

MySQL教程

书栈(BookStack.CN)

目 录

致谢

Introduction

1.介绍

2.MySQL 安装

3.MySQL 管理

4.MySQL PHP 语法

5.MySQL 连接

6.MySQL 创建数据库

7.MySQL 删除数据库

8.MySQL 选择数据库

9.MySQL 数据类型

10.MySQL 创建数据表

11.MySQL 删除数据表

12.MySQL 插入数据

13.MySQL 查询数据

14.MySQL where 子句

15.MySQL UPDATE 查询

16.MySQL DELETE 语句

17.MySQL LIKE 子句

18.MySQL UNION 操作符

19.MySQL 排序

20.MySQL GROUP BY 语句

致谢

当前文档《MySQL教程》由 进击的皇虫 使用 书栈(BookStack.CN) 进行构建, 生成于 2018-10-23。

书栈(BookStack.CN) 仅提供文档编写、整理、归类等功能, 以及对文档内容的生成和导出工具。

文档内容由网友们编写和整理, 书栈(BookStack.CN) 难以确认文档内容知识点是否错漏。如果您在阅读文档获取知识的时候, 发现文档内容有不恰当的地方, 请向我们反馈, 让我们共同携手, 将知识准确、高效且有效地传递给每一个人。

同时, 如果您在日常工作、生活和学习中遇到有价值有营养的知识文档, 欢迎分享到 书栈(BookStack.CN), 为知识的传承献上您的一份力量!

如果当前文档生成时间太久, 请到 书栈(BookStack.CN) 获取最新的文档, 以跟上知识更新换代的步伐。

文档地址: <http://www.bookstack.cn/books/colaly-mysql>

书栈官网: <http://www.bookstack.cn>

书栈开源: <https://github.com/TruthHun>

分享, 让知识传承更久远! 感谢知识的创造者, 感谢知识的分享者, 也感谢每一位阅读到此处的读者, 因为我们都将成为知识的传承者。

Introduction

MySQL教程

Mysql是最流行的关系型数据库管理系统，在WEB应用方面MySQL是最好的RDBMS(Relational Database Management System：关系数据库管理系统)应用软件之一。在本教程中，会让大家快速掌握Mysql的基本知识，并轻松使用Mysql数据库。

原文：<https://colaly.gitbooks.io/mysql/content/>

1.介绍

MySQL教程

MySQL是最流行的关系型数据库管理系统，在WEB应用方面MySQL是最好的RDBMS(Relational Database Management System：关系数据库管理系统)应用软件之一。

在本教程中，会让大家快速掌握MySQL的基本知识，并轻松使用MySQL数据库。

什么是数据库？

数据库(Database)是按照数据结构来组织、存储和管理数据的仓库，

每个数据库都有一个或多个不同的API用于创建，访问，管理，搜索和复制所保存的数据。

我们也可以将数据存储于文件中，但是在文件中读写数据速度相对较慢。

所以，现在我们使用关系型数据库管理系统(RDBMS)来存储和管理的大数据量。所谓的关系型数据库，是建立在关系模型基础上的数据库，借助于集合代数等数学概念和方法来处理数据库中的数据。

RDBMS即关系数据库管理系统(Relational Database Management System)的特点：

- 1.数据以表格的形式出现
 - 2.每行为各种记录名称
 - 3.每列为记录名称所对应的数据域
 - 4.许多的行和列组成一张表单
 - 5.若干的表单组成database
-

RDBMS 术语

在我们开始学习MySQL 数据库前，让我们先了解下RDBMS的一些术语：

- 数据库：数据库是一些关联表的集合。
- 数据表：表是数据的矩阵。在一个数据库中的表看起来像一个简单的电子表格。
- 列：一列(数据元素)包含了相同的数据，例如邮政编码的数据。
- 行：一行(=元组，或记录)是一组相关的数据，例如一条用户订阅的数据。
- 冗余：存储两倍数据，冗余降低了性能，但提高了数据的安全性。
- 主键：主键是唯一的。一个数据表中只能包含一个主键。你可以使用主键来查询数据。
- 外键：外键用于关联两个表。

- 复合键：复合键（组合键）将多个列作为一个索引键，一般用于复合索引。
- 索引：使用索引可快速访问数据库表中的特定信息。索引是对数据库表中一列或多列的值进行排序的一种结构。类似于书籍的目录。
- 参照完整性：参照的完整性要求关系中不允许引用不存在的实体。与实体完整性是关系模型必须满足的完整性约束条件，目的是保证数据的一致性。

Mysql数据库

MySQL是一个关系型数据库管理系统，由瑞典MySQL AB公司开发，目前属于Oracle公司。MySQL是一种关联数据库管理系统，关联数据库将数据保存在不同的表中，而不是将所有数据放在一个大仓库内，这样就增加了速度并提高了灵活性。

- Mysql是开源的，所以你不需要支付额外的费用。
- Mysql支持大型的数据库。可以处理拥有上千万条记录的大型数据库。
- MySQL使用标准的SQL数据语言形式。
- Mysql可以允许于多个系统上，并且支持多种语言。这些编程语言包括C、C++、Python、Java、Perl、PHP、Eiffel、Ruby和Tcl等。
- Mysql对PHP有很好的支持，PHP是目前最流行的web开发语言。
- MySQL支持大型数据库，支持5000万条记录的数据仓库，32位系统表文件最大可支持4GB，64位系统支持最大的表文件为8TB。
- Mysql是可以定制的，采用了GPL协议，你可以修改源码来开发自己的Mysql系统。

在开始学习本教程前你应该了解？

在开始学习本教程前你应该了解PHP和HTML的基础知识，并能简单的应用。

本教程的很多例子都跟PHP语言有关，我们的实例基本上是采用PHP语言来演示

原文：<https://colaly.gitbooks.io/mysql/content/1jie-shao.html>

2.MySQL 安装

MySQL安装

所有平台的Mysql下载地址为：[MySQL 下载](#)。挑选你需要的MySQL Community Server版本及对应的平台。

Linux/UNIX上安装Mysql

Linux平台上推荐使用RPM包来安装Mysql, MySQL AB提供了以下RPM包的下载地址：

- **MySQL**
 - MySQL服务器。你需要该选项，除非你只想连接运行在另一台机器上的MySQL服务器。
 - **MySQL-client**
 - MySQL 客户端程序，用于连接并操作Mysql服务器。
 - **MySQL-devel**
 - 库和包含文件，如果你想要编译其它MySQL客户端，例如Perl模块，则需要安装该RPM包。
 - **MySQL-shared**
 - 该软件包包含某些语言和应用程序需要动态装载的共享库(libmysqlclient.so*)，使用MySQL。
 - **MySQL-bench**
 - MySQL数据库服务器的基准和性能测试工具。
- 以下安装Mysql RPM的实例是在SuSE Linux系统上进行，当然该安装步骤也适合应用于其他支持RPM的Linux系统，如：Centos。

安装步骤如下：

使用root用户登陆你的Linux系统。

下载Mysql RPM包，下载地址为：[MySQL 下载](#)。

通过以下命令执行Mysql安装，rpm包为你下载的rpm包：

```
1. [root@host]# rpm -i MySQL-5.0.9-0.i386.rpm
```

以上安装mysql服务器的过程会创建mysql用户，并创建一个mysql配置文件my.cnf。

你可以在/usr/bin和/usr/sbin中找到所有与MySQL相关的二进制文件。所有数据表和数据库将在/var/lib/mysql目录中创建。

以下是一些mysql可选包的安装过程，你可以根据自己的需要来安装：

```
1. [root@host]# rpm -i MySQL-client-5.0.9-0.i386.rpm
2. [root@host]# rpm -i MySQL-devel-5.0.9-0.i386.rpm
3. [root@host]# rpm -i MySQL-shared-5.0.9-0.i386.rpm
4. [root@host]# rpm -i MySQL-bench-5.0.9-0.i386.rpm
```

Window上安装Mysql

Window上安装Mysql相对来说会较为简单，你只需要载[MySQL 下载](#)中下载window版本的mysql安装包，并解压安装包。

双击 setup.exe 文件，接下来你只需要安装默认的配置点击"next"即可，默认情况下安装信息会在 C:\mysql 目录中。

接下来你可以通过"开始" =》在搜索框中输入 " cmd" 命令 =》 在命令提示符上切换到 C:\mysql\bin 目录，并输入一下命令：

```
1. mysqld.exe --console
```

如果安装成功以上命令将输出一些mysql启动及InnoDB信息。

验证Mysql安装

在成功安装Mysql后，一些基础表会表初始化，在服务器启动后，你可以通过简单的测试来验证Mysql是否工作正常。

使用 mysqladmin 工具来获取服务器状态：

使用 mysqladmin 命令俩检查服务器的版本,在linux上该二进制文件位于 /usr/bin on linux，在window上该二进制文件位于C:\mysql\bin 。

```
1. [root@host]# mysqladmin --version
```

linux上该命令将输出以下结果，该结果基于你的系统信息：

```
1. mysqladmin Ver 8.23 Distrib 5.0.9-0, for redhat-linux-gnu on i386
```

如果以上命令执行后未输入任何信息，说明你的Mysql未安装成功。

使用 MySQL Client(Mysql客户端) 执行简单的SQL命令

你可以在 MySQL Client(Mysql客户端) 使用 `mysql` 命令连接到Mysql服务器上，默认情况下Mysql服务器的密码为空，所以本实例不需要输入密码。

命令如下：

```
1. [root@host]# mysql
```

以上命令执行后会输出 `mysql>`提示符，这说明你已经成功连接到Mysql服务器上，你可以在 `mysql>`提示符执行SQL命令：

```
1. mysql> SHOW DATABASES;
2. +-----+
3. | Database |
4. +-----+
5. | mysql   |
6. | test    |
7. +-----+
8. 2 rows in set (0.13 sec)
```

Mysql安装后需要做的

Mysql安装成功后，默认的root用户密码为空，你可以使用以下命令来创建root用户的密码：

```
1. [root@host]# mysqladmin -u root password "new_password";
```

现在你可以通过以下命令来连接到Mysql服务器：

```
1. [root@host]# mysql -u root -p
2. Enter password:*****
```

