsoft Windows**

|  |  |  |
| --- | --- | --- |
| **Directory** | **Contents of Directory** | **Notes** |
| bin | [**mysqld**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\programs.html#mysqld) server, client and utility programs |  |
| %PROGRAMDATA%\MySQL\MySQL Server 8.0\ | Log files, databases | The Windows system variable %PROGRAMDATA% defaults to C:\ProgramData. |
| docs | Release documentation | With MySQL Installer, use the **Modify** operation to select this optional folder. |
| include | Include (header) files |  |
| lib | Libraries |  |
| share | Miscellaneous support files, including error messages, character set files, sample configuration files, SQL for database installation |  |

### 2.3.2 选择一个安装包

对于MySQL 8.0，在Windows上安装MySQL时有多种安装包格式可供选择。本节中描述的包格式是。

* MySQL安装程序
* MySQL noinstall ZIP档案
* MySQL Docker图像

程序数据库（PDB）文件（文件名扩展名为pdb）在出现问题时为调试你的MySQL安装提供信息。这些文件包括在MySQL的ZIP档案分发中（但不是MSI分发）。

**MySQL安装程序**

这个软件包的文件名类似于mysql-installer-community-8.0.26.0.msi或mysql-installer-commercial-8.0.26.0.msi，它利用MSI自动安装MySQL服务器和其他产品。MySQL安装程序下载并将更新应用到自身和每个已安装的产品上。它还配置已安装的MySQL服务器（包括一个沙盒InnoDB集群测试设置）和MySQL路由器。建议大多数用户使用MySQL安装器。

MySQL安装器可以安装和管理（添加、修改、升级和删除）许多其他MySQL产品，包括。

* 应用程序--MySQL Workbench、MySQL for Visual Studio、MySQL Shell和MySQL Router（见<https://dev.mysql.com/doc/mysql-compat-matrix/en/>）
* 连接器--MySQL Connector/C++, MySQL Connector/NET, Connector/ODBC, MySQL Connector/Python, MySQL Connector/J, MySQL Connector/Node.js
* 文档 - MySQL手册（PDF格式）、样本和示例

MySQL安装程序可在所有MySQL支持的Windows版本上运行（见https://www.mysql.com/support/supportedplatforms/database.html）。

注意

由于MySQL安装程序不是微软Windows的本地组件，而是依赖于.NET，因此它不能与Windows Server的Server Core版本等最小安装选项一起使用。

关于如何使用MySQL安装器安装MySQL的说明，请参见第2.3.3节 "Windows的MySQL安装器"。

**MySQL noinstall ZIP档案**

这些包包含完整的MySQL服务器安装包中的文件，但GUI除外。这种格式不包括自动安装程序，必须手动安装和配置。

noinstall ZIP档案被分割成两个独立的压缩文件。主包被命名为mysql-VERSION-winx64.zip。这包含在你的系统上使用MySQL所需的组件。可选的MySQL测试套件、MySQL基准套件和调试二进制文件/信息组件（包括PDB文件）都在一个单独的压缩文件中，名为mysql-VERSION-winx64-debug-test.zip。

如果你选择安装noinstall ZIP档案，请参见第2.3.4节，"使用noinstall ZIP档案在Microsoft Windows上安装MySQL"。

**MySQL Docker图像**

关于在Windows平台上使用Oracle提供的MySQL Docker映像的信息，请参见第2.5.6.3节，"使用Docker在Windows和其他非Linux平台上部署MySQL"。

警告

甲骨文提供的MySQL Docker镜像是专门为Linux平台构建的。其他平台不受支持，用户在这些平台上运行甲骨文提供的MySQL Docker映像，其风险自负。

### 2.3.3 用于Windows的MySQL安装程序

MySQL Installer是一个独立的应用程序，旨在减轻在Microsoft Windows上运行的MySQL产品的安装和配置的复杂性。它支持以下MySQL产品。

* MySQL服务器

MySQL Installer可以同时在同一主机上安装和管理多个独立的MySQL服务器实例。例如，MySQL Installer可以在同一主机上安装、配置和升级MySQL 5.6、MySQL 5.7和MySQL 8.0的独立实例。MySQL安装程序不允许在主要和次要版本号之间进行服务器升级，但允许在一个发布系列内进行升级（如8.0.21到8.0.22）。

注意

MySQL安装程序不能在同一主机上同时安装MySQL服务器的社区和商业版本。如果你需要在同一台主机上安装这两个版本，请考虑使用ZIP存档分发来安装其中一个版本。

* MySQL应用程序

MySQL Workbench、MySQL Shell、MySQL Router和MySQL for Visual Studio。

* MySQL连接器

MySQL Connector/NET、MySQL Connector/Python、MySQL Connector/ODBC、MySQL Connector/J，以及MySQL Connector/C++。要安装MySQL Connector/Node.js，请参阅https://dev.mysql.com/downloads/connector/nodejs/。

* 文档和示例

PDF格式的MySQL参考手册（按版本）和MySQL数据库样本（按版本）。

**安装要求**

MySQL安装程序需要Microsoft .NET Framework 4.5.2或更高版本。如果主机上没有安装这个版本，你可以通过访问微软网站下载。

需要互联网连接来下载包含最新MySQL产品元数据的清单，这些产品不属于完整捆绑的一部分。当你第一次启动应用程序时，MySQL安装程序会尝试下载清单，然后以可配置的时间间隔定期下载（见MySQL安装程序选项）。或者，你可以通过点击MySQL安装程序仪表板中的目录来手动检索更新的清单。

注意

如果第一次或随后的清单下载不成功，会记录一个错误，并且在你的会话期间，你对MySQL产品的访问可能受到限制。MySQL安装程序会在每次启动时尝试下载清单，直到初始清单结构被更新。有关查找产品的帮助，请参阅查找要安装的产品。

**MySQL安装程序社区版**

从https://dev.mysql.com/downloads/installer/ 下载软件以安装Windows下所有MySQL产品的社区版。选择以下MySQL安装程序包选项之一。

* 网络。仅包含MySQL安装程序和配置文件。网络包选项只下载你选择安装的MySQL产品，但每次下载都需要互联网连接。该文件的大小约为2MB。文件名的形式是mysql-installer-community-web-VERSION.N.msi，其中VERSION是MySQL服务器版本号，如8.0，N是包号，从0开始。
* 完整或当前捆绑。捆绑了所有用于Windows的MySQL产品（包括MySQL服务器）。文件大小超过300MB，名称具有mysql-installer-community-VERSION.N.msi的形式，其中VERSION是MySQL服务器版本号，如8.0，N是包号，从0开始。

**MySQL安装程序商业版**

从https://edelivery.oracle.com/ 下载软件以安装Windows版MySQL产品的商业版（标准版或企业版）。如果你登录了你的My Oracle Support（MOS）帐户，商业版包括社区版中的所有当前和以前的GA版本，但它不包括开发里程碑版本。当你没有登录时，你只能看到你已经下载的捆绑产品的列表。

商业版还包括以下产品。

* Workbench SE/EE
* MySQL企业备份
* MySQL企业防火墙

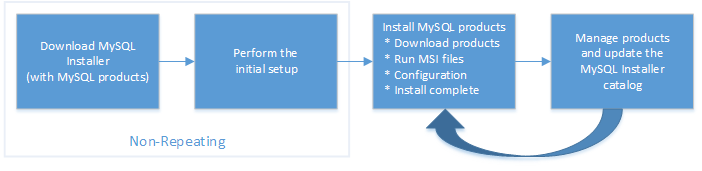
商业版与你的MOS账户集成。有关知识库内容和补丁，请参见My Oracle支持

#### 2.3.3.1 MySQL安装程序初始设置

* 选择一个安装类型
* 路径冲突
* 检查要求
* MySQL安装器配置文件

当你第一次下载MySQL Installer时，一个设置向导会引导你完成MySQL产品的初始安装。如下图所示，初始设置是整个过程中的一个一次性活动。在初始设置期间，MySQL Installer会检测到主机上安装的现有MySQL产品，并将它们添加到要管理的产品列表中。

**Figure 2.7 MySQL Installer Process Overview**



在初始设置期间，MySQL安装程序会提取配置文件（稍后描述）到主机的硬盘上。尽管MySQL安装程序是一个32位的应用程序，但它可以安装32位和64位的二进制文件。

初始设置会在 "开始 "菜单的MySQL组下添加一个链接。点击开始、所有程序、MySQL、MySQL安装器以打开MySQL安装器。

**选择一个安装类型**

在初始设置期间，会提示你选择要在主机上安装的MySQL产品。一种选择是使用符合你的设置要求的预先确定的设置类型。默认情况下，GA和预发布产品都包括在下载和安装的开发者默认、仅客户端和完整设置类型中。选择 "仅安装GA产品 "选项，以限制产品集，使其在使用这些设置类型时仅包括GA产品。

选择以下设置类型之一仅决定初始安装，不限制你以后为Windows安装或更新MySQL产品的能力。

* 开发人员默认设置：安装以下产品，这些产品可用于MySQL的应用开发。
  + MySQL服务器（安装你在下载MySQL安装程序时选择的版本。
  + MySQL Shell
  + MySQL Router
  + MySQL工作台（MySQL Workbench
  + 用于Visual Studio的MySQL
  + MySQL连接器（用于.NET / Python / ODBC / Java / C++）。
  + MySQL文档
  + MySQL样本和例子
* 仅限服务器。只安装MySQL服务器。这种设置类型会安装你在下载MySQL安装程序时选择的一般可用性（GA）或开发版本服务器。它使用默认的安装和数据路径。
* 仅客户端。只安装最新的MySQL应用程序和MySQL连接器。这种设置类型类似于开发者默认类型，除了它不包括MySQL服务器或通常与服务器捆绑的客户端程序，如mysql或mysqladmin。
* 完整：安装所有可用的MySQL产品。
* 自定义。自定义设置类型使你能够从MySQL安装程序目录中过滤和选择单个MySQL产品。

注意事项

对于MySQL Server 8.0.20（和更早的版本）、5.7和5.6版本，你用来运行MySQL Installer的账户可能没有足够的权限来安装服务器数据文件，这可能会中断安装，因为ExecSecureObjects MSI动作不能被执行。要继续进行，在尝试再次安装服务器之前取消选择服务器数据文件功能。如需帮助，请参阅《要安装的产品功能》）。)

服务器数据文件复选框已从MySQL Server 8.0.21（及更高版本）的功能树中移除。

使用自定义设置类型进行安装。

* 一个产品或产品版本无法从通常的下载位置获得。该目录包含所有产品版本，包括预发布（或开发）和GA之间的其他版本。
* 一个使用替代安装路径、数据路径或两者的MySQL服务器实例。关于如何调整路径的说明，见第2.3.3.2节 "用MySQL安装程序设置替代服务器路径"。
* 同一主机上同时有两个或多个MySQL服务器版本（例如，5.6、5.7和8.0）。
* 不作为预定设置类型提供的产品和功能的特定组合。例如，你可以安装单一产品，如MySQL Workbench，而不是安装Windows的所有客户端应用程序。

**路径冲突**

当要安装的产品的默认安装或数据文件夹（MySQL服务器要求的）已经存在于主机上时，向导会显示路径冲突步骤，以识别每个冲突，并使你能够采取行动，避免现有文件夹中的文件被新的安装覆盖。只有当MySQL安装程序检测到冲突时，你才会在初始设置中看到这个步骤。

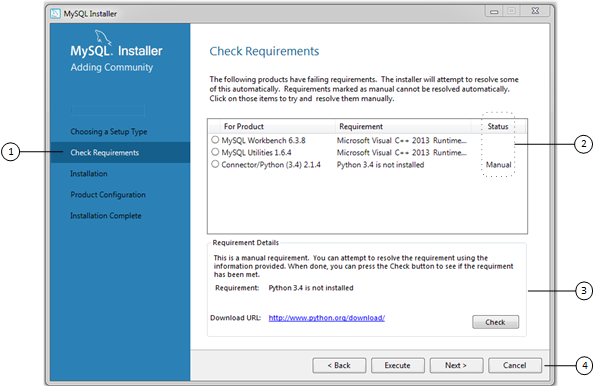
要解决路径冲突，请执行以下操作之一。

* 从列表中选择一个产品以显示冲突选项。一个警告符号表明哪个路径有冲突。使用浏览按钮选择一个新的路径，然后点击下一步。
* 点击 "返回"，选择不同的设置类型或产品版本（如果适用）。自定义设置类型使你能够选择单个产品版本。
* 点击 "下一步 "忽略冲突，覆盖现有文件夹中的文件。
* 删除现有产品。单击 "取消 "以停止初始设置并关闭MySQL安装程序。从 "开始 "菜单再次打开MySQL Installer，并使用MySQL Installer仪表板上的删除操作从主机上删除已安装的产品。

**检查要求**

MySQL Installer使用package-rules.xml文件中的条目来确定每个产品的先决条件软件是否安装在主机上。当需求检查失败时，MySQL安装程序显示检查需求步骤以帮助你更新主机。每次你下载一个新产品（或版本）进行安装时，都会对需求进行评估。下图识别并描述了该步骤的关键区域。

**Figure 2.8 Check Requirements**



检查要求元素的描述

1、显示初始设置中的当前步骤。此列表中的步骤可能会略有变化，这取决于主机上已经安装的产品、先决条件软件的可用性以及将在主机上安装的产品。

2、按产品列出所有待安装的要求，并表示状态如下。

* 状态栏中的空白表示MySQL安装程序可以尝试为你下载和安装所需软件。
* 状态栏中的单词Manual意味着你必须手动满足要求。选择列表中的每个产品以查看其需求细节。

3、详细描述了需求，以帮助你完成每一个手动解决。如果可能的话，会提供一个下载的URL。在你下载和安装所需的软件后，点击检查来验证需求是否得到满足。

4、提供以下设定的操作来进行。

* 返回 - 返回到上一步。该操作使你能够选择不同的设置类型。
* 执行 - 让MySQL安装程序尝试下载和安装所有没有手动状态的项目的所需软件。手动要求由你解决，并通过点击检查进行验证。
* 下一步 - 不执行自动应用要求的请求，继续安装，不包括检查要求步骤失败的产品。
* 取消 - 停止MySQL产品的安装。因为MySQL Installer已经安装好了，所以当你从开始菜单中打开MySQL Installer并从仪表板中点击添加时，初始设置会重新开始。关于可用的管理操作的描述，请参阅产品目录。

**MySQL安装器配置文件**

所有MySQL安装器文件都位于C:\Program Files（x86）和C:\ProgramData文件夹内。下表描述了将MySQL Installer定义为一个独立应用程序的文件和文件夹。

**注意**

当你更新或卸载MySQL Installer时，已安装的MySQL产品不会被改变或删除。

**Table 2.5 MySQL Installer Configuration Files**

|  |  |  |
| --- | --- | --- |
| **File or Folder** | **Description** | **Folder Hierarchy** |
| MySQL Installer for Windows | This folder contains all of the files needed to run MySQL Installer and [MySQLInstallerConsole.exe](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#MySQLInstallerConsole), a command-line program with similar functionality. | C:\Program Files (x86) |
| Templates | The Templates folder has one file for each version of MySQL server. Template files contain keys and formulas to calculate some values dynamically. | C:\ProgramData\MySQL\MySQL Installer for Windows\Manifest |
| package-rules.xml | This file contains the prerequisites for every product to be installed. | C:\ProgramData\MySQL\MySQL Installer for Windows\Manifest |
| produts.xml | The products file (or product catalog) contains a list of all products available for download. | C:\ProgramData\MySQL\MySQL Installer for Windows\Manifest |
| Product Cache | The Product Cache folder contains all standalone .msi files bundled with the full package or downloaded afterward. | C:\ProgramData\MySQL\MySQL Installer for Windows |

#### 2.3.3.2 用MySQL安装程序设置备选服务器路径

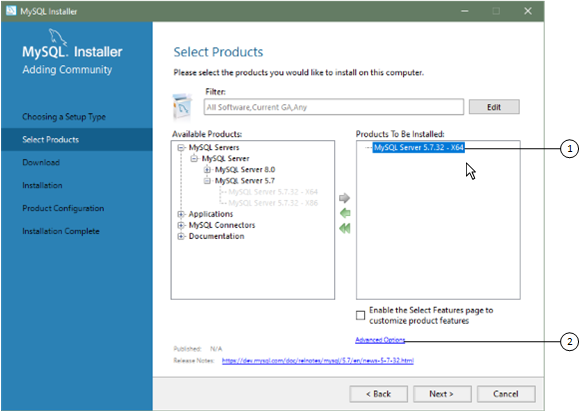
当你安装MySQL服务器时，你可以改变默认安装路径、数据路径或两者。在你安装了服务器后，如果不删除并重新安装服务器实例，就不能改变路径。

**要改变MySQL服务器的路径**

1. 识别要改变的MySQL服务器，并启用高级选项链接。
   1. 通过以下操作之一导航到 "选择产品 "页面。
      1. 如果这是MySQL Installer的初始设置，选择自定义设置类型并点击下一步。
      2. 如果已经安装了MySQL Installer，从 "开始 "菜单中启动它，然后从仪表板上点击添加。
   2. 单击 "编辑"，在 "可用产品 "中显示的产品列表上应用一个过滤器（见 "定位要安装的产品"）。
   3. 选定服务器实例后，使用箭头将选定的服务器移到待安装的产品列表中。
   4. 点击服务器来选择它。当你选择服务器时，高级选项链接在待安装产品列表下面被启用（见下图）。

2、点击 "高级选项"，打开一个对话框，你可以输入其他路径名称。在路径名称被验证后，点击下一步继续配置步骤。

图2.9 更改MySQL服务器路径



#### 2.3.3.3 使用MySQL安装器的安装工作流程

MySQL安装器提供了一个类似向导的工具来安装和配置Windows下的新MySQL产品。与只运行一次的初始设置不同，MySQL安装器在每次下载或安装新产品时都会调用该向导。对于首次安装，初始设置的步骤直接进入安装的步骤。有关产品选择的帮助，请参见查找要安装的产品。

注意

执行MySQL安装程序的用户被授予所有生成文件的全部权限，如my.ini。这不适用于特定产品的文件和目录，例如由SYSTEM拥有的位于%ProgramData%的MySQL服务器数据目录。

在主机上安装和配置的产品遵循一个一般的模式，在各个步骤中可能需要你的输入。如果你试图安装一个与现有MySQL服务器版本（或为升级而选择的版本）不兼容的产品，你会被提醒可能出现不匹配。

* MySQL安装程序提供了以下适用于不同工作流程的操作顺序。选择产品。 如果你在初始设置期间选择了自定义设置类型，或从MySQL安装器仪表板点击了添加，MySQL安装器在侧边栏中包括这一操作。从这个页面，你可以应用一个过滤器来修改可用产品列表，然后选择一个或多个产品来移动（使用箭头键）到要安装的产品列表中。

选择本页的复选框，激活 "选择功能 "动作，你可以在产品下载后自定义产品的功能。

* 下载。 如果你安装了完整的（非网络）MySQL安装包，所有的.msi文件在初始设置期间被加载到产品缓存文件夹中，不会再次下载。否则，单击 "执行 "以开始下载。每个产品的状态会从 "准备下载 "变为 "正在下载"，然后变为 "已下载"。
* 选择要安装的功能（默认为禁用）。 在MySQL安装程序下载产品的.msi文件后，如果你之前在选择产品操作期间启用了可选复选框，你可以自定义功能。
* 要在安装后定制产品功能，请在MySQL安装程序仪表板上单击 "修改"。
* 安装。 列表中每个产品的状态会从准备安装到安装，最后到完成。在这个过程中，单击 "显示细节 "以查看安装操作。

如果你在这时取消安装，产品已经安装，但服务器（如果已安装）还没有配置。要重新启动服务器配置，从 "开始 "菜单中打开MySQL安装程序，在仪表板中相应的服务器旁边点击重新配置

* 产品配置。 此步骤仅适用于MySQL服务器、MySQL路由器和样本。列表中每个项目的状态应显示准备配置。点击 "下一步"，启动列表中所有项目的配置向导。这一步中出现的配置选项是针对你选择安装的数据库或路由器的版本。

点击 "执行 "开始应用配置选项，或者点击 "返回"（重复）返回到每个配置页面。

* 安装完成。 对于不需要配置的产品，这一步最终完成了安装。它使你能够将日志复制到剪贴板并启动某些应用程序，如MySQL Workbench和MySQL Shell。单击 "完成 "以打开MySQL安装程序仪表板。

##### 2.3.3.3.1 用MySQL安装程序配置MySQL服务器

MySQL安装程序执行MySQL服务器的初始配置。例如。

* 它创建配置文件（my.ini），用于配置MySQL服务器。写入该文件的值受你在安装过程中做出的选择的影响。有些定义与主机有关。例如，如果主机少于三个核心，则启用query\_cache。

注意

查询缓存在MySQL 5.7中被废弃，并在MySQL 8.0（及更高版本）中被删除。

* 默认情况下，为MySQL服务器添加了一个Windows服务。
* 为MySQL服务器提供默认的安装和数据路径。关于如何改变默认路径的说明，见第2.3.3.2节，"用MySQL安装器设置替代服务器路径"。
* 它可以选择性地创建具有基于一般角色的可配置权限的MySQL服务器用户账户，如DB管理员、DB设计师和备份管理员。它可以选择性地创建一个名为MysqlSys的具有有限权限的Windows用户，然后由其运行MySQL服务器。

用户账户也可以在MySQL Workbench中添加和配置。

* 勾选 "显示高级选项 "可以设置额外的日志选项。这包括为错误日志、一般日志、慢速查询日志（包括配置执行查询所需的秒数）和二进制日志定义自定义文件路径。

在配置过程中，单击 "下一步 "进入下一个步骤，或单击 "返回 "返回到上一个步骤。在最后一步单击 "执行 "以应用服务器配置。

下面的章节描述了适用于Windows上MySQL服务器的服务器配置选项。你安装的服务器版本将决定你可以配置哪些步骤和选项。配置MySQL服务器可能包括部分或全部步骤。

###### 2.3.3.3.1.1 类型和网络

* 服务器配置类型

选择描述你的设置的MySQL服务器配置类型。该设置定义了分配给你的MySQL服务器实例的系统资源（内存）的数量。

* 开发。一台承载了许多其他应用程序的计算机，通常这是你的个人工作站。此设置将MySQL配置为使用最少的内存量。
* 服务器。预计其他几个应用程序将在这台计算机上运行，如Web服务器。服务器设置将MySQL配置为使用中等数量的内存。
* 专用的。一台专门用于运行MySQL服务器的计算机。因为没有其他主要应用程序在此服务器上运行，此设置将MySQL配置为使用大部分可用内存。
* 连接性

连接性选项控制与MySQL的连接方式。选项包括。

* TCP/IP：默认情况下，该选项被选中。你可以禁用TCP/IP网络，只允许本地主机连接。在选择了TCP/IP连接选项后，你可以修改以下项目。
* 经典MySQL协议连接的端口。默认值为3306。
* 仅在配置MySQL 8.0服务器时显示的X协议端口。默认值是33060
* 用于网络访问的开放的Windows防火墙端口，默认为TCP/IP连接所选。

如果一个端口号已经在使用，你会看到默认值旁边的信息图标（），并且下一步被禁用，直到你提供一个新的端口号。

* 命名的管道。启用并定义管道名称，类似于设置named\_pipe系统变量。默认名称是MySQL。
* 共享内存。启用并定义内存名称，类似于设置shared\_memory系统变量。默认名称是MySQL。
* 高级配置

勾选显示高级和日志选项，以便在后面的步骤中设置自定义日志和高级选项。日志选项步骤使你能够为错误日志、一般日志、慢速查询日志（包括其执行查询所需的秒数配置）和二进制日志定义自定义文件路径。高级选项步骤使你能够设置在复制拓扑中启用二进制日志时需要的唯一服务器ID。

* MySQL企业防火墙（仅企业版

启用MySQL企业防火墙复选框在默认情况下未被选中。选择此选项可启用安全列表，提供对某些类型攻击的保护。需要额外的安装后配置（见第6.4.7节，"MySQL企业防火墙"）。

重要提示

MySQL 8.0.19有一个问题，如果在服务器配置步骤中选择了MySQL Enterprise Firewall，则服务器无法启动。如果服务器启动操作失败，单击 "取消 "以结束配置过程并返回仪表板。你必须卸载服务器。

解决办法是在不选择MySQL企业防火墙的情况下运行MySQL安装程序。(也就是说，不要选择启用MySQL企业防火墙复选框。)然后使用手动安装的说明在之后安装MySQL企业防火墙(见第6.4.7.2节，"安装或卸载MySQL企业防火墙")。

###### 2.3.3.3.1.2 认证方法

认证方法步骤仅在安装或升级MySQL 8.0.4或更高版本时可见。它引入了两个服务器端认证选项的选择。你在下一步创建的MySQL用户账户将使用你在该步骤中选择的认证方法。

使用libmysqlclient 8.0的MySQL 8.0连接器和社区驱动程序现在支持mysql\_native\_password默认认证插件。然而，如果你无法更新你的客户端和应用程序以支持这种新的认证方法，你可以配置MySQL服务器以使用mysql\_native\_password进行传统认证。有关这一变化的影响的更多信息，请参阅缓存\_sha2\_password作为首选认证插件。

如果你正在安装或升级到MySQL 8.0.4或更高版本，请选择以下认证方法之一。

* 使用强密码加密认证（推荐）。

MySQL 8.0支持基于改进的、更强大的基于SHA256的密码方法的新认证。建议所有新安装的MySQL服务器今后都使用这种方法。

重要提示

服务器上的caching\_sha2\_password认证插件需要新版本的连接器和客户端，它们增加了对新的MySQL 8.0默认认证的支持。

* 使用传统的认证方法（保留MySQL 5.x的兼容性

使用旧的MySQL 5.x遗留认证方法应仅在以下情况下考虑。

* 应用程序不能被更新以使用MySQL 8.0连接器和驱动程序
* 重新编译一个现有的应用程序是不可行的。
* 更新的、特定于语言的连接器或驱动程序尚不可用。

###### 2.3.3.3.1.3 账户和角色

* 根账户密码

指定一个根密码是必需的，在执行其他MySQL安装程序操作时，你将被要求提供该密码。当你在提供的方框中重复密码时，将评估密码强度。关于密码要求或状态的描述性信息，当信息图标（）出现时，将鼠标指针移到它上面。

* MySQL用户账户（可选

单击 "添加用户 "或 "编辑用户 "来创建或修改具有预定义角色的MySQL用户账户。接下来，输入所需的账户凭证。

* 用户名。MySQL用户名最多可以有32个字符。
* 主机。仅为本地连接选择localhost，或者当需要远程连接到服务器时选择<所有主机(%)>。
* 角色。每个预定义的角色，如DB管理员，都配置有自己的一套权限。例如，DB Admin角色比DB Designer角色有更多的权限。角色下拉列表包含每个角色的描述。
* 密码：当你输入密码时，会进行密码强度评估。密码必须被确认。MySQL允许空白或空密码（被认为是不安全的）。

仅限MySQL安装程序商业版。 用于Windows的MySQL企业版是一种商业产品，它也支持在Windows上执行外部认证的认证方法。由Windows操作系统认证的账户可以访问MySQL服务器，而无需提供额外的密码。

要创建一个使用Windows认证的新MySQL账户，请输入用户名，然后为主机和角色选择一个值。单击Windows认证以启用authentication\_windows插件。在Windows安全令牌区域，为每个可以用MySQL用户名认证的Windows用户（或组）输入一个令牌。MySQL账户可以包括本地Windows用户和属于一个域的Windows用户的安全令牌。多个安全令牌由分号（;）分隔，并对本地和域账户使用以下格式。

* 本地帐户

输入简单的Windows用户名作为每个本地用户或组的安全令牌；例如，finley;jeffrey;admin。

* 域账户

使用标准的Windows语法（domain/domainuser）或MySQL语法（domain/domainuser）来输入Windows域用户和组。

对于域账户，如果运行MySQL安装程序的账户缺乏查询活动目录的权限，你可能需要使用域内管理员的凭证。如果是这种情况，选择验证活动目录用户，以激活域管理员的凭证。

Windows认证允许你在每次添加或修改令牌时测试所有的安全令牌。点击测试安全令牌来验证（或重新验证）每个令牌。无效的令牌会产生一个描述性的错误信息，以及一个红色的X图标和红色的令牌文本。当所有令牌解决为有效时（绿色文本没有X图标），你可以点击确定来保存更改。

###### 2.3.3.3.1.4 Windows服务

在Windows平台上，MySQL服务器可以作为一个由操作系统管理的命名服务运行，并被配置为在Windows启动时自动启动。另外，你也可以将MySQL服务器配置为需要手动配置的可执行程序来运行。

* 将MySQL服务器配置为一个Windows服务（默认选择。

当选择默认配置选项时，你还可以选择以下内容。

* 在系统启动时启动MySQL服务器

当选择时（默认），服务启动类型被设置为自动；否则，启动类型被设置为手动。

* 运行Windows服务为

当选择标准系统账户时（默认），服务作为网络服务登录。

自定义用户选项必须有权限以服务方式登录Windows系统。下一步 "按钮将被禁用，直到该用户被配置为所需权限。

自定义用户账户是通过在开始菜单中搜索 "本地安全策略 "在Windows中配置的。在 "本地安全策略 "窗口中，选择 "本地策略"、"用户权限分配"，然后选择 "作为服务登录 "来打开属性对话框。单击 "添加用户或组 "以添加自定义用户，然后在每个对话框中单击 "确定 "以保存更改。

* 取消选择Windows服务选项

###### 2.3.3.3.1.5 日志选项

如果在类型和网络步骤中选择了显示高级配置复选框，则此步骤可用。要现在启用这个步骤，请单击 "返回 "返回到 "类型和网络 "步骤，并选择该复选框。

高级配置选项与下列MySQL日志文件有关。

* 错误日志
* 一般日志
* 慢速查询日志
* 仓库日志

注意

对于MySQL 5.7及以上版本，二进制日志是默认启用的。

###### 2.3.3.3.1.6 高级选项

如果在类型和网络步骤中选择了显示高级配置复选框，就可以使用这个步骤。要现在启用这个步骤，请单击 "返回 "返回到 "类型和网络 "步骤，并选择复选框。

高级配置选项包括。

* 服务器ID

设置复制拓扑结构中使用的唯一标识符。如果启用了二进制日志，你必须指定一个服务器ID。默认的ID值取决于服务器的版本。欲了解更多信息，请参见server\_id系统变量的描述。

* 表名案例

你可以在服务器的初始和后续配置中设置以下选项。对于MySQL 8.0发布系列，这些选项仅适用于服务器的初始配置。

* 小写字母

将lower\_case\_table\_names选项值设置为1（默认），在这种情况下，表名在磁盘上以小写存储，并且比较不区分大小写。

* 保留给定大小写

将lower\_case\_table\_names选项的值设置为2，在这种情况下，表名会按照给定的大小写存储，但是比较时要用小写。

###### 2.3.3.3.1.7 应用服务器配置

当你点击执行时，所有配置设置都会应用到MySQL服务器上。使用 "配置步骤 "选项卡来跟踪每个操作的进展；成功时，每个操作的图标会从白色切换到绿色（有一个复选标记）。否则，如果某项操作超时，进程就会停止并显示错误信息。单击 "日志 "选项卡以查看日志。

当安装成功完成并点击完成时，MySQL安装程序和已安装的MySQL产品被添加到Microsoft Windows开始菜单的MySQL组下。打开MySQL安装程序会加载仪表板，其中列出了已安装的MySQL产品，并提供了其他MySQL安装程序操作。

##### 2.3.3.3.2 用MySQL安装程序配置MySQL路由器

MySQL安装程序下载并安装了一套用于在Windows上开发和管理关键业务应用程序的工具。该套件由应用程序、连接器、文档和样本组成。

在初始设置期间，选择任何预先确定的设置类型，除了仅服务器，以安装工具的最新GA版本。使用自定义设置类型来安装单个工具或特定版本。如果MySQL Installer已经安装在主机上，使用添加操作从MySQL Installer仪表板上选择并安装工具。

MySQL路由器配置

MySQL安装程序提供了一个配置向导，可以启动已安装的MySQL Router 8.0实例，以引导MySQL应用程序和InnoDB簇之间的流量。当被配置时，MySQL Router作为一个本地Windows服务运行。

注意

在初始安装后，以及当你明确地重新配置已安装的路由器时，会提示你配置MySQL Router。相反，升级操作不要求或提示你配置升级后的产品。

要配置MySQL Router，请执行以下操作。

1、设置InnoDB群集。

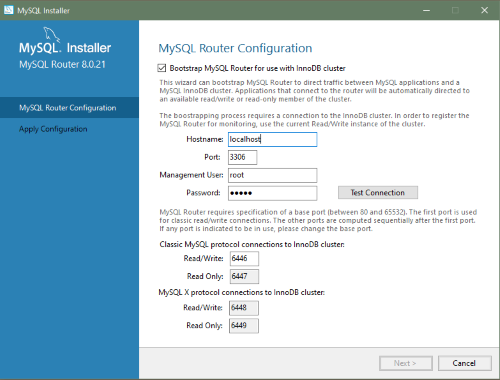
2、使用MySQL安装程序，下载并安装MySQL Router应用程序。安装完成后，配置向导会提示你提供信息。选择为InnoDB簇配置MySQL路由器复选框以开始配置，并提供以下配置值。

* 主机名：InnoDB簇中主（种子）服务器的主机名（默认为localhost）。
* 端口：InnoDB集群中的主（种子）服务器的端口号（默认为3306）。
* 管理用户：具有root级别权限的管理用户。
* 密码：管理用户的密码。
* 与InnoDB Cluster的经典MySQL协议连接

读/写。将第一个基本端口号设置为一个未使用的端口（在80和65532之间），向导将为你选择其余的端口。

下面的图显示了MySQL路由器配置页面的一个例子，第一个基本端口号被指定为6446，其余端口被向导设置为6447、6448和6449。

图2.10 MySQL路由器配置



3、点击 "下一步"，然后点击 "执行 "以应用配置。单击 "完成 "关闭MySQL安装程序或返回到MySQL安装程序仪表板。

在配置MySQL Router之后，根账户在用户表中只作为root@localhost（本地）存在，而不是root@%（远程）。无论路由器和客户端位于何处，即使两者都位于与种子服务器相同的主机上，任何通过路由器的连接都被服务器视为远程，而不是本地。因此，使用本地主机与服务器建立的连接（见下面的例子），不会被认证。

shell> **\c root@localhost:6446**

#### 2.3.3.4 MySQL安装程序产品目录和仪表板

本节介绍MySQL安装程序的产品目录、仪表板以及与产品选择和升级有关的其他操作。

* 产品目录
* MySQL安装器仪表板
* 找到要安装的产品
* 升级MySQL服务器
* 移除MySQL服务器
* 升级MySQL安装程序

**产品目录**

产品目录存储了可从MySQL下载的用于Microsoft Windows的MySQL发布产品的完整列表。默认情况下，当有互联网连接时，MySQL安装程序会尝试每七天在启动时更新目录。你也可以从仪表板上手动更新目录（稍后描述）。

一个最新的目录会执行以下操作。

* 填充 "选择产品 "页面的 "可用产品 "窗格。这个步骤在你选择时出现。
  + 在初始设置中选择自定义设置类型。
  + 从仪表板上的添加操作。
* 确定仪表盘中列出的已安装产品的产品更新何时可用。

该目录包括所有开发版本（Pre-Release）、一般版本（Current GA）和次要版本（Other Releases）。目录中的产品会有些变化，这取决于你下载的MySQL安装程序版本。

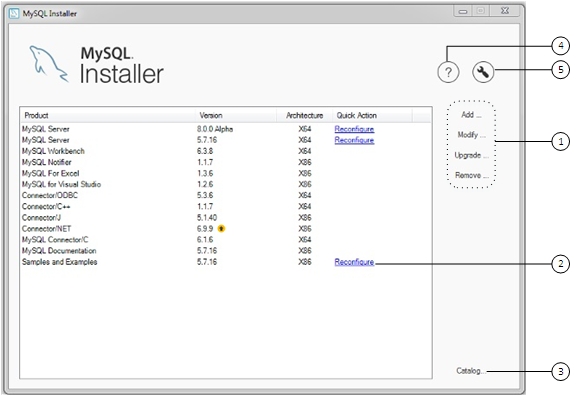
**MySQL安装器仪表板**

MySQL安装器仪表板是初始设置完成后启动MySQL安装器时看到的默认视图。如果你在设置完成之前关闭了MySQL安装器，MySQL安装器在显示仪表板之前会恢复初始设置。

注意

如果安装了Oracle终身持续支持下的产品，可能会出现在仪表板中。这些产品，如MySQL for Excel和MySQL Notifier，只能被修改或删除。

图2.11 MySQL安装器仪表板元素



MySQL安装器仪表板元素的描述

1、MySQL安装程序仪表板操作提供了各种适用于已安装产品或目录中所列产品的操作。要启动以下操作，首先点击操作链接，然后选择要管理的产品或产品。

* 添加。此操作打开 "选择产品 "页面。在那里你可以调整过滤器，选择一个或多个产品来下载（根据需要），并开始安装。关于使用过滤器的提示，请看定位要安装的产品。

使用方向箭头将每个产品从 "可用产品 "栏移到 "待安装产品 "栏。要启用你可以定制功能的 "产品功能 "页面，请点击相关的复选框（默认为禁用）。

注意

对于MySQL Server 8.0.20（和更早的版本）、5.7和5.6版本，你用来运行MySQL Installer的账户可能没有足够的权限来安装服务器数据文件，这可能会中断安装，因为ExecSecureObjects MSI动作不能被执行。要继续进行，在尝试再次安装服务器之前取消选择服务器数据文件功能。

服务器数据文件复选框已从MySQL Server 8.0.21（或更高版本）的特征树中删除。

* 修改。使用此操作来添加或删除与已安装产品相关的功能。你可以修改的功能因产品的复杂性而异。当选择了程序快捷方式复选框时，该产品会出现在开始菜单的MySQL组下。
* 升级：该操作会加载 "选择要升级的产品 "页面，并将所有升级候选产品填充到该页面中。一个已安装的产品可以有一个以上的升级版本，该操作需要一个当前的产品目录。MySQL安装程序在一个操作中升级所有选定的产品。单击 "显示详情 "以查看MySQL安装器执行的操作。
* 移除。此操作打开 "移除产品 "页面，并以安装在主机上的MySQL产品填充它。选择你要删除（卸载）的MySQL产品，然后点击执行，开始删除过程。在操作过程中，有一个指标显示执行的步骤数占所有步骤的百分比。

要选择要删除的产品，请执行以下操作之一。

* 选择一个或多个产品的复选框。
* 选择产品复选框来选择所有产品。

2、在每个已安装的服务器旁边的 "快速行动 "栏中的 "重新配置 "链接加载服务器的当前配置值，然后循环执行所有配置步骤，使你能够改变选项和值。你必须提供具有root权限的凭证来重新配置这些项目。单击 "日志 "选项卡，显示MySQL Installer执行的每个配置步骤的输出。

完成后，MySQL安装程序会停止服务器，应用配置更改，并为你重新启动服务器。关于每个配置选项的描述，请参阅第2.3.3.3.1节 "用MySQL安装程序配置MySQL服务器"。与特定MySQL服务器版本相关的已安装样本和示例也可以被重新配置以应用新的功能设置（如果有的话）。

3、目录链接使你能够手动下载MySQL产品的最新目录，然后将这些产品变化与MySQL安装程序集成。目录下载操作并不对已经安装在主机上的产品进行升级。相反，它返回到仪表板，并为每个具有较新版本的已安装产品在版本栏中添加一个箭头图标。使用升级操作来安装较新的产品版本。

你也可以使用目录链接来显示每个产品的当前变化历史，而不下载新的目录。选择 "此时不更新 "复选框，只查看更改历史。

4、MySQL安装器关于图标（）显示MySQL安装器的当前版本和关于MySQL的一般信息。版本号位于 "返回 "按钮的上方。

提示

在报告MySQL安装器的问题时，总是包括这个版本号。

除了关于MySQL的信息（），你还可以从侧板上选择以下图标。

* MySQL安装器的许可证图标（）。

本产品可能包括第三方软件，在许可证下使用。如果你使用的是MySQL Installer的商业版本，该图标将打开MySQL Installer商业许可信息用户手册，以获取许可信息，包括与可能包含在该商业版本中的第三方软件有关的许可信息。如果你使用的是MySQL Installer的社区版本，该图标将打开MySQL Installer社区许可信息用户手册，以获得许可信息，包括与可能包含在该社区版本中的第三方软件有关的许可信息。

* 资源链接 图标（）指向最新的MySQL产品文档、博客、网络研讨会等。

5、MySQL安装程序选项图标（）包括以下标签。

* 产品目录。管理自动目录更新。默认情况下，MySQL安装程序每七天在启动时检查目录更新。当新产品或产品版本可用时，MySQL安装程序将它们添加到目录中，然后在仪表板中列出的已安装产品的版本号旁边插入一个箭头图标（）。

使用产品目录选项来启用或禁用自动更新，并重置自动下载目录之间的天数。在启动时，MySQL安装程序使用你设置的天数来决定是否应尝试下载。如果MySQL Installer在下载目录时遇到错误，在下次启动时将重复这一操作。

* 连接性设置。由MySQL安装程序执行的一些操作需要互联网访问。这个选项使你能够使用一个默认值来验证连接，或使用一个不同的URL，一个从列表中选择的或由你手动添加的URL。在选择手动选项时，可以添加新的URL，并且可以移动或删除列表中的所有URL。当选择自动选项时，MySQL安装程序会尝试连接到列表中的每个默认URL（按顺序），直到建立连接。如果不能建立连接，它将引发一个错误。

**找到要安装的产品**

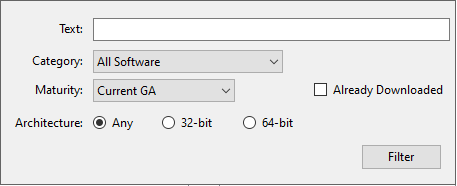
目录中的MySQL产品按类别列出。MySQL服务器、应用程序、MySQL连接器和文档。默认情况下，只有最新的GA版本出现在可用产品窗格中。如果你正在寻找某个产品的预发布版本或旧版本，它可能在默认列表中不可见。

注意

保持产品目录是最新的。单击MySQL安装程序仪表板上的目录以下载最新的清单。

要改变默认的产品列表，在仪表板上点击添加以打开 "选择产品 "页面，然后点击编辑以打开下图所示的对话框。修改设置，然后点击过滤。

图2.12 筛选可用产品



重置以下一个或多个字段来修改可用产品的列表。

* 文本。按文本过滤。
* 种类：通过文字过滤。所有软件（默认）、MySQL服务器、应用程序、MySQL连接器或文档（用于样本和文档）。
* 成熟度。当前的捆绑（最初只出现在完整的软件包中），预发布，当前的GA，或其他版本。如果你看到一个警告，通过点击MySQL安装程序仪表板上的目录，确认你有最新的产品清单。如果MySQL安装程序无法下载清单，你看到的产品范围仅限于捆绑的产品、已经位于产品缓存文件夹中的独立产品MSI，或两者。

注意

当你选择Pre-Release成熟度过滤器时，MySQL Installer的商业版本不显示任何MySQL产品。开发中的产品只能从MySQL Installer的社区版中获得。

* 已经下载（该复选框在默认情况下是取消选择的）。只允许你查看和管理已下载的产品。
* 架构。任何（默认）、32位或64位。

**升级MySQL服务器**

重要的服务器升级条件。

* MySQL安装程序不允许在主要发布版本或次要发布版本之间进行服务器升级，但允许在一个发布系列内进行升级，例如从5.7.18升级到5.7.19。
* 不支持里程碑版本之间的升级（或从一个里程碑版本到GA版本）。在里程碑版本中会发生重大的开发变化，你可能会遇到兼容性问题或启动服务器的问题。
* 对于升级到MySQL 8.0.16服务器及更高版本，一个复选框使你能够跳过系统表的升级检查和处理，而正常检查和处理数据字典表。当以前的服务器升级被跳过或服务器被配置为沙盒InnoDB簇时，MySQL安装程序不会用该复选框提示你。这种行为代表了MySQL服务器执行升级的方式的变化（参见第2.11.3节，"MySQL升级过程升级的内容"），它改变了MySQL安装程序应用于配置过程的步骤顺序。

如果你选择跳过系统表升级检查和过程。不推荐），MySQL安装程序将以--upgrade=MINIMAL服务器选项启动升级后的服务器，它只升级数据字典。如果你停止，然后重新启动服务器而不使用--upgrade=MINIMAL选项，如果需要，服务器会自动升级系统表。

在升级配置（跳过系统表）完成后，以下信息出现在日志选项卡和日志文件中。

WARNING: The system tables upgrade was skipped after upgrading MySQL Server. The

server will be started now with the --upgrade=MINIMAL option, but then each

time the server is started it will attempt to upgrade the system tables, unless

you modify the Windows service (command line) to add --upgrade=MINIMAL to bypass

the upgrade.

FOR THE BEST RESULTS: Run mysqld.exe --upgrade=FORCE on the command line to upgrade

the system tables manually.

要选择一个新的服务器版本。

1、点击升级。确认 "可升级产品 "窗格中的产品名称旁边的复选框有一个复选标记。取消选择你不打算在这个时候升级的产品。

注意

对于同一发行系列中的服务器里程碑版本，MySQL安装程序会取消选择服务器升级，并显示警告以表明不支持该升级，确定继续下去的风险，并提供手动执行逻辑升级的步骤摘要。你可以重新选择服务器升级，风险由你自己承担。关于如何用一个里程碑版本执行逻辑升级的说明，请看逻辑升级。

2、单击列表中的一个产品以突出显示它。此操作会在可升级版本窗格中填入所选产品的每个可用版本的详细信息：版本号、发布日期以及打开该版本的发行说明的更改链接。

**移除MySQL服务器**

要删除一个本地MySQL服务器。

1、确定是否应删除本地数据目录。如果你保留数据目录，另一个服务器安装可以重新使用这些数据。该选项默认为启用（删除数据目录）。

2、点击执行，开始卸载本地服务器。注意，所有你选择删除的产品在这个时候也会被卸载。

3、可选）单击 "日志 "选项卡以显示MySQL Installer当前执行的操作。

**升级MySQL安装器**

MySQL Installer仍然安装在你的计算机上，像其他软件一样，MySQL Installer可以从以前的版本升级。在某些情况下，其他MySQL软件可能需要你升级MySQL安装器以获得兼容性。本节介绍了如何识别MySQL安装器的当前版本以及如何手动升级MySQL安装器。

要找到已安装的MySQL安装器的版本。

1、从搜索菜单中启动MySQL安装器。打开MySQL安装器仪表板。

2、点击MySQL安装器关于图标（）。版本号位于 "返回 "按钮的上方。

要启动MySQL Installer的按需升级。

1. 将安装了MySQL Installer的计算机连接到互联网。
2. 从搜索菜单中启动MySQL安装器。MySQL安装器仪表板打开。
3. 点击仪表板底部的Catalog，打开Update Catalog窗口。
4. 单击 "执行 "以开始该过程。如果已安装的MySQL Installer版本可以升级，你将被提示开始升级。
5. 点击 "下一步 "查看目录的所有更改，然后点击 "完成 "返回仪表板。
6. 验证MySQL Installer的（新）安装版本（见前面的程序）。

#### 2.3.3.5 MySQLInstallerConsole参考

MySQLInstallerConsole.exe提供与MySQL Installer类似的命令行功能。它在最初执行MySQL Installer时被安装，然后在MySQL Installer for Windows目录中可用。默认情况下，它位于C:\Program Files (x86)\MySQL\MySQL Installer for Windows，而且必须以管理权限执行该控制台。

要使用，通过选择 "开始"、"附件 "以管理权限调用命令提示符，然后右键单击命令提示符，选择以管理员身份运行。并从命令行中，有选择地改变MySQLInstallerConsole.exe所在的目录。

C:\> **cd Program Files (x86)\MySQL\MySQL Installer for Windows**

C:\Program Files (x86)\MySQL\MySQL Installer for Windows> **MySQLInstallerConsole.exe help**

=================== Start Initialization ===================

MySQL Installer is running in Community mode

Attempting to update manifest.

Initializing product requirements

Loading product catalog

Checking for product catalog snippets

Checking for product packages in the bundle

Categorizing product catalog

Finding all installed packages.

Your product catalog was last updated at 11/1/2016 4:10:38 PM

=================== End Initialization ===================

The following commands are available:

Configure - Configures one or more of your installed programs.

Help - Provides list of available commands.

Install - Install and configure one or more available MySQL programs.

List - Provides an interactive way to list all products available.

Modify - Modifies the features of installed products.

Remove - Removes one or more products from your system.

Status - Shows the status of all installed products.

Update - Update the current product catalog.

Upgrade - Upgrades one or more of your installed programs.

**MySQL产品名称**

许多MySQLInstallerConsole命令接受一个或多个关键字，代表目录中的MySQL产品（或产品）。当前与命令一起使用的有效关键字集如下表所示。

**Table 2.6 MySQL Product Keywords for MySQLInstallerConsole**

|  |  |
| --- | --- |
| **Keyword** | **MySQL Product** |
| **server** | MySQL Server |
| **workbench** | MySQL Workbench |
| **shell** | MySQL Shell |
| **visual** | MySQL for Visual Studio |
| **router** | MySQL Router |
| **backup** | MySQL Enterprise Backup |
| **net** | MySQL Connector/NET |
| **odbc** | MySQL Connector/ODBC |
| **c++** | MySQL Connector/C++ |
| **python** | MySQL Connector/Python |
| **j** | MySQL Connector/J |
| **documentation** | MySQL Server Documentation |
| **samples** | MySQL Samples (sakila and world databases |

MySQLInstallerConsole命令选项

MySQLInstallerConsole.exe支持下列命令选项。

注意

包含冒号字符（:）的配置块值必须用引号包裹。例如，installdir="C:\MySQL\MySQL Server 8.0"。

* configure [product1]:[setting]=[value]; [product2]:[setting]=[value]; [...]

在你的系统上配置一个或多个MySQL产品。可以为每个产品配置多个设置=值对。

开关包括。

-显示设置

通过在-showsettings后面传入产品名称，显示所选产品的可用选项。

-静音

禁用确认提示。

C:\> **MySQLInstallerConsole configure -showsettings server**

C:\> **MySQLInstallerConsole configure server:port=3307**

* **help *[command]***

显示一个带有使用例子的帮助信息，然后退出。传入一个额外的命令以接收该命令的特定帮助。

C:\> **MySQLInstallerConsole help**

C:\> **MySQLInstallerConsole help install**

* **install [*product*]:[*features*]:[*config block*]:[*config block*]:[*config block*]; [*...*]**

在你的系统上安装一个或多个MySQL产品。如果预发布产品可用，当-type开关的值为Developer、Client或Full时，GA和预发布产品都会安装。使用-only\_ga\_products开关，在使用这些设置类型时，将产品集仅限制为GA产品。

开关和语法选项包括。

-only\_ga\_products

将产品集限制为只包括GA产品。

-type=[SetupType]

安装一个预定义的软件集。设置类型可以是以下之一。

* 开发者。安装一个完整的开发环境。
* 服务器。安装一个单一的MySQL服务器
* 客户端。安装客户端程序和库
* 完全：安装所有的东西
* 自定义。安装用户选择的产品。这是默认选项。

注意

非自定义设置类型只有在没有安装其他MySQL产品时才有效。

**-showsettings**

通过在-showsettings后面传入产品名称，显示所选产品的可用选项。

**-silent**

禁用确认提示。

**[*product*]**

每个产品都可以通过产品关键字指定，也可以不使用分号分隔的版本限定符。单独传入一个产品关键字会选择该产品的最新版本。如果该产品的版本有多个架构，命令会返回清单中的第一个架构，供交互式确认。另外，您可以使用-silent 开关，在产品关键字之后传入准确的版本和架构（x86 或 x64）。

**[*features*]**

与MySQL产品相关的所有特性都是默认安装的。特征块是一个分号分隔的特征列表，或一个星号字符（\*），选择所有的特征。要删除一个特性，请使用修改命令。

**[*config block*]**

可以指定一个或多个配置块。每个配置块是一个由分号分隔的键值对列表。一个块可以包括config或user类型的键；如果没有定义，config是默认类型。

包含冒号字符（:）的配置块值必须用引号包裹。例如，installdir="C:\MySQL\MySQL Server 8.0"。每个产品只能定义一个配置类型块。在产品安装过程中，应该为每个要创建的用户定义一个用户块。

注意

当一个产品被重新配置时，不支持用户类型键。

C:\> **MySQLInstallerConsole install server;5.6.25:\*:port=3307;serverid=2:type=user;username=foo;password=bar;role=DBManager**

C:\> **MySQLInstallerConsole install server;5.6.25;x64 -silent**

一个传入额外配置块的例子，用^分隔以适应。

C:\> **MySQLInstallerConsole install server;5.6.25;x64:\*:type=config;openfirewall=true; ^**

**generallog=true;binlog=true;serverid=3306;enable\_tcpip=true;port=3306;rootpasswd=pass; ^**

**installdir="C:\MySQL\MySQL Server 5.6":type=user;datadir="C:\MySQL\data";username=foo;password=bar;role=DBManager**

**list**

列出一个交互式控制台，在那里可以搜索到所有可用的MySQL产品。执行MySQLInstallerConsole list来启动控制台，并输入子串进行搜索。

C:\> **MySQLInstallerConsole list**

**modify [*product1*:-*removelist*|+*addlist*] [*product2*:-*removelist*|+*addlist*] [*...*]**

修改或显示先前安装的MySQL产品的特性。要显示一个产品的特性，在命令中附加产品关键字，例如。

C:\> **MySQLInstallerConsole modify server**

此命令的语法选项。

-silent

禁用确认提示。

C:\> **MySQLInstallerConsole modify server:+documentation**

C:\> **MySQLInstallerConsole modify server:-debug**

**remove [*product1*] [*product2*] [*...*]**

从你的系统中删除一个或多个产品。开关和语法选项包括。

\*

传入\*以删除所有的MySQL产品。

-continue

即使发生错误也继续操作。

-silent

禁用确认提示。

C:\> **MySQLInstallerConsole remove \***

C:\> **MySQLInstallerConsole remove server**

**status**

提供系统上安装的MySQL产品的快速概览。信息包括产品名称和版本、架构、安装日期和安装位置。

C:\> **MySQLInstallerConsole status**

**update**

将最新的MySQL产品目录下载到你的系统。一旦成功，在下次执行MySQLInstaller或MySQLInstallerConsole时，将应用该目录。

C:\> **MySQLInstallerConsole update**

注意

自动目录更新GUI选项从Windows任务调度器执行此命令。

**upgrade [*product1*:*version*] [*product2*:*version*] [*...*]**

升级您系统中的一个或多个产品。语法选项包括。

\*

传入\*以将所有产品升级到最新版本，或传入特定产品。

!

传入！作为版本号，将MySQL产品升级到其最新版本。

-**-silent**

禁用确认提示。

C:\> **MySQLInstallerConsole upgrade \***

C:\> **MySQLInstallerConsole upgrade workbench:8.0.21**

C:\> **MySQLInstallerConsole upgrade workbench:!**

C:\> **MySQLInstallerConsole upgrade workbench:8.0.21 visual:1.2.9**

### 2.3.4 使用noinstall ZIP档案在Microsoft Windows上安装MySQL

从noinstall软件包安装的用户可以使用本节的说明来手动安装MySQL。从ZIP档案包安装MySQL的过程如下。

1. 提取主压缩包到所需的安装目录

可选的：如果你计划执行MySQL基准和测试套件，也要提取调试-测试存档。

2、创建一个选项文件

3、选择一个MySQL服务器类型

4、初始化MySQL

5、启动MySQL服务器

6、确保默认用户账户的安全

这一过程将在下面的章节中描述。

#### 2.3.4.1 提取安装档案

要手动安装MySQL，请执行以下步骤。

1、如果你从以前的版本升级，在开始升级过程之前，请参考第2.11.10节 "在Windows上升级MySQL"。

2、确保你是以具有管理员权限的用户身份登录的。

3、选择一个安装位置。传统上，MySQL服务器被安装在C:\mysql中。如果你不在C:\mysql安装MySQL，你必须在启动时或在选项文件中指定安装目录的路径。见第2.3.4.2节，"创建选项文件"。

注意

MySQL安装程序将MySQL安装在C:\Program Files\MySQL下。

4、使用你喜欢的文件压缩工具将安装档案提取到所选择的安装位置。有些工具可能会将存档解压到你所选择的安装位置的一个文件夹中。如果发生这种情况，你可以将子文件夹的内容移到选择的安装位置。

#### 2.3.4.2 创建一个选项文件

如果你需要在运行服务器时指定启动选项，你可以在命令行中指出它们，或者把它们放在一个选项文件中。对于每次服务器启动时都要使用的选项，你可能会发现使用一个选项文件来指定你的MySQL配置是最方便的。这在以下情况下尤其如此。

* 安装或数据目录位置与默认位置不同（C:\Program Files\MySQL\MySQL Server 8.0和C:\Program Files\MySQL\MySQL Server 8.0\data）。
* 你需要调整服务器设置，如内存、缓存或InnoDB配置信息。

当MySQL服务器在Windows上启动时，它会在几个地方寻找选项文件，如Windows目录、C:\和MySQL安装目录（关于完整的位置列表，见4.2.2.2节 "使用选项文件"）。Windows目录通常被命名为C:\WINDOWS之类的。你可以使用下面的命令从WINDIR环境变量的值中确定它的确切位置。

C:\> **echo %WINDIR%**

MySQL首先在my.ini文件中寻找每个位置的选项，然后在my.cnf文件中寻找。然而，为了避免混淆，最好只使用一个文件。如果你的电脑使用的是启动加载器，而C：不是启动驱动器，那么你唯一的选择就是使用my.ini文件。无论你使用哪一个选项文件，它都必须是一个纯文本文件。

注意

当使用MySQL安装程序安装MySQL服务器时，它会在默认位置创建my.ini，并且执行MySQL安装程序的用户被授予这个新的my.ini文件的全部权限。

换句话说，要确保MySQL Server用户有阅读my.ini文件的权限。

你也可以利用包括在你的MySQL发行版中的示例选项文件；见第5.1.2节，"服务器配置默认值"。

选项文件可以用任何文本编辑器创建和修改，如记事本。例如，如果MySQL安装在E:\mysql，数据目录在E:\mydata\data，你可以创建一个包含[mysqld]部分的选项文件来指定baseir和datadir的值。

[mysqld]

# set basedir to your installation path

basedir=E:/mysql

# set datadir to the location of your data directory

datadir=E:/mydata/data

Microsoft Windows的路径名称在选项文件中使用（正向）斜线而不是反斜线来指定。如果你使用反斜线，请将其加倍。

[mysqld]

# set basedir to your installation path

basedir=E:\\mysql

# set datadir to the location of your data directory

datadir=E:\\mydata\\data

在选项文件中使用反斜杠的规则在第4.2.2.2节 "使用选项文件 "中给出。

该ZIP档案不包括数据目录。要通过创建数据目录和填充mysql系统数据库中的表来初始化MySQL安装，请使用-initialize或-initialize-insecure初始化MySQL。有关其他信息，见第2.10.1节，"初始化数据目录"。

如果你想在不同的位置使用一个数据目录，你应该把数据目录的全部内容复制到新的位置。例如，如果你想使用E:\mydata作为数据目录，你必须做两件事。

1. 将整个数据目录及其所有内容从默认位置（例如C:\Program Files\MySQL\MySQL Server 8.0data）移动到E:\mydata。
2. 每次启动服务器时，使用--datadir选项来指定新的数据目录位置。

#### 2.3.4.3 选择一个MySQL服务器类型

下表显示了MySQL 8.0中用于Windows的可用服务器。

|  |  |
| --- | --- |
| **Binary** | **Description** |
| [**mysqld**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\programs.html#mysqld) | Optimized binary with named-pipe support |
| [**mysqld-debug**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\programs.html#mysqld) | Like [**mysqld**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\programs.html#mysqld), but compiled with full debugging and automatic memory allocation checking |

所有前面的二进制文件都针对现代英特尔处理器进行了优化，但应该可以在任何英特尔i386级或更高的处理器上工作。

一个发行版中的每一个服务器都支持相同的存储引擎集。SHOW ENGINES语句显示一个给定的服务器支持哪些引擎。

所有Windows MySQL 8.0服务器都支持数据库目录的符号链接。

MySQL支持所有Windows平台上的TCP/IP。如果你在启用named\_pipe系统变量的情况下启动服务器，Windows上的MySQL服务器也支持命名管道。有必要明确地启用这个变量，因为一些用户在使用命名管道时遇到了关闭MySQL服务器的问题。默认是使用TCP/IP，而不考虑平台，因为在许多Windows配置中，命名管道比TCP/IP慢。

#### 2.3.4.4 初始化数据目录

如果你使用noinstall软件包安装MySQL，则不包括数据目录。要初始化数据目录，请使用2.10.1节 "初始化数据目录 "的说明。

#### 2.3.4.5 首次启动服务器

本节给出了启动MySQL服务器的一般概述。以下各节提供了从命令行或作为Windows服务启动MySQL服务器的更具体信息。

这里的信息主要适用于你使用noinstall版本安装MySQL，或者你希望手动配置和测试MySQL，而不是使用MySQL安装程序。

这些章节中的例子假定MySQL安装在C:\Program Files\MySQL\MySQL Server 8.0的默认位置。如果你将MySQL安装在不同的位置，请调整例子中显示的路径名称。

客户端有两个选择。他们可以使用TCP/IP，或者如果服务器支持命名管道连接，他们可以使用命名管道。

如果服务器在启动时启用了shared\_memory系统变量，Windows版MySQL也支持共享内存连接。客户端可以通过使用--protocol=MEMORY选项通过共享内存进行连接。

关于运行哪种服务器二进制文件的信息，见2.3.4.3节 "选择MySQL服务器类型"。

测试最好在控制台窗口（或 "DOS窗口"）的命令提示符下进行。通过这种方式，你可以让服务器在窗口中显示状态信息，因为它们很容易看到。如果你的配置出了问题，这些信息使你更容易识别和解决任何问题。

注意

在启动MySQL之前，必须对数据库进行初始化。关于初始化过程的其他信息，见第2.10.1节，"初始化数据目录"。

要启动服务器，请输入这个命令。

C:\> **"C:\Program Files\MySQL\MySQL Server 8.0\bin\mysqld" --console**

对于包含InnoDB支持的服务器，当它启动时，你应该看到与下面类似的信息（路径名称和大小可能不同）。

InnoDB: The first specified datafile c:\ibdata\ibdata1 did not exist:

InnoDB: a new database to be created!

InnoDB: Setting file c:\ibdata\ibdata1 size to 209715200

InnoDB: Database physically writes the file full: wait...

InnoDB: Log file c:\iblogs\ib\_logfile0 did not exist: new to be created

InnoDB: Setting log file c:\iblogs\ib\_logfile0 size to 31457280

InnoDB: Log file c:\iblogs\ib\_logfile1 did not exist: new to be created

InnoDB: Setting log file c:\iblogs\ib\_logfile1 size to 31457280

InnoDB: Log file c:\iblogs\ib\_logfile2 did not exist: new to be created

InnoDB: Setting log file c:\iblogs\ib\_logfile2 size to 31457280

InnoDB: Doublewrite buffer not found: creating new

InnoDB: Doublewrite buffer created

InnoDB: creating foreign key constraint system tables

InnoDB: foreign key constraint system tables created

011024 10:58:25 InnoDB: Started

当服务器完成其启动序列时，你应该看到像这样的东西，这表明服务器已经准备好为客户连接服务。

mysqld: ready for connections

Version: '8.0.26' socket: '' port: 3306

服务器继续向控制台写入它产生的任何进一步的诊断输出。你可以打开一个新的控制台窗口，在其中运行客户程序。

如果你省略了--控制台选项，服务器会将诊断输出写入数据目录中的错误日志（默认为C:\Program Files\MySQL\MySQL Server 8.0\data）。错误日志是以.err为扩展名的文件，可以用--log-error选项来设置。

注意

在MySQL授予表中的初始根账户没有密码。在启动服务器后，你应该使用第2.10.4节 "确保初始MySQL账户的安全 "中的说明为其设置一个密码。

#### 2.3.4.6 从Windows命令行启动MySQL

可以从命令行手动启动MySQL服务器。这可以在任何版本的Windows上进行。

要从命令行启动mysqld服务器，你应该启动一个控制台窗口（或 "DOS窗口"）并输入这个命令。

C:\> **"C:\Program Files\MySQL\MySQL Server 8.0\bin\mysqld"**

mysqld的路径可能会有所不同，这取决于你系统中MySQL的安装位置。

你可以通过执行这个命令来停止MySQL服务器。

C:\> **"C:\Program Files\MySQL\MySQL Server 8.0\bin\mysqladmin" -u root shutdown**

注意

如果MySQL根用户账户有密码，你需要用-p选项调用mysqladmin，并在出现提示时提供该密码。

该命令调用MySQL管理工具mysqladmin连接到服务器并告诉它关闭。该命令以MySQL根用户的身份连接，这是MySQL授予系统中默认的管理账户。

注意

在MySQL授予系统中的用户完全独立于Microsoft Windows下的任何操作系统用户。

如果mysqld没有启动，检查错误日志，看看服务器是否在那里写了任何信息来表明问题的原因。默认情况下，错误日志位于C:\Program Files\MySQL\MySQL Server 8.0\data目录中。它是后缀为.err的文件，或者可以通过传入-log-error选项来指定。另外，你可以尝试用--console选项启动服务器；在这种情况下，服务器可能会在屏幕上显示一些有用的信息以帮助解决问题。

最后一个选项是用--standalone和--debug选项启动mysqld。在这种情况下，mysqld会写一个日志文件C:\mysqld.trace，它应该包含mysqld不启动的原因。参见第5.9.4节，"DBUG包"。

使用 mysqld --verbose --help 来显示 mysqld 支持的所有选项。

#### 2.3.4.7 为MySQL工具定制PATH

警告

当手工编辑你的系统PATH时，你必须非常小心；意外地删除或修改现有PATH值的任何部分都可能使你的系统出现故障甚至无法使用。

为了更容易调用MySQL程序，你可以把MySQL bin目录的路径名称添加到你的Windows系统PATH环境变量中。

* 在Windows桌面上，右击我的电脑图标，选择属性。
* 接下来从出现的系统属性菜单中选择高级标签，并点击环境变量按钮。
* 在系统变量下，选择路径，然后点击编辑按钮。编辑系统变量的对话框应该出现。
* 把你的光标放在出现在标有变量值空间的文本末尾。使用End键来确保你的光标位于这个空间中文本的最末端）。然后输入你的MySQL bin目录的完整路径名称（例如，C:\Program Files\MySQL\MySQL Server 8.0\bin)

注意

必须有一个分号把这个路径和这个字段中的任何值分开。

通过单击 "确定 "来解散这个对话，并依次解散每个对话，直到所有被打开的对话都被解散。新的PATH值现在应该对你打开的任何新的命令外壳可用，允许你在DOS提示符下从系统的任何目录中输入MySQL可执行程序的名称来调用它，而不必提供路径。这包括服务器、mysql客户端和所有MySQL命令行工具，如mysqladmin和mysqldump。

如果你在同一台机器上运行多个MySQL服务器，你不应该将MySQL bin目录添加到你的Windows PATH中。

#### 2.3.4.8 作为Windows服务启动MySQL

在Windows上，运行MySQL的推荐方法是将其安装为Windows服务，这样MySQL就会在Windows启动和停止时自动启动和停止。作为服务安装的MySQL服务器也可以使用NET命令从命令行控制，或者使用图形化的Services工具。一般来说，要把MySQL安装成Windows服务，你应该用一个有管理员权限的账户来登录。

服务工具（Windows服务控制管理器）可以在Windows控制面板中找到。为了避免冲突，建议在从命令行进行服务器安装或移除操作时，关闭服务工具。

**安装服务**

在将MySQL安装为Windows服务之前，如果当前服务器正在运行，你应该首先通过使用以下命令停止它。

C:\> **"C:\Program Files\MySQL\MySQL Server 8.0\bin\mysqladmin"**

**-u root shutdown**

注意

如果MySQL根用户账户有密码，你需要用-p选项调用mysqladmin，并在出现提示时提供该密码。

该命令调用MySQL管理工具mysqladmin连接到服务器并告诉它关闭。该命令以MySQL根用户的身份连接，这是MySQL授予系统中默认的管理账户。

注意

MySQL授予系统中的用户完全独立于Windows下的任何操作系统用户。

使用此命令将服务器安装为一个服务。

C:\> **"C:\Program Files\MySQL\MySQL Server 8.0\bin\mysqld" --install**

服务安装命令并不启动服务器。本节后面将给出相关说明。

为了更容易调用MySQL程序，你可以把MySQL bin目录的路径名称添加到你的Windows系统PATH环境变量中。

* 在Windows桌面上，右击我的电脑图标，选择属性。
* 接下来从出现的系统属性菜单中选择高级标签，并点击环境变量按钮。
* 在系统变量下，选择路径，然后点击编辑按钮。编辑系统变量的对话框应该出现。
* 把你的光标放在出现在标有变量值空间的文本末尾。使用End键来确保你的光标位于这个空间中文本的最末端）。然后输入你的MySQL bin目录的完整路径名称（例如，C:\Program Files\MySQL\MySQL Server 8.0\bin），并且应该有一个分号将此路径与此字段中的任何值分开。通过单击 "确定 "来解散这个对话，并依次解散每个对话，直到所有被打开的对话都被解散。现在你应该能够从系统的任何目录中通过在DOS提示符下输入MySQL可执行程序的名称来调用它，而不必提供路径。这包括服务器、mysql客户端和所有MySQL命令行工具，如mysqladmin和mysqldump。

如果你在同一台机器上运行多个MySQL服务器，你不应该将MySQL bin目录添加到你的Windows PATH。

警告

在手工编辑你的系统PATH时，你必须非常小心；意外地删除或修改现有PATH值的任何部分都可能使你的系统出现故障甚至无法使用。

在安装服务时，可以使用下列附加参数。

* 你可以在 --install 选项后面立即指定一个服务名称。默认的服务名称是MySQL。
* 如果给出了一个服务名称，可以在它后面加上一个选项。按照惯例，这应该是--defaults-file=file\_name，以指定一个选项文件的名称，服务器启动时应该从该文件中读取选项。

可以使用除-defaults-file以外的单个选项，但不鼓励使用。--defaults-file更灵活，因为它使你能够为服务器指定多个启动选项，把它们放在指定的选项文件中。

* 你也可以在服务名称后面指定一个-local-service选项。这将导致服务器使用具有有限系统权限的LocalService Windows账户运行。如果在服务名称后面同时给出-defaults-file和-local-service，它们可以是任何顺序。

对于作为Windows服务安装的MySQL服务器，以下规则决定服务器使用的服务名称和选项文件。

* 如果服务-安装命令在-install选项后面没有指定服务名称或默认服务名称（MySQL），则服务器使用MySQL的服务名称，并从标准选项文件的[mysqld]组中读取选项。
* 如果服务-安装命令在--安装选项后面指定了MySQL以外的服务名称，服务器就会使用该服务名称。它从标准选项文件中的[mysqld]组和与服务同名的组读取选项。这使你能够将[mysqld]组用于所有MySQL服务应使用的选项，并将带有服务名称的选项组用于以该服务名称安装的服务器。
* 如果服务安装命令在服务名称后面指定了一个--defaults-file选项，服务器读取选项的方式与前一项描述的相同，只是它只从命名的文件中读取选项，而忽略标准选项文件。

作为一个更复杂的例子，考虑下面的命令。

C:\> **"C:\Program Files\MySQL\MySQL Server 8.0\bin\mysqld"**

**--install MySQL --defaults-file=C:\my-opts.cnf**

这里，默认的服务名称（MySQL）是在--install选项之后给出的。如果没有给出--defaults-file选项，这个命令的效果是使服务器从标准选项文件中读取[mysqld]组。然而，由于--defaults-file选项的存在，服务器会从[mysqld]选项组中读取选项，并且只从命名的文件中读取。

注意

在Windows上，如果服务器是用--defaults-file和--install选项启动的，--install必须在前。否则，mysqld.exe会尝试启动MySQL服务器。

在启动MySQL服务之前，你也可以在Windows服务工具中指定选项作为启动参数。

最后，在尝试启动MySQL服务之前，确保要运行该服务的操作系统用户的用户变量%TEMP%和%TMP%（还有%TMPDIR%，如果它曾经被设置过）指向该用户有写入权限的文件夹。运行MySQL服务的默认用户是LocalSystem，其%TEMP%和%TMP%的默认值是C:\Windows\Temp，这个目录LocalSystem默认有写入权限。但是，如果该默认设置有任何变化（例如，运行服务的用户或所提及的用户变量有变化，或者使用了--tmpdir选项将临时目录放在其他地方），MySQL服务可能无法运行，因为临时目录的写入权限没有被授予适当的用户。

启动服务

在MySQL服务器实例被安装为服务后，每当Windows启动时，Windows会自动启动该服务。该服务也可以从服务工具中立即启动，或者通过使用sc start mysqld\_service\_name或NET START mysqld\_service\_name命令启动。SC和NET命令是不区分大小写的。

当作为服务运行时，mysqld不能访问控制台窗口，所以不能在那里看到任何消息。如果mysqld不启动，检查错误日志，看看服务器是否在那里写了任何信息以表明问题的原因。错误日志位于MySQL数据目录中（例如，C:\Program Files\MySQL\MySQL Server 8.0\data）。它是后缀为.err的文件。

当MySQL服务器被安装为一个服务，并且该服务正在运行时，当Windows关闭时，Windows会自动停止该服务。也可以使用服务工具、sc stop mysqld\_service\_name 命令、NET START mysqld\_service\_name 命令或 mysqladmin shutdown 命令手动停止服务器。

如果你不希望服务在启动过程中自动启动，你也可以选择将服务器安装为一个手动服务。要做到这一点，使用 --install-manual 选项而不是 --install 选项。

C:\> **"C:\Program Files\MySQL\MySQL Server 8.0\bin\mysqld" --install-manual**

移除服務

要删除作为服务安装的服务器，首先通过执行SC STOP mysqld\_service\_name或NET STOP mysqld\_service\_name停止它，如果它正在运行。然后使用SC DELETE mysqld\_service\_name来删除它。

C:\> **SC DELETE mysql**

或者，使用 mysqld --remove 选项来删除该服务。

C:\> **"C:\Program Files\MySQL\MySQL Server 8.0\bin\mysqld" --remove**

如果mysqld没有作为服务运行，你可以从命令行启动它。有关说明，见第2.3.4.6节，"从Windows命令行启动MySQL"。

如果你在安装过程中遇到困难，请参阅第2.3.5节，"Microsoft Windows MySQL服务器安装的故障排除"。

关于停止或删除Windows服务的更多信息，见第5.8.2.2节，"作为Windows服务启动多个MySQL实例"。

#### 2.3.4.9 测试MySQL安装

你可以通过执行以下任何一个命令来测试MySQL服务器是否在工作。

C:\> **"C:\Program Files\MySQL\MySQL Server 8.0\bin\mysqlshow"**

C:\> **"C:\Program Files\MySQL\MySQL Server 8.0\bin\mysqlshow" -u root mysql**

C:\> **"C:\Program Files\MySQL\MySQL Server 8.0\bin\mysqladmin" version status proc**

C:\> **"C:\Program Files\MySQL\MySQL Server 8.0\bin\mysql" test**

如果mysqld对来自客户端程序的TCP/IP连接响应缓慢，可能是你的DNS出现了问题。在这种情况下，在启用 skip\_name\_resolve 系统变量的情况下启动 mysqld，并在 MySQL 授予表的 Host 列中只使用 localhost 和 IP 地址。(要确保存在一个指定IP地址的账户，否则你可能无法连接)。

你可以通过指定--管道或--协议=PIPE选项，或指定......作为主机名，强迫MySQL客户端使用命名的管道连接而不是TCP/IP。(句号)作为主机名。如果你不想使用默认的管道名称，使用--socket选项来指定管道的名称。

如果你已经为根账户设置了密码，删除了匿名账户，或创建了一个新的用户账户，那么要连接到MySQL服务器，你必须使用适当的-u和-p选项与前面显示的命令。参见第4.2.4节，"使用命令选项连接到MySQL服务器"。

关于mysqlshow的更多信息，参见第4.5.7节 "mysqlshow-显示数据库、表和列的信息"。

### 2.3.5 Windows系统MySQL服务器安装的故障排除

当首次安装和运行MySQL时，你可能会遇到某些错误，使MySQL服务器无法启动。本节帮助你诊断和纠正这些错误中的一些。

在排除服务器问题的故障时，你的第一个资源是错误日志。MySQL服务器使用错误日志来记录与阻止服务器启动的错误有关的信息。错误日志位于你的my.ini文件中指定的数据目录中。默认的数据目录位置是C:\Program Files\MySQL\MySQL Server 8.0\data，或者在Windows 7和Windows Server 2008上是C:\ProgramData\Mysql。C:\ProgramData目录在默认情况下是隐藏的。你需要改变你的文件夹选项才能看到该目录和内容。关于错误日志和理解其内容的更多信息，请参见第5.4.2节，"错误日志"。

关于可能的错误的信息，也可以参考MySQL服务启动时显示的控制台信息。在将mysqld安装为服务后，从命令行中使用SC START mysqld\_service\_name或NET START mysqld\_service\_name命令，查看有关将MySQL服务器作为服务启动的任何错误信息。参见第2.3.4.8节，"将MySQL作为Windows服务启动"。

下面的例子显示了你在安装MySQL和首次启动服务器时可能遇到的其他常见错误信息。

* 如果MySQL服务器找不到mysql特权数据库或其他关键文件，它会显示这些消息。

System error 1067 has occurred.

