

## Mysql 常用命令

show databases; 显示数据库

create database name; 创建数据库

use databasename; 选择数据库

drop database name 直接删除数据库，不提醒

show tables; 显示表

describe tablename; 显示具体的表结构

select 中加上 distinct 去除重复字段

mysqladmin drop databasename 删除数据库前，有提示。

显示当前 mysql 版本和当前日期

select version(),current\_date;

修改 mysql 中 root 的密码:

shell>mysql -h localhost -u root -p //登录

mysql> update user set password=password("xueok654123") where user='root';

mysql> flush privileges //刷新数据库

mysql>use dbname; 打开数据库:

mysql>show databases; 显示所有数据库

mysql>show tables; 显示数据库 mysql 中所有的表: 先 use mysql;然后

mysql>describe user; 显示表 mysql 数据库中 user 表的列信息);

grant

创建用户 firstdb(密码 firstdb)和数据库，并赋予权限于 firstdb 数据库

mysql> create database firstdb;

mysql> grant all on firstdb.\* to firstdb identified by 'firstdb'

会自动创建用户 firstdb

mysql 默认的是本地主机是 localhost,对应的 IP 地址就是 127.0.0.1, 所以你用你的 IP 地址登录会出错, 如果你想用你的 IP 地址登录就要先进行授权用 grant 命令。

mysql>grant all on \*.\* to [root@202.116.39.2](#) identified by "123456";

说明:grant 与 on 之间是各种权限, 例如:insert,select,update 等

on 之后是数据库名和表名,第一个\*表示所有的数据库, 第二个\*表示所有的表

root 可以改成你的用户名, @后可以跟域名或 IP 地址, identified by 后面的是登录用的密码, 可以省略, 即缺省密码或者叫空密码。

drop database firstdb;

创建一个可以从任何地方连接服务器的一个完全的超级用户, 但是必须使用一个口令 something 做这个

mysql> grant all privileges on \*.\* to [user@localhost](#) identified by 'something' with

增加新用户

格式: grant select on 数据库.\* to 用户名@登录主机 identified by "密码"

GRANT ALL PRIVILEGES ON \*.\* TO [monty@localhost](#) IDENTIFIED BY 'something' WITH GRANT OPTION;

GRANT ALL PRIVILEGES ON \*.\* TO [mailto:monty@](#) IDENTIFIED BY 'something' WITH GRANT OPTION;

删除授权:

```
mysql> revoke all privileges on *.* from mailto:root@;
```

```
mysql> delete from user where user="root" and host="%";
```

```
mysql> flush privileges;
```

创建一个用户 custom 在特定客户端 it363.com 登录, 可访问特定数据库 fangchandb

```
mysql >grant select, insert, update, delete, create,drop on fangchandb.* to custom@ it363.com identified by ' passwd'
```

重命名表:

```
mysql > alter table t1 rename t2;
```

mysqldump

备份数据库

```
shell> mysqldump -h host -u root -p dbname >dbname_backup.sql
```

恢复数据库

```
shell> mysqladmin -h myhost -u root -p create dbname
```

```
shell> mysqldump -h host -u root -p dbname < dbname_backup.sql
```

如果只想卸出建表指令, 则命令如下:

```
shell> mysqladmin -u root -p -d databasename > a.sql
```

如果只想卸出插入数据的 sql 命令, 而不需要建表命令, 则命令如下:

```
shell> mysqladmin -u root -p -t databasename > a.sql
```

那么如果我只想要数据, 而不想要什么 sql 命令时, 应该如何操作呢?

mysqldump -T./ phptest driver

其中, 只有指定了-T 参数才可以卸出纯文本文件, 表示卸出数据的目录, ./表示当前目录, 即与 mysqldump 同一目录。如果不指定 driver 表, 则将卸出整个数据库的数据。每个表会生成两个文件, 一个为.sql 文件, 包含建表执行。另一个为.txt 文件, 只包含数据, 且没有 sql 指令。

可将查询存储在一个文件中并告诉 mysql 从文件中读取查询而不是等待键盘输入。可利用外壳程序键入重定向实用程序来完成这项工作。例如, 如果在文件 my\_file.sql 中存放有查

询, 可如下执行这些查询:

例如, 如果您想将建表语句提前写在 sql.txt 中,

```
mysql > mysql -h myhost -u root -p
```

MySQL5.0 支持的字符集

MySQL 中的字符集控制做得比较细, 可以分为数据库级, 表级, 字段级(这一点和 ORACLE 不同)。我上次改的字符集是数据库级的, 对表 sysuser 没有影响, 所以出现了改了字符集却一样无法插入中文的情况。

```
Drop TABLE IF EXISTS `firstdb`.`users`;
```

```
Create TABLE `firstdb`.`users` (  
  `id` int(11) NOT NULL auto_increment,  
  `username` varchar(40) default NULL,  
  `birthday` date default NULL,  
  PRIMARY KEY (`id`)  
) ENGINE=InnoDB DEFAULT CHARSET=gb2312;
```

编译 MySQL 时，指定了一个默认的字符集，这个字符集是 latin1;

安装 MySQL 时，可以在配置文件 (my.ini) 中指定一个默认的字符集，如果没指定，这个值继承自编译时指定的;

启动 mysqld 时，可以在命令行参数中指定一个默认的字符集，如果没指定，这个值继承自配置文件中的;

此时 character\_set\_server 被设定为这个默认的字符集;

当创建一个新的数据库时，除非明确指定，这个数据库的字符集被缺省设定为 character\_set\_server;

当选定了一个数据库时，character\_set\_database 被设定为这个数据库默认的字符集;

在这个数据库里创建一张表时，表默认的字符集被设定为 character\_set\_database，也就是这个数据库默认的字符集;

当在表内设置一栏时，除非明确指定，否则此栏缺省的字符集就是表默认的字符集;

这个字符集就是数据库中实际存储数据采用的字符集，mysqldump 出来的内容就是这个字符集下的;Query Browser1.1 对中文输入的支持太差劲了，可以用 notebook 写好后，再 copy 过去执行  
update firstdb.users set username='以' where id=3;

## MYSQL 常用命令

### 1.导出整个数据库

mysqldump -u 用户名 -p --default-character-set=latin1 数据库名 > 导出的  
文件名(数据库默认编码是 latin1)

```
mysqldump -u wcnc -p smgp_apps_wcnc > wcnc.sql
```

