## Url:

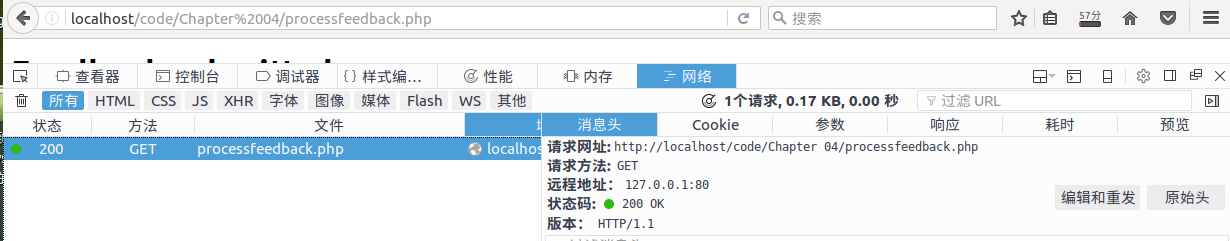
Url中带有空格时,服务器可能无法识别请求, 通常需要将url中的空格转成%20。

如：localhost/code/Chapter 04/processfeedback.php。空格被转成了+导致服务器无法识别出这个请求的url。

https://www.baidu.com/baidu?wd=localhost/code/Chapter+04/processfeedback.php&tn=monline\_dg&ie=utf-8

正确时：localhost/code/Chapter%2004/processfeedback.php

->http://localhost/code/Chapter 04/processfeedback.php

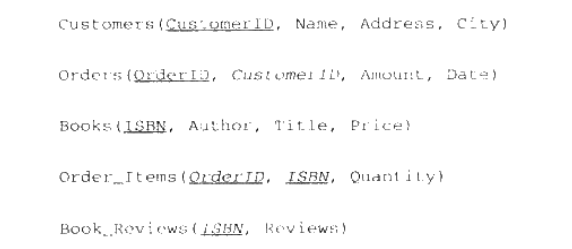




mysql用户：gjy\_book <---> gjy\_book, bookorama <---> bookorama123。

数据库：books。

grant select,insert,update,delete,index,alter,create,drop on books.\* to bookorama identified by 'bookorama123';



magic\_quotes\_gpc设置是否自动为GPC(get,post,cookie)传来的数据中的'"\加上反斜线。可以用get\_magic\_quotes\_gpc()检测系统设置。如果没有打开这项设置，可以使用addslashes()函数添加，它的功能就是给数据库查询语句等的需要在某些字符前加上了反斜线。这些字符是单引号（'）、双引号（"）、反斜线（\）与 NUL（NULL 字符）。

一般用法如下：

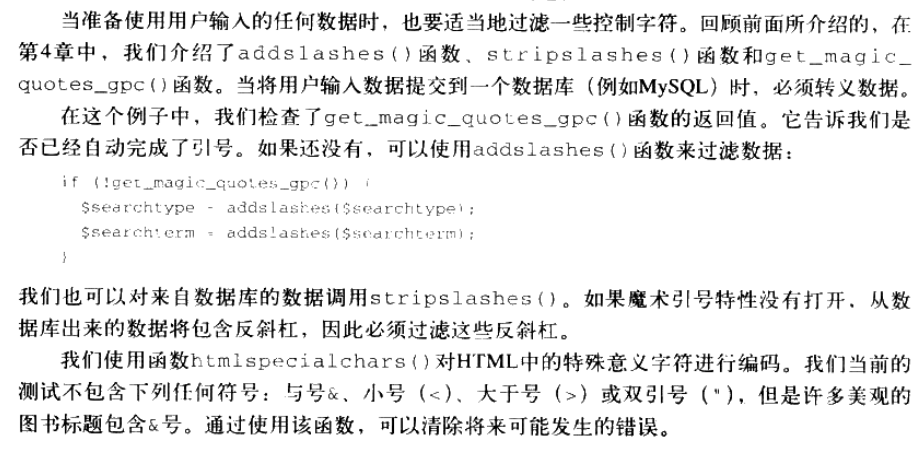
if(!get\_magic\_quotes\_gpc())

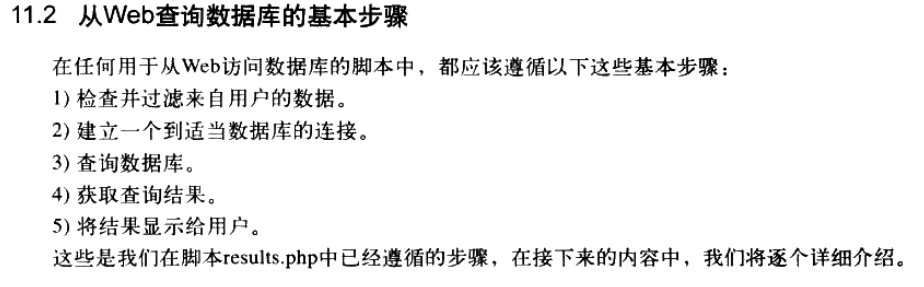
{

  addslashes($prot);