注意：在输入密码时，密码是不会显示了，你正确输入即可。

Linux系统启动时启动 MySQL

如果你需要在Linux系统启动时启动 MySQL 服务器，你需要在 `/etc/rc.local` 文件中添加以下命

令：

```
1. /etc/init.d/mysqld start
```

同样，你需要将 `mysqld` 二进制文件添加到 `/etc/init.d/` 目录中。

原文： <https://colaly.gitbooks.io/mysql/content/2mysql-an-zhuang.html>

3.MySQL 管理

MySQL管理

启动及关闭 MySQL 服务器

首先，我们需要通过以下命令来检查MySQL服务器是否启动：

```
1. ps -ef | grep mysqld
```

如果MySQL已经启动，以上命令将输出mysql进程列表， 如果mysql未启动，你可以使用以下命令来启动mysql服务器：

```
1. root@host# cd /usr/bin
2. ./mysqld_safe
3. &
```

如果你想关闭目前运行的 MySQL 服务器，你可以执行以下命令：

```
1. root@host# cd /usr/bin
2. ./mysqladmin -u root -p shutdown
3. Enter password: *****
```

MySQL 用户设置

如果你需要添加 MySQL 用户，你只需要在 mysql 数据库中的 user 表添加新用户即可。

以下为添加用户的实例，用户名为guest，密码为guest123，并授权用户可进行 SELECT，INSERT 和 UPDATE操作权限：

```
1. root@host# mysql -u root -p
2. Enter password:*****
3. mysql
4. >
5. use mysql;
6. Database changed
```

```

7.
8. mysql
9. >
10. INSERT INTO user
11.      (host, user, password,
12.       select_priv, insert_priv, update_priv)
13.      VALUES ('localhost', 'guest',
14.              PASSWORD('guest123'), 'Y', 'Y', 'Y');
15. Query OK, 1 row affected (0.20 sec)
16.
17. mysql
18. >
19. FLUSH PRIVILEGES;
20. Query OK, 1 row affected (0.01 sec)
21.
22. mysql
23. >
24. SELECT host, user, password FROM user WHERE user = 'guest';
25. +-----+-----+-----+
26. | host      | user    | password          |
27. +-----+-----+-----+
28. | localhost | guest   | 6f8c114b58f2ce9e |
29. +-----+-----+-----+
30. 1 row in set (0.00 sec)

```

在添加用户时，请注意使用MySQL提供的 `PASSWORD()` 函数来对密码进行加密。你可以在以上实例看到用户密码加密后为： `6f8c114b58f2ce9e`。

注意：在 MySQL5.7 中 `user` 表的 `password` 已换成了 **`authentication_string`**。

注意：在注意需要执行 **`FLUSH PRIVILEGES`** 语句。这个命令执行后会重新载入授权表。

如果你不使用该命令，你就无法使用新创建的用户来连接mysql服务器，除非你重启mysql服务器。

你可以在创建用户时，为用户指定权限，在对应的权限列中，在插入语句中设置为 `'Y'` 即可，用户权限列表如下：

- `Select_priv`
- `Insert_priv`
- `Update_priv`
- `Delete_priv`
- `Create_priv`
- `Drop_priv`

- Reload_priv
- Shutdown_priv
- Process_priv
- File_priv
- Grant_priv
- References_priv
- Index_priv
- Alter_priv

另外一种添加用户的方法为通过SQL的 GRANT 命令，你下命令会给指定数据库TUTORIALS添加用户 zara ，密码为 zara123 。

```
1. root@host# mysql -u root -p password;
2. Enter password:*****
3. mysql
4. >
5. use mysql;
6. Database changed
7.
8. mysql
9. >
10. GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP
11. -
12. >
13. ON TUTORIALS.*
14. -
15. >
16. TO 'zara'@'localhost'
17. -
18. >
19. IDENTIFIED BY 'zara123';
```

以上命令会在mysql数据库中的user表创建一条用户信息记录。

注意:MySQL 的SQL语句以分号 (;) 作为结束标识。

/etc/my.cnf 文件配置

一般情况下，你不需要修改该配置文件，该文件默认配置如下：

```
1. [mysqld]
```

```

2. datadir=/var/lib/mysql
3. socket=/var/lib/mysql/mysql.sock
4.
5. [mysql.server]
6. user=mysql
7. basedir=/var/lib
8.
9. [safe_mysqld]
10. err-log=/var/log/mysqld.log
11. pid-file=/var/run/mysqld/mysqld.pid

```

在配置文件中，你可以指定不同的错误日志文件存放的目录，一般你不需要改动这些配置。

管理MySQL的命令

以下列出了使用MySQL数据库过程中常用的命令：

- **USE数据库名**: 选择要操作的MySQL数据库，使用该命令后所有MySQL命令都只针对该数据库。

```

mysql
>
use RUN00B;
Database changed

```

- **SHOW DATABASES**: 列出 MySQL 数据库管理系统的数据库列表。

```

mysql
>
SHOW DATABASES;
+-----+
| Database |
+-----+
| information_schema |
| RUN00B |
| cdcol |
| mysql |
| onethink |
| performance_schema |
| phpmyadmin |
| test |
| wecenter |

```

```
| wordpress |
+-----+
10 rows in set (0.02 sec)
```

- **SHOW TABLES:**显示指定数据库的所有表，使用该命令前需要使用 use 命令来选择要操作的数据库。

```
mysql
>
use RUNOOB;
Database changed
mysql
>
SHOW TABLES;
+-----+
| Tables_in_runoob |
+-----+
| employee_tbl |
| runoob_tbl |
| tcount_tbl |
+-----+
3 rows in set (0.00 sec)
```

- **SHOW COLUMNS FROM**数据表:显示数据表的属性，属性类型，主键信息，是否为 NULL，默认值等其他信息。

```
mysql
>
SHOW COLUMNS FROM runoob_tbl;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| runoob_id | int(11) | NO | PRI | NULL | |
| runoob_title | varchar(255) | YES | | NULL | |
| runoob_author | varchar(255) | YES | | NULL | |
| submission_date | date | YES | | NULL | |
+-----+-----+-----+-----+-----+-----+
4 rows in set (0.01 sec)
```

- **SHOW INDEX FROM**数据表:显示数据表的详细索引信息，包括PRIMARY KEY (主键)。

```
mysql
```

>

SHOW INDEX FROM runoob_tbl;

```

+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
-----+-----+-----+
| Table | Non_unique | Key_name | Seq_in_index | Column_name | Collation | Cardinality | Sub_part | Packed | Null | Index_type | Comment | Index_comment |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
-----+-----+-----+
| runoob_tbl | 0 | PRIMARY | 1 | runoob_id | A | 2 | NULL | NULL | | BTREE | | |
| | |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
-----+-----+-----+
1 row in set (0.00 sec)

```

- **SHOW TABLE STATUS LIKE [FROM db_name] [LIKE 'pattern'] \G**:该命令将输出 Mysql数据库管理系统的性能及统计信息。

mysql

>

SHOW TABLE STATUS FROM RUNOOB; # 显示数据库 RUNOOB 中所有表的信息

mysql

>

SHOW TABLE STATUS from RUNOOB LIKE 'runoob%'; # 表名以runoob开头的表的信息

mysql

>

SHOW TABLE STATUS from RUNOOB LIKE 'runoob%\G; # 加上 \G, 查询结果按列打印

Gif 图演示:



笔记列表

- oocarain

ooocarain@163.com

记录 MySQL 学习过程遇到的问题。

系统: win32 位MySQL 版本: 5.7.17-log

MySQL 语法对大小写不敏感,但是大写更容易看出。

一、启动关闭MySQL服务

1【开始菜单】搜索 services.msc 打开 windows【服务管理器】,可以在这里开启关闭 MySQL 服务。

2 在 cmd 中使用命令:

```
net start mysql #启动mysql服务
```

```
net stop mysql #关闭mysql服务
```

遇到net命令无法识别,如下:

这是环境变量没有配置的原因,究竟是哪一个文件的环境变量没有配置呢?

是 C:\windows\system32\ 这个路径下的 net.exe 没有配置环境变量

现切换到这个路径下试一下可不可以使用 net 命令：

在 Powershell 需要使用

```
.\net stop mysql
```

关闭服务。

在 cmd 中可以直接使用

```
net start mysql
```

启动服务。

将c:\windows\system32添加到系统的Path中后：

成功！！！！

原文：<https://colaly.gitbooks.io/mysql/content/3mysql-guan-li.html>

4.MySQL PHP 语法

MySQL PHP 语法

MySQL 可应用于多种语言，包括 PERL，C，C++，JAVA 和 PHP。在这些语言中，Mysql在PHP的web开发中是应用最广泛。

在本教程中我们大部分实例都采用了PHP语言。如果你想了解Mysql在PHP中的应用，可以访问我们的PHP中使用Mysql介绍。

PHP提供了多种方式来访问和操作Mysql数据库记录。PHP Mysql函数格式如下：

```
1. mysql_function(value,value,...);
```

以上格式中 function部分描述了mysql函数的功能，如

```
1. mysqli_connect($connect);
2. mysqli_query($connect, "SQL 语句");
3. mysql_fetch_array()
4. mysql_connect()
5. mysql_close()
```

以下实例展示了PHP调用mysql函数的语法：

```
1. <html>
2. <head>
3. <meta charset="utf-8">
4. <title>PHP MySQL</title>
5. </head>
6. <body>
7. <?php
8.     $retval = mysql_function(value, [value,...]);
9.     if( !$retval )
10.    {
11.        die ( "相关错误信息" );
12.    }
13.    // 其他 MySQL 或 PHP 语句
14. ?>
15. </body>
```

```
16. </html>
```

从下一章开始，我们将学习到更多的MySQL功能函数。

原文: <https://colaly.gitbooks.io/mysql/content/4mysql-php-yu-fa.html>

5.MySQL 连接

MySQL 连接

使用mysql二进制方式连接

您可以使用MySQL二进制方式进入到mysql命令提示符下来连接MySQL数据库。

实例

以下是从命令行中连接mysql服务器的简单实例：

```
1. [root@host]# mysql -u root -p
2. Enter password:*****
```

在登录成功后会出现 `mysql>` 命令提示窗口，你可以在上面执行任何 SQL 语句。

以上命令执行后，登录成功输出结果如下：

```
1. Welcome to the MySQL monitor.  Commands end with ; or \g.
2. Your MySQL connection id is 2854760 to server version: 5.0.9
3.
4. Type 'help;' or '\h' for help. Type '\c' to clear the buffer.
```