### 2.导出一个表

mysqldump -u 用户名 -p 数据库名 表名> 导出的文件名

```
mysqldump -u wcnc -p smgp_apps_wcnc users> wcnc_users.sql
```

### 3.导出一个数据库结构

```
mysqldump -u wcnc -p -d -add-drop-table smgp_apps_wcnc >d:wcnc_db.sql
```

-d 没有数据 -add-drop-table 在每个 create 语句之前增加一个 drop table

### 4.导入数据库

A:常用 source 命令

进入 mysql 数据库控制台，

如 `mysql -u root -p`

`mysql>use 数据库`

然后使用 source 命令，后面参数为脚本文件(如这里用到的.sql)

`mysql>source wcnc_db.sql`

B:使用 mysqldump 命令

`mysqldump -u username -p dbname < filename.sql`

C:使用 mysql 命令

`mysql -u username -p -D dbname < filename.sql`

## 一、启动与退出

1、进入 MySQL: 启动 MySQL Command Line Client(MySQL 的 DOS 界面)，直接输入安装时的密码即可。此时的提示符是: `mysql>`

2、退出 MySQL: `quit` 或 `exit`

## 二、库操作

### 1、创建数据库

命令: `create database <数据库名>`

例如: 建立一个名为 `xhkdb` 的数据库

`mysql> create database xhkdb;`

### 2、显示所有的数据库

命令: `show databases` (注意: 最后有个 s)

`mysql> show databases;`

### 3、删除数据库

命令: `drop database <数据库名>`

例如: 删除名为 `xhkdb` 的数据库

`mysql> drop database xhkdb;`

### 4、连接数据库

命令: `use <数据库名>`

例如: 如果 `xhkdb` 数据库存在，尝试存取它:

`mysql> use xhkdb;`

屏幕提示: `Database changed`

### 5、查看当前使用的数据库

`mysql> select database();`

### 6、当前数据库包含的表信息:

mysql> show tables; (注意：最后有个 s)

### 三、表操作，操作之前应连接某个数据库

#### 1、建表

命令：create table <表名> ( <字段名 1> <类型 1> [...<字段名 n> <类型 n>]);

```
mysql> create table MyClass(  
> id int(4) not null primary key auto_increment,  
> name char(20) not null,  
> sex int(4) not null default '0',  
> degree double(16,2));
```

#### 2、获取表结构

命令： desc 表名， 或者 show columns from 表名

```
mysql> DESCRIBE MyClass  
mysql> desc MyClass;  
mysql> show columns from MyClass;
```

#### 3、删除表

命令： drop table <表名>

例如：删除表名为 MyClass 的表

```
mysql> drop table MyClass;
```

#### 4、插入数据

命令： insert into <表名> [( <字段名 1>[...<字段名 n > ])] values ( 值 1 ),  
( 值 n )]

例如，往表 MyClass 中插入二条记录，这二条记录表示：编号为 1 的名为 Tom 的成绩为 96.45，编号为 2 的名为 Joan 的成绩为 82.99，编号为 3 的名为 Wang 的成绩为 96.5。

```
mysql> insert into MyClass values(1,'Tom',96.45),(2,'Joan',82.99),  
(2,'Wang', 96.59);
```

#### 5、查询表中的数据

##### 1)、查询所有行

命令： select <字段 1, 字段 2, ...> from < 表名 > where < 表达式 >

例如：查看表 MyClass 中所有数据

```
mysql> select * from MyClass;
```

##### 2)、查询前几行数据

例如：查看表 MyClass 中前 2 行数据

```
mysql> select * from MyClass order by id limit 0,2;
```

或者：

```
mysql> select * from MyClass limit 0,2;
```

## 6、删除表中数据

命令: `delete from 表名 where 表达式`

例如: 删除表 `MyClass` 中编号为 1 的记录

```
mysql> delete from MyClass where id=1;
```

## 7、修改表中数据: `update 表名 set 字段=新值,... where 条件`

```
mysql> update MyClass set name='Mary' where id=1;
```

## 7、在表中增加字段:

命令: `alter table 表名 add 字段 类型 其他;`

例如: 在表 `MyClass` 中添加了一个字段 `passtest`, 类型为 `int(4)`, 默认值为 0

```
mysql> alter table MyClass add passtest int(4) default '0'
```

## 8、更改表名:

命令: `rename table 原表名 to 新表名;`

例如: 在表 `MyClass` 名字更改为 `YouClass`

```
mysql> rename table MyClass to YouClass;
```

更新字段内容

```
update 表名 set 字段名 = 新内容
```

```
update 表名 set 字段名 = replace(字段名,'旧内容','新内容');
```

文章前面加入 4 个空格

```
update article set content=concat('    ',content);
```

字段类型

1.`INT[(M)]` 型: 正常大小整数类型

2.`DOUBLE[(M,D)] [ZEROFILL]` 型: 正常大小(双精密)浮点数字类型

3.`DATE` 日期类型: 支持的范围是 1000-01-01 到 9999-12-31。MySQL 以 `YYYY-MM-DD` 格式来显示 `DATE` 值, 但是允许你使用字符串或数字把值赋给 `DATE` 列

4.`CHAR(M)` 型: 定长字符串类型, 当存储时, 总是是用空格填满右边到指定的长度

5.`BLOB TEXT` 类型, 最大长度为  $65535(2^{16}-1)$  个字符。

6.`VARCHAR` 型: 变长字符串类型

## 5.导入数据库表

(1)创建.sql 文件

(2)先产生一个库如 `auction.c:mysqlbin>mysqladmin -u root -p creat auction`, 会提示输入密码, 然后成功创建。

(2)导入 `auction.sql` 文件

```
c:mysqlbin>mysql -u root -p auction < auction.sql。
```

通过以上操作, 就可以创建了一个数据库 `auction` 以及其中的一个表 `auction`

。

## 6.修改数据库

(1)在 mysql 的表中增加字段:

```
alter table dbname add column userid int(11) not null primary key  
auto_increment;
```

这样,就在表 dbname 中添加了一个字段 userid,类型为 int(11)。

7.mysql 数据库的授权

```
mysql>grant select,insert,delete,create,drop  
on *.* (或 test.*/user.*/*.)  
to 用户名@localhost  
identified by '密码';
```

如:新建一个用户帐号以便可以访问数据库,需要进行如下操作:

```
mysql> grant usage
```

```
-> ON test.*
```

```
-> TO testuser@localhost;
```

```
Query OK, 0 rows affected (0.15 sec)
```

此后就创建了一个新用户叫: testuser, 这个用户只能从 localhost 连接到数据库并可以连接到 test 数据库。下一步,我们必须指定 testuser 这个用户可以执行哪些操作:

```
mysql> GRANT select, insert, delete,update
```

```
-> ON test.*
```

```
-> TO testuser@localhost;
```

```
Query OK, 0 rows affected (0.00 sec)
```

此操作使 testuser 能够在每一个 test 数据库中的表执行 Select, Insert 和 Delete 以及 Update 查询操作。现在我们结束操作并退出 MySQL 客户程序:

```
mysql> exit
```

Bye9!