}

在手册中string addslashes ( string str )介绍的时候有这样一句话说明了get\_magic\_quotes\_gpc的用法以及作用。默认情况下，PHP 指令 magic\_quotes\_gpc 为 on，它主要是对所有的 GET、POST 和 COOKIE 数据自动运行 addslashes()。不要对已经被 magic\_quotes\_gpc 转义过的字符串使用 addslashes()，因为这样会导致双层转义。遇到这种情况时可以使用函数 get\_magic\_quotes\_gpc() 进行检测。

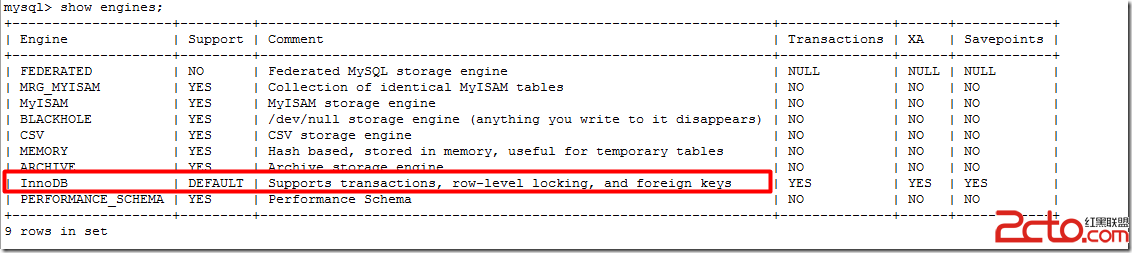




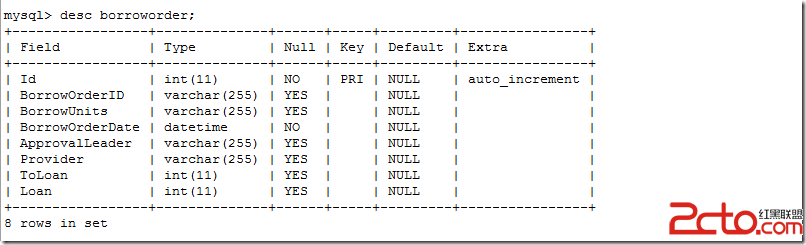
|  |
| --- |
| //连接到数据库  @ $db = new mysqli('localhost', 'bookorama', 'bookorama123', 'books');  //检查连接是否出错  if (mysqli\_connect\_errno()) {  echo '错误信息';  exit;  }  $query = "select 查询语句";  $result = $db->query($query); //执行查询  $num\_results = $result->num\_rows;//获取查询结果中的行数  for ($i=0; $i <$num\_results; $i++) {  $row = $result->fetch\_assoc();//从结果集中取得一行作为关联数组  echo htmlspecialchars(stripslashes($row['列名']));  echo stripslashes($row[‘列名’]); //调用stripslashes()在显示前整理  }  $result->free(); //释放存放查询结果的空间  $db->close(); //断开与数据库的连接--如果没有显式的调用，脚本执行完毕时会被自动关闭 |

## Mysql中查看表的类型

1.查看数据库可以支持的存储引擎   命令：show engines;