Fatal error: Can't open and lock privilege tables:

Table 'mysql.user' doesn't exist

当MySQL基础目录或数据目录安装在与默认位置（分别为C:\Program Files\MySQL Server 8.0和C:\Program Files\MySQL\MySQL Server 8.0\data）不同的位置时，这些消息经常发生。

当MySQL升级并安装到一个新的位置，但配置文件没有更新以反映新的位置时，这种情况可能发生。此外，新旧配置文件可能发生冲突。请确保在升级MySQL时删除或重命名任何旧的配置文件。

如果你已将MySQL安装到C:\Program Files\MySQL\MySQL Server 8.0以外的目录，请确保MySQL服务器通过使用配置（my.ini）文件知道这一点。把my.ini文件放在你的Windows目录下，通常是C:\WINDOWS。要根据WINDIR环境变量的值确定它的确切位置，请从命令提示符中发出以下命令。

C:\> **echo %WINDIR%**

你可以用任何文本编辑器，如记事本，来创建或修改选项文件。例如，如果MySQL安装在E:\mysql，数据目录是D:\MySQLdata，你可以创建选项文件，并设置一个[mysqld]部分来指定baseir和datadir选项的值。

[mysqld]

# set basedir to your installation path

basedir=E:/mysql

# set datadir to the location of your data directory

datadir=D:/MySQLdata

Microsoft Windows的路径名称在选项文件中使用（向前）斜线而不是反斜线来指定。如果你使用反斜线，请将其加倍。

[mysqld]

# set basedir to your installation path

basedir=C:\\Program Files\\MySQL\\MySQL Server 8.0

# set datadir to the location of your data directory

datadir=D:\\MySQLdata

在选项文件值中使用反斜杠的规则在第4.2.2.2节 "使用选项文件 "中给出。

如果你改变了MySQL配置文件中的datadir值，你必须在重新启动MySQL服务器之前移动现有MySQL数据目录的内容。

见第2.3.4.2节，"创建选项文件"。

如果你在没有首先停止和删除现有的MySQL服务的情况下重新安装或升级MySQL，并使用MySQL安装程序安装MySQL，你可能看到这个错误。

Error: Cannot create Windows service for MySql. Error: 0

这发生在配置向导试图安装服务并发现一个具有相同名称的现有服务。

解决这个问题的一个办法是在使用配置向导时选择一个mysql以外的服务名称。这可以使新的服务被正确安装，但留下过时的服务。虽然这没有什么危害，但最好是删除不再使用的旧服务。

要永久地删除旧的mysql服务，以具有管理权限的用户在命令行上执行以下命令。

C:\> **SC DELETE mysql**

[SC] DeleteService SUCCESS

如果SC工具对你的Windows版本不可用，请从http://www.microsoft.com/windows2000/techinfo/reskit/tools/existing/delsrv-o.asp，并使用delsrv mysql语法下载delsrv工具。

### 2.3.6 Windows安装后程序

有一些GUI工具可以执行本节中描述的大多数任务，包括

* MySQL安装程序。用于安装和升级MySQL产品。
* MySQL Workbench。管理MySQL服务器并编辑SQL语句。

如果有必要，初始化数据目录并创建MySQL授予表。由MySQL Installer执行的Windows安装操作会自动初始化数据目录。对于从ZIP存档包进行的安装，按照第2.10.1节 "初始化数据目录 "所述，初始化数据目录。

关于密码，如果你使用MySQL安装程序安装MySQL，你可能已经为初始根账户分配了一个密码。否则，请使用第2.10.4节 "确保初始MySQL账户的安全 "中给出的密码分配程序。

在分配密码之前，你可能想尝试运行一些客户端程序，以确保你能连接到服务器，并且它能正常运行。确保服务器正在运行（见2.3.4.5节，"首次启动服务器"）。你也可以设置一个MySQL服务，在Windows启动时自动运行（见第2.3.4.8节，"作为Windows服务启动MySQL"）。

这些说明假定你的当前位置是MySQL安装目录，并且它有一个bin子目录，包含这里使用的MySQL程序。如果不是这样，请相应调整命令路径名称。

如果你使用MySQL安装程序安装MySQL（见第2.3.3节，"Windows的MySQL安装程序"），默认安装目录是C:\Program Files\MySQL\MySQL Server 8.0。

C:\> **cd "C:\Program Files\MySQL\MySQL Server 8.0"**

從ZIP檔案中安裝的一個常見的安裝位置是C:\mysql。

C:\> **cd C:\mysql**

或者，将bin目录添加到你的PATH环境变量设置中。这可以使你的命令解释器正确地找到MySQL程序，这样你就可以只输入程序的名称而不是路径名称来运行程序。见第2.3.4.7节，"为MySQL工具定制PATH"。

在服务器运行的情况下，发出下列命令以验证你可以从服务器中检索信息。输出应该与这里显示的类似。

使用mysqlshow来查看存在哪些数据库。

C:\> **bin\mysqlshow**

+--------------------+

| Databases |

+--------------------+

| information\_schema |

| mysql |

| performance\_schema |

| sys |

+--------------------+

已安装的数据库列表可能有所不同，但总是至少包括 mysql 和 information\_schema。

如果正确的MySQL账户不存在，前面的命令（以及其他MySQL程序的命令，如mysql）可能无法工作。例如，程序可能会出错而失败，或者你可能无法查看所有数据库。如果你使用MySQL安装程序安装MySQL，根用户会用你提供的密码自动创建。在这种情况下，你应该使用-u root和-p选项。(如果你已经确保了初始MySQL账户的安全，你必须使用这些选项)。使用-p，客户端程序会提示根密码。比如说

C:\> **bin\mysqlshow -u root -p**

Enter password: ***(enter root password here)***

+--------------------+

| Databases |

+--------------------+

| information\_schema |

| mysql |

| performance\_schema |

| sys |

+--------------------+

如果指定数据库名，那么mysqlshow 会列出该数据库中的所有数据表

C:\> **bin\mysqlshow mysql**

Database: mysql

+---------------------------+

| Tables |

+---------------------------+

| columns\_priv |

| component |

| db |

| default\_roles |

| engine\_cost |

| func |

| general\_log |

| global\_grants |

| gtid\_executed |

| help\_category |

| help\_keyword |

| help\_relation |

| help\_topic |

| innodb\_index\_stats |

| innodb\_table\_stats |

| ndb\_binlog\_index |

| password\_history |

| plugin |

| procs\_priv |

| proxies\_priv |

| role\_edges |

| server\_cost |

| servers |

| slave\_master\_info |

| slave\_relay\_log\_info |

| slave\_worker\_info |

| slow\_log |

| tables\_priv |

| time\_zone |

| time\_zone\_leap\_second |

| time\_zone\_name |

| time\_zone\_transition |

| time\_zone\_transition\_type |

| user |

+---------------------------+

使用mysql命令去查询表中记录

C:\> **bin\mysql -e "SELECT User, Host, plugin FROM mysql.user" mysql**

+------+-----------+-----------------------+

| User | Host | plugin |

+------+-----------+-----------------------+

| root | localhost | caching\_sha2\_password |

关于mysql和mysqlshow的更多信息，见第4.5.1节 "mysql--MySQL命令行客户端 "和第4.5.7节 "mysqlshow--显示数据库、表和列的信息"。

### 2.3.7 Windows平台的限制

以下限制适用于在Windows平台上使用MySQL。

* 进程内存

在Windows 32位平台上，默认情况下不可能在单个进程中使用超过2GB的RAM，包括MySQL。这是因为Windows 32位的物理地址限制是4GB，而Windows内部的默认设置是在内核（2GB）和用户/应用程序（2GB）之间分割虚拟地址空间。

一些版本的Windows有一个启动时间设置，通过减少内核应用来实现更大的应用。另外，要使用超过2GB的空间，请使用64位版本的Windows。

* 文件系统别名

当使用MyISAM表时，你不能在Windows内使用别名链接到另一个卷上的数据文件，然后再链接到MySQL的主数据目录位置。

这种设施经常被用来将数据和索引文件转移到RAID或其他快速解决方案中。

* 端口的数量有限

Windows系统有大约4000个端口可供客户端连接，在一个端口上的连接关闭后，需要两到四分钟才能重新使用该端口。在客户端高速连接和断开服务器的情况下，有可能在关闭的端口再次可用之前，所有可用的端口都被用完。如果发生这种情况，即使MySQL服务器正在运行，也会出现无响应的情况。端口可能也被机器上运行的其他应用程序使用，在这种情况下，MySQL可用的端口数量会减少。

关于这个问题的更多信息，见https://support.microsoft.com/kb/196271。

* DATA DIRECTORY和INDEX DIRECTORY

CREATE TABLE语句的DATA DIRECTORY子句在Windows上只支持InnoDB表，如第15.6.1.2节 "在外部创建表 "中所述。对于MyISAM和其他存储引擎，CREATE TABLE的DATA DIRECTORY和INDEX DIRECTORY子句在Windows和其他任何有无功能的realpath()调用的平台上被忽略。

* DROP DATABASE

你不能删除一个正在被其他会话使用的数据库。

* 不区分大小写的名字

文件名在Windows上不区分大小写，所以MySQL数据库和表名在Windows上也不区分大小写。唯一的限制是数据库和表名必须在给定的语句中使用相同的大小写来指定。见第9.2.3节，"标识符大小写敏感性"。

* 目录和文件名

在Windows上，MySQL服务器只支持与当前ANSI代码页兼容的目录和文件名。例如，下面的日文目录名在西方地区设置中不起作用（代码页1252）。

datadir="C:/私たちのプロジェクトのデータ"

同样的限制也适用于SQL语句中提到的目录和文件名，如LOAD DATA中的数据文件路径名。

* The \ path name separator character

Windows中的路径名组件由 \字符分隔，这也是MySQL中的转义字符。如果你使用LOAD DATA或SELECT ... INTO OUTFILE，请使用Unix风格的文件名与/字符。

mysql> **LOAD DATA INFILE 'C:/tmp/skr.txt' INTO TABLE skr;**

mysql> **SELECT \* INTO OUTFILE 'C:/tmp/skr.txt' FROM skr;**

或者，你必须加倍使用 \字符。

mysql> **LOAD DATA INFILE 'C:\\tmp\\skr.txt' INTO TABLE skr;**

mysql> **SELECT \* INTO OUTFILE 'C:\\tmp\\skr.txt' FROM skr;**

* 关于管道的问题

管道在Windows命令行提示下不能可靠地工作。如果管道中包含字符^Z / CHAR(24)，Windows会认为它遇到了文件结束，并中止程序。

这主要是在你试图应用二进制日志时出现的问题，如下所示。

C:\> **mysqlbinlog *binary\_log\_file* | mysql --user=root**

如果你在应用日志时遇到问题，并且怀疑是因为^Z / CHAR(24)字符，你可以使用下面的解决方法。

C:\> **mysqlbinlog *binary\_log\_file* --result-file=/tmp/bin.sql**

C:\> **mysql --user=root --execute "source /tmp/bin.sql"**

后一个命令也可以用来可靠地读取任何可能包含二进制数据的SQL文件。

## 2.4 在macOS上安装MySQL

关于MySQL服务器支持的macOS版本的列表，请参见https://www.mysql.com/support/supportedplatforms/database.html。

用于macOS的MySQL有多种不同的形式。

* 原生软件包安装程序，它使用原生macOS安装程序（DMG）引导你完成MySQL的安装。更多信息，见第2.4.2节，"使用本地包在macOS上安装MySQL"。你可以在macOS上使用软件包安装程序。你用来执行安装的用户必须有管理员权限。
* 压缩的TAR档案，它使用Unix的tar和gzip命令打包的文件。要使用这种方法，你需要打开一个终端窗口。使用这种方法你不需要管理员权限；你可以用这种方法在任何地方安装MySQL服务器。关于使用这种方法的更多信息，你可以使用使用tarball的通用说明，第2.2节，"在Unix/Linux上使用通用二进制文件安装MySQL"。

除了核心安装，软件包安装程序还包括第2.4.3节 "安装和使用MySQL启动守护程序 "和第2.4.4节 "安装和使用MySQL首选项窗格"，以简化对安装的管理。

关于在macOS上使用MySQL的其他信息，见第2.4.1节 "在macOS上安装MySQL的一般注意事项"。

### 2.4.1 在macOS上安装MySQL的一般注意事项

你应该牢记以下问题和注意事项。

* 其他MySQL安装。安装过程不识别由软件包管理器（如Homebrew）安装的MySQL。安装和升级过程是针对我们提供的MySQL包。如果存在其他安装，那么考虑在执行此安装程序之前停止它们，以避免端口冲突。
* Homebrew。例如，如果你使用Homebrew将MySQL服务器安装到它的默认位置，那么MySQL安装程序会安装到一个不同的位置，并且不会从Homebrew升级版本。在这种情况下，你最终会有多个MySQL安装，默认情况下，它们试图使用相同的端口。在运行此安装程序之前，停止其他MySQL服务器实例，例如执行brew services stop mysql来停止Homebrew的MySQL服务。
* Launchd。安装了一个改变MySQL配置选项的Launchd守护程序。如果需要，可以考虑编辑它，更多信息见下面的文档。另外，macOS 10.10删除了对启动项的支持，转而使用launchd守护进程。macOS系统偏好下可选的MySQL偏好窗格使用 launchd 守护程序。
* 用户。你可能需要（或想要）创建一个特定的mysql用户来拥有MySQL目录和数据。你可以通过目录工具做到这一点，而且mysql用户应该已经存在。对于在单用户模式下使用，\_mysql的条目（注意下划线前缀）应该已经存在于系统/etc/passwd文件中。
* 数据。因为MySQL包安装程序将MySQL内容安装到一个版本和平台特定的目录中，你可以用它来升级和迁移你的数据库。你需要把数据目录从旧版本复制到新版本，或者指定一个替代的datadir值来设置数据目录的位置。默认情况下，MySQL目录被安装在/usr/local/下。
* 别名。你可能想在你的shell的资源文件中添加别名，以使它更容易从命令行访问常用的程序，如mysql和mysqladmin。bash的语法是

alias mysql=/usr/local/mysql/bin/mysql

alias mysqladmin=/usr/local/mysql/bin/mysqladmin

对于tcsh，使用

alias mysql /usr/local/mysql/bin/mysql

alias mysqladmin /usr/local/mysql/bin/mysqladmin

甚至更好的是，将/usr/local/mysql/bin添加到你的PATH环境变量中。你可以通过为你的shell修改适当的启动文件来做到这一点。更多信息，见第4.2.1节，"调用MySQL程序"。

* 移除。在你从以前的安装中复制了MySQL数据库文件并成功地启动了新的服务器之后，你应该考虑删除旧的安装文件以节省磁盘空间。此外，你还应该删除位于/Library/Receipts/mysql-VERSION.pkg的旧版本的软件包收据目录。

遗留问题。在OS X 10.7之前，MySQL服务器是与OS X Server捆绑在一起的。

### 2.4.2 使用本地软件包在macOS上安装MySQL

该软件包位于一个磁盘镜像（.dmg）文件中，你首先需要在Finder中双击其图标来挂载。然后，它应该加载镜像并显示其内容。

注意

在继续安装之前，确保通过使用MySQL管理器应用程序（在macOS服务器上）、偏好面板或命令行上的mysqladmin shutdown停止所有正在运行的MySQL服务器实例。

要使用软件包安装程序安装MySQL。

1、下载包含MySQL包安装程序的磁盘镜像（.dmg）文件（社区版本可在此获得）。双击该文件以加载磁盘镜像并查看其内容。

双击磁盘上的MySQL安装程序包。它是根据你所下载的MySQL版本来命名的。例如，对于MySQL服务器8.0.26，它可能被命名为mysql-8.0.26-osx-10.13-x86\_64.pkg。

2、最初的向导介绍屏幕引用要安装的MySQL服务器版本。点击继续，开始安装。

MySQL社区版显示了相关GNU通用公共许可证的副本。点击 "继续"，然后点击 "同意"，继续。

3、在安装类型页面，你可以点击安装，使用所有默认值执行安装向导；点击自定义，改变要安装的组件（MySQL服务器、MySQL测试、首选项窗格、Launchd支持--除了MySQL测试，其他都是默认启用的）。

注意

尽管 "改变安装位置 "选项是可见的，但安装位置不能被改变。

图2.13 MySQL软件包安装器向导。安装类型

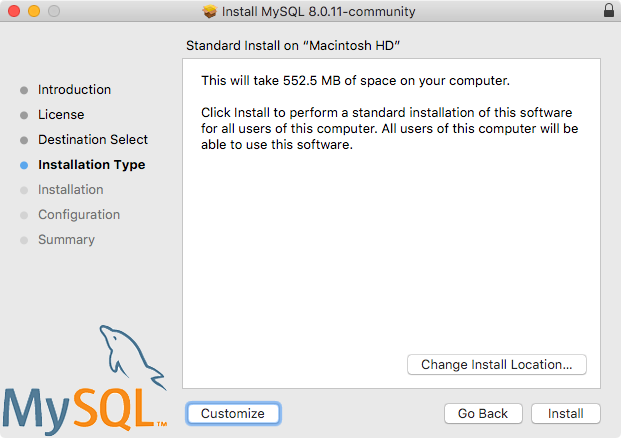
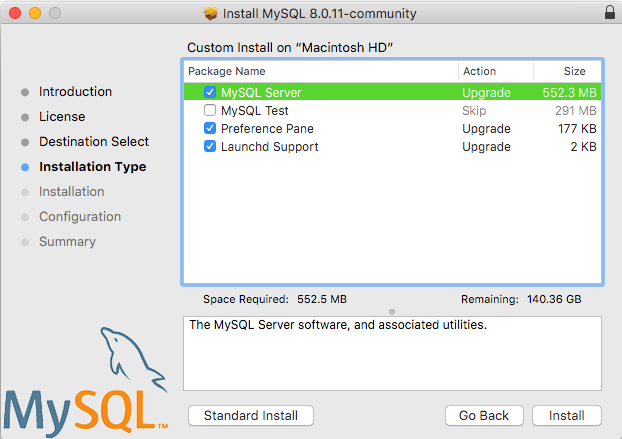


图2.14 MySQL软件包安装向导。自定义



4、点击安装来安装MySQL服务器。如果升级当前的MySQL Server安装，安装过程到此结束，否则请按照向导的附加配置步骤进行新的MySQL Server安装。

5、在新的MySQL服务器安装成功后，通过选择密码的默认加密类型完成配置步骤，定义根密码，并在启动时启用（或禁用）MySQL服务器。

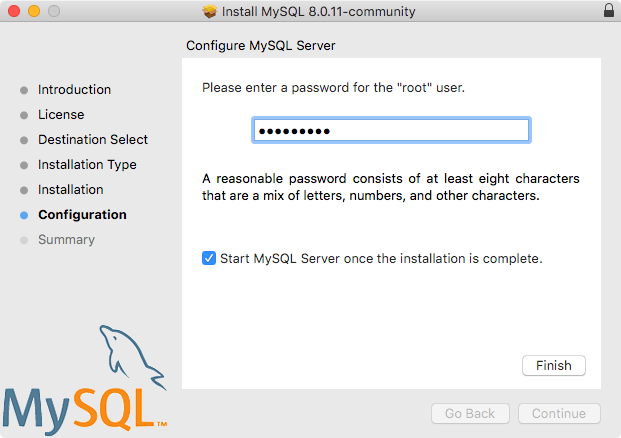
6、默认的MySQL 8.0密码机制是caching\_sha2\_password（强），这个步骤允许你将其改为mysql\_native\_password（遗留）。

图2.15 MySQL包安装器向导。选择一个密码加密类型



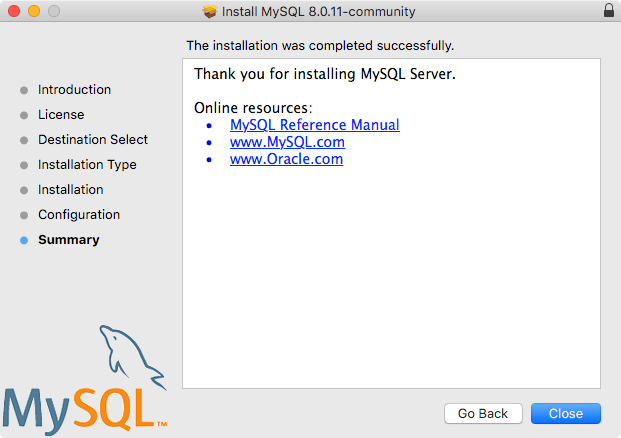
7、为根用户定义一个密码，同时切换MySQL服务器是否应在配置步骤完成后启动。

图2.16 MySQL包安装器向导。定义根密码



8、摘要是最后一步，并参考成功和完整的MySQL服务器安装。关闭该向导。

图2.17 MySQL软件包安装器向导。总结



MySQL服务器现在已经安装完毕。如果你选择不启动MySQL，那么从命令行使用launchctl，或者使用MySQL首选项窗格点击 "启动 "来启动MySQL。有关其他信息，请参见第2.4.3节 "安装和使用MySQL启动守护程序"，以及第2.4.4节 "安装和使用MySQL首选项窗格"。使用MySQL首选项窗格或launchd来配置MySQL在启动时自动启动。

当使用软件包安装程序进行安装时，文件被安装到与安装版本和平台名称相匹配的/usr/local中的一个目录。例如，安装程序文件mysql-8.0.26-osx10.15-x86\_64.dmg将MySQL安装到/usr/local/mysql-8.0.26-osx10.15-x86\_64/，其符号链接为/usr/local/mysql。下表显示了这个MySQL安装目录的布局。

注意

macOS安装过程不创建也不安装my.cnf MySQL配置文件样本。

表2.7 macOS上的MySQL安装布局

|  |  |
| --- | --- |
| **Directory** | **Contents of Directory** |
| bin | [**mysqld**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\programs.html#mysqld) server, client and utility programs |
| data | Log files, databases, where /usr/local/mysql/data/mysqld.local.err is the default error log |
| docs | Helper documents, like the Release Notes and build information |
| include | Include (header) files |
| lib | Libraries |
| man | Unix manual pages |
| mysql-test | MySQL test suite ('MySQL Test' is disabled by default during the installation process when using the installer package (DMG)) |
| share | Miscellaneous support files, including error messages, dictionary.txt, and rewriter SQL |
| support-files | Support scripts, such as mysqld\_multi.server, mysql.server, and mysql-log-rotate. |
| /tmp/mysql.sock | Location of the MySQL Unix socket |

### 2.4.3 安装和使用MySQL启动守护程序

macOS使用启动守护程序来自动启动、停止和管理进程和应用程序，如MySQL。

默认情况下，macOS上的安装包（DMG）安装了一个名为/Library/LaunchDaemons/com.oracle.oss.mysql.mysqld.plist的launchd文件，其中包含一个类似的plist定义。

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE plist PUBLIC "-//Apple Computer//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd">

<plist version="1.0">

<dict>

<key>Label</key> <string>com.oracle.oss.mysql.mysqld</string>

<key>ProcessType</key> <string>Interactive</string>

<key>Disabled</key> <false/>

<key>RunAtLoad</key> <true/>

<key>KeepAlive</key> <true/>

<key>SessionCreate</key> <true/>

<key>LaunchOnlyOnce</key> <false/>

<key>UserName</key> <string>\_mysql</string>

<key>GroupName</key> <string>\_mysql</string>

<key>ExitTimeOut</key> <integer>600</integer>

<key>Program</key> <string>/usr/local/mysql/bin/mysqld</string>

<key>ProgramArguments</key>

<array>

<string>/usr/local/mysql/bin/mysqld</string>

<string>--user=\_mysql</string>

<string>--basedir=/usr/local/mysql</string>

<string>--datadir=/usr/local/mysql/data</string>

<string>--plugin-dir=/usr/local/mysql/lib/plugin</string>

<string>--log-error=/usr/local/mysql/data/mysqld.local.err</string>

<string>--pid-file=/usr/local/mysql/data/mysqld.local.pid</string>

<string>--keyring-file-data=/usr/local/mysql/keyring/keyring</string>

<string>--early-plugin-load=keyring\_file=keyring\_file.so</string>

</array>

<key>WorkingDirectory</key> <string>/usr/local/mysql</string>

</dict>

</plist>

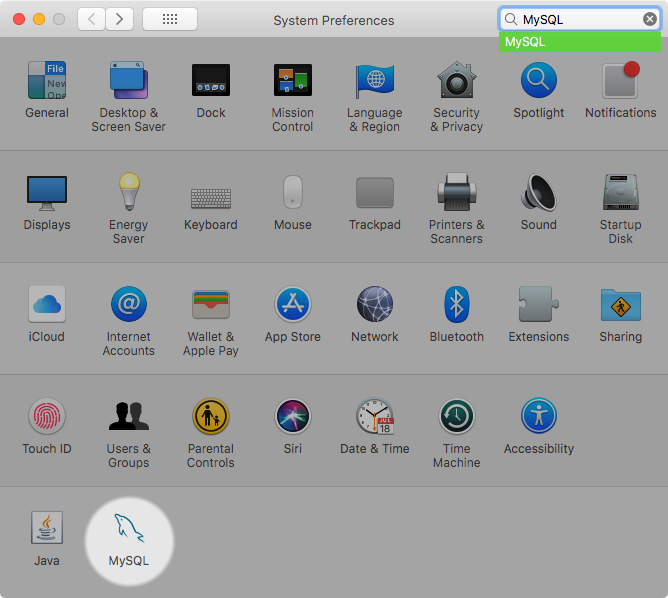
注意事项

一些用户报告说，添加plist DOCTYPE声明会导致 launchd操作失败，尽管它通过了lint检查。我们怀疑这是一个复制-粘贴的错误。包含上述片段的文件的md5校验和是d925f05f6d1b6ee5ce5451b596d6baed。

要启用launchd服务，你可以这样做。

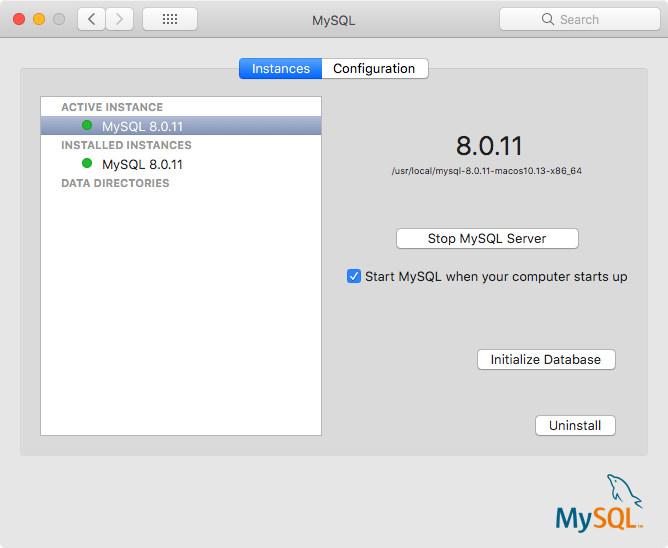
* 打开macOS系统偏好设置，选择MySQL偏好设置面板，然后执行启动MySQL服务器。

图2.18 MySQL首选项面板。位置



实例页面包括一个启动或停止MySQL的选项，而初始化数据库会重新创建数据/目录。Uninstall卸载MySQL服务器，也可以选择卸载MySQL首选项面板和启动d信息。

图2.19 MySQL首选项面板：实例



* 或者，手动加载 launchd 文件。

shell> cd /Library/LaunchDaemons

shell> sudo launchctl load -F com.oracle.oss.mysql.mysqld.plist

* 要配置MySQL在启动时自动启动，你可以。

shell> sudo launchctl load -w com.oracle.oss.mysql.mysqld.plist

注意

当升级MySQL服务器时，launchd安装过程会删除与MySQL服务器5.7.7及以下版本一起安装的旧启动项。

升级也会替换你现有的名为com.oracle.os.mysql.mysqld.plist的launchd文件。

额外的 launchd 相关信息。

* plist 条目覆盖了 my.cnf 条目，因为它们是作为命令行参数传递的。有关传入程序选项的其他信息，请参见第4.2.2节 "指定程序选项"。
* ProgramArguments部分定义了传入程序的命令行选项，在本例中是mysqld二进制。
* 默认的plist定义是在考虑到不太复杂的使用情况下编写的。对于更复杂的设置，你可能想删除一些参数，而依靠MySQL配置文件，如my.cnf。
* 如果你编辑了plist文件，那么在重新安装或升级MySQL时，不要选中安装程序选项。否则，你编辑的plist文件会被覆盖，所有编辑的内容都会丢失。

因为默认的plist定义定义了几个ProgramArguments，你可以删除这些参数中的大部分，而依靠你的my.cnf MySQL配置文件来定义它们。比如说。

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE plist PUBLIC "-//Apple Computer//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd">

<plist version="1.0">

<dict>

<key>Label</key> <string>com.oracle.oss.mysql.mysqld</string>

<key>ProcessType</key> <string>Interactive</string>

<key>Disabled</key> <false/>

<key>RunAtLoad</key> <true/>

<key>KeepAlive</key> <true/>

<key>SessionCreate</key> <true/>

<key>LaunchOnlyOnce</key> <false/>

<key>UserName</key> <string>\_mysql</string>

<key>GroupName</key> <string>\_mysql</string>

<key>ExitTimeOut</key> <integer>600</integer>

<key>Program</key> <string>/usr/local/mysql/bin/mysqld</string>

<key>ProgramArguments</key>

<array>

<string>/usr/local/mysql/bin/mysqld</string>

<string>--user=\_mysql</string>

<string>--basedir=/usr/local/mysql</string>

<string>--datadir=/usr/local/mysql/data</string>

<string>--plugin-dir=/usr/local/mysql/lib/plugin</string>

<string>--log-error=/usr/local/mysql/data/mysqld.local.err</string>

<string>--pid-file=/usr/local/mysql/data/mysqld.local.pid</string>

<string>--keyring-file-data=/usr/local/mysql/keyring/keyring</string>

<string>--early-plugin-load=keyring\_file=keyring\_file.so</string>

</array>

<key>WorkingDirectory</key> <string>/usr/local/mysql</string>

</dict>

</plist>

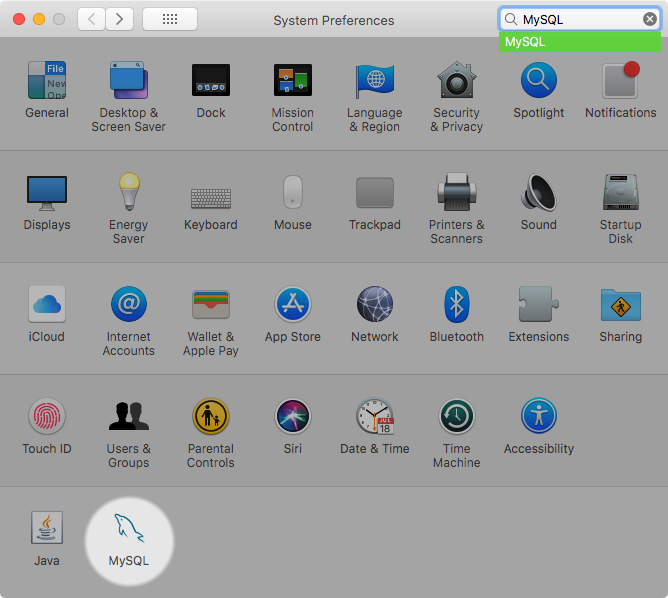
在这种情况下，baseir、datadir、plugin\_dir、log\_error、pid\_file、keyring\_file\_data和--early-plugin-load选项被从默认的plist ProgramArguments定义中移除，你可能已经在my.cnf中定义了。

### 2.4.4 安装和使用MySQL首选项面板

MySQL安装包包括一个MySQL首选项窗格，使你能够在启动MySQL安装时启动、停止和控制自动启动。

这个首选项窗格是默认安装的，并列在你的系统的系统首选项窗口下。

图2.20 MySQL首选项窗格。位置

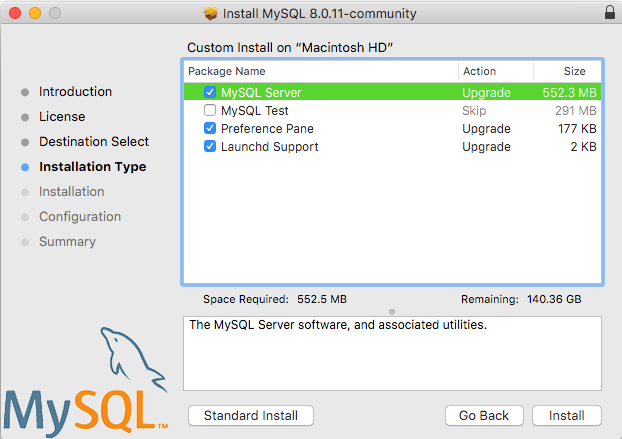


MySQL偏好面板是与安装MySQL服务器的DMG文件一起安装的。通常情况下，它与MySQL服务器一起安装，但它也可以单独安装。

1、通过安装MySQL服务器的过程，如2.4.2节 "在macOS上使用本地包安装MySQL "的文档中所述。

2、在安装类型步骤中单击 "自定义"。这里列出了 "偏好窗格 "选项，并默认为启用；确保它没有被取消选择。其他选项，如MySQL服务器，可以选择或取消选择。

图2.21 MySQL软件包安装向导。自定义



3、完成安装过程。

注意

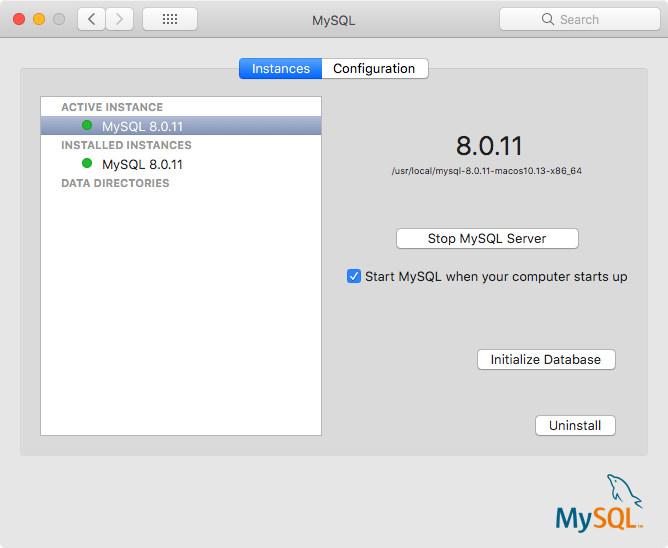
MySQL首选项窗格只启动和停止从已安装在默认位置的MySQL软件包安装的MySQL安装。

一旦安装了MySQL首选项窗格，你就可以使用这个首选项窗格控制你的MySQL服务器实例。

实例页面包括一个启动和停止MySQL的选项，而初始化数据库会重新创建数据/目录。Uninstall卸载MySQL服务器，并可选择卸载paint和launchd信息。

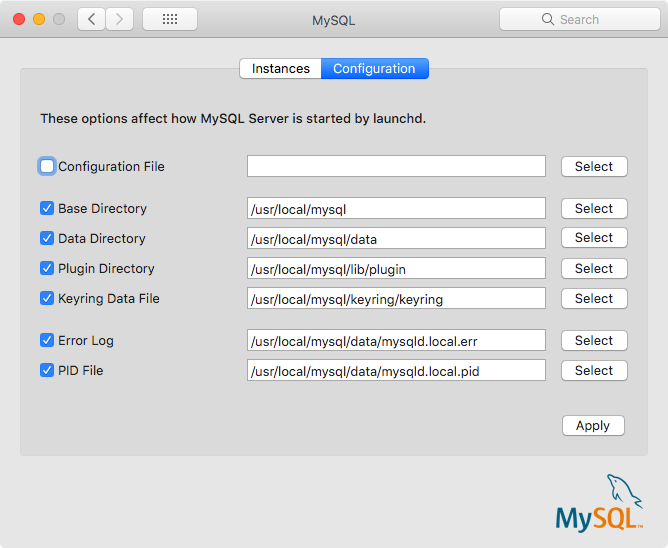
实例页面包括一个启动或停止MySQL的选项，并且初始化数据库会重新创建数据/目录。Uninstall卸载MySQL Server，也可选择卸载MySQL偏好面板和launchd信息。

图2.22 MySQL首选项面板：实例



配置页面显示了MySQL服务器选项，包括MySQL配置文件的路径。

图2.23 MySQL首选项窗格。配置



MySQL首选项窗格显示了MySQL服务器的当前状态，如果服务器没有运行，则显示停止（红色），如果服务器已经启动，则显示运行（绿色）。首选项窗格还显示了当前对MySQL服务器是否被设置为自动启动的设置。

## 2.5 在Linux上安装MySQL

Linux支持许多不同的解决方案来安装MySQL。我们建议你使用Oracle的一个发行版，它有多种安装方法可供选择。

**Table 2.8 Linux Installation Methods and Information**

|  |  |  |
| --- | --- | --- |
| **Type** | **Setup Method** | **Additional Information** |
| **Apt** | Enable the [MySQL Apt repository](https://dev.mysql.com/downloads/repo/apt/) | [Documentation](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#linux-installation-apt-repo) |
| **Yum** | Enable the [MySQL Yum repository](https://dev.mysql.com/downloads/repo/yum/) | [Documentation](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#linux-installation-yum-repo) |
| **Zypper** | Enable the [MySQL SLES repository](https://dev.mysql.com/downloads/repo/suse/) | [Documentation](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#linux-installation-sles-repo) |
| **RPM** | [Download](https://dev.mysql.com/downloads/mysql/) a specific package | [Documentation](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#linux-installation-rpm) |
| **DEB** | [Download](https://dev.mysql.com/downloads/mysql/) a specific package | [Documentation](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#linux-installation-debian) |
| **Generic** | [Download](https://dev.mysql.com/downloads/mysql/) a generic package | [Documentation](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#binary-installation) |
| **Source** | Compile from [source](https://dev.mysql.com/downloads/mysql/) | [Documentation](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#source-installation) |
| **Docker** | Use the [Oracle Container Registry](https://container-registry.oracle.com/). You can also use Docker Hub for MySQL Community Edition and [My Oracle Support](https://support.oracle.com/) for MySQL Enterprise Edition. | [Documentation](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#linux-installation-docker) |
| **Oracle Unbreakable Linux Network** | Use ULN channels | [Documentation](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#uln-installation) |

作为一种选择，你可以使用你系统上的软件包管理器，从你的Linux发行版的本地软件库中自动下载和安装MySQL的软件包。这些本地软件包通常比当前可用的版本晚几个版本。你通常也无法安装开发里程碑版本（DMR），因为这些通常不在本地软件库中提供。关于使用本地软件包安装程序的更多信息，请参见第2.5.7节，"从本地软件库安装Linux上的MySQL"。

注意

对于许多Linux的安装，你想把MySQL设置为在你的机器启动时自动启动。许多本地软件包的安装为你执行这一操作，但对于源码、二进制和RPM解决方案，你可能需要单独进行设置。所需的脚本mysql.server可以在MySQL安装目录下的support-files目录或MySQL源树中找到。你可以把它安装为/etc/init.d/mysql，以便自动启动和关闭MySQL。见第4.3.3节，"mysql.server--MySQL服务器启动脚本"。

### 2.5.1 使用MySQL Yum资源库在Linux上安装MySQL

用于Oracle Linux、Red Hat Enterprise Linux、CentOS和Fedora的MySQL Yum仓库提供了用于安装MySQL服务器、客户端、MySQL Workbench、MySQL Utilities、MySQL Router、MySQL Shell、Connector/ODBC、Connector/Python等的RPM包（并非所有包都适用于所有发行版；详情见用Yum安装其他MySQL产品和组件）。

**在你开始之前**

作为一个流行的开源软件，MySQL以其原始或重新打包的形式被广泛地安装在许多系统上，来源各异，包括不同的软件下载网站、软件库等。下面的说明假定MySQL尚未使用第三方分发的RPM包安装在你的系统上；如果不是这样，请遵循第2.11.7节 "用MySQL Yum Repository升级MySQL "或使用MySQL Yum Repository替换第三方分发的MySQL中给出的说明。

**全新安装MySQL的步骤**

按照下面的步骤，用MySQL Yum资源库安装最新的GA版MySQL。

1、添加MySQL Yum存储库

首 先，将MySQL Yum资源库添加到你的系统的资源库列表。这是一个一次性的操作，可以通过安装MySQL提供的RPM来进行。遵循这些步骤。

1. 转到MySQL开发者区的Download MySQL Yum Repository页面（<https://dev.mysql.com/downloads/repo/yum/>）。
2. 选择并下载适合你的平台的发布包。
3. 用以下命令安装下载的发布包，用下载的RPM包的名称替换平台和版本特定的包名。

shell> **sudo yum install *platform-and-version-specific-package-name*.rpm**

对于基于EL6的系统，该命令的形式为。

shell> **sudo yum install mysql80-community-release-el6-*{version-number}*.noarch.rpm**

对于基于EL7的系统

shell> **sudo yum install mysql80-community-release-el7-*{version-number}*.noarch.rpm**

对于一个基于EL8的系统。

shell> **sudo yum install mysql80-community-release-el8-*{version-number}*.noarch.rpm**

对于Fedora 33

shell> **sudo dnf install mysql80-community-release-fc33-*{version-number}*.noarch.rpm**

对于 Fedora 32:

shell> **sudo dnf install mysql80-community-release-fc32-*{version-number}*.noarch.rpm**

该安装命令将MySQL Yum软件库添加到你系统的软件库列表中，并下载GnuPG密钥以检查软件包的完整性。参见第2.1.4.2节，"使用GnuPG的签名检查"，了解GnuPG密钥检查的细节。

你可以通过以下命令检查MySQL Yum软件库是否已经成功添加（对于启用dnf的系统，在命令中用dnf替换yum）。

shell> **yum repolist enabled | grep "mysql.\*-community.\*"**

注意

一旦MySQL Yum仓库在你的系统上被启用，任何通过yum update命令进行的全系统更新（对于启用dnf的系统，则为dnf upgrade）都会升级你系统上的MySQL包，并替换任何本地第三方包，如果Yum在MySQL Yum仓库中找到它们的替代品；见2.11.7节，"用MySQL Yum仓库升级MySQL"，以讨论在你系统上的一些可能影响，见升级共享的客户端库。

2、选择一个发布系列

当使用MySQL Yum资源库时，最新的GA系列（目前是MySQL 8.0）被默认选择用于安装。如果这是你想要的，你可以跳到下一个步骤，安装MySQL。

在MySQL Yum资源库中，MySQL社区服务器的不同发布系列被托管在不同的子资源库中。最新的GA系列（目前是MySQL 8.0）的子库是默认启用的，所有其他系列的子库（例如，MySQL 8.0系列）是默认禁用的。使用这个命令可以看到MySQL Yum软件库中的所有子软件库，并查看其中哪些是启用或禁用的（对于启用dnf的系统，在命令中用dnf替换yum）。

shell> **yum repolist all | grep mysql**

要安装最新的GA系列的最新版本，不需要配置。要安装最新GA系列以外的特定系列的最新版本，在运行安装命令之前，禁用最新GA系列的子仓库，并启用特定系列的子仓库。如果你的平台支持yum-config-manager，你可以通过发布这些命令来完成，这些命令禁用了5.7系列的子库，启用了8.0系列的子库。

shell> **sudo yum-config-manager --disable mysql57-community**

shell> **sudo yum-config-manager --enable mysql80-community**

对于dnf启用的平台。

shell> **sudo dnf config-manager --disable mysql57-community**

shell> **sudo dnf config-manager --enable mysql80-community**

除了使用yum-config-manager或dnf config-manager命令，你还可以通过手动编辑/etc/yum.repos.d/mysql-community.repo文件来选择一个发布系列。这是该文件中一个典型的发布系列子库的条目。

[mysql57-community]

name=MySQL 5.7 Community Server

baseurl=http://repo.mysql.com/yum/mysql-5.7-community/el/6/$basearch/

enabled=1

gpgcheck=1

gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-mysql

找到你要配置的子库的条目，并编辑 enabled 选项。指定 enabled=0 来禁用一个子库，或者 enabled=1 来启用一个子库。例如，要安装MySQL 8.0，确保你在上述MySQL 5.7的子库条目中启用=0，而在8.0系列的条目中启用=1。

# Enable to use MySQL 8.0

[mysql80-community]

name=MySQL 8.0 Community Server

baseurl=http://repo.mysql.com/yum/mysql-8.0-community/el/6/$basearch/

enabled=1

gpgcheck=1

gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-mysql

你应该在任何时候只为一个发行系列启用子库。当启用了多个发布系列的子库时，Yum会使用最新的系列。

通过运行下面的命令并检查其输出，来验证正确的子仓库已经被启用和禁用（对于支持dnf的系统，用dnf替换命令中的yum）。

shell> **yum repolist enabled | grep mysql**

3、禁用默认的MySQL模块

仅限EL8系统）基于EL8的系统，如RHEL8和Oracle Linux 8包括一个默认启用的MySQL模块。除非该模块被禁用，否则它会掩盖由MySQL存储库提供的软件包。要禁用所包含的模块并使MySQL资源库的软件包可见，使用以下命令（对于支持dnf的系统，在命令中用dnf替换yum）。

shell> **sudo yum module disable mysql**

4、安装MySQL

用下面的命令安装MySQL（对于支持dnf的系统，用dnf代替命令中的yum）。

shell> **sudo yum install mysql-community-server**

这将安装MySQL服务器（mysql-community-server）的包，以及运行服务器所需组件的包，包括客户端（mysql-community-client）、客户端和服务器的常见错误信息和字符集（mysql-community-common）以及共享客户端库（mysql-community-libs）的包。

5、启动MySQL服务器

用以下命令启动MySQL服务器。

shell> **systemctl start mysqld**

你可以用下面的命令检查MySQL服务器的状态。

shell> **systemctl status mysqld**

如果操作系统启用了systemd，应使用标准的systemctl（或者参数相反的service）命令，如stop、start、status和restart来管理MySQL服务器服务。mysqld服务默认是启用的，并且在系统重启时启动。更多信息见2.5.9节 "用systemd管理MySQL服务器"。

在服务器初始启动时，考虑到服务器的数据目录为空，会发生以下情况。

* 服务器被初始化。
* 在数据目录中生成SSL证书和密钥文件。
* validate\_password被安装并启用。
* 一个超级用户账户'root'@'localhost'被创建。超级用户的密码被设置并存储在错误日志文件中。要显示它，请使用以下命令。

shell> **sudo grep 'temporary password' /var/log/mysqld.log**

尽快改变根密码，用生成的临时密码登录，为超级用户账户设置一个自定义密码。

shell> **mysql -uroot -p**

mysql> **ALTER USER 'root'@'localhost' IDENTIFIED BY 'MyNewPass4!';**

注意

validate\_password是默认安装的。validate\_password实现的默认密码策略要求密码至少包含一个大写字母、一个小写字母、一个数字和一个特殊字符，并且密码总长度至少为8个字符。

关于安装后程序的更多信息，请参见第2.10节，"安装后设置和测试"。

注意事项

基于EL7的平台的兼容性信息。来自平台的本地软件库的以下RPM包与来自MySQL Yum库的安装MySQL服务器的包不兼容。一旦你使用MySQL Yum仓库安装了MySQL，你就不能安装这些包（反之亦然）。

* akonadi-mysql

用Yum安装额外的MySQL产品和组件

你可以使用Yum来安装和管理MySQL的各个组件。其中一些组件被托管在MySQL Yum仓库的子仓库中：例如，MySQL连接器可以在MySQL连接器社区子仓库中找到，而MySQL工作台可以在MySQL工具社区中找到。你可以使用下面的命令，从MySQL Yum库中列出所有可用于你的平台的MySQL组件的包（对于支持dnf的系统，在命令中用dnf代替yum）。

shell> **sudo yum --disablerepo=\\* --enablerepo='mysql\*-community\*' list available**

用下面的命令安装你选择的任何包，用包的名称代替包的名称（对于支持dnf的系统，用dnf代替命令中的yum）。

shell> **sudo yum install *package-name***

例如，要在Fedora上安装MySQL Workbench。

shell> **sudo dnf install mysql-workbench-community**

要安装共享客户端库（对于支持dnf的系统，在命令中用dnf代替yum）。

shell> **sudo yum install mysql-community-libs**

特定平台说明

ARM 支持

在Oracle Linux 7上支持ARM 64位（arch64），需要Oracle Linux 7软件集合仓库（ol7\_software\_collections）。例如，要安装服务器。

shell> yum-config-manager --enable ol7\_software\_collections

shell> yum install mysql-community-server

注意

从MySQL 8.0.12开始，ARM 64位（arch64）在Oracle Linux 7上得到支持。

已知的局限性

8.0.12版本要求你在执行yum安装步骤后通过执行ln -s /opt/oracle/oracle-armtoolset-1/root/usr/lib64 /usr/lib64/gcc7来调整libstdc++7路径。

用Yum更新MySQL

除了安装，你还可以使用MySQL Yum仓库执行MySQL产品和组件的更新。详情见第2.11.7节 "用MySQL Yum资源库升级MySQL"。

### 2.5.2 使用MySQL APT资源库在Linux上安装MySQL

MySQL APT资源库提供了用于在当前的Debian和Ubuntu版本上安装和管理MySQL服务器、客户端和其他组件的deb包。

使用MySQL APT资源库的说明见《使用MySQL APT资源库的快速指南》。

### 2.5.3 使用MySQL SLES资源库在Linux上安装MySQL

MySQL SLES资源库提供用于在SUSE Enterprise Linux服务器上安装和管理MySQL服务器、客户端和其他组件的RPM包。

使用MySQL SLES资源库的说明见《使用MySQL SLES资源库的快速指南》。

### 2.5.4 使用Oracle的RPM包在Linux上安装MySQL

在基于RPM的Linux发行版上安装MySQL的推荐方法是使用Oracle提供的RPM包。对于MySQL的社区版，有两个来源可以获得它们。

* 从MySQL软件库中。
  + MySQL Yum库（详见第2.5.1节，"使用MySQL Yum库在Linux上安装MySQL"）。
  + MySQL SLES资源库（详见第2.5.3节，"使用MySQL SLES资源库在Linux上安装MySQL"）。
* 从MySQL开发者区的 "下载MySQL社区服务器 "页面。

注意

其他供应商也提供MySQL的RPM发行版。请注意，它们在特性、功能和惯例（包括通信设置）方面可能与Oracle构建的版本不同，而且本手册中的安装说明不一定适用于它们。应该参考供应商的说明。

MySQL RPM包

**Table 2.9 RPM Packages for MySQL Community Edition**

|  |  |
| --- | --- |
| **Package Name** | **Summary** |
| **mysql-community-client** | MySQL client applications and tools |
| **mysql-community-common** | Common files for server and client libraries |
| **mysql-community-devel** | Development header files and libraries for MySQL database client applications |
| **mysql-community-embedded-compat** | MySQL server as an embedded library with compatibility for applications using version 18 of the library |
| **mysql-community-libs** | Shared libraries for MySQL database client applications |
| **mysql-community-libs-compat** | Shared compatibility libraries for previous MySQL installations |
| **mysql-community-server** | Database server and related tools |
| **mysql-community-server-debug** | Debug server and plugin binaries |
| **mysql-community-test** | Test suite for the MySQL server |
| **mysql-community** | The source code RPM looks similar to mysql-community-8.0.26-1.el7.src.rpm, depending on selected OS |
| Additional \*debuginfo\* RPMs | There are several **debuginfo** packages: mysql-community-client-debuginfo, mysql-community-libs-debuginfo mysql-community-server-debug-debuginfo mysql-community-server-debuginfo, and mysql-community-test-debuginfo. |

**Table 2.10 RPM Packages for the MySQL Enterprise Edition**

|  |  |
| --- | --- |
| **Package Name** | **Summary** |
| **mysql-commercial-backup** | MySQL Enterprise Backup (added in 8.0.11) |
| **mysql-commercial-client** | MySQL client applications and tools |
| **mysql-commercial-common** | Common files for server and client libraries |
| **mysql-commercial-devel** | Development header files and libraries for MySQL database client applications |
| **mysql-commercial-embedded-compat** | MySQL server as an embedded library with compatibility for applications using version 18 of the library |
| **mysql-commercial-libs** | Shared libraries for MySQL database client applications |
| **mysql-commercial-libs-compat** | Shared compatibility libraries for previous MySQL installations; the version of the libraries matches the version of the libraries installed by default by the distribution you are using |
| **mysql-commercial-server** | Database server and related tools |
| **mysql-commercial-test** | Test suite for the MySQL server |
| Additional \*debuginfo\* RPMs | There are several **debuginfo** packages: mysql-commercial-client-debuginfo, mysql-commercial-libs-debuginfo mysql-commercial-server-debug-debuginfo mysql-commercial-server-debuginfo, and mysql-commercial-test-debuginfo. |

rpm包的全名语法如下

***packagename***-***version***-***distribution***-***arch***.rpm

发行版和arch值表示Linux发行版和软件包所对应的处理器类型。关于发行版标识符的列表，见下表。

**table 2.11 MySQL Linux RPM Package Distribution Identifiers**

|  |  |
| --- | --- |
| **Distribution Value** | **Intended Use** |
| el***{version}*** where ***{version}*** is the major Enterprise Linux version, such as **el8** | EL6, EL7, and EL8-based platforms (for example, the corresponding versions of Oracle Linux, Red Hat Enterprise Linux, and CentOS) |
| fc***{version}*** where ***{version}*** is the major Fedora version, such as **fc33** | Fedora 32 and 33 |
| **sles12** | SUSE Linux Enterprise Server 12 |

要查看一个RPM包中的所有文件（例如，mysql-community-server），使用以下命令。

shell> **rpm -qpl mysql-community-server-*version*-*distribution*-*arch*.rpm**

本节其余部分的讨论仅适用于使用直接从Oracle下载的RPM包的安装过程，而不是通过MySQL资源库。

一些包之间存在依赖关系。如果你计划安装许多包，你可能希望下载RPM捆绑焦油文件，它包含上面列出的所有RPM包，因此你不需要单独下载它们。

在大多数情况下，你需要安装mysql-community-server、mysql-community-client、mysql-community-libs、mysql-community-common和mysql-community-libs-compat包来获得一个功能性的、标准的MySQL安装。要执行这样一个标准的、基本的安装，进入包含所有这些包的文件夹（最好是没有其他具有类似名称的RPM包），并发出以下命令。

shell> **sudo yum install mysql-community-{server,client,common,libs}-\***

对于SLES用zypper替换yum，对于Fedora用dnf。

虽然使用yum这样的高级软件包管理工具来安装软件包更为可取，但喜欢直接使用rpm命令的用户可以用rpm -Uvh命令代替yum安装命令；然而，使用rpm -Uvh代替会使安装过程更容易失败，因为安装过程可能遇到潜在的依赖问题。

要只安装客户端程序，你可以在你要安装的软件包列表中跳过mysql-community-server；发出以下命令。

shell> **sudo yum install mysql-community-{client,common,libs}-\***

对于SLES用zypper替换yum，对于Fedora用dnf。

使用RPM包对MySQL进行标准安装，会在系统目录下创建文件和资源，如下表所示

**Table 2.12 MySQL Installation Layout for Linux RPM Packages from the MySQL Developer Zone**

|  |  |
| --- | --- |
| **Files or Resources** | **Location** |
| Client programs and scripts | /usr/bin |
| [**mysqld**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\programs.html#mysqld) server | /usr/sbin |
| Configuration file | /etc/my.cnf |
| Data directory | /var/lib/mysql |
| Error log file | For RHEL, Oracle Linux, CentOS or Fedora platforms: /var/log/mysqld.log  For SLES: /var/log/mysql/mysqld.log |
| Value of [**secure\_file\_priv**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\server-administration.html#sysvar_secure_file_priv) | /var/lib/mysql-files |
| System V init script | For RHEL, Oracle Linux, CentOS or Fedora platforms: /etc/init.d/mysqld  For SLES: /etc/init.d/mysql |
| Systemd service | For RHEL, Oracle Linux, CentOS or Fedora platforms: mysqld  For SLES: mysql |
| Pid file | /var/run/mysql/mysqld.pid |
| Socket | /var/lib/mysql/mysql.sock |
| Keyring directory | /var/lib/mysql-keyring |
| Unix manual pages | /usr/share/man |
| Include (header) files | /usr/include/mysql |
| Libraries | /usr/lib/mysql |
| Miscellaneous support files (for example, error messages, and character set files) | /usr/share/mysql |

该安装还在系统上创建了一个名为mysql的用户和一个名为mysql的组。

注意

使用旧软件包安装先前版本的MySQL可能创建了一个名为/usr/my.cnf的配置文件。强烈建议你检查该文件的内容，并将里面所需的设置迁移到/etc/my.cnf文件，然后删除/usr/my.cnf。

在安装过程结束时，MySQL不会被自动启动。对于Red Hat Enterprise Linux、Oracle Linux、CentOS和Fedora系统，使用以下命令来启动MySQL。

shell> **systemctl start mysqld**

对于SLES系统，命令是相同的，但服务名称不同。

shell> **systemctl start mysql**

如果操作系统启用了systemd，应使用标准的systemctl（或者参数相反的service）命令，如stop、start、status和restart来管理MySQL服务器服务。mysqld服务默认是启用的，并且在系统重启时启动。注意，某些事情在systemd平台上可能会有不同的工作方式：例如，改变数据目录的位置可能会导致问题。更多信息请参见2.5.9节 "用systemd管理MySQL服务器"。

在使用RPM和DEB包进行升级安装时，如果升级发生时MySQL服务器正在运行，那么MySQL服务器会被停止，升级发生后，MySQL服务器会被重新启动。有一个例外：如果在升级期间版本也发生了变化（例如从社区版到商业版，或者反过来），那么MySQL服务器不会被重新启动。

在服务器的初始启动中，鉴于服务器的数据目录是空的，以下情况会发生。

* 服务器被初始化。
* 在数据目录中生成SSL证书和密钥文件。
* validate\_password被安装并启用。
* 一个超级用户账户'root'@'localhost'被创建。超级用户的密码被设置并存储在错误日志文件中。要揭示它，对于RHEL、Oracle Linux、CentOS和Fedora系统使用以下命令。

shell> **sudo grep 'temporary password' /var/log/mysqld.log**

对SLES系统使用下面的命令。

shell> **sudo grep 'temporary password' /var/log/mysql/mysqld.log**

下一步是用生成的临时密码登录，并为超级用户账户设置一个自定义密码。

shell> **mysql -uroot -p**

mysql> **ALTER USER 'root'@'localhost' IDENTIFIED BY 'MyNewPass4!';**

注意

validate\_password是默认安装的。validate\_password实现的默认密码策略要求密码至少包含一个大写字母、一个小写字母、一个数字和一个特殊字符，并且密码总长度至少为8个字符。

如果在安装过程中出了问题，你可能会在错误日志文件/var/log/mysqld.log中找到调试信息。

对于某些Linux发行版，可能需要增加对mysqld可用的文件描述符数量的限制。见B.3.2.16节，"文件未找到和类似错误"

从多个MySQL版本安装客户端库。 有可能安装多个客户库版本，例如，你想保持与针对以前的库链接的旧应用程序的兼容性。要安装一个较早的客户库，使用rpm的--oldpackage选项。例如，要在EL6系统上安装mysql-community-libs-5.5，该系统有来自MySQL 8.0的libmysqlclient.21，使用这样的命令。

shell> **rpm --oldpackage -ivh mysql-community-libs-5.5.50-2.el6.x86\_64.rpm**

调试包。 一个用调试包编译的MySQL服务器的特殊变体已经包含在服务器RPM包中。它执行调试和内存分配检查，并在服务器运行时产生一个跟踪文件。要使用该调试版本，用/usr/sbin/mysqld-debug启动MySQL，而不是作为服务或用/usr/sbin/mysqld启动。关于你可以使用的调试选项，见第5.9.4节，"DBUG包"。

注意

在MySQL 8.0.4中，调试构建的默认插件目录从/usr/lib64/mysql/plugin变为/usr/lib64/mysql/plugin/debug。以前，对于调试构建，必须将plugin\_dir改为/usr/lib64/mysql/plugin/debug。

从源码SRPMs重建RPMs。 MySQL的源代码SRPM包可供下载。它们可以按原样使用，用标准的rpmbuild工具链重建MySQL RPM。

### 2.5.5 使用Oracle的Debian软件包在Linux上安装MySQL

Oracle提供Debian软件包，用于在Debian或类Debian Linux系统上安装MySQL。这些软件包可以通过两个不同的渠道获得。

* MySQL APT资源库。这是在类Debian系统上安装MySQL的首选方法，因为它为安装和更新MySQL产品提供了一种简单而方便的方法。详情请参见第2.5.2节 "使用MySQL APT资源库在Linux上安装MySQL"。
* MySQL开发者专区的下载区。详情请见第2.1.3节 "如何获得MySQL"。下面是关于那里提供的Debian软件包的一些信息以及安装它们的说明。
  + 在MySQL开发区提供了各种Debian软件包，用于在当前的Debian和Ubuntu平台上安装MySQL的不同组件。首选的方法是使用tarball捆绑包，它包含了MySQL基本设置所需的包。压缩包的名称格式为mysql-server\_MVER-DVER\_CPU.deb-bundle.tar。 MVER是MySQL版本，DVER是Linux发行版本。CPU值表示软件包所对应的处理器类型或系列，如下表所示。

表2.13 MySQL Debian和Ubuntu安装包的CPU标识符

**Table 2.13 MySQL Debian and Ubuntu Installation Packages CPU Identifiers**

|  |  |
| --- | --- |
| ***CPU* Value** | **Intended Processor Type or Family** |
| **i386** | Pentium processor or better, 32 bit |
| **amd64** | 64-bit x86 processor |

* 在下载tarball后，用下面的命令解开它。

shell> **tar -xvf mysql-server\_*MVER*-*DVER*\_*CPU*.deb-bundle.tar**

* 如果你的系统上还没有libaio库，你可能需要安装它。

shell> **sudo apt-get install libaio1**

* 用下面的命令预先配置MySQL服务器包。

shell> **sudo dpkg-preconfigure mysql-community-server\_\*.deb**

你会被要求为你的MySQL安装提供一个根用户的密码。你可能还会被问到关于安装的其他问题。

重要提示

确保你记住你设置的根密码。想以后再设置密码的用户可以在对话框中把密码栏留空，只按确定；在这种情况下，对服务器的根访问是使用MySQL Socket Peer-Credential Authentication Plugin对使用Unix socket文件的连接进行验证的。你可以在以后使用mysql\_secure\_installation设置根密码。

* 对于MySQL服务器的基本安装，安装数据库公共文件包、客户端包、客户端元包、服务器包和服务器元包（按照这个顺序）；你可以用一个命令来做。

shell> **sudo dpkg -i mysql-{common,community-client,client,community-server,server}\_\*.deb**

还有一些软件包名称中带有server-core和client-core的软件包。这些包只包含二进制文件，并由标准包自动安装。安装它们本身并不会导致一个正常的MySQL设置。

如果你被dpkg警告有未满足的依赖关系，你可以用apt-get修复它们。

**sudo apt-get -f install**

这里是文件在系统上的安装位置。

* 所有的配置文件（比如my.cnf）都在/etc/mysql下
* 所有的二进制文件、库、头文件等，都在/usr/bin和/usr/sbin下

注意

Debian版的MySQL也由其他供应商提供。请注意，它们可能在特性、功能和惯例（包括通信设置）方面与甲骨文构建的版本不同，而且本手册中的说明不一定适用于安装它们。应该参考供应商的说明。

### 2.5.6 用Docker在Linux上部署MySQL

Docker部署框架支持MySQL服务器的简单安装和配置。本节解释了如何使用MySQL Server Docker镜像。

在使用MySQL Server Docker镜像之前，你需要在你的系统上安装Docker。请参阅安装Docker以获得说明。

重要信息

你需要用sudo运行docker命令，或者创建一个docker用户组，然后将任何想要运行docker命令的用户添加到其中。详情请见这里。由于Docker容器总是以root权限运行，你应该了解Docker守护程序的攻击面，并适当降低相关风险。

#### 2.5.6.1 使用Docker部署MySQL服务器的基本步骤

警告

由MySQL团队维护的MySQL Docker镜像是专门为Linux平台构建的。其他平台不受支持，用户在这些平台上使用这些MySQL Docker映像，其风险自负。关于在非Linux操作系统上运行这些容器的一些已知限制，请参见这里的讨论。

下载MySQL服务器Docker镜像

重要信息

对于MySQL企业版的用户。要使用MySQL企业版的Docker镜像，需要订阅。订阅通过自带许可证模式工作；详情请参见如何购买MySQL产品和服务。

在一个单独的步骤中下载服务器镜像并不是严格必要的；然而，在你创建Docker容器之前执行这个步骤可以确保你的本地镜像是最新的。要下载MySQL社区版镜像，请运行此命令。

**docker pull mysql/mysql-server:*tag***

标签是你想拉的镜像版本的标签（例如，5.6、5.7、8.0或最新）。如果省略了:tag，就会使用最新的标签，并下载MySQL Community Server的最新GA版本的图像。请参考Docker Hub中mysql/mysql-server页面上的可用版本的标签列表。

要从Oracle容器注册中心（OCR）下载MySQL社区版的镜像，请运行此命令。

**docker pull container-registry.oracle.com/mysql/mysql-server:*tag***

要从OCR下载MySQL企业版镜像，你需要首先接受OCR上的许可协议，并使用你的Docker客户端登录到容器仓库。

* 访问OCR：https://container-registry.oracle.com/，并选择MySQL。
* 在MySQL存储库列表下，选择enterpris-server。
* 如果你还没有登录到OCR，点击页面右侧的登录按钮，然后在提示下输入你的Oracle账户凭证。
* 按照页面右侧的指示，接受许可协议。
* 用你的Docker客户端（docker命令）登录OCR，使用docker登录命令。

# **docker login container-registry.oracle.com**

Username: ***Oracle-Account-ID***

Password: ***password***

Login successful.