1:使用 SHOW 语句找出在服务器上当前存在什么数据库:

```
mysql> SHOW DATABASES;
```

2:2、创建一个数据库 MYSQLDATA

```
mysql> Create DATABASE MYSQLDATA;
```

3:选择你所创建的数据库

```
mysql> USE MYSQLDATA; (按回车键出现 Database changed 时说明操作成功!)
```

4:查看现在的数据库中是否存在什么表

```
mysql> SHOW TABLES;
```

5:创建一个数据库表

```
mysql> Create TABLE MYTABLE (name VARCHAR(20), sex CHAR(1));
```

6:显示表的结构:

```
mysql> DESCRIBE MYTABLE;
```

7:往表中加入记录

```
mysql> insert into MYTABLE values ("hyq","M");
```

8:用文本方式将数据装入数据库表中(例如 D:/mysql.txt)

```
mysql> LOAD DATA LOCAL INFILE "D:/mysql.txt" INTO TABLE MYTABLE;
```

9:导入.sql 文件命令(例如 D:/mysql.sql)

```
mysql>use database;
```

```
mysql>source d:/mysql.sql;
```

10:删除表

```
mysql>drop TABLE MYTABLE;
```

11:清空表

```
mysql>delete from MYTABLE;
```

12:更新表中数据

```
mysql>update MYTABLE set sex="f" where name='hyq';
```

以下是无意中在网络看到的使用 **MySql** 的管理心得,

在 windows 中 **MySql** 以服务形式存在, 在使用前应确保此服务已经启动, 未启动可用 `net start mysql` 命令启动。而 Linux 中启动时可用 `/etc/rc.d/init.d/mysqld start` 命令, 注意启动者应具有管理员权限。

刚安装好的 **MySql** 包含一个含空密码的 **root** 帐户和一个匿名帐户, 这是很大的安全隐患, 对于一些重要的应用我们应将安全性尽可能提高, 在这里应把匿名帐户删除、**root** 帐户设置密码, 可用如下命令进行:

```
use mysql;
```

```
delete from User where User="";
```

```
update User set Password=PASSWORD('newpassword') where User='root';
```

如果要对用户所用的登录终端进行限制, 可以更新 **User** 表中相应用户的 **Host** 字段, 在进行了以上更改后应重新启动数据库服务, 此时登录时可用如下类似命令:

```
mysql -uroot -p;
```

```
mysql -uroot -pnewpassword;
```

```
mysql mydb -uroot -p;
```

```
mysql mydb -uroot -pnewpassword;
```

上面命令参数是常用参数的一部分, 详细情况可参考文档。此处的 **mydb** 是要登录的数据库的名称。

在进行开发和实际应用中, 用户不应该只用 **root** 用户进行连接数据库, 虽然使用 **root** 用户进行测试时很方便, 但会给系统带来重大安全隐患, 也不利于管理技术的提高。我们给一个应用中使用的用户赋予最恰当的数据库权限。如一个只进行数据



插入的用户不应赋予其删除数据的权限。MySQL 的用户管理是通过 **User** 表来实现的，添加新用户常用的方法有两个，一是在 **User** 表插入相应的数据行，同时设置相应的权限;二是通过 **GRANT** 命令创建具有某种权限的用户。其中 **GRANT** 的常用用法如下：

```
grant all on mydb.* to NewUserName@HostName identified by "password" ;
grant usage on *.* to NewUserName@HostName identified by "password";
grant select,insert,update on mydb.* to NewUserName@HostName identified
by "password";
grant update,delete on mydb.TestTable to NewUserName@HostName identified
by "password";
```

若要给此用户赋予他在相应对象上的权限的管理能力，可在 **GRANT** 后面添加 **WITH GRANT OPTION** 选项。而对于用插入 **User** 表添加的用户，**Password** 字段应用 **PASSWORD** 函数进行更新加密，以防不轨之人窃看密码。对于那些已经不用的用户应给予清除，权限过界的用户应及时回收权限，回收权限可以通过更新 **User** 表相应字段，也可以使用 **REVOKE** 操作。

下面给出本人获得的对常用权限的解释：

全局管理权限：

**FILE**: 在 MySQL 服务器上读写文件。

**PROCESS**: 显示或杀死属于其它用户的服务线程。

**RELOAD**: 重载访问控制表，刷新日志等。

**SHUTDOWN**: 关闭 MySQL 服务。

数据库/数据表/数据列权限：

**Alter**: 修改已存在的数据表(例如增加/删除列)和索引。

**Create**: 建立新的数据库或数据表。

**Delete**: 删除表的记录。

**Drop**: 删除数据表或数据库。

**INDEX**: 建立或删除索引。

**Insert**: 增加表的记录。

**Select**: 显示/搜索表的记录。

**Update**: 修改表中已存在的记录。

特别的权限：

**ALL**: 允许做任何事(和 root 一样)。

**USAGE**: 只允许登录--其它什么也不允许做。

-----

## MYSQL 常用命令

有很多朋友虽然安装好了 `mysql` 但却不知如何使用它。在这篇文章中我们就从连接 `MYSQL`、修改密码、增加用户等方面来学习一些 `MYSQL` 的常用命令。

有很多朋友虽然安装好了 `mysql` 但却不知如何使用它。在这篇文章中我们就从连接 `MYSQL`、修改密码、增加用户等方面来学习一些 `MYSQL` 的常用命令。

### 一、连接 MYSQL

格式: `mysql -h 主机地址 -u 用户名 -p 用户密码`

#### 1、例 1: 连接到本机上的 `MYSQL`

首先在打开 `DOS` 窗口, 然后进入目录 `mysqlbin`, 再键入命令 `mysql -uroot -p`, 回车后提示你输密码, 如果刚安装好 `MYSQL`, 超级用户 `root` 是没有密码的, 故直接回车即可进入到 `MYSQL` 中了, `MYSQL` 的提示符是: `mysql>`