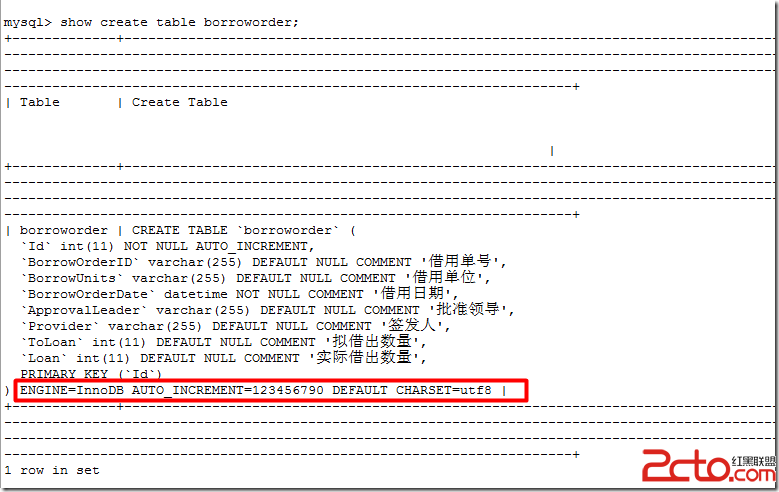


2.查看表结构 命令：desc table\_name；



1. 显示表的创建语句

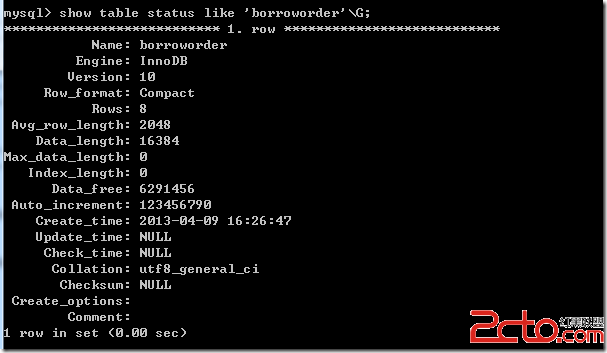
show create table table\_name;



注：   以上显示了表的默认创建语句，从上面可以看出，mysql5.5默认情况下使用的是InnoDB的引擎，使用的编码方式utf-8 .

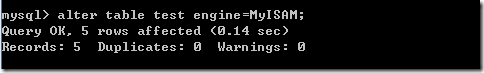
1. 显示表的当前状态值

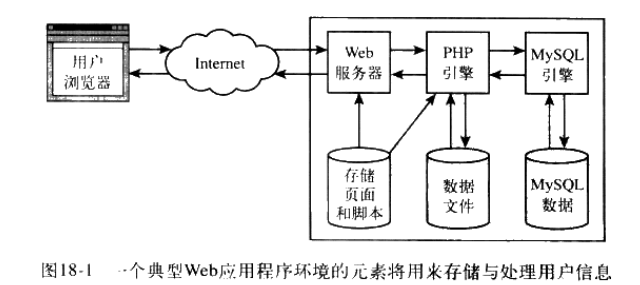
show table status like ‘table\_name’\G;



1. 修改表的数据库引擎

alter table table\_name engine=engine\_name;





GPG

gpg --gen-key 生成一副新的密钥对

gpg --export > filename 从新的密钥中导出公钥(二进制文件)

gpg --export -a > filename 从新的密钥中导出公钥(ASCII文件)

gpg --import filename 导入密钥

gpg --edit-key 'gujiayuan [<w2504187153@sina.com>'](mailto:<w2504187153@sina.com>') 编辑密钥属性

gpg> trust

gpg> sign

gpg> save

测试

gpg -a --recipient 'gujiayuan <w2504187153@sina.com>' --encrypt test.txt

说明：单引号内的文本是密钥的名称，它由名称，注释(我填的是空)，E-mail地址组成。它将生成一个test.txt.asc的加密文件。

gpg test.txt.asc 解密获得原始文件

gpg -d test.txt.asc 解密并将文本输出到显示器上

关于openssl指令

#openssl [操作]  -out  filename  [bits]

参数说明：

[操作]  主要的操作有如下两个：

      genrsa，建立RSA加密的Public key

      req，建立凭证要求文件或者是凭证文件

-out ，后面加上输出的文件名，就是那把key name

bits，用在genrsa加密的公钥的长度

-x509，X.509，CertificateData  Management.  一种验证的管理方式

例：建立一支长度为1024bits的Public Key，注意文件名。

#openssl genrsa  -out  Server.key 1024

生成证书请求命令：

#Openssl req  -new  -key file.key  -out  file.csr -config  /path/to/openssl.cnf

-config：指定openssl的配置文件路径，不指定时，默认会访问Unix格式的默认路径：/usr/local/ssl/openssl.cnf。

例：#openssl req -new -key server.key -outserver.csr

[更改Apache默认起始(索引)页面：DirectoryIndex](http://www.cnblogs.com/elfsundae/archive/2010/12/14/1905942.html)

Apache默认索引页面是index.html,修改成其他文件需要修改httpd.conf文件（对于我安装的apache5.6修改的是mods-available/dir.conf文件）。

#  
# DirectoryIndex: sets the file that Apache will serve if a directory  
# is requested.  
#  
<IfModule dir\_module>  
 DirectoryIndex index.html index.php  
</IfModule>

可在Apache根目录建立一个负责跳转的index.php，比如输入IP地址直接跳转到http://ip/myweb/

<?php  
 if (!empty($\_SERVER['HTTPS']) && ('on' == $\_SERVER['HTTPS'])) {  
 $uri = 'https://';  
 } else {  
 $uri = 'http://';  
 }  
 $uri .= $\_SERVER['HTTP\_HOST'];  
 header('Location: '.$uri.'/myweb/');  
 exit;  
?>

ttp头信息

头信息中的Location

跳转：当浏览器接受到头信息中的 Location: xxxx 后，就会自动跳转到 xxxx 指向的URL地址，这点有点类似用 js 写跳转。但是这个跳转只有浏览器知道，不管体内容里有没有东西，用户都看不到。