用这个命令从OCR下载MySQL企业版的Docker镜像。

**docker pull container-registry.oracle.com/mysql/enterprise-server:*tag***

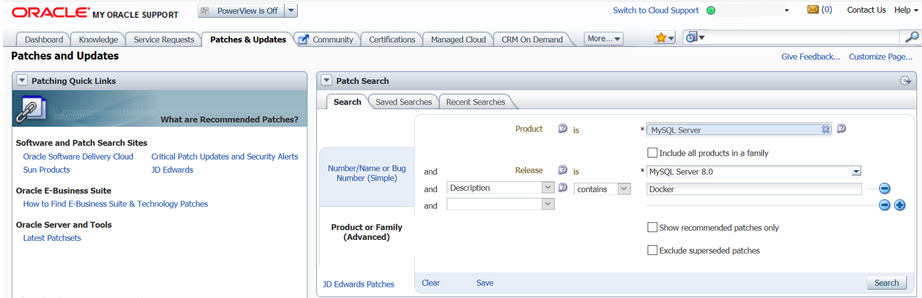
标签有不同的选择，对应于OCR提供的MySQL Docker镜像的不同版本。

* 8.0、8.0.x（x是8.0系列中的最新版本号）、最新。MySQL 8.0，最新的GA
* 5.7，5.7.y（y是5.7系列中的最新版本号）。MySQL 5.7

要从 "我的Oracle支持 "网站下载MySQL企业版镜像，请进入该网站，登录到你的Oracle账户，一旦你进入登陆页面，请执行这些步骤。

* 选择 "补丁和更新 "标签。
* 进入补丁搜索区域，在搜索标签上，切换到产品或家族（高级）子标签。
* 在产品字段中输入 "MySQL Server"，在版本字段中输入所需的版本号。
* 使用附加过滤器的下拉菜单，选择 "描述-包含"，并在文本字段中输入 "Docker"。

下图显示了MySQL Server 8.0的MySQL企业版图像的搜索设置。



* 点击搜索按钮，从结果列表中选择你想要的版本，并点击下载按钮。
* 在出现的文件下载对话框中，点击并下载Docker镜像的.zip文件。

解压缩下载的.zip档案，以获得里面的tar球（mysql-enterprise-server-version.tar），然后通过运行此命令加载镜像。

**docker load -i mysql-enterprise-server-*version*.tar**

你可以用这个命令列出下载的Docker镜像。

shell> docker images

shell> **docker images**

REPOSITORY TAG IMAGE ID CREATED SIZE

mysql/mysql-server latest 3157d7f55f8d 4 weeks ago 241MB

启动MySQL服务器实例

要为MySQL服务器启动一个新的Docker容器，使用以下命令。

**docker run --name=*container\_name* --restart on-failure -d *image\_name*:*tag***

镜像名称可以使用docker images命令获得，如下载MySQL服务器Docker镜像中所解释的。

--name选项用于为你的服务器容器提供一个自定义名称，是可选的；如果没有提供容器名称，就会生成一个随机名称。

--restart选项用于配置你的容器的重启策略；它应该被设置为on-failure值，以便在客户端会话中支持服务器重启（例如，当客户端执行RESTART语句或在配置InnoDB集群实例时发生）。启用对重启的支持后，在客户端会话中发出重启会导致服务器和容器停止，然后重新启动。对服务器重启的支持适用于MySQL 8.0.21及更高版本。

例如，要为MySQL社区服务器启动一个新的Docker容器，使用这个命令。

**docker run --name=mysql1 --restart on-failure -d mysql/mysql-server:8.0**

要使用从OCR下载的Docker镜像为MySQL企业服务器启动一个新的Docker容器，请使用此命令。

**docker run --name=mysql1 --restart on-failure -d container-registry.oracle.com/mysql/enterprise-server:8.0**

要使用从My Oracle Support下载的Docker镜像为MySQL企业服务器启动一个新的Docker容器，请使用此命令。

要用从OCR下载的Docker镜像为MySQL企业服务器启动一个新的Docker容器，请使用此命令。

**docker run --name=mysql1 --restart on-failure -d container-registry.oracle.com/mysql/enterprise-server:8.0**

要使用从My Oracle Support下载的Docker镜像为MySQL企业服务器启动一个新的Docker容器，请使用此命令。

**docker run --name=mysql1 --restart on-failure -d mysql/enterprise-server:8.0**

如果指定名称和标签的Docker镜像没有被先前的docker pull或docker run命令下载，那么现在就会下载该镜像。容器的初始化开始了，当你运行docker ps命令时，该容器会出现在正在运行的容器列表中。比如说

shell> **docker ps**

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

a24888f0d6f4 mysql/mysql-server "/entrypoint.sh my..." 14 seconds ago Up 13 seconds (health: starting) 3306/tcp, 33060/tcp mysql1

容器初始化可能需要一些时间。当服务器可以使用时，docker ps命令输出中的容器STATUS会从（健康：开始）变为（健康）。

上面的docker run命令中使用的-d选项使容器在后台运行。使用此命令来监控容器的输出。

**docker logs mysql1**

一旦初始化完成，该命令的输出将包含为根用户生成的随机密码；例如，用这个命令检查密码。

shell> **docker logs mysql1 2>&1 | grep GENERATED**

GENERATED ROOT PASSWORD: Axegh3kAJyDLaRuBemecis&EShOs

从容器内连接到MySQL服务器

一旦服务器准备好了，你可以在你刚刚启动的MySQL服务器容器内运行mysql客户端，并将其连接到MySQL服务器。使用docker exec -it命令在你启动的Docker容器内启动一个mysql客户端，像下面这样。

**docker exec -it mysql1 mysql -uroot -p**

当被要求时，输入生成的根密码（见上面启动MySQL服务器实例的最后一步，如何找到密码）。因为MYSQL\_ONETIME\_PASSWORD选项默认为真，在你将一个mysql客户端连接到服务器后，你必须通过发出这个语句来重置服务器根密码。

mysql> **ALTER USER 'root'@'localhost' IDENTIFIED BY '*password*';**

用你选择的密码来代替密码。一旦密码被重置，服务器就可以使用了。

容器外壳访问

要对你的MySQL服务器容器进行shell访问，使用docker exec -it命令在容器内启动一个bash shell。

shell> **docker exec -it mysql1 bash**

bash-4.2#

然后你可以在容器内运行Linux命令。例如，要在容器内查看服务器的数据目录中的内容，使用这个命令。

bash-4.2# **ls /var/lib/mysql**

auto.cnf ca.pem client-key.pem ib\_logfile0 ibdata1 mysql mysql.sock.lock private\_key.pem server-cert.pem sys

ca-key.pem client-cert.pem ib\_buffer\_pool ib\_logfile1 ibtmp1 mysql.sock performance\_schema public\_key.pem server-key.pem

停止和删除一个MySQL容器

要停止我们已经创建的MySQL服务器容器，请使用此命令。

**docker stop mysql1**

docker stop向mysqld进程发送一个SIGTERM信号，这样服务器就被优雅地关闭了。

还要注意的是，当一个容器的主进程（在MySQL服务器容器的情况下是mysqld）被停止时，Docker容器会自动停止。

要再次启动MySQL服务器容器

**docker start mysql1**

要用一条命令停止和再次启动MySQL服务器容器

**docker restart mysql1**

要删除MySQL容器，首先停止它，然后使用docker rm命令。

**docker stop mysql1**

**docker rm mysql1**

如果你想同时删除服务器数据目录的Docker卷，在docker rm命令中添加-v选项。

升级MySQL服务器容器

重要提示

在对MySQL进行任何升级之前，请仔细遵循第2.11节 "升级MySQL "中的说明。在那里讨论的其他指示中，在升级前备份你的数据库是特别重要的。

本节中的说明要求服务器的数据和配置已经在主机上被持久化。详情请看持久化数据和配置变化。

按照这些步骤，将MySQL 5.7的Docker安装升级到8.0。

* 停止MySQL 5.7服务器（容器名称在本例中为mysql57）。

**docker stop mysql57**

* 下载MySQL 8.0服务器Docker镜像。见下载MySQL服务器Docker镜像中的说明；确保你为MySQL 8.0使用正确的标签。
* 启动一个新的MySQL 8.0 Docker容器（在本例中命名为mysql80），使用已经在主机上持久存在的旧服务器数据和配置（如果需要的话，进行适当的修改--见第2.11节 "升级MySQL"）。对于MySQL社区服务器，运行这个命令。
* **docker run --name=mysql80 \**
* **--mount type=bind,src=/path-on-host-machine/my.cnf,dst=/etc/my.cnf \**
* **--mount type=bind,src=/path-on-host-machine/datadir,dst=/var/lib/mysql \**
* **-d mysql/mysql-server:8.0**

如果需要，将mysql/mysql-server调整为正确的映像名称--例如，对于从OCR下载的MySQL企业版映像，将其替换为container-registry.oracle.com/mysql/enterprise-server；对于从My Oracle Support下载的MySQL企业版映像，则替换为mysql/enterprise-server。

* 等待服务器完成启动。你可以使用docker ps命令检查服务器的状态（关于如何做，请参阅启动MySQL服务器实例）。
* 对于MySQL 8.0.15和更早的版本。在MySQL 8.0服务器容器中运行mysql\_upgrade工具（对于MySQL 8.0.16及更高版本不需要）。
* **docker exec -it mysql80 mysql\_upgrade -uroot -p**

当出现提示时，输入旧的MySQL 5.7服务器的根密码。

通过重新启动MySQL 8.0服务器容器完成升级。

**docker restart mysql80**

关于用Docker部署MySQL服务器的更多主题

关于用Docker部署MySQL服务器的更多主题，如服务器配置、持久化数据和配置、服务器错误日志和容器环境变量，请参见第2.5.6.2节 "关于用Docker部署MySQL服务器的更多主题"。

#### 2.5.6.2 关于用Docker部署MySQL服务器的更多主题

注意

下面的大多数示例命令都将mysql/mysql-server作为必须指定的Docker映像库（如docker pull和docker run命令）；如果你的映像来自其他映像库，请将其改为container-registry.oracle.com/mysql/enterprise-server，用于从Oracle容器注册中心（OCR）下载的MySQL企业版映像，或用于从My Oracle Support下载的MySQL企业版映像的mysql/enterprise-server。

为Docker优化的MySQL安装

用于MySQL的Docker镜像针对代码大小进行了优化，这意味着它们只包括预计与在Docker容器中运行MySQL实例的大多数用户有关的关键组件。MySQL Docker安装与普通的、非Docker安装在以下方面有所不同。

包括的二进制文件仅限于。

* /usr/bin/my\_print\_defaults
* /usr/bin/mysql
* /usr/bin/mysql\_config
* /usr/bin/mysql\_install\_db
* /usr/bin/mysql\_tzinfo\_to\_sql
* /usr/bin/mysql\_upgrade
* /usr/bin/mysqladmin
* /usr/bin/mysqlcheck
* /usr/bin/mysqldump
* /usr/bin/mysqlpump
* /usr/bin/mysqlbackup（仅适用于MySQL企业版8.0）。
* /usr/sbin/mysqld

所有二进制文件都被剥离；它们不包含任何调试信息。

配置MySQL服务器

当你启动MySQL Docker容器时，你可以通过docker运行命令将配置选项传递给服务器。比如说

**docker run --name mysql1 -d mysql/mysql-server:*tag* --character-set-server=utf8mb4 --collation-server=utf8mb4\_col**

该命令以utf8mb4作为数据库的默认字符集和utf8mb4\_col作为数据库的默认整理方式启动你的MySQL服务器。

配置MySQL服务器的另一种方法是准备一个配置文件并将其挂载在容器内的服务器配置文件的位置。详情请参见持久化数据和配置更改。

保存数据和配置变更

Docker容器原则上是短暂的，如果容器被删除或损坏，任何数据或配置都会丢失（见这里的讨论）。然而，Docker卷提供了一种机制来持久化在Docker容器内创建的数据。在初始化时，MySQL Server容器为服务器数据目录创建一个Docker卷。在容器上运行docker inspect命令的JSON输出有一个Mount键，其值提供了关于数据目录卷的信息。

shell> **docker inspect mysql1**

...

"Mounts": [

{

"Type": "volume",

"Name": "4f2d463cfc4bdd4baebcb098c97d7da3337195ed2c6572bc0b89f7e845d27652",

"Source": "/var/lib/docker/volumes/4f2d463cfc4bdd4baebcb098c97d7da3337195ed2c6572bc0b89f7e845d27652/\_data",

"Destination": "/var/lib/mysql",

"Driver": "local",

"Mode": "",

"RW": true,

"Propagation": ""

}

],

...

输出显示，源文件夹/var/lib/docker/volumes/4f2d463cfc4bdd4baebcb098c97d7da3337195ed2c6572bc0b89f7e845d27652/\_data，其中的数据在主机上被持久化，已经被挂载到/var/lib/mysql，即容器内部的服务器数据目录。

另一种保存数据的方法是在创建容器时使用-mount选项绑定挂载一个主机目录。同样的技术也可以用来保存服务器的配置。下面的命令创建了一个MySQL服务器容器，并绑定挂载了数据目录和服务器配置文件。

**docker run --name=mysql1 \**

**--mount type=bind,src=*/path-on-host-machine/my.cnf*,dst=/etc/my.cnf \**

**--mount type=bind,src=*/path-on-host-machine/datadir*,dst=/var/lib/mysql \**

**-d mysql/mysql-server:*tag***

该命令将path-on-host-machine/my.cnf挂载在/etc/my.cnf（容器内的服务器配置文件），并将path-on-host-machine/datadir挂载在/var/lib/mysql（容器内的数据目录）。绑定挂载必须满足以下条件才能发挥作用。

* 配置文件 path-on-host-machine/my.cnf 必须已经存在，并且它必须包含使用用户 mysql 启动服务器的规范。
* [mysqld]
* user=mysql

你也可以在该文件中包括其他服务器配置选项。

* 数据目录path-on-host-machine/datadir必须已经存在。为了使服务器初始化发生，该目录必须是空的。你也可以挂载一个预先填充了数据的目录，然后用它来启动服务器；然而，你必须确保以与创建数据的服务器相同的配置来启动Docker容器，并且在启动容器时挂载任何需要的主机文件或目录。

运行额外的初始化脚本

如果有任何.sh或.sql脚本你想在数据库创建后立即运行，你可以把它们放入一个主机目录，然后在容器内的/docker-entrypoint-initdb.d/挂载该目录。比如说

**docker run --name=mysql1 \**

**--mount type=bind,src=*/path-on-host-machine/scripts/*,dst=/docker-entrypoint-initdb.d/ \**

**-d mysql/mysql-server:*tag***

从另一个Docker容器中的应用程序连接到MySQL

通过设置Docker网络，你可以允许多个Docker容器相互通信，这样另一个Docker容器中的客户端应用程序就可以访问服务器容器中的MySQL服务器。首先，创建一个Docker网络。

**docker network create *my-custom-net***

然后，当你创建并启动服务器和客户端容器时，使用--network选项将它们放在你创建的网络上。比如说

**docker run --name=mysql1 --network=*my-custom-net* -d mysql/mysql-server**

**docker run --name=myapp1 --network=*my-custom-net* -d myapp**

然后，myapp1容器可以用mysql1的主机名连接到mysql1容器，反之亦然，因为Docker自动为给定的容器名设置了一个DNS。在下面的例子中，我们从myapp1容器内部运行mysql客户端，以连接到它自己容器中的主机mysql1。

**docker exec -it myapp1 mysql --host=mysql1 --user=myuser --password**

关于容器的其他网络技术，请参阅Docker文档中的Docker容器网络部分。

服务器错误日志

当MySQL服务器首次与你的服务器容器一起启动时，如果以下任一条件为真，则不会生成服务器错误日志。

* 来自主机的服务器配置文件已被挂载，但该文件不包含系统变量log\_error（参见关于绑定挂载服务器配置文件的持续数据和配置更改）。
* 来自主机的服务器配置文件没有被挂载，但Docker环境变量MYSQL\_LOG\_CONSOLE为真（这是MySQL 8.0服务器容器的变量的默认状态）。然后，MySQL服务器的错误日志被重定向到stderr，这样错误日志就会进入Docker容器的日志，并可使用docker logs mysqld-tainer命令查看。

要使MySQL服务器在两个条件中的任何一个为真时生成错误日志，使用-log-error选项来配置服务器在容器内的特定位置生成错误日志。要持久化错误日志，请在容器内错误日志的位置挂载一个主机文件，如持久化数据和配置更改中所解释的。然而，你必须确保你的MySQL服务器在其容器内有对挂载的主机文件的写入权限。

在Docker中使用MySQL企业备份

MySQL Enterprise Backup是一个商业许可的MySQL Server备份工具，可与MySQL企业版一起使用。MySQL企业备份包含在MySQL企业版的Docker安装中

在下面的例子中，我们假设你已经有一个在Docker容器中运行的MySQL服务器（参见第2.5.6.1节，"用Docker部署MySQL服务器的基本步骤"，关于如何用Docker启动MySQL服务器实例）。为了使MySQL企业备份能够备份MySQL服务器，它必须能够访问该服务器的数据目录。例如，这可以通过在启动MySQL服务器时在其数据目录上绑定挂载一个主机目录来实现。

**docker run --name=mysqlserver \**

**--mount type=bind,src=*/path-on-host-machine/datadir/*,dst=/var/lib/mysql \**

**-d mysql/enterprise-server:8.0**

通过这个命令，MySQL服务器以MySQL企业版的Docker镜像启动，主机目录/path-on-host-machine/datadir/已经被挂载到服务器容器内的数据目录（/var/lib/mysql）。我们还假设，在服务器启动后，也为 MySQL 企业备份设置了访问服务器所需的权限（详见授予备份管理员 MySQL 权限）。然后使用以下步骤来备份和恢复MySQL服务器实例。

要使用MySQL Enterprise Backup with Docker来备份在Docker容器中运行的MySQL Server实例。

1、在运行MySQL Server容器的同一主机上，用MySQL企业版的镜像启动另一个容器，用MySQL企业备份命令backup-to-image进行备份。使用我们在上一步创建的绑定挂载提供对服务器的数据目录的访问。同时，将主机目录（本例中为/path-on-host-machine/backups/）挂载到容器中备份的存储文件夹上（本例中为/data/backups），以保持我们正在创建的备份的持续性。下面是这一步的示例命令，其中MySQL企业备份是用从My Oracle支持中心下载的Docker镜像启动的。)

shell> **docker run \**

**--mount type=bind,src=*/path-on-host-machine/datadir/*,dst=/var/lib/mysql \**

**--mount type=bind,src=*/path-on-host-machine/backups/*,dst=/data/backups \**

**--rm mysql/enterprise-server:8.0 \**

**mysqlbackup -u*mysqlbackup* -p*password* --backup-dir=/tmp/backup-tmp --with-timestamp \**

**--backup-image=/data/backups/db.mbi backup-to-image**

[Entrypoint] MySQL Docker Image 8.0.11-1.1.5

MySQL Enterprise Backup version 8.0.11 Linux-4.1.12-61.1.16.el7uek.x86\_64-x86\_64 [2018-04-08 07:06:45]

Copyright (c) 2003, 2018, Oracle and/or its affiliates. All Rights Reserved.

180921 17:27:25 MAIN INFO: A thread created with Id '140594390935680'

180921 17:27:25 MAIN INFO: Starting with following command line ...

...

-------------------------------------------------------------

Parameters Summary

-------------------------------------------------------------

Start LSN : 29615616

End LSN : 29651854

-------------------------------------------------------------

mysqlbackup completed OK!

检查mysqlbackup输出的结尾是很重要的，以确保备份已经成功完成。

2、一旦备份工作完成，容器就会退出，而且，用--rm选项来启动它，它退出后就会被删除。一个镜像备份已经被创建，可以在上一步安装的用于存储备份的主机目录中找到。

shell> ls /tmp/backups

db.mbi

要使用MySQL Enterprise Backup with Docker恢复Docker容器中的MySQL Server实例。

1、停止MySQL服务器容器，这也会停止里面运行的MySQL服务器。

docker stop mysqlserver

2、在主机上，删除MySQL服务器数据目录的绑定挂载中的所有内容。

rm -rf ***/path-on-host-machine/datadir***/\*

3、用MySQL企业版的映像启动一个容器，用MySQL企业备份命令copy-back-apply-log执行还原。像我们备份服务器时那样，绑定挂载服务器的数据目录和备份的存储文件夹。

shell> **docker run \**

**--mount type=bind,src=*/path-on-host-machine/datadir/*,dst=/var/lib/mysql \**

**--mount type=bind,src=*/path-on-host-machine/backups/*,dst=/data/backups \**

**--rm mysql/enterprise-server:8.0 \**

**mysqlbackup --backup-dir=/tmp/backup-tmp --with-timestamp \**

**--datadir=/var/lib/mysql --backup-image=/data/backups/db.mbi copy-back-and-apply-log**

[Entrypoint] MySQL Docker Image 8.0.11-1.1.5

MySQL Enterprise Backup version 8.0.11 Linux-4.1.12-61.1.16.el7uek.x86\_64-x86\_64 [2018-04-08 07:06:45]

Copyright (c) 2003, 2018, Oracle and/or its affiliates. All Rights Reserved.

180921 22:06:52 MAIN INFO: A thread created with Id '139768047519872'

180921 22:06:52 MAIN INFO: Starting with following command line ...

...

180921 22:06:52 PCR1 INFO: We were able to parse ibbackup\_logfile up to

lsn 29680612.

180921 22:06:52 PCR1 INFO: Last MySQL binlog file position 0 155, file name binlog.000003

180921 22:06:52 PCR1 INFO: The first data file is '/var/lib/mysql/ibdata1'

and the new created log files are at '/var/lib/mysql'

180921 22:06:52 MAIN INFO: No Keyring file to process.

180921 22:06:52 MAIN INFO: Apply-log operation completed successfully.

180921 22:06:52 MAIN INFO: Full Backup has been restored successfully.

mysqlbackup completed OK! with 3 warnings

一旦备份工作完成，该容器就会退出，在启动它时使用--rm选项，它退出后会被删除。

4、重新启动服务器容器，这也会重新启动恢复的服务器。

**docker restart mysqlserver**

或者，在恢复的数据目录上启动一个新的MySQL服务器。

**docker run --name=mysqlserver2 \**

**--mount type=bind,src=*/path-on-host-machine/datadir/*,dst=/var/lib/mysql \**

**-d mysql/enterprise-server:8.0**

登录到服务器上，检查服务器是否正在使用恢复后的数据运行。

已知问题

* 当使用服务器系统变量audit\_log\_file配置审计日志文件名时，请使用松散选项修改器，否则Docker将无法启动服务器。

Docker环境变量

当你创建MySQL服务器容器时，你可以通过使用--env选项（简称-e）并指定以下一个或多个环境变量来配置MySQL实例。

注意事项

* 如果你挂载的数据目录不是空的，下面的变量都没有任何作用，因为那时不会尝试进行服务器初始化（更多细节请参见持久化数据和配置更改）。文件夹中任何先前存在的内容，包括任何旧的服务器设置，在容器启动期间都不会被修改。
* 包括MYSQL\_RANDOM\_ROOT\_PASSWORD、MYSQL\_ONETIME\_PASSWORD、MYSQL\_ALLOW\_EMPTY\_PASSWORDMYSQL\_LOG\_CONSOLE在内的布尔变量，通过将它们设置为非零长度的任何字符串而使之为真。因此，将它们设置为例如 "0"、"false "或 "no "并不会使它们变成假的，而是实际上使它们变成真的。这是MySQL服务器容器的一个已知问题。
* MYSQL\_RANDOM\_ROOT\_PASSWORD：当这个变量为真时（这是它的默认状态，除非MYSQL\_ROOT\_PASSWORD被设置或MYSQL\_ALLOW\_EMPTY\_PASSWORD被设置为真），当Docker容器被启动时，会生成一个服务器根用户的随机密码。该密码被打印到容器的stdout，并且可以通过查看容器的日志找到（见启动MySQL服务器实例）。
* MYSQL\_ONETIME\_PASSWORD：当该变量为真时（这是它的默认状态，除非MYSQL\_ROOT\_PASSWORD被设置或MYSQL\_ALLOW\_EMPTY\_PASSWORD被设置为真），根用户的密码被设置为过期，必须在正常使用MySQL之前进行更改。
* MYSQL\_DATABASE：这个变量允许你指定在图像启动时创建的数据库的名称。如果在MYSQL\_USER和MYSQL\_PASSWORD中提供了用户名和密码，就会创建用户并授予该数据库的超级用户权限（对应于GRANT ALL）。指定的数据库是由CREATE DATABASE IF NOT EXIST语句创建的，所以如果数据库已经存在，这个变量就没有作用。
* MYSQL\_USER, MYSQL\_PASSWORD：这些变量结合使用，用于创建一个用户并设置该用户的密码，该用户被授予MYSQL\_DATABASE变量所指定数据库的超级用户权限。MYSQL\_USER和MYSQL\_PASSWORD都是创建用户的必要条件--如果这两个变量中的任何一个没有设置，另一个将被忽略。如果这两个变量都设置了，但MYSQL\_DATABASE没有设置，那么创建的用户就没有任何权限。

注意

没有必要使用这个机制来创建根超级用户，除非MYSQL\_ALLOW\_EMPTY\_PASSWORD为真，否则默认情况下，根超级用户的密码是由MYSQL\_ROOT\_PASSWORD和MYSQL\_RANDOM\_ROOT\_PASSWORD描述中讨论的任何一种机制设置的。

* MYSQL\_ROOT\_HOST：默认情况下，MySQL创建'root'@'localhost'账户。这个账户只能从容器内连接到，如在容器内连接到MySQL服务器中所述。要允许来自其他主机的root连接，请设置这个环境变量。例如，172.17.0.1这个值是默认的Docker网关IP，允许从运行容器的主机上进行连接。该选项只接受一个条目，但允许使用通配符（例如，MYSQL\_ROOT\_HOST=172.\*.\*.\*或MYSQL\_ROOT\_HOST=%）。
* MYSQL\_LOG\_CONSOLE：当该变量为真时（这是MySQL 8.0服务器容器的默认状态），MySQL服务器的错误日志会被重定向到stderr，这样错误日志就会进入Docker容器的日志，并可使用docker logs mysqld-tainer命令查看。

注意

如果来自主机的服务器配置文件已经被挂载，则该变量没有作用（参见关于绑定挂载配置文件的持久化数据和配置更改）。

MYSQL\_ROOT\_PASSWORD：该变量指定了为MySQL根账户设置的密码。

警告

在命令行上设置MySQL根用户密码是不安全的。作为明确指定密码的替代方法，你可以用一个密码文件的容器文件路径来设置变量，然后从你的主机挂载一个包含密码的文件到容器文件路径。这仍然不是很安全，因为密码文件的位置仍然暴露。最好是使用MYSQL\_RANDOM\_ROOT\_PASSWORD和MYSQL\_ONETIME\_PASSWORD这两个默认设置为真。

* mysql\_allow\_empty\_password。将其设置为 "true"，以允许容器以根用户的空白密码启动。

警告

将这个变量设置为 "true "是不安全的，因为它将使你的MySQL实例完全不受保护，允许任何人获得完全的超级用户访问。最好使用MYSQL\_RANDOM\_ROOT\_PASSWORD和MYSQL\_ONETIME\_PASSWORD这两个默认设置为真。

#### 2.5.6.3 用Docker在Windows和其他非Linux平台上部署MySQL

警告

甲骨文提供的MySQL Docker镜像是专门为Linux平台构建的。其他平台不受支持，用户在这些平台上运行甲骨文提供的MySQL Docker映像，其风险自负。本节讨论了在非Linux平台上使用图像时的一些已知问题。

在Windows上使用来自Oracle的MySQL服务器Docker镜像的已知问题包括。

* 如果你在容器的MySQL数据目录上进行绑定挂载（详见保存数据和配置变更），你必须用--socket选项将服务器套接字文件的位置设置为MySQL数据目录之外的地方；否则，服务器将无法启动。这是因为Docker for Windows处理文件挂载的方式不允许主机文件在套接字文件上被绑定挂载。

### 2.5.7 从本地软件库在Linux上安装MySQL

许多Linux发行版在其本地软件库中包括MySQL服务器、客户端工具和开发组件的版本，可以用平台的标准软件包管理系统进行安装。本节提供了使用这些软件包管理系统安装MySQL的基本说明。

重要事项

本地软件包通常比当前可用的版本晚几个版本。你通常也无法安装开发里程碑版本（DMR），因为这些版本通常不在本地存储库中提供。在继续之前，我们建议你检查一下第2.5节 "在Linux上安装MySQL "中描述的其他安装选项。

发行版的具体说明如下。

* Red Hat Linux, Fedora, CentOS

注意

对于一些Linux发行版，你可以使用MySQL Yum软件库而不是平台的本地软件库来安装MySQL。详情见第2.5.1节，"使用MySQL Yum仓库在Linux上安装MySQL"。

对于Red Hat和类似的发行版，MySQL发行版被分为若干独立的包，mysql用于客户端工具，mysql-server用于服务器和相关工具，mysql-libs用于库。如果你想从不同的语言和环境（如Perl、Python和其他）提供连接，则需要这些库。

要安装，使用yum命令来指定你要安装的包。比如说

root-shell> yum install mysql mysql-server mysql-libs mysql-server

Loaded plugins: presto, refresh-packagekit

Setting up Install Process

Resolving Dependencies

--> Running transaction check

---> Package mysql.x86\_64 0:5.1.48-2.fc13 set to be updated

---> Package mysql-libs.x86\_64 0:5.1.48-2.fc13 set to be updated

---> Package mysql-server.x86\_64 0:5.1.48-2.fc13 set to be updated

--> Processing Dependency: perl-DBD-MySQL for package: mysql-server-5.1.48-2.fc13.x86\_64

--> Running transaction check

---> Package perl-DBD-MySQL.x86\_64 0:4.017-1.fc13 set to be updated

--> Finished Dependency Resolution

Dependencies Resolved

================================================================================

Package Arch Version Repository Size

================================================================================

Installing:

mysql x86\_64 5.1.48-2.fc13 updates 889 k

mysql-libs x86\_64 5.1.48-2.fc13 updates 1.2 M

mysql-server x86\_64 5.1.48-2.fc13 updates 8.1 M

Installing for dependencies:

perl-DBD-MySQL x86\_64 4.017-1.fc13 updates 136 k

Transaction Summary

================================================================================

Install 4 Package(s)

Upgrade 0 Package(s)

Total download size: 10 M

Installed size: 30 M

Is this ok [y/N]: y

Downloading Packages:

Setting up and reading Presto delta metadata

Processing delta metadata

Package(s) data still to download: 10 M

(1/4): mysql-5.1.48-2.fc13.x86\_64.rpm | 889 kB 00:04

(2/4): mysql-libs-5.1.48-2.fc13.x86\_64.rpm | 1.2 MB 00:06

(3/4): mysql-server-5.1.48-2.fc13.x86\_64.rpm | 8.1 MB 00:40

(4/4): perl-DBD-MySQL-4.017-1.fc13.x86\_64.rpm | 136 kB 00:00

--------------------------------------------------------------------------------

Total 201 kB/s | 10 MB 00:52

Running rpm\_check\_debug

Running Transaction Test

Transaction Test Succeeded

Running Transaction

Installing : mysql-libs-5.1.48-2.fc13.x86\_64 1/4

Installing : mysql-5.1.48-2.fc13.x86\_64 2/4

Installing : perl-DBD-MySQL-4.017-1.fc13.x86\_64 3/4

Installing : mysql-server-5.1.48-2.fc13.x86\_64 4/4

Installed:

mysql.x86\_64 0:5.1.48-2.fc13 mysql-libs.x86\_64 0:5.1.48-2.fc13

mysql-server.x86\_64 0:5.1.48-2.fc13

Dependency Installed:

perl-DBD-MySQL.x86\_64 0:4.017-1.fc13

Complete!

现在，MySQL和MySQL服务器应该已经安装完毕。一个配置文件的样本被安装到/etc/my.cnf中。使用systemctl来启动MySQL服务器。

shell> **systemctl start mysqld**

如果数据库表还不存在，会自动为你创建。然而，你应该运行 mysql\_secure\_installation 来设置服务器上的根密码。

* Debian、Ubuntu、Kubuntu

注意

对于支持的Debian和Ubuntu版本，MySQL可以使用MySQL APT资源库而不是平台的本地软件资源库来安装。详情请参见第2.5.2节，"使用MySQL APT资源库在Linux上安装MySQL"。

在Debian和相关发行版上，在其软件仓库中有两个MySQL的软件包，即mysql-client和mysql-server，分别用于客户端和服务器组件。你应该指定一个明确的版本，例如mysql-client-5.1，以确保你安装的是你想要的MySQL的版本。

要下载和安装，包括任何依赖项，使用apt-get命令，指定你要安装的包。

注意

在安装之前，确保你更新你的apt-get索引文件，以确保你下载的是最新的可用版本。

注意

apt-get命令安装了许多软件包，包括MySQL服务器，以便提供典型的工具和应用环境。这可能意味着，除了主要的MySQL包之外，你还要安装大量的包。

在安装过程中，初始数据库被创建，并提示你输入MySQL根密码（和确认）。一个配置文件在/etc/mysql/my.cnf中被创建。在/etc/init.d/mysql中创建了一个初始脚本。

服务器应该已经被启动了。你可以使用以下方法手动启动和停止服务器。

root-shell> service mysql [start|stop]

。

该服务被自动添加到2、3和4的运行级别，停止脚本在单一、关闭和重启级别。

### 2.5.8 用Juju在Linux上安装MySQL

Juju部署框架支持MySQL服务器的简易安装和配置。有关说明，请参见https://jujucharms.com/mysql/。

### 2.5.9 用systemd管理MySQL服务器

如果你在以下Linux平台上使用RPM或Debian软件包安装MySQL，服务器的启动和关闭由systemd管理。

* RPM包平台。
  + 企业版Linux变种7及以上版本
  + USE Linux Enterprise Server 12及以上版本
  + Fedora 29及更高版本
* Debian系列平台。
  + Debian平台
  + Ubuntu平台

如果你在一个使用systemd的平台上从通用二进制发行版安装MySQL，你可以按照《MySQL 8.0安全部署指南》中安装后设置部分提供的说明，手动配置systemd对MySQL的支持。

如果你在一个使用systemd的平台上从源码发行版安装MySQL，可以通过使用-DWITH\_SYSTEMD=1 CMake选项配置发行版来获得对MySQL的系统支持。参见第2.9.7节，"MySQL源码配置选项"。

下面的讨论涵盖了这些主题。

* systemd概述
* 为MySQL配置systemd
* 使用systemd配置多个MySQL实例
* 从mysqld\_safe迁移到systemd

注意

在安装了systemd支持MySQL的平台上，mysqld\_safe和System V初始化脚本等脚本是不必要的，不会被安装。例如，mysqld\_safe可以处理服务器重启，但systemd也提供了同样的功能，而且是以与其他服务的管理一致的方式，而不是通过使用特定的应用程序。

由于systemd具有在已安装systemd支持的平台上管理多个MySQL实例的能力，所以mysqld\_multi和mysqld\_multi.server是不必要的，不需要安装。

**systemd概述**

systemd提供了MySQL服务器的自动启动和关闭。它还可以使用systemctl命令手动管理服务器。举例来说

shell> **systemctl *{start|stop|restart|status}* mysqld**

另外，使用service命令（参数相反），它与System V系统兼容。

shell> **service mysqld *{start|stop|restart|status}***

注意

对于systemctl命令（和替代的服务命令），如果MySQL服务名称不是mysqld，则使用适当的名称。例如，在基于Debian和SLES的系统上，使用mysql而不是mysqld。

对systemd的支持包括这些文件。

* mysqld.service（RPM平台），mysql.service（Debian平台）：systemd服务单元配置文件，包含MySQL服务的详细信息。
* mysqld@.service（RPM平台），mysql@.service（Debian平台）。与mysqld.service或mysql.service类似，但用于管理多个MySQL实例。
* mysqld.tmpfiles.d。文件包含支持临时文件功能的信息。该文件以mysql.conf的名义安装。
* mysqld\_pre\_systemd（RPM平台），mysql-system-start（Debian平台）。支持单元文件的脚本。该脚本仅在日志位置符合某种模式（RPM平台为/var/log/mysql\*.log，Debian平台为/var/log/mysql/\*.log）时协助创建错误日志文件。在其他情况下，错误日志目录必须是可写的，或者错误日志必须对运行mysqld进程的用户存在并可写。

**为MySQL配置systemd**

要为MySQL添加或更改systemd选项，有以下几种方法可供选择。

* 使用本地化的 systemd 配置文件。
* 安排systemd为MySQL服务器进程设置环境变量。
* 设置MYSQLD\_OPTS的systemd变量。

要使用本地化的systemd配置文件，请创建/etc/systemd/system/mysqld.service.d目录（如果它不存在）。在该目录下，创建一个包含[Service]部分的文件，列出所需设置。比如说

[Service]

LimitNOFILE=***max\_open\_files***

Nice=***nice\_level***

LimitCore=***core\_file\_limit***

Environment="LD\_PRELOAD=***/path/to/malloc/library***"

Environment="TZ=***time\_zone\_setting***"

这里的讨论使用override.conf作为该文件的名称。新版本的 systemd 支持以下命令，打开一个编辑器，允许你编辑该文件。

systemctl edit mysqld # RPM platforms

systemctl edit mysql # Debian platforms

每当你创建或修改override.conf时，都要重新加载systemd配置，然后告诉systemd重新启动MySQL服务。

systemctl daemon-reload

systemctl restart mysqld # RPM platforms

systemctl restart mysql # Debian platforms

在systemd中，某些参数必须使用override.conf配置方法，而不是在MySQL选项文件的[mysqld]、[mysqld\_safe]或[safe\_mysqld]组中设置。

* 对于某些参数，必须使用override.conf，因为systemd本身必须知道它们的值，而它无法读取MySQL选项文件来获得这些值。
* 对于那些只能用mysqld\_safe知道的选项来设置的参数，必须用systemd来指定，因为没有相应的mysqld参数。

关于使用systemd而不是mysqld\_safe的其他信息，请参见从mysqld\_safe迁移到systemd。

你可以在override.conf中设置以下参数。

* 要设置MySQL服务器可用的文件描述符的数量，在override.conf中使用LimitNOFILE，而不是mysqld的open\_files\_limit系统变量或mysqld\_safe的--open-files-limit选项。
* 要设置最大的核心文件大小，使用override.conf中的LimitCore，而不是mysqld\_safe的--核心文件大小选项。
* 要设置MySQL服务器的调度优先级，在override.conf中使用Nice而不是mysqld\_safe的--nice选项。

一些MySQL参数是用环境变量配置的。

* LD\_PRELO。如果MySQL服务器应该使用一个特定的内存分配库，则设置这个变量。
* NOTIFY\_SOCKET：这个环境变量指定了mysqld用来与systemd交流启动完成和服务状态变化通知的套接字。该变量由systemd在启动mysqld服务时设置。mysqld服务会读取该变量设置并写入定义的位置。

在MySQL 8.0中，mysqld使用Type=notify进程启动类型。(MySQL 5.7中使用Type=forking。)在Type=notify下，systemd自动配置一个套接字文件，并将路径导出到NOTIFY\_SOCKET环境变量。

* TZ：设置该变量以指定服务器的默认时区。

有多种方法可以指定环境变量值供systemd管理的MySQL服务器进程使用。

* 使用override.conf文件中的环境行。语法见前面描述如何使用该文件的讨论中的例子。
* 在/etc/sysconfig/mysql文件中指定数值（如果该文件不存在，则创建该文件）。使用以下语法指定数值。
* LD\_PRELOAD=***/path/to/malloc/library***
* TZ=***time\_zone\_setting***

在修改 /etc/sysconfig/mysql 后，重新启动服务器以使修改生效。

systemctl restart mysqld # RPM platforms

systemctl restart mysql # Debian platforms

要指定mysqld的选项而不直接修改systemd配置文件，可以设置或不设置MYSQLD\_OPTS systemd变量。例如

systemctl set-environment MYSQLD\_OPTS="--general\_log=1"

systemctl unset-environment MYSQLD\_OPTS

MYSQLD\_OPTS 也可以在 /etc/sysconfig/mysql 文件中设置。

修改systemd环境后，重新启动服务器，使修改生效。

systemctl restart mysqld # RPM platforms

systemctl restart mysql # Debian platforms

对于使用systemd的平台，如果服务器启动时是空的，那么数据目录就会被初始化。如果数据目录是一个暂时消失的远程挂载点，这可能是个问题。挂载点会显示为一个空的数据目录，然后会被初始化为一个新的数据目录。要抑制这种自动初始化行为，请在/etc/sysconfig/mysql文件中指定以下一行（如果该文件不存在，则创建该文件）。

NO\_INIT=true

**使用systemd配置多个MySQL实例**

本节介绍了如何为MySQL的多个实例配置systemd。

注意

由于systemd具有在已安装systemd支持的平台上管理多个MySQL实例的能力，所以mysqld\_multi和mysqld\_multi.server是不必要的，不需要安装。

要使用多实例功能，需要修改my.cnf选项文件，包括配置每个实例的关键选项。这些文件位置是典型的。

* /etc/my.cnf或/etc/mysql/my.cnf（RPM平台
* /etc/mysql/mysql.conf.d/mysqld.cnf (Debian平台)

例如，要管理两个名为replica01和replica02的实例，在选项文件中添加类似这样的内容。

[mysqld@replica01]

datadir=/var/lib/mysql-replica01

socket=/var/lib/mysql-replica01/mysql.sock

port=3307

log-error=/var/log/mysqld-replica01.log

[mysqld@replica02]

datadir=/var/lib/mysql-replica02

socket=/var/lib/mysql-replica02/mysql.sock

port=3308

log-error=/var/log/mysqld-replica02.log

Debian平台。

[mysqld@replica01]

datadir=/var/lib/mysql-replica01

socket=/var/lib/mysql-replica01/mysql.sock

port=3307

log-error=/var/log/mysql/replica01.log

[mysqld@replica02]

datadir=/var/lib/mysql-replica02

socket=/var/lib/mysql-replica02/mysql.sock

port=3308

log-error=/var/log/mysql/replica02.log

这里显示的副本名称使用@作为分隔符，因为这是 systemd 唯一支持的分隔符。

然后通过正常的systemd命令来管理实例，例如。

systemctl start mysqld@replica01

systemctl start mysqld@replica02

要使实例在启动时运行，请这样做。

systemctl enable mysqld@replica01

systemctl enable mysqld@replica02

也支持使用通配符。例如，该命令显示所有复制实例的状态。

systemctl status 'mysqld@replica\*'

对于同一台机器上的多个MySQL实例的管理，systemd自动使用不同的单元文件。

* mysqld@.service而不是mysqld.service（RPM平台
* mysql@.service，而不是 mysql.service（Debian平台）。

在单元文件中，%I和%i引用@标记后传入的参数，用于管理特定的实例。对于像这样的命令

systemctl start mysqld@replica01

systemd启动服务器，使用这样的命令。

mysqld --defaults-group-suffix=@%I ...

结果是[server]、[mysqld]和[mysqld@replica01]选项组被读取并用于该服务的实例。

注意

在 Debian 平台上，AppArmor 阻止服务器读取或写入 /var/lib/mysql-replica\*，或除默认位置之外的任何内容。要解决这个问题，你必须自定义或禁用 /etc/apparmor.d/usr.sbin.mysqld 中的配置文件。

注意

在 Debian 平台上，用于 MySQL 卸载的打包脚本目前不能处理 mysqld@ 实例。在删除或升级软件包之前，你必须先手动停止任何额外的实例。

**从 mysqld\_safe 迁移到 systemd**

因为mysqld\_safe不安装在使用systemd管理MySQL的平台上，所以以前为该程序指定的选项（例如，在[mysqld\_safe]或[safe\_mysqld]选项组中）必须以其他方式指定。

* 一些mysqld\_safe选项也能被mysqld理解，可以从[mysqld\_safe]或[safe\_mysqld]选项组移到[mysqld]组。这不包括--pid-file、--open-files-limit或--nice。要指定这些选项，请使用override.conf systemd文件，如前所述。

注意

在systemd平台上，不支持使用[mysqld\_safe]和[safe\_mysqld]选项组，可能会导致意外行为。

* 对于某些 mysqld\_safe 选项，有替代的 mysqld 程序。例如，启用syslog日志的mysqld\_safe选项是--syslog，它已被废弃。要将错误日志输出写入系统日志，请使用第5.4.2.8节 "将错误日志写入系统日志 "中的说明。
* mysqld\_safe选项不被mysqld理解，可以在override.conf或环境变量中指定。例如，在mysqld\_safe中，如果服务器应该使用一个特定的内存分配库，可以使用--malloc-lib选项指定。对于用systemd管理服务器的安装，安排设置LD\_PRELOAD环境变量代替，如前所述。

## 2.6 使用Unbreakable Linux Network（ULN）安装MySQL

Linux支持许多不同的解决方案来安装MySQL，这在第2.5节 "在Linux上安装MySQL "中有所涉及。本节涉及的其中一种方法是从Oracle的Unbreakable Linux Network（ULN）中安装。你可以在http://linux.oracle.com/ 下找到关于Oracle Linux和ULN的信息。

要使用ULN，你需要获得一个ULN登录账号，并在ULN上注册用于安装的机器。这在 ULN FAQ 中有详细描述。该页面还介绍了如何安装和更新软件包。

社区包和商业包都被支持，每个包都提供三个MySQL通道。

* 服务器。MySQL服务器
* 连接器。MySQL Connector/C++、MySQL Connector/J、MySQL Connector/ODBC和MySQL Connector/Python。
* 工具。MySQL Router、MySQL Shell和MySQL Workbench

所有ULN用户都可以使用社区频道。

在oracle.linux.com访问商业的MySQL ULN包需要你提供一个具有MySQL（企业或标准）有效商业许可证的CSI。截至本文撰写时，有效的购买是60944、60945、64911和64912。适当的CSI会在你的ULN GUI界面中提供商业MySQL订阅渠道。

一旦使用ULN安装了MySQL，你可以在第2.5.7节 "从本地软件库安装Linux上的MySQL"，特别是在第2.5.4节 "使用来自Oracle的RPM包在Linux上安装MySQL "中找到关于启动和停止服务器等的信息。

如果你正在改变你的包的来源以使用ULN，而不改变你正在使用的MySQL的构建，那么备份你的数据，删除你现有的二进制文件，并用来自ULN的二进制文件替换它们。如果涉及到构建的改变，我们建议备份是一个转储（mysqldump或mysqlpump或来自MySQL Shell的备份工具），以备你在新的二进制文件到位后需要重建你的数据。如果向ULN的转变跨越了版本边界，请在继续进行之前查阅本节。第2.11节，"升级MySQL"。

注意

Oracle Linux 8从MySQL 8.0.17开始被支持，社区工具和连接器通道在MySQL 8.0.24发布时被添加。

## 2.7 在Solaris上安装MySQL

2.7.1 使用Solaris PKG在Solaris上安装MySQL

注意

MySQL 8.0支持Solaris 11.4及以上版本

Solaris上的MySQL有许多不同的格式。

关于使用本地Solaris PKG格式进行安装的信息，请参阅第2.7.1节 "使用Solaris PKG在Solaris上安装MySQL"。

要使用标准的tar二进制安装，请使用第2.2节 "使用通用二进制文件在Unix/Linux上安装MySQL "中提供的说明。检查本节末尾的说明和提示，了解安装前后可能需要的Solaris特定说明。

重要事项

安装包对Oracle Developer Studio 12.6 Runtime Libraries有依赖性，在运行MySQL安装包之前必须安装该程序。请参阅这里的Oracle Developer Studio的下载选项。该安装包使你能够只安装运行时库而不是完整的Oracle Developer Studio；请参阅在Oracle Solaris 11上只安装运行时库中的说明。

要获得Solaris的tarball或PKG格式的二进制MySQL分布，https://dev.mysql.com/downloads/mysql/8.0.html。

在Solaris上安装和使用MySQL时应注意的其他注意事项。

如果你想用mysql用户和组来使用MySQL，请使用groupadd和useradd命令。

groupadd mysql

useradd -g mysql -s /bin/false mysql

如果你在Solaris上使用二进制tar球发行版安装MySQL，因为Solaris tar不能处理长文件名，使用GNU tar（gtar）来解压发行版。如果你的系统上没有GNU tar，请用以下命令安装它。

pkg install archiver/gnu-tar

你应该用forcedirectio选项来挂载你打算存储InnoDB文件的任何文件系统。在这个平台上使用InnoDB存储引擎时，如果不这样做会导致性能的显著下降。

如果你想让MySQL自动启动，你可以把support-files/mysql.server复制到/etc/init.d，并为它创建一个符号链接，命名为/etc/rc3.d/S99mysql.server。

如果太多的进程试图非常迅速地连接到mysqld，你应该在MySQL日志中看到这个错误。

Error in accept: Protocol error

你可以尝试用-back\_log=50选项启动服务器，作为解决这个问题的方法。

要在Solaris上配置核心文件的生成，你应该使用coreadm命令。由于在setuid()程序上生成核心的安全问题，默认情况下，Solaris不支持在setuid()程序上生成核心文件。然而，你可以使用coreadm修改这一行为。如果你为当前用户启用了setuid()核心文件，它们将使用模式600生成，并由超级用户拥有。

### 2.7.1 使用Solaris PKG在Solaris上安装MySQL

你可以使用原生Solaris PKG格式的二进制包在Solaris上安装MySQL，而不是使用二进制tarball分发。

重要的是安装包对Oracle Developer Studio 12.6 Runtime Libraries有依赖性，在运行MySQL安装包之前必须安装该程序。请参阅这里的Oracle Developer Studio的下载选项。该安装包使你能够只安装运行时库，而不是完整的Oracle Developer Studio；请参阅在Oracle Solaris 11上只安装运行时库的说明。

要使用这个包，请下载相应的mysql-VERSION-solaris11-PLATFORM.pkg.gz文件，然后将其解压。例如

shell> **gunzip mysql-*8.0.26*-solaris11-x86\_64.pkg.gz**

要安装一个新的软件包，使用 pkgadd 并按照屏幕上的提示操作。你必须有root权限来执行这个操作。

shell> **pkgadd -d mysql-*8.0.26*-solaris11-x86\_64.pkg**

The following packages are available:

1 mysql MySQL Community Server (GPL)

(i86pc) 8.0.26

Select package(s) you wish to process (or 'all' to process

all packages). (default: all) [?,??,q]:

PKG安装程序会安装所有需要的文件和工具，如果你的数据库不存在，则初始化数据库。为了完成安装，你应该按照安装结束时的说明，设置MySQL的根密码。或者，你可以运行安装时附带的mysql\_secure\_installation脚本。

默认情况下，PKG包将MySQL安装在根路径/opt/mysql下。当使用pkgadd时，你可以只改变安装根路径，它可以用来在不同的Solaris区域安装MySQL。如果你需要安装在一个特定的目录下，请使用二进制tar文件分发。

pkg安装程序为MySQL复制一个合适的启动脚本到/etc/init.d/mysql。为了使MySQL能够自动启动和关闭，你应该在这个文件和init脚本目录之间建立一个链接。例如，为了确保MySQL的安全启动和关闭，你可以使用以下命令来添加正确的链接。

shell> **ln /etc/init.d/mysql /etc/rc3.d/S91mysql**

shell> **ln /etc/init.d/mysql /etc/rc0.d/K02mysql**

要删除MySQL，安装的软件包名称是mysql。你可以将其与pkgrm命令结合起来使用来删除安装。

在使用Solaris软件包文件格式时要进行升级，在安装更新的软件包之前必须删除现有的安装。移除软件包并不删除现有的数据库信息，只是删除服务器、二进制文件和支持文件。因此，典型的升级顺序是

shell> **mysqladmin shutdown**

shell> **pkgrm mysql**

shell> **pkgadd -d mysql-*8.0.26*-solaris11-x86\_64.pkg**

shell> **mysqld\_safe &**

shell> **mysql\_upgrade** # prior to MySQL 8.0.16 only

在执行任何升级之前，你应该检查第2.11节 "升级MySQL "中的说明。

## 2.8 在FreeBSD上安装MySQL

本节提供关于在FreeBSD Unix的变体上安装MySQL的信息。

你可以通过使用Oracle提供的二进制分布在FreeBSD上安装MySQL。欲了解更多信息，请参见第2.2节，"在Unix/Linux上使用通用二进制文件安装MySQL"。

安装MySQL的最简单（也是首选）方式是使用http://www.freebsd.org/ 上的mysql-server和mysql-client端口。使用这些端口会给你带来以下好处。

* 一个可以工作的MySQL，并启用所有已知可以在你的FreeBSD版本上工作的优化功能。
* 自动配置和构建。
* 安装在 /usr/local/etc/rc.d 的启动脚本。
* 能够使用 pkg\_info -L 来查看哪些文件被安装。
* 如果你不再需要MySQL在你的机器上，可以使用pkg\_delete来删除它。

MySQL的构建过程需要GNU make（gmake）来工作。如果GNU make不可用，你必须在编译MySQL之前先安装它。

注意

根据ldd mysqld的前提库：libthr, libcrypt, libkrb5, libm, librt, libexecinfo, libunwind, and libssl。

要使用 ports 系统进行安装。

# cd /usr/ports/databases/mysql80-server

# make

...

# cd /usr/ports/databases/mysql80-client

# make

标准端口安装将服务器放在/usr/local/libexec/mysqld中，MySQL服务器的启动脚本放在/usr/local/etc/rc.d/mysql-server中。

关于BSD实现的一些附加说明。

在使用 ports 系统安装后要删除 MySQL。

# cd /usr/ports/databases/mysql80-server

# make deinstall

...

# cd /usr/ports/databases/mysql80-client

# make deinstall

...

...

如果你在MySQL中遇到当前日期的问题，设置TZ变量应该有帮助。见第4.9节，"环境变量"。

## 2.9 从源代码安装MySQL

从源代码构建MySQL使你可以定制构建参数、编译器优化和安装位置。关于已知可以运行MySQL的系统的列表，见https://www.mysql.com/support/supportedplatforms/database.html。

在你从源码进行安装之前，请检查Oracle是否为你的平台生产了预编译的二进制发行版，以及它是否对你有效。我们花了很大的力气来确保我们的二进制文件是用最好的选项构建的，以获得最佳性能。在第2.2节 "使用通用二进制文件在Unix/Linux上安装MySQL "中提供了安装二进制发行版的说明。

如果你对使用与Oracle在你的平台上生产二进制发行版相同或类似的构建选项从源发行版构建MySQL感兴趣，可以获得一个二进制发行版，将其解压，并查看docs/INFO\_BIN文件，其中包含有关该MySQL发行版如何配置和编译的信息。

警告

用非标准选项构建MySQL可能导致功能、性能或安全性降低。

MySQL源代码包含使用Doxygen编写的内部文档。生成的Doxygen内容可在https://dev.mysql.com/doc/index-other.html。也可以使用第2.9.10节 "生成MySQL Doxygen文档内容 "中的说明，从MySQL源分发中本地生成这些内容。

### 2.9.1 源码安装方法

有两种方法可以从源码安装MySQL。

* 使用标准的MySQL源码发行版。要获得一个标准的发行版，见第2.1.3节 "如何获得MySQL"。关于从标准发行版构建的说明，见第2.9.4节，"使用标准源发行版安装MySQL"。

标准发行版可作为压缩的tar文件、Zip档案或RPM包提供。发行文件的名称是mysql-VERSION.tar.gz、mysql-VERSION.zip或mysql-VERSION.rpm，其中VERSION是一个数字，如8.0.26。源码发行版的文件名可以与预编译的二进制发行版的文件名区分开来，因为源码发行版的文件名是通用的，不包括平台名称，而二进制发行版的文件名包括一个平台名称，表明该发行版要用于的系统类型（例如，pc-linux-i686或winx64）。

* 使用MySQL开发树。有关从某个开发树中构建的信息，请参见第2.9.5节 "使用开发源码树安装MySQL"。

### 2.9.2 源码安装的先决条件

从源码安装MySQL需要几个开发工具。无论你使用的是标准源码分发还是开发源码树，其中一些工具都是需要的。其他工具的要求则取决于你使用的安装方法。

要从源码安装MySQL，无论采用何种安装方法，都必须满足以下系统要求。

* CMake，它被用作所有平台上的构建框架。CMake可以从http://www.cmake.org 下载。
* 一个好的make程序。尽管有些平台有自己的make实现，但强烈建议你使用GNU make 3.75或更高版本。它可能已经在你的系统上以gmake的形式存在。GNU make可从<http://www.gnu.org/software/make/>。
* MySQL 8.0源代码允许使用C++14特性。为了在所有支持的平台上实现良好的C++14支持水平，适用以下最小编译器版本。
  + Linux。GCC 5.3.1（Ubuntu 16.04），GCC 10（EL7，EL8
  + FreeBSD。Clang 4.0.0
  + macOS：XCode 9；但只有Oracle用于构建二进制包的版本才被保证支持。
  + Solaris。Developer Studio 12.6
  + Windows。Visual Studio 2019
* MySQL C API需要一个C++或C99编译器来编译。
* 需要一个SSL库来支持加密连接、用于随机数生成的熵以及其他与加密相关的操作。默认情况下，编译时使用安装在主机系统上的OpenSSL库。要明确指定该库，请在调用CMake时使用WITH\_SSL选项。有关其他信息，请参见第2.9.6节 "配置SSL库支持"。
* 构建MySQL需要Boost C++库（但不是为了使用它）。MySQL的编译需要一个特定的Boost版本。通常情况下，那是当前的Boost版本，但如果一个特定的MySQL源码发行版需要不同的版本，配置过程就会停止，并显示出它所需要的Boost版本。要获得Boost及其安装说明，请访问官方网站。Boost安装完毕后，通过在调用CMake时定义WITH\_BOOST选项，告诉构建系统Boost文件的位置。比如说

cmake . -DWITH\_BOOST=/usr/local/boost\_***version\_number***

根据需要调整路径以配合你的安装。

* ncurses库。
* 足够的自由内存。如果你在编译大的源文件时遇到 "内部编译器错误 "等问题，可能是你的内存太少。如果在虚拟机上进行编译，可以尝试增加内存分配。
* 如果你打算运行测试脚本，就需要Perl。大多数类似Unix的系统包括Perl。在Windows上，你可以使用ActiveState Perl这样的版本。

要从一个标准的源码分布中安装MySQL，需要使用以下工具之一来解压分布文件。

* 对于一个.tar.gz压缩的tar文件。GNU gunzip来解压发行文件，并使用合理的tar来解压。如果你的 tar 程序支持 z 选项，它可以同时解压缩和解压文件。

GNU tar 是已知的工作。一些操作系统提供的标准tar不能解压MySQL发行版中的长文件名。你应该下载并安装GNU tar，或者如果有的话，使用预装的GNU tar版本。通常这可以作为gnutar、gtar，或作为GNU或自由软件目录下的tar，如/usr/sfw/bin或/usr/local/bin。GNU tar可以从http://www.gnu.org/software/tar/。

* 对于一个.zip的压缩文件。WinZip或其他可以读取.zip文件的工具。
* 对于一个.rpm的RPM包。用来构建发行版的rpmbuild程序将其解压。

要从开发源码树安装MySQL，需要以下额外的工具。

* 需要Git修订控制系统来获取开发源代码。GitHub帮助提供了在不同平台上下载和安装Git的说明。MySQL于2014年9月正式加入GitHub。关于MySQL迁移到GitHub的更多信息，请参考MySQL发布工程博客上的公告。MySQL在GitHub上
* bison 2.1或更高版本，可从http://www.gnu.org/software/bison/。(第1版已不再支持。)尽可能使用最新版本的bison；如果你遇到问题，请升级到较高版本，而不是恢复到较早版本。

bison可以从http://www.gnu.org/software/bison/。bison for Windows可以从http://gnuwin32.sourceforge.net/packages/bison.htm。下载标有 "完整包，不包括源代码 "的软件包。在Windows上，bison的默认位置是C:\Program Files\GnuWin32目录。由于目录名称中有空格，有些工具可能找不到bison。另外，如果路径中存在空格，Visual Studio可能会直接挂起。你可以通过安装到一个不包含空格的目录（例如C:\GnuWin32）来解决这些问题。

* 在Solaris Express上，除了bison，还必须安装m4。m4可以从http://www.gnu.org/software/m4/。

注意

如果你必须安装任何程序，请修改你的PATH环境变量，以包括这些程序所在的任何目录。见第4.2.9节，"设置环境变量"。

如果你遇到问题并需要提交错误报告，请使用1.6节 "如何报告错误或问题 "中的说明。

### 2.9.3 源码安装的MySQL布局

默认情况下，当你从源码编译后安装MySQL时，安装步骤会在/usr/local/mysql下安装文件。安装目录下的组件位置与二进制发行版相同。参见表2.3，"通用Unix/Linux二进制包的MySQL安装布局"，以及第2.3.1节，"Microsoft Windows上的MySQL安装布局"。要配置不同于默认值的安装位置，请使用第2.9.7节 "MySQL源配置选项 "中描述的选项。

### 2.9.4 使用标准源发布版安装MySQL

要从一个标准的源码发行版安装MySQL。

1、确认你的系统满足第2.9.2节 "源码安装先决条件 "中列出的工具要求。

2、使用第2.1.3节 "如何获得MySQL "中的说明获得一个分发文件。

3、使用本节中的说明配置、构建和安装分发文件。

4、使用第2.10节 "安装后设置和测试 "中的说明执行安装后程序。

MySQL在所有平台上使用CMake作为构建框架。这里给出的说明应能使你产生一个工作的安装。关于使用CMake构建MySQL的其他信息，请参阅如何用CMake构建MySQL服务器。

如果你从一个源码RPM开始，使用下面的命令制作一个你可以安装的二进制RPM。如果你没有rpmbuild，用rpm代替。

shell> **rpmbuild --rebuild --clean MySQL-*VERSION*.src.rpm**

结果是一个或多个二进制RPM包，你可以按照第2.5.4节 "使用来自Oracle的RPM包在Linux上安装MySQL "中的指示进行安装。

从压缩的tar文件或Zip归档的源发布文件进行安装的顺序类似于从通用二进制发布文件进行安装的过程（见第2.2节，"使用通用二进制文件在Unix/Linux上安装MySQL"），只是它用于所有平台，包括配置和编译发布文件的步骤。例如，在Unix上使用压缩的tar文件源发布，基本的安装命令序列看起来像这样。

# Preconfiguration setup

shell> **groupadd mysql**

shell> **useradd -r -g mysql -s /bin/false mysql**

# Beginning of source-build specific instructions

shell> **tar zxvf mysql-*VERSION*.tar.gz**

shell> **cd mysql-*VERSION***

shell> **mkdir bld**

shell> **cd bld**

shell> **cmake ..**

shell> **make**

shell> **make install**

# End of source-build specific instructions

# Postinstallation setup

shell> **cd /usr/local/mysql**

shell> **mkdir mysql-files**

shell> **chown mysql:mysql mysql-files**

shell> **chmod 750 mysql-files**

shell> **bin/mysqld --initialize --user=mysql**

shell> **bin/mysql\_ssl\_rsa\_setup**

shell> **bin/mysqld\_safe --user=mysql &**

# Next command is optional

shell> **cp support-files/mysql.server /etc/init.d/mysql.server**

下面是一个更详细的源码构建具体说明的版本。

注意

这里显示的程序并没有为MySQL账户设置任何密码。按照该程序操作后，进入第2.10节 "安装后设置和测试"，进行安装后设置和测试。

* 执行预配置设置
* 获取并解压发行版
* 配置该发行版
* 构建发行版
* 安装发行版
* 执行安装后设置

**执行预配置设置**

在Unix上，设置mysql用户和组，用于运行和执行MySQL服务器，并拥有数据库目录。详情请参见创建mysql用户和组。然后以mysql用户的身份执行下列步骤，除非有说明。

获取并解压发行版

选择你想解压发行版的目录，并将位置改到该目录下。

使用第2.1.3节 "如何获得MySQL "中的说明获得一个分发文件。

将分发文件解压到当前目录中。

* 要解压一个压缩的tar文件，如果tar有z选项支持，它可以解压并解压发行版。

shell> **tar zxvf mysql-*VERSION*.tar.gz**

如果你的 tar 没有 z 选项支持，使用 gunzip 来解压发行版，然后用 tar 来解压。

shell> **gunzip < mysql-*VERSION*.tar.gz | tar xvf -**

或者，CMake可以解压和解压发行版。

shell> **cmake -E tar zxvf mysql-*VERSION*.tar.gz**

* 要解压Zip档案，请使用WinZip或其他可以读取.zip文件的工具。

解压后的发行文件会创建一个名为mysql-VERSION的目录。

**配置分发文件**

改变位置进入解压后的发行文件的顶层目录。

shell> **cd mysql-*VERSION***

在源码树之外构建，以保持源码树的清洁。如果顶级源码目录在你当前工作目录下被命名为 mysql-src，你可以在同一级别的名为 bld 的目录中构建。创建该目录并进入那里。

shell> **mkdir bld**

shell> **cd bld**

配置构建目录。最小配置命令不包括覆盖配置默认值的选项。

shell> **cmake ../mysql-src**

构建目录不需要在源树之外。例如，你可以在顶层源代码树下名为bld的目录中进行构建。要做到这一点，从mysql-src作为你的当前工作目录开始，创建目录bld，然后去那里。

shell> **mkdir bld**

shell> **cd bld**

配置构建目录。最小配置命令不包括覆盖配置默认值的选项。

shell> **cmake ..**

如果你在同一级别有多个源树（例如，要构建多个版本的MySQL），第二种策略可能是有利的。第一种策略将所有构建目录放在同一级别，这要求你为每个目录选择一个独特的名字。使用第二种策略，你可以在每个源树中使用相同的名字作为构建目录。下面的说明是假设采用第二种策略。

在Windows上，指定开发环境。例如，下面的命令分别为32位或64位构建配置MySQL。

shell> **cmake .. -G "Visual Studio 12 2013"**

shell> **cmake .. -G "Visual Studio 12 2013 Win64"**

在macOS上，要使用Xcode IDE。

shell> **cmake .. -G Xcode**

当你运行cmake时，你可能想在命令行中添加选项。这里有一些例子。

* -DBUILD\_CONFIG=mysql\_release: 用Oracle用于生产官方MySQL版本的二进制发行版的相同构建选项来配置源。
* -DCMAKE\_INSTALL\_PREFIX=dir\_name：配置发行版以安装在一个特定的位置。
* -DCPACK\_MONOLITHIC\_INSTALL=1：使make包生成一个安装文件而不是多个文件。
* -DWITH\_DEBUG=1: 用调试支持来构建发行版。

关于更多的选项列表，见第2.9.7节 "MySQL源配置选项"。

要列出配置选项，请使用下列命令之一。

shell> **cmake .. -L** # overview

shell> **cmake .. -LH** # overview with help text

shell> **cmake .. -LAH** # all params with help text

shell> **ccmake ..** # interactive display

如果CMake失败了，你可能需要用不同的选项再次运行它来重新配置。如果你重新配置，请注意以下几点。

* 如果CMake在之前被运行之后被运行，它可能会使用在之前的调用中收集的信息。这些信息被保存在CMakeCache.txt中。当CMake启动时，它会寻找该文件并读取其内容（如果它存在），前提是这些信息仍然正确。这个假设在你重新配置时是无效的。
* 每次运行CMake时，你必须再次运行make来重新编译。然而，你可能想先删除以前构建的旧对象文件，因为它们是用不同的配置选项编译的。

为了防止旧的对象文件或配置信息被使用，在重新运行CMake之前，在Unix的构建方向上运行这些命令。

shell> **make clean**

shell> **rm CMakeCache.txt**

或者，在Windows上。

shell> **devenv MySQL.sln /clean**

shell> **del CMakeCache.txt**

在MySQL社区Slack上询问之前，请检查CMakeFiles目录中的文件，以获取关于失败的有用信息。要提交错误报告，请使用1.6节 "如何报告错误或问题 "中的说明。

**构建发行版**

在 Unix 上。

shell> **make**

shell> **make VERBOSE=1**

第二条命令设置VERBOSE来显示每个编译后的源代码的命令。

在使用GNU make并且已经安装了gmake的系统上，使用gmake代替。

在Windows上。

shell> **devenv MySQL.sln /build RelWithDebInfo**

如果你已经到了编译阶段，但发行版没有编译，请参阅第2.9.8节 "处理编译MySQL的问题"，以获得帮助。如果这还不能解决问题，请使用第1.6节 "如何报告错误或问题 "中的说明将其输入我们的错误数据库。如果你已经安装了所需工具的最新版本，而它们在试图处理我们的配置文件时崩溃，也请报告。然而，如果你得到一个未找到命令的错误或所需工具的类似问题，请不要报告它。相反，请确保所有需要的工具都已安装，并且你的PATH变量设置正确，以便你的shell可以找到它们。

**安装发行版**

在Unix上。

shell> **make install**

这将在配置的安装目录下安装文件（默认为/usr/local/mysql）。你可能需要以root身份运行该命令。

要在一个特定的目录下安装，可以在命令行中添加一个DESTDIR参数。

shell> **make install DESTDIR="/opt/mysql"**

或者，生成安装包文件，你可以安装在你喜欢的地方。

shell> **make package**

这个操作会产生一个或多个.tar.gz文件，可以像一般的二进制发布包一样安装。见第2.2节，"使用通用二进制文件在Unix/Linux上安装MySQL"。如果你用-DCPACK\_MONOLITHIC\_INSTALL=1运行CMake，该操作产生一个文件。否则，它将产生多个文件。

在Windows上，生成数据目录，然后创建一个.zip档案安装包。

shell> **devenv MySQL.sln /build RelWithDebInfo /project initial\_database**

shell> **devenv MySQL.sln /build RelWithDebInfo /project package**

你可以把生成的.zip压缩包安装到你喜欢的地方。参见第2.3.4节，"使用noinstall ZIP档案在Microsoft Windows上安装MySQL"。

**执行安装后的设置**

安装过程的剩余部分包括设置配置文件、创建核心数据库和启动MySQL服务器。有关说明，见第2.10节 "安装后设置和测试"。

注意

列在MySQL授予表中的账户最初没有密码。在启动服务器后，你应该使用第2.10节 "安装后设置和测试 "中的说明为它们设置密码。

### 2.9.5 使用开发源码树安装MySQL

本节描述了如何从GitHub上托管的最新开发源代码安装MySQL。为了从这个源码托管服务中获得MySQL服务器源码，你可以设置一个本地MySQL Git库。

在GitHub上，MySQL Server和其他MySQL项目可以在MySQL页面找到。MySQL Server项目是一个单一的仓库，包含了几个MySQL系列的分支。

MySQL于2014年9月正式加入GitHub。关于MySQL迁移到GitHub的更多信息，请参考MySQL发布工程博客上的公告。MySQL在GitHub上

* 从开发源码安装的先决条件
* 设置MySQL的Git存储库

**从开发源安装的先决条件**

要从开发源码树安装MySQL，你的系统必须满足第2.9.2节 "源码安装先决条件 "中列出的工具要求。

**设置一个MySQL的Git存储库**

要在你的机器上建立一个MySQL Git仓库。

1、克隆MySQL Git 仓库到你的机器上。下面的命令将MySQL Git仓库克隆到名为mysql-server的目录中。最初的下载可能需要一些时间来完成，这取决于你的连接速度。

$ git clone https://github.com/mysql/mysql-server.git

Cloning into 'mysql-server'...

remote: Counting objects: 1198513, done.

remote: Total 1198513 (delta 0), reused 0 (delta 0), pack-reused 1198513

Receiving objects: 100% (1198513/1198513), 1.01 GiB | 7.44 MiB/s, done.