在以上实例中，我们使用了root用户登录到mysql服务器，当然你也可以使用其他mysql用户登录。

如果用户权限足够，任何用户都可以在mysql的命令提示窗口中进行SQL操作。

退出 `mysql>` 命令提示窗口可以使用 `exit` 命令，如下所示：

```
1. mysql> exit
2. Bye
```

使用 PHP 脚本连接 MySQL

PHP 提供了 `mysql_connect()` 函数来连接数据库。

该函数有5个参数，在成功链接到MySQL后返回连接标识，失败返回 `FALSE` 。

语法

```
1. connection mysql_connect(server,user,passwd,new_link,client_flag);
```

参数说明：

参数	描述
server	可选。规定要连接的服务器。可以包括端口号，例如 "hostname:port"，或者到本地套接字的路径，例如对于 localhost 的 ":/path/to/socket"。如果 PHP 指令 mysql.default_host 未定义（默认情况），则默认值是 'localhost:3306'。
user	可选。用户名。默认值是服务器进程所有者的用户名。
passwd	可选。密码。默认值是空密码。
new_link	可选。如果用同样的参数第二次调用 mysql_connect()，将不会建立新连接，而将返回已经打开的连接标识。参数 new_link 改变此行为并使 mysql_connect() 总是打开新的连接，甚至当 mysql_connect() 曾在前面被用同样的参数调用过。
client_flag	可选。client_flags 参数可以是以下常量的组合：MYSQL_CLIENT_SSL - 使用 SSL 加密MYSQL_CLIENT_COMPRESS - 使用压缩协议MYSQL_CLIENT_IGNORE_SPACE - 允许函数名后的间隔MYSQL_CLIENT_INTERACTIVE - 允许关闭连接之前的交互超时非活动时间

你可以使用PHP的 mysql_close() 函数来断开与MySQL数据库的连接。

该函数只有一个参数为mysql_connect()函数创建连接成功后返回的 MySQL 连接标识符。

语法

```
1. bool mysql_close ( resource $link_identifier );
```

本函数关闭指定的连接标识所关联的到 MySQL 服务器的非持久连接。如果没有指定 link_identifier，则关闭上一个打开的连接。

提示：通常不需要使用 mysql_close()，因为已打开的非持久连接会在脚本执行完毕后自动关闭。

注释：mysql_close() 不会关闭由 mysql_pconnect() 建立的持久连接。

实例

你可以尝试以下实例来连接到你的 MySQL 服务器：

```
1. <html>
2. <head>
3. <meta charset="utf-8">
4. <title>Connecting MySQL Server</title>
5. </head>
6. <body>
```

```
7.  <?php
8.      $dbhost = 'localhost:3306'; //mysql服务器主机地址
9.      $dbuser = 'guest';          //mysql用户名
10.     $dbpass = 'guest123'; //mysql用户名密码
11.     $conn = mysql_connect($dbhost, $dbuser, $dbpass);
12.     if(! $conn )
13.     {
14.         die('Could not connect: ' . mysql_error());
15.     }
16.     echo 'Connected successfully';
17.     mysql_close($conn);
18. ?>
19. </body>
20. </html>
```

原文: <https://colaly.gitbooks.io/mysql/content/5mysql-lian-jie.html>

6.MySQL 创建数据库

MySQL 创建数据库

使用 mysqladmin 创建数据库

使用普通用户，你可能需要特定的权限来创建或者删除 MySQL 数据库。

所以我们这边使用root用户登录，root用户拥有最高权限，可以使用 `mysql mysqladmin` 命令来创建数据库。

实例

以下命令简单的演示了创建数据库的过程，数据名为 RUNOOB：

```
1. [root@host]# mysqladmin -u root -p create RUNOOB
2. Enter password:*****
```

以上命令执行成功后会创建 MySQL 数据库 RUNOOB。

使用 PHP脚本 创建数据库

PHP使用 `mysql_query` 函数来创建或者删除 MySQL 数据库。

该函数有两个参数，在执行成功时返回 `TRUE`，否则返回 `FALSE`。

语法

```
1. bool mysql_query( sql, connection );
```

参数	描述
sql	必需。规定要发送的 SQL 查询。注释：查询字符串不应以分号结束。
connection	可选。规定 SQL 连接标识符。如果未规定，则使用上一个打开的连接。

实例

以下实例演示了使用PHP来创建一个数据库：


```
1. <html>
2. <head>
3. <meta charset="utf-8">
4. <title>创建 MySQL 数据库</title>
5. </head>
6. <body>
7. <?php
8. $dbhost = 'localhost:3036';
9. $dbuser = 'root';
10. $dbpass = 'rootpassword';
11. $conn = mysql_connect($dbhost, $dbuser, $dbpass);
12. if(! $conn )
13. {
14.     die('连接错误: ' . mysql_error());
15. }
16. echo '连接成功<br />';
17. $sql = 'CREATE DATABASE RUNOOB';
18. $retval = mysql_query( $sql, $conn );
19. if(! $retval )
20. {
21.     die('创建数据库失败: ' . mysql_error());
22. }
23. echo "数据库 RUNOOB 创建成功\n";
24. mysql_close($conn);
25. ?>
26. </body>
27. </html>
```

原文: <https://colaly.gitbooks.io/mysql/content/6mysql-chuang-jian-shu-ju-ku.html>

7.MySQL 删除数据库

MySQL 删除数据库

使用 mysqladmin 删除数据库

使用普通用户登陆mysql服务器，你可能需要特定的权限来创建或者删除 MySQL 数据库。

所以我们这边使用root用户登录，root用户拥有最高权限，可以使用 `mysql mysqladmin` 命令来创建数据库。

在删除数据库过程中，务必要十分谨慎，因为在执行删除命令后，所有数据将会消失。

以下实例删除数据库RUNOOB(该数据库在前一章节已创建)：

1. `[root@host]#mysqladmin -u root -p drop RUNOOB`
2. `Enter password:*****`

执行以上删除数据库命令后，会出现一个提示框，来确认是否真的删除数据库：

1. `Dropping` the database `is` potentially a very bad thing to `do`.
2. `Any` data stored `in` the database will be destroyed.
- 3.
4. `Do` you really want to drop the `'RUNOOB'` database `[y/N]` `y`
5. `Database "RUNOOB"` dropped

使用PHP脚本删除数据库

PHP使用 `mysql_query` 函数来创建或者删除 MySQL 数据库。

该函数有两个参数，在执行成功时返回 `TRUE`，否则返回 `FALSE`。

语法

1. `bool mysql_query(sql, connection);`

参数	描述

sql	必需。规定要发送的 SQL 查询。注释：查询字符串不应以分号结束。
connection	可选。规定 SQL 连接标识符。如果未规定，则使用上一个打开的连接。

实例

以下实例演示了使用PHP `mysql_query`函数来删除数据库：

```

1.  <html>
2.  <head>
3.  <meta charset="utf-8">
4.  <title>删除 MySQL 数据库</title>
5.  </head>
6.  <body>
7.  <?php
8.  $dbhost = 'localhost:3036';
9.  $dbuser = 'root';
10. $dbpass = 'rootpassword';
11. $conn = mysql_connect($dbhost, $dbuser, $dbpass);
12. if(! $conn )
13. {
14.     die('连接失败: ' . mysql_error());
15. }
16. echo '连接成功<br />';
17. $sql = 'DROP DATABASE RUNOOB';
18. $retval = mysql_query( $sql, $conn );
19. if(! $retval )
20. {
21.     die('删除数据库失败: ' . mysql_error());
22. }
23. echo "数据库 RUNOOB 删除成功\n";
24. mysql_close($conn);
25. ?>
26. </body>
27. </html>

```

注意：在使用PHP脚本删除数据库时，不会出现确认是否删除信息，会直接删除指定数据库，所以你在删除数据库时要特别小心。

原文：<https://colaly.gitbooks.io/mysql/content/7mysql-shan-chu-shu-ju-ku.html>

8.MySQL 选择数据库

MySQL 选择数据库

在你连接到 MySQL 数据库后，可能有多个可以操作的数据库，所以你需要选择你要操作的数据库。

从命令提示窗口中选择MySQL数据库

在 `mysql>` 提示窗口中可以很简单的选择特定的数据库。你可以使用SQL命令来选择指定的数据库。

实例

以下实例选取了数据库 RUNOOB：

```
1. [root@host]# mysql -u root -p
2. Enter password:*****
3. mysql> use RUNOOB;
4. Database changed
5. mysql>
```

执行以上命令后，你就已经成功选择了 RUNOOB 数据库，在后续的操作中都会在 RUNOOB 数据库中执行。

注意：所有的数据库名，表名，表字段都是区分大小写的。所以你在使用SQL命令时需要输入正确的名称。

使用PHP脚本选择MySQL数据库

PHP 提供了函数 `mysql_select_db` 来选取一个数据库。函数在执行成功后返回 `TRUE`，否则返回 `FALSE`。

语法

```
1. bool mysql_select_db( db_name, connection );
```

	描述
db_name	必需。规定要选择的数据库。

connection	可选。规定 MySQL 连接。如果未指定，则使用上一个连接。
------------	--------------------------------

实例

以下实例展示了如何使用 `mysql_select_db` 函数来选取一个数据库：

```
1. <html>
2. <head>
3. <meta charset="utf-8">
4. <title>选择 MySQL 数据库</title>
5. </head>
6. <body>
7. <?php
8. $dbhost = 'localhost:3036';
9. $dbuser = 'guest';
10. $dbpass = 'guest123';
11. $conn = mysql_connect($dbhost, $dbuser, $dbpass);
12. if(! $conn )
13. {
14.     die('连接失败: ' . mysql_error());
15. }
16. echo '连接成功';
17. mysql_select_db( 'RUNOOB' );
18. mysql_close($conn);
19. ?>
20. </body>
21. </html>
```

原文: <https://colaly.gitbooks.io/mysql/content/8mysql-xuan-ze-shu-ju-ku.html>

9.MySQL 数据类型

MySQL 数据类型

MySQL中定义数据字段的类型对你数据库的优化是非常重要的。

MySQL支持多种类型，大致可以分为三类：数值、日期/时间和字符串(字符)类型。

数值类型

MySQL支持所有标准SQL数值数据类型。

这些类型包括严格数值数据类型(INTEGER、SMALLINT、DECIMAL和NUMERIC)，以及近似数值数据类型(FLOAT、REAL和DOUBLE PRECISION)。