#### 2、例 2: 连接到远程主机上的 `MYSQL`

假设远程主机的 IP 为: `110.110.110.110`, 用户名为 `root`, 密码为 `abcd123`。则键入以下命令:

```
mysql -h110.110.110.110 -uroot -pabcd123
```

(注: `u` 与 `root` 可以不用加空格, 其它也一样)

#### 3、退出 `MYSQL` 命令: `exit` (回车)

### 二、修改密码

格式: `mysqladmin -u 用户名 -p 旧密码 password 新密码`

1、例 1: 给 `root` 加个密码 `ab12`。首先在 `DOS` 下进入目录 `mysqlbin`, 然后键入以下命令

```
mysqladmin -uroot -password ab12
```

注: 因为开始时 `root` 没有密码, 所以 `-p` 旧密码一项就可以省略了。

#### 2、例 2: 再将 `root` 的密码改为 `djg345`

```
mysqladmin -uroot -pab12 password djg345
```

### MYSQL 常用命令(下)

#### 一、操作技巧

1、如果你打命令时, 回车后发现忘记加分号, 你无须重打一遍命令, 只要打个分号回车就可以了。也就是说你可以把一个完整的命令分成几行来打, 完后用分号作结束标志就 `OK`。

2、你可以使用光标上下键调出以前的命令。但以前我用过的一个 `MYSQL` 旧版本不支持。我现在用的是 `mysql-3.23.27-beta-win`。

#### 二、显示命令

##### 1、显示数据库列表。

```
show databases;
```

刚开始时才两个数据库: `mysql` 和 `test`。`mysql` 库很重要它里面有 `MYSQL` 的系统信息, 我们改密码和新增用户, 实际上就是用这个库进行操作。

2、显示库中的数据表:

`use mysql;` //打开库, 学过 FOXBASE 的一定不会陌生吧

`show tables;`

3、显示数据表的结构:

`describe` 表名;

4、建库:

`create database` 库名;

5、建表:

`use` 库名;

`create table` 表名 (字段设定列表);

6、删库和删表:

`drop database` 库名;

`drop table` 表名;

7、将表中记录清空:

`delete from` 表名;

8、显示表中的记录:

`select * from` 表名;

三、一个建库和建表以及插入数据的实例

`drop database if exists school;` //如果存在 SCHOOL 则删除

`create database school;` //建立库 SCHOOL

`use school;` //打开库 SCHOOL

`create table teacher` //建立表 TEACHER

(

`id int(3) auto_increment not null primary key,`

`name char(10) not null,`

`address varchar(50) default '深圳',`

`year date`

); //建表结束

//以下为插入字段

`insert into teacher values('','glchengang','深圳一中','1976-10-10');`

`insert into teacher values('','jack','深圳一中','1975-12-23');`

注: 在建表中(1)将 ID 设为长度为 3 的数字字段:int(3)并让它每个记录自动加一:auto\_increment 并不能为空:not null 而且让他成为主字段 primary key

(2)将 NAME 设为长度为 10 的字符字段

(3)将 ADDRESS 设为长度 50 的字符字段, 而且缺省值为深圳。varchar 和 char 有什么区别呢, 只有等以后的文章再说了。

(4)将 YEAR 设为日期字段。

如果你在 **mysql** 提示符键入上面的命令也可以，但不方便调试。你可以将以上命令原样写入一个文本文件中假设为 **school.sql**，然后复制到 **c:\**下，并在 **DOS** 状态进入目录**mysql\bin**，然后键入以下命令：

```
mysql -uroot -p 密码 < c:\school.sql
```

如果成功，空出一行无任何显示；如有错误，会有提示。（以上命令已经调试，你只要将//的注释去掉即可使用）。

四、将文本数据转到数据库中

1、文本数据应符合的格式：字段数据之间用 **tab** 键隔开，**null** 值用**\n** 来代替。

例：

```
3 rose 深圳二中 1976-10-10
```

```
4 mike 深圳一中 1975-12-23
```

2、数据传入命令 **load data local infile "文件名" into table** 表名；

注意：你最好将文件复制到**mysql\bin** 目录下，并且要先用 **use** 命令打表所在的库。

五、备份数据库：（命令在 **DOS** 的**mysql\bin** 目录下执行）

```
mysqldump --opt school>school.bbb
```

注释：将数据库 **school** 备份到 **school.bbb** 文件，**school.bbb** 是一个文本文件，文件名任取，打开看看你会有新发现。

一.Select 语句的完整语法为：

```
Select[ALL|DISTINCT|DISTINCTROW|TOP]
{*[table.*|[table.]field1[AS alias1],[table.]field2[AS alias2],[...]]}
FROM tableexpression[,...][IN externaldatabase]
[Where...]
[GROUP BY...]
[HAVING...]
[ORDER BY...]
[WITH OWNERACCESS OPTION]
```

说明：

用中括号(**[]**)括起来的部分表示是可选的，用大括号(**{}**)括起来的部分是表示必须从中选择其中的一个。

1 **FROM** 子句

**FROM** 子句指定了 **Select** 语句中字段的来源。**FROM** 子句后面是包含一个或多个的表达式(由逗号分开)，其中的表达式可为单一表名称、已保存的查询或由 **INNER JOIN**、**LEFT JOIN** 或 **RIGHT JOIN** 得到的复合结果。如果表或查询存储在外部数据库，在 **IN** 子句之后指明其完整路径。

例：下列 **SQL** 语句返回所有有定单的客户：

```
Select orderID, Customer.customerID
```

FROM orders Customers

Where orders.CustomerID=Customers.CustomeersID

2 ALL、DISTINCT、DISTINCTROW、TOP 谓词

(1) ALL 返回满足 SQL 语句条件的所有记录。如果没有指明这个谓词，默认为 ALL。

例：Select ALL FirstName,LastName

FROM Employees

(2) DISTINCT 如果有多个记录的选择字段的数据相同，只返回一个。

(3) DISTINCTROW 如果有重复的记录，只返回一个

(4) TOP 显示查询头尾若干记录。也可返回记录的百分比，这是要用 TOP N

PERCENT 子句(其中 N 表示百分比)

例：返回 5%定货额最大的定单

Select TOP 5 PERCENT\*

FROM [ order Details]

orDER BY UnitPrice\*Quantity\*(1-Discout) DESC

3 用 AS 子句为字段取别名

如果想为返回的列取一个新的标题，或者，经过对字段的计算或总结之后，产生了一个新的值，希望把它放到一个新的列里显示，则用 AS 保留。

例：返回 FirstName 字段取别名为 NickName

Select FirstName AS NickName ,LastName ,City

FROM Employees

例：返回新的一列显示库存价值

Select ProductName ,UnitPrice ,UnitsInStock ,UnitPrice\*UnitsInStock AS  
valueInStock