Resolving deltas: 100% (993200/993200), done.

Checking connectivity... done.

Checking out files: 100% (25510/25510), done.

2、当克隆操作完成后，你的本地 MySQL Git 仓库的内容会出现类似于下面的情况。

~$ cd mysql-server

~/mysql-server$ ls

client extra mysys storage

cmake include packaging strings

CMakeLists.txt INSTALL plugin support-files

components libbinlogevents README testclients

config.h.cmake libbinlogstandalone router unittest

configure.cmake libmysql run\_doxygen.cmake utilities

Docs libservices scripts VERSION

Doxyfile-ignored LICENSE share vio

Doxyfile.in man sql win

doxygen\_resources mysql-test sql-common

3、使用git branch -r命令查看MySQL存储库的远程跟踪分支。

~/mysql-server$ git branch -r

origin/5.5

origin/5.6

origin/5.7

origin/8.0

origin/HEAD -> origin/8.0

origin/cluster-7.2

origin/cluster-7.3

origin/cluster-7.4

origin/cluster-7.5

origin/cluster-7.6

4、要查看你本地仓库中签出的分支，请发出git branch命令。当你克隆MySQL Git仓库时，最新的MySQL GA分支会被自动签出。星号标识了活动分支。

~/mysql-server$ git branch

\* 8.0

5、要签出一个较早的MySQL分支，运行git checkout命令，指定分支名称。例如，签出MySQL 5.7分支。

~/mysql-server$ git checkout 5.7

Checking out files: 100% (9600/9600), done.

Branch 5.7 set up to track remote branch 5.7 from origin.

Switched to a new branch '5.7'

6、要获得在你最初设置MySQL Git仓库后所做的修改，请切换到你想更新的分支，并发出git pull命令。

~/mysql-server$ git checkout 8.0

~/mysql-server$ git pull

要检查提交历史，请使用git log选项。

~/mysql-server$ git log

你也可以在GitHub MySQL网站上浏览提交历史和源代码。

如果你看到有问题的变化或代码，请在MySQL社区Slack上提问。关于贡献补丁的信息，请参见贡献给MySQL服务器。

7、在你克隆了MySQL的Git仓库并检查出你想要构建的分支后，你可以从源代码构建MySQL服务器。第2.9.4节 "使用标准源码发行版安装MySQL "中提供了说明，只是你跳过了关于获取和解压发行版的部分。

在生产机器上安装来自发行版源码树的构建时要小心。安装命令可能会覆盖你的实时发布安装。如果你已经安装了MySQL并且不想覆盖它，运行CMake时，CMAKE\_INSTALL\_PREFIX、MYSQL\_TCP\_PORT和MYSQL\_UNIX\_ADDR选项的值与你的生产服务器使用的不同。关于防止多个服务器相互干扰的其他信息，请参见第5.8节，"在一台机器上运行多个MySQL实例"。

在你的新安装中努力发挥。例如，尝试使新功能崩溃。从运行make test开始。见《MySQL测试套件》。

### 2.9.6 配置SSL库支持

需要一个SSL库来支持加密连接、用于随机数生成的熵以及其他与加密有关的操作。

如果你从一个源码发行版编译MySQL，CMake将发行版配置为默认使用已安装的OpenSSL库。

要使用OpenSSL进行编译，请使用这个程序。

1、确保OpenSSL 1.0.1或更高版本已安装在你的系统上。如果安装的OpenSSL版本低于1.0.1，CMake会在MySQL配置时产生一个错误。如果有必要获得OpenSSL，请访问http://www.openssl.org。

2、WITH\_SSL CMake选项决定在编译MySQL时使用哪个SSL库（见第2.9.7节，"MySQL源配置选项"）。默认是-DWITH\_SSL=system，它使用OpenSSL。要明确这一点，请在CMake命令行中指定该选项。例如

cmake . -DWITH\_SSL=system

该命令将发行版配置为使用已安装的OpenSSL库。另外，要明确指定OpenSSL安装的路径名称，请使用以下语法。如果你安装了多个版本的OpenSSL，这可能很有用，可以防止CMake选择错误的版本。

cmake . -DWITH\_SSL=***path\_name***

3、编译并安装该发行版。

要检查一个mysqld服务器是否支持加密连接，检查have\_ssl系统变量的值。

mysql> **SHOW VARIABLES LIKE 'have\_ssl';**

+---------------+-------+

| Variable\_name | Value |

+---------------+-------+

| have\_ssl | YES |

+---------------+-------+

如果值为YES，服务器支持加密连接。如果值为DISABLED，则服务器能够支持加密连接，但在启动时没有使用适当的--ssl-xxx选项来启用加密连接；见第6.3.1节，"配置MySQL使用加密连接"。

### 2.9.7 MySQL源配置选项

CMake程序对你如何配置MySQL源码分发提供了大量的控制。通常，你使用CMake命令行上的选项来做这件事。关于CMake支持的选项的信息，请在顶级源码目录下运行这些命令中的任何一个。

cmake . -LH

ccmake .

你也可以使用某些环境变量来影响CMake。参见第4.9节 "环境变量"。

对于布尔选项，值可以指定为1或ON以启用该选项，或者指定为0或OFF以禁用该选项。

许多选项配置了编译时的默认值，可以在服务器启动时被覆盖。例如，CMAKE\_INSTALL\_PREFIX、MYSQL\_TCP\_PORT和MYSQL\_UNIX\_ADDR选项配置了默认的安装基础目录位置、TCP/IP端口号和Unix套接字文件，可以在服务器启动时通过mysqld的-basedir、-port和-socket选项进行更改。在适用的地方，配置选项描述指出了相应的mysqld启动选项。

下面的章节提供了关于CMake选项的更多信息。

CMake选项参考

* 一般选项
* 安装布局选项
* 存储引擎选项
* 功能选项
* 编译器标志
* 用于编译NDB Cluster的CMake选项

**CMake选项参考**

下表显示了可用的CMake选项。在默认列中，PREFIX代表CMAKE\_INSTALL\_PREFIX选项的值，它指定了安装基础目录。这个值被用作若干安装子目录的父目录。

**Table 2.14 MySQL Source-Configuration Option Reference (CMake)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Formats** | **Description** | **Default** | **Introduced** | **Removed** |
| [**ADD\_GDB\_INDEX**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_add_gdb_index) | Whether to enable generation of .gdb\_index section in binaries |  | 8.0.18 |  |
| [**BUILD\_CONFIG**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_build_config) | Use same build options as official releases |  |  |  |
| [**BUNDLE\_RUNTIME\_LIBRARIES**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_bundle_runtime_libraries) | Bundle runtime libraries with server MSI and Zip packages for Windows | **OFF** |  |  |
| [**CMAKE\_BUILD\_TYPE**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_cmake_build_type) | Type of build to produce | **RelWithDebInfo** |  |  |
| [**CMAKE\_CXX\_FLAGS**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_cmake_cxx_flags) | Flags for C++ Compiler |  |  |  |
| [**CMAKE\_C\_FLAGS**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_cmake_c_flags) | Flags for C Compiler |  |  |  |
| [**CMAKE\_INSTALL\_PREFIX**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_cmake_install_prefix) | Installation base directory | **/usr/local/mysql** |  |  |
| **CMAKE\_INSTALL\_PRIV\_LIBDIR** | Installation private library directory |  | 8.0.18 |  |
| [**COMPILATION\_COMMENT**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_compilation_comment) | Comment about compilation environment |  |  |  |
| [**COMPILATION\_COMMENT\_SERVER**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_compilation_comment_server) | Comment about compilation environment for use by mysqld |  | 8.0.14 |  |
| [**COMPRESS\_DEBUG\_SECTIONS**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_compress_debug_sections) | Compress debug sections of binary executables | **OFF** | 8.0.22 |  |
| [**CPACK\_MONOLITHIC\_INSTALL**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_cpack_monolithic_install) | Whether package build produces single file | **OFF** |  |  |
| [**DEFAULT\_CHARSET**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_default_charset) | The default server character set | **utf8mb4** |  |  |
| [**DEFAULT\_COLLATION**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_default_collation) | The default server collation | **utf8mb4\_0900\_ai\_ci** |  |  |
| **DISABLE\_DATA\_LOCK** | Exclude the performance schema data lock instrumentation | **OFF** |  |  |
| [**DISABLE\_PSI\_COND**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_disable_psi_cond) | Exclude Performance Schema condition instrumentation | **OFF** |  |  |
| [**DISABLE\_PSI\_ERROR**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_disable_psi_error) | Exclude the performance schema server error instrumentation | **OFF** |  |  |
| [**DISABLE\_PSI\_FILE**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_disable_psi_file) | Exclude Performance Schema file instrumentation | **OFF** |  |  |
| [**DISABLE\_PSI\_IDLE**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_disable_psi_idle) | Exclude Performance Schema idle instrumentation | **OFF** |  |  |
| [**DISABLE\_PSI\_MEMORY**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_disable_psi_memory) | Exclude Performance Schema memory instrumentation | **OFF** |  |  |
| [**DISABLE\_PSI\_METADATA**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_disable_psi_metadata) | Exclude Performance Schema metadata instrumentation | **OFF** |  |  |
| [**DISABLE\_PSI\_MUTEX**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_disable_psi_mutex) | Exclude Performance Schema mutex instrumentation | **OFF** |  |  |
| [**DISABLE\_PSI\_PS**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_disable_psi_ps) | Exclude the performance schema prepared statements | **OFF** |  |  |
| [**DISABLE\_PSI\_RWLOCK**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_disable_psi_rwlock) | Exclude Performance Schema rwlock instrumentation | **OFF** |  |  |
| [**DISABLE\_PSI\_SOCKET**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_disable_psi_socket) | Exclude Performance Schema socket instrumentation | **OFF** |  |  |
| [**DISABLE\_PSI\_SP**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_disable_psi_sp) | Exclude Performance Schema stored program instrumentation | **OFF** |  |  |
| [**DISABLE\_PSI\_STAGE**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_disable_psi_stage) | Exclude Performance Schema stage instrumentation | **OFF** |  |  |
| [**DISABLE\_PSI\_STATEMENT**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_disable_psi_statement) | Exclude Performance Schema statement instrumentation | **OFF** |  |  |
| [**DISABLE\_PSI\_STATEMENT\_DIGEST**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_disable_psi_statement_digest) | Exclude Performance Schema statements\_digest instrumentation | **OFF** |  |  |
| [**DISABLE\_PSI\_TABLE**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_disable_psi_table) | Exclude Performance Schema table instrumentation | **OFF** |  |  |
| [**DISABLE\_PSI\_THREAD**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_disable_psi_thread) | Exclude the performance schema thread instrumentation | **OFF** |  |  |
| [**DISABLE\_PSI\_TRANSACTION**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_disable_psi_transaction) | Exclude the performance schema transaction instrumentation | **OFF** |  |  |
| [**DISABLE\_SHARED**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_disable_shared) | Do not build shared libraries, compile position-dependent code | **OFF** |  | 8.0.18 |
| [**DOWNLOAD\_BOOST**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_download_boost) | Whether to download the Boost library | **OFF** |  |  |
| [**DOWNLOAD\_BOOST\_TIMEOUT**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_download_boost_timeout) | Timeout in seconds for downloading the Boost library | **600** |  |  |
| [**ENABLED\_LOCAL\_INFILE**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_enabled_local_infile) | Whether to enable LOCAL for LOAD DATA | **OFF** |  |  |
| [**ENABLED\_PROFILING**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_enabled_profiling) | Whether to enable query profiling code | **ON** |  |  |
| [**ENABLE\_DOWNLOADS**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_enable_downloads) | Whether to download optional files | **OFF** |  | 8.0.25 |
| [**ENABLE\_EXPERIMENTAL\_SYSVARS**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_enable_experimental_sysvars) | Whether to enabled experimental InnoDB system variables | **OFF** |  |  |
| [**ENABLE\_GCOV**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_enable_gcov) | Whether to include gcov support |  |  |  |
| [**ENABLE\_GPROF**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_enable_gprof) | Enable gprof (optimized Linux builds only) | **OFF** |  |  |
| [**FORCE\_INSOURCE\_BUILD**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_force_insource_build) | Whether to force an in-source build | **OFF** | 8.0.14 |  |
| [**FORCE\_UNSUPPORTED\_COMPILER**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_force_unsupported_compiler) | Whether to permit unsupported compiler | **OFF** |  |  |
| [**FPROFILE\_GENERATE**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_fprofile_generate) | Whether to generate profile guided optimization data | **OFF** | 8.0.19 |  |
| [**FPROFILE\_USE**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_fprofile_use) | Whether to use profile guided optimization data | **OFF** | 8.0.19 |  |
| [**HAVE\_PSI\_MEMORY\_INTERFACE**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_have_psi_memory_interface) | Enable performance schema memory tracing module for memory allocation functions used in dynamic storage of over-aligned types | **OFF** | 8.0.25 |  |
| [**IGNORE\_AIO\_CHECK**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_ignore_aio_check) | With -DBUILD\_CONFIG=mysql\_release, ignore libaio check | **OFF** |  |  |
| [**INSTALL\_BINDIR**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_install_bindir) | User executables directory | **PREFIX/bin** |  |  |
| [**INSTALL\_DOCDIR**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_install_docdir) | Documentation directory | **PREFIX/docs** |  |  |
| [**INSTALL\_DOCREADMEDIR**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_install_docreadmedir) | README file directory | **PREFIX** |  |  |
| [**INSTALL\_INCLUDEDIR**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_install_includedir) | Header file directory | **PREFIX/include** |  |  |
| [**INSTALL\_INFODIR**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_install_infodir) | Info file directory | **PREFIX/docs** |  |  |
| [**INSTALL\_LAYOUT**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_install_layout) | Select predefined installation layout | **STANDALONE** |  |  |
| [**INSTALL\_LIBDIR**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_install_libdir) | Library file directory | **PREFIX/lib** |  |  |
| [**INSTALL\_MANDIR**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_install_mandir) | Manual page directory | **PREFIX/man** |  |  |
| [**INSTALL\_MYSQLKEYRINGDIR**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_install_mysqlkeyringdir) | Directory for keyring\_file plugin data file | **platform specific** |  |  |
| [**INSTALL\_MYSQLSHAREDIR**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_install_mysqlsharedir) | Shared data directory | **PREFIX/share** |  |  |
| [**INSTALL\_MYSQLTESTDIR**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_install_mysqltestdir) | mysql-test directory | **PREFIX/mysql-test** |  |  |
| [**INSTALL\_PKGCONFIGDIR**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_install_pkgconfigdir) | Directory for mysqlclient.pc pkg-config file | **INSTALL\_LIBDIR/pkgconfig** |  |  |
| [**INSTALL\_PLUGINDIR**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_install_plugindir) | Plugin directory | **PREFIX/lib/plugin** |  |  |
| [**INSTALL\_SBINDIR**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_install_sbindir) | Server executable directory | **PREFIX/bin** |  |  |
| [**INSTALL\_SECURE\_FILE\_PRIVDIR**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_install_secure_file_privdir) | secure\_file\_priv default value | **platform specific** |  |  |
| [**INSTALL\_SHAREDIR**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_install_sharedir) | aclocal/mysql.m4 installation directory | **PREFIX/share** |  |  |
| [**INSTALL\_STATIC\_LIBRARIES**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_install_static_libraries) | Whether to install static libraries | **ON** |  |  |
| [**INSTALL\_SUPPORTFILESDIR**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_install_supportfilesdir) | Extra support files directory | **PREFIX/support-files** |  |  |
| [**LINK\_RANDOMIZE**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_link_randomize) | Whether to randomize order of symbols in mysqld binary | **OFF** |  |  |
| [**LINK\_RANDOMIZE\_SEED**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_link_randomize_seed) | Seed value for LINK\_RANDOMIZE option | **mysql** |  |  |
| [**MAX\_INDEXES**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_max_indexes) | Maximum indexes per table | **64** |  |  |
| [**MUTEX\_TYPE**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_mutex_type) | InnoDB mutex type | **event** |  |  |
| [**MYSQLX\_TCP\_PORT**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_mysqlx_tcp_port) | TCP/IP port number used by X Plugin | **33060** |  |  |
| [**MYSQLX\_UNIX\_ADDR**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_mysqlx_unix_addr) | Unix socket file used by X Plugin | **/tmp/mysqlx.sock** |  |  |
| [**MYSQL\_DATADIR**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_mysql_datadir) | Data directory |  |  |  |
| [**MYSQL\_MAINTAINER\_MODE**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_mysql_maintainer_mode) | Whether to enable MySQL maintainer-specific development environment | **OFF** |  |  |
| [**MYSQL\_PROJECT\_NAME**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_mysql_project_name) | Windows/OS X project name | **MySQL** |  |  |
| [**MYSQL\_TCP\_PORT**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_mysql_tcp_port) | TCP/IP port number | **3306** |  |  |
| [**MYSQL\_UNIX\_ADDR**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_mysql_unix_addr) | Unix socket file | **/tmp/mysql.sock** |  |  |
| [**NDB\_UTILS\_LINK\_DYNAMIC**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_ndb_utils_link_dynamic) | Cause NDB tools to be dynamically linked to ndbclient |  | 8.0.22-ndb-8.0.22 |  |
| [**ODBC\_INCLUDES**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_odbc_includes) | ODBC includes directory |  |  |  |
| [**ODBC\_LIB\_DIR**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_odbc_lib_dir) | ODBC library directory |  |  |  |
| [**OPTIMIZER\_TRACE**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_optimizer_trace) | Whether to support optimizer tracing |  |  |  |
| [**REPRODUCIBLE\_BUILD**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_reproducible_build) | Take extra care to create a build result independent of build location and time |  |  |  |
| [**SYSCONFDIR**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_sysconfdir) | Option file directory |  |  |  |
| [**SYSTEMD\_PID\_DIR**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_systemd_pid_dir) | Directory for PID file under systemd | **/var/run/mysqld** |  |  |
| [**SYSTEMD\_SERVICE\_NAME**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_systemd_service_name) | Name of MySQL service under systemd | **mysqld** |  |  |
| [**TMPDIR**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_tmpdir) | tmpdir default value |  |  |  |
| [**USE\_LD\_GOLD**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_use_ld_gold) | Whether to use GNU gold linker | **ON** |  |  |
| [**USE\_LD\_LLD**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_use_ld_lld) | Whether to use llvm lld linker | **ON** | 8.0.16 |  |
| [**WIN\_DEBUG\_NO\_INLINE**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_win_debug_no_inline) | Whether to disable function inlining | **OFF** |  |  |
| [**WITHOUT\_xxx\_STORAGE\_ENGINE**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_storage_engine_options) | Exclude storage engine xxx from build |  |  |  |
| [**WITH\_ANT**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_ant) | Path to Ant for building GCS Java wrapper |  |  |  |
| [**WITH\_ASAN**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_asan) | Enable AddressSanitizer | **OFF** |  |  |
| [**WITH\_ASAN\_SCOPE**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_asan_scope) | Enable AddressSanitizer -fsanitize-address-use-after-scope Clang flag | **OFF** |  |  |
| [**WITH\_AUTHENTICATION\_CLIENT\_PLUGINS**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_authentication_client_plugins) | Enabled automatically if any corresponding server authentication plugins are built |  | 8.0.25 |  |
| [**WITH\_AUTHENTICATION\_LDAP**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_authentication_ldap) | Whether to report error if LDAP authentication plugins cannot be built | **OFF** |  |  |
| [**WITH\_AUTHENTICATION\_PAM**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_authentication_pam) | Build PAM authentication plugin | **OFF** |  |  |
| [**WITH\_AWS\_SDK**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_aws_sdk) | Location of Amazon Web Services software development kit |  |  |  |
| [**WITH\_BOOST**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_boost) | The location of the Boost library sources |  |  |  |
| [**WITH\_CLIENT\_PROTOCOL\_TRACING**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_client_protocol_tracing) | Build client-side protocol tracing framework | **ON** |  |  |
| [**WITH\_CURL**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_curl) | Location of curl library |  |  |  |
| [**WITH\_DEBUG**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_debug) | Whether to include debugging support | **OFF** |  |  |
| [**WITH\_DEFAULT\_COMPILER\_OPTIONS**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_default_compiler_options) | Whether to use default compiler options | **ON** |  |  |
| [**WITH\_DEFAULT\_FEATURE\_SET**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_default_feature_set) | Whether to use default feature set | **ON** |  | 8.0.22 |
| [**WITH\_EDITLINE**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_editline) | Which libedit/editline library to use | **bundled** |  |  |
| [**WITH\_GMOCK**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_gmock) | Path to googlemock distribution |  |  | 8.0.25 |
| [**WITH\_ICU**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_icu) | Type of ICU support | **bundled** |  |  |
| [**WITH\_INNODB\_EXTRA\_DEBUG**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_innodb_extra_debug) | Whether to include extra debugging support for InnoDB. | **OFF** |  |  |
| [**WITH\_INNODB\_MEMCACHED**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_innodb_memcached) | Whether to generate memcached shared libraries. | **OFF** |  |  |
| [**WITH\_JEMALLOC**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_jemalloc) | Whether to link with -ljemalloc | **OFF** | 8.0.16 |  |
| [**WITH\_KEYRING\_TEST**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_keyring_test) | Build the keyring test program | **OFF** |  |  |
| [**WITH\_LIBEVENT**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_libevent) | Which libevent library to use | **bundled** |  |  |
| [**WITH\_LIBWRAP**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_libwrap) | Whether to include libwrap (TCP wrappers) support | **OFF** |  |  |
| [**WITH\_LOCK\_ORDER**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_lock_order) | Whether to enable LOCK\_ORDER tooling | **OFF** | 8.0.17 |  |
| [**WITH\_LSAN**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_lsan) | Whether to run LeakSanitizer, without AddressSanitizer | **OFF** | 8.0.16 |  |
| [**WITH\_LTO**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_lto) | Enable link-time optimizer | **OFF** | 8.0.13 |  |
| [**WITH\_LZ4**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_lz4) | Type of LZ4 library support | **bundled** |  |  |
| [**WITH\_LZMA**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_lzma) | Type of LZMA library support | **bundled** |  | 8.0.16 |
| [**WITH\_MECAB**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_mecab) | Compiles MeCab |  |  |  |
| [**WITH\_MSAN**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_msan) | Enable MemorySanitizer | **OFF** |  |  |
| [**WITH\_MSCRT\_DEBUG**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_mscrt_debug) | Enable Visual Studio CRT memory leak tracing | **OFF** |  |  |
| [**WITH\_MYSQLX**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_mysqlx) | Whether to disable X Protocol | **ON** |  |  |
| [**WITH\_NUMA**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_numa) | Set NUMA memory allocation policy |  |  |  |
| [**WITH\_PROTOBUF**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_protobuf) | Which Protocol Buffers package to use | **bundled** |  |  |
| [**WITH\_RAPID**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_rapid) | Whether to build rapid development cycle plugins | **ON** |  |  |
| [**WITH\_RAPIDJSON**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_rapidjson) | Type of RapidJSON support | **bundled** | 8.0.13 |  |
| [**WITH\_RE2**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_re2) | Type of RE2 library support | **bundled** |  | 8.0.18 |
| [**WITH\_ROUTER**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_router) | Whether to build MySQL Router | **ON** | 8.0.16 |  |
| [**WITH\_SSL**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_ssl) | Type of SSL support | **system** |  |  |
| [**WITH\_SYSTEMD**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_systemd) | Enable installation of systemd support files | **OFF** |  |  |
| [**WITH\_SYSTEMD\_DEBUG**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_systemd_debug) | Enable additional systemd debug information | **OFF** | 8.0.22 |  |
| [**WITH\_SYSTEM\_LIBS**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_system_libs) | Set system value of library options not set explicitly | **OFF** |  |  |
| [**WITH\_TCMALLOC**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_tcmalloc) | Whether to link with -ltcmalloc | **OFF** | 8.0.22 |  |
| [**WITH\_TEST\_TRACE\_PLUGIN**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_test_trace_plugin) | Build test protocol trace plugin | **OFF** |  |  |
| [**WITH\_TSAN**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_tsan) | Enable ThreadSanitizer | **OFF** |  |  |
| [**WITH\_UBSAN**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_ubsan) | Enable Undefined Behavior Sanitizer | **OFF** |  |  |
| [**WITH\_UNIT\_TESTS**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_unit_tests) | Compile MySQL with unit tests | **ON** |  |  |
| [**WITH\_UNIXODBC**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_unixodbc) | Enable unixODBC support | **OFF** |  |  |
| [**WITH\_VALGRIND**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_valgrind) | Whether to compile in Valgrind header files | **OFF** |  |  |
| [**WITH\_ZLIB**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_zlib) | Type of zlib support | **bundled** |  |  |
| [**WITH\_ZSTD**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_with_zstd) | Type of zstd support | **bundled** | 8.0.18 |  |
| [**WITH\_xxx\_STORAGE\_ENGINE**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\installing.html#option_cmake_storage_engine_options) | Compile storage engine xxx statically into server |  |  |  |

**一般选项**

* -DBUILD\_CONFIG=mysql\_release

该选项配置了一个源码发行版，其构建选项与Oracle为官方MySQL发行版生产二进制发行版所使用的相同。

* -DBUNDLE\_RUNTIME\_LIBRARIES=bool

是否将运行时库与Windows的服务器MSI和Zip包捆绑。

* -DCMAKE\_BUILD\_TYPE=type

要产生的构建类型。

* + RelWithDebInfo。启用优化并生成调试信息。这是默认的MySQL构建类型。
  + 发布。启用优化，但省略调试信息以减少构建规模。这种构建类型是在MySQL 8.0.13中添加的。
  + 调试。禁用优化并生成调试信息。如果启用了WITH\_DEBUG选项，也会使用这种构建类型。也就是说，-DWITH\_DEBUG=1与-DCMAKE\_BUILD\_TYPE=Debug的效果相同。
* -DCPACK\_MONOLITHIC\_INSTALL=bool

这个选项影响make package操作是产生多个安装包文件还是一个文件。如果禁用，该操作会产生多个安装包文件，如果你想只安装完整MySQL安装的一个子集，这可能很有用。如果启用，它将产生一个用于安装所有内容的单一文件。

* -DFORCE\_INSOURCE\_BUILD=bool

定义是否要强制进行源内构建。推荐使用源外构建，因为它允许从同一源头进行多次构建，并且可以通过删除构建目录快速进行清理。要强制进行源内构建，请用 -DFORCE\_INSOURCE\_BUILD=ON 来调用 CMake。

**安装布局选项**

CMAKE\_INSTALL\_PREFIX选项表示基本安装目录。其他名称为INSTALL\_xxx的选项表示组件的位置，是相对于前缀解释的，其值是相对路径名。它们的值不应该包括前缀。

* -DCMAKE\_INSTALL\_PREFIX=dir\_name

安装基础目录。

这个值可以在服务器启动时通过-basedir选项来设置。

* -DINSTALL\_BINDIR=dir\_name

安装用户程序的地方。

-DINSTALL\_DOCDIR=dir\_name

安装文档的地方。

* -DINSTALL\_DOCREADMEDIR=dir\_name

安装README文件的地方。

* -DINSTALL\_INCLUDEDIR=dir\_name

安装头文件的地方。

* -DINSTALL\_INFODIR=dir\_name

安装信息文件的地方。

* -DINSTALL\_LAYOUT=name

选择一个预定义的安装布局。

* + STANDALONE：与用于.tar.gz和.zip包的布局相同。这是默认的。
  + RPM。类似于RPM包的布局。
  + SVR4: Solaris软件包布局。
  + DEB: DEB包的布局（实验性的）。

你可以选择一个预定义的布局，但通过指定其他选项来修改单个组件的安装位置。比如说。

cmake . -DINSTALL\_LAYOUT=SVR4 -DMYSQL\_DATADIR=/var/mysql/data

INSTALL\_LAYOUT值决定了secure\_file\_priv、keyring\_encrypted\_file\_data和keyring\_file\_data系统变量的默认值。参见第5.1.8节 "服务器系统变量 "和第6.4.4.18节 "钥匙圈系统变量 "中关于这些变量的描述。

* -DINSTALL\_LIBDIR=dir\_name

安装库文件的位置。

* -DINSTALL\_MANDIR=dir\_name

安装手册页的地方。

* -DINSTALL\_MYSQLKEYRINGDIR=dir\_path

作为keyring\_file插件数据文件位置的默认目录。默认值与平台有关，取决于INSTALL\_LAYOUT CMake选项的值；参见5.1.8节 "服务器系统变量 "中关于keyring\_file\_data系统变量的描述。

* -DINSTALL\_MYSQLSHAREDIR=dir\_name

安装共享数据文件的位置。

* -DINSTALL\_MYSQLTESTDIR=dir\_name

在哪里安装mysql-test目录。要抑制这个目录的安装，明确地将该选项设置为空值（-DINSTALL\_MYSQLTESTDIR=）。

* -DINSTALL\_PKGCONFIGDIR=dir\_name

安装mysqlclient.pc文件的目录，供pkg-config使用。默认值是INSTALL\_LIBDIR/pkgconfig，除非INSTALL\_LIBDIR以/mysql结尾，在这种情况下会首先删除。

* -DINSTALL\_PLUGINDIR=dir\_name

插件目录的位置。

这个值可以在服务器启动时用--plugin\_dir选项设置。

* -DINSTALL\_PRIV\_LIBDIR=dir\_name

动态库目录的位置。

默认位置。RPM = /usr/lib64/mysql/private/，DEB = /usr/lib/mysql/private/，和TAR = lib/private/。

这个选项是在MySQL 8.0.18中添加的。

对于Protobuf：因为这是一个私有的位置，加载器（例如Linux上的ld-linux.so）在没有帮助的情况下可能找不到libprotobuf.so文件。为了指导加载器，在mysqld和mysqlxtest中加入了RPATH，其值为$ORIGIN/.../$INSTALL\_PRIV\_LIBDIR。这在大多数情况下是有效的，但当使用资源组功能时，mysqld被设置为id，然后加载器忽略了包含$ORIGIN的RPATH。为了克服这个问题，在DEB和RPM变体的mysqld中设置了一个明确的目录全路径，因为目标目的地是已知的。对于tarball安装，需要用patchelf这样的工具对mysqld进行修补。

* -DINSTALL\_SBINDIR=dir\_name

在哪里安装mysqld服务器。

* -DINSTALL\_SECURE\_FILE\_PRIVDIR=dir\_name
* secure\_file\_priv 系统变量的默认值。默认值与平台有关， 并取决于 INSTALL\_LAYOUT CMake 选项的值； 请参阅第 5.1.8 节 "服务器系统变量" 中关于 secure\_file\_priv 系统变量的描述。
* -DINSTALL\_SHAREDIR=dir\_name

安装aclocal/mysql.m4的位置。

* -DINSTALL\_STATIC\_LIBRARIES=bool

是否安装静态库。默认为ON。如果设置为OFF，则不安装这些库：libmysqlclient.a, libmysqlservices.a。

* -DINSTALL\_SUPPORTFILESDIR=dir\_name

将额外的支持文件安装在哪里。

* -DLINK\_RANDOMIZE=bool

是否随机化mysqld二进制中的符号顺序。默认为关闭。这个选项应该只为调试目的而启用。

* -DLINK\_RANDOMIZE\_SEED=val

LINK\_RANDOMIZE选项的种子值。该值是一个字符串。默认是mysql，这是一个任意的选择。

* -DMYSQL\_DATADIR=dir\_name

MySQL数据目录的位置。

这个值可以在服务器启动时用-datadir选项设置。

* -DODBC\_INCLUDES=dir\_name

ODBC包括目录的位置，可在配置Connector/ODBC时使用。

* -DODBC\_LIB\_DIR=dir\_name

ODBC库目录的位置，可在配置Connector/ODBC时使用。

* -DSYSCONFDIR=dir\_name

默认的my.cnf选项文件目录。

这个位置不能在服务器启动时设置，但你可以使用--defaults-file=file\_name选项以给定的选项文件启动服务器，其中file\_name是该文件的全路径名称。

* -DSYSTEMD\_PID\_DIR=dir\_name

当MySQL由systemd管理时，创建PID文件的目录名称。默认为/var/run/mysqld；可能会根据INSTALL\_LAYOUT值隐式改变。

除非启用 WITH\_SYSTEMD，否则该选项被忽略。

* -DSYSTEMD\_SERVICE\_NAME=name

当MySQL由systemd管理时，要使用的MySQL服务的名称。默认为 mysqld；根据 INSTALL\_LAYOUT 值，该名称可能被隐含地改变。

除非启用WITH\_SYSTEMD，否则该选项被忽略。

* -DTMPDIR=dir\_name

默认使用的tmpdir系统变量的位置。如果没有指定，该值默认为<stdio.h>中的P\_tmpdir。

**存储引擎选项**

存储引擎是作为插件构建的。你可以将插件构建为静态模块（编译到服务器中）或动态模块（构建为动态库，在使用前必须使用INSTALL PLUGIN语句或--plugin-load选项安装到服务器中）。有些插件可能不支持静态或动态构建。

InnoDB、MyISAM、MERGE、MEMORY和CSV引擎是强制性的（总是被编译到服务器中），不需要明确安装。

要将存储引擎静态地编译到服务器中，请使用-DWITH\_engine\_STORAGE\_ENGINE=1。一些允许的引擎值是：ARCHIVE、BLACKHOLE、EXAMPLE、FEDERATED、NDB或NDBCLUSTER（支持NDB）。例如。

-DWITH\_ARCHIVE\_STORAGE\_ENGINE=1

-DWITH\_BLACKHOLE\_STORAGE\_ENGINE=1

注意事项

在没有Performance Schema支持的情况下，是不可能进行编译的。如果需要在没有特定类型的工具的情况下进行编译，可以通过以下CMake选项来实现。

DISABLE\_PSI\_COND

DISABLE\_PSI\_DATA\_LOCK

DISABLE\_PSI\_ERROR

DISABLE\_PSI\_FILE

DISABLE\_PSI\_IDLE

DISABLE\_PSI\_MEMORY

DISABLE\_PSI\_METADATA

DISABLE\_PSI\_MUTEX

DISABLE\_PSI\_PS

DISABLE\_PSI\_RWLOCK

DISABLE\_PSI\_SOCKET

DISABLE\_PSI\_SP

DISABLE\_PSI\_STAGE

DISABLE\_PSI\_STATEMENT

DISABLE\_PSI\_STATEMENT\_DIGEST

DISABLE\_PSI\_TABLE

DISABLE\_PSI\_THREAD

DISABLE\_PSI\_TRANSACTION

例如，如果要在没有互斥工具的情况下编译，使用-DDISABLE\_PSI\_MUTEX=1选项配置MySQL。

要从构建中排除一个存储引擎，使用-DWITH\_engine\_STORAGE\_ENGINE=0。 示例。

-DWITH\_ARCHIVE\_STORAGE\_ENGINE=0

-DWITH\_EXAMPLE\_STORAGE\_ENGINE=0

-DWITH\_FEDERATED\_STORAGE\_ENGINE=0

也可以使用 -DWITHOUT\_engine\_STORAGE\_ENGINE=1 将存储引擎排除在构建之外（但首选是 -DWITH\_engine\_STORAGE\_ENGINE=0）。例子。

-DWITHOUT\_ARCHIVE\_STORAGE\_ENGINE=1

-DWITHOUT\_EXAMPLE\_STORAGE\_ENGINE=1

-DWITHOUT\_FEDERATED\_STORAGE\_ENGINE=1

如果既没有为某个存储引擎指定-DWITH\_engine\_STORAGE\_ENGINE，也没有指定-DWITHOUT\_engine\_STORAGE\_ENGINE，则该引擎将作为共享模块构建，如果不能作为共享模块构建，则排除。

**特性选项**

* -DADD\_GDB\_INDEX=bool

该选项决定是否启用在二进制文件中生成.gdb\_index部分，这可以使它们在调试器中的加载速度更快。该选项在默认情况下是禁用的。如果使用ld链接器，则该选项被禁用。 如果使用ld或GNU gold以外的链接器，则该选项没有影响。

这个选项是在MySQL 8.0.18中添加的。

* -DCOMPILATION\_COMMENT=string

关于编译环境的描述性注释。从MySQL 8.0.14开始，mysqld使用COMPILATION\_COMMENT\_SERVER。其他程序继续使用COMPILATION\_COMMENT。

* -DCOMPRESS\_DEBUG\_SECTIONS=bool

是否压缩二进制可执行文件的调试部分（仅Linux）。压缩可执行文件的调试部分可以节省空间，但在构建过程中要付出额外的CPU时间。

默认是关闭。如果没有明确设置这个选项，但设置了COMPRESS\_DEBUG\_SECTIONS环境变量，则该选项从该变量中取值。

这个选项是在MySQL 8.0.22中添加的。

* -DCOMPILATION\_COMMENT\_SERVER=string

关于编译环境的描述性注释，供mysqld使用（例如，设置version\_comment系统变量）。这个选项是在MySQL 8.0.14中添加的。在8.0.14之前，服务器使用COMPILATION\_COMMENT。

* -DDEFAULT\_CHARSET=charset\_name

服务器的字符集。默认情况下，MySQL使用utf8mb4字符集。

charset\_name可以是二进制、armcii8、ascii、big5、cp1250、cp1251、cp1256、cp1257、cp850、cp852、cp866、cp932、dec8、eucjpms、euckr、gb2312、gbk、geostd8, greek, hebrew, hp8, keybcs2, koi8r, koi8u, latin1, latin2, latin5, latin7, macce, macroman, sjis, swe7, tis620, ucs2, ujis, utf8, utf8mb4, utf16, utf16le, utf32. 允许的字符集在cmake/character\_sets.cmake文件中作为CHARSETS\_AVAILABLE的值列出。

这个值可以在服务器启动时用 --character\_set\_server 选项来设置。

* -DDEFAULT\_COLLATION=collation\_name

服务器的拼写方式。默认情况下，MySQL使用utf8mb4\_0900\_ai\_ci。使用SHOW COLLATION语句来确定每个字符集有哪些排序。

这个值可以在服务器启动时用--collation\_server选项设置。

* -DDISABLE\_PSI\_COND=bool

是否排除Performance Schema条件下的仪表。默认为OFF（包括）。

* -DDISABLE\_PSI\_FILE=bool

是否排除Performance Schema文件检测。默认值是关闭（包括）。

* -DDISABLE\_PSI\_IDLE=bool

是否排除Performance Schema的空闲仪表。缺省是关闭（包括）。

* -DDISABLE\_PSI\_MEMORY=bool

是否排除Performance Schema的内存检测。缺省是关闭（包括）。

* -DDISABLE\_PSI\_METADATA=bool

是否排除对Performance Schema元数据的检测。缺省是关闭（包括）。

* -DDISABLE\_PSI\_MUTEX=bool

是否排除Performance Schema mutex工具的使用。缺省是关闭（包括）。

* -DDISABLE\_PSI\_RWLOCK=bool

是否排除Performance Schema rwlock工具的使用。缺省是关闭（包括）。

* -DDISABLE\_PSI\_SOCKET=bool

是否排除Performance Schema套接字工具的使用。缺省是关闭（包括）。

* -DDISABLE\_PSI\_SP=bool

是否排除对Performance Schema存储程序的检测。缺省是关闭（包括）。

* -DDISABLE\_PSI\_STAGE=bool

是否排除Performance Schema的阶段工具。缺省是关闭（包括）。

* -DDISABLE\_PSI\_STATEMENT=bool

是否排除对Performance Schema语句的检测。缺省是关闭（包括）。

* -DDISABLE\_PSI\_STATEMENT\_DIGEST=bool

是否排除Performance Schema语句摘要工具的使用。缺省是关闭（包括）。

* -DDISABLE\_PSI\_TABLE=bool

是否排除对Performance Schema表的检测。缺省是关闭（包括）。

* -DDISABLE\_SHARED=bool

是否禁止构建共享库和编译依赖位置的代码。默认为OFF（编译与位置无关的代码）。

这个选项没有使用，在MySQL 8.0.18中被删除。

* -DDISABLE\_PSI\_PS=bool

排除性能模式准备好的语句实例仪表。默认是关闭（包括）。

* -DDISABLE\_PSI\_THREAD=bool

排除性能模式的线程仪表。默认是关闭（包括）。

只有在构建时没有任何工具化时才禁用线程，因为其他工具化对线程有依赖性。

* -DDISABLE\_PSI\_TRANSACTION=bool

排除性能模式的事务仪表。默认为OFF（包括）。

* -DDISABLE\_PSI\_DATA\_LOCK=bool

排除性能模式的数据锁检测。默认值是关闭（包括）。

* -DDISABLE\_PSI\_ERROR=bool

排除性能模式的服务器错误检测。默认值是关闭（包括）。

* -DDOWNLOAD\_BOOST=bool

是否下载Boost库。默认为关闭。

参见WITH\_BOOST选项，了解关于使用Boost的更多讨论。

* -DDOWNLOAD\_BOOST\_TIMEOUT=seconds

下载Boost库的超时时间，单位是秒。默认是600秒。

关于使用Boost的其他讨论，请参见WITH\_BOOST选项。

* -DENABLE\_DOWNLOADS=bool

是否要下载可选文件。例如，启用该选项后，CMake会下载测试套件用于运行单元测试的Google Test发行版，或构建GCS Java包装器所需的Ant和JUnit。

从MySQL 8.0.25开始，MySQL源码发行版捆绑了Google Test源代码，用于运行基于Google Test的单元测试。因此，从该版本开始，WITH\_GMOCK和ENABLE\_DOWNLOADS CMake选项被删除，如果被指定，则被忽略。

* -DENABLE\_EXPERIMENTAL\_SYSVARS=bool

是否启用实验性InnoDB系统变量。实验性系统变量是为从事MySQL开发的人准备的，只应在开发或测试环境中使用，并可能在未来的MySQL版本中不经通知就被删除。关于实验性系统变量的信息，请参考MySQL源代码树中的/storage/innobase/handler/ha\_innodb.cc。实验性系统变量可以通过搜索 "PLUGIN\_VAR\_EXPERIMENTAL "来识别。

* -DENABLE\_GCOV=bool

是否包括gcov支持（仅Linux）。

-DENABLE\_GPROF=bool

是否启用 gprof (仅优化的 Linux 构建)。

* -DENABLED\_LOCAL\_INFILE=bool

该选项控制了MySQL客户端库的默认LOCAL编译能力。因此，没有明确安排的客户端会根据在MySQL构建时指定的ENABLED\_LOCAL\_INFILE设置禁用或启用LOCAL能力。

默认情况下，MySQL二进制发行版中的客户端库是在禁用ENABLED\_LOCAL\_INFILE的情况下编译的。如果你从源码编译MySQL，根据没有明确安排的客户端是否应该禁用或启用LOCAL能力，分别以ENABLED\_LOCAL\_INFILE禁用或启用的方式配置它。

ENABLED\_LOCAL\_INFILE控制客户端LOCAL能力的默认值。对于服务器，local\_infile系统变量控制服务器端的LOCAL能力。要明确地使服务器拒绝或允许LOAD DATA LOCAL语句(不管客户程序和库在构建时或运行时是如何配置的)，在禁用或启用local\_infile的情况下启动mysqld。 local\_infile也可以在运行时设置。参见第6.1.6节，"LOAD DATA LOCAL的安全考虑"。

* -DENABLED\_PROFILING=bool

是否启用查询分析代码（用于SHOW PROFILE和SHOW PROFILES语句）。

* -DFORCE\_UNSUPPORTED\_COMPILER=bool

默认情况下，CMake检查支持的编译器的最小版本。Visual Studio 2015（Windows）；GCC 4.8或Clang 3.4（Linux）；Developer Studio 12.5（Solaris服务器）；Developer Studio 12.4或GCC 4.8（Solaris客户端库）；Clang 3.6（macOS），Clang 3.4（FreeBSD）。要禁用这个检查，请使用 -DFORCE\_UNSUPPORTED\_COMPILER=ON。

* -DFPROFILE\_GENERATE=bool

是否生成配置文件引导的优化（PGO）数据。这个选项适用于用GCC进行PGO实验。有关使用FPROFILE\_GENERATE和FPROFILE\_USE的信息，请参见MySQL源码分发中的cmake/fprofile.cmake文件。这些选项已在GCC 8和9中测试过。

这个选项是在MySQL 8.0.19中添加的。

* -DFPROFILE\_USE=bool

是否使用配置文件引导的优化（PGO）数据。这个选项可用于在GCC中试验PGO。有关使用FPROFILE\_GENERATE和FPROFILE\_USE的信息，请参见MySQL源码分发中的cmake/fprofile.cmake文件。这些选项已经在GCC 8和9中进行了测试。

启用FPROFILE\_USE还可以启用WITH\_LTO。

这个选项是在MySQL 8.0.19中添加的。

* -DHAVE\_PSI\_MEMORY\_INTERFACE=bool

是否启用用于动态存储超对齐类型的内存分配函数（ut::aligned\_name库函数）的性能模式内存跟踪模块。

-DIGNORE\_AIO\_CHECK=bool

如果在Linux上给出了-DBUILD\_CONFIG=mysql\_release选项，libaio库必须被默认链接进来。如果你没有libaio或者不想安装它，你可以通过指定-DIGNORE\_AIO\_CHECK=1抑制对它的检查。

* -DMAX\_INDEXES=num

每个表的最大索引数。默认是64。最大为255。小于64的值被忽略，使用默认的64。

* -DMYSQL\_MAINTAINER\_MODE=bool

是否启用MySQL维护者专用开发环境。如果启用，该选项会使编译器警告变成错误。

* -DMUTEX\_TYPE=type

InnoDB使用的mutex类型。选项包括。

事件。使用事件互斥。这是默认值，也是最初的InnoDB mutex实现。

sys: 在UNIX系统上使用POSIX mutexes。如果有的话，在Windows上使用CRITICAL\_SECTION onjects。

futex。使用Linux的futexes而不是条件变量来安排等待的线程。

* -DMYSQLX\_TCP\_PORT=port\_num

X Plugin用于监听TCP/IP连接的端口号。默认是33060。

这个值可以在服务器启动时通过mysqlx\_port系统变量设置。

* -DMYSQLX\_UNIX\_ADDR=file\_name

服务器监听X插件套接字连接的Unix套接字文件路径。这必须是一个绝对路径名。默认是/tmp/mysqlx.sock。

这个值可以在服务器启动时通过mysqlx\_port系统变量设置。

* -DMYSQL\_PROJECT\_NAME=name

对于Windows或macOS，项目名称要纳入项目文件名。

* -DMYSQL\_TCP\_PORT=port\_num

服务器监听TCP/IP连接的端口号。默认为3306。

这个值可以在服务器启动时用--port选项设置。

* -DMYSQL\_UNIX\_ADDR=file\_name

服务器监听套接字连接的Unix套接字文件路径。这必须是一个绝对路径名。默认是/tmp/mysql.sock。

这个值可以在服务器启动时通过-socket选项设置。

* -DOPTIMIZER\_TRACE=bool

是否支持优化器跟踪。见MySQL内部。追踪优化器。

* -DREPROCIBLE\_BUILD=bool

对于在Linux系统上的构建，该选项控制是否采取额外的措施来创建一个独立于构建地点和时间的构建结果。

这个选项是在MySQL 8.0.11中添加的。从MySQL 8.0.12开始，对于RelWithDebInfo的构建，它默认为ON。

-DUSE\_LD\_GOLD=bool

如果GNU gold链接器可用且未明确禁用，CMake会使构建过程与它链接。要禁止使用该链接器，请指定 -DUSE\_LD\_GOLD=OFF 选项。

* -DUSE\_LD\_LLD=bool

如果Clang的llvm lld链接器可用，并且没有被明确禁用，CMake会使构建过程与它链接。要禁止使用该链接器，指定-DUSE\_LD\_LLD=OFF选项。

这个选项是在MySQL 8.0.16中添加的。

* -DWIN\_DEBUG\_NO\_INLINE=bool

是否在Windows上禁用函数内联。默认是关闭（启用内联）。

* -DWITH\_ANT=path\_name

设置Ant的路径，在构建GCS Java包装器时需要。与现有的 WITH\_BOOST CMake 选项的工作方式类似。将 WITH\_ANT 设为保存 Ant tarball 或已经解压的归档文件的目录路径。当 WITH\_ANT 没有被设置，或者被设置为特殊值系统时，构建会假定 $PATH 中存在一个二进制 ant。

* -DWITH\_ASAN=bool

是否为支持 AddressSanitizer 的编译器启用该功能。默认为关闭。

* -DWITH\_ASAN\_SCOPE=bool

是否启用AddressSanitizer -fsanite-address-use-after-scope Clang标志，用于检测使用后范围。默认为关闭。要使用这个选项，必须同时启用-DWITH\_ASAN。

* -DWITH\_AUTHENTICATION\_CLIENT\_PLUGINS=bool

如果构建了任何相应的服务器认证插件，该选项将自动启用。因此，它的值取决于其他CMake选项，不应明确设置。

这个选项是在MySQL 8.0.25中添加的。

* -DWITH\_AUTHENTICATION\_LDAP=bool

如果LDAP认证插件不能被构建，是否报告错误。

如果这个选项被禁用（默认），如果找到所需的头文件和库，LDAP插件就会被构建。如果没有，CMake会显示一个相关说明。

如果这个选项被激活，如果找不到所需的头文件和库，CMake会产生一个错误，阻止服务器被构建。

* -DWITH\_AUTHENTICATION\_PAM=bool

是否构建PAM认证插件，用于包含该插件的源码树。(参见第6.4.1.5节，"PAM可插拔认证"。)如果指定了这个选项，而该插件不能被编译，则构建失败。

* -DWITH\_AWS\_SDK=path\_name

亚马逊网络服务软件开发工具包的位置。

* -DWITH\_BOOST=path\_name

构建MySQL时需要Boost库。这些CMake选项可以控制库的来源位置，以及是否自动下载。

* -DWITH\_BOOST=path\_name指定Boost库的目录位置。也可以通过设置BOOST\_ROOT或WITH\_BOOST环境变量来指定Boost的位置。
* -DWITH\_BOOST=system也是允许的，表示正确的Boost版本已经安装在编译主机的标准位置。在这种情况下，会使用已安装的Boost版本，而不是MySQL源码分发中包含的任何版本。
* -DDOWNLOAD\_BOOST=bool指定如果Boost源码不在指定位置，是否要下载它。默认是关闭。
* -DDOWNLOAD\_BOOST\_TIMEOUT=seconds下载Boost库的超时时间，单位为秒。默认值是600秒。

例如，如果你通常在构建MySQL时将对象输出放在MySQL源码树的bld子目录下，你可以像这样用Boost构建。

mkdir bld

cd bld

cmake .. -DDOWNLOAD\_BOOST=ON -DWITH\_BOOST=$HOME/my\_boost

这将导致Boost被下载到你主目录下的my\_boost目录。如果所需的Boost版本已经在那里了，则不进行下载。如果所需的Boost版本改变了，就会下载较新的版本。

如果本地已经安装了Boost，并且你的编译器可以自己找到Boost头文件，那么可能就没有必要指定前面的CMake选项。然而，如果MySQL所需的Boost版本改变了，而本地安装的版本没有升级，你可能会遇到构建问题。使用CMake选项应该可以成功构建。

在上述允许Boost下载到指定位置的设置下，当所需的Boost版本改变时，你需要删除bld文件夹，重新创建，并再次执行cmake步骤。否则，新的Boost版本可能不会被下载，而且编译可能会失败。

* -DWITH\_CLIENT\_PROTOCOL\_TRACING=bool

是否在客户端库中构建客户端协议追踪框架。默认情况下，这个选项是启用的。

关于编写协议跟踪客户端插件的信息，请参见编写协议跟踪插件。

另请参见WITH\_TEST\_TRACE\_PLUGIN选项。

* -DWITH\_CURL=curl\_type

curl库的位置。curl\_type可以是系统（使用系统的curl库）或curl库的路径名。

* -DWITH\_DEBUG=bool

是否包括调试支持。

配置具有调试支持的MySQL使你能够在启动服务器时使用--debug="d,parser\_debug "选项。这将导致用于处理SQL语句的Bison解析器向服务器的标准错误输出转储一个解析器跟踪。通常情况下，这个输出被写到错误日志中。

InnoDB存储引擎的同步调试检查定义在UNIV\_DEBUG下，当使用WITH\_DEBUG选项编译调试支持时，可以使用。当调试支持被编译进来时，innodb\_sync\_debug配置选项可以用来启用或禁用InnoDB同步调试检查。

启用WITH\_DEBUG也可以启用调试同步。这个功能是用来测试和调试的。当编译进来时，Debug Sync在运行时默认是禁用的。要启用它，用--debug-sync-timeout=N选项启动mysqld，其中N是一个大于0的超时值（默认值是0，禁用Debug Sync），N成为单个同步点的默认超时。

当使用WITH\_DEBUG选项编译了调试支持时，InnoDB存储引擎的同步调试检查是可用的。

关于Debug Sync设施和如何使用同步点的描述，请参见MySQL内部。测试同步。

* -DWITH\_DEFAULT\_FEATURE\_SET=bool

是否使用cmake/build\_configurations/feature\_set.cmake的标志。这个选项在MySQL 8.0.22中被删除。

* -DWITH\_EDITLINE=value

要使用哪个libedit/editline库。允许的值是捆绑（默认）和系统。

* -DWITH\_GMOCK=path\_name

到googlemock分布的路径，用于基于Google Test的单元测试。该选项的值是分布Zip文件的路径。或者，将WITH\_GMOCK环境变量设置为路径名。也可以使用-DENABLE\_DOWNLOADS=1，这样CMake就会从GitHub下载发行版。

如果你构建的MySQL没有基于Google Test的单元测试（通过配置wihout WITH\_GMOCK），CMake会显示一条信息，说明如何下载它。

从MySQL 8.0.25开始，MySQL源码发行版捆绑了Google Test源代码，用于运行基于Google Test的单元测试。因此，从该版本开始，WITH\_GMOCK和ENABLE\_DOWNLOADS CMake选项被删除，如果被指定，则被忽略。

* -DWITH\_ICU={icu\_type|path\_name}。

MySQL使用International Components for Unicode（ICU）来支持正则表达式操作。WITH\_ICU选项表示要包括的ICU支持类型或要使用的ICU安装路径名称。

* icu\_type可以是以下值之一。
  + bundled:使用与发行版捆绑的ICU库。这是默认的，也是唯一支持Windows的选项。
  + system: 使用系统ICU库。
* path\_name 是要使用的 ICU 安装的路径名称。这可能比使用icu\_type值system更好，因为它可以防止CMake检测到并使用系统上安装的旧的或不正确的ICU版本。(另一种允许的方式是将 WITH\_ICU 设为 system， 并将 CMAKE\_PREFIX\_PATH 选项设为 path\_name。)
* -DWITH\_INNODB\_EXTRA\_DEBUG=bool

是否包含额外的InnoDB调试支持。

启用 WITH\_INNODB\_EXTRA\_DEBUG 会打开额外的 InnoDB 调试检查。这个选项只有在 WITH\_DEBUG 被启用时才能被启用。

* -DWITH\_INNODB\_MEMCACHED=bool

是否生成memcached共享库（libmemcached.so和innodb\_engine.so）。

* -DWITH\_JEMALLOC=bool

是否用-ljemalloc链接。如果启用，内置的malloc()、calloc()、realloc()和free()程序将被禁用。默认是关闭。

WITH\_JEMALLOC和WITH\_TCMALLOC是相互排斥的。

这个选项是在MySQL 8.0.16中添加的。

* -DWITH\_KEYRING\_TEST=bool

是否要建立伴随keyring\_file插件的测试程序。默认是关闭。测试文件的源代码位于plugin/keyring/keyring-test目录中。

* -DWITH\_LIBEVENT=string

要使用哪个libevent库。允许的值是捆绑（默认）和系统。在MySQL 8.0.21之前，如果你指定了system，如果存在的话，就会使用系统libevent库，否则就会发生错误。在MySQL 8.0.21及以后的版本中，如果指定了system而没有找到系统libevent库，则无论如何都会发生错误，并且不使用捆绑的libevent。

InnoDB memcached、X Plugin和MySQL Router需要libevent库。

-DWITH\_LIBWRAP=bool

是否包括libwrap（TCP包装器）支持。

* -DWITH\_LOCK\_ORDER=bool

是否启用LOCK\_ORDER工具。默认情况下，这个选项是禁用的，服务器构建不包含工具。如果启用了工具，LOCK\_ORDER工具是可用的，可以按照第5.9.3节 "LOCK\_ORDER工具 "中的描述使用。

注意

在启用WITH\_LOCK\_ORDER选项时，MySQL的构建需要使用flex程序。

这个选项是在MySQL 8.0.17中添加的。

* -DWITH\_LSAN=bool

是否运行LeakSanitizer，而不运行AddressSanitizer。默认为关闭。

这个选项是在MySQL 8.0.16中添加的。

* -DWITH\_LTO=bool

如果编译器支持，是否启用链接时优化器。默认是关闭，除非启用了PROFILE\_USE。

这个选项是在MySQL 8.0.13中添加的。

* -DWITH\_LZ4=lz4\_type

WITH\_LZ4表示zlib支持的来源。

bundled:使用与发行版捆绑的lz4库。这是默认的。

system: 使用系统的 lz4 库。如果 WITH\_LZ4 被设置为这个值， lz4\_decompress 工具不会被建立。在这种情况下，可以使用系统的 lz4 命令来代替。

* -DWITH\_LZMA=lzma\_type

lzma\_type可以是下列值之一。

* **bundled**:的。使用与发行版捆绑的LZMA库。这是默认的。
* system: 使用系统LZMA库。

这个选项在MySQL 8.0.16中被删除。

-DWITH\_MECAB={disabled|system|path\_name}。

使用该选项来编译MeCab解析器。如果你已经将MeCab安装到它的默认安装目录，请设置-DWITH\_MECAB=system。系统选项适用于从源码或使用本地软件包管理工具从二进制文件进行的MeCab安装。如果您将MeCab安装到一个自定义的安装目录，请指定MeCab的安装路径。例如，-DWITH\_MECAB=/opt/mecab。如果系统选项不起作用，指定MeCab的安装路径在所有情况下都应该起作用。

关于相关信息，请参见第12.10.9节，"MeCab全文解析器插件"。

* -DWITH\_MSAN=bool

是否启用MemorySanitizer，用于支持它的编译器。默认是关闭。

为了使该选项在启用时产生效果，所有与MySQL链接的库必须在启用该选项的情况下进行编译。

* -DWITH\_MSCRT\_DEBUG=bool

是否启用Visual Studio CRT内存泄漏跟踪。默认为关闭。

* -DWITH\_MYSQLX=bool

是否支持X插件的构建。默认为ON。参见第20章，将MySQL用作文档存储。

* -DWITH\_NUMA=bool

明确设置NUMA内存分配策略。CMake根据当前平台是否有NUMA支持来设置默认的WITH\_NUMA值。对于没有NUMA支持的平台，CMake的行为如下。

在没有NUMA选项的情况下（正常情况），CMake继续正常工作，只产生这个警告。NUMA库缺失或所需版本不可用

在-DWITH\_NUMA=ON的情况下，CMake会以这样的错误中止。NUMA库缺失或所需版本不可用

* -DWITH\_PROTOBUF=protobuf\_type

使用哪个协议缓冲包。protobuf\_type可以是下列值之一。

bundled:使用与发行版捆绑的软件包。这是默认的。可以选择使用INSTALL\_PRIV\_LIBDIR来修改动态Protobuf库目录。

system。使用安装在系统上的软件包。

其他值会被忽略，并回退到捆绑的状态。

* -DWITH\_RAPID=bool

是否要构建快速开发周期插件。启用时，会在构建树中创建一个包含这些插件的快速目录。当禁用时，在构建树中不创建快速目录。默认值是ON，除非快速目录从源树中删除，在这种情况下，默认值变为OFF。

* -DWITH\_RAPIDJSON=rapidjson\_type

包含的RapidJSON库支持类型。 rapidjson\_type可以是以下值之一。

bundled:使用与发行版捆绑的 RapidJSON 库。这是默认的。

系统。使用系统的RapidJSON库。需要1.1.0或更高的版本。

这个选项是在MySQL 8.0.13中添加的。

* -DWITH\_RE2=re2\_type

要包括的RE2库支持类型。re2\_type可以是以下值之一。

bundled:使用与发行版捆绑的RE2库。这是默认的。

系统。使用系统RE2库。

从MySQL 8.0.18开始，MySQL不再使用RE2库，这个选项被删除。

* -DWITH\_ROUTER=bool

是否建立MySQL Router。默认为ON。

这个选项是在MySQL 8.0.16中添加的。

* -DWITH\_SSL={ssl\_type|path\_name}。

为了支持加密连接、用于随机数生成的熵以及其他与加密有关的操作，必须使用SSL库构建MySQL。这个选项指定了要使用哪个SSL库。

* ssl\_type可以是以下值之一。
  + 系统。使用系统的OpenSSL库。这是默认值。

在macOS和Windows上，使用system配置MySQL，就像调用CMake时path\_name指向一个手动安装的OpenSSL库。这是因为它们没有系统SSL库。在macOS上，brew install openssl会安装到/usr/local/opt/openssl，这样系统就能找到它。在Windows上，它会检查%ProgramFiles%/OpenSSL、%ProgramFiles%/OpenSSL-Win32、%ProgramFiles%/OpenSSL-Win64、C:/OpenSSL、C:/OpenSSL-Win32和C:/OpenSSL-Win64。

* yes这是系统的一个同义词。

path\_name是要使用的OpenSSL安装的路径名称。这比使用系统的ssl\_type值要好，因为它可以防止CMake检测到并使用系统上安装的旧的或不正确的OpenSSL版本。另一种允许的方式是将WITH\_SSL设置为system，并将CMAKE\_PREFIX\_PATH选项设置为path\_name）。

关于配置SSL库的其他信息，参见第2.9.6节 "配置SSL库支持"。

* -DWITH\_SYSTEMD=bool

是否启用 systemd 支持文件的安装。默认情况下，该选项为禁用。启用后，将安装systemd支持文件，而不安装mysqld\_safe和System V初始化脚本等脚本。在 systemd 不可用的平台上，启用 WITH\_SYSTEMD 会导致 CMake 出错。

关于使用systemd的更多信息，请参见2.5.9节 "用systemd管理MySQL服务器"。该节还包括关于指定先前在 [mysqld\_safe] 选项组中指定的选项的信息。由于使用systemd时，mysqld\_safe不会被安装，因此此类选项必须以其他方式指定。

* -DWITH\_SYSTEM\_LIBS=bool

该选项作为一个 "总括 "选项，为以下没有明确设置的CMake选项设置系统值。with\_curl, with\_editline, with\_icu, with\_libevent, with\_lz4, with\_lzma, with\_protobuf, with\_re2, with\_ssl, with\_zlib, with\_zstd。

* -DWITH\_SYSTEMD\_DEBUG=bool

对于使用systemd运行MySQL的平台，是否要产生额外的systemd调试信息。默认为关闭。

该选项在MySQL 8.0.22中加入。

* -DWITH\_TCMALLOC=bool

是否与-ltcmalloc链接。如果启用，内置的malloc()、calloc()、realloc()和free()程序将被禁用。默认是关闭。

WITH\_TCMALLOC和WITH\_JEMALLOC是相互排斥的。

这个选项是在MySQL 8.0.22中添加的。

* -DWITH\_TEST\_TRACE\_PLUGIN=bool

是否要建立测试协议跟踪客户端插件（见使用测试协议跟踪插件）。默认情况下，这个选项是禁用的。除非启用WITH\_CLIENT\_PROTOCOL\_TRACING选项，否则启用该选项没有效果。如果MySQL在配置时启用了这两个选项，libmysqlclient客户端库在构建时就内置了测试协议跟踪插件，而且所有标准的MySQL客户端都会加载该插件。然而，即使测试插件被启用，它在默认情况下也没有效果。对该插件的控制是通过环境变量来实现的；参见使用测试协议跟踪插件。

注意

如果你想使用你自己的协议跟踪插件，不要启用WITH\_TEST\_TRACE\_PLUGIN选项，因为一次只能加载一个这样的插件，如果试图加载第二个，会出现错误。如果你已经在启用了测试协议跟踪插件的情况下构建了MySQL，看看它是如何工作的，你必须在使用你自己的插件之前重建没有它的MySQL。

关于编写跟踪插件的信息，见编写协议跟踪插件。

* -DWITH\_TSAN=bool

是否为支持它的编译器启用ThreadSanitizer。默认是关闭。

* -DWITH\_UBSAN=bool

是否启用未定义行为净化器（Undefined Behavior Sanitizer），适用于支持该功能的编译器。缺省是关闭。

* -DWITH\_UNIT\_TEST={ON|OFF}。

如果启用，用单元测试编译MySQL。默认为ON，除非服务器没有被编译。

* -dwith\_unixodbc=1

启用unixODBC支持，用于Connector/ODBC。

* -DWITH\_VALGRIND=bool

是否在Valgrind头文件中进行编译，这将Valgrind API暴露给MySQL代码。默认是关闭。

为了生成一个Valgrind感知的调试版本，-DWITH\_VALGRIND=1通常与-DWITH\_DEBUG=1结合。参见构建调试配置。

* -DWITH\_ZLIB=zlib\_type

一些功能要求服务器在构建时支持压缩库，比如COMPRESS()和UNCOMPRESS()函数，以及客户端/服务器协议的压缩。WITH\_ZLIB表示zlib支持的来源。

* bundled:使用与发行版捆绑的zlib库。这是默认的。
* system: 使用系统的 zlib 库。如果 WITH\_ZLIB 被设置为这个值，zlib\_decompress 工具不会被构建。在这种情况下，可以使用系统的 openssl zlib 命令来代替。
* -DWITH\_ZSTD=zstd\_type

使用zstd算法的连接压缩（见第4.2.8节，"连接压缩控制"）要求服务器在构建时支持zstd库。WITH\_ZSTD表示zstd支持的来源。

bundled:使用与发行版捆绑的 zstd 库。这是默认的。

system: 使用系统的zstd库。

这个选项是在MySQL 8.0.18中添加的。

**编译器标志**

* -DCMAKE\_C\_FLAGS="flags"

用于C语言编译器的标志。

* -DCMAKE\_CXX\_FLAGS="flags"

用于C++编译器的标志。

* -DWITH\_DEFAULT\_COMPILER\_OPTIONS=bool

是否使用cmake/build\_configurations/compiler\_options.cmake中的标志。

注意

所有优化标志都是由MySQL构建团队精心选择和测试的。覆盖它们可能导致意外的结果，风险自负。

要指定你自己的C和C++编译器标志，对于不影响优化的标志，使用CMAKE\_C\_FLAGS和CMAKE\_CXX\_FLAGS CMake选项。

当提供你自己的编译器标志时，你可能想同时指定CMAKE\_BUILD\_TYPE。

例如，要在64位的Linux机器上创建一个32位的发布版本，请这样做。

mkdir bld

cd bld

cmake .. -DCMAKE\_C\_FLAGS=-m32 \

-DCMAKE\_CXX\_FLAGS=-m32 \

-DCMAKE\_BUILD\_TYPE=RelWithDebInfo

如果你设置了影响优化的标志（-Onumber），你必须设置CMAKE\_C\_FLAGS\_build\_type和/或CMAKE\_CXX\_FLAGS\_build\_type选项，其中build\_type对应于CMAKE\_BUILD\_TYPE值。要为默认的编译类型（RelWithDebInfo）指定不同的优化，请设置CMAKE\_C\_FLAGS\_RELWITHDEBINFO和CMAKE\_CXX\_FLAGS\_RELWITHDEBINFO选项。例如，要在Linux上用-O3和调试符号进行编译，请这样做。

cmake .. -DCMAKE\_C\_FLAGS\_RELWITHDEBINFO="-O3 -g" \

-DCMAKE\_CXX\_FLAGS\_RELWITHDEBINFO="-O3 -g"

**用于编译NDB集群的CMake选项**

以下选项是在构建支持NDB Cluster的MySQL 8.0源时使用的。

* -DMEMCACHED\_HOME=dir\_name

使用安装在dir\_name指示的系统目录中的memcached（1.6或更高版本）进行构建。在构建中使用的该安装文件包括memcached二进制文件、头文件和库，以及memcached\_utilities库和头文件engine\_testapp.h。

在使用捆绑的memcached源（WITH\_BUNDLED\_MEMCACHED选项）构建ndbmemcache时，你必须不设置这个选项；换句话说，默认情况下使用捆绑的源）。

虽然额外的CMake选项--例如SASL授权和提供dtrace支持--在从外部资源编译memcached时可以使用，但这些选项目前还没有对与NDB Cluster捆绑的memcached源启用。

* -ndb\_utils\_link\_dynamic={ON|OFF}。

控制NDB工具（如ndb\_drop\_table）是与ndbclient静态链接（OFF）还是动态链接（ON）；默认为OFF（静态链接）。通常在构建这些工具时使用静态链接，以避免LD\_LIBRARY\_PATH的问题，或者在安装多个版本的ndbclient时使用。这个选项是为了创建Docker镜像，以及其他可能的情况，即目标环境受到精确控制，并且希望减少镜像的大小。

在NDB 8.0.22中添加。

* -dwith\_bundled\_libevent={On|off}

在构建支持ndbmemcached的NDB Cluster时，使用NDB Cluster源中包含的libevent。默认情况下是启用的。OFF会导致使用系统的libevent来代替。

* -dwith\_bundled\_memcached={on|off}

构建包含在NDB Cluster源代码树中的memcached源，然后在构建ndbmemcache引擎时使用产生的memcached服务器。在这种情况下，make install将memcached二进制文件放在安装bin目录下，将ndbmemcache引擎共享库文件ndb\_engine.so放在安装lib目录下。

这个选项在默认情况下是打开的。

-DWITH\_CLASSPATH=path

设置用于构建NDB Cluster Connector for Java的classpath。默认为空。如果使用-DWITH\_NDB\_JAVA=OFF，这个选项将被忽略。

-DWITH\_ERROR\_INSERT={ON|OFF}在NDB集群连接器中启用错误注入。

启用NDB内核中的错误注入。仅用于测试；不打算用于构建生产二进制文件。默认值是OFF。

* -dwith\_ndbcluster\_storage\_engine={on|off}

这是 WITH\_NDBCLUSTER 的一个别名。

-dwith\_ndbcluster={on|off} 构建并链接对NBCLUSTER的支持。

在mysqld中建立并链接对NDB（NDBCLUSTER）存储引擎的支持。默认为ON。

-dwith\_ndbmtd={on|off}

建立多线程数据节点可执行文件ndbmtd。默认值为ON。

* -dwith\_ndb\_binlog={on|off}默认为打开。

在使用该选项构建的mysqld中默认启用二进制日志。默认为打开。

* -dwith\_ndb\_debug={on|off} 启用构建调试版本。

启用构建NDB Cluster二进制文件的调试版本。默认情况下为关闭。

* -dwith\_ndb\_java={on|off} 启用构建带有Java的NDB集群。

启用构建NDB Cluster的Java支持，包括ClusterJ。

该选项默认为打开。如果你不希望用Java支持来编译NDB Cluster，你必须在运行CMake时指定-DWITH\_NDB\_JAVA=OFF来明确禁用它。否则，如果找不到Java，配置构建就会失败。

* -DWITH\_NDB\_PORT=port

导致所构建的NDB Cluster管理服务器（ndb\_mgmd）默认使用这个端口。如果这个选项没有被设置，那么生成的管理服务器默认会尝试使用1186端口。

* -dwith\_ndb\_test={on|off}。

如果启用，包括一组NDB API测试程序。默认为关闭。

* -dwith\_plugin\_ndbcluster={on|off} 如果启用，包括一组NDB API测试程序。

WITH\_NDBCLUSTER的别名。

### 2.9.8 处理编译MySQL时遇到的问题

许多问题的解决方案涉及重新配置。如果你真的重新配置，请注意以下几点。

* 如果CMake在之前被运行后被运行，它可能会使用在之前的调用中收集的信息。这些信息被保存在CMakeCache.txt中。当CMake启动时，它会寻找该文件并读取其内容（如果它存在），前提是这些信息仍然正确。当你重新配置时，这个假设就无效了。
* 每次运行CMake时，你必须再次运行make来重新编译。然而，你可能想先删除以前构建的旧对象文件，因为它们是用不同的配置选项编译的。

为了防止旧的对象文件或配置信息被使用，在重新运行CMake之前运行以下命令。

在Unix上。

shell> **make clean**

shell> **rm CMakeCache.txt**

在Windows上。

shell> **devenv MySQL.sln /clean**

shell> **del CMakeCache.txt**

如果你在源码树之外构建，在重新运行CMake之前删除并重新创建你的构建目录。关于在源码树外构建的说明，见如何用CMake构建MySQL服务器。

在某些系统上，由于系统包含文件的不同，可能会出现警告。下面的列表描述了在编译MySQL时发现的最常发生的其他问题。

要定义使用哪个C和C++编译器，你可以定义CC和CXX环境变量。例如

shell> **CC=gcc**

shell> **CXX=g++**

shell> **export CC CXX**

要指定你自己的C和C++编译器标志，可以使用CMAKE\_C\_FLAGS和CMAKE\_CXX\_FLAGS CMake选项。参见编译器标志。

要查看你可能需要指定哪些标志，用--cflags和--cxxflags选项调用mysql\_config。

* 要查看在编译阶段执行什么命令，在使用CMake配置MySQL后，运行make VERBOSE=1，而不是仅仅运行make。
* 如果编译失败，检查MYSQL\_MAINTAINER\_MODE选项是否被启用。这种模式导致编译器警告变成错误，所以禁用它可能会使编译继续进行。
* 如果你的编译失败了，出现了以下的错误，你必须把你的make版本升级到GNU make。
* make: Fatal error in reader: Makefile, line 18:
* Badly formed macro assignment

或者。

make: file `Makefile' line 18: Must be a separator (:

或者。

pthread.h: No such file or directory

众所周知，Solaris和FreeBSD的make程序很麻烦。

已知GNU make 3.75可以工作。

* sql\_yacc.cc文件是由sql\_yacc.yy生成的。通常，构建过程不需要创建sql\_yacc.cc，因为MySQL带有一个预先生成的副本。然而，如果你确实需要重新创建它，你可能会遇到这个错误。

"sql\_yacc.yy"，第xxx行致命的：默认动作导致潜在的......