关键字INT是INTEGER的同义词，关键字DEC是DECIMAL的同义词。

BIT数据类型保存位字段值，并且支持MyISAM、MEMORY、InnoDB和BDB表。

作为SQL标准的扩展，MySQL也支持整数类型TINYINT、MEDIUMINT和BIGINT。下面的表显示了需要的每个整数类型的存储和范围。

类型	大小	范围（有符号）	范围（无符号）	用途
TINYINT	1 字节	(-128, 127)	(0, 255)	小整数值
SMALLINT	2 字节	(-32 768, 32 767)	(0, 65 535)	大整数值
MEDIUMINT	3 字节	(-8 388 608, 8 388 607)	(0, 16 777 215)	大整数值
INT或 INTEGER	4 字节	(-2 147 483 648, 2 147 483 647)	(0, 4 294 967 295)	大整数值
BIGINT	8 字节	(-9 223 372 036 854 775 808, 9 223 372 036 854 775 807)	(0, 18 446 744 073 709 551 615)	极大整数值

FLOAT	4 字节	(-3.402 823 466 E+38, -1.175 494 351 E-38), 0, (1.175 494 351 E-38, 3.402 823 466 351 E+38)	0, (1.175 494 351 E-38, 3.402 823 466 E+38)	单精度浮点数值
DOUBLE	8 字节	(-1.797 693 134 862 315 7 E+308, -2.225 073 858 507 201 4 E-308), 0, (2.225 073 858 507 201 4 E-308, 1.797 693 134 862 315 7 E+308)	0, (2.225 073 858 507 201 4 E-308, 1.797 693 134 862 315 7 E+308)	双精度浮点数值
DECIMAL	对 DECIMAL(M,D) , 如果M>D, 为 M+2否则为D+2	依赖于M和D的值	依赖于M和D的值	小数

日期和时间类型

表示时间值的日期和时间类型为DATETIME、DATE、TIMESTAMP、TIME和YEAR。

每个时间类型有一个有效值范围和一个"零"值，当指定不合法的MySQL不能表示的值时使用"零"值。

TIMESTAMP类型有专有的自动更新特性，将在后面描述。

类型	大小		(字节)	范围	格式	用途
		DATE	3	1000-01-01/9999-12-31	YYYY-MM-DD	日期值
		TIME	3	'-838:59:59'/'838:59:59'	HH <input type="text"/> SS	时间值或持续时间
		YEAR	1	1901/2155	YYYY	年份值
		DATETIME	8	1000-01-01 00:00:00/9999-12-31 23:59:59	YYYY-MM-DD HH <input type="text"/> SS	混合日期和时间值
		TIMESTAMP	4	1970-01-01 00:00:00/2037 年某时	YYYYMMDD HHMMSS	混合日期和时间值，时间戳

字符串类型

字符串类型指CHAR、VARCHAR、BINARY、VARBINARY、BLOB、TEXT、ENUM和SET。该节描述了这些类型如何工作以及如何在查询中使用这些类型。

类型	大小	用途
CHAR	0-255字节	定长字符串

VARCHAR	0-65535 字节	变长字符串
TINYBLOB	0-255字节	不超过 255 个字符的二进制字符串
TINYTEXT	0-255字节	短文本字符串
BLOB	0-65 535字节	二进制形式的长文本数据
TEXT	0-65 535字节	长文本数据
MEDIUMBLOB	0-16 777 215字节	二进制形式的中等长度文本数据
MEDIUMTEXT	0-16 777 215字节	中等长度文本数据
LONGBLOB	0-4 294 967 295字节	二进制形式的极大文本数据
LONGTEXT	0-4 294 967 295字节	极大文本数据

CHAR和VARCHAR类型类似，但它们保存和检索的方式不同。它们的最大长度和是否尾部空格被保留等方面也不同。在存储或检索过程中不进行大小写转换。

BINARY和VARBINARY类类似于CHAR和VARCHAR，不同的是它们包含二进制字符串而不要非二进制字符串。也就是说，它们包含字节字符串而不是字符串。这说明它们没有字符集，并且排序和比较基于列值字节的数值值。

BLOB是一个二进制大对象，可以容纳可变数量的数据。有4种BLOB类型：TINYBLOB、BLOB、MEDIUMBLOB和LONGBLOB。它们只是可容纳值的最大长度不同。

有4种TEXT类型：TINYTEXT、TEXT、MEDIUMTEXT和LONGTEXT。这些对应4种BLOB类型，有相同的最大长度和存储需求。

原文：<https://colaly.gitbooks.io/mysql/content/9mysql-shu-ju-lei-xing.html>

10.MySQL 创建数据表

MySQL 创建数据表

创建MySQL数据表需要以下信息：

- 表名
- 表字段名
- 定义每个表字段

语法

以下为创建MySQL数据表的SQL通用语法：

```
1. CREATE TABLE table_name (column_name column_type);
```

以下例子中我们将在 RUNOOB 数据库中创建数据表runoob_tbl：

```
1. runoob_tbl(  
2.     runoob_id INT NOT NULL AUTO_INCREMENT,  
3.     runoob_title VARCHAR(100) NOT NULL,  
4.     runoob_author VARCHAR(40) NOT NULL,  
5.     submission_date DATE,  
6.     PRIMARY KEY ( runoob_id )  
7. );
```

实例解析：

- 如果你不想字段为**NULL**可以设置字段的属性为**NOT NULL**， 在操作数据库时如果输入该字段的数据为**NULL**，就会报错。
- **AUTO_INCREMENT**定义列为自增的属性，一般用于主键，数值会自动加1。
- **PRIMARY KEY**关键字用于定义列为主键。 您可以使用多列来定义主键，列间以逗号分隔。

通过命令提示符创建表

通过 `mysql>` 命令窗口可以很简单的创建MySQL数据表。你可以使用 SQL 语句**CREATE TABLE**来创建数据表。

实例

以下为创建数据表 runoob_tbl 实例：

```

1. root@host# mysql -u root -p
2. Enter password:*****
3. mysql> use RUNOOB;
4. Database changed
5. mysql> CREATE TABLE runoob_tbl(
6.     -> runoob_id INT NOT NULL AUTO_INCREMENT,
7.     -> runoob_title VARCHAR(100) NOT NULL,
8.     -> runoob_author VARCHAR(40) NOT NULL,
9.     -> submission_date DATE,
10.    -> PRIMARY KEY ( runoob_id )
11.    -> );
12. Query OK, 0 rows affected (0.16 sec)
13. mysql>

```

注意：MySQL命令终止符为分号（；）。

使用PHP脚本创建数据表

你可以使用PHP的 `mysql_query()` 函数来创建已存在数据库的数据表。

该函数有两个参数，在执行成功时返回 `TRUE`，否则返回 `FALSE`。

语法

```
1. bool mysql_query( sql, connection );
```

参数	描述
sql	必需。规定要发送的 SQL 查询。注释：查询字符串不应以分号结束。
connection	可选。规定 SQL 连接标识符。如果未规定，则使用上一个打开的连接。

实例

以下实例使用了PHP脚本来创建数据表：

```

1. <html>
2. <head>

```

```
3. <meta charset="utf-8">
4. <title>创建 MySQL 数据表</title>
5. </head>
6. <body>
7. <?php
8. $dbhost = 'localhost:3036';
9. $dbuser = 'root';
10. $dbpass = 'rootpassword';
11. $conn = mysql_connect($dbhost, $dbuser, $dbpass);
12. if(! $conn )
13. {
14.     die('连接失败: ' . mysql_error());
15. }
16. echo '连接成功<br />';
17. $sql = "CREATE TABLE runoob_tbl( ".
18.         "runoob_id INT NOT NULL AUTO_INCREMENT, ".
19.         "runoob_title VARCHAR(100) NOT NULL, ".
20.         "runoob_author VARCHAR(40) NOT NULL, ".
21.         "submission_date DATE, ".
22.         "PRIMARY KEY ( runoob_id )); ";
23. mysql_select_db( 'RUNOOB' );
24. $retval = mysql_query( $sql, $conn );
25. if(! $retval )
26. {
27.     die('数据表创建失败: ' . mysql_error());
28. }
29. echo "数据表创建成功\n";
30. mysql_close($conn);
31. ?>
32. </body>
33. </html>
```

原文: <https://colaly.gitbooks.io/mysql/content/10mysql-chuang-jian-shu-ju-biao.html>

11.MySQL 删除数据表

MySQL 删除数据表

MySQL中删除数据表是很容易操作的，但是你再进行删除表操作时要非常小心，因为执行删除命令后所有数据都会消失。

语法

以下为删除MySQL数据表的通用语法：

```
1. DROP TABLE table_name ;
```

在命令提示窗口中删除数据表

在mysql>命令提示窗口中删除数据表SQL语句为**DROP TABLE**：

实例

以下实例删除了数据表runoob_tbl：

```
1. root@host# mysql -u root -p
2. Enter password:*****
3. mysql> use RUNOOB;
4. Database changed
5. mysql> DROP TABLE runoob_tbl
6. Query OK, 0 rows affected (0.8 sec)
7. mysql>
```

使用PHP脚本删除数据表

PHP使用 `mysql_query` 函数来删除 MySQL 数据表。

该函数有两个参数，在执行成功时返回 `TRUE`，否则返回 `FALSE`。

语法

```
1. bool mysql_query( sql, connection );
```

参数	描述
sql	必需。规定要发送的 SQL 查询。注释：查询字符串不应以分号结束。
connection	可选。规定 SQL 连接标识符。如果未规定，则使用上一个打开的连接。

实例

以下实例使用了PHP脚本删除数据表runoob_tbl：

```
1. <html>
2. <head>
3. <meta charset="utf-8">
4. <title>创建 MySQL 数据表</title>
5. </head>
6. <body>
7. <?php
8. $dbhost = 'localhost:3036';
9. $dbuser = 'root';
10. $dbpass = 'rootpassword';
11. $conn = mysql_connect($dbhost, $dbuser, $dbpass);
12. if(! $conn )
13. {
14.     die('连接失败: ' . mysql_error());
15. }
16. echo '连接成功<br />';
17. $sql = "DROP TABLE runoob_tbl";
18. mysql_select_db( 'RUNOOB' );
19. $retval = mysql_query( $sql, $conn );
20. if(! $retval )
21. {
22.     die('数据表删除失败: ' . mysql_error());
23. }
24. echo "数据表删除成功\n";
25. mysql_close($conn);
26. ?>
27. </body>
28. </html>
```

原文: <https://colalyl.gitbooks.io/mysql/content/11mysql-shan-chu-shu-ju-biao.html>