例：header("Location: http://www.xker.com/");

PHP安装扩展模块(curl)

安装扩展模块需要PHP的源代码文件。

php安装扩展模块的思路：

1，首先找到需要安装的扩展模块。一般在php源码文件的ext目录下。 但是有的模块php源码中不一定有，需要自己下载比如memcache、redis等。

2，通过phpize工具生成configure文件。

3，./configure -with-php-config=/usr/bin/php-config(php-config文件的路径可能不同)

4,make&&make install (可能需要加上sudo)

5,通过/usr/bin/php -i |grep 'extension\_dir' 找到放.so文件的目录，查看编译安装的模块是否在这个目录下面。如果是，只需在php.ini在添加extension=模块名，如果不是在则把该模块到该目录下，再在php.ini加入extension=模块名即可。一般情况下都是在extension\_dir中的，至少我没看到例外。这里有点像apache扩展模块的安装 不过apache可以自动写入配置文件。

6,查看加载是否成功 /usr/local/php/bin/php -m |grep 模块名

**正式开始，**

1,cd 到php源码包中的ext目录下，由于我们是要扩展安装curl模块，所以我们进入到curl目录下。

执行命令 /usr/bin/phpize

没有报错信息了。ls 一下查看是否生成了configure这个文件。

2，./configure --with-php-config=/usr/bin/php-config 又有报错信息

configure: error: Please reinstall the libcurl distribution -

easy.h should be in <curl-dir>/include/curl/

解决方法：重新安装curl。我使用的是sudo apt-get install libcurl4-gnutls-dev

3,make &&make install (需要加上sudo以获取权限)

4，我们可以看到如下提示

Installing shared extensions: /usr/lib/php/20131226/

在命令行下执行 /usr/bin/php -i |grep 'extension\_dir'



可以发现,模块就安装在extension\_dir目录下。

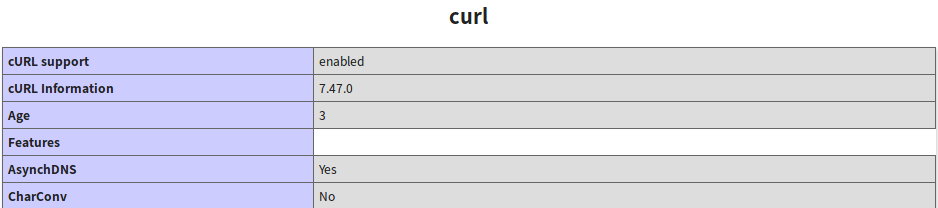
5，在php.ini中添加extension=curl.so,

并将extension\_dir 设置为"/usr/lib/php/20131226/"

6，查看加载是否成功。

/usr/bin/php -m |grep curl

当然，我们可以重启apache服务，通过访问phpinfo()这个页面也可以查看到模块。



我遇到的问题是：

进行到了第6步，重启apache后。访问phpinfo()页面可以找到这个模块，但是执行/usr/bin/php -m命令在列出的模块中却没有curl模块。

我在Geany中运行执行curl\_init()函数的程序，得到如下错误信息。



经网上搜索，知道php的在apache2目录下有一个配置文本(php.ini)，在cli目录下也有一个配置文件(php.ini)。如下：

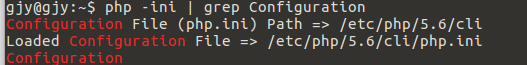
列出网友的原话：

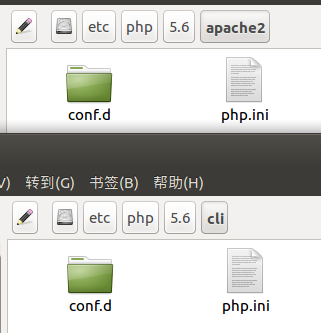
php添加扩展 在phpinfo能看到该扩展，但在cli用php -m 却看不到，为什么呢，求指教

1. 没有出现的原因是：执行时添加上php.ini的文件就可以了$ /usr/local/php/bin/php -c /usr/local/php/etc/php.ini -m | grep 模块名

2. $ /usr/local/php/bin/php --ini  | grep Configuration

    把php.ini文件复制一份到这个目录下就可以了





我只修改了apache下的配置文件，没有修改cli目录下的配置文件，所以出现了上述现象。

在我对cli目录下的配置文件做了同样的修改后，便可以通过usr/bin/php -m命令找到该模块了 。

这里有一个需要弄明白的问题：

php在apache2下的配置文件和在cli下的配置文件有什么不同。为什么要这么设置？