这是一个迹象，表明你的yacc版本有缺陷。你可能需要安装最新版本的bison（yacc的GNU版本）并使用它来代替。

早于1.75的bison版本可能会报告这个错误。

sql\_yacc.yy:#####: fatal error: maximum table size (32767) exceeded

实际上并没有超过最大表的大小；这个错误是由旧版本的bison的bug引起的。

关于获取或更新工具的信息，请参见第2.9节 "从源头安装MySQL "中的系统要求。

### 2.9.9 MySQL配置和第三方工具

需要从MySQL源头确定MySQL版本的第三方工具可以读取顶级源头目录中的VERSION文件。该文件分别列出了版本的各个部分。例如，如果版本是MySQL 8.0.4-rc，该文件看起来像这样。

MYSQL\_VERSION\_MAJOR=8

MYSQL\_VERSION\_MINOR=0

MYSQL\_VERSION\_PATCH=4

MYSQL\_VERSION\_EXTRA=-rc

如果源码不是通用的（GA）版本，MYSQL\_VERSION\_EXTRA的值是不空的。在刚才的例子中，该值对应于 "Release Candidate"。

要从版本组件中构建一个五位数的数字，使用这个公式。

MYSQL\_VERSION\_MAJOR\*10000 + MYSQL\_VERSION\_MINOR\*100 + MYSQL\_VERSION\_PATCH

### 2.9.10 生成MySQL Doxygen文档内容

MySQL源代码包含使用Doxygen编写的内部文档。生成的Doxygen内容可在https://dev.mysql.com/doc/index-other.html。也可以使用下面的程序从MySQL源码分布中本地生成这些内容。

1、安装doxygen 1.8.11或更高版本。分发版可在这里获得：http://www.doxygen.nl/。

安装doxygen后，验证版本号。

shell> **doxygen --version**

1.8.13

2、安装PlantUML。

当你在Windows上安装PlantUML时（在Windows 10上测试），你必须至少以管理员身份运行一次，以便它创建注册表键。打开一个管理员控制台，运行这个命令。

shell> java -jar ***path-to-plantuml.jar***

该命令应该打开一个GUI窗口，并且在控制台中没有返回错误。

3、将 PLANTUML\_JAR\_PATH 环境设置为你安装 PlantUML 的位置。比如说

shell> export PLANTUML\_JAR\_PATH=***path-to-plantuml.jar***

4、安装Graphviz dot命令。

安装Graphviz后，验证dot的可用性。例如。

shell> **which dot**

/usr/bin/dot

shell> **dot -V**

dot - graphviz version 2.28.0 (20130928.0220)

5、改变位置到你的MySQL源分发的顶层目录，并做以下工作。

首先，执行cmake。

shell> cd your-mysql-source-directory

shell> mkdir bld

shell> cd bld

shell> cmake ..

接下来，生成doxygen文档。

shell> make doxygen

检查错误日志。它可以在顶层目录的doxyerror.log文件中找到。假设构建成功执行，用浏览器查看生成的输出。例如。

shell> firefox doxygen/html/index.html

## 2.10 安装后的设置和测试

本节讨论了安装MySQL后应执行的任务。

* 如果有必要，初始化数据目录并创建MySQL授予表。对于某些MySQL安装方法，数据目录初始化可能会自动为你完成。
  + 由MySQL安装程序执行的Windows安装操作。
  + 在Linux上使用Oracle的服务器RPM或Debian发行版进行安装。
  + 在许多平台上使用本地打包系统进行安装，包括Debian Linux、Ubuntu Linux、Gentoo Linux和其他。
  + 在macOS上使用DMG发行版进行安装。

对于其他平台和安装类型，你必须手动初始化数据目录。这包括在Unix和类Unix系统上从通用的二进制和源码发行版进行安装，以及在Windows上从ZIP档案包进行安装。有关说明，请参阅第2.10.1节，"初始化数据目录"。

* 启动服务器并确保它可以被访问。说明见第2.10.2节，"启动服务器"，和第2.10.3节，"测试服务器"。
* 如果在数据目录初始化过程中没有为初始根账户分配密码，请在授予表中为其分配密码。密码可以防止对MySQL服务器的非法访问。有关说明，见第2.10.4节 "确保初始MySQL账户的安全"。
* 可以选择安排服务器在你的系统启动和停止时自动启动和停止。有关说明，见第2.10.5节 "自动启动和停止MySQL"。
* 可选地，填充时区表，以便能够识别指定的时区。有关说明，见第5.1.15节，"MySQL服务器时区支持"。

当你准备创建其他用户账户时，你可以在第6.2节 "访问控制和账户管理 "中找到关于MySQL访问控制系统和账户管理的信息。

### 2.10.1 初始化数据目录

在MySQL安装后，必须初始化数据目录，包括mysql系统模式中的表。

* 对于某些MySQL安装方法，数据目录的初始化是自动的，如第2.10节 "安装后的设置和测试 "中所述。
* 对于其他安装方法，你必须手动初始化数据目录。这些方法包括在Unix和类似Unix的系统上从通用的二进制和源码分布中安装，以及在Windows上从ZIP档案包中安装。

本节描述了对于数据目录初始化不是自动的MySQL安装方法，如何手动初始化数据目录。关于一些建议的命令，可以测试服务器是否可以访问和正常工作，见第2.10.3节 "测试服务器"。

注意

在MySQL 8.0中，默认的认证插件已经从mysql\_native\_password变为caching\_sha2\_password，'root'@'localhost'管理账户默认使用caching\_sha2\_password。如果你喜欢根账户使用以前的默认认证插件（mysql\_native\_password），请参阅caching\_sha2\_password和根管理账户。

* 数据目录初始化概述
* 数据目录初始化程序
* 数据目录初始化期间的服务器操作
* 初始化后根密码的分配

**数据目录初始化概述**

在这里显示的例子中，服务器打算在mysql登录账户的用户ID下运行。如果该账户不存在，可以创建该账户（参见创建mysql用户和组），或者用你计划用于运行服务器的另一个现有登录账户的名称代替。

1、改变位置到你的MySQL安装的顶级目录，通常是/usr/local/mysql（根据需要为你的系统调整路径名称）。

cd /usr/local/mysql

在这个目录中，你可以找到几个文件和子目录，包括包含服务器的bin子目录，以及客户端和实用程序。

2、secure\_file\_priv系统变量将导入和导出操作限制在一个特定的目录中。创建一个目录，其位置可以被指定为该变量的值。

mkdir mysql-files

将目录的用户和组所有权授予mysql用户和mysql组，并适当地设置目录的权限。

chown mysql:mysql mysql-files

chmod 750 mysql-files

3、使用服务器初始化数据目录，包括包含初始MySQL授予表的mysql模式，该表决定用户如何被允许连接到服务器。例如

bin/mysqld --initialize --user=mysql

关于该命令的重要信息，特别是关于你可能使用的命令选项，见数据目录初始化程序。关于服务器如何执行初始化的细节，见数据目录初始化期间的服务器动作。

通常，数据目录的初始化只需要在你第一次安装MySQL之后进行。(对于现有安装的升级，请执行升级程序；见第2.11节，"升级MySQL")。然而，初始化数据目录的命令不会覆盖任何现有的mysql模式表，所以它在任何情况下都可以安全地运行。

4、如果你想在部署服务器时自动支持安全连接，使用mysql\_ssl\_rsa\_setup工具来创建默认的SSL和RSA文件。

bin/mysql\_ssl\_rsa\_setup

更多信息，见第4.4.3节 "mysql\_ssl\_rsa\_setup--创建SSL/RSA文件"。

5、如果没有任何选项文件，服务器以其默认设置启动。见第5.1.2节 "服务器配置默认值"。）要明确指定MySQL服务器在启动时应使用的选项，请将它们放在一个选项文件中，如/etc/my.cnf或/etc/mysql/my.cnf。见第4.2.2.2节，"使用选项文件"。）例如，你可以使用选项文件来设置secure\_file\_priv系统变量。

6、要安排MySQL在系统启动时无需人工干预而启动，见第2.10.5节，"自动启动和停止MySQL"。

7、数据目录初始化在mysql模式中创建时区表，但不填充它们。要这样做，请使用第5.1.15节 "MySQL服务器时区支持 "中的说明。

**数据目录初始化程序**

改变位置到你的MySQL安装的顶级目录，通常是/usr/local/mysql（根据需要为你的系统调整路径名称）。

cd /usr/local/mysql

要初始化数据目录，用--initialize或--initialize-insecure选项调用mysqld，这取决于你是否希望服务器为 "root'@'localhost "账户生成一个随机的初始密码，或者创建没有密码的账户。

* 使用-initialize进行 "默认安全 "安装（即包括生成一个随机的初始root密码）。在这种情况下，密码被标记为过期，你必须选择一个新密码。
* 使用 --initialize-insecure，不生成根密码。这是不安全的；假定你打算在将服务器投入生产使用前及时给账户分配一个密码。

关于分配一个新的'root'@'localhost'密码的说明，请看初始化后的root密码分配。

注意

服务器会将任何信息（包括任何初始密码）写入其标准错误输出。这可能会被重定向到错误日志，所以如果你在屏幕上没有看到这些信息，就去那里看看。关于错误日志的信息，包括它的位置，见第5.4.2节，"错误日志"。

在Windows上，使用--console选项将信息导向控制台。

在Unix和类似Unix的系统中，数据库目录和文件由mysql登录账户拥有是很重要的，这样当你以后运行服务器时就有对它们的读写权限。为了确保这一点，从系统根账户启动mysqld，并包括--user选项，如下面所示。

bin/mysqld --initialize --user=mysql

bin/mysqld --initialize-insecure --user=mysql

或者，在以mysql身份登录时执行mysqld，在这种情况下，你可以省略命令中的--用户选项。

在Windows上，使用这些命令中的一个。

bin\mysqld --initialize --console

bin\mysqld --initialize-insecure --console

注意

如果缺少必要的系统库，数据目录初始化可能会失败。例如，你可能会看到一个类似这样的错误。

bin/mysqld: error while loading shared libraries:

libnuma.so.1: 无法打开共享对象文件。

没有这样的文件或目录

如果发生这种情况，你必须手动或用你系统的软件包管理器安装缺少的库。然后重试数据目录初始化命令。

如果mysqld不能识别安装目录或数据目录的正确位置，可能需要指定其他选项，如-basedir或-datadir。例如（在单行输入命令）。

bin/mysqld --initialize --user=mysql

--basedir=/opt/mysql/mysql

--datadir=/opt/mysql/mysql/data

或者，将相关的选项设置放在一个选项文件中，并将该文件的名称传递给mysqld。对于Unix和类Unix系统，假设选项文件名是/opt/mysql/mysql/etc/my.cnf。把这些行放在文件中。

[mysqld]

basedir=/opt/mysql/mysql

datadir=/opt/mysql/mysql/data

然后按如下方式调用mysqld（在单行输入命令，先输入--defaults-file选项）。

bin/mysqld --defaults-file=/opt/mysql/mysql/etc/my.cnf

--initialize --user=mysql

在Windows上，假设C:\my.ini包含这些行。

[mysqld]

basedir=C:\\Program Files\\MySQL\\MySQL Server 8.0

datadir=D:\\MySQLdata

然后按以下方式调用mysqld（在单行输入命令，先输入--defaults-file选项）。

bin\mysqld --defaults-file=C:\my.ini

--initialize --console

**数据目录初始化期间的服务器操作**

注意

服务器执行的数据目录初始化序列并不能替代mysql\_secure\_installation和mysql\_ssl\_rsa\_setup执行的操作。参见第4.4.2节 "mysql\_secure\_installation--提高MySQL安装安全性"，以及第4.4.3节 "mysql\_ssl\_rsa\_setup--创建SSL/RSA文件"。

当用-initialize或-initialize-insecure选项调用时，mysqld在数据目录初始化序列中执行下列操作。

1、服务器检查数据目录是否存在，如下所示。

* 如果没有数据目录存在，服务器会创建它。
* 如果数据目录存在但不是空的（也就是说，它包含文件或子目录），服务器在产生一个错误信息后退出。
* [ERROR] --initialize specified but the data directory exists. Aborting.

在这种情况下，删除或重命名数据目录，然后重试。

如果每个条目都有一个以句号（.）开头的名称，则允许现有的数据目录为非空。

2、在数据目录中，服务器创建了mysql系统模式及其表，包括数据字典表、授予表、时区表和服务器端帮助表。参见第5.3节，"mysql系统模式"。

3、服务器初始化系统表空间和管理InnoDB表所需的相关数据结构。

注意

在mysqld设置了InnoDB系统表空间后，对表空间特性的某些改变需要设置一个全新的实例。符合条件的变化包括系统表空间中第一个文件的文件名和Undo logs的数量。如果你不想使用默认值，在运行mysqld之前，确保innodb\_data\_file\_path和innodb\_log\_file\_size配置参数的设置在MySQL配置文件中已经就绪。还要确保在必要时指定影响InnoDB文件创建和位置的其他参数，例如innodb\_data\_home\_dir和innodb\_log\_group\_home\_dir。

如果这些选项在你的配置文件中，但该文件不在MySQL默认读取的位置，在运行mysqld时使用-defaults-extra-file选项指定文件位置。

4、服务器创建一个'root'@'localhost'超级用户账户和其他保留账户（见第6.2.9节，"保留账户"）。一些保留账户被锁定，不能被客户使用，但'root'@'localhost'是用于管理的，你应该给它分配一个密码。

服务器对'root'@'localhost'账户的密码的操作取决于你如何调用它。

* 使用 --initialize 但不使用 --initialize-insecure，服务器会生成一个随机密码，将其标记为过期，并写一个显示密码的消息。

[Warning] A temporary password is generated for root@localhost:

iTag\*AfrH5ej

* 使用 --initialize-insecure，（无论是否有 --initialize，因为 --initialize-insecure 意味着 --initialize），服务器不会生成密码或标记为过期，并写下一条警告信息。

[Warning] root@localhost is created with an empty password ! Please

consider switching off the --initialize-insecure option.

关于分配新的'root'@'localhost'密码的说明，请参阅初始化后的root密码分配。

5、服务器会填充用于HELP语句的服务器端帮助表（参见第13.8.3节，"HELP语句"）。服务器不填充时区表。要手动操作，请参见第5.1.15节 "MySQL服务器时区支持"。

6、如果init\_file系统变量被赋予了一个SQL语句文件的名称，服务器就会执行该文件中的语句。这个选项使你能够执行自定义的引导序列。

当服务器在引导模式下运行时，一些限制文件中允许的语句的功能不可用。这些包括与账户管理（如CREATE USER或GRANT）、复制和全局事务标识符有关的语句。

服务器退出。

**初始化后根密码的分配**

在你通过使用 --initialize 或 --initialize-insecure 启动服务器来初始化数据目录后，正常启动服务器（也就是说，没有这两个选项）并为'root @'localhost'账户分配一个新密码。

1、启动服务器。有关说明，请参见第2.10.2节，"启动服务器"。

2、连接到服务器。

* 如果你使用了 --initialize 但没有使用 --initialize-insecure 来初始化数据目录，以root身份连接到服务器。
* mysql -u root -p

然后，在密码提示符下，输入服务器在初始化序列中生成的随机密码。

输入密码：（在此输入随机的根密码

如果你不知道这个密码，请查看服务器的错误日志。

* 如果你使用了--initialize-insecure来初始化数据目录，以root身份连接到服务器，不需要密码。

Enter password: ***(enter the random root password here)***

3、连接后，使用ALTER USER语句分配一个新的root密码。

ALTER USER 'root'@'localhost' IDENTIFIED BY '***root-password'***;

另见第2.10.4节，"确保初始MySQL账户的安全"。

注意

连接到主机127.0.0.1的尝试通常会解析到localhost账户。然而，如果服务器在运行时启用了skip\_name\_resolve，则会失败。如果你打算这样做，请确保存在一个能够接受连接的账户。例如，为了能够以root身份使用--host=127.0.0.1或--host=::1连接，创建这些账户。

CREATE USER 'root'@'127.0.0.1' IDENTIFIED BY '***root-password***';

CREATE USER 'root'@'::1' IDENTIFIED BY '***root-password***';

可以把这些语句放在一个文件中，使用init\_file系统变量来执行，正如在数据目录初始化过程中的服务器操作中所讨论的那样。

### 2.10.2 启动服务器

2.10.2.1 启动MySQL服务器的故障排除问题

本节描述了如何在Unix和类Unix系统上启动服务器。(对于Windows，见第2.3.4.5节，"首次启动服务器"。)对于一些建议的命令，你可以用来测试服务器是否可以访问和正常工作，见第2.10.3节，"测试服务器"。

如果你的安装包括mysqld\_safe，像这样启动MySQL服务器。

shell> **bin/mysqld\_safe --user=mysql &**

注意

对于使用RPM包安装MySQL的Linux系统，服务器的启动和关闭使用systemd而不是mysqld\_safe来管理，mysqld\_safe没有安装。参见第2.5.9节，"用systemd管理MySQL服务器"。

如果你的安装包含systemd支持，请像这样启动服务器。

shell> **systemctl start mysqld**

如果服务名称与mysqld不同，请替换相应的服务名称（例如，SLES系统上的mysql）。

使用非特权（非root）登录账户运行MySQL服务器是很重要的。为了确保这一点，以root身份运行mysqld\_safe，并包括--user选项，如图所示。否则，你应该在以mysql登录时执行该程序，在这种情况下，你可以从命令中省略--用户选项。

关于以非特权用户身份运行MySQL的进一步说明，见第6.1.5节，"如何以普通用户身份运行MySQL"。

如果命令立即失败并打印出mysqld结束，在错误日志（默认情况下是数据目录中的host\_name.err文件）中寻找信息。

如果服务器无法访问它启动的数据目录或读取mysql模式中的授予表，它就会在其错误日志中写一条信息。如果你在进行这一步之前忽略了通过初始化数据目录来创建授予表，或者你在运行初始化数据目录的命令时没有使用--用户选项，就可能出现这种问题。删除数据目录，用--用户选项运行命令。

如果你有其他启动服务器的问题，请参见第2.10.2.1节 "启动MySQL服务器的故障排除"。关于mysqld\_safe的更多信息，参见第4.3.2节，"mysqld\_safe--MySQL服务器启动脚本"。更多关于systemd支持的信息，参见第2.5.9节 "用systemd管理MySQL服务器"。

#### 2.10.2.1 启动MySQL服务器问题的故障排除

本节提供了启动服务器问题的故障排除建议。关于Windows系统的其他建议，请参见第2.3.5节 "Microsoft Windows MySQL Server安装的故障排除"。

如果你在启动服务器时遇到问题，这里有一些事情可以尝试。

* 检查错误日志，看看为什么服务器不能启动。日志文件位于数据目录中（在Windows中通常是C:\Program Files\MySQL\MySQL Server 8.0\data，对于Unix/Linux二进制发行版，是/usr/local/mysql/data，而对于Unix/Linux源发行版，是/usr/local/var）。在数据目录中寻找名称为host\_name.err和host\_name.log的文件，其中host\_name是你的服务器主机的名称。然后检查这些文件的最后几行。使用tail来显示它们。

shell> **tail *host\_name*.err**

shell> **tail *host\_name*.log**

* 指定你所使用的存储引擎需要的任何特殊选项。你可以创建一个 my.cnf 文件，为你计划使用的引擎指定启动选项。如果你打算使用支持事务表的存储引擎（InnoDB、NDB），请确保在启动服务器之前将它们配置成你想要的方式。如果你正在使用InnoDB表，请参阅第15.8节 "InnoDB配置 "的指导原则和第15.14节 "InnoDB启动选项和系统变量 "的选项语法。

尽管存储引擎对你省略的选项使用默认值，但Oracle建议你查看可用的选项，并为任何默认值不适合你安装的选项指定明确的值。

确保服务器知道在哪里可以找到数据目录。mysqld服务器使用该目录作为其当前目录。这是它期望找到数据库的地方，也是它期望写日志文件的地方。服务器还在data目录中写pid（进程ID）文件。

默认的数据目录位置是在编译服务器的时候硬编码的。要确定默认的路径设置是什么，用--verbose和--help选项调用mysqld。如果数据目录位于你系统的其他地方，请在命令行或选项文件中用 mysqld 或 mysqld\_safe 的 --datadir 选项指定该位置。否则，服务器不能正常工作。作为--datadir选项的替代方法，你可以用--basedir指定mysqld安装MySQL的基础目录的位置，mysqld会在那里寻找数据目录。

要检查指定路径选项的效果，用这些选项和--verbose及--help选项调用mysqld。例如，如果你改变位置到安装mysqld的目录，然后运行下面的命令，它显示了以/usr/local为基本目录启动服务器的效果。

shell> **./mysqld --basedir=/usr/local --verbose --help**

你也可以指定其他选项，如--datadir，但--verbose和--help必须是最后一个选项。

一旦你确定了你想要的路径设置，就在没有--verbose和--help的情况下启动服务器。

如果mysqld当前正在运行，你可以通过执行这个命令找出它正在使用的路径设置。

shell> **mysqladmin variables**

或者

shell> **mysqladmin -h *host\_name* variables**

host\_name是MySQL服务器主机的名称。

* 确保服务器可以访问数据目录。数据目录及其内容的所有权和权限必须允许服务器读取和修改它们。

如果你在启动mysqld时得到Errcode 13（这意味着Permission denied），这意味着数据目录或其内容的权限不允许服务器访问。在这种情况下，你要改变所涉及的文件和目录的权限，使服务器有权利使用它们。你也可以以root身份启动服务器，但这引起了安全问题，应该避免。

改变位置到数据目录，并检查数据目录及其内容的所有权，以确保服务器有访问权。例如，如果数据目录是/usr/local/mysql/var，使用这个命令。

shell> **ls -la /usr/local/mysql/var**

如果数据目录或其文件或子目录不属于你用来运行服务器的登录账户，请将其所有权改为该账户。如果该账户名为mysql，使用这些命令。

shell> **chown -R mysql /usr/local/mysql/var**

shell> **chgrp -R mysql /usr/local/mysql/var**

即使有正确的所有权，如果你的系统上有其他安全软件在运行，管理应用程序对文件系统各部分的访问，MySQL可能无法启动。在这种情况下，重新配置该软件，使mysqld能够访问它在正常操作中使用的目录。

* 验证服务器想要使用的网络接口是否可用。

如果出现以下任一错误，意味着其他程序（也许是另一个 mysqld 服务器）正在使用 mysqld 试图使用的 TCP/IP 端口或 Unix 套接字文件。

Can't start server: Bind on TCP/IP port: Address already in use

Can't start server: Bind on unix socket...

使用ps来确定你是否有另一个mysqld服务器在运行。如果是，在再次启动 mysqld 之前关闭该服务器。(如果另一个服务器正在运行，并且你真的想运行多个服务器，你可以在第5.8节 "在一台机器上运行多个MySQL实例 "中找到关于如何做的信息)。

如果没有其他服务器正在运行，执行命令telnet your\_host\_name tcp\_ip\_port\_number。(默认的MySQL端口号是3306。)然后按回车键数次。如果你没有得到一个类似telnet的错误信息。无法连接到远程主机：连接被拒绝，一些其他程序正在使用 mysqld 试图使用的 TCP/IP 端口。追踪这是什么程序并禁用它，或者用--端口选项告诉mysqld监听一个不同的端口。在这种情况下，当使用TCP/IP连接到服务器时，为客户程序指定相同的非默认端口号。

该端口无法访问的另一个原因是，你有一个正在运行的防火墙，阻止了对它的连接。如果是这样，请修改防火墙设置，允许对该端口的访问。

如果服务器启动了，但你不能连接到它，确保你在/etc/hosts中有一个条目，看起来像这样。

127.0.0.1 localhost

如果你不能让mysqld启动，尝试通过使用--debug选项制作一个跟踪文件来找到问题所在。参见第5.9.4节，"DBUG包"。

### 2.10.3 测试服务器

在数据目录被初始化并且你已经启动了服务器之后，执行一些简单的测试以确保它能令人满意地工作。本节假设你的当前位置是MySQL安装目录，并且它有一个bin子目录，包含这里使用的MySQL程序。如果不是这样，请相应地调整命令路径名称。

或者，将bin目录添加到你的PATH环境变量设置中。这可以使你的shell（命令解释器）正确地找到MySQL程序，这样你就可以只输入程序的名称而不是路径名称来运行程序。见第4.2.9节，"设置环境变量"。

使用mysqladmin来验证服务器是否在运行。下面的命令提供了简单的测试，以检查服务器是否启动并对连接作出响应。

shell> **bin/mysqladmin version**

shell> **bin/mysqladmin variables**

如果你不能连接到服务器，指定一个-u root选项，以root身份连接。如果你已经为root账户分配了一个密码，你还需要在命令行中指定-p，并在提示时输入密码。比如说

shell> **bin/mysqladmin -u root -p version**

Enter password: ***(enter root password here)***

mysqladmin版本的输出略有不同，取决于你的平台和MySQL的版本，但应该与这里显示的类似。

shell> **bin/mysqladmin version**

mysqladmin Ver 14.12 Distrib 8.0.26, for pc-linux-gnu on i686

...

Server version 8.0.26

Protocol version 10

Connection Localhost via UNIX socket

UNIX socket /var/lib/mysql/mysql.sock

Uptime: 14 days 5 hours 5 min 21 sec

Threads: 1 Questions: 366 Slow queries: 0

Opens: 0 Flush tables: 1 Open tables: 19

Queries per second avg: 0.000

要看看你还能用mysqladmin做什么，用--help选项调用它。

确认你可以关闭服务器（如果根账户已经有了密码，包括一个-p选项）。

shell> **bin/mysqladmin -u root shutdown**

确认你可以再次启动服务器。通过使用 mysqld\_safe 或直接调用 mysqld 来做到这一点。例如。

shell> **bin/mysqld\_safe --user=mysql &**

如果mysqld\_safe失败，请参阅第2.10.2.1节，"启动MySQL服务器问题的故障排除"。

运行一些简单的测试以验证你可以从服务器中检索信息。输出应该与这里显示的类似。

使用mysqlshow来查看存在哪些数据库。

shell> **bin/mysqlshow**

+--------------------+

| Databases |

+--------------------+

| information\_schema |

| mysql |

| performance\_schema |

| sys |

+--------------------+

已安装的数据库列表可能有所不同，但总是至少包括 mysql 和 information\_schema。

如果你指定了一个数据库名称，mysqlshow会显示该数据库中的表的列表。

shell> **bin/mysqlshow mysql**

Database: mysql

+---------------------------+

| Tables |

+---------------------------+

| columns\_priv |

| component |

| db |

| default\_roles |

| engine\_cost |

| func |

| general\_log |

| global\_grants |

| gtid\_executed |

| help\_category |

| help\_keyword |

| help\_relation |

| help\_topic |

| innodb\_index\_stats |

| innodb\_table\_stats |

| ndb\_binlog\_index |

| password\_history |

| plugin |

| procs\_priv |

| proxies\_priv |

| role\_edges |

| server\_cost |

| servers |

| slave\_master\_info |

| slave\_relay\_log\_info |

| slave\_worker\_info |

| slow\_log |

| tables\_priv |

| time\_zone |

| time\_zone\_leap\_second |

| time\_zone\_name |

| time\_zone\_transition |

| time\_zone\_transition\_type |

| user |

+---------------------------+

使用mysql程序从mysql模式中的一个表中选择信息。

shell> **bin/mysql -e "SELECT User, Host, plugin FROM mysql.user" mysql**

+------+-----------+-----------------------+

| User | Host | plugin |

+------+-----------+-----------------------+

| root | localhost | caching\_sha2\_password |

+------+-----------+-----------------------+

在这一点上，你的服务器正在运行，你可以访问它。如果你还没有给初始账户分配密码，要加强安全，请按照第2.10.4节 "确保初始MySQL账户的安全 "中的说明进行。

关于mysql、mysqladmin和mysqlshow的更多信息，见第4.5.1节 "mysql--MySQL命令行客户端"，第4.5.2节 "mysqladmin--MySQL服务器管理程序"，以及第4.5.7节 "mysqlshow--显示数据库、表和列的信息"。

### 2.10.4 保证初始MySQL账户的安全

MySQL的安装过程涉及初始化数据目录，包括mysql系统模式中定义MySQL账户的授予表。详情请参见第2.10.1节 "初始化数据目录"。

本节描述了如何为在MySQL安装过程中创建的初始根账户分配一个密码，如果你还没有这样做的话。

注意

执行本节所述过程的其他方法。

* 在Windows上，你可以在用MySQL安装程序安装时执行该过程（见第2.3.3节，"Windows的MySQL安装程序"）。
* 在所有平台上，MySQL发行版包括mysql\_secure\_installation，这是一个命令行工具，可自动完成保护MySQL安装的大部分过程。
* 在所有平台上，MySQL Workbench是可用的，并提供了管理用户账户的能力（见第31章，MySQL Workbench ）。

在这种情况下，可能已经为初始账户分配了一个密码。

* 在Windows上，使用MySQL安装程序进行的安装可以让你选择分配密码。
* 使用macOS安装程序进行的安装会生成一个初始随机密码，安装程序会在一个对话框中向用户显示。
* 使用RPM软件包的安装会产生一个初始随机密码，它被写入服务器错误日志。
* 使用Debian软件包的安装可以选择指定一个密码。
* 对于使用mysqld --initialize手动执行的数据目录初始化，mysqld会生成一个初始随机密码，标记为过期，并将其写入服务器错误日志。参见第2.10.1节，"初始化数据目录"。

mysql.user grant表定义了初始MySQL用户账户及其访问权限。安装MySQL只创建一个'root'@'localhost'超级用户账户，该账户拥有所有权限，可以做任何事情。如果root账户有一个空的密码，你的MySQL安装是不受保护的。任何人都可以在没有密码的情况下以root身份连接到MySQL服务器，并被授予所有权限。

root'@'localhost'账户在mysql.proxies\_priv表中也有一行，可以为''@'授予PROXY权限，也就是说，为所有用户和所有主机授予PROXY权限。这使得root可以设置代理用户，也可以将设置代理用户的权限委托给其他账户。见第6.2.18节，"代理用户"。

要为最初的MySQL根账户分配一个密码，请使用以下程序。用你想使用的密码替换例子中的root-password。

如果服务器没有运行，请启动它。有关说明，见第2.10.2节，"启动服务器"。

* 最初的root账户可能有密码，也可能没有。选择下列程序中适用的一种。

如果根账户存在一个已经过期的初始随机密码，用该密码以根身份连接到服务器，然后选择一个新密码。如果数据目录是使用mysqld --initialize初始化的，无论是手动还是使用安装程序，在安装操作中不给你指定密码的选项，就属于这种情况。因为密码存在，你必须使用它来连接到服务器。但是，由于密码已经过期，除了选择一个新的密码外，你不能将该账户用于任何其他目的，直到你选择一个新的密码。

* 如果你不知道最初的随机密码，请在服务器的错误日志中查找。
* 使用该密码以root身份连接到服务器。

shell> **mysql -u root -p**

Enter password: ***(enter the random root password here)***

* 选择一个新的密码来替换随机密码。

mysql> **ALTER USER 'root'@'localhost' IDENTIFIED BY '*root-password*';**

* 如果根账户存在但没有密码，以根身份连接到服务器，不使用密码，然后分配一个密码。如果你使用mysqld --initialize-insecure初始化了数据目录，就是这种情况。

1. 以root身份连接到服务器，不使用密码。

shell> **mysql -u root --skip-password**

1. 指定一个密码。
2. mysql> **ALTER USER 'root'@'localhost' IDENTIFIED BY '*root-password*';**

在给root账户分配了一个密码后，每当你使用该账户连接到服务器时，你必须提供该密码。例如，要使用mysql客户端连接到服务器，使用这个命令。

shell> **mysql -u root -p**

Enter password: ***(enter root password here)***

要用mysqladmin关闭服务器，使用这条命令。

shell> **mysqladmin -u root -p shutdown**

Enter password: ***(enter root password here)***

注意

关于设置密码的其他信息，见第6.2.14节，"分配账户密码"。如果你在设置根密码后忘记了，请参见第B.3.3.2节，"如何重设根密码"。

要设置其他账户，请参见第6.2.8节，"添加账户、分配权限和放弃账户"。

### 2.10.5 自动启动和停止MySQL

本节讨论了启动和停止MySQL服务器的方法。

一般来说，你以这些方式之一启动mysqld服务器。

* 直接调用mysqld。这在任何平台上都适用。
* 在Windows上，你可以设置一个MySQL服务，在Windows启动时自动运行。见第2.3.4.8节，"作为Windows服务启动MySQL"。
* 在Unix和类Unix系统上，你可以调用mysqld\_safe，它试图确定mysqld的适当选项，然后用这些选项运行它。见第4.3.2节，"mysqld\_safe--MySQL服务器启动脚本"。
* 在支持systemd的Linux系统上，你可以用它来控制服务器。参见第2.5.9节，"用systemd管理MySQL服务器"。
* 在使用System V风格的运行目录（即/etc/init.d和运行级特定目录）的系统上，调用mysql.server。这个脚本主要在系统启动和关机时使用。它通常以mysql的名字安装。mysql.server脚本通过调用mysqld\_safe启动服务器。见第4.3.3节，"mysql.server--MySQL服务器启动脚本"。
* 在macOS上，安装一个 launchd 守护进程，以便在系统启动时自动启动 MySQL。该守护进程通过调用mysqld\_safe启动服务器。详情请参见第2.4.3节 "安装和使用MySQL启动守护程序"。一个MySQL首选项窗格也提供了通过系统首选项启动和停止MySQL的控制。见第2.4.4节，"安装和使用MySQL首选项窗格"。
* 在Solaris上，使用服务管理框架（SMF）系统来启动和控制MySQL的启动。

systemd、mysqld\_safe和mysql.server脚本、Solaris SMF和macOS启动项（或MySQL首选项窗格）可用于手动启动服务器，或在系统启动时自动启动。 systemd、mysql.server和启动项还可用于停止服务器。

下表显示了服务器和启动脚本从选项文件中读取哪些选项组。

表2.15 MySQL启动脚本和支持的服务器选项组

|  |  |
| --- | --- |
| **Script** | **Option Groups** |
| [**mysqld**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\programs.html#mysqld) | **[mysqld]**, **[server]**, **[mysqld-*major\_version*]** |
| [**mysqld\_safe**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\programs.html#mysqld-safe) | **[mysqld]**, **[server]**, **[mysqld\_safe]** |
| [**mysql.server**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\programs.html#mysql-server) | **[mysqld]**, **[mysql.server]**, **[server]** |

[mysqld-major\_version]意味着名称为[mysqld-5.7]和[mysqld-8.0]的组被具有5.7.x、8.0.x等版本的服务器读取。这个特性可以用来指定只能由给定版本系列内的服务器读取的选项。

为了向后兼容，mysql.server也读取[mysql\_server]组，mysqld\_safe也读取[safe\_mysqld]组。为了与时俱进，你应该更新你的选项文件，以使用[mysql.server]和[mysqld\_safe]组代替。

关于MySQL配置文件及其结构和内容的更多信息，见第4.2.2.2节 "使用选项文件"。

## 2.11 升级MySQL

本节描述了升级MySQL安装的步骤。

升级是一个常见的程序，因为你会在同一个MySQL版本系列中获得错误修复，或在主要的MySQL版本之间获得重要的功能。你首先在一些测试系统上执行这个程序，以确保一切工作顺利，然后再在生产系统上执行。

注意

在下面的讨论中，必须使用具有管理权限的MySQL账户运行的MySQL命令包括命令行上的-u root以指定MySQL根用户。需要root密码的命令也包括一个-p选项。因为-p后面没有选项值，这类命令会提示输入密码。在提示时输入密码并按回车键。

可以使用mysql命令行客户端执行SQL语句（以root身份连接，确保你有必要的权限）。

### 2.11.1 在你开始之前

在升级前查看本节中的信息。执行任何建议的操作。

* 了解升级过程中可能发生的情况。见第2.11.3节，"MySQL升级过程中的升级内容"。
* 通过创建一个备份来保护你的数据。该备份应包括mysql系统数据库，它包含MySQL数据字典表和系统表。参见第7.2节，"数据库备份方法"。

重要提示

不支持从MySQL 8.0降级到MySQL 5.7，或从MySQL 8.0版本降级到以前的MySQL 8.0版本。唯一支持的替代方法是恢复升级前的备份。因此，在开始升级过程之前，你必须备份你的数据。

查看第2.11.2节，"升级路径"，以确保你的预定升级路径得到支持。

* 查看第2.11.4节 "MySQL 8.0中的变化"，了解你在升级前应注意的变化。有些变化可能需要采取行动。
* 查看第1.3节 "MySQL 8.0中的新内容"，了解废弃和删除的功能。如果你使用这些功能，升级可能需要对其进行修改。
* 查看第1.4节，"MySQL 8.0中增加、废弃或删除的服务器和状态变量及选项"。如果你使用被废弃或删除的变量，升级可能需要改变配置。
* 查看发行说明，了解有关修复、更改和新功能的信息。
* 如果你使用复制，请查看第17.5.3节，"升级复制设置"。
* 升级程序因平台和初始安装的方式而异。使用适用于你当前MySQL安装的程序。
  + 对于非Windows平台上的二进制和基于包的安装，请参阅第2.11.6节，"升级Unix/Linux上的MySQL二进制或基于包的安装"。

注意

对于支持的Linux发行版，升级基于软件包的安装的首选方法是使用MySQL软件库（MySQL Yum Repository、MySQL APT Repository和MySQL SLES Repository）。

* 对于在企业Linux平台或Fedora上使用MySQL Yum Repository的安装，请参阅第2.11.7节 "使用MySQL Yum Repository升级MySQL"。
* 对于使用MySQL APT资源库在Ubuntu上的安装，请参阅第2.11.8节，"使用MySQL APT资源库升级MySQL"。
* 对于使用MySQL SLES资源库在SLES上的安装，请参阅第2.11.9节，"使用MySQL SLES资源库升级MySQL"。
* 对于使用Docker进行的安装，请参阅第2.11.11节，"升级MySQL的Docker安装"。
* 对于在Windows上的安装，请参阅第2.11.10节，"在Windows上升级MySQL"。
* 如果你的MySQL安装包含大量的数据，在原地升级后可能需要很长时间来转换，那么创建一个测试实例来评估需要的转换和执行这些转换所涉及的工作可能是有用的。要创建一个测试实例，制作一个包含mysql数据库和其他没有数据的数据库的MySQL实例的副本。在测试实例上运行升级程序，以评估执行实际数据转换所涉及的工作。
* 当你安装或升级到新版本的MySQL时，建议重建和重新安装MySQL语言接口。这适用于MySQL接口，如PHP mysql扩展和Perl DBD::mysql模块。

### 2.11.2 升级路径

* 支持从MySQL 5.7升级到8.0。然而，只支持在普遍可用（GA）版本之间进行升级。对于MySQL 8.0，要求你从MySQL 5.7 GA版本（5.7.9或更高）升级。不支持从MySQL 5.7的非GA版本进行升级。
* 在升级到下一个版本之前，建议先升级到最新的版本。例如，在升级到MySQL 8.0之前升级到最新的MySQL 5.7版本。不支持跳过版本的升级。例如，不支持直接从MySQL 5.6升级到8.0。
* 一旦一个系列的版本达到普遍可用（GA）状态，就支持在该系列中进行升级（从一个GA版本到另一个GA版本）。例如，支持从MySQL 8.0.x升级到8.0.y。(不支持涉及开发状态的非GA版本的升级。) 也支持跳过一个版本。例如，支持从MySQL 8.0.x升级到8.0.z。MySQL 8.0.11是MySQL 8.0版本系列中第一个GA状态的版本。

### 2.11.3 MySQL升级过程所升级的内容

安装新版本的MySQL可能需要升级现有安装中的这些部分。

* mysql系统模式，它包含存储MySQL服务器运行时所需信息的表（参见第5.3节 "mysql系统模式"）。 mysql模式表分为两大类。
  + 数据字典表，它存储数据库对象元数据。
  + 系统表（也就是剩下的非数据字典表），用于其他操作目的。
* 其他模式，其中一些是内置的，可以被认为是由服务器 "拥有 "的，还有一些不是。
  + 性能模式、INFORMATION\_SCHEMA、ndbinfo、和sys模式。
  + 用户模式。

两个不同的版本号与可能需要升级的安装部分相关。

* 数据字典版本。这适用于数据字典表。
* 服务器版本，也被称为MySQL版本。这适用于系统表和其他模式中的对象。

在这两种情况下，适用于现有MySQL安装的实际版本被存储在数据字典中，而当前的预期版本被编译到新版本的MySQL中。当实际版本低于当前的预期版本时，与该版本相关的那些安装部分必须升级到当前版本。如果两个版本都表明需要升级，那么数据字典的升级必须首先发生。

作为刚才提到的两个不同版本的反映，升级分两步进行。

* 第1步：数据字典升级。

这一步是升级。

* mysql模式中的数据字典表。如果实际的数据字典版本低于当前的预期版本，服务器会用更新的定义创建数据字典表，将持久化的元数据复制到新表中，用新表原子化地替换旧表，并重新初始化数据字典。
* 性能模式，INFORMATION\_SCHEMA，和ndbinfo。
* 步骤2：服务器升级。

这一步包括所有其他升级任务。如果现有的MySQL安装的服务器版本低于新安装的MySQL版本，其他一切都必须升级。

* mysql模式中的系统表（剩余的非数据字典表）。
* sys模式。
* 用户模式。

数据字典的升级（第1步）是服务器的责任，它在启动时根据需要执行这项任务，除非用一个选项来阻止它这样做。从MySQL 8.0.16开始，该选项为--upgrade=NONE，在MySQL 8.0.16之前为--no-dd-upgrade。

如果数据字典已经过期，但服务器被阻止升级，服务器就不会运行，而是以错误退出。例如。

[ERROR] [MY-013381] [Server] Server shutting down because upgrade is

required, yet prohibited by the command line option '--upgrade=NONE'.

[ERROR] [MY-010334] [Server] Failed to initialize DD Storage Engine

[ERROR] [MY-010020] [Server] Data Dictionary initialization failed.

第2步的责任在MySQL 8.0.16中发生了一些变化。

* 在MySQL 8.0.16之前，mysql\_upgrade升级了性能模式、INFORMATION\_SCHEMA和步骤2中描述的对象。预计DBA将在启动服务器后手动调用mysql\_upgrade。
* 从MySQL 8.0.16开始，服务器会执行以前由mysql\_upgrade处理的所有任务。尽管升级仍然是一个两步操作，但服务器同时执行这两步操作，从而使过程更简单。

根据你要升级的MySQL版本，就地升级和逻辑升级中的说明表明服务器是否执行所有升级任务，或者你是否必须在服务器启动后调用mysql\_upgrade。

注意

因为服务器从MySQL 8.0.16开始升级性能模式、INFORMATION\_SCHEMA和步骤2中描述的对象，所以mysql\_upgrade是不需要的，并且从该版本开始被废弃；预计它将在MySQL的未来版本中被删除。

在步骤2中发生的大多数方面在MySQL 8.0.16之前和之后都是一样的，尽管可能需要不同的命令选项来实现一个特定的效果。

从MySQL 8.0.16开始，-升级服务器选项控制服务器是否以及如何在启动时执行自动升级。

* 在没有选项或--upgrade=AUTO的情况下，服务器会升级它确定为过时的任何东西（步骤1和2）。
* 使用 --upgrade=NONE，服务器不升级（跳过步骤 1 和 2），但如果数据字典必须升级，也会以错误退出。服务器不可能在一个过时的数据字典中运行；服务器坚持要么升级，要么退出。
* 如果使用 --upgrade=MINIMAL，服务器会升级数据字典、性能模式和INFORMATION\_SCHEMA，如果需要的话（步骤1）。请注意，在使用这个选项进行升级后，组复制不能被启动，因为复制内部所依赖的系统表没有被更新，而且在其他领域可能也会出现功能下降的情况。
* 使用--upgrade=FORCE，服务器会升级数据字典、性能模式和INFORMATION\_SCHEMA（如果需要的话）（步骤1），并强制升级其他所有东西（步骤2）。由于服务器检查所有模式中的所有对象，预计使用这个选项时服务器启动的时间会更长。

如果服务器认为没有必要的话，FORCE对于强制执行步骤2的操作是非常有用的。FORCE与AUTO的不同之处在于，使用FORCE时，服务器会重新创建系统表，例如帮助表或时区表，如果它们丢失的话。

下面的列表显示了MySQL 8.0.16之前的升级命令和MySQL 8.0.16及以上版本的等效命令。

* 执行正常的升级（必要时步骤1和2）。
  + 在MySQL 8.0.16之前：mysqld然后是mysql\_upgrade
  + 从MySQL 8.0.16开始：mysqld
* 根据需要只执行步骤1。
  + 在MySQL 8.0.16之前：不可能执行步骤1中描述的所有升级任务而排除步骤2中描述的任务。然而，你可以使用mysqld和mysql\_upgrade与-upgrade-system-tables和-skip-sys-schema选项避免升级用户模式和系统模式。
  + 从MySQL 8.0.16开始： mysqld --upgrade=MINIMAL
* 根据需要执行步骤1，并强制执行步骤2。
  + 在MySQL 8.0.16之前：mysqld接着mysql\_upgrade –force
  + 从MySQL 8.0.16开始：mysqld --upgrade=FORCE

在MySQL 8.0.16之前，某些mysql\_upgrade选项影响它所执行的操作。下表显示了从MySQL 8.0.16开始要使用哪些服务器-升级选项值来实现类似的效果。这些不一定是完全等同的，因为一个给定的-升级选项值可能有额外的效果）。

mysql\_upgrade选项服务器选项

|  |  |
| --- | --- |
| **mysql\_upgrade Option** | **Server Option** |
| [--skip-sys-schema](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\programs.html#option_mysql_upgrade_skip-sys-schema) | [--upgrade=NONE](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\server-administration.html#option_mysqld_upgrade) or [--upgrade=MINIMAL](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\server-administration.html#option_mysqld_upgrade) |
| [--upgrade-system-tables](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\programs.html#option_mysql_upgrade_upgrade-system-tables) | [--upgrade=NONE](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\server-administration.html#option_mysqld_upgrade) or [--upgrade=MINIMAL](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\server-administration.html#option_mysqld_upgrade) |
| [--force](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\programs.html#option_mysql_upgrade_force) | [--upgrade=FORCE](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\server-administration.html#option_mysqld_upgrade) |

关于升级步骤2中发生的事情的附加说明。

* 如果系统模式没有安装，第2步将其安装，否则将其升级到当前版本。如果一个sys模式存在但没有版本视图，就会发生错误，因为假设没有这个模式就表示是用户创建的模式。
* A sys schema exists with no sys.version view. If
* you have a user created sys schema, this must be renamed for the
* upgrade to succeed.

要在这种情况下升级，首先要删除或重命名现有的系统模式。然后再次执行升级程序。可能有必要强制执行第2步）。

为了防止系统模式的检查。

* 从MySQL 8.0.16开始：用--upgrade=NONE或--upgrade=MINIMAL选项启动服务器。
* 在MySQL 8.0.16之前：用--skip-sys-schema选项调用mysql\_upgrade。
* 第2步根据需要处理所有用户模式中的所有表。表的检查可能需要很长的时间才能完成。每个表都被锁定，因此在处理时对其他会话不可用。检查和修复操作可能很耗时，特别是对于大表来说。表检查使用CHECK TABLE语句的FOR UPGRADE选项。关于这个选项的详细内容，请参见第13.7.3.2节 "CHECK TABLE语句"。

要防止表检查。

* 从MySQL 8.0.16开始：用 --upgrade=NONE或 --upgrade=MINIMAL选项启动服务器。
* 在MySQL 8.0.16之前：用-upgrade-system-tables选项调用mysql\_upgrade。

要强制检查表。

* 从MySQL 8.0.16开始：用-upgrade=FORCE选项启动服务器。
* 在MySQL 8.0.16之前：用--force选项调用mysql\_upgrade。
* 第2步将MySQL版本号保存在数据目录中一个名为mysql\_upgrade\_info的文件中。

要忽略mysql\_upgrade\_info文件，无论如何都要执行检查。

* 从MySQL 8.0.16开始：用-upgrade=FORCE选项启动服务器。
* 在MySQL 8.0.16之前：用--force选项调用mysql\_upgrade。

注意

mysql\_upgrade\_info文件已被废弃；预计它将在MySQL的未来版本中被删除。

* 第2步用当前的MySQL版本号标记所有检查和修复的表。这确保下次升级检查发生在相同版本的服务器上时，可以确定是否有必要再次检查或修复一个给定的表。
* 第2步升级系统表，确保它们具有当前的结构。无论是服务器还是mysql\_upgrade执行该步骤，都是如此。关于帮助表和时区表的内容，mysql\_upgrade不加载这两类表，而服务器加载帮助表，但不加载时区表。(也就是说，在MySQL 8.0.16之前，服务器只在数据目录初始化时加载帮助表。从MySQL 8.0.16开始，它在初始化和升级时加载帮助表）。) 加载时区表的程序与平台有关，需要DBA作出决定，所以不能自动完成。

### 2.11.4 MySQL 8.0的变化

在升级到MySQL 8.0之前，请审查本节中描述的变化，以确定那些适用于你当前的MySQL安装和应用的变化。执行任何建议的操作。

标记为不兼容的变化是与早期版本的MySQL不兼容的变化，可能需要你在升级前注意。我们的目的是避免这些变化，但偶尔也有必要纠正比版本间不兼容更糟糕的问题。如果适用于你的安装的升级问题涉及不兼容，请遵循描述中给出的指示。

* 数据字典的变化
* caching\_sha2\_password作为首选的认证插件
* 配置变更
* 服务器变化
* InnoDB变化
* SQL的变化
* 更改了服务器默认值

**数据字典的变化**

MySQL服务器8.0纳入了一个全局数据字典，其中包含关于事务表中的数据库对象的信息。在以前的MySQL系列中，字典数据存储在元数据文件和非事务性系统表中。因此，升级程序要求你通过检查特定的先决条件来验证你的安装是否准备好升级。欲了解更多信息，请参见第2.11.5节，"为升级准备您的安装"。启用了数据字典的服务器需要一些一般的操作差异；见第14.7节，"数据字典使用差异"。

**caching\_sha2\_password作为首选的认证插件**

caching\_sha2\_password和sha256\_password认证插件比mysql\_native\_password插件提供更安全的密码加密，而且caching\_sha2\_password比sha256\_password提供更好的性能。由于caching\_sha2\_password的这些优越的安全和性能特性，从MySQL 8.0开始，它是首选的认证插件，也是默认的认证插件，而不是mysql\_native\_password。这一变化同时影响到服务器和libmysqlclient客户端库。

* 对于服务器，default\_authentication\_plugin系统变量的默认值从mysql\_native\_password变为caching\_sha2\_password。

这一变化仅适用于安装或升级到MySQL 8.0或更高版本后创建的新账户。对于在升级后的安装中已经存在的账户，其认证插件保持不变。希望切换到caching\_sha2\_password的现有用户可以使用ALTER USER语句来实现。

ALTER USER ***user***

IDENTIFIED WITH caching\_sha2\_password

BY '***password***';

* libmysqlclient库将caching\_sha2\_password作为默认认证插件，而不是mysql\_native\_password。

下面几节将讨论caching\_sha2\_password更突出的作用所带来的影响。

caching\_sha2\_password 兼容性问题和解决方案

* caching\_sha2\_password-兼容的客户端和连接器
* caching\_sha2\_password和根管理账户
* caching\_sha2\_password与复制
* caching\_sha2\_password的兼容性问题和解决方案

重要提示

如果你的MySQL安装必须为8.0以前的客户提供服务，并且在升级到MySQL 8.0或更高版本后遇到了兼容性问题，解决这些问题并恢复8.0以前的兼容性的最简单方法是重新配置服务器以恢复到以前的默认认证插件（mysql\_native\_password）。例如，在服务器选项文件中使用这些行。

[mysqld]

default\_authentication\_plugin=mysql\_native\_password

该设置使8.0前的客户端能够连接到8.0服务器，直到你安装的客户端和连接器升级到知道caching\_sha2\_password。然而，这个设置应该被看作是暂时的，而不是长期或永久的解决方案，因为它导致在设置生效的情况下创建的新账户放弃了由caching\_sha2\_password提供的改进的认证安全性。

使用caching\_sha2\_password比mysql\_native\_password提供更安全的密码散列（以及随之而来的改进的客户端连接认证）。然而，它也有兼容性问题，可能会影响现有的MySQL安装。

* 尚未更新以了解caching\_sha2\_password的客户端和连接器可能难以连接到配置有caching\_sha2\_password作为默认认证插件的MySQL 8.0服务器，甚至无法使用不使用caching\_sha2\_password进行认证的账户。这个问题的发生是因为服务器向客户指定了其默认认证插件的名称。如果客户机或连接器是基于客户机/服务器协议实现的，而该协议不能优雅地处理未被识别的默认认证插件，那么它可能会出现类似以下的错误。

Authentication plugin 'caching\_sha2\_password' is not supported

Authentication plugin 'caching\_sha2\_password' cannot be loaded:

dlopen(/usr/local/mysql/lib/plugin/caching\_sha2\_password.so, 2):

image not found

Warning: mysqli\_connect(): The server requested authentication

method unknown to the client [caching\_sha2\_password]

关于编写连接器以优雅地处理来自服务器的未知默认认证插件的请求的信息，请参阅认证插件连接器-编写注意事项。

* 使用用caching\_sha2\_password认证的账户的客户必须使用安全连接（使用TLS/SSL凭证的TCP、Unix套接字文件或共享内存进行），或支持使用RSA密钥对交换密码的未加密连接。这一安全要求不适用于mysql\_native\_passsword，所以切换到caching\_sha2\_password可能需要额外的配置（见第6.4.1.2节，"缓存SHA-2可插拔认证"）。然而，MySQL 8.0中的客户端连接默认喜欢使用TLS/SSL，所以已经符合该偏好的客户端可能不需要额外的配置。
* 没有被更新以了解caching\_sha2\_password的客户端和连接器不能连接到用caching\_sha2\_password认证的账户，因为它们不承认这个插件是有效的。这是客户端/服务器认证插件兼容性要求的一个特殊例子，在认证插件客户端/服务器兼容性中讨论过）。要解决这个问题，请将客户端与MySQL 8.0或更高版本的libmysqlclient重新链接，或获得一个更新的连接器，以识别caching\_sha2\_password。
* 因为caching\_sha2\_password现在也是libmysqlclient客户端库中的默认认证插件，对于从MySQL 8.0客户端连接到使用mysql\_native\_password（以前的默认认证插件）的账户，认证需要在客户端/服务器协议中进行额外的往返，除非客户端程序被调用时带有-default-auth=mysql\_native\_password选项。

8.0之前的MySQL版本的libmysqlclient客户端库能够连接到MySQL 8.0服务器（使用caching\_sha2\_password认证的账户除外）。这意味着基于libmysqlclient的8.0前客户端也应该能够连接。例子。

* 标准的MySQL客户端，如mysql和mysqladmin是基于libmysqlclient的。
* Perl DBI的DBD::mysql驱动是基于libmysqlclient的。
* MySQL Connector/Python有一个C扩展模块，是基于libmysqlclient的。要使用它，在连接时包括use\_pure=False选项。

当现有的MySQL 8.0安装升级到MySQL 8.0.4或更高版本时，一些基于libmysqlclient的旧客户端可能会 "自动 "升级，如果它们是动态链接的，因为它们使用升级后安装的新客户端库。例如，如果Perl DBI的DBD::mysql驱动使用动态链接，在升级到MySQL 8.0.4或更高版本后，它可以在原地使用libmysqlclient，结果是这样的。

* 在升级之前，使用DBD::mysql的DBI脚本可以连接到MySQL 8.0服务器，除了用caching\_sha2\_password认证的账户。
* 在升级后，同样的脚本也变得能够使用caching\_sha2\_password账户。

然而，出现前面的结果是因为8.0.4之前的MySQL 8.0安装的libmysqlclient实例是二进制兼容的。它们都使用一个共享库的主要版本号为21。对于从MySQL 5.7或更早的版本链接到libmysqlclient的客户端，它们链接到一个具有不同版本号的共享库，该共享库不具有二进制兼容。在这种情况下，客户端必须根据8.0.4或更高版本的libmysqlclient重新编译，以便与MySQL 8.0服务器和缓存\_sha2\_password账户完全兼容。

MySQL Connector/J 5.1至8.0.8能够连接到MySQL 8.0服务器，但使用caching\_sha2\_password验证的账户除外。(Connector/J 8.0.9或更高版本需要连接到caching\_sha2\_password账户。)

使用libmysqlclient以外的客户/服务器协议实现的客户端可能需要升级到能理解新认证插件的较新版本。例如，在PHP中，MySQL的连接通常基于mysqlnd，目前它不知道caching\_sha2\_password。在mysqlnd的更新版本出现之前，使PHP客户端能够连接到MySQL 8.0的方法是重新配置服务器，恢复到mysql\_native\_password作为默认的认证插件，如之前讨论的那样。

如果客户端或连接器支持明确指定默认认证插件的选项，请使用它来命名一个除caching\_sha2\_password之外的插件。例子。

* 一些MySQL客户端支持一个--default-auth选项。(标准的MySQL客户端，如mysql和mysqladmin支持这个选项，但没有它也能成功连接到8.0服务器。然而，其他客户端可能支持一个类似的选项。如果是这样的话，值得一试）。)
* 使用libmysqlclient C API的程序可以用MYSQL\_DEFAULT\_AUTH选项调用mysql\_options()函数。
* 使用客户端/服务器协议的本地Python实现的MySQL Connector/Python脚本可以指定auth\_plugin连接选项。(或者，使用Connector/Python C扩展，它能够连接到MySQL 8.0服务器而不需要auth\_plugin。)

caching\_sha2\_password-兼容的客户端和连接器

如果有一个客户端或连接器已经更新到知道caching\_sha2\_password，那么在连接到配置有caching\_sha2\_password作为默认认证插件的MySQL 8.0服务器时，使用它是确保兼容性的最好方法。

这些客户端和连接器已经升级，以支持caching\_sha2\_password。

* MySQL 8.0（8.0.4或更高）中的libmysqlclient客户端库。标准的MySQL客户端，如mysql和mysqladmin是基于libmysqlclient的，所以它们也是兼容的。
* MySQL 5.7（5.7.23或更高）中的libmysqlclient客户端库。标准的MySQL客户端，如mysql和mysqladmin是基于libmysqlclient的，所以它们也是兼容的。
* MySQL Connector/C++ 1.1.11或更高版本或8.0.7或更高版本。
* MySQL Connector/J 8.0.9或更高版本。
* MySQL Connector/NET 8.0.10或更高版本（通过经典的MySQL协议）。
* MySQL Connector/Node.js 8.0.9或更高。
* PHP：X DevAPI PHP扩展（mysql\_xdevapi）支持caching\_sha2\_password。
* PHP：PDO\_MySQL和ext/mysqli扩展不支持caching\_sha2\_password。此外，当与 7.1.16 之前的 PHP 版本和 7.2.4 之前的 PHP 7.2 一起使用时，即使没有使用 caching\_sha2\_password，它们也无法与 default\_authentication\_plugin=caching\_sha2\_password 连接。

caching\_sha2\_password和根管理账户

对于升级到MySQL 8.0，现有账户的认证插件保持不变，包括'root'@'localhost'管理账户的插件。

对于新的MySQL 8.0安装，当你初始化数据目录时（使用第2.10.1节 "初始化数据目录 "的说明），'root'@'localhost'账户被创建，并且该账户默认使用caching\_sha2\_password。因此，要在数据目录初始化后连接到服务器，你必须使用一个支持caching\_sha2\_password的客户端或连接器。如果你能做到这一点，但更希望根账户在安装后使用mysql\_native\_password，请安装MySQL并像通常那样初始化数据目录。然后以root身份连接到服务器，使用如下的ALTER USER来改变账户认证插件和密码。

ALTER USER 'root'@'localhost'

IDENTIFIED WITH mysql\_native\_password

BY '***password***';

如果你使用的客户端或连接器还不支持缓存\_sha2\_password，你可以使用一个修改过的数据目录初始化过程，在创建账户时就将根账户与mysql\_native\_password关联起来。要做到这一点，请使用这些技术中的任何一种。

* 与--initialize或--initialize-insecure一起提供一个--default-authentication-plugin=mysql\_native\_password选项。
* 在一个选项文件中把default\_authentication\_plugin设置为mysql\_native\_password，并使用-defaults-file选项与-initialize或-initialize-insecure一起命名该选项文件。(在这种情况下，如果你在随后的服务器启动中继续使用该选项文件，新账户将用mysql\_native\_password而不是caching\_sha2\_password创建，除非你从选项文件中删除default\_authentication\_plugin设置。)

caching\_sha2\_password和复制

在所有服务器都已升级到MySQL 8.0.4或更高版本的复制方案中，到源服务器的复制连接可以使用用caching\_sha2\_password认证的账户。对于这样的连接，与使用用caching\_sha2\_password认证的账户的其他客户端的要求相同：使用安全连接或基于RSA的密码交换。

要连接到caching\_sha2\_password账户进行源/副本复制。

* 使用下列任何一个 CHANGE MASTER TO 选项。

MASTER\_SSL = 1

GET\_MASTER\_PUBLIC\_KEY = 1

MASTER\_PUBLIC\_KEY\_PATH='***path to RSA public key file***'

* 另外， 如果在服务器启动时提供了所需的密钥， 您可以使用 RSA 公钥相关的选项。

要连接到组复制的caching\_sha2\_password账户。

* 对于使用OpenSSL构建的MySQL，设置以下任何一个系统变量。

SET GLOBAL group\_replication\_recovery\_use\_ssl = ON;

SET GLOBAL group\_replication\_recovery\_get\_public\_key = 1;

SET GLOBAL group\_replication\_recovery\_public\_key\_path = '***path to RSA public key file***';

* 另外，如果在服务器启动时提供了所需的密钥，你可以使用RSA公钥相关选项。

**配置变更**

* 不兼容的变化。一个MySQL存储引擎现在负责提供它自己的分区处理程序，MySQL服务器不再提供通用的分区支持。InnoDB和NDB是唯一提供本地分区处理程序的存储引擎，在MySQL 8.0中得到支持。使用任何其他存储引擎的分区表必须在升级服务器之前进行更改--要么将其转换为InnoDB或NDB，要么移除其分区，否则之后就无法使用。

关于将MyISAM表转换为InnoDB的信息，参见章节15.6.1.5，"将表从MyISAM转换为InnoDB"。

在MySQL 8.0中，使用没有这种支持的存储引擎的表创建语句会导致分区表的失败，并出现错误（ER\_CHECK\_NOT\_IMPLEMENTED）。如果你使用mysqldump从MySQL 5.7（或更早）创建的转储文件中导入数据库到MySQL 8.0服务器，你必须确保任何创建分区表的语句不会指定一个不支持的存储引擎，要么删除任何对分区的引用，要么指定存储引擎为InnoDB或允许它被默认设置为InnoDB。

注意

第2.11.5节 "为升级准备你的安装 "给出的程序描述了如何在升级到MySQL 8.0之前识别必须改变的分区表。

有关进一步信息，请参见第24.6.2节 "与存储引擎有关的分区限制"。

* 不兼容的改变。一些服务器错误代码不再使用，并已被删除（关于列表，请参见MySQL 8.0中删除的功能）。对其中任何一个进行专门测试的应用程序都应该被更新。
* 重要变化。默认字符集已从latin1改为utf8mb4。这些系统变量受到影响。
  + character\_set\_server和character\_set\_database系统变量的默认值已经从latin1变为utf8mb4。
  + collation\_server和collation\_database系统变量的默认值已经从latin1\_swedish\_ci变为utf8mb4\_0900\_ai\_ci。

因此，新对象的默认字符集和排序与以前不同，除非明确指定字符集和排序。这包括数据库和其中的对象，如表、视图和存储程序。假设使用了以前的默认值，保留它们的一个方法是在my.cnf文件中用这些行启动服务器。

[mysqld]

character\_set\_server=latin1

collation\_server=latin1\_swedish\_ci

在复制的设置中，当从MySQL 5.7升级到8.0时，建议在升级前将默认字符集改回MySQL 5.7中使用的字符集。升级完成后，可将默认字符集改为utf8mb4。

* 不兼容的改变。从MySQL 8.0.11开始，禁止以不同于服务器初始化时使用的设置的 lower\_case\_table\_names设置启动服务器。这个限制是必要的，因为各种数据字典表字段使用的排序是基于服务器初始化时定义的 lower\_case\_table\_names设置，用不同的设置重新启动服务器会在标识符的排序和比较方面引入不一致。

**服务器变化**

* 在MySQL 8.0.11中，与账户管理有关的几个废弃功能已被删除，例如使用GRANT语句来修改用户账户的非特权特性、NO\_AUTO\_CREATE\_USER SQL模式、PASSWORD()函数和old\_passwords系统变量。

从MySQL 5.7到8.0复制引用这些已删除特性的语句可能导致复制失败。应修订使用任何已删除功能的应用程序，以避免这些功能，并尽可能使用替代品，如MySQL 8.0中删除的功能所述。

为了避免在MySQL 8.0上出现启动失败，请从MySQL选项文件中的sql\_mode系统变量设置中删除NO\_AUTO\_CREATE\_USER的任何实例。

将存储程序定义中包括NO\_AUTO\_CREATE\_USER SQL模式的转储文件加载到MySQL 8.0服务器会导致失败。从MySQL 5.7.24和MySQL 8.0.13开始，mysqldump从存储程序定义中删除了NO\_AUTO\_CREATE\_USER。用早期版本的mysqldump创建的转储文件必须手动修改，以删除NO\_AUTO\_CREATE\_USER的实例。

* 在MySQL 8.0.11中，这些废弃的兼容SQL模式被删除。DB2、maxdb、mssql、mysql323、mysql40、oracle、postgresql、no\_field\_options、no\_key\_options和no\_table\_options。它们不能再被分配给sql\_mode系统变量或作为mysqldump --兼容选项的允许值。

移除MAXDB意味着CREATE TABLE或ALTER TABLE的TIMESTAMP数据类型不再被视为DATETIME。

从MySQL 5.7到8.0复制引用已删除的SQL模式的语句可能导致复制失败。这包括复制存储程序（存储过程和函数、触发器和事件）的CREATE语句，这些语句在当前的sql\_mode值包括任何被移除的模式时被执行。使用任何已删除模式的应用程序应进行修改以避免它们。

* 从MySQL 8.0.3开始，空间数据类型允许一个SRID属性，以明确指示存储在该列中的值的空间参考系统（SRS）。见第11.4.1节，"空间数据类型"。

一个具有明确的SRID属性的空间列是受SRID限制的。该列只接受具有该ID的值，并且该列上的SPATIAL索引将被优化器使用。优化器忽略了没有SRID属性的空间列上的SPATIAL索引。参见第8.3.3节 "SPATIAL索引的优化"。如果你想让优化器考虑没有SRID限制的空间列上的SPATIAL索引，应该对每个这样的列进行修改。

* + 验证该列内的所有值是否具有相同的SRID。为了确定几何列col\_name中包含的SRID，使用下面的查询。

SELECT DISTINCT ST\_SRID(***col\_name***) FROM ***tbl\_name***;

如果查询返回多条记录，说明该列包含混合的SRID。在这种情况下，修改其内容，使所有的值都具有相同的SRID。

* 重新定义该列，使其具有一个明确的SRID属性。
* 重新创建SPATIAL索引。
* 在MySQL 8.0.0中删除了几个空间函数，原因是空间函数命名空间的变化，对执行精确操作的函数实施ST\_前缀，或对基于最小边界矩形执行操作的函数实施MBR前缀。在生成的列定义中使用已删除的空间函数可能导致升级失败。在升级之前，运行mysqlcheck --check-upgrade以查找已删除的空间函数，并用它们的ST\_或MBR命名的替代品来替换任何发现的空间函数。关于已删除的空间函数的列表，请参阅《MySQL 8.0中删除的功能》。
* 当执行原地升级到MySQL 8.0.3或更高版本时，BACKUP\_ADMIN权限会自动授予具有RELOAD权限的用户。
* 从MySQL 8.0.13开始，由于基于行或混合复制模式与基于语句的复制模式在处理临时表的方式上存在差异，在运行时切换二进制日志格式方面有新的限制。
  + 如果会话有任何开放的临时表，SET @@SESSION.binlog\_format不能被使用。
  + 如果任何复制通道有任何开放的临时表，SET @@global.binlog\_format和SET @@persist.binlog\_format不能被使用。如果复制通道有开放的临时表，SET @@persist\_only.binlog\_format是允许的，因为与PERSIST不同，PERSIST\_ONLY不会修改运行时的全局系统变量值。
  + 如果任何复制通道应用者正在运行，SET @@global.binlog\_format和SET @@persist.binlog\_format不能被使用。这是因为只有当复制通道的应用者重新启动时，这种改变才会在复制通道上生效，这时复制通道可能有开放的临时表。这种行为比以前更有限制性。如果任何复制通道的应用者正在运行，则允许设置 @@persist\_only.binlog\_format。

**InnoDB变化**

* 基于InnoDB系统表的INFORMATION\_SCHEMA视图被数据字典表的内部系统视图取代。受影响的InnoDB INFORMATION\_SCHEMA视图被重新命名。
* **Table 2.16 Renamed InnoDB Information Schema Views**

|  |  |
| --- | --- |
| **Old Name** | **New Name** |
| **INNODB\_SYS\_COLUMNS** | **INNODB\_COLUMNS** |
| **INNODB\_SYS\_DATAFILES** | **INNODB\_DATAFILES** |
| **INNODB\_SYS\_FIELDS** | **INNODB\_FIELDS** |
| **INNODB\_SYS\_FOREIGN** | **INNODB\_FOREIGN** |
| **INNODB\_SYS\_FOREIGN\_COLS** | **INNODB\_FOREIGN\_COLS** |
| **INNODB\_SYS\_INDEXES** | **INNODB\_INDEXES** |
| **INNODB\_SYS\_TABLES** | **INNODB\_TABLES** |
| **INNODB\_SYS\_TABLESPACES** | **INNODB\_TABLESPACES** |
| **INNODB\_SYS\_TABLESTATS** | **INNODB\_TABLESTATS** |
| **INNODB\_SYS\_VIRTUAL** | **INNODB\_VIRTUAL** |

在升级到MySQL 8.0.3或更高版本后，更新任何引用先前InnoDB INFORMATION\_SCHEMA视图名称的脚本。

* 与MySQL捆绑的zlib库版本从1.2.3版提高到1.2.11版。

zlib 1.2.11中的zlib compressBound()函数，与zlib 1.2.3版本相比，返回压缩给定长度的字节所需的缓冲区大小的估计值略高。compressBound()函数被InnoDB函数调用，这些函数在创建压缩的InnoDB表或在压缩的InnoDB表中插入和更新行时确定允许的最大行大小。因此，CREATE TABLE ... ROW\_FORMAT=COMPRESSED、INSERT和UPDATE操作的行数非常接近最大行数，在早期版本中是成功的，现在可能会失败。为了避免这个问题，在升级之前，在MySQL 8.0测试实例上测试具有大行的压缩InnoDB表的CREATE TABLE语句。

* 随着-innodb-directories功能的引入，用绝对路径或在数据目录之外的位置创建的每表文件和一般表空间文件的位置应该被添加到innodb\_directories参数值中。否则，InnoDB在恢复过程中就无法定位这些文件。要查看表空间文件的位置，查询INFORMATION\_SCHEMA.FILES表。

SELECT TABLESPACE\_NAME, FILE\_NAME FROM INFORMATION\_SCHEMA.FILES \G

* Undo logs不能再驻留在系统表空间中。在MySQL 8.0中，Undo logs默认驻留在两个undo tablespaces中。更多信息，见第15.6.3.4节 "undo tablespaces"。

当从MySQL 5.7升级到MySQL 8.0时，存在于MySQL 5.7实例中的任何undo tablespaces被删除，并由两个新的默认undo tablespaces取代。默认的undo tablespaces是在innodb\_undo\_directory变量定义的位置创建的。如果innodb\_undo\_directory变量未定义，则undo tablespaces将在数据目录中创建。从MySQL 5.7升级到MySQL 8.0需要缓慢关机，以确保MySQL 5.7实例中的undo tablespaces是空的，从而允许它们被安全地删除。

当从早期的MySQL 8.0版本升级到MySQL 8.0.14或更高版本时，由于innodb\_undo\_tablespaces设置大于2而存在于升级前实例中的undo tablespaces被视为用户定义的undo tablespaces，在升级后可以分别使用ALTER UNDO TABLESPACE和DROP UNDO TABLESPACE语法停用和删除。在MySQL 8.0版本系列内的升级可能并不总是需要缓慢关闭，这意味着现有的undo tablespaces可能包含Undo logs。因此，现有的undo tablespaces不会被升级过程删除。

* 不兼容的变化。从MySQL 8.0.17开始，CREATE TABLESPACE ... ADD DATAFILE子句不允许循环目录引用。例如，以下语句中的循环目录引用（/.../）是不允许的。

CREATE TABLESPACE ts1 ADD DATAFILE ts1.ibd '***any\_directory***/../ts1.ibd';

这个限制在Linux上有一个例外，如果前面的目录是一个符号链接，那么循环目录引用是允许的。例如，如果any\_directory是一个符号链接，上面例子中的数据文件路径是允许的。(仍然允许数据文件路径以'.../'开头)。

为了避免升级问题，在升级到MySQL 8.0.17或更高版本之前，从表空间数据文件路径中删除任何循环目录引用。要检查表空间路径，请查询INFORMATION\_SCHEMA.INNODB\_DATAFILES表。

* 由于在MySQL 8.0.14中引入的回归，对于具有分区表和 lower\_case\_table\_names=1的实例，在对大小写敏感的文件系统上从MySQL 5.7或MySQL 8.0.14之前的版本升级到MySQL 8.0.16失败。该故障是由与分区表文件名有关的大小写不匹配问题引起的。引入回归的修复被恢复，这允许从MySQL 5.7或MySQL 8.0 14之前的版本升级到MySQL 8.0.17，以正常运作。然而，在MySQL 8.0.14、8.0.15和8.0.16版本中仍然存在这种退步。

在大小写敏感的文件系统上从MySQL 8.0.14、8.0.15或8.0.16就地升级到MySQL 8.0.17，如果存在分区表且 lower\_case\_table\_names=1，则在将二进制文件或软件包升级到MySQL 8.0.17后启动服务器时出现下列错误。

Upgrading from server version ***version\_number*** with

partitioned tables and lower\_case\_table\_names == 1 on a case sensitive file

system may cause issues, and is therefore prohibited. To upgrade anyway, restart

the new server version with the command line option 'upgrade=FORCE'. When

upgrade is completed, please execute 'RENAME TABLE ***part\_table\_name***

TO ***new\_table\_name***; RENAME TABLE ***new\_table\_name***

TO ***part\_table\_name***;' for each of the partitioned tables.