12.MySQL 插入数据

MySQL 插入数据

MySQL 表中使用**INSERT INTO**SQL语句来插入数据。

你可以通过 `mysql>` 命令提示窗口中向数据表中插入数据，或者通过PHP脚本来插入数据。

语法

以下为向MySQL数据表插入数据通用的**INSERT INTO**SQL语法：

```
1. INSERT INTO table_name ( field1, field2,...fieldN )
2.                               VALUES
3.                               ( value1, value2,...valueN );
```

如果数据是字符型，必须使用单引号或者双引号，如："value"。

通过命令提示窗口插入数据

以下我们将使用 **SQLINSERT INTO**语句向 MySQL 数据表 `runoob_tbl` 插入数据

实例

以下实例中我们将想 `runoob_tbl` 表插入三条数据：

```
1. root@host# mysql -u root -p password;
2. Enter password:*****
3. mysql> use RUNOOB;
4. Database changed
5. mysql> INSERT INTO runoob_tbl
6.     ->(runoob_title, runoob_author, submission_date)
7.     ->VALUES
8.     ->("Learn PHP", "John Pou1", NOW());
9. Query OK, 1 row affected (0.01 sec)
10. mysql> INSERT INTO runoob_tbl
11.     ->(runoob_title, runoob_author, submission_date)
12.     ->VALUES
13.     ->("Learn MySQL", "Abdul S", NOW());
```

```

14. Query OK, 1 row affected (0.01 sec)
15. mysql> INSERT INTO runoob_tbl
16.      ->(runoob_title, runoob_author, submission_date)
17.      ->VALUES
18.      ->("JAVA Tutorial", "Sanjay", '2007-05-06');
19. Query OK, 1 row affected (0.01 sec)
20. mysql>

```

注意：使用箭头标记(->)不是SQL语句的一部分，它仅仅表示一个新行，如果一条SQL语句太长，我们可以通过回车键来创建一个新行来编写SQL语句，SQL语句的命令结束符为分号(;)。

在以上实例中，我们并没有提供 runoob_id 的数据，因为该字段我们在创建表的时候已经设置它为 AUTO_INCREMENT(自动增加) 属性。 所以，该字段会自动递增而不需要我们去设置。实例中 NOW() 是一个 MySQL 函数，该函数返回日期和时间。

使用PHP脚本插入数据

你可以使用PHP 的 `mysql_query()` 函数来执行SQL **INSERT INTO**命令来插入数据。

该函数有两个参数，在执行成功时返回 TRUE，否则返回 FALSE。

语法

```
1. bool mysql_query( sql, connection );
```

参数	描述
sql	必需。规定要发送的 SQL 查询。注释：查询字符串不应以分号结束。
connection	可选。规定 SQL 连接标识符。如果未规定，则使用上一个打开的连接。

实例

以下实例中程序接收用户输入的三个字段数据，并插入数据表中：

```

1. <html>
2. <head>
3. <meta charset="utf-8">
4. <title>向 MySQL 数据库添加数据</title>
5. </head>
6. <body>
7. <?php

```

```

8.  if(isset($_POST['add']))
9.  {
10. $dbhost = 'localhost:3036';
11. $dbuser = 'root';
12. $dbpass = 'rootpassword';
13. $conn = mysql_connect($dbhost, $dbuser, $dbpass);
14. if(! $conn )
15. {
16.     die('Could not connect: ' . mysql_error());
17. }
18.
19. if(! get_magic_quotes_gpc() )
20. {
21.     $runoob_title = addslashes ($_POST['runoob_title']);
22.     $runoob_author = addslashes ($_POST['runoob_author']);
23. }
24. else
25. {
26.     $runoob_title = $_POST['runoob_title'];
27.     $runoob_author = $_POST['runoob_author'];
28. }
29. $submission_date = $_POST['submission_date'];
30.
31. $sql = "INSERT INTO runoob_tbl ".
32.         "(runoob_title,runoob_author, submission_date) ".
33.         "VALUES ".
34.         "('$runoob_title','$runoob_author','$submission_date')";
35. mysql_select_db('RUNOOB');
36. $retval = mysql_query( $sql, $conn );
37. if(! $retval )
38. {
39.     die('Could not enter data: ' . mysql_error());
40. }
41. echo "Entered data successfully\n";
42. mysql_close($conn);
43. }
44. else
45. {
46.     ?>
47.     <form method="post" action="<?php $_PHP_SELF ?>">
48.     <table width="600" border="0" cellspacing="1" cellpadding="2">
49.     <tr>

```



```

50. <td width="250">Tutorial Title</td>
51. <td>
52. <input name="runoob_title" type="text" id="runoob_title">
53. </td>
54. </tr>
55. <tr>
56. <td width="250">Tutorial Author</td>
57. <td>
58. <input name="runoob_author" type="text" id="runoob_author">
59. </td>
60. </tr>
61. <tr>
62. <td width="250">Submission Date [ yyyy-mm-dd ]</td>
63. <td>
64. <input name="submission_date" type="text" id="submission_date">
65. </td>
66. </tr>
67. <tr>
68. <td width="250"> </td>
69. <td> </td>
70. </tr>
71. <tr>
72. <td width="250"> </td>
73. <td>
74. <input name="add" type="submit" id="add" value="Add Tutorial">
75. </td>
76. </tr>
77. </table>
78. </form>
79. <?php
80. }
81. ?>
82. </body>
83. </html>

```

在我们接收用户提交的数据时，为了数据的安全性我们需要使用 `get_magic_quotes_gpc()` 函数来判断特殊字符的转义是否已经开启。如果这个选项为 `off`（未开启），返回 `0`，那么我们就必须调用 `addslashes` 这个函数来为字符串增加转义。

义。

你也可以添加其他检查数据的方法，比如邮箱格式验证，电话号码验证，是否为整数验证等。

原文: <https://colaly.gitbooks.io/mysql/content/12mysql-cha-ru-shu-ju.html>

MySQL 查询数据

你可以通过 `mysql>` 命令提示窗口中在数据库中查询数据，或者通过PHP脚本来查询数据。

语法

```
1. SELECT column_name, column_name
2. FROM table_name
3. [WHERE Clause]
4. [OFFSET M ] [LIMIT N]
```

- 查询语句中你可以使用一个或者多个表，表之间使用逗号(,)分割，并使用WHERE语句来设定查询条件。
- SELECT 命令可以读取一条或者多条记录。
- 你可以使用星号(*)来代替其他字段，SELECT语句会返回表的所有字段数据
- 你可以使用 WHERE 语句来包含任何条件。
- 你可以通过OFFSET指定SELECT语句开始查询的数据偏移量。默认情况下偏移量为0。
- 你可以使用 LIMIT 属性来设定返回的记录数。

通过命令提示符获取数据

实例

```
1. root@host# mysql -u root -p password;
2. Enter password:*****
3. mysql> use RUNOOB;
4. Database changed
5. mysql> SELECT * from runoob_tbl
6. +-----+-----+-----+-----+
```

```

7. | runoob_id | runoob_title | runoob_author | submission_date |
8. +-----+-----+-----+-----+
9. |          1 | Learn PHP    | John Poul     | 2007-05-21      |
10. |          2 | Learn MySQL  | Abdul S       | 2007-05-21      |
11. |          3 | JAVA Tutorial | Sanjay        | 2007-05-21      |
12. +-----+-----+-----+-----+
13. 3 rows in set (0.01 sec)
14.
15. mysql>

```

使用PHP脚本来获取数据

使用PHP函数的mysql_query()及SQL SELECT命令来获取数据。

该函数用于执行SQL命令，然后通过 PHP 函数 mysql_fetch_array() 来使用或输出所有查询的数据。

mysql_fetch_array() 函数从结果集中取得一行作为关联数组，或数字数组，或二者兼有 返回根据从结果集取得的行生成的数组，如果没有更多行则返回 false。

以下实例为从数据表 runoob_tbl 中读取所有记录。

实例

尝试以下实例来显示数据表 runoob_tbl 的所有记录。

```

1. <?php
2. $dbhost = 'localhost:3036';
3. $dbuser = 'root';
4. $dbpass = 'rootpassword';
5. $conn = mysql_connect($dbhost, $dbuser, $dbpass);
6. if(! $conn )
7. {
8.     die('Could not connect: ' . mysql_error());
9. }
10. $sql = 'SELECT runoob_id, runoob_title,
11.         runoob_author, submission_date
12.         FROM runoob_tbl';
13.
14. mysql_select_db('RUNOOB');
15. $retval = mysql_query( $sql, $conn );

```

```

16. if(! $retval )
17. {
18.     die('Could not get data: ' . mysql_error());
19. }
20. while($row = mysql_fetch_array($retval, MYSQL_ASSOC))
21. {
22.     echo "Tutorial ID :{$row['runoob_id']} <br> ".
23.         "Title: {$row['runoob_title']} <br> ".
24.         "Author: {$row['runoob_author']} <br> ".
25.         "Submission Date : {$row['submission_date']} <br> ".
26.         "-----<br>";
27. }
28. echo "Fetched data successfully\n";
29. mysql_close($conn);
30. ?>

```

以上实例中，读取的每行记录赋值给变量\$row，然后再打印出每个值。

注意：记住如果你需要在字符串中使用变量，请将变量置于花括号。

在上面的例子中，PHP `mysql_fetch_array()`函数第二个参数为MYSQL_ASSOC， 设置该参数查询结果返回关联数组，你可以使用字段名称来作为数组的索引。

PHP提供了另外一个函数`mysql_fetch_assoc()`，该函数从结果集中取得一行作为关联数组。 返回根据从结果集取得的行生成的关联数组，如果没有更多行，则返回 `false`。

实例

尝试以下实例，该实例使用了`mysql_fetch_assoc()`函数来输出数据表runoob_tbl的所有记录：

```

1. <?php
2. $dbhost = 'localhost:3036';
3. $dbuser = 'root';
4. $dbpass = 'rootpassword';
5. $conn = mysql_connect($dbhost, $dbuser, $dbpass);
6. if(! $conn )
7. {
8.     die('Could not connect: ' . mysql_error());
9. }
10. $sql = 'SELECT runoob_id, runoob_title,
11.         runoob_author, submission_date
12.         FROM runoob_tbl';
13.