FROM Products

二 .Where 子句指定查询条件

1 比较运算符

比较运算符 含义

= 等于

> 大于

< 小于

>= 大于等于

<= 小于等于

<> 不等于

!> 不大于

!< 不小于

例：返回 96 年 1 月的定单

Select orderID, CustomerID, orderDate

FROM orders

Where orderDate>#1/1/96# AND orderDate<#1/30/96#

注意:

Microsoft JET SQL 中, 日期用'#'定界。日期也可以用 Datevalue()函数来代替。

在比较字符型的数据时, 要加上单引号", 尾空格在比较中被忽略。

例:

Where orderDate>#96-1-1#

也可以表示为:

Where orderDate>Datevalue('1/1/96')

使用 NOT 表达式求反。

例: 查看 96 年 1 月 1 日以后的定单

Where Not orderDate<=#1/1/96#

## 2 范围(BETWEEN 和 NOT BETWEEN)

BETWEEN ...AND...运算符指定了要搜索的一个闭区间。

例: 返回 96 年 1 月到 96 年 2 月的定单。

Where orderDate Between #1/1/96# And #2/1/96#

## 3 列表(IN , NOT IN)

IN 运算符用来匹配列表中的任何一个值。IN 子句可以代替用 OR 子句连接的一连串的条件。

例: 要找出住在 London、Paris 或 Berlin 的所有客户

Select CustomerID, CompanyName, ContactName, City

FROM Customers

Where City In('London',' Paris',' Berlin')

## 4 模式匹配(LIKE)

LIKE 运算符检验一个包含字符串数据的字段值是否匹配一指定模式。

LIKE 运算符里使用的通配符

通配符 含义

? 任何一个单一的字符

\* 任意长度的字符

# 0~9 之间的单一数字

[字符列表] 在字符列表里的任一值

[!字符列表] 不在字符列表里的任一值

- 指定字符范围, 两边的值分别为其上下限

例: 返回邮政编码在(171)555-0000 到(171)555-9999 之间的客户

Select CustomerID ,CompanyName,City,Phone

FROM Customers

Where Phone Like '(171)555-####'

LIKE 运算符的一些样式及含义

样式 含义 不符合

LIKE 'A\*' A 后跟任意长度的字符 Bc,c255

LIKE '5[\*]' 5\*5 555

LIKE '5?5' 5 与 5 之间有任何一个字符 55,5wer5

LIKE '5##5' 5235, 5005 5kd5,5346

LIKE '[a-z]' a-z 间的任意一个字符 5,%

LIKE '[!0-9]' 非 0-9 间的任意一个字符 0,1

LIKE '[ ]' 1,\*

三 .用 ORDER BY 子句排序结果

orDER 子句按一个或多个(最多 16 个)字段排序查询结果，可以是升序(ASC)也可以是降序(DESC)，缺省是升序。ORDER 子句通常放在 SQL 语句的最后。

orDER 子句中定义了多个字段，则按照字段的先后顺序排序。

例：

```
Select ProductName,UnitPrice, UnitInStock
```

```
FROM Products
```

```
orDER BY UnitInStock DESC , UnitPrice DESC, ProductName
```

orDER BY 子句中可以用字段在选择列表中的位置号代替字段名，可以混合字段名和位置号。

例：下面的语句产生与上列相同的效果。

```
Select ProductName,UnitPrice, UnitInStock
```

```
FROM Products
```

```
orDER BY 1 DESC , 2 DESC,3
```

四 .运用连接关系实现多表查询

例：找出同一个城市中供应商和客户的名字

```
Select Customers.CompanyName, Suppliers.ComPany.Name
```

```
FROM Customers, Suppliers
```

```
Where Customers.City=Suppliers.City
```

例：找出产品库存量大于同一种产品的定单的数量的产品和定单

```
Select ProductName,OrderID, UnitInStock, Quantity
```

```
FROM Products, [Order Deails]
```

```
Where Product.productID=[Order Details].ProductID
```

```
AND UnitsInStock>Quantity
```

另一种方法是用 Microsof JET SQL 独有的 JNNER JOIN

语法：

```
FROM table1 INNER JOIN table2
```

```
ON table1.field1 comparision table2.field2
```

其中 comparison 就是前面 Where 子句用到的比较运算符。

```
Select FirstName,lastName,OrderID,CustomerID,OrderDate  
FROM Employees
```

```
INNER JOIN orders ON Employees.EmployeeID=Orders.EmployeeID
```

注意:

INNER JOIN 不能连接 Memo OLE Object Single Double 数据类型字段。

在一个 JOIN 语句中连接多个 ON 子句

语法:

```
Select fields
```

```
FROM table1 INNER JOIN table2
```

```
ON table1.field1 compopr table2.field1 AND
```

```
ON table1.field2 compopr table2.field2 or
```

```
ON table1.field3 compopr table2.field3
```

也可以

```
Select fields
```

```
FROM table1 INNER JOIN
```

```
(table2 INNER JOIN [( ]table3
```

```
[INNER JOER] [( ]tablex[INNER JOIN]
```

```
ON table1.field1 compopr table2.field1
```

```
ON table1.field2 compopr table2.field2
```

```
ON table1.field3 compopr table2.field3
```

外部连接返回更多记录，在结果中保留不匹配的记录，不管存不存在满足条件的记录都要返回另一侧的所有记录。

```
FROM table [LEFT|RIGHT]JOIN table2
```

```
ON table1.field1comparison table.field2
```

用左连接来建立外部连接，在表达式的左边的表会显示其所有的数据

例：不管有没有定货量，返回所有商品

```
Select ProductName ,OrderID
```

```
FROM Products
```

```
LEFT JOIN orders ON Products.PrductsID=Orders.ProductID
```

右连接与左连接的差别在于：不管左侧表里有没有匹配的记录，它都从左侧表中返回所有记录。

例：如果想了解客户的信息，并统计各个地区的客户分布，这时可以用一个右连接，即使某个地区没有客户，也要返回客户信息。

空值不会相互匹配，可以通过外连接才能测试被连接的某个表的字段是否有空值。

```
Select *
```

```
FROM talbe1
```



LEFT JOIN table2 ON table1.a=table2.c

1 连接查询中使用 Iif 函数实现以 0 值显示空值

Iif 表达式: Iif(IsNull(Amount,0,Amout)

例: 无论定货大于或小于 ¥50, 都要返回一个标志。

Iif([Amount]>50,?Big order?,?Small order?)