Please see the documentation for further information.

如果你在升级到MySQL 8.0.17时遇到这个错误，请执行以下解决方法。

1. 用-upgrade=force重新启动服务器，以强制进行升级操作。
2. 用小写的分区名分隔符（#p#或#sp#）来识别分区表的文件名。
3. mysql> SELECT FILE\_NAME FROM INFORMATION\_SCHEMA.FILES WHERE FILE\_NAME LIKE '%#p#%' OR FILE\_NAME LIKE '%#sp#%';
4. 对于确定的每个文件，使用一个临时名称重命名相关的表，然后将表重命名为其原始名称。
5. mysql> RENAME TABLE ***table\_name*** TO ***temporary\_table\_name***;
6. mysql> RENAME TABLE ***temporary\_table\_name*** TO ***table\_name***;
7. 验证没有分区表文件名小写的分区名分隔符（应该返回一个空的结果集）。
8. mysql> **SELECT FILE\_NAME FROM INFORMATION\_SCHEMA.FILES WHERE FILE\_NAME LIKE '%#p#%' OR FILE\_NAME LIKE '%#sp#%';**
9. Empty set (0.00 sec)

e. 在每个重命名的表上运行ANALYZE TABLE，以更新mysql.innodb\_index\_stats和mysql.innodb\_table\_stats表中的优化器统计数据。

由于在MySQL 8.0.14、8.0.15和8.0.16版本中仍然存在回归，在 lower\_case\_table\_names=1的大小写敏感文件系统中，不支持从MySQL 8.0.14、8.0.15或8.0.16导入分区表至MySQL 8.0.17。试图这样做的结果是 "表空间缺少表 "的错误。

* MySQL在构建表空间名称和表分区的文件名称时使用分隔符串。分区名称前有一个 "#p#"分隔符串，子分区名称前有一个 "#sp#"分隔符串，如所示。

***schema\_name***.***table\_name***#p#***partition\_name***#sp#***subpartition\_name***

***table\_name***#p#***partition\_name***#sp#***subpartition\_name***.ibd

历史上，分隔符字符串在对大小写敏感的文件系统（如Linux）上是大写的（#P#和#SP#），而在对大小写不敏感的文件系统（如Windows）上是小写的（#p#和#sp#）。从MySQL 8.0.19开始，定界符串在所有文件系统上都是小写。这一变化防止了在大小写敏感和不敏感的文件系统之间迁移数据目录时的问题。不再使用大写的定界符串。

此外，基于用户指定的分区或子分区名称生成的分区表空间名称和文件名称，可以指定为大写或小写，现在以小写生成（并在内部存储），而不管 lower\_case\_table\_names 设置如何，以确保大小写不敏感。例如，如果以PART\_1的名称创建了一个表分区，那么表空间名称和文件名称将以小写生成。

***schema\_name***.***table\_name***#p#***part\_1***

***table\_name***#p#***part\_1***.ibd

在升级期间，MySQL检查并在必要时进行修改。

* + 磁盘上和数据字典中的分区文件名，以确保小写定界符和分区名称。
  + 数据字典中的分区元数据，以解决以前的错误修复所带来的相关问题。
  + InnoDB统计数据，用于解决以前的错误修复所带来的相关问题。

在表空间导入操作期间，将检查磁盘上的分区表空间文件名，并在必要时进行修改，以确保小写定界符和分区名称。

* 从MySQL 8.0.21开始，在启动时或从MySQL 5.7升级时，如果发现表空间数据文件驻留在未知目录中，将向错误日志写入警告。已知目录是那些由datadir、innodb\_data\_home\_dir和innodb\_directories变量定义的目录。要使一个目录为人所知，就把它添加到innodb\_directories设置中。让目录为人所知可以确保在恢复过程中可以找到数据文件。欲了解更多信息，请参见崩溃恢复期间的表空间发现。

**SQL变化**

* 不兼容的变化。从MySQL 8.0.13开始，GROUP BY子句中被废弃的ASC或DESC限定符已被删除。以前依赖GROUP BY排序的查询可能产生与以前的MySQL版本不同的结果。要产生一个给定的排序顺序，请提供一个ORDER BY子句。

来自MySQL 8.0.12或更低版本的查询和存储程序定义，如果对GROUP BY子句使用ASC或DESC限定词，则应进行修改。否则，升级到MySQL 8.0.13或更高版本可能会失败，复制到MySQL 8.0.13或更高版本的复制服务器也可能失败。

在MySQL 8.0中可能保留了一些在MySQL 5.7中没有保留的关键字。见第9.3节，"关键词和保留字"。这可能导致以前用作标识符的词变得非法。要修复受影响的语句，请使用标识符引号。参见第9.2节，"模式对象名称"。

在升级后，建议你测试应用程序代码中指定的优化器提示，以确保这些提示仍然是实现所需优化策略的必要条件。优化器的增强有时会使某些优化器提示成为不必要的。在某些情况下，不必要的优化器提示甚至可能产生反作用。

* 不兼容的变化。在MySQL 5.7中，为InnoDB表指定一个没有CONSTRAINT符号子的FOREIGN KEY定义，或指定没有符号的CONSTRAINT关键字，导致InnoDB使用一个生成的约束名称。这种行为在MySQL 8.0中有所改变，InnoDB使用FOREIGN KEY index\_name值而不是生成的名称。因为约束名称必须在每个模式（数据库）中是唯一的，这种变化导致了由于外键索引名称在每个模式中不唯一的错误。为了避免这种错误，新的约束命名行为已经在MySQL 8.0.16中被恢复了，InnoDB再次使用生成的约束名称。

为了与InnoDB保持一致，基于MySQL 8.0.16或更高版本的NDB版本在没有指定CONSTRAINT符号子句或指定CONSTRAINT关键字时使用生成的约束名称。基于MySQL 5.7的NDB版本和早期的MySQL 8.0版本使用FOREIGN KEY index\_name值。

上述变化可能会给依赖以前的外键约束命名行为的应用程序带来不兼容的情况。

**改变的服务器默认值**

MySQL 8.0带有改进的默认值，目的是尽可能提供最好的开箱即用体验。这些变化是由以下事实驱动的：技术在进步（机器有更多的CPUS，使用SSD等等），更多的数据被存储，MySQL在发展（InnoDB、Group Replication、AdminAPI）等等。下表总结了已经改变的默认值，以便为大多数用户提供最佳的MySQL体验。

|  |  |  |
| --- | --- | --- |
| **Option/Parameter** | **Old Default** | **New Default** |
| *Server changes* |  |  |
| [**character\_set\_server**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\server-administration.html#sysvar_character_set_server) | latin1 | utf8mb4 |
| [**collation\_server**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\server-administration.html#sysvar_collation_server) | latin1\_swedish\_ci | utf8mb4\_0900\_ai\_ci |
| [**explicit\_defaults\_for\_timestamp**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\server-administration.html#sysvar_explicit_defaults_for_timestamp) | OFF | ON |
| [**optimizer\_trace\_max\_mem\_size**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\server-administration.html#sysvar_optimizer_trace_max_mem_size) | 16KB | 1MB |
| [**validate\_password\_check\_user\_name**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\security.html#sysvar_validate_password_check_user_name) | OFF | ON |
| [**back\_log**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\server-administration.html#sysvar_back_log) | -1 (autosize) changed from : back\_log = 50 + (max\_connections / 5) | -1 (autosize) changed to : back\_log = max\_connections |
| [**max\_allowed\_packet**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\server-administration.html#sysvar_max_allowed_packet) | 4194304 (4MB) | 67108864 (64MB) |
| [**max\_error\_count**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\server-administration.html#sysvar_max_error_count) | 64 | 1024 |
| [**event\_scheduler**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\server-administration.html#sysvar_event_scheduler) | OFF | ON |
| [**table\_open\_cache**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\server-administration.html#sysvar_table_open_cache) | 2000 | 4000 |
| [**log\_error\_verbosity**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\server-administration.html#sysvar_log_error_verbosity) | 3 (Notes) | 2 (Warning) |
| *InnoDB changes* |  |  |
| [**innodb\_undo\_tablespaces**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\innodb-storage-engine.html#sysvar_innodb_undo_tablespaces) | 0 | 2 |
| [**innodb\_undo\_log\_truncate**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\innodb-storage-engine.html#sysvar_innodb_undo_log_truncate) | OFF | ON |
| [**innodb\_flush\_method**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\innodb-storage-engine.html#sysvar_innodb_flush_method) | NULL | fsync (Unix), unbuffered (Windows) |
| [**innodb\_autoinc\_lock\_mode**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\innodb-storage-engine.html#sysvar_innodb_autoinc_lock_mode) | 1 (consecutive) | 2 (interleaved) |
| [**innodb\_flush\_neighbors**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\innodb-storage-engine.html#sysvar_innodb_flush_neighbors) | 1 (enable) | 0 (disable) |
| [**innodb\_max\_dirty\_pages\_pct\_lwm**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\innodb-storage-engine.html#sysvar_innodb_max_dirty_pages_pct_lwm) | 0 (%) | 10 (%) |
| [**innodb\_max\_dirty\_pages\_pct**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\innodb-storage-engine.html#sysvar_innodb_max_dirty_pages_pct) | 75 (%) | 90 (%) |
| *Performance Schema changes* |  |  |
| **performance-schema-instrument='wait/lock/metadata/sql/%=ON'** | OFF | ON |
| **performance-schema-instrument='memory/%=COUNTED'** | OFF | COUNTED |
| **performance-schema-consumer-events-transactions-current=ON** | OFF | ON |
| **performance-schema-consumer-events-transactions-history=ON** | OFF | ON |
| **performance-schema-instrument='transaction%=ON'** | OFF | ON |
| *Replication changes* |  |  |
| [**log\_bin**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\replication.html#sysvar_log_bin) | OFF | ON |
| [**server\_id**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\replication.html#sysvar_server_id) | 0 | 1 |
| [**log-slave-updates**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\replication.html#sysvar_log_slave_updates) | OFF | ON |
| [**expire\_logs\_days**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\replication.html#sysvar_expire_logs_days) | 0 | 30 |
| [**master-info-repository**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\replication.html#sysvar_master_info_repository) | FILE | TABLE |
| [**relay-log-info-repository**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\replication.html#sysvar_relay_log_info_repository) | FILE | TABLE |
| [**transaction-write-set-extraction**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\replication.html#sysvar_transaction_write_set_extraction) | OFF | XXHASH64 |
| [**slave\_rows\_search\_algorithms**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\replication.html#sysvar_slave_rows_search_algorithms) | INDEX\_SCAN, TABLE\_SCAN | INDEX\_SCAN, HASH\_SCAN |
| [**slave\_pending\_jobs\_size\_max**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\replication.html#sysvar_slave_pending_jobs_size_max) | 16M | 128M |
| [**gtid\_executed\_compression\_period**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\replication.html#sysvar_gtid_executed_compression_period) | 1000 | 0 |
| *Group Replication changes* |  |  |
| [**group\_replication\_autorejoin\_tries**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\group-replication.html#sysvar_group_replication_autorejoin_tries) | 0 | 3 |
| [**group\_replication\_exit\_state\_action**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\group-replication.html#sysvar_group_replication_exit_state_action) | ABORT\_SERVER | READ\_ONLY |
| [**group\_replication\_member\_expel\_timeout**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\group-replication.html#sysvar_group_replication_member_expel_timeout) | 0 | 5 |

关于已经添加的选项或变量的更多信息，请参见《MySQL服务器版本参考》中mysqld 8.0的选项/变量变化。

下面几节解释了对默认值的改变以及它们对你的部署可能产生的任何影响。

**服务器默认值**

* character\_set\_server系统变量和命令行选项 --character-set-server的默认值从latin1变为utf8mb4。这是服务器的默认字符集。此时，UTF8MB4是网络上的主流字符编码，这一变化使绝大多数MySQL用户的生活更容易。从5.7升级到8.0并不改变任何现有数据库对象的任何字符集。但是，除非你指定character\_set\_server回到你以前的默认值或明确设置字符集，那么新的模式、表或列默认使用utf8mb4。我们建议你尽可能地转移到utf8mb4。
* Collation\_server系统变量和命令行参数--collation-server的默认值从latin1\_swedish\_ci改为utf8mb4\_0900\_ai\_ci。这是服务器的默认校对，即字符集中字符的排序。整理和字符集之间存在着联系，因为每个字符集都有一个可能的整理列表。从5.7升级到8.0不会改变任何现有数据库对象的排序，但会对新对象生效。
* explicit\_defaults\_for\_timestamp系统变量的默认值从OFF（MySQL传统行为）变为ON（SQL标准行为）。这个选项最初在5.6中引入，在5.6和5.7中是OFF。
* optimizer\_trace\_max\_mem\_size系统变量的默认值从16KB变为1MB。旧的默认值导致优化器跟踪被截断，因为任何非琐碎的查询。这一变化确保了大多数查询的优化器跟踪是有用的。
* validate\_password\_check\_user\_name系统变量的默认值从OFF改为ON。这意味着，当validate\_password插件被启用时，默认情况下，它现在会拒绝与当前会话用户名匹配的密码。
* back\_log系统变量的autosize算法已经改变。 autosize（-1）的值现在被设置为max\_connections的值，它比50+（max\_connections / 5）的计算值大。在服务器无法跟上传入请求的情况下，back\_log会将传入的IP连接请求排队。在最坏的情况下，有了max\_connections数量的客户端试图同时重新连接，例如在网络故障后，它们都可以被缓冲，拒绝-重试的循环也就避免了。
* max\_allowed\_packet系统变量的默认值从4194304（4M）变为67108864（64M）。这个较大的默认值的主要优点是减少收到关于插入或查询大于max\_allowed\_packet的错误的机会。它应该和你想使用的最大的第11.3.4节 "BLOB和TEXT类型 "一样大。要恢复到以前的行为，设置max\_allowed\_packet=4194304。
* max\_error\_count系统变量的默认值从64变为1024。这确保了MySQL能够处理更多的警告，例如一个UPDATE语句触及了1000多条记录，其中很多都发出了转换警告。对于许多工具来说，批量更新是很常见的，以帮助减少复制滞后。外部工具如pt-online-schema-change默认为1000，而gh-ost默认为100。MySQL 8.0涵盖了这两种使用情况的全部错误历史。没有静态分配，所以这个变化只影响产生大量警告的语句的内存消耗。
* event\_scheduler系统变量的默认值从OFF变为ON。换句话说，事件调度器默认是启用的。这是SYS中新功能的一个启动器，例如 "杀死空闲事务"。
* table\_open\_cache系统变量的默认值从2000改为4000。这是一个微小的变化，增加了表访问的会话并发性。

log\_error\_verbosity系统变量的默认值从3（注意）改为2（警告）。其目的是使MySQL 8.0的错误日志在默认情况下不那么冗长。

**InnoDB默认值**

* 不兼容的变化 innodb\_undo\_tablespaces系统变量的默认值从0改为2。 该变量配置了InnoDB使用的undo tablespaces的数量。在MySQL 8.0中，innodb\_undo\_tablespaces的最小值是2，并且回滚段不能再系统表空间中创建。因此，这是一个不能恢复到5.7行为的情况。这一变化的目的是能够自动截断Undo日志（见下一项），回收（偶尔）长事务（如mysqldump）使用的磁盘空间。
* innodb\_undo\_log\_truncate系统变量的默认值从OFF变为ON。当启用时，超过innodb\_max\_undo\_log\_size定义的阈值的undo表空间被标记为截断。只有撤消的表空间可以被截断。不支持截断驻留在系统表空间中的Undo logs。从5.7升级到8.0会自动将你的系统转换为使用undo tablespaces，使用系统表空间在8.0中不是一个选项。
* innodb\_flush\_method系统变量的默认值在类Unix系统上从NULL变为fsync，在Windows系统上从NULL变为unbuffered。这更像是一个术语和选项的清理，没有任何实际的影响。对于Unix来说，这只是一个文档的变化，因为在5.7中默认也是fsync（默认NULL意味着fsync）。同样，在Windows上，innodb\_flush\_method默认值NULL在5.7中是指async\_unbuffered，在8.0中被默认unbuffered取代，这与现有的默认innodb\_use\_native\_aio=ON有相同的效果。
* 不兼容的变化 innodb\_autoinc\_lock\_mode系统变量的默认值从1（连续）变为2（交错）。将交错锁模式作为默认设置的变化反映了从基于语句的复制到基于行的复制作为默认复制类型的变化，这发生在MySQL 5.7。基于语句的复制需要连续的自动递增锁模式，以确保自动递增值以可预测和可重复的顺序分配给特定的SQL语句序列，而基于行的复制对SQL语句的执行顺序不敏感。因此，这个变化与基于语句的复制不兼容，可能会破坏一些依赖顺序自动递增的应用程序或用户生成的测试套件。以前的默认值可以通过设置innodb\_autoinc\_lock\_mode=1来恢复。
* innodb\_flush\_neighbors系统变量的默认值从1（启用）变为0（禁用）。这样做是因为快速IO（SSD）现在是部署的默认值。我们预计，对于大多数用户来说，这将导致一个小的性能提升。使用较慢硬盘的用户可能会看到性能损失，我们鼓励他们通过设置innodb\_flush\_neighbors=1恢复到以前的默认值。
* innodb\_max\_dirty\_pages\_pct\_lwm系统变量的默认值从0（%）变为10（%）。当innodb\_max\_dirty\_pages\_pct\_lwm=10时，InnoDB会在>10%的缓冲池包含修改过的（"脏"）页面时增加其刷新活动。这个变化的目的是为了稍微牺牲一下峰值吞吐量，以换取更稳定的性能。
* nnodb\_max\_dirty\_pages\_pct系统变量的默认值从75（%）变为90（%）。这一变化与innodb\_max\_dirty\_pages\_pct\_lwm的变化相结合，它们共同确保了InnoDB的平滑冲刷行为，避免了冲刷的突发性。要恢复到以前的行为，设置innodb\_max\_dirty\_pages\_pct=75和innodb\_max\_dirty\_pages\_pct\_lwm=0。

**性能模式的默认值**

* 性能模式的元数据锁定（MDL）工具默认是打开的。Performance-schema-instrument='wait/lock/metadata/sql/%=ON'的编译默认值从OFF变为ON。 这是在SYS中添加面向MDL的视图的一个促成因素。
* 性能模式内存仪表默认为打开。Performance-schema-instrument='memory/%=COUNTED'的编译默认值从OFF变为COUNTED。这一点很重要，因为如果在服务器启动后启用仪表，核算是不正确的，而且你可能会因为错过分配而得到一个负的余额，但抓住了一个空闲。
* 默认情况下，性能模式事务检测是打开的。Performance-schema-consumer-events-transactions-current=ON、performance-schema-consumer-events-transactions-history=ON和performance-schema-instrument='transaction%=ON'的编译默认值从OFF变为ON。

**复制默认值**

* log\_bin系统变量的默认值从OFF变为ON。换句话说，二进制日志是默认启用的。几乎所有的生产安装都启用了二进制日志，因为它被用于复制和时间点恢复。因此，通过默认启用二进制日志，我们消除了一个配置步骤，以后启用它需要重新启动mysqld。默认启用它也提供了更好的测试覆盖率，并且更容易发现性能回归。记得也要设置server\_id（见下面的修改）。8.0的默认行为就像你发出./mysqld --log-bin --server-id=1。如果你是在8.0版本，并且想要5.7版本的行为，你可以发布./mysqld --skip-log-bin --server-id=0。
* server\_id系统变量的默认值从0变为1（与log\_bin=ON的变化相结合）。服务器可以用这个默认ID启动，但实际上你必须根据正在部署的复制基础设施来设置服务器ID，以避免出现重复的服务器ID。
* log-slave-updates系统变量的默认值从OFF改为ON。 这将导致一个副本将复制的事件记录到它自己的二进制日志中。这个选项是组复制所需要的，同时也确保了在各种复制链设置中的正确行为，这已经成为今天的常态。
* expire\_logs\_days系统变量的默认值从0变为30。新的默认值30使mysqld定期清除超过30天的未使用的二进制日志。这一变化有助于防止在不再需要复制或恢复的二进制日志上浪费过多的磁盘空间。旧的值是0，禁用任何自动二进制日志清除。
* master\_info\_repository和relay\_log\_info\_repository系统变量的默认值从FILE变为TABLE。因此在8.0中，复制元数据默认存储在InnoDB中。这增加了可靠性，以尝试实现默认情况下的崩溃安全复制。
* 事务-写-集-提取系统变量的默认值从OFF变为XXHASH64。这个变化默认启用了事务写集。通过使用事务写集，源必须做稍多的工作来生成写集，但其结果对冲突检测有帮助。这是组复制的一个要求，新的默认值使得在源上启用二进制日志写集并行化很容易，以加快复制的速度。
* slave\_rows\_search\_algorithms系统变量的默认值从INDEX\_SCAN,TABLE\_SCAN改变为INDEX\_SCAN,HASH\_SCAN。这一变化通过减少复制应用者为将变化应用到没有主键的表而必须进行的表扫描的数量来加速基于行的复制。
* slave\_pending\_jobs\_size\_max系统变量的默认值从16M变为128M。这一变化增加了多线程复制的可用内存量。
* gtid\_executed\_compression\_period系统变量的默认值从1000改为0。这一变化确保mysql.gtid\_executed表的压缩只在需要时隐式发生。

**组复制默认值**

* group\_replication\_autorejoin\_tries的默认值从0变为3，这意味着默认情况下启用了自动重合。这个系统变量指定了成员在被驱逐或在达到group\_replication\_unreachable\_majority\_timeout设置之前无法与群组中的大多数成员联系时，自动重新加入群组的尝试次数。
* group\_replication\_exit\_state\_action的默认值从ABORT\_SERVER改为READ\_ONLY。这意味着当一个成员退出组时，例如在网络故障后，该实例会变成只读，而不是被关闭。
* group\_replication\_member\_expel\_timeout的默认值从0变成了5，这意味着被怀疑与群组失去联系的成员在5秒的检测时间后会被驱逐。

这些默认值中的大部分对于开发和生产环境都是合理的。但有一个例外，我们决定将名为innodb\_dedicated\_server的新选项设置为OFF，尽管我们建议在生产环境下将其打开。默认为OFF的原因是它会导致共享环境（如开发人员的笔记本电脑）变得无法使用，因为它占用了所有能找到的内存。

对于生产环境，我们建议将innodb\_dedicated\_server设置为ON。 当设置为ON时，以下InnoDB变量（如果没有明确指定的话）会根据可用内存进行自动缩放 innodb\_buffer\_pool\_size，innodb\_log\_file\_size和innodb\_flush\_method。见第15.8.12节，"为专门的MySQL服务器启用自动配置"。

虽然新的默认值是大多数使用情况下的最佳配置选择，但也有一些特殊情况，以及使用现有5.7配置选择的传统原因。例如，有些人喜欢升级到8.0，并尽可能少地改变他们的应用程序或操作环境。我们建议评估所有的新默认值并尽可能多地使用。大多数新的默认值可以在5.7中测试，所以你可以在升级到8.0之前在5.7生产中验证新的默认值。对于少数你需要旧的5.7值的默认值，在你的运行环境中设置相应的配置变量或启动选项。

在MySQL 8.0中，有一个性能模式variables\_info表，为每个系统变量显示它最近被设置的来源，以及它的数值范围。因此，在8.0中，你可以通过SQL访问关于配置变量及其值的所有信息。参见第27.12.14.2节，"性能模式变量信息表"。

### 2.11.5 为升级准备你的安装

在升级到最新的MySQL 8.0版本之前，通过执行下面描述的初步检查，确保你当前的MySQL 5.7或MySQL 8.0服务器实例的升级准备就绪。否则升级过程可能会失败。

提示

考虑使用MySQL Shell升级检查器工具，该工具使你能够验证MySQL服务器实例是否准备好升级。你可以选择一个你计划升级的目标MySQL Server版本，范围从MySQL Server 8.0.11到与当前MySQL Shell版本号匹配的MySQL Server版本号。升级检查器实用程序执行与指定的目标版本相关的自动检查，并向你建议你应该手动进行的进一步相关检查。升级检查器适用于MySQL 5.7和8.0的所有GA版本。MySQL Shell的安装说明可以在这里找到。

初步检查。

1、必须不存在以下问题。

* 必须没有使用过时的数据类型或函数的表。

如果表包含5.6.4之前格式的旧时间列（不支持小数秒精度的TIME、DATETIME和TIMESTAMP列），则不支持原地升级到MySQL 8.0。如果你的表仍然使用旧的时间列格式，在尝试就地升级到MySQL 8.0之前，使用REPAIR TABLE对它们进行升级。欲了解更多信息，请参阅《MySQL 5.7参考手册》中的服务器变更。

* 必须没有无主的.frm文件。
* 触发器不能有缺失或空的定义器或无效的创建环境（由SHOW TRIGGERS或INFORMATION\_SCHEMA TRIGGERS表显示的character\_set\_client、collation\_connection、Database Collation属性所表示）。任何这样的触发器必须被转储和恢复以解决这个问题。

要检查这些问题，执行这个命令。

mysqlcheck -u root -p --all-databases --check-upgrade

如果mysqlcheck报告任何错误，请纠正这些问题。

2、必须没有使用不具有本地分区支持的存储引擎的分区表。要识别这样的表，执行这个查询。

SELECT TABLE\_SCHEMA, TABLE\_NAME

FROM INFORMATION\_SCHEMA.TABLES

WHERE ENGINE NOT IN ('innodb', 'ndbcluster')

AND CREATE\_OPTIONS LIKE '%partitioned%';

任何由查询报告的表都必须被改变为使用InnoDB或者被变成非分区的。要把一个表的存储引擎改为InnoDB，执行这个语句。

ALTER TABLE ***table\_name*** ENGINE = INNODB;

关于将MyISAM表转换为InnoDB的信息，参见章节15.6.1.5，"将表从MyISAM转换为InnoDB"。

要使一个分区的表成为非分区的，执行这个语句。

ALTER TABLE ***table\_name*** REMOVE PARTITIONING;

3、在MySQL 8.0中可能保留了一些以前没有保留的关键字。见第9.3节，"关键词和保留字"。这可能导致以前用作标识符的词变得非法。要修复受影响的语句，请使用标识符引号。参见第9.2节，"模式对象名称"。

4、在MySQL 5.7 mysql系统数据库中必须没有与MySQL 8.0数据字典所使用的表具有相同名称的表。要识别具有这些名称的表，请执行此查询。

SELECT TABLE\_SCHEMA, TABLE\_NAME

FROM INFORMATION\_SCHEMA.TABLES

WHERE LOWER(TABLE\_SCHEMA) = 'mysql'

and LOWER(TABLE\_NAME) IN

(

'catalogs',

'character\_sets',

'check\_constraints',

'collations',

'column\_statistics',

'column\_type\_elements',

'columns',

'dd\_properties',

'events',

'foreign\_key\_column\_usage',

'foreign\_keys',

'index\_column\_usage',

'index\_partitions',

'index\_stats',

'indexes',

'parameter\_type\_elements',

'parameters',

'resource\_groups',

'routines',

'schemata',

'st\_spatial\_reference\_systems',

'table\_partition\_values',

'table\_partitions',

'table\_stats',

'tables',

'tablespace\_files',

'tablespaces',

'triggers',

'view\_routine\_usage',

'view\_table\_usage'

);

任何由查询报告的表必须被删除或重新命名（使用 RENAME TABLE）。这也可能需要对使用受影响表的应用程序进行修改。

5、不能有任何表的外键约束名称超过64个字符。使用这个查询来识别那些约束名称过长的表。

SELECT TABLE\_SCHEMA, TABLE\_NAME

FROM INFORMATION\_SCHEMA.TABLES

WHERE TABLE\_NAME IN

(SELECT LEFT(SUBSTR(ID,INSTR(ID,'/')+1),

INSTR(SUBSTR(ID,INSTR(ID,'/')+1),'\_ibfk\_')-1)

FROM INFORMATION\_SCHEMA.INNODB\_SYS\_FOREIGN

WHERE LENGTH(SUBSTR(ID,INSTR(ID,'/')+1))>64);

对于一个约束名称超过64个字符的表，删除该约束，然后用不超过64个字符的约束名称将其添加回来（使用ALTER TABLE）。

必须没有由sql\_mode系统变量定义的过时的SQL模式。试图使用6、一个过时的SQL模式会阻止MySQL 8.0的启动。应修改使用过时的SQL模式的应用程序以避免它们。有关在MySQL 8.0中删除的SQL模式的信息，请参阅服务器变更。

7、不得有明确定义的列名超过64个字符的视图（在MySQL 5.7中允许列名不超过255个字符的视图）。为避免升级错误，在升级前应改变此类视图。目前，识别列名超过64个字符的视图的唯一方法是使用SHOW CREATE VIEW检查视图定义。也可以通过查询 INFORMATION\_SCHEMA.VIEWS 表来检查视图定义。

8、不得有任何表或存储过程的单个ENUM或SET列元素的长度超过255个字符或1020字节。在MySQL 8.0之前，ENUM或SET列元素的最大组合长度是64K。在MySQL 8.0中，单个ENUM或SET列元素的最大字符长度为255个字符，最大字节长度为1020字节。1020字节的限制支持多字节的字符集）。在升级到MySQL 8.0之前，修改任何超过新限制的ENUM或SET列元素。如果不这样做，就会导致升级失败并出现错误。

9、在升级到MySQL 8.0.13或更高版本之前，必须没有驻留在共享InnoDB表空间中的表分区，其中包括系统表空间和一般表空间。通过查询INFORMATION\_SCHEMA来识别共享表空间中的表分区。

如果从MySQL 5.7升级，运行这个查询。

SELECT DISTINCT NAME, SPACE, SPACE\_TYPE FROM INFORMATION\_SCHEMA.INNODB\_SYS\_TABLES

WHERE NAME LIKE '%#P#%' AND SPACE\_TYPE NOT LIKE 'Single';

如果从早期的MySQL 8.0版本升级，请运行这个查询。

SELECT DISTINCT NAME, SPACE, SPACE\_TYPE FROM INFORMATION\_SCHEMA.INNODB\_TABLES

WHERE NAME LIKE '%#P#%' AND SPACE\_TYPE NOT LIKE 'Single';

使用ALTER TABLE将表的分区从共享表空间移到按表文件的表空间... 重新组织分区。

ALTER TABLE ***table\_name*** REORGANIZE PARTITION ***partition\_name***

INTO (***partition\_definition*** TABLESPACE=innodb\_file\_per\_table);

10必须没有来自MySQL 8.0.12或更低版本的查询和存储程序定义在GROUP BY子句中使用ASC或DISC限定词。否则，升级到MySQL 8.0.13或更高版本可能会失败，复制到MySQL 8.0.13或更高版本的复制服务器也可能失败。有关其他细节，请参阅SQL变更。

11、你的MySQL 5.7安装不得使用MySQL 8.0不支持的功能。这里的任何变化都必然是针对安装的，但下面的例子说明了要寻找的那种东西。

一些服务器启动选项和系统变量已在MySQL 8.0中被删除。参见《MySQL 8.0中删除的特性》和第1.4节 "MySQL 8.0中增加、废弃或删除的服务器和状态变量及选项"。如果你使用其中任何一个，升级需要改变配置。

例子。由于数据字典提供了关于数据库对象的信息，服务器不再检查数据目录中的目录名来寻找数据库。因此，--ignore-db-dir选项是多余的，已经被删除。为了处理这个问题，从你的启动配置中删除任何 --ignore-db-dir的实例。此外，在升级到MySQL 8.0之前，删除或移动命名的数据目录子目录。(或者，让8.0服务器将这些目录作为数据库添加到数据字典中，然后使用DROP DATABASE删除这些数据库中的每个。)

12、如果你打算在升级时将 lower\_case\_table\_names 设置改为 1，请在升级前确保模式和表名是小写的。否则，由于模式或表名字母大小写不匹配，可能会发生故障。你可以使用下面的查询来检查模式和表名是否包含大写字母。

mysql> SELECT TABLE\_NAME FROM INFORMATION\_SCHEMA.TABLES WHERE TABLE\_NAME != LOWER(TABLE\_NAME) AND TABLE\_TYPE = 'BASE TABLE';

mysql> SELECT SCHEMA\_NAME FROM INFORMATION\_SCHEMA.SCHEMATA WHERE SCHEMA\_NAME != LOWER(SCHEMA\_NAME);

从MySQL 8.0.19开始，如果lower\_case\_table\_names=1，升级过程将检查表和模式名称，以确保所有字符是小写的。如果发现表或模式名称包含大写字符，则升级过程会出错。

注意

不建议在升级时改变 lower\_case\_table\_names设置。

如果由于上述任何问题而导致升级到MySQL 8.0失败，服务器将恢复对数据目录的所有更改。在这种情况下，删除所有redo log文件，并在现有的数据目录上重启MySQL 5.7服务器以解决错误。redo log文件（ib\_logfile\*）默认驻留在MySQL数据目录中。在错误被修复后，在再次尝试升级前执行缓慢关机（通过设置innodb\_fast\_shutdown=0）。

### 2.11.6 在Unix/Linux上升级MySQL二进制或基于包的安装

本节描述了如何在Unix/Linux上升级MySQL二进制和基于软件包的安装。描述了原地和逻辑升级方法。

* 就地升级
* 逻辑升级
* MySQL群集升级

**就地升级**

原地升级包括关闭旧的MySQL服务器，用新的MySQL二进制文件或软件包替换旧的MySQL二进制文件或软件包，在现有的数据目录下重新启动MySQL，并升级现有安装中需要升级的任何剩余部分。有关可能需要升级的内容的详情，请参见第2.11.3节，"MySQL升级过程中升级的内容"。

注意

如果你正在升级一个最初通过安装多个RPM包产生的安装，请升级所有的包，而不仅仅是一些。例如，如果你以前安装了服务器和客户端RPM，不要只升级服务器RPM。

对于某些Linux平台，从RPM或Debian软件包中安装的MySQL包括对管理MySQL服务器启动和关闭的systemd支持。在这些平台上，mysqld\_safe没有被安装。在这种情况下，使用systemd来启动和关闭服务器，而不是下面说明中使用的方法。参见第2.5.9节，"用systemd管理MySQL服务器"。

关于MySQL簇安装的升级，另见MySQL簇升级。

要执行原地升级。

1. 查看第2.11.1节 "开始之前 "中的信息。
2. 通过完成第2.11.5节 "为升级准备你的安装 "中的初步检查，确保你的安装准备就绪。
3. 如果你使用InnoDB的XA事务，在升级前运行XA RECOVER来检查是否有未提交的XA事务。如果返回结果，通过发布XA COMMIT或XA ROLLBACK语句，提交或回滚XA事务。
4. 如果你从MySQL 5.7.11或更早的版本升级到MySQL 8.0，并且有加密的InnoDB表空间，通过执行此语句旋转钥匙圈主钥匙。

ALTER INSTANCE ROTATE INNODB MASTER KEY;

5、 如果你通常在运行你的MySQL服务器时将innodb\_fast\_shutdown设置为2（冷关机），那么通过执行这些语句中的任何一条将其配置为执行快速或缓慢关机。

SET GLOBAL innodb\_fast\_shutdown = 1; -- fast shutdown

SET GLOBAL innodb\_fast\_shutdown = 0; -- slow shutdown

通过快速或慢速关机，InnoDB将其Undo logs和数据文件保留在一个可以在不同版本的文件格式不同的情况下进行处理的状态。

6、关掉旧的MySQL服务器。比如说

mysqladmin -u root -p shutdown

1. 升级MySQL二进制文件或软件包。如果升级二进制安装，解压新的MySQL二进制分发包。参见获取和解压分发包。对于基于包的安装，安装新的包。
2. 启动MySQL 8.0服务器，使用现有的数据目录。例如。
3. mysqld\_safe --user=mysql --datadir=***/path/to/existing-datadir*** &

如果有加密的InnoDB表空间，使用--early-plugin-load选项来加载钥匙圈插件。

当你启动MySQL 8.0服务器时，它会自动检测是否存在数据字典表。如果没有，服务器会在数据目录中创建它们，用元数据填充它们，然后继续其正常的启动顺序。在这个过程中，服务器升级了所有数据库对象的元数据，包括数据库、表空间、系统和用户表、视图和存储程序（存储过程和函数、触发器和事件调度器事件）。服务器还删除了以前用于元数据存储的文件。例如，从MySQL 5.7升级到MySQL 8.0后，你可能会注意到，表不再有.frm文件。

如果这个步骤失败，服务器会恢复数据目录的所有变化。在这种情况下，你应该删除所有redo log文件，在同一数据目录上启动你的MySQL 5.7服务器，并修复任何错误的原因。然后对5.7服务器进行另一次缓慢关机，并启动MySQL 8.0服务器再次尝试。

9、在上一个步骤中，服务器根据需要升级了数据字典。现在有必要执行任何剩余的升级操作

* 从MySQL 8.0.16开始，服务器作为前一个步骤的一部分这样做，在MySQL 5.7和MySQL 8.0之间对mysql系统数据库进行任何必要的修改，以便你可以利用新的权限或能力。它还为MySQL 8.0带来性能模式、INFORMATION\_SCHEMA和sys数据库的更新，并检查所有用户数据库是否与当前版本的MySQL不兼容。
* 在MySQL 8.0.16之前，服务器只在前一步中升级数据字典。在MySQL 8.0服务器成功启动后，执行mysql\_upgrade以执行剩余的升级任务。
* mysql\_upgrade -u root -p

然后关闭并重启MySQL服务器，以确保对系统表所做的任何更改都会生效。例如

mysqladmin -u root -p shutdown

mysqld\_safe --user=mysql --datadir=***/path/to/existing-datadir*** &

第一次启动MySQL 8.0服务器时（在前面的步骤中），你可能会注意到错误日志中关于未升级的表的信息。如果mysql\_upgrade已经成功运行，在你第二次启动服务器时应该没有这样的信息。

注意

升级过程不升级时区表的内容。关于升级说明，见第5.1.15节，"MySQL服务器时区支持"。

如果升级过程使用mysql\_upgrade（也就是在MySQL 8.0.16之前），该过程也不会升级帮助表的内容。关于这种情况下的升级说明，见第5.1.17节，"服务器端帮助支持"。

**逻辑升级**

逻辑升级包括使用备份或导出工具（如mysqldump或mysqlpump）从旧的MySQL实例导出SQL，安装新的MySQL服务器，并将SQL应用到新的MySQL实例。关于可能需要升级的细节，见第2.11.3节，"MySQL升级过程中升级的内容"。

注意

对于某些Linux平台，从RPM或Debian软件包中安装的MySQL包括用于管理MySQL服务器启动和关闭的systemd支持。在这些平台上，mysqld\_safe没有被安装。在这种情况下，使用systemd来启动和关闭服务器，而不是下面说明中使用的方法。参见第2.5.9节，"用systemd管理MySQL服务器"。

警告

将从以前的MySQL版本中提取的SQL应用到新的MySQL版本中，可能会因为新的、改变的、废弃的或移除的特性和功能所带来的不兼容而导致错误。因此，从以前的MySQL版本中提取的SQL可能需要修改以实现逻辑升级。

要在升级到最新的MySQL 8.0版本之前确定不兼容的地方，请执行第2.11.5节 "为升级准备你的安装 "中描述的步骤。

要执行逻辑升级。

1. 查看第2.11.1节 "开始之前 "中的信息。
2. 从以前的MySQL安装中导出你的现有数据。

mysqldump -u root -p

--add-drop-table --routines --events

--all-databases --force > data-for-upgrade.sql

注意

如果你的数据库包括存储程序，请在mysqldump中使用--routines和--events选项（如上所示）。--all-databases选项包括转储中的所有数据库，包括保存系统表的mysql数据库。

重要的是

如果你有包含生成列的表，使用MySQL 5.7.9或更高版本提供的mysqldump工具来创建你的转储文件。早期版本提供的mysqldump工具对生成的列定义使用了不正确的语法（Bug #20769542）。你可以使用INFORMATION\_SCHEMA.COLUMNS表来识别具有生成列的表。

3、 关闭旧的MySQL服务器。例如

mysqladmin -u root -p shutdown

4、安装MySQL 8.0。关于安装说明，见第2章，安装和升级MySQL。

1. 初始化一个新的数据目录，如第2.10.1节 "初始化数据目录 "所述。例如

mysqld --initialize --datadir=***/path/to/8.0-datadir***

复制显示在屏幕上的临时'root'@'localhost'密码或写入错误日志，以便以后使用。

6、启动MySQL 8.0服务器，使用新的数据目录。例如

mysqld\_safe --user=mysql --datadir=***/path/to/8.0-datadir*** &

7、重置根密码。

shell> **mysql -u root -p**

Enter password: **\*\*\*\*** <- enter temporary root password

mysql> **ALTER USER USER() IDENTIFIED BY '*your new password*';**

8、将先前创建的转储文件加载到新的MySQL服务器。例如

mysql -u root -p --force < data-for-upgrade.sql

注意

如果你的转储文件包括系统表，不建议在服务器上启用GTIDs（gtid\_mode=ON）时加载转储文件。mysqldump为使用非事务型MyISAM存储引擎的系统表发出DML指令，当GTIDs被启用时，这种组合是不允许的。还要注意的是，从一个启用了GTIDs的服务器加载转储文件到另一个启用了GTIDs的服务器，会导致产生不同的事务标识符。

9、执行任何剩余的升级

* 在MySQL 8.0.16和更高版本中，关闭服务器，然后用-upgrade=FORCE选项重新启动它，以执行剩余的升级任务。

mysqladmin -u root -p shutdown

mysqld\_safe --user=mysql --datadir=***/path/to/8.0-datadir*** --upgrade=FORCE &

在用-upgrade=FORCE重新启动时，服务器会在MySQL 5.7和MySQL 8.0之间对mysql系统模式进行任何必要的更改，以便你可以利用新的权限或能力。它还为MySQL 8.0带来性能模式、INFORMATION\_SCHEMA和sys模式的更新，并检查所有用户模式是否与当前版本的MySQL不兼容。

* 在MySQL 8.0.16之前，执行mysql\_upgrade来执行剩余的升级任务。
* mysql\_upgrade -u root -p

然后关闭并重新启动MySQL服务器，以确保对系统表所做的任何更改都会生效。例如

mysqladmin -u root -p shutdown

mysqld\_safe --user=mysql --datadir=***/path/to/8.0-datadir*** &

注意

升级过程不升级时区表的内容。关于升级说明，见第5.1.15节 "MySQL服务器时区支持"。

如果升级过程使用mysql\_upgrade（也就是在MySQL 8.0.16之前），该过程也不会升级帮助表的内容。关于这种情况下的升级说明，见第5.1.17节 "服务器端帮助支持"。

注意

加载包含MySQL 5.7 mysql模式的转储文件会重新创建两个不再使用的表：event和proc。相应的MySQL 8.0表是events和routine，这两个表都是数据字典表，是受保护的）。在你确信升级成功后，你可以通过执行这些SQL语句来删除事件和程序表。

DROP TABLE mysql.event;

DROP TABLE mysql.proc;

**MySQL群集升级**

本节中的信息是对就地升级中描述的就地升级程序的辅助，供你在升级MySQL簇时使用。

从MySQL 8.0.16开始，MySQL簇的升级可以作为常规的滚动升级来执行，遵循通常的三个有序步骤。

1. 升级MGM节点。
2. 一次性升级数据节点。
3. 一次一次地升级API节点（包括MySQL服务器）。

升级每个节点的方式与MySQL 8.0.16之前几乎一样，因为升级数据字典和升级系统表之间是分开的。升级每个单独的mysqld有两个步骤。

1、导入数据字典。

用-upgrade=MINIMAL选项启动新服务器，升级数据字典，但不升级系统表。这与MySQL 8.0.16之前启动服务器而不调用mysql\_upgrade的操作基本相同。

为了完成这个阶段，MySQL服务器必须连接到NDB。如果有任何NDB或NDBINFO表存在，而服务器不能连接到集群，它将以错误信息退出。

Failed to Populate DD tables.

2、升级系统表。

在MySQL 8.0.16之前，DBA调用mysql\_upgrade客户端来升级系统表。从MySQL 8.0.16开始，服务器会执行这一操作。要升级系统表，重启每个单独的mysqld，不要使用--upgrade=MINIMAL选项。

### 2.11.7 使用MySQL Yum仓库升级MySQL

对于受支持的基于Yum的平台（参见第2.5.1节，"使用MySQL Yum仓库在Linux上安装MySQL"，获取列表），你可以使用MySQL Yum仓库对MySQL进行就地升级（即替换旧版本，然后使用旧数据文件运行新版本）。

注意事项

* 在对MySQL进行任何更新之前，请仔细遵循第2.11节 "升级MySQL "中的说明。在那里讨论的其他指示中，在更新前备份你的数据库是特别重要的。
* 下面的说明假定你已经用MySQL Yum库或直接从MySQL开发区的MySQL下载页面下载的RPM包安装了MySQL；如果不是这种情况，请按照《使用MySQL Yum库替换第三方分发的MySQL》中的说明进行。

1、选择一个目标系列

默认情况下，MySQL Yum资源库会将MySQL更新到你在安装时选择的发行系列中的最新版本（详见选择发行系列），这意味着，例如，5.7.x的安装不会自动更新到8.0.x版本。要更新到另一个发行系列，你必须首先禁用已选择的系列的子库（默认的，或自己选择的），并启用目标系列的子库。要做到这一点，请参阅选择发布系列中给出的一般说明。对于从MySQL 5.7升级到8.0，执行选择发布系列中说明的相反步骤，禁用MySQL 5.7系列的子库，为MySQL 8.0系列启用子库。

作为一般规则，要从一个发布系列升级到另一个发布系列，应转到下一个系列，而不是跳过一个系列。

重要信息

关于从MySQL 5.7升级到8.0的重要信息，请参阅从MySQL 5.7升级到8.0。

2、升级MySQL

对于未启用dnf的平台，可通过以下命令升级MySQL及其组件。

sudo yum update mysql-server

对于已启用dnf的平台。

sudo dnf upgrade mysql-server

或者，你可以通过告诉Yum更新你系统上的所有东西来更新MySQL，这可能需要相当多的时间。对于未启用dnf的平台。

sudo yum update

对于已启用dnf的平台。

sudo dnf upgrade

3、重启MySQL

在Yum更新后，MySQL服务器总是重新启动。在MySQL 8.0.16之前，在服务器重启后运行mysql\_upgrade，以检查并可能解决旧数据和升级后的软件之间的任何不兼容问题。mysql\_upgrade还执行其他功能；详情见4.4.5节，"mysql\_upgrade--检查并升级MySQL表"。从MySQL 8.0.16开始，这个步骤就不需要了，因为服务器会执行所有以前由mysql\_upgrade处理的任务。

你也可以只更新一个特定的组件。使用下面的命令来列出所有已安装的MySQL组件的包（对于支持dnf的系统，在命令中用dnf替换yum）。

sudo yum list installed | grep "^mysql"

在确定了你所选择的组件的软件包名称后，用下面的命令更新软件包，用软件包的名称代替package-name。对于未启用dnf的平台。

sudo yum update ***package-name***

对于启用了dnf的平台。

sudo dnf upgrade ***package-name***

**升级共享的客户端库**

在使用Yum资源库更新MySQL后，用旧版本的共享客户端库编译的应用程序应该继续工作。

如果你重新编译应用程序并将它们与更新的库动态链接。作为新版本共享库的典型情况，在较新和较旧的库之间存在符号版本的差异或增加（例如，较新的、标准的8.0共享客户端库和一些较旧的、由Linux发行版软件库原生提供的共享库或变种版本之间，或从一些其他来源），任何使用更新的、较新的共享库编译的应用程序在部署应用程序的系统上需要这些更新的库。正如预期的那样，如果这些库没有到位，需要共享库的应用程序就会失败。出于这个原因，请确保在这些系统上部署来自MySQL的共享库包。要做到这一点，请将MySQL Yum仓库添加到这些系统中（参见添加MySQL Yum仓库），并使用 "用Yum安装其他MySQL产品和组件 "中给出的说明安装最新的共享库。

### 2.11.8 用MySQL APT资源库升级MySQL

在Debian和Ubuntu平台上，要执行MySQL及其组件的原地升级，请使用MySQL APT资源库。见《使用MySQL APT资源库快速指南》中的用MySQL APT资源库升级MySQL。

### 2.11.9 使用MySQL SLES资源库升级MySQL

在SUSE Linux Enterprise Server（SLES）平台上，要对MySQL及其组件进行原地升级，请使用MySQL SLES资源库。请参阅《使用MySQL SLES资源库快速指南》中用MySQL SLES资源库升级MySQL。

### 2.11.10 在Windows上升级MySQL

有两种方法可以在Windows上升级MySQL。

* 使用MySQL安装程序
* 使用Windows的ZIP档案分发

你选择的方法取决于现有安装是如何进行的。在继续进行之前，请查看第2.11节 "升级MySQL"，以了解有关升级MySQL的其他信息，这些信息不是针对Windows的。

注意

无论你选择哪种方法，在执行升级之前一定要备份你当前的MySQL安装。参见第7.2节，"数据库备份方法"。

不支持非GA版本之间的升级（或从非GA版本到GA版本）。在非GA版本中会发生重大的开发变化，你可能会遇到兼容性问题或启动服务器的问题。

注意

MySQL安装程序不支持社区版和商业版之间的升级。如果你需要这种类型的升级，请使用ZIP存档方法执行。

**用MySQL安装器升级MySQL**

如果当前的服务器安装是用MySQL Installer进行的，并且升级是在当前的版本系列内，那么用MySQL Installer执行升级是最好的方法。MySQL安装程序不支持版本系列之间的升级，例如从5.7到8.0，而且它不提供升级指示器来提示你进行升级。有关在发布系列之间升级的说明，请参见使用Windows ZIP发行版升级MySQL。

要使用MySQL Installer执行升级**。**

1. 启动MySQL安装程序。
2. 从仪表板上，点击目录，下载目录的最新变化。只有当仪表板上显示服务器的版本号旁边有一个箭头时，已安装的服务器才能被升级。
3. 点击升级。所有具有较新版本的产品现在都出现在一个列表中。

注意

MySQL安装程序对于同一发布系列中的里程碑版本（Pre-Release）取消选择服务器升级选项。此外，它还显示一个警告，表明不支持升级，确定继续下去的风险，并提供一个手动执行升级的步骤摘要。你可以重新选择服务器升级，并在你自己的风险下进行。

4、除MySQL服务器产品外，取消选择所有产品，除非你打算在这个时候升级其他产品，然后点击下一步。

5、单击 "执行 "以开始下载。当下载完成后，点击下一步开始升级操作。

升级到MySQL 8.0.16及更高版本可能会显示一个选项，以跳过系统表的升级检查和过程。关于这个选项的更多信息，请参见重要的服务器升级条件。

6、配置服务器。

**使用Windows ZIP发行版升级MySQL**

1、要使用Windows ZIP档案分发版执行升级。

从https://dev.mysql.com/downloads/，下载最新的MySQL的Windows ZIP存档发行版。

2、如果服务器正在运行，停止它。如果服务器是作为服务安装的，从命令提示符中用以下命令停止服务。

C:\> **SC STOP *mysqld\_service\_name***

或者，使用NET STOP mysqld\_service\_name 。

如果你没有将MySQL服务器作为服务运行，使用mysqladmin来停止它。例如，在从MySQL 5.7升级到8.0之前，从MySQL 5.7使用mysqladmin，如下所示。

C:\> **"C:\Program Files\MySQL\MySQL Server 5.7\bin\mysqladmin" -u root shutdown**

注意

如果MySQL根用户账户有密码，用-p选项调用mysqladmin，并在提示时输入密码。

3、提取ZIP档案。你可以覆盖你现有的MySQL安装（通常位于C:\mysql），或将其安装到不同的目录，如C:\mysql8。建议覆盖现有的安装。

4、重新启动服务器。例如，如果你将MySQL作为服务运行，使用SC START mysqld\_service\_name或NET START mysqld\_service\_name命令，否则直接调用mysqld。

5、在MySQL 8.0.16之前，以管理员身份运行mysql\_upgrade来检查你的表，如果有必要，尝试修复它们，如果你的授予表已经改变，则更新它们，以便你可以利用任何新的功能。见第4.4.5节，"mysql\_upgrade--检查和升级MySQL表"。从MySQL 8.0.16开始，这个步骤就不需要了，因为服务器会执行所有以前由mysql\_upgrade处理的任务。

6、如果你遇到错误，请参阅第2.3.5节，"Microsoft Windows MySQL服务器安装的故障排除"。

### 2.11.11 升级MySQL的Docker安装

要升级MySQL的Docker安装，请参阅升级MySQL服务器容器。

### 2.11.12 升级故障排除

* 在MySQL 5.7实例中，表的.frm文件和InnoDB数据字典之间的模式不匹配可能导致升级到MySQL 8.0失败。这种不匹配可能是由于.frm文件损坏。为了解决这个问题，在再次尝试升级之前转储和恢复受影响的表。
* 如果出现问题，例如新的mysqld服务器不启动，请确认你没有以前安装时的旧的my.cnf文件。你可以用 --print-defaults 选项来检查（例如，mysqld --print-defaults）。如果该命令显示除程序名以外的任何内容，你有一个活动的my.cnf文件，会影响服务器或客户端的操作。
* 如果在升级后，你在编译客户程序时遇到问题，如命令不同步或意外的核心转储，你可能在编译程序时使用了旧的头文件或库文件。在这种情况下，检查你的mysql.h文件和libmysqlclient.a库的日期，以验证它们是否来自新的MySQL发行版。如果不是，用新的头文件和库重新编译你的程序。如果库的主要版本号改变了（例如，从libmysqlclient.so.20到libmysqlclient.so.21），针对共享客户端库编译的程序可能也需要重新编译。
* 如果你用一个给定的名称创建了一个用户定义的函数（UDF），并将MySQL升级到一个实现了相同名称的新内置函数的版本，该UDF会变得无法访问。要纠正这一点，使用DROP FUNCTION删除UDF，然后使用CREATE FUNCTION用不同的不冲突的名字重新创建UDF。如果MySQL的新版本实现了一个与现有存储函数同名的内置函数，情况也是如此。参见第9.2.5节 "函数名称解析和解决"，以了解描述服务器如何解释对不同种类的函数的引用的规则。
* 如果由于第2.11.5节 "为升级准备你的安装 "中概述的任何问题而导致升级到MySQL 8.0失败，服务器将恢复对数据目录的所有更改。在这种情况下，删除所有redo log文件，并在现有的数据目录上重启MySQL 5.7服务器，以解决这些错误。redo log文件（ib\_logfile\*）默认驻留在MySQL数据目录中。在错误被修复后，在再次尝试升级前执行缓慢关机（通过设置innodb\_fast\_shutdown=0）。

### 2.11.13 重建或修复表或索引

本节描述了如何重建或修复表或索引，这可能是必要的。

* 对MySQL处理数据类型或字符集方式的改变。例如，整理的错误可能已被纠正，需要重建表以更新使用该整理的字符列的索引。
* 需要通过CHECK TABLE、mysqlcheck或mysql\_upgrade报告的表修复或升级。

重建表的方法包括

* Dump and Reload方法
* ALTER TABLE方法
* REPAIR TABLE方法

**转储和重新加载方法**

如果你重建表是因为不同版本的MySQL在二进制（原地）升级或降级后不能处理它们，你必须使用转储和重载方法。在升级或降级之前，使用你的原始版本的MySQL转储表。然后在升级或降级后重新加载这些表。

如果你使用dump-and-reload方法，重建表只是为了重建索引，你可以在升级或降级之前或之后执行dump。重新加载仍然必须在之后进行。

如果你需要重建一个InnoDB表，因为CHECK TABLE操作表明需要升级表，使用mysqldump创建一个转储文件，并使用mysql重新加载该文件。如果CHECK TABLE操作指示有损坏或导致InnoDB失败，请参阅15.21.2节 "强制InnoDB恢复"，了解使用innodb\_force\_recovery选项重启InnoDB的信息。要了解CHECK TABLE可能遇到的问题类型，请参考章节13.7.3.2 "CHECK TABLE语句 "中的InnoDB说明。

要通过转储和重载重建一个表，使用mysqldump创建一个转储文件和mysql重载该文件。

mysqldump ***db\_name*** t1 > dump.sql

mysql ***db\_name*** < dump.sql

要重建单个数据库中的所有表，请指定数据库名称，不要有任何后面的表名。

mysqldump ***db\_name*** > dump.sql

mysql ***db\_name*** < dump.sql

要重建所有数据库中的所有表，使用--all-databases选项。

mysqldump --all-databases > dump.sql

mysql < dump.sql

**ALTER TABLE方法**

要用ALTER TABLE重建表，使用 "空 "的改变；也就是说，一个ALTER TABLE语句 "改变 "表，使其使用已经拥有的存储引擎。例如，如果t1是一个InnoDB表，使用这个语句。

ALTER TABLE t1 ENGINE = InnoDB;

如果你不确定在ALTER TABLE语句中指定哪个存储引擎，使用SHOW CREATE TABLE来显示表的定义。

REPAIR TABLE方法

REPAIR TABLE方法只适用于MyISAM, ARCHIVE, 和CSV表。

如果表检查操作表明存在损坏或需要升级，可以使用 REPAIR TABLE。例如，要修复一个MyISAM表，使用这个语句。

REPAIR TABLE t1;

mysqlcheck --repair提供了对REPAIR TABLE语句的命令行访问。这可能是一种更方便的修复表的手段，因为你可以使用--数据库或--所有数据库选项，分别修复特定数据库或所有数据库中的所有表。

mysqlcheck --repair --databases ***db\_name*** ...

mysqlcheck --repair --all-databases

### 2.11.14 将MySQL数据库复制到另一台机器上

在需要在不同架构之间转移数据库的情况下，你可以使用mysqldump来创建一个包含SQL语句的文件。然后，你可以将该文件传输到另一台机器上，并将其作为输入提供给mysql客户端。

使用mysqldump --help查看有哪些选项可用。

注意

如果在你创建转储的服务器上正在使用GTIDs(gtid\_mode=ON)，默认情况下，mysqldump在转储中包括gtid\_executed集的内容，以将这些内容转移到新机器上。其结果可能因涉及的MySQL Server版本而不同。查看mysqldump的-set-gtid-purged选项的描述，了解你使用的版本会发生什么，以及如果默认行为的结果不适合你的情况，如何改变行为。

在两台机器之间移动数据库的最简单（尽管不是最快）的方法是在数据库所在的机器上运行以下命令。

mysqladmin -h '***other\_hostname***' create ***db\_name***

mysqldump ***db\_name*** | mysql -h '***other\_hostname***' ***db\_name***

如果想通过慢速网络从远程机器复制数据库，可以使用这些命令。

如果你想通过慢速网络从远程机器复制一个数据库，你可以使用这些命令。

mysqladmin create ***db\_name***

mysqldump -h '***other\_hostname***' --compress ***db\_name*** | mysql ***db\_name***

你也可以将转储存储在一个文件中，将文件传输到目标机，然后将文件加载到那里的数据库中。例如，你可以像这样在源机器上把数据库转储到一个压缩文件中。

mysqldump --quick ***db\_name*** | gzip > ***db\_name***.gz

将包含数据库内容的文件传输到目标机上，并在那里运行这些命令。

mysqladmin create ***db\_name***

gunzip < ***db\_name***.gz | mysql ***db\_name***

你也可以使用mysqldump和mysqlimport来传输数据库。对于大表，这比简单地使用mysqldump快得多。在下面的命令中，DUMPDIR代表你用来存储mysqldump输出的目录的完整路径名称。

首先，为输出文件创建目录并转储数据库。

mkdir ***DUMPDIR***

mysqldump --tab=***DUMPDIR***

***db\_name***

然后将DUMPDIR目录中的文件转移到目标机上的一些相应目录中，并将文件加载到那里的MySQL中。

mysqladmin create ***db\_name*** # create database

cat ***DUMPDIR***/\*.sql | mysql ***db\_name*** # create tables in database

mysqlimport ***db\_name***

***DUMPDIR***/\*.txt # load data into tables

不要忘记复制mysql数据库，因为那是授予表的存储地。你可能不得不在新机器上以MySQL根用户的身份运行命令，直到你有了mysql数据库。

在新机器上导入mysql数据库后，执行mysqladmin flush-privileges，这样服务器就会重新加载授予表的信息。

## 2.12 降级MySQL

不支持从MySQL 8.0降级到MySQL 5.7，或从MySQL 8.0版本降级到以前的MySQL 8.0版本。唯一支持的替代方法是恢复升级前的备份。因此，在开始升级过程之前，你必须备份你的数据。

## 2.13 Perl安装注意事项

Perl DBI模块为数据库访问提供了一个通用接口。你可以编写一个DBI脚本，在不改变的情况下与许多不同的数据库引擎一起工作。要使用DBI，你必须安装DBI模块，以及为你要访问的每种类型的数据库服务器安装一个数据库驱动（DBD）模块。对于MySQL，这个驱动是DBD::mysql模块。

注意

*Perl支持不包括在MySQL发行版中。你可以从Unix的http://search.cpan.org，或通过在Windows上使用ActiveState ppm程序获得必要的模块。下面几节描述了如何做到这一点。*

DBI/DBD接口需要Perl 5.6.0，最好是5.6.1或更高版本。如果你有一个旧版本的Perl，DBI就不能工作。你应该使用DBD::mysql 4.009或更高版本。尽管有早期的版本，但它们不支持MySQL 8.0的全部功能。

### 2.13.1 在Unix上安装Perl

对MySQL Perl的支持要求你已经安装了MySQL客户端编程支持（库和头文件）。大多数安装方法都会安装必要的文件。如果你在Linux上从RPM文件中安装MySQL，一定要同时安装开发者RPM。客户端程序在客户端RPM中，但客户端编程支持在开发者RPM中。

你需要的Perl支持文件可以从CPAN（综合Perl档案网络）获得，网址是http://search.cpan.org。

在Unix上安装Perl模块的最简单方法是使用CPAN模块。比如说

shell> **perl -MCPAN -e shell**

cpan> **install DBI**

cpan> **install DBD::mysql**

DBD::mysql的安装运行了一些测试。这些测试试图使用默认的用户名和密码连接到本地 MySQL 服务器。(默认的用户名是你在Unix上的登录名，在Windows上的ODBC。默认密码是 "无密码"）。) 如果你不能用这些值连接到服务器（例如，如果你的账户有一个密码），测试就会失败。你可以使用强制安装DBD::mysql来忽略失败的测试。

DBI需要Data::Dumper模块。它可能已经安装了；如果没有，你应该在安装DBI之前安装它。

也可以以压缩的tar文件的形式下载模块发行版，并手动构建模块。例如，要解压并构建一个DBI发行版，可以使用这样的程序。

1、解压发行版到当前目录中。

shell> **gunzip < DBI-*VERSION*.tar.gz | tar xvf -**

这个命令创建了一个名为DBI-VERSION的目录。

2、改变位置到解压后的发行版的顶层目录中。

shell> **cd DBI-*VERSION***

3、建立发行版并编译所有内容。

shell> **perl Makefile.PL**

shell> **make**

shell> **make test**

shell> **make install**

make test 命令很重要，因为它验证了模块是否工作。注意，当你在DBD::mysql安装期间运行该命令以锻炼接口代码时，MySQL服务器必须正在运行，否则测试失败。

每当你安装一个新的MySQL版本时，重建并重新安装DBD::mysql发行版是个好主意。这可以确保正确安装最新版本的MySQL客户端库。

如果你没有在系统目录下安装Perl模块的访问权限，或者你想安装本地Perl模块，下面的参考资料可能很有用：http://learn.perl.org/faq/perlfaq8.html#How-do-I-keep-my-own-module-library-directory-

### 2.13.2 在Windows上安装ActiveState Perl

在Windows上，你应该做以下事情来安装带有ActiveState Perl的MySQL DBD模块。

1、从http://www.activestate.com/Products/ActivePerl/ 获取ActiveState Perl并安装它。

2、打开一个控制台窗口。

3、如果有必要，设置HTTP\_proxy变量。例如，你可以尝试这样的设置。

:\> **set HTTP\_proxy=my.proxy.com:3128**

4、启动PPM程序。

C:\> **C:\perl\bin\ppm.pl**

5、如果你以前没有这样做，请安装DBI。

ppm> **install DBI**

6、如果这成功了，运行下面的命令。

ppm> **install DBD-mysql**

这个程序应该在ActiveState Perl 5.6或更高版本中工作。

如果你不能使该程序工作，你应该安装ODBC驱动程序，而不是通过ODBC连接到MySQL服务器。

use DBI;

$dbh= DBI->connect("DBI:ODBC:$dsn",$user,$password) ||

die "Got error $DBI::errstr when connecting to $dsn\n";

### 2.13.3 使用Perl DBI/DBD接口的问题

如果Perl报告说它找不到./mysql/mysql.so模块，那么问题可能是Perl无法找到libmysqlclient.so共享库。你应该可以通过以下方法解决这个问题。

* 将 libmysqlclient.so 复制到其他共享库所在的目录中（可能是 /usr/lib 或 /lib）。
* 修改用于编译DBD::mysql的-L选项，以反映libmysqlclient.so的实际位置。
* 在Linux上，你可以在/etc/ld.so.conf文件中添加libmysqlclient.so所在目录的路径名称。
* 在LD\_RUN\_PATH环境变量中加入libmysqlclient.so所在目录的路径名。有些系统使用LD\_LIBRARY\_PATH代替。

注意，如果有其他链接器找不到的库，你可能还需要修改-L选项。例如，如果链接器找不到 libc，因为它在 /lib 中，而链接命令指定的是 -L/usr/lib，请将 -L 选项改为 -L/lib 或在现有的链接命令中添加 -L/lib。

如果你从DBD::mysql得到以下错误，你可能使用的是gcc（或使用用gcc编译的旧二进制文件）。

/usr/bin/perl: can't resolve symbol '\_\_moddi3'

/usr/bin/perl: can't resolve symbol '\_\_divdi3'

在链接命令中加入 -L/usr/lib/gcc-lib/... -lgcc到链接命令中，当mysql.so库被构建时（当你编译Perl客户端时检查mysql.so的make输出）。-L选项应该指定libgcc.a在你的系统中所在的目录的路径名称。

这个问题的另一个原因可能是Perl和MySQL不是同时用gcc编译的。在这种情况下，你可以通过用gcc编译两者来解决不匹配的问题。

# 指南

本章通过展示如何使用mysql客户端程序来创建和使用一个简单的数据库，提供了对MySQL的教程性介绍。mysql（有时被称为 "终端监视器 "或只是 "监视器"）是一个交互式程序，使你能够连接到MySQL服务器，运行查询，并查看结果。mysql也可以在批处理模式下使用：你事先将查询放在一个文件中，然后告诉mysql执行该文件的内容。这里涵盖了使用mysql的两种方式。

要查看mysql提供的选项列表，可以用--help选项来调用它。

shell> **mysql --help**

本章假设mysql已经安装在你的机器上，并且有一个你可以连接的MySQL服务器。如果这不是真的，请联系你的MySQL管理员。(如果你是管理员，你需要查阅本手册的相关部分，如第5章，MySQL服务器管理。)

本章描述了设置和使用数据库的整个过程。如果你只对访问一个现有的数据库感兴趣，你可能想跳过描述如何创建数据库和它所包含的表的章节。

由于本章属于教程性质，许多细节必然被省略。请查阅手册中的相关章节，以了解这里所涉及的主题的更多信息。

## 3.1 与服务器的连接和断开连接

要连接到服务器，你通常需要在调用mysql时提供一个MySQL用户名，而且很可能还有一个密码。如果服务器在你登录的机器以外的机器上运行，你还必须指定一个主机名。联系你的管理员，了解你应该使用什么连接参数来连接（也就是说，使用什么主机、用户名和密码）。一旦你知道了正确的参数，你应该能够像这样连接。

shell> **mysql -h *host* -u *user* -p**

Enter password: **\*\*\*\*\*\*\*\***

host和user代表你的MySQL服务器运行的主机名和你的MySQL账户的用户名。根据你的设置，用适当的值代替。\*\*\*\*\*\*\*\* 代表你的密码；当mysql显示输入密码：提示时，输入它。

如果成功，你应该看到一些介绍性的信息，然后是mysql>提示。

shell> **mysql -h *host* -u *user* -p**

Enter password: **\*\*\*\*\*\*\*\***

Welcome to the MySQL monitor. Commands end with ; or \g.