```

```

14. mysql_select_db('RUNOOB');
15. $retval = mysql_query( $sql, $conn );
16. if(! $retval )
17. {
18.     die('Could not get data: ' . mysql_error());
19. }
20. while($row = mysql_fetch_assoc($retval))
21. {
22.     echo "Tutorial ID :{$row['runoob_id']} <br> ".
23.         "Title: {$row['runoob_title']} <br> ".
24.         "Author: {$row['runoob_author']} <br> ".
25.         "Submission Date : {$row['submission_date']} <br> ".
26.         "-----<br>";
27. }
28. echo "Fetched data successfully\n";
29. mysql_close($conn);
30. ?>

```

你也可以使用常量 `MYSQL_NUM` 作为PHP `mysql_fetch_array()`函数的第二个参数，返回数字数组。

实例

以下实例使用`MYSQL_NUM`参数显示数据表`runoob_tbl`的所有记录：

```

1. <?php
2. $dbhost = 'localhost:3036';
3. $dbuser = 'root';
4. $dbpass = 'rootpassword';
5. $conn = mysql_connect($dbhost, $dbuser, $dbpass);
6. if(! $conn )
7. {
8.     die('Could not connect: ' . mysql_error());
9. }
10. $sql = 'SELECT runoob_id, runoob_title,
11.         runoob_author, submission_date
12.         FROM runoob_tbl';
13.
14. mysql_select_db('RUNOOB');
15. $retval = mysql_query( $sql, $conn );
16. if(! $retval )
17. {

```

```

18.     die('Could not get data: ' . mysql_error());
19. }
20. while($row = mysql_fetch_array($retval, MYSQL_NUM))
21. {
22.     echo "Tutorial ID :{$row[0]} <br> ".
23.         "Title: {$row[1]} <br> ".
24.         "Author: {$row[2]} <br> ".
25.         "Submission Date : {$row[3]} <br> ".
26.         "-----<br>";
27. }
28. echo "Fetched data successfully\n";
29. mysql_close($conn);
30. ?>

```

以上三个实例输出结果都一样。

内存释放

在我们执行完SELECT语句后，释放游标内存是一个很好的习惯。 。可以通过PHP函数 `mysql_free_result()` 来实现内存的释放。

以下实例演示了该函数的使用方法。

实例

尝试以下实例：

```

1.  <?php
2.  $dbhost = 'localhost:3036';
3.  $dbuser = 'root';
4.  $dbpass = 'rootpassword';
5.  $conn = mysql_connect($dbhost, $dbuser, $dbpass);
6.  if(! $conn )
7.  {
8.      die('Could not connect: ' . mysql_error());
9.  }
10. $sql = 'SELECT runoob_id, runoob_title,
11.         runoob_author, submission_date
12.         FROM runoob_tbl';
13.
14. mysql_select_db('RUNOOB');

```

```
15. $retval = mysql_query( $sql, $conn );
16. if( ! $retval )
17. {
18.     die('Could not get data: ' . mysql_error());
19. }
20. while($row = mysql_fetch_array($retval, MYSQL_NUM))
21. {
22.     echo "Tutorial ID :{$row[0]} <br> ".
23.         "Title: {$row[1]} <br> ".
24.         "Author: {$row[2]} <br> ".
25.         "Submission Date : {$row[3]} <br> ".
26.         "-----<br>";
27. }
28. mysql_free_result($retval);
29. echo "Fetched data successfully\n";
30. mysql_close($conn);
31. ?>
```

原文: <https://colaly.gitbooks.io/mysql/content/13mysql-cha-xun-shu-ju.html>

14.MySQL where 子句

MySQL where 子句

我们知道从MySQL表中使用SQL SELECT 语句来读取数据。

如需有条件地从表中选取数据，可将 WHERE 子句添加到 SELECT 语句中。

语法

以下是SQL SELECT 语句使用 WHERE 子句从数据表中读取数据的通用语法：

```
1. SELECT field1, field2,...fieldN FROM table_name1, table_name2...
2. [WHERE condition1 [AND [OR]] condition2.....
```

- 查询语句中你可以使用一个或者多个表，表之间使用逗号(,)分割，并使用WHERE语句来设定查询条件。
- 你可以在WHERE子句中指定任何条件。
- 你可以使用AND或者OR指定一个或多个条件。
- WHERE子句也可以运用于SQL的 DELETE 或者 UPDATE 命令。
- WHERE 子句类似于程序语言中的if条件，根据 MySQL 表中的字段值来读取指定的数据。

以下为操作符列表，可用于 WHERE 子句中。

下表中实例假定 A为10 B为20

操作符	描述	实例
=	等号，检测两个值是否相等，如果相等返回true	(A = B) 返回 false。
<>, !=	不等于，检测两个值是否相等，如果不相等返回true	(A != B) 返回 true。
>	大于号，检测左边的值是否大于右边的值，如果左边的值大于右边的值返回 true	(A > B) 返回 false。
<	小于号，检测左边的值是否小于右边的值，如果左边的值小于右边的值返回 true	(A < B) 返回 true。
>=	大于等于号，检测左边的值是否大于或等于右边的值，如果左边的值大于或等于右边的值返回true	(A >= B) 返回 false。
<=	小于等于号，检测左边的值是否小于或等于右边的值，如果左边的值小于或等于右边的值返回true	(A <= B) 返回 true。

如果我们想再MySQL数据表中读取指定的数据，WHERE 子句是非常有用的。

使用主键来作为 WHERE 子句的条件查询是非常快速的。

如果给定的条件在表中没有任何匹配的记录，那么查询不会返回任何数据。

从命令提示符中读取数据

我们将在SQL SELECT语句使用WHERE子句来读取MySQL数据表 runoob_tbl 中的数据：

实例

以下实例将读取 runoob_tbl 表中 runoob_author 字段值为 Sanjay 的所有记录：

```

1. root@host# mysql -u root -p password;
2. Enter password:*****
3. mysql> use RUNOOB;
4. Database changed
5. mysql> SELECT * from runoob_tbl WHERE runoob_author='Sanjay';
6. +-----+-----+-----+-----+
7. | runoob_id | runoob_title | runoob_author | submission_date |
8. +-----+-----+-----+-----+
9. |          3 | JAVA Tutorial | Sanjay         | 2007-05-21      |
10. +-----+-----+-----+-----+
11. 1 rows in set (0.01 sec)
12.
13. mysql>

```

MySQL的WHERE子句的字符串比较是不区分大小写的。 你可以使用 BINARY 关键字来设定WHERE子句的字符串比较是区分大小写的。

如下实例

```

1. root@host# mysql -u root -p password;
2. Enter password:*****
3. mysql> use RUNOOB;
4. Database changed
5. mysql> SELECT * from runoob_tbl \
6.          WHERE BINARY runoob_author='sanjay';
7. Empty set (0.02 sec)
8.
9. mysql>

```

使用PHP脚本读取数据

你可以使用PHP函数的mysql_query()及相同的SQL SELECT 带上 WHERE 子句的命令来获取数据。

该函数用于执行SQL命令，然后通过 PHP 函数 mysql_fetch_array() 来输出所有查询的数据。

实例

以下实例将从 runoob_tbl 表中返回使用 runoob_author 字段值为 Sanjay 的记录：

```

1.  <?php
2.  $dbhost = 'localhost:3036';
3.  $dbuser = 'root';
4.  $dbpass = 'rootpassword';
5.  $conn = mysql_connect($dbhost, $dbuser, $dbpass);
6.  if(! $conn )
7.  {
8.      die('Could not connect: ' . mysql_error());
9.  }
10. $sql = 'SELECT runoob_id, runoob_title,
11.         runoob_author, submission_date
12.         FROM runoob_tbl
13.         WHERE runoob_author="Sanjay"';
14.
15. mysql_select_db('RUNOOB');
16. $retval = mysql_query( $sql, $conn );
17. if(! $retval )
18. {
19.     die('Could not get data: ' . mysql_error());
20. }
21. while($row = mysql_fetch_array($retval, MYSQL_ASSOC))
22. {
23.     echo "Tutorial ID :{$row['runoob_id']} <br> ".
24.         "Title: {$row['runoob_title']} <br> ".
25.         "Author: {$row['runoob_author']} <br> ".
26.         "Submission Date : {$row['submission_date']} <br> ".
27.         "-----<br>";
28. }
29. echo "Fetched data successfully\n";
30. mysql_close($conn);
31. ?>

```

原文: <https://colaly.gitbooks.io/mysql/content/14mysql-where-zi-ju.html>

15.MySQL UPDATE 查询

MySQL UPDATE 查询

如果我们需要修改或更新MySQL中的数据，我们可以使用 SQL UPDATE 命令来操作。

语法

以下是 UPDATE 命令修改 MySQL 数据表数据的通用SQL语法：

```
1. UPDATE table_name SET field1=new-value1, field2=new-value2
2. [WHERE Clause]
```

- 你可以同时更新一个或多个字段。
- 你可以在 WHERE 子句中指定任何条件。
- 你可以在一个单独表中同时更新数据。

当你需要更新数据表中指定行的数据时 WHERE 子句是非常有用的。

通过命令提示符更新数据

以下我们将在 SQL UPDATE 命令使用 WHERE子句来更新runoob_tbl表中指定的数据：

实例

以下实例将更新数据表中 runoob_id 为 3 的 runoob_title 字段值：

```
1. root@host# mysql -u root -p password;
2. Enter password:*****
3. mysql> use RUNOOB;
4. Database changed
5. mysql> UPDATE runoob_tbl
6.     -> SET runoob_title='Learning JAVA'
7.     -> WHERE runoob_id=3;
8. Query OK, 1 row affected (0.04 sec)
9. Rows matched: 1  Changed: 1  Warnings: 0
10.
11. mysql>
```

使用PHP脚本更新数据

PHP中使用函数mysql_query()来执行SQL语句，你可以在SQL UPDATE语句中使用或者不适用WHERE子句。

该函数与在mysql>命令提示符中执行SQL语句的效果是一样的。

实例

以下实例将更新 runoob_id 为3的 runoob_title 字段的数据。

```
1.  <?php
2.  $dbhost = 'localhost:3036';
3.  $dbuser = 'root';
4.  $dbpass = 'rootpassword';
5.  $conn = mysql_connect($dbhost, $dbuser, $dbpass);
6.  if(! $conn )
7.  {
8.      die('Could not connect: ' . mysql_error());
9.  }
10. $sql = 'UPDATE runoob_tbl
11.         SET runoob_title="Learning JAVA"
12.         WHERE runoob_id=3';
13.
14. mysql_select_db('RUNOOB');
15. $retval = mysql_query( $sql, $conn );
16. if(! $retval )
17. {
18.     die('Could not update data: ' . mysql_error());
19. }
20. echo "Updated data successfully\n";
21. mysql_close($conn);
22. ?>
```