五. 分组和总结查询结果

在 SQL 的语法里, GROUP BY 和 HAVING 子句用来对数据进行汇总。GROUP BY 子句指明了按照哪几个字段来分组, 而将记录分组后, 用 HAVING 子句过滤这些记录。

GROUP BY 子句的语法

Select fiddlist

FROM table

Where criteria

[GROUP BY groupfieldlist [HAVING groupcriteria]]

注: Microsoft Jet 数据库 Jet 不能对备注或 OLE 对象字段分组。

GROUP BY 字段中的 Null 值以备分组但是不能被省略。

在任何 SQL 合计函数中不计算 Null 值。

GROUP BY 子句后最多可以带有十个字段, 排序优先级按从左到右的顺序排列。

例: 在'WA'地区的雇员表中按头衔分组后, 找出具有同等头衔的雇员数目大于 1 人的所有头衔。

Select Title ,Count(Title) as Total

FROM Employees

Where Region = 'WA'

GROUP BY Title

HAVING Count(Title)>1

JET SQL 中的聚积函数

聚集函数 意义

SUM ( ) 求和

AVG ( ) 平均值

COUNT ( ) 表达式中记录的数目

COUNT (\*) 计算记录的数目

MAX 最大值

MIN 最小值

VAR 方差

STDEV 标准误差

FIRST 第一个值

LAST 最后一个值

六. 用 Parameters 声明创建参数查询

Parameters 声明的语法:

**PARAMETERS** name datatype[,name datatype[, ...]]

其中 name 是参数的标志符,可以通过标志符引用参数.

Datatype 说明参数的数据类型.

使用时要把 **PARAMETERS** 声明置于任何其他语句之前.

例:

**PARAMETERS**[Low price] Currency,[Beginning date]datetime

**Select** orderID ,OrderAmount

**FROM** orders

**Where** orderAMount>[low price]

**AND** orderDate>=[Beginning date]

## 七. 功能查询

所谓功能查询,实际上是一种操作查询,它可以对数据库进行快速高效的操作.它以选择查询为目的,挑选出符合条件的数据,再对数据进行批处理.功能查询包括更新查询,删除查询,添加查询,和生成表查询.

### 1 更新查询

**Update** 子句可以同时更改一个或多个表中的数据.它也可以同时更改多个字段的值.

更新查询语法:

**Update** 表名

**SET** 新值

**Where** 准则

例:英国客户的定货量增加 5%,货运量增加 3%

**Update** OEDERS

**SET** orderAmount = orderAmount \*1.1

**Freight** = Freight\*1.03

**Where** ShipCountry = 'UK'

### 2 删除查询

**Delete** 子句可以使用户删除大量的过时的或冗于的数据.

注:删除查询的对象是整个记录.

**Delete** 子句的语法:

**Delete** [表名.\*]

**FROM** 来源表

**Where** 准则

例: 要删除所有 94 年前的定单

**Delete** \*

**FROM** orders

**Where** orderData<#94-1-1#

### 3 追加查询

Insert 子句可以将一个或一组记录追加到一个或多个表的尾部.

INTO 子句指定接受新记录的表

valueS 关键字指定新记录所包含的数据值.

Insert 子句的语法:

INSETR INTO 目的表或查询(字段 1,字段 2,...)

valueS(数值 1,数值 2,...)

例:增加一个客户

Insert INTO Employees(FirstName,LastName,title)

valueS('Harry','Washington','Trainee')

### 4 生成表查询

可以一次性地把所有满足条件的记录拷贝到一张新表中.通常制作记录的备份或副本或作为报表的基础.

Select INTO 子句用来创建生成表查询语法:

Select 字段 1,字段 2,...

INTO 新表[IN 外部数据库]

FROM 来源数据库

Where 准则

例:为订单制作一个存档备份

Select \*

INTO ordersArchive

FROM orders

### 八. 联合查询

UNION 运算可以把多个查询的结果合并到一个结果集里显示.

UNION 运算的一般语法:

[表]查询 1 UNION [ALL]查询 2 UNION ...

例:返回巴西所有供给商和客户的名字和城市

Select CompanyName,City

FROM Suppliers

Where Country = 'Brazil'

UNION

Select CompanyName,City

FROM Customers

Where Country = 'Brazil'

注:

缺省的情况下,UNION 子句不返回重复的记录.如果想显示所有记录,可以加 ALL 选项

UNION 运算要求查询具有相同数目的字段.但是,字段数据类型不必相同.

每一个查询参数中可以使用 **GROUP BY** 子句 或 **HAVING** 子句进行分组.要想以指定的顺序来显示返回的数据,可以在最后一个查询的尾部使用 **ORDER BY** 子句.

## 九. 交叉查询

交叉查询可以对数据进行总和,平均,计数或其他总和计算法的计算,这些数据通过两种信息进行分组:一个显示在表的左部,另一个显示在表的顶部.

Microsoft Jet SQL 用 **TRANSFORM** 语句创建交叉表查询语法:

**TRANSFORM** aggfunction

**Select** 语句

**GROUP BY** 子句

**PIVOT** pivotfield[**IN**(value1 [,value2[,...]])]

Aggfunction 指 SQL 聚积函数,

**Select** 语句选择作为标题的的字段,

**GROUP BY** 分组

说明:

**Pivotfield** 在查询结果集中创建列标题时用的字段或表达式,用可选的 **IN** 子句限制它的取值.

**value** 代表创建列标题的固定值.

例:显示在 1996 年里每一季度每一位员工所接的定单的数目:

**TRANSFORM** Count(**OrderID**)

**Select** FirstName&"&LastName AS FullName

**FROM** Employees **INNER JOIN** orders

**ON** Employees.EmployeeID = orders.EmployeeID

**Where** DatePart("yyyy",OrderDate)= '1996'

**GROUP BY** FirstName&"&LastName

**ORDER BY** FirstName&"&LastName

**POVOT** DatePart("q",OrderDate)&'季度'

## 十. 子查询

子查询可以理解为 套查询.子查询是一个 **Select** 语句.