Your MySQL connection id is 25338 to server version: 8.0.26-standard

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql>

mysql>提示告诉你，mysql已经准备好让你输入SQL语句。

如果你在MySQL运行的同一台机器上登录，你可以省略主机，并简单地使用以下方法。

shell> **mysql -u *user* -p**

如果，当你试图登录时，你得到一个错误信息，如 ERROR 2002 (HY000)。不能通过套接字'/tmp/mysql.sock'（2）连接到本地MySQL服务器，这意味着MySQL服务器守护程序（Unix）或服务（Windows）没有运行。请咨询管理员或参阅第2章 "安装和升级MySQL "中适合你的操作系统的部分。

关于尝试登录时经常遇到的其他问题的帮助，见B.3.2节，"使用MySQL程序时的常见错误"。

一些MySQL安装允许用户以匿名（未命名）用户的身份连接到在本地主机上运行的服务器。如果你的机器是这种情况，你应该能够通过调用mysql而不使用任何选项来连接到该服务器。

shell> **mysql**

在你成功连接后，你可以在任何时候通过在mysql> prompt下输入QUIT（或\q）断开连接。

mysql> **QUIT**

Bye

在Unix中，你也可以通过按Control+D断开连接。

下面几节中的大多数例子都假定你已经连接到服务器。它们通过mysql> prompt表明这一点。

## 3.2 输入查询

确保你已经连接到了服务器，如上一节所述。这样做本身并没有选择任何数据库来工作，但这也是可以的。在这一点上，了解一下如何发布查询信息比直接创建表、向其中加载数据以及从其中检索数据更重要。本节描述了输入查询的基本原则，使用几个你可以尝试的查询来熟悉mysql的工作方式。

这里有一个简单的查询，要求服务器告诉你其版本号和当前日期。按这里所示，在mysql>提示符后输入，并按回车键。

mysql> **SELECT VERSION(), CURRENT\_DATE;**

+-----------+--------------+

| VERSION() | CURRENT\_DATE |

+-----------+--------------+

| 5.8.0-m17 | 2015-12-21 |

+-----------+--------------+

1 row in set (0.02 sec)

mysql>

这个查询说明了关于mysql的几件事。

* 一个查询通常由一个SQL语句和一个分号组成。(有一些例外情况，分号可以被省略。前面提到的QUIT就是其中之一。我们稍后会讨论其他的）。)
* 当你发出一个查询时，mysql将其发送到服务器执行，并显示结果，然后打印另一个mysql>提示，表明它已准备好进行另一个查询。
* mysql以表格的形式显示查询输出（行和列）。第一行包含各列的标签。后面的行是查询结果。通常情况下，列标签是你从数据库表中获取的列的名称。如果你检索的是表达式的值，而不是表的列（如刚才的例子），mysql用表达式本身给列贴标签。
* mysql显示了返回的行数和查询的执行时间，这让你对服务器的性能有一个大致的了解。这些值是不精确的，因为它们代表挂钟时间（而不是CPU或机器时间），而且它们受到服务器负载和网络延迟等因素的影响。为了简洁起见，在本章其余的例子中，有时不显示 "集合中的行 "一行）。

关键词可以用任何字母大写输入。下面的查询是等价的。

mysql> **SELECT VERSION(), CURRENT\_DATE;**

mysql> **select version(), current\_date;**

mysql> **SeLeCt vErSiOn(), current\_DATE;**

这里是另一个查询。它展示了你可以使用mysql作为一个简单的计算器。

mysql> **SELECT SIN(PI()/4), (4+1)\*5;**

+------------------+---------+

| SIN(PI()/4) | (4+1)\*5 |

+------------------+---------+

| 0.70710678118655 | 25 |

+------------------+---------+

1 row in set (0.02 sec)

到目前为止，所显示的查询都是相对较短的单行语句。你甚至可以在一行中输入多个语句。只要用分号来结束每一条。

mysql> **SELECT VERSION(); SELECT NOW();**

+-----------+

| VERSION() |

+-----------+

| 8.0.13 |

+-----------+

1 row in set (0.00 sec)

+---------------------+

| NOW() |

+---------------------+

| 2018-08-24 00:56:40 |

+---------------------+

1 row in set (0.00 sec)

一个查询不需要在一行中全部给出，所以需要几行的冗长查询不是问题。mysql通过寻找结束的分号来确定语句的结束，而不是寻找输入行的结束。(换句话说，mysql接受自由格式的输入：它收集输入行，但在看到分号之前不执行它们。)

下面是一个简单的多行语句。

mysql> **SELECT**

-> **USER()**

-> **,**

-> **CURRENT\_DATE;**

+---------------+--------------+

| USER() | CURRENT\_DATE |

+---------------+--------------+

| jon@localhost | 2018-08-24 |

+---------------+--------------+

在这个例子中，注意在你输入多行查询的第一行后，提示符如何从mysql>变成->。这是mysql表明它还没有看到一个完整的语句，正在等待其余的语句。提示符是你的朋友，因为它提供了有价值的反馈。如果你使用这种反馈，你就可以一直知道mysql在等待什么。

如果你决定不想执行一个正在输入的查询，可以通过键入 \c 来取消它。

mysql> **SELECT**

-> **USER()**

-> **\c**

mysql>

这里也要注意提示。在你输入\c后，它又切换回mysql>，提供反馈以表明mysql已准备好进行新的查询。

下表显示了你可能看到的每个提示，并总结了它们对mysql所处状态的含义。

|  |  |
| --- | --- |
| **Prompt** | **Meaning** |
| **mysql>** | Ready for new query |
| **->** | Waiting for next line of multiple-line query |
| **'>** | Waiting for next line, waiting for completion of a string that began with a single quote (**'**) |
| **">** | Waiting for next line, waiting for completion of a string that began with a double quote (**"**) |
| **`>** | Waiting for next line, waiting for completion of an identifier that began with a backtick (**`**) |
| **/\*>** | Waiting for next line, waiting for completion of a comment that began with **/\*** |

当你打算在单行上发出一个查询，但忘记了结尾的分号，多行语句通常会意外发生。在这种情况下，mysql会等待更多的输入。

mysql> **SELECT USER()**

->

如果这种情况发生在你身上（你认为你已经输入了一条语句，但唯一的反应是->提示），很可能是mysql在等待分号。如果你没有注意到提示的内容，你可能在那里坐了一会儿才意识到你需要做什么。输入分号来完成语句，mysql就会执行它。

mysql> **SELECT USER()**

-> **;**

+---------------+

| USER() |

+---------------+

| jon@localhost |

+---------------+

'>和">提示发生在字符串收集期间（另一种说法是MySQL正在等待字符串的完成）。在MySQL中，你可以写由'或 "字符包围的字符串（例如，'hello'或 "goodbye"），而且mysql允许你输入跨越多行的字符串。当你看到'>或">提示时，这意味着你已经输入了一个包含以'或 "引号字符开始的字符串的行，但还没有输入结束该字符串的匹配引号。这通常表明你无意中漏掉了一个引号字符。比如说

mysql> **SELECT \* FROM my\_table WHERE name = 'Smith AND age < 30;**

'>

如果你输入这个SELECT语句，然后按回车键并等待结果，什么也不会发生。与其想知道为什么这个查询需要这么长时间，不如注意'>提示提供的线索。它告诉你，mysql期望看到一个未结束的字符串的其余部分。(你看到语句中的错误吗？字符串'Smith'缺少第二个单引号）。)

在这一点上，你该怎么做？最简单的事情是取消查询。然而，在这种情况下，你不能只是输入\c，因为mysql把它解释为它正在收集的字符串的一部分。相反，输入结束的引号字符（所以mysql知道你已经完成了这个字符串），然后输入 \c。

mysql> **SELECT \* FROM my\_table WHERE name = 'Smith AND age < 30;**

'> **'\c**

mysql>

提示符变回mysql>，表明mysql已经准备好接受新的查询。

`>提示与'>和">提示类似，但表示你已经开始但没有完成反引号标识符。

知道">"、">"和">"提示的含义很重要，因为如果你错误地输入了一个未结束的字符串，你输入的任何其他行都会被mysql忽略，包括包含QUIT的行。这可能相当令人困惑，特别是如果你不知道在取消当前查询之前需要提供结束语。

注意

从这一点开始，多行语句的书写没有二次（->或其他）提示，以便于复制和粘贴语句，让自己尝试。

## 3.3 创建和使用数据库

一旦你知道如何输入SQL语句，你就可以访问数据库了。

假设你在家里有几只宠物（你的宠物），你想跟踪关于它们的各种信息。你可以通过创建表来保存数据，并将所需的信息加载到表中。然后你可以通过从表中检索数据来回答关于你的动物的各种问题。本节向你展示如何执行下列操作。

* 创建一个数据库
* 创建一个表
* 向表中加载数据
* 以各种方式从表中检索数据
* 使用多个表

驯兽师数据库很简单（故意的），但不难想到现实世界中可能使用类似类型的数据库的情况。例如，一个农民可以用这样的数据库来跟踪牲畜，或者一个兽医来跟踪病人的记录。可以从MySQL网站上获得一个包含以下章节中使用的一些查询和样本数据的managerie分布。它以压缩的tar文件和Zip格式提供，网址是https://dev.mysql.com/doc/。

使用SHOW语句找出服务器上目前存在哪些数据库。

mysql> **SHOW DATABASES;**

+----------+

| Database |

+----------+

| mysql |

| test |

| tmp |

+----------+

mysql数据库描述了用户的访问权限。测试数据库通常作为工作区供用户试用。

语句显示的数据库列表在你的机器上可能不同；如果你没有SHOW DATABASES权限，SHOW DATABASES不会显示你没有权限的数据库。参见第13.7.7.14节，"SHOW DATABASES语句"。

如果测试数据库存在，尝试访问它。

mysql> **USE test**

Database changed

USE，像QUIT一样，不需要分号。(如果你愿意，你可以用分号来结束这样的语句；这没有什么坏处)。USE 语句在另一个方面也很特别：它必须在单行上给出。

你可以在后面的例子中使用测试数据库(如果你有权限的话)，但你在该数据库中创建的任何东西都可以被其他有权限的人删除。由于这个原因，你也许应该向你的MySQL管理员申请许可，以使用你自己的数据库。假设你想调用你的menagerie。管理员需要执行这样的语句。

mysql> **GRANT ALL ON menagerie.\* TO 'your\_mysql\_name'@'your\_client\_host';**

其中 your\_mysql\_name 是分配给你的 MySQL 用户名， your\_client\_host 是你连接到服务器的主机。

### 3.3.1 创建和选择一个数据库

如果管理员在设置你的权限时为你创建了数据库，你就可以开始使用它。否则，你需要自己创建它。

mysql> **CREATE DATABASE menagerie;**

在Unix下，数据库名称是区分大小写的（与SQL关键字不同），所以你必须总是把你的数据库称为menagerie，而不是Menagerie、MENAGERIE或其他变体。这对表名来说也是如此。(在Windows下，这个限制并不适用，尽管你必须在一个给定的查询中使用相同的大写字母来指代数据库和表。然而，由于各种原因，推荐的最佳做法是始终使用创建数据库时使用的相同字母大小写）。)

注意

如果你得到一个错误，例如 ERROR 1044 (42000): 拒绝用户'micah'@'localhost'对数据库'menagerie'的访问，这意味着你的用户账户没有必要的权限来创建数据库。请与管理员讨论这个问题，或者参见第6.2节，"访问控制和账户管理"。

创建一个数据库并不意味着选择它来使用；你必须明确地这样做。要使menagerie成为当前的数据库，使用这个语句。

mysql> **USE menagerie**

Database changed

你的数据库只需要创建一次，但你必须在每次开始mysql会话时选择它来使用。你可以通过发布USE语句来做到这一点，如例子中所示。或者，当你调用mysql时，你可以在命令行上选择该数据库。只要在你可能需要提供的任何连接参数之后指定它的名称。例如

shell> **mysql -h *host* -u *user* -p menagerie**

Enter password: **\*\*\*\*\*\*\*\***

重要提示

刚才的命令中的menagerie不是你的密码。如果你想在命令行中的-p选项之后提供你的密码，你必须这样做，并且没有中间的空格（例如，作为-ppassword，而不是作为-p password）。然而，不建议将你的密码放在命令行上，因为这样做会使你的密码被登录在你机器上的其他用户窥探到。

注意

你可以在任何时候使用SELECT DATABASE()查看当前选择的数据库。

### 3.3.2 创建一个表

创建数据库是容易的部分，但此时它是空的，正如SHOW TABLES告诉你的。

mysql> **SHOW TABLES;**

Empty set (0.00 sec)

更难的部分是决定你的数据库结构应该是什么：你需要什么表，每个表里应该有什么列。

你想要一个表，其中包含你的每个宠物的记录。这可以被称为宠物表，它至少应该包含每个动物的名字。因为名字本身并不十分有趣，该表应该包含其他信息。例如，如果你家里不止一个人养宠物，你可能想列出每个动物的主人。你可能还想记录一些基本的描述性信息，如物种和性别。

那么年龄呢？这可能是有意义的，但它不是一个好东西，可以存储在数据库中。年龄随着时间的推移而变化，这意味着你必须经常更新你的记录。相反，最好是存储一个固定值，如出生日期。然后，每当你需要年龄时，你可以把它计算为当前日期和出生日期之间的差异。MySQL提供了做日期运算的函数，所以这并不困难。存储出生日期而不是年龄也有其他好处。

* 你可以使用数据库来完成一些任务，例如为即将到来的宠物生日生成提醒信息。如果你认为这种类型的查询有点傻，那么请注意，在商业数据库中，你可能会问同样的问题，以确定你需要在本周或本月向其发送生日问候的客户，以获得计算机辅助的个人感觉。
* 你可以计算与当前日期以外的日期有关的年龄。例如，如果你在数据库中存储了死亡日期，你可以很容易地计算出一只宠物死亡时的年龄。

你可能还能想到其他类型的信息在宠物表中是有用的，但到目前为止所确定的信息已经足够了：名字、主人、物种、性别、出生和死亡。

使用CREATE TABLE语句来指定你的表的布局。

mysql> **CREATE TABLE pet (name VARCHAR(20), owner VARCHAR(20),**

**species VARCHAR(20), sex CHAR(1), birth DATE, death DATE);**

VARCHAR是名字、所有者和物种列的一个很好的选择，因为这些列值的长度是不同的。这些列定义中的长度不一定都是一样的，也不一定是20。你通常可以选择从1到65535的任何长度，只要是你认为最合理的。如果你做了一个糟糕的选择，后来发现你需要一个更长的字段，MySQL提供一个ALTER TABLE语句。

可以选择几种类型的值来代表动物记录中的性别，例如 "m "和 "f"，或者是 "male "和 "female"。最简单的方法是使用单个字符'm'和'f'。

为出生和死亡列使用DATE数据类型是一个相当明显的选择。

一旦你创建了一个表，SHOW TABLES应该产生一些输出。

mysql> **SHOW TABLES;**

+---------------------+

| Tables in menagerie |

+---------------------+

| pet |

+---------------------+

为了验证你的表是否按照你预期的方式创建，使用DESCRIBE语句。

mysql> **DESCRIBE pet;**

+---------+-------------+------+-----+---------+-------+

| Field | Type | Null | Key | Default | Extra |

+---------+-------------+------+-----+---------+-------+

| name | varchar(20) | YES | | NULL | |

| owner | varchar(20) | YES | | NULL | |

| species | varchar(20) | YES | | NULL | |

| sex | char(1) | YES | | NULL | |

| birth | date | YES | | NULL | |

| death | date | YES | | NULL | |

+---------+-------------+------+-----+---------+-------+

你可以在任何时候使用DESCRIBE，例如，如果你忘记了你表中的列的名称或它们的类型。

关于MySQL数据类型的更多信息，见第11章，数据类型。

### 3.3.3 向表中加载数据

在创建你的表之后，你需要填充它。LOAD DATA和INSERT语句在这方面很有用。

假设你的宠物记录可以被描述为如下所示。(注意，MySQL期望的日期是 "YYYY-MM-DD "格式；这可能与你习惯的不同。)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **name** | **owner** | **species** | **sex** | **birth** | **death** |
| **Fluffy** | Harold | cat | f | 1993-02-04 |  |
| **Claws** | Gwen | cat | m | 1994-03-17 |  |
| **Buffy** | Harold | dog | f | 1989-05-13 |  |
| **Fang** | Benny | dog | m | 1990-08-27 |  |
| **Bowser** | Diane | dog | m | 1979-08-31 | 1995-07-29 |
| **Chirpy** | Gwen | bird | f | 1998-09-11 |  |
| **Whistler** | Gwen | bird |  | 1997-12-09 |  |
| **Slim** | Benny | snake | m | 1996-04-29 |  |

因为你是从一个空表开始的，所以填充它的一个简单方法是创建一个文本文件，其中包含每个动物的一行，然后用一个语句将文件的内容加载到表中。

你可以创建一个文本文件pet.txt，每行包含一条记录，数值用制表符分隔，并按照CREATE TABLE语句中列的顺序给出。对于缺失的值（比如未知的性别或仍然活着的动物的死亡日期），你可以使用NULL值。要在你的文本文件中表示这些，请使用 \N（反斜杠，大写N）。例如，鸟类Whistler的记录看起来是这样的（其中数值之间的空白是一个单一的制表符）。

Whistler Gwen bird \N 1997-12-09 \N

要将文本文件pet.txt加载到pet表中，使用这个语句。

mysql> **LOAD DATA LOCAL INFILE '/path/pet.txt' INTO TABLE pe**

如果你在Windows上用一个使用\r\n作为行结束符的编辑器创建该文件，你应该使用这个语句来代替。

mysql> **LOAD DATA LOCAL INFILE '/path/pet.txt' INTO TABLE pet**

**LINES TERMINATED BY '\r\n';**

(在运行macOS的苹果机器上，你可能想使用LINES TERMINATED BY '\r')。

如果你愿意，你可以在LOAD DATA语句中明确指定列值分隔符和行尾标记，但默认的是TAB和换行符。这些足以让语句正确读取文件pet.txt。

如果语句失败，可能是你的MySQL安装没有默认启用本地文件功能。参见第6.1.6节，"LOAD DATA LOCAL的安全考虑"，以了解如何改变这种情况。

当你想一次添加新记录时，INSERT语句很有用。在其最简单的形式中，你为每一列提供数值，按照列在CREATE TABLE语句中的顺序排列。假设Diane得到一只新的仓鼠，名叫 "Puffball"。你可以使用像这样的INSERT语句添加一条新记录。

mysql> **INSERT INTO pet**

**VALUES ('Puffball','Diane','hamster','f','1999-03-30',NULL);**

这里的字符串和日期值被指定为带引号的字符串。另外，通过INSERT，你可以直接插入NULL来表示一个缺失的值。你不像在LOAD DATA中那样使用\N。

从这个例子中，你应该可以看到，最初使用几个INSERT语句而不是一个单一的LOAD DATA语句来加载你的记录，会有更多的打字工作。

### 3.3.4 从表中检索信息

SELECT语句是用来从一个表中提取信息的。该语句的一般形式是

SELECT ***what\_to\_select***

FROM ***which\_table***

WHERE ***conditions\_to\_satisfy***;

what\_to\_select表示你想看到的东西。这可以是一个列的列表，或者用\*来表示 "所有的列"。WHERE子句是可选的。如果它存在，conditions\_to\_satisfy指定了一个或多个条件，这些行必须满足这些条件才有资格被检索。

#### 3.3.4.1 选择所有数据

最简单的SELECT形式可以从一个标签中检索所有数据

mysql> **SELECT \* FROM pet;**

+----------+--------+---------+------+------------+------------+

| name | owner | species | sex | birth | death |

+----------+--------+---------+------+------------+------------+

| Fluffy | Harold | cat | f | 1993-02-04 | NULL |

| Claws | Gwen | cat | m | 1994-03-17 | NULL |

| Buffy | Harold | dog | f | 1989-05-13 | NULL |

| Fang | Benny | dog | m | 1990-08-27 | NULL |

| Bowser | Diane | dog | m | 1979-08-31 | 1995-07-29 |

| Chirpy | Gwen | bird | f | 1998-09-11 | NULL |

| Whistler | Gwen | bird | NULL | 1997-12-09 | NULL |

| Slim | Benny | snake | m | 1996-04-29 | NULL |

| Puffball | Diane | hamster | f | 1999-03-30 | NULL |

+----------+--------+---------+------+------------+------------+

这种形式的SELECT使用\*，它是 "选择所有列 "的缩写。如果你想查看整个表，例如，在你刚刚加载了你的初始数据集之后，这很有用。例如，你可能碰巧觉得Bowser的出生日期似乎不大对劲。咨询你的原始血统文件，你发现正确的出生年份应该是1989年，而不是1979年。

至少有两种方法可以解决这个问题。

* 编辑文件pet.txt以纠正错误，然后清空表并使用DELETE和LOAD DATA重新加载它。

mysql> **DELETE FROM pet;**

mysql> **LOAD DATA LOCAL INFILE 'pet.txt' INTO TABLE pet;**

然而，如果你这样做，你也必须重新输入Puffball的记录。

* 用UPDATE语句只修复错误的记录。

mysql> **UPDATE pet SET birth = '1989-08-31' WHERE name = 'Bowser';**

UPDATE只改变了有问题的记录，不要求你重新加载表。

SELECT \*选择所有列的原则有一个例外。如果一个表包含不可见的列，\*不包括它们。更多信息请参见章节13.1.20.10, "不可见列"。

#### 3.3.4.2 选择特定行

如上一节所示，检索整个表是很容易的。只要在SELECT语句中省略WHERE子句即可。但是，通常你不想看到整个表，特别是当它变得很大的时候。相反，你通常对回答一个特定的问题更感兴趣，在这种情况下，你会对你想要的信息指定一些约束。让我们从它们所回答的关于你的宠物的问题方面来看看一些选择查询。

你可以只从你的表中选择特定的行。例如，如果你想验证你对Bowser的出生日期所做的改变，可以像这样选择Bowser的记录。

mysql> **SELECT \* FROM pet WHERE name = 'Bowser';**

+--------+-------+---------+------+------------+------------+

| name | owner | species | sex | birth | death |

+--------+-------+---------+------+------------+------------+

| Bowser | Diane | dog | m | 1989-08-31 | 1995-07-29 |

+--------+-------+---------+------+------------+------------+

输出结果证实，年份被正确记录为1989年，而不是1979年。

字符串比较通常是不区分大小写的，所以你可以指定名称为'bowser'、'BOWSER'等。查询的结果也是一样的。

你可以在任何列上指定条件，而不仅仅是名字。例如，如果你想知道哪些动物是在1998年期间或之后出生的，可以测试出生列。

mysql> **SELECT \* FROM pet WHERE birth >= '1998-1-1';**

+----------+-------+---------+------+------------+-------+

| name | owner | species | sex | birth | death |

+----------+-------+---------+------+------------+-------+

| Chirpy | Gwen | bird | f | 1998-09-11 | NULL |

| Puffball | Diane | hamster | f | 1999-03-30 | NULL |

你可以结合条件，例如，找到母狗。

mysql> **SELECT \* FROM pet WHERE species = 'dog' AND sex = 'f';**

+-------+--------+---------+------+------------+-------+

| name | owner | species | sex | birth | death |

+-------+--------+---------+------+------------+-------+

| Buffy | Harold | dog | f | 1989-05-13 | NULL |

+-------+--------+---------+------+------------+-------+

前面的查询使用了AND逻辑运算符。还有一个OR运算符

mysql> **SELECT \* FROM pet WHERE species = 'snake' OR species = 'bird';**

+----------+-------+---------+------+------------+-------+

| name | owner | species | sex | birth | death |

+----------+-------+---------+------+------------+-------+

| Chirpy | Gwen | bird | f | 1998-09-11 | NULL |

| Whistler | Gwen | bird | NULL | 1997-12-09 | NULL |

| Slim | Benny | snake | m | 1996-04-29 | NULL |

+----------+-------+---------+------+------------+-------+

AND和OR可以混合使用，尽管AND的优先级比OR高。如果你同时使用这两个运算符，最好使用圆括号来明确指出条件应该如何分组。

mysql> **SELECT \* FROM pet WHERE (species = 'cat' AND sex = 'm')**

**OR (species = 'dog' AND sex = 'f');**

+-------+--------+---------+------+------------+-------+

| name | owner | species | sex | birth | death |

+-------+--------+---------+------+------------+-------+

| Claws | Gwen | cat | m | 1994-03-17 | NULL |

| Buffy | Harold | dog | f | 1989-05-13 | NULL |

+-------+--------+---------+------+------------+-------+

#### 3.3.4.3 选择特定列

如果你不想从你的表中看到整个行，只需命名你感兴趣的列，用逗号隔开。例如，如果你想知道你的动物是什么时候出生的，就选择名字和出生列。

mysql> **SELECT name, birth FROM pet;**

+----------+------------+

| name | birth |

+----------+------------+

| Fluffy | 1993-02-04 |

| Claws | 1994-03-17 |

| Buffy | 1989-05-13 |

| Fang | 1990-08-27 |

| Bowser | 1989-08-31 |

| Chirpy | 1998-09-11 |

| Whistler | 1997-12-09 |

| Slim | 1996-04-29 |

| Puffball | 1999-03-30 |

要了解谁拥有宠物，请使用这个查询。

mysql> **SELECT owner FROM pet;**

+--------+

| owner |

+--------+

| Harold |

| Gwen |

| Harold |

| Benny |

| Diane |

| Gwen |

| Gwen |

| Benny |

| Diane |

+--------+

请注意，该查询只是从每条记录中检索所有者列，而其中有些记录出现了不止一次。为了尽量减少输出，可以通过添加关键字DISTINCT，只检索每条唯一的输出记录一次。

mysql> **SELECT DISTINCT owner FROM pet;**

+--------+

| owner |

+--------+

| Benny |

| Diane |

| Gwen |

| Harold |

+--------+

你可以使用WHERE子句来结合行选择和列选择。例如，如果只想获得狗和猫的出生日期，可以使用这个查询

mysql> **SELECT name, species, birth FROM pet**

**WHERE species = 'dog' OR species = 'cat';**

+--------+---------+------------+

| name | species | birth |

+--------+---------+------------+

| Fluffy | cat | 1993-02-04 |

| Claws | cat | 1994-03-17 |

| Buffy | dog | 1989-05-13 |

| Fang | dog | 1990-08-27 |

| Bowser | dog | 1989-08-31 |

+--------+---------+------------+

#### 3.3.4.4 对行进行排序

你可能已经注意到在前面的例子中，结果行的显示没有特定的顺序。当行以某种有意义的方式进行排序时，通常会更容易检查查询的输出。要对一个结果进行排序，可以使用ORDER BY子句。

下面是动物的生日，按日期排序。

mysql> **SELECT name, birth FROM pet ORDER BY birth;**

+----------+------------+

| name | birth |

+----------+------------+

| Buffy | 1989-05-13 |

| Bowser | 1989-08-31 |

| Fang | 1990-08-27 |

| Fluffy | 1993-02-04 |

| Claws | 1994-03-17 |

| Slim | 1996-04-29 |

| Whistler | 1997-12-09 |

| Chirpy | 1998-09-11 |

| Puffball | 1999-03-30 |

+----------+------------+

在字符类型的列上，排序与所有其他比较操作一样，通常以不区分大小写的方式进行。这意味着，对于除了大小写之外完全相同的列，其顺序是不确定的。你可以通过使用BINARY来强制对一个列进行大小写排序，就像这样。ORDER BY BINARY col\_name。

默认的排序顺序是升序，最小的值在前。要以反向（降序）排序，请在你要排序的列的名称上添加DESC关键字。

mysql> **SELECT name, birth FROM pet ORDER BY birth DESC;**

+----------+------------+

| name | birth |

+----------+------------+

| Puffball | 1999-03-30 |

| Chirpy | 1998-09-11 |

| Whistler | 1997-12-09 |

| Slim | 1996-04-29 |

| Claws | 1994-03-17 |

| Fluffy | 1993-02-04 |

| Fang | 1990-08-27 |

| Bowser | 1989-08-31 |

| Buffy | 1989-05-13 |

+----------+------------+

你可以对多列进行排序，也可以对不同的列按不同的方向进行排序。例如，要按动物类型以升序排序，然后按动物类型中的出生日期以降序排序（最年轻的动物在前），请使用以下查询。

mysql> **SELECT name, species, birth FROM pet**

**ORDER BY species, birth DESC;**

+----------+---------+------------+

| name | species | birth |

+----------+---------+------------+

| Chirpy | bird | 1998-09-11 |

| Whistler | bird | 1997-12-09 |

| Claws | cat | 1994-03-17 |

| Fluffy | cat | 1993-02-04 |

| Fang | dog | 1990-08-27 |

| Bowser | dog | 1989-08-31 |

| Buffy | dog | 1989-05-13 |

| Puffball | hamster | 1999-03-30 |

| Slim | snake | 1996-04-29 |

+----------+---------+------------+

DESC关键字只适用于紧接在它前面的列名（出生）；它不影响物种列的排序顺序。

#### 3.3.4.5 日期计算

MySQL提供了几个函数，你可以用来对日期进行计算，例如，计算年龄或提取日期的一部分。

要确定你的每个宠物的年龄是多少岁，可以使用TIMESTAMPDIFF()函数。它的参数是你希望结果表达的单位，以及要取差的两个日期。下面的查询显示了每个宠物的出生日期、当前日期和年龄。一个别名（age）被用来使最后的输出列标签更有意义。

mysql> **SELECT name, birth, CURDATE(),**

**TIMESTAMPDIFF(YEAR,birth,CURDATE()) AS age**

**FROM pet;**

+----------+------------+------------+------+

| name | birth | CURDATE() | age |

+----------+------------+------------+------+

| Fluffy | 1993-02-04 | 2003-08-19 | 10 |

| Claws | 1994-03-17 | 2003-08-19 | 9 |

| Buffy | 1989-05-13 | 2003-08-19 | 14 |

| Fang | 1990-08-27 | 2003-08-19 | 12 |

| Bowser | 1989-08-31 | 2003-08-19 | 13 |

| Chirpy | 1998-09-11 | 2003-08-19 | 4 |

| Whistler | 1997-12-09 | 2003-08-19 | 5 |

| Slim | 1996-04-29 | 2003-08-19 | 7 |

| Puffball | 1999-03-30 | 2003-08-19 | 4 |

+----------+------------+------------+------+

这个查询是有效的，但是如果这些行以某种顺序呈现的话，可以更容易地扫描出结果。这可以通过添加一个ORDER BY name子句来实现，以按名字对输出进行排序。

mysql> **SELECT name, birth, CURDATE(),**

**TIMESTAMPDIFF(YEAR,birth,CURDATE()) AS age**

**FROM pet ORDER BY name;**

+----------+------------+------------+------+

| name | birth | CURDATE() | age |

+----------+------------+------------+------+

| Bowser | 1989-08-31 | 2003-08-19 | 13 |

| Buffy | 1989-05-13 | 2003-08-19 | 14 |

| Chirpy | 1998-09-11 | 2003-08-19 | 4 |

| Claws | 1994-03-17 | 2003-08-19 | 9 |

| Fang | 1990-08-27 | 2003-08-19 | 12 |

| Fluffy | 1993-02-04 | 2003-08-19 | 10 |

| Puffball | 1999-03-30 | 2003-08-19 | 4 |

| Slim | 1996-04-29 | 2003-08-19 | 7 |

| Whistler | 1997-12-09 | 2003-08-19 | 5 |

+----------+------------+------------+------+

要按年龄而不是姓名对输出进行排序，只需使用不同的ORDER BY子句。

mysql> **SELECT name, birth, CURDATE(),**

**TIMESTAMPDIFF(YEAR,birth,CURDATE()) AS age**

**FROM pet ORDER BY age;**

+----------+------------+------------+------+

| name | birth | CURDATE() | age |

+----------+------------+------------+------+

| Chirpy | 1998-09-11 | 2003-08-19 | 4 |

| Puffball | 1999-03-30 | 2003-08-19 | 4 |

| Whistler | 1997-12-09 | 2003-08-19 | 5 |

| Slim | 1996-04-29 | 2003-08-19 | 7 |

| Claws | 1994-03-17 | 2003-08-19 | 9 |

| Fluffy | 1993-02-04 | 2003-08-19 | 10 |

| Fang | 1990-08-27 | 2003-08-19 | 12 |

| Bowser | 1989-08-31 | 2003-08-19 | 13 |

| Buffy | 1989-05-13 | 2003-08-19 | 14 |

+----------+------------+------------+------+

一个类似的查询可以用来确定已经死亡的动物的死亡年龄。你可以通过检查死亡值是否为NULL来确定这些动物。然后，对于那些非NULL值的动物，计算死亡值和出生值之间的差异。

mysql> **SELECT name, birth, death,**

**TIMESTAMPDIFF(YEAR,birth,death) AS age**

**FROM pet WHERE death IS NOT NULL ORDER BY age;**

+--------+------------+------------+------+

| name | birth | death | age |

+--------+------------+------------+------+

| Bowser | 1989-08-31 | 1995-07-29 | 5 |

+--------+------------+------------+------+

这个查询使用的是death IS NOT NULL，而不是death <> NULL，因为NULL是一个特殊的值，不能用通常的比较运算符来比较。这一点将在后面讨论。参见第3.3.4.6节，"处理NULL值"。

如果你想知道哪些动物在下个月有生日呢？对于这种类型的计算，年和日是不相关的；你只是想提取出生列的月份部分。MySQL提供了几个函数用于提取日期的一部分，如YEAR()、MONTH()和DAYOFMONTH()。MONTH()是这里合适的函数。为了看看它是如何工作的，运行一个简单的查询，显示出生和MONTH(出生)的值。

mysql> **SELECT name, birth, MONTH(birth) FROM pet;**

+----------+------------+--------------+

| name | birth | MONTH(birth) |

+----------+------------+--------------+

| Fluffy | 1993-02-04 | 2 |

| Claws | 1994-03-17 | 3 |

| Buffy | 1989-05-13 | 5 |

| Fang | 1990-08-27 | 8 |

| Bowser | 1989-08-31 | 8 |

| Chirpy | 1998-09-11 | 9 |

| Whistler | 1997-12-09 | 12 |

| Slim | 1996-04-29 | 4 |

| Puffball | 1999-03-30 | 3 |

+----------+------------+--------------+

寻找生日在下个月的动物也很简单。假设当前的月份是四月。那么月份值是4，你可以这样寻找在5月（第5个月）出生的动物。

mysql> **SELECT name, birth FROM pet WHERE MONTH(birth) = 5;**

+-------+------------+

| name | birth |

+-------+------------+

| Buffy | 1989-05-13 |

+-------+------------+

如果当前的月份是12月，就会出现一个小的复杂情况。你不能仅仅在月号（12）上加一，然后寻找在13月出生的动物，因为没有这样的月份。相反，你要寻找出生在一月（第1个月）的动物。

你可以编写查询，这样无论当前的月份是什么，它都可以工作，这样你就不必使用某个特定月份的数字。DATE\_ADD()可以使你在一个给定的日期上增加一个时间间隔。如果你在CURDATE()的值中加入一个月份，然后用MONTH()提取月份部分，其结果是产生一个可以寻找生日的月份。

mysql> **SELECT name, birth FROM pet**

**WHERE MONTH(birth) = MONTH(DATE\_ADD(CURDATE(),INTERVAL 1 MONTH));**

完成同样任务的另一种方法是，在使用模数函数（MOD）将月份值包裹为0（如果当前是12）之后，加1来获得当前月份之后的下一个月。

mysql> **SELECT name, birth FROM pet**

**WHERE MONTH(birth) = MOD(MONTH(CURDATE()), 12) + 1;**

MONTH()返回1到12之间的一个数字。而MOD(something,12)返回0到11之间的数字。所以加法必须在MOD()之后，否则我们会从11月（11）到1月（1）。

如果一个计算使用了无效的日期，计算就会失败并产生警告。

mysql> **SELECT '2018-10-31' + INTERVAL 1 DAY;**

+-------------------------------+

| '2018-10-31' + INTERVAL 1 DAY |

+-------------------------------+

| 2018-11-01 |

+-------------------------------+

mysql> **SELECT '2018-10-32' + INTERVAL 1 DAY;**

+-------------------------------+

| '2018-10-32' + INTERVAL 1 DAY |

+-------------------------------+

| NULL |

+-------------------------------+

mysql> **SHOW WARNINGS;**

+---------+------+----------------------------------------+

| Level | Code | Message |

+---------+------+----------------------------------------+

| Warning | 1292 | Incorrect datetime value: '2018-10-32' |

+---------+------+----------------------------------------+

#### 3.3.4.6 与NULL值一起工作

在你习惯它之前，NULL值可能会令人惊讶。从概念上讲，NULL意味着 "一个缺失的未知值"，它的处理方式与其他值有些不同。

为了测试NULL，可以使用IS NULL和IS NOT NULL操作符，如图所示。

mysql> **SELECT 1 IS NULL, 1 IS NOT NULL;**

+-----------+---------------+

| 1 IS NULL | 1 IS NOT NULL |

+-----------+---------------+

| 0 | 1 |

+-----------+---------------+

你不能使用算术比较运算符，如=，<，或<>来测试NULL。为了证明这一点，请尝试下面的查询。

mysql> **SELECT 1 = NULL, 1 <> NULL, 1 < NULL, 1 > NULL;**

+----------+-----------+----------+----------+

| 1 = NULL | 1 <> NULL | 1 < NULL | 1 > NULL |

+----------+-----------+----------+----------+

| NULL | NULL | NULL | NULL |

+----------+-----------+----------+----------+

因为与NULL进行任何算术比较的结果也是NULL，你不能从这种比较中获得任何有意义的结果。

在MySQL中，0或NULL意味着假，其他的都意味着真。布尔运算的默认真值是1。

对NULL的这种特殊处理是为什么在上一节中，有必要使用死亡不是NULL而不是死亡<>NULL来确定哪些动物已经不在世。

在GROUP BY中，两个NULL值被认为是相等的。

当做ORDER BY时，如果你做ORDER BY ...，NULL值会首先出现，如果你做ORDER BY ...，NULL值会最后出现。ASC，最后一个是ORDER BY ... DESC。

在处理NULL时，一个常见的错误是认为不可能在定义为NOT NULL的列中插入一个零或空字符串，但事实并非如此。这些实际上是值，而NULL意味着 "没有值"。你可以通过使用IS [NOT] NULL来测试这一点，如图所示。

mysql> **SELECT 0 IS NULL, 0 IS NOT NULL, '' IS NULL, '' IS NOT NULL;**

+-----------+---------------+------------+----------------+

| 0 IS NULL | 0 IS NOT NULL | '' IS NULL | '' IS NOT NULL |

+-----------+---------------+------------+----------------+

| 0 | 1 | 0 | 1 |

+-----------+---------------+------------+----------------+

因此，完全可以在NOT NULL列中插入一个零或空的字符串，因为这些实际上是NOT NULL。参见B.3.4.3节，"NULL值的问题"。

#### 3.3.4.7 模式匹配

MySQL提供标准的SQL模式匹配，以及一种基于扩展的正则表达式的模式匹配，类似于Unix工具（如vi、grep和sed）使用的模式。

SQL模式匹配使你能够使用\_来匹配任何单个字符，使用%来匹配任意数量的字符（包括零字符）。在MySQL中，SQL模式默认是不区分大小写的。这里显示了一些例子。当你使用SQL模式时，不要使用=或<>。使用LIKE或NOT LIKE比较运算符来代替。

要查找以b开头的名字。

mysql> **SELECT \* FROM pet WHERE name LIKE 'b%';**

+--------+--------+---------+------+------------+------------+

| name | owner | species | sex | birth | death |

+--------+--------+---------+------+------------+------------+

| Buffy | Harold | dog | f | 1989-05-13 | NULL |

| Bowser | Diane | dog | m | 1989-08-31 | 1995-07-29 |

+--------+--------+---------+------+------------+------------+

要找到以fy结尾的名字。

mysql> **SELECT \* FROM pet WHERE name LIKE '%fy';**

+--------+--------+---------+------+------------+-------+

| name | owner | species | sex | birth | death |

+--------+--------+---------+------+------------+-------+

| Fluffy | Harold | cat | f | 1993-02-04 | NULL |

| Buffy | Harold | dog | f | 1989-05-13 | NULL |

+--------+--------+---------+------+------------+-------+

要找到含有w的名字。

mysql> **SELECT \* FROM pet WHERE name LIKE '%w%';**

+----------+-------+---------+------+------------+------------+

| name | owner | species | sex | birth | death |

+----------+-------+---------+------+------------+------------+

| Claws | Gwen | cat | m | 1994-03-17 | NULL |

| Bowser | Diane | dog | m | 1989-08-31 | 1995-07-29 |

| Whistler | Gwen | bird | NULL | 1997-12-09 | NULL |

+----------+-------+---------+------+------------+------------+

要查找正好包含五个字符的名称，请使用\_模式字符的五个实例。

mysql> **SELECT \* FROM pet WHERE name LIKE '\_\_\_\_\_';**

+-------+--------+---------+------+------------+-------+

| name | owner | species | sex | birth | death |

+-------+--------+---------+------+------------+-------+

| Claws | Gwen | cat | m | 1994-03-17 | NULL |

| Buffy | Harold | dog | f | 1989-05-13 | NULL |

+-------+--------+---------+------+------------+-------+

MySQL提供的另一种类型的模式匹配使用扩展的正则表达式。当你测试这种类型的模式的匹配时，使用REGEXP\_LIKE()函数（或REGEXP或RLIKE操作符，它们是REGEXP\_LIKE()的同义词）。

下面的列表描述了扩展正则表达式的一些特征。

* .匹配任何单个字符。
* 字符类 [...] 匹配括号内的任何字符。例如，[abc] 匹配a、b或c。要命名一个字符范围，使用破折号。[a-z] 匹配任何字母，而[0-9] 匹配任何数字。
* \*匹配前面的事物的零或多个实例。例如，x\*匹配任何数量的x字符，[0-9]\*匹配任何数量的数字，而.\*匹配任何数量的东西
* 如果一个正则表达式模式匹配在被测试值的任何地方匹配，那么该模式就会成功。这与LIKE模式匹配不同，后者只在模式匹配整个值时才会成功）。
* 要锚定一个模式，使其必须匹配被测试值的开头或结尾，可以在模式的开头使用^，在结尾使用$。

为了演示扩展的正则表达式是如何工作的，前面显示的LIKE查询在这里被改写为使用REGEXP\_LIKE()。

要查找以b开头的名字，使用^来匹配名字的开头

mysql> **SELECT \* FROM pet WHERE REGEXP\_LIKE(name, '^b');**

+--------+--------+---------+------+------------+------------+

| name | owner | species | sex | birth | death |

+--------+--------+---------+------+------------+------------+

| Buffy | Harold | dog | f | 1989-05-13 | NULL |

| Bowser | Diane | dog | m | 1979-08-31 | 1995-07-29 |

+--------+--------+---------+------+------------+------------+

要强制正则表达式比较区分大小写，请使用区分大小写的整理方式，或使用BINARY关键字使其中一个字符串成为二进制字符串，或指定c匹配控制字符。这些查询中的每一个都只匹配名字开头的小写b。

SELECT \* FROM pet WHERE REGEXP\_LIKE(name, '^b' COLLATE utf8mb4\_0900\_as\_cs);

SELECT \* FROM pet WHERE REGEXP\_LIKE(name, BINARY '^b');

SELECT \* FROM pet WHERE REGEXP\_LIKE(name, '^b', 'c');

要找到以fy结尾的名字，用$来匹配名字的结尾。

mysql> **SELECT \* FROM pet WHERE REGEXP\_LIKE(name, 'fy$');**

+--------+--------+---------+------+------------+-------+

| name | owner | species | sex | birth | death |

+--------+--------+---------+------+------------+-------+

| Fluffy | Harold | cat | f | 1993-02-04 | NULL |

| Buffy | Harold | dog | f | 1989-05-13 | NULL |

+--------+--------+---------+------+------------+-------+

要找到含有w的名字，请使用这个查询。

mysql> **SELECT \* FROM pet WHERE REGEXP\_LIKE(name, 'w');**

+----------+-------+---------+------+------------+------------+

| name | owner | species | sex | birth | death |

+----------+-------+---------+------+------------+------------+

| Claws | Gwen | cat | m | 1994-03-17 | NULL |

| Bowser | Diane | dog | m | 1989-08-31 | 1995-07-29 |

| Whistler | Gwen | bird | NULL | 1997-12-09 | NULL |

+----------+-------+---------+------+------------+------------+

因为正则表达式模式在值的任何地方出现都会匹配，所以在前面的查询中，没有必要像SQL模式那样，在模式的两边加上通配符来让它匹配整个值。

要找到正好包含五个字符的名字，使用^和$来匹配名字的开头和结尾，以及中间的五个.实例。

mysql> **SELECT \* FROM pet WHERE REGEXP\_LIKE(name, '^.....$');**

+-------+--------+---------+------+------------+-------+

| name | owner | species | sex | birth | death |

+-------+--------+---------+------+------------+-------+

| Claws | Gwen | cat | m | 1994-03-17 | NULL |

| Buffy | Harold | dog | f | 1989-05-13 | NULL |

+-------+--------+---------+------+------------+-------+

你也可以用{n}（"重复-n次"）操作符来写前面的查询

mysql> **SELECT \* FROM pet WHERE REGEXP\_LIKE(name, '^.{5}$');**

+-------+--------+---------+------+------------+-------+

| name | owner | species | sex | birth | death |

+-------+--------+---------+------+------------+-------+

| Claws | Gwen | cat | m | 1994-03-17 | NULL |

| Buffy | Harold | dog | f | 1989-05-13 | NULL |

+-------+--------+---------+------+------------+-------+

关于正则表达式语法的更多信息，请参见第12.8.2节，"正则表达式"。

#### 3.3.4.8 计算行数

数据库经常被用来回答这样一个问题："某种类型的数据在表中出现的频率如何？" 例如，你可能想知道你有多少只宠物，或者每个主人有多少只宠物，或者你可能想对你的动物进行各种普查操作。

计算你拥有的动物总数与 "宠物表有多少行？"是同一个问题，因为每个宠物有一条记录。COUNT(\*)计算行数，所以计算动物的查询看起来像这样。

mysql> **SELECT COUNT(\*) FROM pet;**

+----------+

| COUNT(\*) |

+----------+

| 9 |

+----------+

刚才，你检索了拥有宠物的人的名字。如果你想知道每个主人有多少只宠物，你可以使用COUNT()。

mysql> **SELECT owner, COUNT(\*) FROM pet GROUP BY owner;**

+--------+----------+

| owner | COUNT(\*) |

+--------+----------+

| Benny | 2 |

| Diane | 2 |

| Gwen | 3 |

| Harold | 2 |

+--------+----------+

前面的查询使用GROUP BY来分组每个所有者的所有记录。结合GROUP BY使用COUNT()对于在各种分组下描述你的数据特征是非常有用的。下面的例子显示了执行动物普查操作的不同方式。

每个物种的动物数量。

mysql> **SELECT species, COUNT(\*) FROM pet GROUP BY species;**

+---------+----------+

| species | COUNT(\*) |

+---------+----------+

| bird | 2 |

| cat | 2 |

| dog | 3 |

| hamster | 1 |

| snake | 1 |

+---------+----------+

每个性别的动物数量。

mysql> **SELECT sex, COUNT(\*) FROM pet GROUP BY sex;**

+------+----------+

| sex | COUNT(\*) |

+------+----------+

| NULL | 1 |

| f | 4 |

| m | 4 |

+------+----------+

在此输出中，NULL表示性别未知）。

每个物种和性别组合的动物数量。

mysql> **SELECT species, sex, COUNT(\*) FROM pet GROUP BY species, sex;**

+---------+------+----------+

| species | sex | COUNT(\*) |

+---------+------+----------+

| bird | NULL | 1 |

| bird | f | 1 |

| cat | f | 1 |

| cat | m | 1 |

| dog | f | 1 |

| dog | m | 2 |

| hamster | f | 1 |

| snake | m | 1 |

+---------+------+----------+

当你使用COUNT()时，你不需要检索整个表。例如，前面的查询，当只对狗和猫进行查询时，看起来像这样。

mysql> **SELECT species, sex, COUNT(\*) FROM pet**

**WHERE species = 'dog' OR species = 'cat'**

**GROUP BY species, sex;**

+---------+------+----------+

| species | sex | COUNT(\*) |

+---------+------+----------+

| cat | f | 1 |

| cat | m | 1 |

| dog | f | 1 |

| dog | m | 2 |

+---------+------+----------+

或者，如果你只想知道性别已知的动物的每一性别的数量。

mysql> **SELECT species, sex, COUNT(\*) FROM pet**

**WHERE sex IS NOT NULL**

**GROUP BY species, sex;**

+---------+------+----------+

| species | sex | COUNT(\*) |

+---------+------+----------+

| bird | f | 1 |

| cat | f | 1 |

| cat | m | 1 |

| dog | f | 1 |

| dog | m | 2 |

| hamster | f | 1 |

| snake | m | 1 |

+---------+------+----------+

如果你在COUNT()值之外还命名了要选择的列，那么应该有一个GROUP BY子句来命名这些相同的列。否则会出现以下情况。

* 如果ONLY\_FULL\_GROUP\_BY SQL模式被启用，会发生错误。

mysql> **SET sql\_mode = 'ONLY\_FULL\_GROUP\_BY';**

Query OK, 0 rows affected (0.00 sec)

mysql> **SELECT owner, COUNT(\*) FROM pet;**

ERROR 1140 (42000): In aggregated query without GROUP BY, expression

#1 of SELECT list contains nonaggregated column 'menagerie.pet.owner';

this is incompatible with sql\_mode=only\_full\_group\_by

* 如果没有启用ONLY\_FULL\_GROUP\_BY，那么在处理查询时，会把所有的记录作为一个单一的组，但是为每个命名的列选择的值是不确定的。服务器可以自由地从任何行中选择值。

mysql> **SET sql\_mode = '';**

Query OK, 0 rows affected (0.00 sec)

mysql> **SELECT owner, COUNT(\*) FROM pet;**

+--------+----------+

| owner | COUNT(\*) |

+--------+----------+

| Harold | 8 |

+--------+----------+

1 row in set (0.00 sec)

参见第12.20.3节，"MySQL对GROUP BY的处理"。关于COUNT(expr)行为和相关优化的信息，请参见第12.20.1节 "聚合函数描述"。

#### 3.3.4.9 使用多于一个表

宠物表记录了你有哪些宠物。如果你想记录关于它们的其他信息，比如它们生活中的事件，如去看兽医或什么时候出生的孩子，你需要另一个表。这个表应该是什么样子的？它需要包含以下信息。

* 宠物的名字，以便你知道每个事件与哪个动物有关。
* 一个日期，以便你知道事件发生的时间。
* 一个描述该事件的字段。
* 一个事件类型字段，如果你希望能够对事件进行分类。

考虑到这些因素，事件表的CREATE TABLE语句可能看起来像这样。

mysql> **CREATE TABLE event (name VARCHAR(20), date DATE,**

**type VARCHAR(15), remark VARCHAR(255));**

与宠物表一样，最简单的方法是通过创建一个包含以下信息的带制表符的文本文件来加载初始记录。

|  |  |  |  |
| --- | --- | --- | --- |
| **name** | **date** | **type** | **remark** |
| **Fluffy** | 1995-05-15 | litter | 4 kittens, 3 female, 1 male |
| **Buffy** | 1993-06-23 | litter | 5 puppies, 2 female, 3 male |
| **Buffy** | 1994-06-19 | litter | 3 puppies, 3 female |
| **Chirpy** | 1999-03-21 | vet | needed beak straightened |
| **Slim** | 1997-08-03 | vet | broken rib |
| **Bowser** | 1991-10-12 | kennel |  |
| **Fang** | 1991-10-12 | kennel |  |
| **Fang** | 1998-08-28 | birthday | Gave him a new chew toy |
| **Claws** | 1998-03-17 | birthday | Gave him a new flea collar |
| **Whistler** | 1998-12-09 | birthday | First birthday |

像这样加载记录。

mysql> **LOAD DATA LOCAL INFILE 'event.txt' INTO TABLE event;**

基于你从对宠物表的查询中学到的东西，你应该能够对事件表中的记录进行检索；原理是一样的。但是，什么时候事件表本身不足以回答你可能提出的问题？

假设你想找出每只宠物产仔的年龄。我们前面看到如何从两个日期计算年龄。母亲的产仔日期在事件表中，但要计算她在该日期的年龄，你需要她的出生日期，该日期存储在宠物表中。这意味着该查询需要两个表。

mysql> **SELECT pet.name,**

**TIMESTAMPDIFF(YEAR,birth,date) AS age,**

**remark**

**FROM pet INNER JOIN event**

**ON pet.name = event.name**

**WHERE event.type = 'litter';**

+--------+------+-----------------------------+

| name | age | remark |

+--------+------+-----------------------------+

| Fluffy | 2 | 4 kittens, 3 female, 1 male |

| Buffy | 4 | 5 puppies, 2 female, 3 male |

| Buffy | 5 | 3 puppies, 3 female |

+--------+------+-----------------------------+

关于这个查询，有几件事需要注意。

* FROM子句连接了两个表，因为该查询需要从这两个表中提取信息。
* 当结合（连接）来自多个表的信息时，你需要指定如何将一个表中的记录与另一个表中的记录进行匹配。这很容易，因为它们都有一个名字列。该查询使用一个ON子句，根据名字的值来匹配两个表中的记录。该查询使用一个INNER JOIN来合并两个表。当且仅当两个表都满足ON子句中指定的条件时，INNER JOIN允许任何一个表的记录出现在结果中。在这个例子中，ON子句指定了宠物表中的名字列必须与事件表中的名字列相匹配。如果一个名字出现在一个表中，但是没有出现在另一个表中，那么该行就不会出现在结果中，因为ON子句中的条件失败了。
* 因为名字列同时出现在两个表中，所以在引用该列时，你必须具体说明是指哪个表。这可以通过在列名前加上表名来实现。

你不需要有两个不同的表来执行一个连接。有时，如果你想把一个表中的记录与同一表中的其他记录进行比较，那么把一个表与它本身连接起来是很有用的。例如，为了在你的宠物中寻找繁殖对，你可以将宠物表与它本身连接起来，以产生同类物种的活体雄性和雌性的候选对。

mysql> **SELECT p1.name, p1.sex, p2.name, p2.sex, p1.species**

**FROM pet AS p1 INNER JOIN pet AS p2**

**ON p1.species = p2.species**

**AND p1.sex = 'f' AND p1.death IS NULL**

**AND p2.sex = 'm' AND p2.death IS NULL;**

+--------+------+-------+------+---------+

| name | sex | name | sex | species |

+--------+------+-------+------+---------+

| Fluffy | f | Claws | m | cat |

| Buffy | f | Fang | m | dog |

+--------+------+-------+------+---------+

在这个查询中，我们指定了表名的别名来引用列，并直接保持每列引用与表的哪个实例相关。

## 3.4 获取关于数据库和表的信息

如果你忘记了一个数据库或表的名称，或者一个给定的表的结构是什么（例如，它的列被称为什么），怎么办？MySQL通过提供有关它支持的数据库和表的信息的几个语句来解决这个问题。

你以前看过SHOW DATABASES，它列出了服务器管理的数据库。要想知道当前选择的是哪个数据库，请使用DATABASE()函数。

mysql> **SELECT DATABASE();**

+------------+

| DATABASE() |

+------------+

| menagerie |

+------------+

如果你还没有选择任何数据库，结果是NULL。

要想知道默认数据库包含哪些表（例如，当你不确定某个表的名称时），请使用这个语句。

mysql> **SHOW TABLES;**

+---------------------+

| Tables\_in\_menagerie |

+---------------------+

| event |

| pet |

+---------------------+

这个语句产生的输出中的列名总是Tables\_in\_db\_name，其中db\_name是数据库的名称。更多信息请参见章节13.7.7.39, "SHOW TABLES 语句"。

如果你想了解一个表的结构，DESCRIBE语句是很有用的；它可以显示一个表的每一列的信息。

mysql> **DESCRIBE pet;**

+---------+-------------+------+-----+---------+-------+

| Field | Type | Null | Key | Default | Extra |

+---------+-------------+------+-----+---------+-------+

| name | varchar(20) | YES | | NULL | |

| owner | varchar(20) | YES | | NULL | |

| species | varchar(20) | YES | | NULL | |

| sex | char(1) | YES | | NULL | |

| birth | date | YES | | NULL | |

| death | date | YES | | NULL | |

+---------+-------------+------+-----+---------+-------+

Field表示列的名称，Type是该列的数据类型，NULL表示该列是否可以包含NULL值，Key表示该列是否有索引，Default指定该列的默认值。Extra显示关于列的特殊信息。如果一个列是用AUTO\_INCREMENT选项创建的，其值是自动递增而不是空的。

DESC是DESCRIBE的缩写形式。更多信息请参见第13.8.1节，"DESCRIBE语句"。

你可以使用SHOW CREATE TABLE语句获得创建一个现有表所需的CREATE TABLE语句。参见第13.7.7.10节，"SHOW CREATE TABLE语句"。

如果你在一个表上有索引，SHOW INDEX FROM tbl\_name可以产生关于它们的信息。关于这个语句的更多信息，请参见第13.7.7.22节，"SHOW INDEX语句"。

## 3.5 在批处理模式下使用mysql

在前面的章节中，你以交互方式使用mysql，输入语句并查看结果。你也可以在批处理模式下运行mysql。要做到这一点，把你想运行的语句放在一个文件中，然后告诉mysql从文件中读取其输入。

shell> **mysql < *batch-file***

如果你在Windows下运行mysql，并且文件中有一些导致问题的特殊字符，你可以这样做。

C:\> **mysql -e "source *batch-file*"**

如果你需要在命令行上指定连接参数，命令可能看起来像这样。

shell> **mysql -h *host* -u *user* -p < *batch-file***

Enter password: **\*\*\*\*\*\*\*\***

当你这样使用mysql时，你正在创建一个脚本文件，然后执行该脚本。

如果你想让脚本继续下去，即使其中的一些语句产生错误，你应该使用--force命令行选项。

为什么要使用脚本？这里有几个原因。

* 如果你重复运行一个查询（例如，每天或每周），把它变成一个脚本，就可以避免每次执行时都重新输入。
* 你可以通过复制和编辑脚本文件，从现有的类似查询中生成新的查询。
* 批量模式在你开发查询时也很有用，特别是对于多行语句或多语句序列。如果你犯了一个错误，你不需要重新输入所有的内容。只要编辑你的脚本以纠正错误，然后告诉mysql再次执行它。
* 如果你有一个产生大量输出的查询，你可以通过一个寻呼机运行输出，而不是看着它从屏幕顶部滚动。

shell> **mysql < *batch-file* | more**

* 你可以在一个文件中捕捉输出，以便进一步处理。

shell> **mysql < *batch-file* > mysql.out**

* 你可以把你的脚本分发给其他人，这样他们也可以运行这些语句。
* 有些情况不允许交互式使用，例如，当你从一个cron job中运行查询时。在这种情况下，你必须使用批处理模式。

当你以批处理模式运行mysql时，默认的输出格式与你交互式使用时不同（更简洁）。例如，当mysql以交互方式运行时，SELECT DISTINCT species FROM pet的输出看起来像这样。

+---------+

| species |

+---------+

| bird |

| cat |

| dog |

| hamster |

| snake |

+---------+

在批处理模式下，输出结果看起来像这样，而不是。

species

bird

cat

dog

hamster

snake

如果你想在批处理模式下获得交互式输出格式，使用mysql -t。要将执行的语句回显到输出，使用mysql -v。

你也可以通过使用source命令或\.命令从mysql提示符中使用脚本。

mysql> **source *filename*;**

mysql> **\. *filename***

更多信息见第4.5.1.5节 "从文本文件执行SQL语句"。

## 3.6 常见查询的例子

下面是如何解决MySQL的一些常见问题的例子。

一些例子使用表shop来保存某些交易者（经销商）的每件物品（物品编号）的价格。假设每个商人对每件物品有一个固定的价格，那么（物品，经销商）就是记录的主键。

启动命令行工具mysql并选择一个数据库。

shell> **mysql *your-database-name***

要创建和填充示例表，使用这些语句。

CREATE TABLE shop (

article INT UNSIGNED DEFAULT '0000' NOT NULL,

dealer CHAR(20) DEFAULT '' NOT NULL,

price DECIMAL(16,2) DEFAULT '0.00' NOT NULL,

PRIMARY KEY(article, dealer));

INSERT INTO shop VALUES

(1,'A',3.45),(1,'B',3.99),(2,'A',10.99),(3,'B',1.45),

(3,'C',1.69),(3,'D',1.25),(4,'D',19.95);

发出这些语句后，表应该有以下内容。

SELECT \* FROM shop ORDER BY article;

+---------+--------+-------+

| article | dealer | price |

+---------+--------+-------+

| 1 | A | 3.45 |

| 1 | B | 3.99 |

| 2 | A | 10.99 |

| 3 | B | 1.45 |

| 3 | C | 1.69 |

| 3 | D | 1.25 |

| 4 | D | 19.95 |

+---------+--------+-------+

### 3.6.1 一列的最高值

"最高的项目编号是多少？"

SELECT MAX(article) AS article FROM shop;

+---------+

| article |

+---------+

| 4 |

+---------+

### 3.6.2 持有某列最大值的行

任务。找到最贵的物品的数量、经销商和价格。

这很容易通过一个子查询完成。

SELECT article, dealer, price

FROM shop

WHERE price=(SELECT MAX(price) FROM shop);

+---------+--------+-------+

| article | dealer | price |

+---------+--------+-------+

| 0004 | D | 19.95 |

+---------+--------+-------+

其他的解决方案是使用LEFT JOIN或按价格降序排序所有的记录，并使用MySQL特有的LIMIT子句只获得第一行。

SELECT s1.article, s1.dealer, s1.price

FROM shop s1

LEFT JOIN shop s2 ON s1.price < s2.price

WHERE s2.article IS NULL;

SELECT article, dealer, price

FROM shop

ORDER BY price DESC

LIMIT 1;

注意

如果有几件最贵的物品，每件的价格都是19.95，那么LIMIT方案将只显示其中一件。

### 3.6.3 每组列的最大值

任务。找到每篇文章的最高价格。

SELECT article, MAX(price) AS price

FROM shop

GROUP BY article

ORDER BY article;

+---------+-------+

| article | price |

+---------+-------+

| 0001 | 3.99 |

| 0002 | 10.99 |

| 0003 | 1.69 |

| 0004 | 19.95 |

+---------+-------+

### 3.6.4 持有某一列的组别最大值的行

任务。对于每件物品，找出价格最贵的一个或多个经销商。

这个问题可以用一个类似这样的子查询来解决。

SELECT article, dealer, price

FROM shop s1

WHERE price=(SELECT MAX(s2.price)

FROM shop s2

WHERE s1.article = s2.article)

ORDER BY article;

+---------+--------+-------+

| article | dealer | price |

+---------+--------+-------+

| 0001 | B | 3.99 |

| 0002 | A | 10.99 |

| 0003 | C | 1.69 |

| 0004 | D | 19.95 |

+---------+--------+-------+

前面的例子使用了一个相关的子查询，这可能是低效的（参见章节13.2.11.7，"相关子查询"）。解决这个问题的其他可能性是在FROM子句中使用一个不相关的子查询，一个LEFT JOIN，或者一个带有窗口函数的公共表表达。

不相关的子查询。

SELECT s1.article, dealer, s1.price

FROM shop s1

JOIN (

SELECT article, MAX(price) AS price

FROM shop

GROUP BY article) AS s2

ON s1.article = s2.article AND s1.price = s2.price

ORDER BY article;

**LEFT JOIN:**

SELECT s1.article, s1.dealer, s1.price

FROM shop s1

LEFT JOIN shop s2 ON s1.article = s2.article AND s1.price < s2.price

WHERE s2.article IS NULL

ORDER BY s1.article;

LEFT JOIN的工作原理是：当s1.price达到最大值时，没有s2.price的值更大，因此对应的s2.article的值为NULL。参见章节13.2.10.2, "JOIN条款"。

带有窗口函数的普通表表达式。

WITH s1 AS (

SELECT article, dealer, price,

RANK() OVER (PARTITION BY article

ORDER BY price DESC

) AS `Rank`

FROM shop

)

SELECT article, dealer, price

FROM s1

WHERE `Rank` = 1

ORDER BY article;

### 3.6.5 使用用户定义的变量

你可以使用MySQL用户变量来记住结果，而不必将它们存储在客户端的临时变量中。参见第9.4节，"用户定义的变量"）。

例如，要找到具有最高和最低价格的文章，你可以这样做。

mysql> **SELECT @min\_price:=MIN(price),@max\_price:=MAX(price) FROM shop;**

mysql> **SELECT \* FROM shop WHERE price=@min\_price OR price=@max\_price;**

+---------+--------+-------+

| article | dealer | price |

+---------+--------+-------+

| 0003 | D | 1.25 |

| 0004 | D | 19.95 |

+---------+--------+-------+

注意

也可以在用户变量中存储数据库对象的名称，如表或列，然后在SQL语句中使用这个变量；但是，这需要使用预处理语句。更多信息请参见章节13.5, "预处理语句"。

### 3.6.6 使用外键

在MySQL中，InnoDB表支持外键约束的检查。参见第15章，InnoDB存储引擎，以及第1.7.2.3节，"FOREIGN KEY约束的差异"。

外键约束并不是仅仅为了连接两个表而需要的。对于InnoDB以外的存储引擎，在定义列的时候可以使用REFERENCES tbl\_name(col\_name)子句，这个子句没有实际作用，只是作为一个备忘或注释，告诉你当前定义的列是为了引用另一个表中的列。在使用这种语法时，认识到以下几点是极其重要的。

* MySQL不执行任何类型的检查以确保col\_name实际存在于tbl\_name中（甚至tbl\_name本身存在）。
* MySQL不对tbl\_name执行任何类型的操作，例如在你定义的表中的行上采取的行动中删除行；换句话说，这个语法没有引起任何ON DELETE或ON UPDATE行为。(尽管你可以在REFERENCES子句中写一个ON DELETE或ON UPDATE子句，但它也会被忽略。)

这个语法创建了一个列；它并没有创建任何类型的索引或键。

你可以使用这样创建的列作为一个连接列，如下面所示。

CREATE TABLE person (

id SMALLINT UNSIGNED NOT NULL AUTO\_INCREMENT,

name CHAR(60) NOT NULL,

PRIMARY KEY (id)

);

CREATE TABLE shirt (

id SMALLINT UNSIGNED NOT NULL AUTO\_INCREMENT,

style ENUM('t-shirt', 'polo', 'dress') NOT NULL,

color ENUM('red', 'blue', 'orange', 'white', 'black') NOT NULL,

owner SMALLINT UNSIGNED NOT NULL REFERENCES person(id),

PRIMARY KEY (id)

);

INSERT INTO person VALUES (NULL, 'Antonio Paz');

SELECT @last := LAST\_INSERT\_ID();

INSERT INTO shirt VALUES

(NULL, 'polo', 'blue', @last),

(NULL, 'dress', 'white', @last),

(NULL, 't-shirt', 'blue', @last);

INSERT INTO person VALUES (NULL, 'Lilliana Angelovska');

SELECT @last := LAST\_INSERT\_ID();

INSERT INTO shirt VALUES

(NULL, 'dress', 'orange', @last),

(NULL, 'polo', 'red', @last),

(NULL, 'dress', 'blue', @last),

(NULL, 't-shirt', 'white', @last);

SELECT \* FROM person;

+----+---------------------+

| id | name |

+----+---------------------+

| 1 | Antonio Paz |

| 2 | Lilliana Angelovska |

+----+---------------------+

SELECT \* FROM shirt;

+----+---------+--------+-------+

| id | style | color | owner |

+----+---------+--------+-------+

| 1 | polo | blue | 1 |

| 2 | dress | white | 1 |

| 3 | t-shirt | blue | 1 |

| 4 | dress | orange | 2 |

| 5 | polo | red | 2 |

| 6 | dress | blue | 2 |

| 7 | t-shirt | white | 2 |

+----+---------+--------+-------+

SELECT s.\* FROM person p INNER JOIN shirt s

ON s.owner = p.id

WHERE p.name LIKE 'Lilliana%'

AND s.color <> 'white';

+----+-------+--------+-------+

| id | style | color | owner |

+----+-------+--------+-------+

| 4 | dress | orange | 2 |

| 5 | polo | red | 2 |

| 6 | dress | blue | 2 |

+----+-------+--------+-------+

当以这种方式使用时，REFERENCES子句不会显示在SHOW CREATE TABLE或DESCRIBE的输出中。

SHOW CREATE TABLE shirt\G

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 1. row \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Table: shirt

Create Table: CREATE TABLE `shirt` (

`id` smallint(5) unsigned NOT NULL auto\_increment,

`style` enum('t-shirt','polo','dress') NOT NULL,

`color` enum('red','blue','orange','white','black') NOT NULL,

`owner` smallint(5) unsigned NOT NULL,

PRIMARY KEY (`id`)

) ENGINE=MyISAM DEFAULT CHARSET=utf8mb4

以这种方式使用REFERENCES作为列定义中的注释或 "提醒"，对MyISAM表是有效的。

### 3.6.7 在两个键上进行搜索

使用单键的OR是很好的优化，和AND的处理一样。

一个棘手的情况是，在两个不同的键上搜索，并结合OR。

SELECT field1\_index, field2\_index FROM test\_table

WHERE field1\_index = '1' OR field2\_index = '1'

。

这种情况被优化。参见章节8.2.1.3, "索引合并优化"。

你也可以通过使用合并两个独立的SELECT语句的输出的UNION来有效地解决这个问题。参见章节13.2.10.3, "UNION子句"。

每个SELECT只搜索一个键，可以进行优化。

SELECT field1\_index, field2\_index

FROM test\_table WHERE field1\_index = '1'