原文: <https://colalys.gitbooks.io/mysql/content/15mysql-update-cha-xun.html>

16.MySQL DELETE 语句

MySQL DELETE 语句

你可以使用 SQL 的 DELETE FROM 命令来删除 MySQL 数据表中的记录。

你可以在mysql>命令提示符或PHP脚本中执行该命令。

语法

以下是SQL DELETE 语句从MySQL数据表中删除数据的通用语法：

```
1. DELETE FROM table_name [WHERE Clause]
```

- 如果没有指定 WHERE 子句，MySQL表中的所有记录将被删除。
- 你可以在 WHERE 子句中指定任何条件
- 您可以在单个表中一次性删除记录。

当你想删除数据表中指定的记录时 WHERE 子句是非常有用的。

从命令行中删除数据

这里我们将在 SQL DELETE 命令中使用 WHERE 子句来删除MySQL数据表runoob_tbl所选的数据。

实例

以下实例将删除 runoob_tbl 表中 runoob_id 为3 的记录：

```
1. root@host# mysql -u root -p password;  
2. Enter password:*****  
3. mysql> use RUNOOB;  
4. Database changed  
5. mysql> DELETE FROM runoob_tbl WHERE runoob_id=3;  
6. Query OK, 1 row affected (0.23 sec)  
7.  
8. mysql>
```

使用 PHP 脚本删除数据

PHP使用 `mysql_query()` 函数来执行SQL语句， 你可以在SQL `DELETE`命令中使用或不使用 `WHERE` 子句。

该函数与 `mysql>`命令符执行SQL命令的效果是一样的。

实例

以下PHP实例将删除runoob_tbl表中runoob_id为3的记录：

```
1. <?php
2. $dbhost = 'localhost:3036';
3. $dbuser = 'root';
4. $dbpass = 'rootpassword';
5. $conn = mysql_connect($dbhost, $dbuser, $dbpass);
6. if(! $conn )
7. {
8.     die('Could not connect: ' . mysql_error());
9. }
10. $sql = 'DELETE FROM runoob_tbl
11.         WHERE runoob_id=3';
12.
13. mysql_select_db('RUNOOB');
14. $retval = mysql_query( $sql, $conn );
15. if(! $retval )
16. {
17.     die('Could not delete data: ' . mysql_error());
18. }
19. echo "Deleted data successfully\n";
20. mysql_close($conn);
21. ?>
```

原文： <https://colaly.gitbooks.io/mysql/content/16mysql-delete-yu-ju.html>

17.MySQL LIKE 子句

MySQL LIKE 子句

我们知道在MySQL中使用 SQL SELECT 命令来读取数据，同时我们可以在 SELECT 语句中使用 WHERE 子句来获取指定的记录。

WHERE 子句中可以使用等号 (=) 来设定获取数据的条件，如 "runoob_author = 'Sanjay'"。

但是有时候我们需要获取 runoob_author 字段含有 "jay" 字符的所有记录，这时我们就需要在 WHERE 子句中使用 SQL LIKE 子句。

SQL LIKE 子句中使用百分号(%)字符来表示任意字符，类似于UNIX或正则表达式中的星号 (*)。

如果没有使用百分号(%), LIKE 子句与等号(=)的效果是一样的。

语法

以下是SQL SELECT 语句使用 LIKE 子句从数据表中读取数据的通用语法：

```
1. SELECT field1, field2,...fieldN table_name1, table_name2...
2. WHERE field1 LIKE condition1 [AND [OR]] filed2 = 'somevalue'
```

- 你可以在WHERE子句中指定任何条件。
- 你可以在WHERE子句中使用LIKE子句。
- 你可以使用LIKE子句代替等号(=)。
- LIKE 通常与 % 一同使用，类似于一个元字符的搜索。
- 你可以使用AND或者OR指定一个或多个条件。
- 你可以在 DELETE 或 UPDATE 命令中使用 WHERE...LIKE 子句来指定条件。

在命令提示符中使用 LIKE 子句

以下我们将在 SQL SELECT 命令中使用 WHERE...LIKE 子句来从MySQL数据表 runoob_tbl 中读取数据。

实例

以下是我们将runoob_tbl表中获取runoob_author字段中以"jay"为结尾的所有记录：

```

1. root@host# mysql -u root -p password;
2. Enter password:*****
3. mysql> use RUNOOB;
4. Database changed
5. mysql> SELECT * from runoob_tbl
6.      -> WHERE runoob_author LIKE '%jay';
7. +-----+-----+-----+-----+
8. | runoob_id | runoob_title | runoob_author | submission_date |
9. +-----+-----+-----+-----+
10. |          3 | JAVA Tutorial | Sanjay        | 2007-05-21      |
11. +-----+-----+-----+-----+
12. 1 rows in set (0.01 sec)
13.
14. mysql>

```

在PHP脚本中使用 LIKE 子句

你可以使用PHP函数的mysql_query()及相同的SQL SELECT 带上 WHERE...LIKE 子句的命令来获取数据。

该函数用于执行SQL命令，然后通过 PHP 函数 mysql_fetch_array() 来输出所有查询的数据。

但是如果是DELETE或者UPDATE中使用 WHERE...LIKE 子句的SQL语句，则无需使用 mysql_fetch_array() 函数。

实例

以下是我们使用PHP脚本在runoob_tbl表中读取runoob_author字段中以"jay"为结尾的所有记录：

```

1. <?php
2. $dbhost = 'localhost:3036';
3. $dbuser = 'root';
4. $dbpass = 'rootpassword';
5. $conn = mysql_connect($dbhost, $dbuser, $dbpass);
6. if(! $conn )
7. {
8.     die('Could not connect: ' . mysql_error());
9. }
10. $sql = 'SELECT runoob_id, runoob_title,
11.         runoob_author, submission_date

```

```
12.         FROM runoob_tbl
13.         WHERE runoob_author LIKE "%jay";
14.
15. mysql_select_db('RUNOOB');
16. $retval = mysql_query( $sql, $conn );
17. if(! $retval )
18. {
19.     die('Could not get data: ' . mysql_error());
20. }
21. while($row = mysql_fetch_array($retval, MYSQL_ASSOC))
22. {
23.     echo "Tutorial ID :{$row['runoob_id']} <br> ".
24.         "Title: {$row['runoob_title']} <br> ".
25.         "Author: {$row['runoob_author']} <br> ".
26.         "Submission Date : {$row['submission_date']} <br> ".
27.         "-----<br>";
28. }
29. echo "Fetched data successfully\n";
30. mysql_close($conn);
31. ?>
```

原文: <https://colalyl.gitbooks.io/mysql/content/17mysql-like-zh-zu.html>

18.MySQL UNION 操作符

MySQL UNION 操作符

本教程为大家介绍 MySQL UNION 操作符的语法和实例。

描述

MySQL UNION 操作符用于连接两个以上的 SELECT 语句的结果组合到一个结果集合中。多个 SELECT 语句会删除重复的数据。

语法

MySQL UNION 操作符语法格式：

```
1. SELECT expression1, expression2, ... expression_n
2. FROM tables
3. [WHERE conditions]
4. UNION [ALL | DISTINCT]
5. SELECT expression1, expression2, ... expression_n
6. FROM tables
7. [WHERE conditions];
```

参数

- **expression1, expression2, ... expression_n**: 要检索的列。
- **tables**: 要检索的数据表。
- **WHERE conditions**: 可选， 检索条件。
- **DISTINCT**: 可选，删除结果集中重复的数据。默认情况下 UNION 操作符已经删除了重复数据，所以 DISTINCT 修饰符对结果没啥影响。
- **ALL**: 可选，返回所有结果集，包含重复数据。

演示数据库

在本教程中，我们将使用 RUNOOB 样本数据库。

下面是选自 "Websites" 表的数据：

```
1. mysql> SELECT * FROM Websites;
2. +-----+-----+-----+-----+-----+
3. | id | name          | url                               | alexa | country |
4. +-----+-----+-----+-----+-----+
5. | 1  | Google        | https://www.google.cm/          | 1     | USA     |
6. | 2  | 淘宝          | https://www.taobao.com/         | 13    | CN      |
7. | 3  | 菜鸟教程      | http://www.runoob.com/          | 4689  | CN      |
8. | 4  | 微博          | http://weibo.com/               | 20    | CN      |
9. | 5  | Facebook      | https://www.facebook.com/       | 3     | USA     |
10. | 7  | stackoverflow | http://stackoverflow.com/       | 0     | IND     |
11. +-----+-----+-----+-----+-----+
```

下面是 "apps" APP 的数据：

```
1. mysql> SELECT * FROM apps;
2. +-----+-----+-----+-----+
3. | id | app_name      | url                               | country |
4. +-----+-----+-----+-----+
5. | 1  | QQ APP       | http://im.qq.com/               | CN      |
6. | 2  | 微博 APP    | http://weibo.com/               | CN      |
7. | 3  | 淘宝 APP    | https://www.taobao.com/         | CN      |
8. +-----+-----+-----+-----+
9. 3 rows in set (0.00 sec)
```

SQL UNION 实例

下面的 SQL 语句从 "Websites" 和 "apps" 表中选取所有不同的country (只有不同的值)：

实例

```
SELECT country FROM Websites
```

```
UNION
```

```
SELECT country FROM apps
```

```
ORDER BY country;
```

执行以上 SQL 输出结果如下：

```
mysql> SELECT country FROM Websites
-> UNION
-> SELECT country FROM apps
-> ORDER BY country;
```

```
+-----+
| country |
+-----+
| CN      |
| IND     |
| USA     |
+-----+
3 rows in set (0.00 sec)
```

注释：UNION 不能用于列出两个表中所有的country。如果一些网站和APP来自同一个国家，每个国家只会列出一行。UNION 只会选取不同的值。请使用 UNION ALL 来选取重复的值！

SQL UNION ALL 实例

下面的 SQL 语句使用 UNION ALL 从 "Websites" 和 "apps" 表中选取所有的country（也有重复的值）：

实例

```
SELECT country FROM Websites
```

```
UNION ALL
```

```
SELECT country FROM apps
```

```
ORDER BY country;
```