1 表达式的值与子查询返回的单一值做比较

语法:

表达式 **comparision** [**ANY**|**ALL**|**SOME**](子查询)

说明:

**ANY** 和 **SOME** 谓词是同义词,与比较运算符(=,<,>,<=,>=)一起使用.返回一个布尔值 **True** 或 **False**.**ANY** 的意思是,表达式与子查询返回的一系列的值逐一比较,只要其中的一次比较产生 **True** 结果,**ANY** 测试的返回 **True** 值(既 **Where** 子句的结果),对应于该表达式的当前记录将进入主查询的结果中.**ALL** 测试则要求表达式与子查询返回的一系列的值的比较都产生 **True** 结果,才回返回 **True** 值.

例:主查询返回单价比任何一个折扣大于等于 25%的产品的单价要高的所有产品

```
Select * FROM Products
```

```
Where UnitPrice>ANY
```

```
(Select UnitPrice FROM[Order Details] Where Discount>0.25)
```

2 检查表达式的值是否匹配子查询返回的一组值的某个值

语法:

```
[NOT]IN(子查询)
```

例:返回库存价值大于等于 1000 的产品.

```
Select ProductName FROM Products
```

```
Where ProductID IN
```

```
(Select PrdoctID FROM [Order DEtails]
```

```
Where UnitPrice*Quantity>= 1000)
```

3 检测子查询是否返回任何记录

语法:

```
[NOT]EXISTS (子查询)
```

例:用 EXISTS 检索英国的客户

```
Select ComPanyName,ContactName
```

```
FROM orders
```

```
Where EXISTS
```

```
(Select *
```

```
FROM Customers
```

```
Where Country = 'UK' AND
```

```
Customers.CustomerID= orders.CustomerID)
```

1:使用 SHOW 语句找出在服务器上当前存在什么数据库:

```
mysql> SHOW DATABASES;
```

2:2、创建一个数据库 MYSQLDATA

```
mysql> Create DATABASE MYSQLDATA;
```

3:选择你所创建的数据库

```
mysql> USE MYSQLDATA; (按回车键出现 Database changed 时说明操作成功!)
```

4:查看现在的数据库中是否存在什么表

```
mysql> SHOW TABLES;
```

5:创建一个数据库表

```
mysql> Create TABLE MYTABLE (name VARCHAR(20), sex CHAR(1));
```

6:显示表的结构:

```
mysql> DESCRIBE MYTABLE;
```

7:往表中加入记录

```
mysql> insert into MYTABLE values ("hyq","M");
```

8:用文本方式将数据装入数据库表中(例如 D:/mysql.txt)

```
mysql> LOAD DATA LOCAL INFILE "D:/mysql.txt" INTO TABLE MYTABLE;
```

9:导入.sql 文件命令(例如 D:/mysql.sql)

```
mysql>use database;
```

```
mysql>source d:/mysql.sql;
```

10:删除表

```
mysql>drop TABLE MYTABLE;
```

11:清空表

```
mysql>delete from MYTABLE;
```

12:更新表中数据

```
mysql>update MYTABLE set sex="f" where name='hyq';
```

以下是无意中在网络看到的使用 MySQL 的管理心得,

摘自:[http://www1.xjtusky.com/article/htmldata/2004\\_12/3/57/article\\_1060\\_1.html](http://www1.xjtusky.com/article/htmldata/2004_12/3/57/article_1060_1.html)

在 windows 中 MySQL 以服务形式存在, 在使用前应确保此服务已经启动, 未启动可用 `net start mysql` 命令启动。而 Linux 中启动时可用“`/etc/rc.d/init.d/mysqld start`”命令, 注意启动者应具有管理员权限。

刚安装好的 MySQL 包含一个含空密码的 root 帐户和一个匿名帐户, 这是很大的安全隐患, 对于一些重要的应用我们应将安全性尽可能提高, 在这里应把匿名帐户删除、 root 帐户设置密码, 可用如下命令进行:

```
use mysql;
```

```
delete from User where User="";
```

```
update User set Password=PASSWORD('newpassword') where User='root';
```

如果要对用户所用的登录终端进行限制, 可以更新 User 表中相应用户的 Host 字段, 在进行了以上更改后应重新启动数据库服务, 此时登录时可用如下类似命令:

```
mysql -uroot -p;
```

```
mysql -uroot -pnewpassword;
```

```
mysql mydb -uroot -p;
```

```
mysql mydb -uroot -pnewpassword;
```

上面命令参数是常用参数的一部分, 详细情况可参考文档。此处的 mydb 是要登录的数据库的名称。

在进行开发和实际应用中, 用户不应该只用 root 用户进行连接数据库, 虽然使用 root 用户进行测试时很方便, 但会给系统带来重大安全隐患, 也不利于管理技术的提高。我们给一个应用中使用的用户赋予最恰当的数据库权限。如一个只进行数据插入的用户不应赋予其删除数据的权限。MySQL 的用户管理是通过 User 表来实现的, 添加新用户常用的方法有两个, 一是在 User 表插入相应的数据行, 同时设置相应的权限;二是通过 GRANT 命令创建具有某种权限的用户。其中 GRANT 的常用用法如下:

```
grant all on mydb.* to NewUserName@HostName identified by "password" ;
```

```
grant usage on *.* to NewUserName@HostName identified by "password";
```

```
grant select,insert,update on mydb.* to NewUserName@HostName identified by "password";  
grant update,delete on mydb.TestTable to NewUserName@HostName identified by  
"password";
```

若要给此用户赋予他在相应对象上的权限的管理能力，可在 GRANT 后面添加 WITH GRANT OPTION 选项。而对于用插入 User 表添加的用户，Password 字段应用 PASSWORD 函数进行更新加密，以防不轨之人窃看密码。对于那些已经不用的用户应给予清除，权限过界的用户应及时回收权限，回收权限可以通过更新 User 表相应字段，也可以使用 REVOKE 操作。