UNION

SELECT field1\_index, field2\_index

FROM test\_table WHERE field2\_index = '1';

### 3.6.8 计算每天的访问量

下面的例子显示了如何使用位组函数来计算用户每月访问一个网页的天数。

CREATE TABLE t1 (year YEAR, month INT UNSIGNED,

day INT UNSIGNED);

INSERT INTO t1 VALUES(2000,1,1),(2000,1,20),(2000,1,30),(2000,2,2),

(2000,2,23),(2000,2,23);

该示例表包含年-月-日的数值，代表用户对该页面的访问。为了确定这些访问在每个月有多少个不同的日子发生，请使用这个查询。

SELECT year,month,BIT\_COUNT(BIT\_OR(1<<day)) AS days FROM t1

GROUP BY year,month;

这将返回。

+------+-------+------+

| year | month | days |

+------+-------+------+

| 2000 | 1 | 3 |

| 2000 | 2 | 2 |

+------+-------+------+

该查询计算出每个年/月的组合有多少个不同的日子出现在表中，并自动删除重复的条目。

### 3.6.9 使用AUTO\_INCREMENT

AUTO\_INCREMENT属性可以用来为新的记录生成一个唯一的标识。

CREATE TABLE animals (

id MEDIUMINT NOT NULL AUTO\_INCREMENT,

name CHAR(30) NOT NULL,

PRIMARY KEY (id)

);

INSERT INTO animals (name) VALUES

('dog'),('cat'),('penguin'),

('lax'),('whale'),('ostrich');

SELECT \* FROM animals;

SELECT \* FROM animals;

其中返回。

+----+---------+

| id | name |

+----+---------+

| 1 | dog |

| 2 | cat |

| 3 | penguin |

| 4 | lax |

| 5 | whale |

| 6 | ostrich |

+----+---------+

没有为AUTO\_INCREMENT列指定值，所以MySQL自动分配序列号。你也可以显式地给该列分配0来生成序列号，除非启用NO\_AUTO\_VALUE\_ON\_ZERO SQL模式。例如

INSERT INTO animals (id,name) VALUES(0,'groundhog');

如果列被声明为NOT NULL，也可以将NULL赋值给该列，以生成序列号。例如。

INSERT INTO animals (id,name) VALUES(NULL,'squirrel');

当你在AUTO\_INCREMENT列中插入任何其他值时，该列被设置为该值，并且序列被重置，以便下一个自动生成的值从最大的列值开始依次出现。例如。

**INSERT INTO animals (id,name) VALUES(100,'rabbit');**

**INSERT INTO animals (id,name) VALUES(NULL,'mouse');**

**SELECT \* FROM animals;**

+-----+-----------+

| id | name |

+-----+-----------+

| 1 | dog |

| 2 | cat |

| 3 | penguin |

| 4 | lax |

| 5 | whale |

| 6 | ostrich |

| 7 | groundhog |

| 8 | squirrel |

| 100 | rabbit |

| 101 | mouse |

+-----+-----------+

更新现有的AUTO\_INCREMENT列值也会重置AUTO\_INCREMENT序列。

你可以用LAST\_INSERT\_ID()SQL函数或mysql\_insert\_id()C API函数检索最近的自动生成的AUTO\_INCREMENT值。这些函数是特定于连接的，所以它们的返回值不会受到另一个也在执行插入的连接的影响。

为AUTO\_INCREMENT列使用最小的整数数据类型，其大小足以容纳你需要的最大序列值。当该列达到数据类型的上限时，下一次生成序列号的尝试会失败。如果可能的话，使用UNSIGNED属性以允许更大的范围。例如，如果你使用TINYINT，允许的最大序列号是127。对于TINYINT UNSIGNED，最大是255。关于所有整数类型的范围，请参阅第11.1.2节 "整数类型（精确值）--INTEGER、INT、SMALLINT、TINYINT、MEDIUMINT、BIGINT"。

注意

对于多行插入，LAST\_INSERT\_ID()和mysql\_insert\_id()实际上返回插入行中第一行的AUTO\_INCREMENT键。这使得多行插入可以在复制设置中的其他服务器上正确重现。

要以1以外的AUTO\_INCREMENT值开始，用CREATE TABLE或ALTER TABLE设置该值，像这样。

mysql> **ALTER TABLE tbl AUTO\_INCREMENT = 100;**

**InnoDB注意事项**

关于InnoDB特定的AUTO\_INCREMENT用法的信息，请参见15.6.1.6节 "InnoDB中的AUTO\_INCREMENT处理"。

**MyISAM注意事项**

对于MyISAM表，你可以在一个多列索引中的二级列上指定AUTO\_INCREMENT。在这种情况下，AUTO\_INCREMENT列的生成值被计算为MAX(auto\_increment\_column) + 1 WHERE prefix=given-preix。当你想把数据放入有序的组时，这很有用。

CREATE TABLE animals (

grp ENUM('fish','mammal','bird') NOT NULL,

id MEDIUMINT NOT NULL AUTO\_INCREMENT,

name CHAR(30) NOT NULL,

PRIMARY KEY (grp,id)

) ENGINE=MyISAM;

INSERT INTO animals (grp,name) VALUES

('mammal','dog'),('mammal','cat'),

('bird','penguin'),('fish','lax'),('mammal','whale'),

('bird','ostrich');

SELECT \* FROM animals ORDER BY grp,id;

其中返回。

+--------+----+---------+

| grp | id | name |

+--------+----+---------+

| fish | 1 | lax |

| mammal | 1 | dog |

| mammal | 2 | cat |

| mammal | 3 | whale |

| bird | 1 | penguin |

| bird | 2 | ostrich |

+--------+----+---------+

在这种情况下（当AUTO\_INCREMENT列是多列索引的一部分时），如果你删除任何一组中具有最大AUTO\_INCREMENT值的行，AUTO\_INCREMENT值会被重新使用。这种情况甚至发生在MyISAM表中，对于MyISAM表来说，AUTO\_INCREMENT值通常不会被重复使用。

如果AUTO\_INCREMENT列是多个索引的一部分，MySQL使用以AUTO\_INCREMENT列开始的索引生成序列值，如果有一个的话。例如，如果动物表包含索引PRIMARY KEY（grp，id）和INDEX（id），MySQL在生成序列值时将忽略PRIMARY KEY。结果是，该表将包含一个单一的序列，而不是每个grp值的序列。

**进一步阅读**

关于AUTO\_INCREMENT的更多信息可以在这里找到。

* 如何为一个列分配AUTO\_INCREMENT属性。第13.1.20节，"CREATE TABLE语句"，以及第13.1.9节，"ALTER TABLE语句"。
* 在NO\_AUTO\_VALUE\_ON\_ZERO SQL模式下，AUTO\_INCREMENT是如何表现的。第5.1.11节，"服务器SQL模式"。
* 如何使用LAST\_INSERT\_ID()函数来查找包含最新AUTO\_INCREMENT值的记录。第12.16节，"信息函数"。
* 设置要使用的AUTO\_INCREMENT值。第5.1.8节，"服务器系统变量"。
* 第15.6.1.6节，"InnoDB中的AUTO\_INCREMENT处理"
* AUTO\_INCREMENT和复制。第17.5.1.1节，"复制和AUTO\_INCREMENT"。
* 与AUTO\_INCREMENT有关的服务器系统变量（auto\_increment\_increment和auto\_increment\_offset），可以用于复制。第5.1.8节，"服务器系统变量"

## 3.7 在Apache中使用MySQL

有一些程序可以让你从MySQL数据库中验证你的用户，也可以让你把你的日志文件写到MySQL表中。

你可以改变Apache的日志格式，使其容易被MySQL读取，方法是将以下内容放入Apache的配置文件中。

LogFormat \

"\"%h\",%{%Y%m%d%H%M%S}t,%>s,\"%b\",\"%{Content-Type}o\", \

\"%U\",\"%{Referer}i\",\"%{User-Agent}i\""

要把这种格式的日志文件加载到MySQL中，你可以使用类似这样的语句。

LOAD DATA INFILE '***/local/access\_log***' INTO TABLE ***tbl\_name***

FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '"' ESCAPED BY '\\'

命名的表应该被创建为具有与LogFormat行写到日志文件中的列相对应的列。

# MySQL程序

# MySQL服务管理

# 安全

# 备份与恢复

# 优化

# 语言结构

# 字符集、对齐方式、统一编码

# 数据类型

# 函数和operator

# SQL表达式

的

# MySQL数据目录

MySQL服务器集成了一个事务型数据字典，用于存储有关数据库对象的信息。在以前的MySQL版本中，字典数据存储在元数据文件、非事务性表和特定存储引擎的数据字典中。

本章描述了数据字典的主要特征、好处、使用差异和限制。关于数据字典功能的其他含义，请参阅MySQL 8.0发布说明中的 "数据字典说明 "部分。

MySQL数据字典的好处包括

* 集中式数据字典模式的简单性，统一存储字典数据。见第14.1节，"数据字典模式"。
* 消除了基于文件的元数据存储。参见第 14.2 节，"去除基于文件的元数据存储"。
* 对字典数据进行事务性的、防崩溃的存储。参见第 14.3 节，"字典数据的事务性存储"。
* 统一的、集中的字典对象的缓存。参见第 14.4 节，"字典对象的缓存"。
* 对一些INFORMATION\_SCHEMA表进行了更简单和改进的实现。参见第 14.5 节 "INFORMATION\_SCHEMA 和数据字典集成"。
* 原子型DDL。见第13.1.1节，"原子数据定义语句支持"。

重要的是

*与没有数据字典的服务器相比，启用了数据字典的服务器需要一些一般的操作差异；见第14.7节，"数据字典使用差异"。此外，对于升级到MySQL 8.0，升级过程与以前的MySQL版本有些不同，需要你通过检查特定的先决条件来验证你的安装是否准备好升级。欲了解更多信息，请参阅第2.11节 "升级MySQL"，特别是第2.11.5节 "为升级准备你的安装"。*

## 14.1 数据字典模式

数据字典表是受保护的，只能在MySQL的调试构建中访问。然而，MySQL支持通过INFORMATION\_SCHEMA表和SHOW语句访问存储在数据字典表中的数据。关于构成数据字典的表的概述，见数据字典表。

MySQL系统表在MySQL 8.0中仍然存在，可以通过在mysql系统数据库上发出SHOW TABLES语句来查看。一般来说，MySQL数据字典表和系统表之间的区别是，数据字典表包含执行SQL查询所需的元数据，而系统表包含辅助数据，如时区和帮助信息。MySQL系统表和数据字典表在升级方式上也有所不同。MySQL服务器负责管理数据字典的升级。SQL服务器。见数据字典如何升级。升级MySQL系统表需要运行完整的MySQL升级程序。参见第2.11.3节，"MySQL升级过程升级的内容"。

**数据字典是如何升级的**

新版本的MySQL可能包括对数据字典表定义的更改。这种变化存在于新安装的MySQL版本中，但当执行MySQL二进制文件的就地升级时，当使用新的二进制文件重新启动MySQL服务器时，就会应用这些变化。在启动时，服务器的数据字典版本与存储在数据字典中的版本信息进行比较，以确定是否应该升级数据字典表。如果升级是必要的并且被支持，服务器会用更新的定义创建数据字典表，把持久化的元数据复制到新的表中，用新的表原子化地替换旧的表，并重新初始化数据字典。如果不需要升级，则继续启动而不更新数据字典表。

数据字典表的升级是一个原子操作，这意味着所有的数据字典表都会根据需要进行升级，否则操作会失败。如果升级操作失败，服务器启动时就会出现错误。在这种情况下，可以使用旧的服务器二进制文件和旧的数据目录来启动服务器。当再次使用新的服务器二进制文件来启动服务器时，就会重新尝试进行数据字典的升级。

一般来说，在数据字典表成功升级后，不可能用旧的服务器二进制文件重新启动服务器。因此，在数据字典表被升级后，不支持将MySQL服务器二进制文件降级到以前的MySQL版本。

mysqld --no-dd-upgrade选项可以用来阻止启动时自动升级数据字典表。当--no-dd-upgrade被指定时，如果服务器发现服务器的数据字典版本与存储在数据字典中的版本不同，则启动失败，出现一个错误，说明禁止数据字典升级。

**使用MySQL的调试版本查看数据字典表**

数据字典表默认是受保护的，但可以通过编译支持调试的MySQL（使用-DWITH\_DEBUG=1 CMake选项）并指定+d,skip\_dd\_table\_access\_check调试选项和修改器来访问。关于编译调试构建的信息，见第5.9.1.1节，"编译调试的MySQL"。

警告

*不建议直接修改或写入数据字典表，这可能使你的MySQL实例无法运行。*

在用调试支持编译MySQL后，使用这个SET语句使数据字典表对mysql客户会话可见。

mysql> **SET SESSION debug='+d,skip\_dd\_table\_access\_check';**

使用这个查询来检索数据字典表的列表。

mysql> **SELECT name, schema\_id, hidden, type FROM mysql.tables where schema\_id=1 AND hidden='System';**

使用SHOW CREATE TABLE来查看数据字典表的定义。例如

mysql> **SHOW CREATE TABLE mysql.catalogs\G**

## 14.2 移除基于文件的元数据存储

在以前的MySQL版本中，字典数据被部分地存储在元数据文件中。基于文件的元数据存储的问题包括昂贵的文件扫描、易受文件系统相关错误的影响、处理复制和崩溃恢复故障状态的复杂代码，以及缺乏可扩展性，难以为新功能和关系对象添加元数据。

下面列出的元数据文件已从MySQL中删除。除非另有说明，以前存储在元数据文件中的数据现在存储在数据字典表中。

* .frm文件。表元数据文件。随着.frm文件的删除。
  + frm文件结构所规定的64KB的表定义大小限制被取消。
  + INFORMATION\_SCHEMA.TABLES VERSION列报告一个硬编码值10，这是MySQL 5.7中使用的最后一个.frm文件版本。
* .par文件。分区定义文件。InnoDB在MySQL 5.7中停止使用分区定义文件，因为引入了对InnoDB表的本地分区支持。
* .TRN文件。触发器命名空间文件。
* .TRG文件。触发器参数文件。
* .isl文件。InnoDB符号链接文件，包含了在数据目录之外创建的每个表的文件空间的位置。
* db.opt文件。数据库配置文件。这些文件，每个数据库目录一个，包含数据库默认的字符集属性。
* ddl\_log.log文件。该文件包含由数据定义语句产生的元数据操作记录，例如DROP TABLE和ALTER TABLE。

## 14.3 词典数据的事务性存储

数据字典模式在事务性（InnoDB）表中存储字典数据。数据字典表与非数据字典系统表一起位于mysql数据库中。

数据字典表在一个名为mysql.ibd的单一InnoDB表空间中创建，该表空间位于MySQL数据目录中。mysql.ibd表空间文件必须驻留在MySQL数据目录中，其名称不能被修改或被其他表空间使用。

字典数据受到与保护存储在InnoDB表中的用户数据相同的提交、回滚和崩溃恢复能力的保护。

## 14.4 字典对象缓存

字典对象缓存是一个共享的全局缓存，它将以前访问过的数据字典对象存储在内存中，以实现对象的重用，并使磁盘I/O最小化。与MySQL使用的其他缓存机制类似，字典对象缓存使用基于LRU的驱逐策略，从内存中驱逐最近使用的最小对象。

字典对象缓存包括存储不同对象类型的缓存分区。一些缓存分区的大小限制是可配置的，而另一些是硬编码的。

* 表空间定义高速缓存分区。存储表空间定义对象。tablespace\_definition\_cache选项为可以存储在字典对象缓存中的表空间定义对象的数量设置了一个限制。默认值是256。
* 模式定义缓存分区。存储模式定义对象。schema\_definition\_cache选项为可以存储在字典对象缓存中的模式定义对象的数量设置了一个限制。默认值是256。
* 表定义缓存分区。存储表定义对象。对象的限制被设置为max\_connections的值，它的默认值是151。

表定义缓存分区与使用table\_definition\_cache配置选项配置的表定义缓存平行存在。两个缓存都存储表定义，但为MySQL服务器的不同部分服务。一个缓存中的对象不依赖于另一个缓存中对象的存在。

* 存储程序定义缓存分区。存储存储的程序定义对象。stored\_program\_definition\_cache选项为可以存储在字典对象缓存中的存储程序定义对象的数量设定了一个限制。默认值是256。

存储程序定义缓存分区与存储过程和存储函数缓存并行存在，后者是通过store\_program\_cache选项配置的。

stored\_program\_cache选项为每个连接的存储过程或函数的缓存数量设置了一个软上限，并且在每次连接执行存储过程或函数时检查这个上限。另一方面，存储程序定义缓存分区是一个共享的缓存，用于存储其他用途的存储程序定义对象。存储程序定义缓存分区中对象的存在与存储过程缓存或存储函数缓存中对象的存在没有关系，反之亦然。

字符集定义缓存分区。存储字符集定义对象，硬编码的对象限制为256。

* 整理定义缓存分区。存储整理定义对象，硬编码的对象限制为256。

关于字典对象缓存配置选项的有效值，请参阅第5.1.8节，"服务器系统变量"。

## 14.5 INFORMATION\_SCHEMA和数据字典集成

随着数据字典的引入，以下INFORMATION\_SCHEMA表被实现为数据字典表的视图。

* [**CHARACTER\_SETS**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\information-schema.html#information-schema-character-sets-table)
* [**CHECK\_CONSTRAINTS**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\information-schema.html#information-schema-check-constraints-table)
* [**COLLATIONS**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\information-schema.html#information-schema-collations-table)
* [**COLLATION\_CHARACTER\_SET\_APPLICABILITY**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\information-schema.html#information-schema-collation-character-set-applicability-table)
* [**COLUMNS**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\information-schema.html#information-schema-columns-table)
* [**COLUMN\_STATISTICS**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\information-schema.html#information-schema-column-statistics-table)
* [**EVENTS**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\information-schema.html#information-schema-events-table)
* [**FILES**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\information-schema.html#information-schema-files-table)
* [**INNODB\_COLUMNS**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\information-schema.html#information-schema-innodb-columns-table)
* [**INNODB\_DATAFILES**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\information-schema.html#information-schema-innodb-datafiles-table)
* [**INNODB\_FIELDS**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\information-schema.html#information-schema-innodb-fields-table)
* [**INNODB\_FOREIGN**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\information-schema.html#information-schema-innodb-foreign-table)
* [**INNODB\_FOREIGN\_COLS**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\information-schema.html#information-schema-innodb-foreign-cols-table)
* [**INNODB\_INDEXES**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\information-schema.html#information-schema-innodb-indexes-table)
* [**INNODB\_TABLES**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\information-schema.html#information-schema-innodb-tables-table)
* [**INNODB\_TABLESPACES**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\information-schema.html#information-schema-innodb-tablespaces-table)
* [**INNODB\_TABLESPACES\_BRIEF**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\information-schema.html#information-schema-innodb-tablespaces-brief-table)
* [**INNODB\_TABLESTATS**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\information-schema.html#information-schema-innodb-tablestats-table)
* [**KEY\_COLUMN\_USAGE**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\information-schema.html#information-schema-key-column-usage-table)
* **KEYWORDS**
* [**PARAMETERS**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\information-schema.html#information-schema-parameters-table)
* [**PARTITIONS**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\information-schema.html#information-schema-partitions-table)
* [**REFERENTIAL\_CONSTRAINTS**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\information-schema.html#information-schema-referential-constraints-table)
* [**RESOURCE\_GROUPS**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\information-schema.html#information-schema-resource-groups-table)
* [**ROUTINES**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\information-schema.html#information-schema-routines-table)
* [**SCHEMATA**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\information-schema.html#information-schema-schemata-table)
* [**STATISTICS**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\information-schema.html#information-schema-statistics-table)
* [**ST\_GEOMETRY\_COLUMNS**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\information-schema.html#information-schema-st-geometry-columns-table)
* [**ST\_SPATIAL\_REFERENCE\_SYSTEMS**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\information-schema.html#information-schema-st-spatial-reference-systems-table)
* [**TABLES**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\information-schema.html#information-schema-tables-table)
* [**TABLE\_CONSTRAINTS**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\information-schema.html#information-schema-table-constraints-table)
* [**TRIGGERS**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\information-schema.html#information-schema-triggers-table)
* [**VIEWS**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\information-schema.html#information-schema-views-table)
* [**VIEW\_ROUTINE\_USAGE**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\information-schema.html#information-schema-view-routine-usage-table)
* [**VIEW\_TABLE\_USAGE**](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\information-schema.html#information-schema-view-table-usage-table)

对这些表的查询现在更有效率了，因为他们从数据字典表获得信息，而不是通过其他更慢的方式。特别是对于每个INFORMATION\_SCHEMA表，它是数据字典表上的一个视图。

* 服务器不再必须为INFORMATION\_SCHEMA表的每次查询创建一个临时表。
* 当底层数据字典表存储了以前通过目录扫描（例如，列举数据库名称或数据库内的表名称）或文件打开操作（例如，从.frm文件中读取信息）获得的值时，INFORMATION\_SCHEMA查询这些值现在使用表查找。(另外，即使对于非视图的INFORMATION\_SCHEMA表，数据库和表名等值也是通过数据字典的查找来获取的，不需要目录或文件扫描。)
* 在底层数据字典表上的索引允许优化器构建有效的查询执行计划，这在以前的实现中是不存在的，它是通过每个查询使用一个临时表来处理 INFORMATION\_SCHEMA 表。

前面的改进也适用于显示对应于INFORMATION\_SCHEMA表的信息的SHOW语句，这些表是数据字典表的视图。例如，SHOW DATABASES显示与SCHEMATA表相同的信息。

除了在数据字典表上引入视图之外，在STATISTICS和TABLES表中包含的表的统计数据现在被缓存起来，以提高INFORMATION\_SCHEMA的查询性能。information\_schema\_stats\_expiry系统变量定义了缓存的表统计信息过期前的时间段。默认是86400秒（24小时）。如果没有缓存的统计数据或者统计数据已经过期，那么在查询表的统计列时，会从存储引擎中检索统计数据。要在任何时候更新某个表的缓存值，请使用ANALYZE TABLE

information\_schema\_stats\_expiry可以被设置为0，让INFORMATION\_SCHEMA查询直接从存储引擎检索最新的统计数据，这不如检索缓存的统计数据快。

更多信息请参见第8.2.3节，"优化INFORMATION\_SCHEMA查询"。

在MySQL 8.0中，INFORMATION\_SCHEMA表与数据字典紧密联系在一起，导致了一些使用上的差异。见第14.7节，"数据字典使用差异"。

## 14.6 串行化字典信息（SDI

除了在数据字典中存储关于数据库对象的元数据外，MySQL还以序列化的形式存储它。这种数据被称为序列化字典信息（SDI）。InnoDB在其表空间文件中存储SDI数据。NDBCLUSTER将SDI数据存储在NDB字典中。其他存储引擎将SDI数据存储在.sdi文件中，这些文件是在表的数据库目录下为一个给定的表创建的。SDI数据以一种紧凑的JSON格式生成。

串行化的字典信息（SDI）存在于所有InnoDB表空间文件中，除了临时表空间和undo tablespaces文件。InnoDB表空间文件中的SDI记录只描述表空间中包含的表和表空间对象。

SDI数据通过对表的DDL操作或CHECK TABLE FOR UPGRADE进行更新。当MySQL服务器升级到一个新版本或版本时，SDI数据不会被更新。

SDI数据的存在提供了元数据的冗余性。例如，如果数据字典变得不可用，可以使用ibd2sdi工具直接从InnoDB表空间文件中提取对象元数据。

对于InnoDB，一个SDI记录需要一个索引页，默认大小为16KB。然而，SDI数据被压缩以减少存储空间的占用。

对于由多个表空间组成的分区InnoDB表，SDI数据被存储在第一个分区的表空间文件中。

MySQL服务器使用一个内部API，在DDL操作期间访问该API以创建和维护SDI记录。

IMPORT TABLE语句根据.sdi文件中包含的信息导入MyISAM表。更多信息，请参见第13.2.5节，"IMPORT TABLE语句"。

## 14.7 数据字典使用方面的差异

与没有数据字典的服务器相比，使用启用了数据字典的MySQL服务器会带来一些操作上的差异。

* 以前，启用innodb\_read\_only系统变量可以阻止只为InnoDB存储引擎创建和删除表。从MySQL 8.0开始，启用innodb\_read\_only会阻止所有存储引擎的这些操作。任何存储引擎的表创建和删除操作都会修改mysql系统数据库中的数据字典表，但这些表使用InnoDB存储引擎，在启用innodb\_read\_only时不能被修改。这个原则也适用于其他需要修改数据字典表的表操作。例子。
  + ANALYZE TABLE失败，因为它更新了存储在数据字典中的表的统计数据。
  + ALTER TABLE tbl\_name ENGINE=engine\_name失败，因为它更新了存储引擎的名称，而这个名称是存储在数据字典中的。

注意

*启用innodb\_read\_only对mysql系统数据库中的非数据字典表也有重要影响。详情请参见第15.14节 "InnoDB启动选项和系统变量 "中对innodb\_read\_only的描述。*

* 以前，mysql系统数据库中的表对于DML和DDL语句是可见的。从MySQL 8.0开始，数据字典表是不可见的，不能被直接修改或查询。然而，在大多数情况下，有相应的INFORMATION\_SCHEMA表可以被查询。这使得底层的数据字典表可以随着服务器开发的进行而改变，同时保持一个稳定的INFORMATION\_SCHEMA接口供应用程序使用。
* 在MySQL 8.0中，INFORMATION\_SCHEMA表与数据字典紧密相连，导致了一些使用上的差异。
  + 以前，INFORMATION\_SCHEMA查询STATISTICS和TABLES表中的表的统计数据，直接从存储引擎检索统计数据。从MySQL 8.0开始，默认使用缓存的表统计数据。information\_schema\_stats\_expiry系统变量定义了缓存表统计数据过期前的时间段。默认是86400秒（24小时）。要想在任何时候更新某个表的缓存值，请使用ANALYZE TABLE）。如果没有缓存的统计信息或者统计信息已经过期，在查询表的统计列时，会从存储引擎中检索统计信息。要想总是直接从存储引擎中获取最新的统计数据，请将information\_schema\_stats\_expiry设置为0。 更多信息请参见8.2.3节 "优化INFORMATION\_SCHEMA查询"。
  + 几个INFORMATION\_SCHEMA表是数据字典表的视图，这使得优化器可以在这些底层表上使用索引。因此，根据优化器的选择，INFORMATION\_SCHEMA查询结果的行顺序可能与之前的结果不同。如果一个查询结果必须有特定的行排序特征，请包括一个ORDER BY子句。
  + 对INFORMATION\_SCHEMA表的查询可能会返回与早期MySQL系列不同的字母大写的列名。应用程序应以不区分大小写的方式测试结果集列名。如果这不可行，一个变通方法是在选择列表中使用列别名，以要求的字母大小写返回列名。比如说。

SELECT TABLE\_SCHEMA AS table\_schema, TABLE\_NAME AS table\_name

FROM INFORMATION\_SCHEMA.TABLES WHERE TABLE\_NAME = 'users';

* mysqldump和mysqlpump不再转储INFORMATION\_SCHEMA数据库，即使在命令行中明确命名。
* CREATE TABLE dst\_tbl LIKE src\_tbl要求src\_tbl是一个基表，如果它是一个INFORMATION\_SCHEMA表，是数据字典表的视图，则失败。
* 以前，从INFORMATION\_SCHEMA表中选择的列的结果集标题使用查询中指定的大写字母。这个查询产生的结果集的标题是table\_name。

SELECT table\_name FROM INFORMATION\_SCHEMA.TABLES;

从MySQL 8.0开始，这些头是大写的；前面的查询产生一个头为TABLE\_NAME的结果集。如果有必要，可以用一个列的别名来实现不同的字母大小写。例如。

SELECT table\_name AS 'table\_name' FROM INFORMATION\_SCHEMA.TABLES;

* 数据目录会影响mysqldump和mysqlpump从mysql系统数据库转储信息的方式。
* 以前，可以转储mysql系统数据库中的所有表。从MySQL 8.0开始，mysqldump和mysqlpump只转储该数据库中的非数据字典表。
  + 以前，可以转储mysql系统数据库中的所有表。从MySQL 8.0开始，mysqldump和mysqlpump只转储该数据库中的非数据字典表。
  + 以前，当使用--all-databases选项时，不需要--routines和--events选项来包括存储例程和事件。转储包括mysql系统数据库，因此也包括含有存储例程和事件定义的proc和事件表。从MySQL 8.0开始，事件表和proc表不被使用。相应对象的定义被存储在数据字典表中，但这些表没有被转储。要在使用--all-databases进行的转储中包括存储的例程和事件，请明确使用--routines和--events选项。
  + 以前， --routines选项需要proc表的SELECT权限。从MySQL 8.0开始，该表不被使用；--routines需要全局SELECT权限来代替。
  + 以前，可以通过转储proc和event表来转储存储的例程和事件定义，以及它们的创建和修改时间戳。从MySQL 8.0开始，这些表不被使用，所以不可能转储时间戳。
* 以前，创建一个包含非法字符的存储例程会产生一个警告。从MySQL 8.0开始，这就是一个错误。

## 14.8 数据字典的限制

本节介绍了用MySQL数据字典引入的临时限制。

* 不支持在数据目录下手动创建数据库目录（例如，用mkdir）。手动创建的数据库目录不被MySQL服务器所识别。
* 由于向存储、Undo logs和redo log而不是.frm文件写入，DDL操作需要更长的时间。

# InnoDB存储引擎

# 其他存储引擎

# 复制

# 分组复制

# MySQL SHELL

MySQL Shell是MySQL服务器的一个高级客户端和代码编辑器。除了提供与mysql类似的SQL功能外，MySQL Shell还提供了JavaScript和Python的脚本功能，并包括与MySQL一起工作的API。MySQL Shell是一个组件，你可以单独安装。

下面的讨论简要地描述了MySQL Shell的功能。欲了解更多信息，请参阅MySQL Shell手册，可在<https://dev.mysql.com/doc/mysql-shell/en/>

MySQL Shell包括以下用JavaScript和Python实现的API，你可以用它们来开发与MySQL交互的代码。

* 当MySQL Shell使用X协议连接到MySQL服务器时，X DevAPI使开发者能够处理关系型数据和文档型数据。这使你能够将MySQL用作文档存储，有时被称为 "使用NoSQL"。欲了解更多信息，请参见第20章，将MySQL用作文档存储。关于在MySQL Shell中实现的X DevAPI的概念和用法的文档，请参见X DevAPI用户指南。
* AdminAPI使数据库管理员能够与InnoDB Cluster一起工作，它为使用基于InnoDB的MySQL数据库的高可用性和可扩展性提供了一个综合解决方案，而不需要高级MySQL专业知识。AdminAPI还包括对InnoDB ReplicaSet的支持，它使您能够以与InnoDB Cluster类似的方式管理一组运行基于GTID的异步复制的MySQL实例。此外，AdminAPI使MySQL Router的管理更容易，包括与InnoDB Cluster和InnoDB ReplicaSet的整合。参见使用MySQL AdminAPI。

MySQL Shell有两个版本：社区版和商业版。社区版是免费提供的。商业版以低廉的价格提供额外的企业功能。

# MySQL文档存储

# InnoDB 集群

本章介绍了MySQL InnoDB Cluster，它结合了MySQL技术，使你能够为MySQL部署和管理一个完整的集成高可用性解决方案。本内容是InnoDB Cluster的高层次概述，完整的文档请参见MySQL InnoDB Cluster。

重要提示

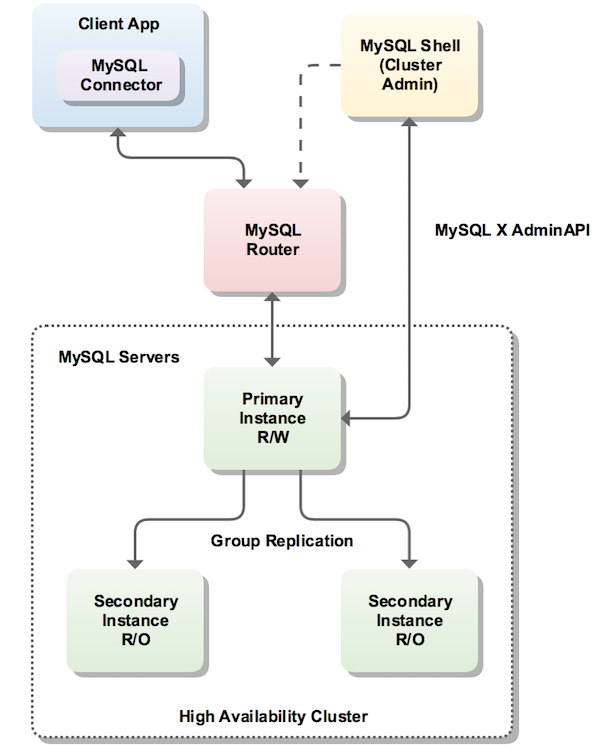
*InnoDB Cluster不提供对MySQL NDB Cluster的支持。关于MySQL NDB Cluster的更多信息，请参见第23章，MySQL NDB Cluster 8.0和第23.1.6节，"使用InnoDB的MySQL服务器与NDB Cluster相比*"。

一个InnoDB Cluster由至少三个MySQL Server实例组成，它提供了高可用性和扩展功能。InnoDB Cluster使用以下MySQL技术。

* MySQL Shell，它是MySQL的一个高级客户端和代码编辑器。
* MySQL服务器，以及组复制，它使一组MySQL实例能够提供高可用性。InnoDB Cluster提供了一种替代性的、易于使用的编程方式来与Group Replication一起工作。
* MySQL Router，一个轻量级的中间件，在你的应用程序和InnoDB Cluster之间提供透明的路由。

下图显示了这些技术如何协同工作的概况。

**Figure 21.1 InnoDB Cluster overview**



作为建立在MySQL组复制上的，提供了诸如自动成员管理、容错、自动故障转移等功能。一个InnoDB集群通常以单主模式运行，有一个主实例（读写）和多个辅助实例（只读）。高级用户也可以利用多主模式，即所有实例都是主实例。你甚至可以在InnoDB Cluster在线时改变集群的拓扑结构，以确保尽可能高的可用性。

你可以使用作为MySQL Shell一部分提供的AdminAPI与InnoDB Cluster一起工作。AdminAPI在JavaScript和Python中可用，很适合于MySQL的脚本和自动化部署，以实现高可用性和可扩展性。通过使用MySQL Shell的AdminAPI，你可以避免手动配置许多实例的需要。相反，AdminAPI为各套MySQL实例提供了一个有效的现代接口，使你能够从一个中央工具中配置、管理和监控你的部署。

要开始使用InnoDB Cluster，你需要下载并安装MySQL Shell。你需要一些安装了MySQL服务器实例的主机，你也可以安装MySQL Router。

InnoDB Cluster支持MySQL Clone，这使你能够简单地配置实例。在过去，要在一个新的实例加入一组MySQL实例之前配置它，你需要以某种方式手动将事务转移到加入的实例。这可能涉及到制作文件副本，手动复制它们，等等。使用InnoDB Cluster，你可以简单地将一个实例添加到集群中，它将被自动配置。

同样，InnoDB Cluster与MySQL Router紧密结合，你可以使用AdminAPI来与它们一起工作。MySQL Router可以基于InnoDB Cluster自动配置自己，在一个称为引导的过程中，这消除了你手动配置路由的需要。然后，MySQL Router透明地将客户端应用程序连接到InnoDB Cluster，为客户端连接提供路由和负载平衡。这种整合还使你能够使用AdminAPI管理针对InnoDB Cluster启动的MySQL Router的某些方面。InnoDB集群的状态信息包括针对该集群启动的MySQL路由器的详细信息。操作使你能够在集群层面上创建MySQL路由器用户，与针对集群启动的MySQL路由器一起工作，等等。

关于这些技术的更多信息，请参阅描述中链接的用户文档。除了这些用户文档外，在MySQL Shell JavaScript API参考或MySQL Shell Python API参考中也有所有AdminAPI方法的开发者文档，可从连接器和API中获得。

# InnoDB 副本集

本章介绍了MySQL InnoDB ReplicaSet，它结合了MySQL技术，使你能够部署和管理第17章，复制。本内容是对InnoDB ReplicaSet的高级概述，完整的文档请参见MySQL InnoDB ReplicaSet。

一个InnoDB ReplicaSet由至少两个MySQL服务器实例组成，它提供了你所熟悉的所有MySQL Replication特性，如读取扩展和数据安全。InnoDB ReplicaSet使用了以下MySQL技术。

* MySQL Shell，它是MySQL的一个高级客户端和代码编辑器。
* MySQL服务器，以及第17章复制，它使一组MySQL实例能够提供可用性和异步读出扩展。InnoDB ReplicaSet提供了一种替代的、易于使用的编程方式来处理Replication。
* MySQL Router，一个轻量级的中间件，在你的应用程序和InnoDB ReplicaSet之间提供透明的路由。

通往InnoDB ReplicaSet的接口与MySQL InnoDB Cluster类似，你使用MySQL Shell来与作为ReplicaSet的MySQL服务器实例工作，MySQL Router也以与InnoDB Cluster相同的方式紧密集成。

基于MySQL Replication，InnoDB ReplicaSet有一个主实例，它复制到一个或多个辅助实例上。InnoDB ReplicaSet不提供InnoDB Cluster提供的所有功能，如自动故障切换或多主模式。但是，它确实支持一些功能，比如以类似的方式配置、添加和删除实例。你可以手动切换或故障切换到一个辅助实例，例如在发生故障的时候。你甚至可以采用一个现有的Replication部署，然后将其作为InnoDB ReplicaSet进行管理。

你可以使用AdminAPI与InnoDB ReplicaSet一起工作，AdminAPI作为MySQL Shell的一部分提供。AdminAPI在JavaScript和Python中可用，并且很适合MySQL部署的脚本和自动化，以实现高可用性和可扩展性。通过使用MySQL Shell的AdminAPI，你可以避免手动配置许多实例的需要。相反，AdminAPI为各套MySQL实例提供了一个有效的现代接口，使你能够从一个中央工具中配置、管理和监控你的部署。

要开始使用InnoDB ReplicaSet，你需要下载并安装MySQL Shell。你需要一些安装了MySQL Server实例的主机，你也可以安装MySQL Router。

InnoDB ReplicaSet支持MySQL Clone，这使你能够简单地配置实例。在过去，要在一个新的实例加入MySQL Replication部署之前配置它，你需要以某种方式手动将事务转移到加入的实例。这可能涉及制作文件副本，手动复制它们，等等。你可以简单地将一个实例添加到复制集，它就会被自动配置。

同样，InnoDB ReplicaSet与MySQL Router紧密结合，你可以使用AdminAPI来与它们一起工作。MySQL Router可以根据InnoDB ReplicaSet自动配置自己，这个过程被称为引导，这就不需要你手动配置路由了。然后，MySQL Router透明地将客户端应用程序连接到InnoDB ReplicaSet，为客户端连接提供路由和负载平衡。这种集成还使你能够使用AdminAPI来管理针对InnoDB ReplicaSet启动的MySQL Router的某些方面。InnoDB ReplicaSet状态信息包括针对ReplicaSet启动的MySQL路由器的详细信息。通过操作，你可以在ReplicaSet级别创建MySQL路由器用户，与针对ReplicaSet启动的MySQL路由器一起工作，等等。

关于这些技术的更多信息，请参阅描述中链接的用户文档。除了这些用户文档外，在MySQL Shell JavaScript API参考或MySQL Shell Python API参考中也有所有AdminAPI方法的开发者文档，可从连接器和API中获得。

# MySQL NDB 集群 8.0

# 分区

# 存储对象

# INFORMATION 表

# MySQL性能schema

# MySQL sys schema

# 连接器与API

MySQL连接器为客户程序提供与MySQL服务器的连接。APIs提供低级别的使用经典的MySQL协议或X协议来访问MySQL资源。连接器和API都使你能够从另一种语言或环境中连接和执行MySQL语句。包括ODBC、Java（JDBC）、C++、Python、Node.js、PHP、Perl、Ruby和C。

**MySQL连接器**

Oracle开发了许多连接器。

* Connector/C++使C++应用程序能够连接到MySQL。
* Connector/J提供了驱动程序支持，以便使用标准的Java应用程序连接到MySQL。Java数据库连接（JDBC）API，为从Java应用程序连接到MySQL提供驱动支持。
* Connector/NET使开发人员能够创建连接到MySQL的.NET应用程序。连接器/NET实现了一个全功能的ADO.NET接口，并提供了对使用ADO.NET感知工具的支持。工具。使用Connector/NET的应用程序可以用任何支持的.NET语言编写。

MySQL for Visual Studio可与Connector/NET和Microsoft Visual Studio 2012、2013、2015。和2017。MySQL for Visual Studio提供了对Visual Studio中的MySQL对象和数据的访问。作为作为一个Visual Studio包，它直接集成到Server Explorer中，提供了创建新连接和处理MySQL数据库对象的能力。连接并与MySQL数据库对象一起工作。

* Connector/ODBC为使用开放数据库连接（ODBC）API连接到MySQL提供驱动支持。连通性（ODBC）API连接到MySQL。支持来自Windows、Unix和macOS平台的ODBC连接。平台的ODBC连接。
* Connector/Python为使用符合Python DB的API从Python应用程序连接到MySQL提供驱动支持。该API符合Python DB API 2.0版本。不需要额外的Python模块或MySQL客户端需要额外的Python模块或MySQL客户端库。
* Connector/Node.js提供了一个异步API，用于使用X协议从Node.js应用程序连接到MySQL。应用程序连接到MySQL。Connector/Node.js支持管理数据库会话和模式。与MySQL文档存储集合一起工作，并使用原始SQL语句。

**MySQL的C语言API**

为了在C语言应用程序中直接使用MySQL，C语言API通过libmysqlclient协议提供了对MySQL客户端/服务器的低级访问。

通过libmysqlclient客户端库提供对MySQL客户端/服务器协议的低级访问。这是主要的方法

用于连接到MySQL服务器实例的主要方法，并且被MySQL命令行客户端

以及此处详述的许多MySQL连接器和第三方API都使用这种方法。libmysqlclient包含在MySQL的分发版本中。

另见MySQL C API实现。

要从C语言应用程序访问MySQL，或为本章中的连接器或API不支持的语言建立一个MySQL接口章中的连接器或API不支持的语言，C API是开始的地方。有一些程序员的实用程序可用于帮助这一过程；见第4.7节，"程序开发工具"。

**第三方MySQL APIs**

本章描述的其余API提供了从特定应用语言到MySQL的接口。语言的接口。这些第三方解决方案不是由Oracle开发或支持的。关于它们的用法和能力的基本信息，它们的用法和能力的基本信息在此提供，仅供参考。

所有的第三方语言API都是通过两种方法中的一种开发的，即使用libmysqlclient或通过实现一个本地驱动程序。这两种解决方案提供了不同的好处。

* 使用libmysqlclient可以与MySQL完全兼容，因为它使用与MySQL客户端应用程序相同的库

因为它使用与MySQL客户端应用程序相同的库。然而，其功能集仅限于通过libmysqlclient暴露的实现和然而，功能集仅限于通过libmysqlclient暴露的接口，而且性能可能较低，因为数据是在本地语言和MySQL之间复制的。

而且，由于数据在本地语言和MySQL API组件之间复制，性能可能会降低

* 本地驱动程序是完全在主机语言或环境中对MySQL网络协议的实现。或环境中实现MySQL网络协议。本机驱动程序速度快，因为组件之间的数据复制较少，而且它们可以提供标准MySQL API无法提供的高级功能。本机驱动程序也终端用户更容易构建和部署，因为不需要MySQL客户端库的副本来构建原生驱动组件。

表29.1，"MySQL APIs和接口 "列出了许多可用于MySQL的库和接口。

**Table 29.1 MySQL APIs and Interfaces**

|  |  |  |  |
| --- | --- | --- | --- |
| **Environment** | **API** | **Type** | **Notes** |
| **Ada** | GNU Ada MySQL Bindings | **libmysqlclient** | See [MySQL Bindings for GNU Ada](http://gnade.sourceforge.net/) |
| **C** | C API | **libmysqlclient** | See [MySQL 8.0 C API Developer Guide](https://dev.mysql.com/doc/c-api/8.0/en/). |
| **C++** | Connector/C++ | **libmysqlclient** | See [MySQL Connector/C++ 8.0 Developer Guide](https://dev.mysql.com/doc/connector-cpp/8.0/en/). |
|  | MySQL++ | **libmysqlclient** | See [MySQL++ website](http://tangentsoft.net/mysql++/doc/). |
|  | MySQL wrapped | **libmysqlclient** | See [MySQL wrapped](http://www.alhem.net/project/mysql/). |
| **Cocoa** | MySQL-Cocoa | **libmysqlclient** | Compatible with the Objective-C Cocoa environment. See <http://mysql-cocoa.sourceforge.net/> |
| **D** | MySQL for D | **libmysqlclient** | See [MySQL for D](http://www.steinmole.de/d/). |
| **Eiffel** | Eiffel MySQL | **libmysqlclient** | See [Section 29.13, “MySQL Eiffel Wrapper”](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\connectors-apis.html#apis-eiffel). |
| **Erlang** | **erlang-mysql-driver** | **libmysqlclient** | See [**erlang-mysql-driver**.](http://code.google.com/p/erlang-mysql-driver/) |
| **Haskell** | Haskell MySQL Bindings | Native Driver | See [Brian O'Sullivan's pure Haskell MySQL bindings](http://www.serpentine.com/blog/software/mysql/). |
|  | **hsql-mysql** | **libmysqlclient** | See [MySQL driver for Haskell](http://hackage.haskell.org/cgi-bin/hackage-scripts/package/hsql-mysql-1.7). |
| **Java/JDBC** | Connector/J | Native Driver | See [MySQL Connector/J 5.1 Developer Guide](https://dev.mysql.com/doc/connector-j/5.1/en/). |
| **Kaya** | MyDB | **libmysqlclient** | See [MyDB](http://kayalang.org/library/latest/MyDB). |
| **Lua** | LuaSQL | **libmysqlclient** | See [LuaSQL](http://keplerproject.github.io/luasql/doc/us/). |
| **.NET/Mono** | Connector/NET | Native Driver | See [MySQL Connector/NET Developer Guide](https://dev.mysql.com/doc/connector-net/en/). |
| **Objective Caml** | OBjective Caml MySQL Bindings | **libmysqlclient** | See [MySQL Bindings for Objective Caml](http://raevnos.pennmush.org/code/ocaml-mysql/). |
| **Octave** | Database bindings for GNU Octave | **libmysqlclient** | See [Database bindings for GNU Octave](http://octave.sourceforge.net/database/index.html). |
| **ODBC** | Connector/ODBC | **libmysqlclient** | See [MySQL Connector/ODBC Developer Guide](https://dev.mysql.com/doc/connector-odbc/en/). |
| **Perl** | **DBI**/**DBD::mysql** | **libmysqlclient** | See [Section 29.9, “MySQL Perl API”](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\connectors-apis.html#apis-perl). |
|  | **Net::MySQL** | Native Driver | See [**Net::MySQL**](http://search.cpan.org/dist/Net-MySQL/MySQL.pm) at CPAN |
| **PHP** | **mysql**, **ext/mysql** interface (deprecated) | **libmysqlclient** | See [Original MySQL API](https://dev.mysql.com/doc/apis-php/en/apis-php-mysql.html). |
|  | **mysqli**, **ext/mysqli** interface | **libmysqlclient** | See [MySQL Improved Extension](https://dev.mysql.com/doc/apis-php/en/apis-php-mysqli.html). |
|  | **PDO\_MYSQL** | **libmysqlclient** | See [MySQL Functions (PDO\_MYSQL)](https://dev.mysql.com/doc/apis-php/en/apis-php-pdo-mysql.html). |
|  | PDO mysqlnd | Native Driver |  |
| **Python** | Connector/Python | Native Driver | See [MySQL Connector/Python Developer Guide](https://dev.mysql.com/doc/connector-python/en/). |
| **Python** | Connector/Python C Extension | **libmysqlclient** | See [MySQL Connector/Python Developer Guide](https://dev.mysql.com/doc/connector-python/en/). |
|  | MySQLdb | **libmysqlclient** | See [Section 29.10, “MySQL Python API”](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\connectors-apis.html#apis-python). |
| **Ruby** | MySQL/Ruby | **libmysqlclient** | Uses **libmysqlclient**. See [Section 29.11.1, “The MySQL/Ruby API”](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\connectors-apis.html#apis-ruby-mysqlruby). |
|  | Ruby/MySQL | Native Driver | See [Section 29.11.2, “The Ruby/MySQL API”](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\connectors-apis.html#apis-ruby-rubymysql). |
| **Scheme** | **Myscsh** | **libmysqlclient** | See [**Myscsh**](https://github.com/aehrisch/myscsh). |
| **SPL** | **sql\_mysql** | **libmysqlclient** | See [**sql\_mysql** for SPL](http://www.clifford.at/spl/spldoc/sql_mysql.html). |
| **Tcl** | MySQLtcl | **libmysqlclient** | See [Section 29.12, “MySQL Tcl API”](file:///C:\Users\Administrator\Downloads\refman-8.0-en.html-chapter\refman-8.0-en.html-chapter\connectors-apis.html#apis-tcl). |

## 29.1 MySQL Connector/C++

MySQL Connector/C++手册以独立的形式发布，不作为MySQL参考手册的一部分。有关信息，请参见这些文件。

* 主手册。MySQL Connector/C++ 8.0开发者指南
* 发行说明。MySQL Connector/C++发布说明

## 29.2 MySQL Connector/J

MySQL Connector/J手册以独立的形式发布，而不是作为MySQL参考手册的一部分。有关信息，请参见这些文件。

* 主手册。MySQL Connector/J开发者指南
* 发布说明。MySQL Connector/J发布说明

## 29.3 MySQL Connector/NET

MySQL Connector/NET手册是以独立的形式发布的，而不是作为MySQL参考手册的一部分。有关信息，请参见这些文件。

* 主手册。MySQL Connector/NET开发者指南
* 发布说明。MySQL Connector/NET发布说明

## 29.4 MySQL Connector/ODBC

MySQL Connector/ODBC手册以独立的形式发布，不作为MySQL参考手册的一部分。有关信息，请参见这些文件。

* 主手册。MySQL Connector/ODBC开发者指南
* 发行说明。MySQL Connector/ODBC发布说明

## 29.5 MySQL Connector/Python

MySQL Connector/Python手册以独立的形式发布，而不是作为MySQL参考手册的一部分。有关信息，请参见这些文件。

* 主手册。MySQL Connector/Python开发者指南
* 发行说明。MySQL Connector/Python发布说明

## 29.6 MySQL Connector/Node.js

MySQL Connector/Node.js手册以独立的形式发布，而不是作为MySQL参考手册的一部分。有关信息，请参见这些文件。

* 发布说明。MySQL Connector/Node.js发布说明

## 29.7 MySQL C API

MySQL C API开发者指南以独立的形式发布，而不是作为《MySQL参考手册》的一部分。参见《MySQL 8.0 C API开发者指南》。

## 29.8 MySQL的PHP API

MySQL PHP API手册现在以独立的形式发布，而不是作为《MySQL参考手册》的一部分。参见MySQL和PHP。

## 29.9 MySQL的Perl API

Perl DBI模块为数据库访问提供了一个通用接口。你可以编写一个DBI脚本，在不改变的情况下与许多不同的数据库引擎一起工作。要在MySQL中使用DBI，请安装以下内容。

1. DBI模块。
2. DBD::mysql模块。这是Perl的DataBase Driver (DBD)模块。
3. 可选地，为你想访问的任何其他类型的数据库服务器的DBD模块。

Perl DBI 是推荐的 Perl 接口。它取代了一个叫做mysqlperl的旧接口，后者应该被认为是过时的。

这些部分包含了关于在MySQL中使用Perl以及用Perl编写MySQL应用程序的信息。

* 于Perl DBI支持的安装说明，见第2.13节，"Perl安装说明"。
* 关于从选项文件中读取选项的例子，见第5.8.4节，"在多服务器环境中使用客户端程序"。
* 关于安全编码的提示，见第6.1.1节 "安全指南"。
* 调试技巧，见第5.9.1.4节 "在gdb下调试mysqld"。
* 关于一些Perl特定的环境变量，见第4.9节 "环境变量"。
* 关于在macOS上运行的注意事项，见2.4节 "在macOS上安装MySQL"。
* 关于引用字符串字数的方法，见第9.1.1节，"字符串字数"。

DBI信息可以在命令行、在线或以打印形式获得。

* 一旦你安装了DBI和DBD::mysql模块，你可以在命令行中用perldoc命令获得关于它们的信息。

shell> **perldoc DBI**

shell> **perldoc DBI::FAQ**

shell> **perldoc DBD::mysql**

你也可以使用 pod2man、pod2html 等将这些信息翻译成其他格式。

* 关于Perl DBI的在线信息，请访问DBI网站，http://dbi.perl.org/。该网站主办了一个普通的DBI邮件列表。
* 关于印刷信息，官方的 DBI 书籍是 Programming the Perl DBI (Alligator Descartes and Tim Bunce, O'Reilly & Associates, 2000)。关于这本书的信息可以在DBI网站上获得，http://dbi.perl.org/。

## 29.10 MySQL Python API

MySQLdb是一个第三方驱动，为Python提供MySQL支持，符合Python DB API 2.0版本。它可以在 http://sourceforge.net/projects/mysql-python/ 上找到。

新的MySQL Connector/Python组件提供了一个到相同的Python API的接口，它内置于MySQL服务器中，并由Oracle支持。参见《MySQL Connector/Python开发者指南》，了解关于连接器的详细信息，以及Python应用程序的编码指南和Python代码示例。

## 29.11 MySQL Ruby APIs

对于开发MySQL应用程序的Ruby程序员，有两个API可用。

* MySQL/Ruby API是基于libmysqlclient API库的。关于安装和使用MySQL/Ruby API的信息，见第29.11.1节 "MySQL/Ruby API"。
* 编写Ruby/MySQL API是为了使用本地MySQL网络协议（一个本地驱动程序）。关于安装和使用Ruby/MySQL API的信息，请参见第29.11.2节，"Ruby/MySQL API"。

关于Ruby语言的背景和语法信息，见Ruby编程语言。

### 29.11.1 MySQL/Ruby API

MySQL/Ruby模块通过libmysqlclient提供对使用Ruby的MySQL数据库的访问。

有关安装该模块的信息，以及暴露的功能，请参见MySQL/Ruby。

### 29.11.2 Ruby/MySQL API

Ruby/MySQL模块通过使用MySQL网络协议的本地驱动接口，提供对使用Ruby的MySQL数据库的访问。

有关安装该模块的信息，以及暴露的功能，请参见Ruby/MySQL。

## 29.12 MySQL Tcl API

MySQLtcl是一个简单的API，用于从Tcl编程语言访问MySQL数据库服务器。它可以在http://www.xdobry.de/mysqltcl/。

## 29.13 MySQL Eiffel Wrapper

Eiffel MySQL是一个使用Eiffel编程语言访问MySQL数据库服务器的接口，由Michael Ravits编写。它可以在http://efsa.sourceforge.net/archive/ravits/mysql.htm。

# MySQL企业版

**章节目录**

MySQL企业版是一种商业产品。与MySQL社区版一样，MySQL企业版包括MySQL服务器，这是一个完全集成的事务安全、符合ACID标准的数据库，具有完全提交、回滚、崩溃恢复和行级锁定功能。此外，MySQL企业版还包括以下组件，旨在提供监控和在线备份，以及提高安全性和可扩展性。

以下各节简要讨论了这些组件中的每一个，并指出在哪里可以找到更详细的信息。要了解更多关于商业产品的信息，请参阅https://www.mysql.com/products/。

* MySQL企业监控器
* MySQL企业级备份
* MySQL企业安全
* MySQL企业级加密
* MySQL企业审计
* MySQL企业级防火墙
* MySQL企业级线程池
* MySQL企业版数据屏蔽和解除识别

## 30.1 MySQL企业监控器概述

MySQL企业监控器是一个用于MySQL的企业监控系统，它可以密切关注你的MySQL服务器，通知你潜在的问题和难题，并建议你如何解决问题。MySQL企业监控器可以监控各种配置，从对你的业务很重要的单个MySQL服务器，一直到为一个繁忙的网站提供动力的巨大的MySQL服务器群。

下面的讨论简要地总结了构成MySQL企业监控器产品的基本组件。欲了解更多信息，请参阅MySQL企业监控器手册，可在https://dev.mysql.com/doc/mysql-monitor/en/。

根据你的数据库和网络拓扑结构，MySQL企业监控器组件可以以不同的配置进行安装，以便为你提供可靠和反应迅速的监控数据的最佳组合，并使数据库服务器机器上的开销最小。一个典型的MySQL企业监控器的安装包括：。

* 一个或多个要监控的MySQL服务器。MySQL企业监控器可以同时监控社区和企业版的MySQL服务器
* 每台被监控的主机的MySQL企业监控器代理。
* 一个单一的MySQL企业服务管理器，它整理来自代理的信息，并提供收集数据的用户界面。

MySQL企业监控器被设计用于监控一个或多个MySQL服务器。监测信息是通过使用一个代理，即MySQL企业监测代理来收集的。该代理与它所监控的主机和MySQL服务器进行通信，收集变量、状态和健康信息，并将这些信息发送到MySQL企业服务管理器。

代理所收集的关于你所监控的每个MySQL服务器和主机的信息被发送到MySQL企业服务管理器。该服务器对来自代理的所有信息进行整理。在整理代理发送的信息时，MySQL企业服务管理器不断测试所收集的数据，将服务器的状态与合理值进行比较。当达到阈值时，服务器可以触发一个事件（包括警报和通知），以突出一个潜在的问题，如低内存、高CPU使用率，或更复杂的条件，如缓冲区大小和状态信息不足。我们把每个测试，及其相关的阈值，称为规则。

这些规则，以及警报和通知，都被称为MySQL企业顾问。顾问构成了MySQL企业服务管理器的一个关键部分，因为它们提供关于潜在问题的警告信息和故障排除建议。

MySQL企业服务管理器包括一个Web服务器，你可以通过任何Web浏览器与它进行交互。这个界面，即MySQL企业监控器用户界面，显示由代理收集的所有信息，并让你作为一个组或单独查看所有的服务器及其当前状态。你可以使用MySQL企业监控器用户界面控制和配置服务的所有方面。

由MySQL企业监控器代理进程提供的信息还包括统计和查询信息，你可以以图表的形式查看这些信息。例如，你可以以图表的形式查看诸如服务器负载、查询数量或索引使用信息等方面的时间。该图表可以让你准确定位服务器上的问题或潜在问题，并可以通过检查特定时间间隔的数据来帮助诊断数据库或外部问题（如外部系统或网络故障）的影响。

MySQL企业监控代理也可以被配置为收集关于在你的服务器上执行的查询的详细信息，包括执行每个查询的行数和性能时间。你可以将详细的查询数据与图形信息联系起来，以确定当你遇到一个特别高的负载、索引或其他问题时，哪些查询正在执行。查询数据由一个叫做查询分析器的系统支持，数据可以根据你的需要以不同的方式呈现。

## 30.2 MySQL企业备份概述

MySQL企业备份为MySQL数据库执行热备份操作。该产品是为InnoDB存储引擎创建的表的高效和可靠备份而设计的。为了完整起见，它还可以备份MyISAM和其他存储引擎的表。

下面的讨论简要地总结了MySQL企业备份。欲了解更多信息，请参阅MySQL企业备份手册，可在https://dev.mysql.com/doc/mysql-enterprise-backup/en/。

热备份是在数据库运行和应用程序对其进行读写时进行的。这种类型的备份不会阻碍正常的数据库操作，它甚至可以捕捉到备份发生时的变化。由于这些原因，当你的数据库 "长大 "时，热备份是可取的--当数据大到备份需要大量时间时，以及当你的数据对你的业务足够重要时，你必须捕获每一个变化，而不使你的应用程序、网站或Web服务离线。

MySQL企业备份对所有使用InnoDB存储引擎的表进行热备份。对于使用MyISAM或其他非InnoDB存储引擎的表，它进行 "暖 "备份，数据库继续运行，但在备份时不能修改这些表。为了提高备份操作的效率，你可以指定InnoDB作为新表的默认存储引擎，或者转换现有的表来使用InnoDB存储引擎。

## 30.3 MySQL企业版安全概述

MySQL企业版提供了使用外部服务实现安全功能的插件。

* MySQL企业版包括一个认证插件，使MySQL服务器能够使用PAM（可插拔认证模块）来认证MySQL用户。PAM使系统能够使用一个标准接口来访问各种认证方法，如Unix密码或LDAP目录。更多信息，请参见第6.4.1.5节 "PAM可插拔认证"。
* MySQL企业版包括一个在Windows上执行外部认证的认证插件，使MySQL服务器能够使用本地Windows服务来认证客户端连接。已经登录Windows的用户可以根据他们环境中的信息从MySQL客户端程序连接到服务器，而无需指定额外的密码。欲了解更多信息，请参见第6.4.1.6节 "Windows可插拔认证"。
* MySQL企业版包括一组基于OpenSSL库的加密函数，这些函数在SQL级别上暴露了OpenSSL功能。这些函数能够使用几种方法掩盖现有数据，如混淆（去除识别特征）、生成格式化的随机数据以及数据替换或替代。欲了解更多信息，请参见第30.4节 "MySQL企业版加密概述"。
* MySQL企业版5.7及更高版本包括一个密钥插件，它使用Oracle Key Vault作为密钥存储的后端。欲了解更多信息，请参见第6.4.4节 "MySQL钥匙圈"。

关于其他相关的企业安全特性，见第30.4节 "MySQL企业加密概述"

## 30.4 MySQL企业版加密概述

MySQL企业版包括一组基于OpenSSL库的加密函数，这些函数在SQL级别暴露了OpenSSL的功能。这些函数使企业应用程序能够执行以下操作。

* 使用公钥非对称加密法实施额外的数据保护
* 创建公钥和私钥以及数字签名
* 执行非对称加密和解密
* 使用加密散列法进行数字签名和数据验证和确认

欲了解更多信息，请参见第6.6节 "MySQL企业加密"。

关于其他相关的企业安全功能，见第30.3节 "MySQL企业安全概述"。

## 30.5 MySQL企业审计概述

MySQL企业版包括MySQL企业审计，使用一个服务器插件实现。MySQL企业审计使用开放的MySQL审计API来实现对特定MySQL服务器上执行的连接和查询活动的标准、基于策略的监控和记录。MySQL企业审计是为满足Oracle审计规范而设计的，它为受内部和外部监管准则制约的应用提供了一个开箱即用、易于使用的审计和合规性解决方案。

安装后，审计插件使MySQL服务器能够产生一个包含服务器活动审计记录的日志文件。日志内容包括客户何时连接和断开连接，以及他们在连接时执行了哪些操作，例如他们访问了哪些数据库和表。

欲了解更多信息，请参见第6.4.5节 "MySQL企业审计"

## 30.6 MySQL企业防火墙概述

MySQL企业版包括MySQL企业防火墙，这是一个应用级的防火墙，使数据库管理员能够根据与所接受的语句模式的允许列表的匹配来允许或拒绝SQL语句的执行。这有助于加强MySQL服务器对诸如SQL注入或试图通过在其合法的查询工作量特征之外使用应用程序来利用它们的攻击。

在防火墙上注册的每个MySQL账户都有自己的语句允许列表，从而能够为每个账户定制保护。对于一个给定的账户，防火墙可以在记录或保护模式下运行，以便在接受的语句模式中进行培训，或对不可接受的语句进行保护。

欲了解更多信息，请参阅第6.4.7节 "MySQL企业防火墙"。

## 30.7 MySQL企业线程池概述

MySQL企业版包括MySQL企业线程池，使用一个服务器插件实现。MySQL服务器中的默认线程处理模式是使用每个客户端连接的一个线程来执行语句。随着越来越多的客户端连接到服务器并执行语句，整体性能会下降。在MySQL企业版中，线程池插件提供了一个替代的线程处理模型，旨在减少开销和提高性能。该插件实现了一个线程池，通过有效管理大量客户端连接的语句执行线程来提高服务器性能。

欲了解更多信息，请参见第5.6.3节 "MySQL企业版线程池"

## 30.8 MySQL企业版数据屏蔽和取消识别概述

MySQL企业版5.7及更高版本包括MySQL企业数据屏蔽和取消识别，以包含一个插件和一组用户定义的函数的插件库形式实现。数据屏蔽通过用替代物替换真实值来隐藏敏感信息。MySQL企业数据屏蔽和取消识别功能可以使用几种方法来屏蔽现有数据，如混淆（去除识别特征）、生成格式化的随机数据以及数据替换或替代。

欲了解更多信息，请参见第6.5节 "MySQL企业数据屏蔽和取消识别"

# MySQL WORKBENCH

MySQL Workbench为处理MySQL服务器和数据库提供了一个图形化的工具。MySQL Workbench完全支持MySQL 5.5及以上版本。

下面的讨论简要地描述了MySQL Workbench的功能。欲了解更多信息，请参见MySQL Workbench手册，可在<https://dev.mysql.com/doc/workbench/en/>

MySQL Workbench提供了五个主要的功能领域。

**SQL开发：**使你能够创建和管理与数据库服务器的连接。除了使你能够配置连接参数外，MySQL Workbench还提供了使用内置SQL编辑器在数据库连接上执行SQL查询的能力。这个功能取代了以前由查询浏览器独立应用程序提供的功能。

**数据建模：**使你能够以图形方式创建数据库模式的模型，在模式和实时数据库之间进行反向和正向工程，并使用全面的表编辑器编辑数据库的所有方面。表编辑器为编辑表、列、索引、触发器、分区、选项、插入和权限、例程和视图提供易于使用的设施。

**服务器管理：**使你能够创建和管理服务器实例。

**数据迁移：**允许你从Microsoft SQL Server、Sybase ASE、SQLite、SQL Anywhere、PostreSQL和其他RDBMS表、对象和数据迁移到MySQL。迁移还支持从MySQL的早期版本迁移到最新版本。

**MySQL企业支持**：支持企业产品，如MySQL企业备份和MySQL审计。

MySQL Workbench有两个版本，即社区版和商业版。社区版是免费提供的。商业版以低价提供额外的企业功能，如数据库文档生成。

# MySQL OCI 市场

**章节目录**

本章介绍了如何将MySQL企业版部署为Oracle云基础设施（OCI）市场应用。这是一个BYOL产品。

注意

有关OCI市场的更多信息，请参阅市场概述。

MySQL企业版市场应用是一个OCI计算实例，运行Oracle Linux 7.7，带有MySQL EE 8.0。在部署的镜像上的MySQL EE安装类似于RPM安装，如第2.5.4节 "使用来自Oracle的RPM包在Linux上安装MySQL "中所述。

关于MySQL企业版的更多信息，见第30章，MySQL企业版。

关于MySQL高级配置的更多信息，见《安全部署指南》。

关于Oracle Linux 7的更多信息，见Oracle Linux文档

本产品由用户管理，意味着你负责升级和维护。

## 32.1 在Oracle云基础设施上部署MySQL EE的先决条件

作出以下假设。

* 你熟悉OCI的术语。如果你是OCI的新手，请参阅 "入门"
* 你可以访问一个正确配置的虚拟云网络（VCN）和子网。欲了解更多信息，请参阅虚拟网络。
* 你有在你的租约的隔间内部署OCI市场应用程序所需的权限。欲了解更多信息，请参阅政策如何工作。

## 32.2 在Oracle云基础设施上部署MySQL EE

要在Oracle云基础设施上部署MySQL EE，请执行以下操作。

1. 打开OCI Marketplace，选择MySQL。

会显示MySQL列表。

1. 单击 "启动实例 "以开始应用程序的启动过程。

显示 "创建计算实例 "对话框。

有关如何完成字段的信息，请参阅创建Linux实例。

默认情况下，MySQL服务器监听端口为3306，并配置了一个用户，即root。

重要提示

当部署完成后，MySQL服务器被启动，你必须连接到计算实例，并找回写入MySQL日志文件的默认root密码。

参见用SSH连接以获取更多信息。

安装了以下MySQL软件。

* MySQL Server EE
* MySQL企业备份
* MySQL Shell
* MySQL Router

**MySQL配置**

为了安全起见，启用了以下内容。

* SELinux：更多信息，见配置和使用SELinux
* firewalld：更多信息，见firewalld的当前状态和设置

下列MySQL插件已被启用。

* thread\_pool
* validate\_password

在启动时，会发生以下情况。

* MySQL服务器读取/etc/my.cnf和/etc/my.cnf.d/中所有名为\*.cnf的文件。
* /etc/my.cnf.d/perf-tuning.cnf由/usr/bin/mkcnf根据选定的OCI形状创建。

注意

要禁用这一机制，请删除/etc/systemd/system/mysqld.service.d/perf-tuning.conf。

性能调优是为以下形状配置的。

* VM.Standard2.1
* VM.Standard2.2
* VM.Standard2.4
* VM.Standard2.8
* VM.Standard2.16
* VM.Standard2.24
* VM.Standard.E2.1
* VM.Standard.E2.2
* VM.Standard.E2.4
* VM.Standard.E2.8
* BM.Standard2.52

对于所有其他形状，使用VM.Standard2.1的调谐。

## 32.3 配置网络访问

关于OCI安全规则的信息，请参见安全规则。

重要信息

你必须在端口22（SSH）和3306（MySQL）上启用入口，如果你打算使用MySQL X协议，还可以选择33060。

## 32.4 连接

本节描述了连接到OCI计算实例上部署的MySQL服务器的各种连接方法

**用SSH连接**

本节给出了一些关于从类似UNIX的平台连接到OCI计算的细节。关于用SSH连接的更多信息，请参阅使用SSH访问Oracle Linux实例和连接到您的实例。

要用SSH连接到计算实例上运行的Oracle Linux，请运行以下命令。

ssh opc@***computeIP***

其中 opc 是计算用户，computeIP 是你的计算实例的 IP 地址。

要找到为根用户创建的临时根密码，运行以下命令。

sudo grep 'temporary password' /var/log/mysqld.log

要改变你的默认密码，使用生成的临时密码登录服务器，使用以下命令： mysql -uroot -p。然后运行以下命令。

ALTER USER 'root'@'localhost' IDENTIFIED BY 'MyNewPass4!';

**与MySQL客户端连接**

注意

要从你的本地MySQL客户端连接，你必须首先在MySQL服务器上创建一个允许远程登录的用户。

要从你的本地MySQL客户端连接到MySQL服务器，从你的shell会话中运行以下命令。

mysql -uroot -p -h***computeIP***

其中computeIP是你的计算实例的IP地址。

**用MySQL Shell连接**

要从你的本地MySQL Shell连接到MySQL服务器，请运行以下命令来启动你的shell会话。

mysqlsh \connect root@***computeIP***

其中computeIP是你的计算实例的IP地址。

关于MySQL Shell连接的更多信息，请参阅MySQL Shell连接。

**与工作台连接**

要从MySQL工作台连接到MySQL服务器，请参阅MySQL工作台的连接。

## 32.5 维护

本产品是用户管理的，这意味着你要负责升级和维护。

**升级MySQL**

现有的安装是基于RPM的，要升级MySQL服务器，请参阅第2.11.6节，"升级Unix/Linux上基于二进制或包的MySQL安装"。

你可以使用scp将所需的RPM复制到OCI计算实例中，或者从OCI对象存储中复制它，如果你已经配置了对它的访问。文件存储也是一种选择。欲了解更多信息，请参阅文件存储和NFS。

**备份和恢复**

MySQL企业备份是首选的备份和恢复解决方案。欲了解更多信息，请参阅备份到云存储。

有关MySQL企业备份的信息，请参见《MySQL企业备份入门》。

有关默认的MySQL备份和恢复的信息，见第7章，备份和恢复。

# 附录A

# 附录B

# 附录C

# MySQL词汇表