执行以上 SQL 输出结果如下：

```
mysql> SELECT country FROM Websites
-> UNION ALL
-> SELECT country FROM apps
-> ORDER BY country;
```

```
+-----+
| country |
+-----+
| CN      |
| CN      |
| CN      |
| CN      |
| CN      |
| IND     |
| USA     |
| USA     |
| USA     |
+-----+
```

帶有 WHERE 的 SQL UNION ALL

下面的 SQL 语句使用 UNION ALL 从 "Websites" 和 "apps" 表中选取所有的中国(CN)的数据（也有重复的值）：

实例

```
SELECT country, name FROM Websites
```

```
WHERE country='CN'
```

```
UNION ALL
```

```
SELECT country, app_name FROM apps
```

```
WHERE country='CN'
```

```
ORDER BY country;
```

执行以上 SQL 输出结果如下：

```
mysql> SELECT country, name FROM Websites
WHERE country='CN'
UNION ALL
SELECT country, app_name FROM apps
WHERE country='CN'
ORDER BY country;
+-----+-----+
| country | name          |
+-----+-----+
| CN      | 淘宝          |
| CN      | QQ APP        |
| CN      | 菜鸟教程      |
| CN      | 微博 APP      |
| CN      | 微博          |
| CN      | 淘宝 APP      |
+-----+-----+
6 rows in set (0.05 sec)
```

原文: <https://colaly.gitbooks.io/mysql/content/18mysql-union-cao-zuo-fu.html>

19.MySQL 排序

MySQL 排序

我们知道从 MySQL 表中使用 SQL SELECT 语句来读取数据。

如果我们需要对读取的数据进行排序，我们就可以使用 MySQL 的 **ORDER BY** 子句来设定你想按哪个字段哪种方式来进行排序，再返回搜索结果。

本章节使用的数据库结构及数据下载：[RUNOOB.sql](#)。

语法

以下是 SQL SELECT 语句使用 ORDER BY 子句将查询数据排序后再返回数据：

```
1. SELECT field1, field2,...fieldN table_name1, table_name2...
2. ORDER BY field1, [field2...] [ASC [DESC]]
```

- 你可以使用任何字段来作为排序的条件，从而返回排序后的查询结果。
- 你可以设定多个字段来排序。
- 你可以使用 ASC 或 DESC 关键字来设置查询结果是按升序或降序排列。默认情况下，它是按升序排列。
- 你可以添加 WHERE...LIKE 子句来设置条件。

在命令提示符中使用 ORDER BY 子句

以下将在 SQL SELECT 语句中使用 ORDER BY 子句来读取MySQL 数据表 runoob_tbl 中的数据：

实例

尝试以下实例，结果将按升序排列

```
1. root@host# mysql -u root -p password;
2. Enter password:*****
3. mysql> use RUNOOB;
4. Database changed
5. mysql> SELECT * from runoob_tbl ORDER BY runoob_author ASC;
```

```

6.  +-----+-----+-----+-----+
7.  | runoob_id | runoob_title | runoob_author | submission_date |
8.  +-----+-----+-----+-----+
9.  |          2 | Learn MySQL | Abdul S       | 2007-05-24      |
10. |          1 | Learn PHP   | John Poul     | 2007-05-24      |
11. |          3 | JAVA Tutorial | Sanjay        | 2007-05-06      |
12.  +-----+-----+-----+-----+
13. 3 rows in set (0.00 sec)
14.
15. mysql> SELECT * from runoob_tbl ORDER BY runoob_author DESC;
16.  +-----+-----+-----+-----+
17.  | runoob_id | runoob_title | runoob_author | submission_date |
18.  +-----+-----+-----+-----+
19.  |          3 | JAVA Tutorial | Sanjay        | 2007-05-06      |
20.  |          1 | Learn PHP     | John Poul     | 2007-05-24      |
21.  |          2 | Learn MySQL   | Abdul S       | 2007-05-24      |
22.  +-----+-----+-----+-----+
23. 3 rows in set (0.00 sec)
24.
25. mysql>

```

读取 runoob_tbl 表中所有数据并按 runoob_author 字段的升序排列。

在PHP脚本中使用 ORDER BY 子句

你可以使用PHP函数的mysql_query()及相同的SQL SELECT 带上 ORDER BY 子句的命令来获取数据。该函数用于执行SQL命令，然后通过 PHP 函数 mysql_fetch_array() 来输出所有查询的数据。

实例

尝试以下实例，查询后的数据按 runoob_author 字段的降序排列后返回。

```

1. <?php
2. $dbhost = 'localhost:3036';
3. $dbuser = 'root';
4. $dbpass = 'rootpassword';
5. $conn = mysql_connect($dbhost, $dbuser, $dbpass);
6. if(! $conn )
7. {
8.     die('Could not connect: ' . mysql_error());

```

```
9.  }
10. $sql = 'SELECT runoob_id, runoob_title,
11.         runoob_author, submission_date
12.         FROM runoob_tbl
13.         ORDER BY runoob_author DESC';
14.
15. mysql_select_db('RUNOOB');
16. $retval = mysql_query( $sql, $conn );
17. if(! $retval )
18. {
19.     die('Could not get data: ' . mysql_error());
20. }
21. while($row = mysql_fetch_array($retval, MYSQL_ASSOC))
22. {
23.     echo "Tutorial ID :{$row['runoob_id']} <br> ".
24.         "Title: {$row['runoob_title']} <br> ".
25.         "Author: {$row['runoob_author']} <br> ".
26.         "Submission Date : {$row['submission_date']} <br> ".
27.         "-----<br>";
28. }
29. echo "Fetched data successfully\n";
30. mysql_close($conn);
31. ?>
```

原文: <https://colaly.gitbooks.io/mysql/content/19mysql-pai-xu.html>

20.MySQL GROUP BY 语句

MySQL GROUP BY 语句

GROUP BY 语句根据一个或多个列对结果集进行分组。

在分组的列上我们可以使用 COUNT, SUM, AVG, 等函数。

GROUP BY 语法

```
1. SELECT column_name, function(column_name)
2. FROM table_name
3. WHERE column_name operator value
4. GROUP BY column_name;
```

实例演示

本章节实例使用到了以下表结构及数据，使用前我们可以先将以下数据导入数据库中。

```
1. SET NAMES utf8;
2. SET FOREIGN_KEY_CHECKS = 0;
3.
4. -- -----
5. -- Table structure for `employee_tbl`
6. -- -----
7. DROP TABLE IF EXISTS `employee_tbl`;
8. CREATE TABLE `employee_tbl` (
9.   `id` int(11) NOT NULL,
10.  `name` char(10) NOT NULL DEFAULT '',
11.  `date` datetime NOT NULL,
12.  `singin` tinyint(4) NOT NULL DEFAULT '0' COMMENT '登录次数',
13.  PRIMARY KEY (`id`)
14. ) ENGINE=InnoDB DEFAULT CHARSET=utf8;
15.
16. -- -----
17. -- Records of `employee_tbl`
18. -- -----
19. BEGIN;
```

```

20. INSERT INTO `employee_tbl` VALUES
21. ('1', '小明', '2016-04-22 15:25:33', '1'),
22. ('2', '小王', '2016-04-20 15:25:47', '3'),
23. ('3', '小丽', '2016-04-19 15:26:02', '2'),
24. ('4', '小王', '2016-04-07 15:26:14', '4'),
25. ('5', '小明', '2016-04-11 15:26:40', '4'),
26. ('6', '小明', '2016-04-04 15:26:54', '2');
27. COMMIT;
28.
29. SET FOREIGN_KEY_CHECKS = 1;

```

导入成功后，执行以下 SQL 语句：

```

1. mysql> set names utf8;
2. mysql> SELECT * FROM employee_tbl;
3. +----+-----+-----+-----+
4. | id | name  | date                | singin |
5. +----+-----+-----+-----+
6. |  1 | 小明 | 2016-04-22 15:25:33 |      1 |
7. |  2 | 小王 | 2016-04-20 15:25:47 |      3 |
8. |  3 | 小丽 | 2016-04-19 15:26:02 |      2 |
9. |  4 | 小王 | 2016-04-07 15:26:14 |      4 |
10. |  5 | 小明 | 2016-04-11 15:26:40 |      4 |
11. |  6 | 小明 | 2016-04-04 15:26:54 |      2 |
12. +----+-----+-----+-----+
13. 6 rows in set (0.00 sec)

```

接下来我们使用 GROUP BY 语句 将数据表按名字进行分组，并统计每个人有多少条记录：

```

1. mysql> SELECT name, COUNT(*) FROM employee_tbl GROUP BY name;
2. +-----+-----+
3. | name  | COUNT(*) |
4. +-----+-----+
5. | 小丽 |      1 |
6. | 小明 |      3 |
7. | 小王 |      2 |
8. +-----+-----+
9. 3 rows in set (0.01 sec)

```

使用 WITH ROLLUP

WITH ROLLUP 可以实现在分组统计数据基础上再进行相同的统计 (SUM, AVG, COUNT...)。

例如我们将以上的数据表按名字进行分组，再统计每个人登录的次数：

```
1. mysql> SELECT name, SUM(singin) as singin_count FROM employee_tbl GROUP BY
   name WITH ROLLUP;
2. +-----+-----+
3. | name   | singin_count |
4. +-----+-----+
5. | 小丽   | 2           |
6. | 小明   | 7           |
7. | 小王   | 7           |
8. | NULL   | 16          |
9. +-----+-----+
10. 4 rows in set (0.00 sec)
```

其中记录 NULL 表示所有人的登录次数。

我们可以使用 coalesce 来设置一个可以取代 NULL 的名称，coalesce 语法：

```
1. select coalesce(a,b,c);
```

参数说明：如果a=null,则选择b；如果b=null,则选择c；如果a!=null,则选择a；如果a b c都为null，则返回为null（没意义）。

以下实例中如果名字为空我们使用总数代替：

```
1. mysql> SELECT coalesce(name, '总数'), SUM(singin) as singin_count FROM
   employee_tbl GROUP BY name WITH ROLLUP;
2. +-----+-----+
3. | coalesce(name, '总数') | singin_count |
4. +-----+-----+
5. | 小丽                   | 2           |
6. | 小明                   | 7           |
7. | 小王                   | 7           |
8. | 总数                   | 16          |
9. +-----+-----+
10. 4 rows in set (0.01 sec)
```

原文：<https://colaly.gitbooks.io/mysql/content/20mysql-group-by-yu-ju.html>