下面给出本人从其它资料(<http://www.cn-java.com/>)获得的对常用权限的解释：

全局管理权限：

FILE: 在 MySQL 服务器上读写文件。

PROCESS: 显示或杀死属于其它用户的服务线程。

RELOAD: 重载访问控制表，刷新日志等。

SHUTDOWN: 关闭 MySQL 服务。

数据库/数据表/数据列权限：

Alter: 修改已存在的数据表(例如增加/删除列)和索引。

Create: 建立新的数据库或数据表。

Delete: 删除表的记录。

Drop: 删除数据表或数据库。

INDEX: 建立或删除索引。

Insert: 增加表的记录。

Select: 显示/搜索表的记录。

Update: 修改表中已存在的记录。

特别的权限：

ALL: 允许做任何事(和 root 一样)。

USAGE: 只允许登录--其它什么也不允许做。

## 1、MySQL 常用命令

create database name; 创建数据库

use databasename; 选择数据库

drop database name 直接删除数据库，不提醒

show tables; 显示表

describe tablename; 表的详细描述

select 中加上 distinct 去除重复字段

mysqladmin drop databasename 删除数据库前，有提示。

显示当前 mysql 版本和当前日期

```
select version(),current_date;
```

## 2、修改 mysql 中 root 的密码：

```
shell>mysql -u root -p
```

```
mysql> update user set password=password("xueok654123") where user='root';
```

```
mysql> flush privileges //刷新数据库
```

```
mysql> use dbname; 打开数据库:
```

```
mysql> show databases; 显示所有数据库
```

```
mysql> show tables; 显示数据库 mysql 中所有的表: 先 use mysql;然后
```

```
mysql> describe user; 显示表 mysql 数据库中 user 表的列信息);
```

### 3、grant

创建一个可以从任何地方连接服务器的一个完全的超级用户，但是必须使用一个口令 something 做这个

```
mysql> grant all privileges on *.* to user@localhost identified by 'something' with
```

增加新用户

格式: grant select on 数据库.\* to 用户名@登录主机 identified by “密码”

```
GRANT ALL PRIVILEGES ON *.* TO monty@localhost IDENTIFIED BY 'something' WITH  
GRANT OPTION;
```

```
GRANT ALL PRIVILEGES ON *.* TO mailto:monty@ IDENTIFIED BY 'something' WITH  
GRANT OPTION;
```

删除授权:

```
mysql> revoke all privileges on *.* from mailto:root@;
```

```
mysql> delete from user where user="root" and host="%";
```

```
mysql> flush privileges;
```

创建一个用户 custom 在特定客户端 it363.com 登录，可访问特定数据库 fangchandb

```
mysql > grant select, insert, update, delete, create, drop on fangchandb.* to custom@ it363.com  
identified by ' passwd'
```

重命名表:

```
mysql > alter table t1 rename t2;
```

### 4、mysqldump

备份数据库

```
shell> mysqldump -h host -u root -p dbname > dbname_backup.sql
```

恢复数据库

```
shell> mysqladmin -h myhost -u root -p create dbname
```

```
shell> mysqldump -h host -u root -p dbname < dbname_backup.sql
```

如果只想卸出建表指令，则命令如下:

```
shell> mysqladmin -u root -p -d databasename > a.sql
```

如果只想卸出插入数据的 sql 命令，而不需要建表命令，则命令如下:

```
shell> mysqladmin -u root -p -t databasename > a.sql
```

那么如果我只想要数据，而不想要什么 sql 命令时，应该如何操作呢?

```
mysqldump -T/ phptest driver
```



其中，只有指定了-T 参数才可以卸出纯文本文件，表示卸出数据的目录，./表示当前目录，即与mysqldump 同一目录。如果不指定 driver 表，则将卸出整个数据库的数据。每个表会生成两个文件，一个为.sql 文件，包含建表执行。另一个为.txt 文件，只包含数据，且没有 sql 指令。

5、可将查询存储在一个文件中并告诉 mysql 从文件中读取查询而不是等待键盘输入。可利用外壳程序键入重定向实用程序来完成这项工作。例如，如果在文件 my\_file.sql 中存放有查

询，可如下执行这些查询：

例如，如果您想将建表语句提前写在 sql.txt 中：

```
mysql > mysql -h myhost -u root -p database < sql.txt
```

```
// 启动服务
```

```
mysqld --console
```

```
// 停止服务
```

```
mysqladmin -u root shutdown
```

```
// 登录后使用数据库 mysql
```

```
mysql -u root -p mysql
```

```
mysql -u root -p -h 11.11.11.11 database
```

```
// 创建数据库
```

```
create database db_name [default character set=gbk]
```

```
// 设置数据库默认字符集
```

```
alter database db_name default character set gbk
```

```
// 更换数据库 use database test after log on
```

```
use test
```

```
// 创建一个带图像字段的表 create a table mypic to store picture
```

```
create table mypic (picid int, picname varchar(20), content blob);
```

```
// 显示表的结构 describe table mypic
```

```
desc mypic
```

```
// 显示当前表的建表语句
```

```
show create table table_name
```

```
// 更改表类型
```

```
alter table table_name engine innodb|myisam|memory
```

```
// 插入一条记录 insert a record
```

```
insert into mypic values (1, '第二章', 0x2134545);
```

```
// 显示当前用户 show current user
```

```
select user();
```

```
// 显示当前用户密码 show current password
```

```
select password('root');
```

```
// 显示当前日期 show current date
```

```
select now();
```

```

// 更改用户密码 change user password
update user set password=password('xxx') where user='root';
// 分配用户权限 grant
grant all privileges on *.* toroot@localhost
grant select,insert,delete,update,alter,create,drop on lybbs.* mailto:tolybbs@" identified by
"lybbs";
grant select,insert,delete,update,alter,create,drop on lybbs.* tolybbs@localhostidentified by
"lybbs";
// 在不重启的情况下刷新用户权限 flush privileges
flush privileges
// 向表中增加一个主键 add primary key
alter table mypic add primary key (picid)
// 修改表结构增加一个新的字段 add a new column userid after picid
alter table mypic add column userid int after picid
// 更改列类型,当存储图像过大时, 使用默认 blob 超不过 100k
alter table userpic change image image longblob;
alter table userpic modify image longblob;
// 设置默认字符集为 gb2312
mysql --default-character-set=gb2312
// 显示详细信息, 包括字符集编码
show full columns from userpic;
// 改变表的编码
Alter TABLE userpic CHARACTER SET gb2312;
// mysql jdbc 连接 url 使用中文
jdbc:mysql://localhost/test?useUnicode=true&characterEncoding=gb2312
// 执行外部脚本
source

```

source

MySQL 是最受欢迎的开源 SQL 数据库管理系统, 由 MySQL AB 开发、发布和支持。MySQL AB 是一家基于 MySQL 开发人员的商业公司, 是一家使用了一种成功的商业模式来结合开源价值和方\*\*\*\*的第二代开源公司。MySQL 是 MySQL AB 的注册商标。

MySQL 是一个快速的、多线程、多用户和健壮的 SQL 数据库服务器。MySQL 服务器支持关键任务、重负载生产系统的使用, 也可以将它嵌入到一个大配置(mass-deployed)的软件中去。