PHP面试题

Maggie

目　录

PHP常见面试题集锦

1. 抓取远程图片到本地,你会用什么函数?
   fsockopen();

```
//获取远程图片的大小
$fp = fsockopen("www.baidu.com", 80, $errno, $errstr, 30);
if ($fp) {
    //这里请求设置为HEAD就行了
    $out = "HEAD /img/baidu_sylogo1.gif HTTP/1.1\r\n";
    $out .= "Host: www.baidu.com\r\n";
    $out .= "Connection: Close\r\n\r\n";
    fwrite($fp, $out);
    while (!feof($fp)) {
        $header = fgets($fp);
        if (stripos($header, 'Content-Length') !== false) {
            $size = trim(substr($header, strpos($header, ':') + 1));
            echo 'ddd'.$size;
        }
    }
    fclose($fp);
} else {

    echo "jkj.$errstr ($errno)";
}
```

没找到使用fsockopen函数到本地的具体代码，有他的一些类似用法

2.简述pOST 和GET传输的最大容量分别是多少?
2MB,1024B

3.用最少的代码写一个求3值最大值的函数.

```
function lar($a,$b,$c){
return $a>$b? ($a>$c? $a : $c) : ($b>$c? $b : $c );
}
```

4.求两个日期的差数，例如2007-2-5 ~ 2007-3-6 的日期差数

```
//先转为时间戳
$begin=strtotime('2007-2-5');
$end=strtotime('2007-3-6');
echo ($end-$begin)/(24*3600);
```

5.请写一个函数，实现以下功能：
字符串“open_door” 转换成 “OpenDoor”、” make_by_id” 转换成 ” MakeById” 。

```
function str_change($str) {
    //将_替换成空格
    $str = str_replace ( “_”, " ", $str );
    $str = ucwords ( $str );
    //将空格换成空
    $str = str_replace ( " ", “”, $str );
    return $str;
}
```

6.要求写一段程序，实现以下数组$arr1转换成数组$arr2：

$arr1 = array (
‘0’ => array (‘fid’ => 1, ‘tid’ => 1, ‘name’ =>’ Name1’),
‘1’ => array (‘fid’ => 1, ‘tid’ => 2 , ‘name’ =>’ Name2’),
‘2’ => array (‘fid’ => 1, ‘tid’ => 5 , ‘name’ =>’ Name3’),
‘3’ => array (‘fid’ => 1, ‘tid’ => 7 , ‘name’ =>’ Name4’),
‘4’ => array (‘fid’ => 3, ‘tid’ => 9, ‘name’ =>’ Name5’)
);
$arr2 = array (
‘0’ => array (
‘0’ => array (‘tid’ => 1, ‘name’ => ‘Name1’),
‘1’ => array (‘tid’ => 2, ‘name’ => ‘Name2’),
‘2’ => array (‘tid’ => 5, ‘name’ => ‘Name3’),
‘3’ => array (‘tid’ => 7, ‘name’ => ‘Name4’)),
‘1’ => array (
‘0’ => array (‘tid’ => 9, ‘name’ => ‘Name5’)));

```
//将tid相同的元素归一个数组
$arr1 = array (
'0' => array ('fid' => 1, 'tid' => 1, 'name' =>'Name1' ),
'1' => array ('fid' => 1, 'tid' => 2 , 'name' =>'Name2' ),
'2' => array ('fid' => 1, 'tid' => 5 , 'name' =>'Name3' ),
'3' => array ('fid' => 1, 'tid' => 7 , 'name' =>'Name4' ),
'4' => array ('fid' => 3, 'tid' => 9, 'name' =>'Name5' )
);
function changeArrayStyle($arr){
foreach($arr as $key=>$value){
$result[$value[fid]][]=$value;
}
return array_values($result);
}
$arr2=changeArrayStyle($arr1);
echo "<pre>";
```

```
var_dump($arr2);
```

7.写一个函数，能够遍历一个文件夹下的所有文件和子文件夹。（目录操作）

```
//echo __file__."<br>";
//echo dirname(__file__)."<br>";
$d = dir(dirname(__file__));
//var_dump($d) ;
//echo "<br>Handle: " . $d->handle . "\\n";
//echo "<br>path: " . $d->path . "\\n";
$entry = $d->read ();
//echo "<br>";
//var_dump($entry);
while ( false !== ($entry = $d->read ()) ) {
echo "<br>".$entry . "<br />";
}
$d->close ();
```

8.两张表 city表和province表。分别为城市与省份的关系表。

city:

id City provinceid

1 广州 1

2 深圳 1

3 惠州 1

4 长沙 2

5 武汉 3

..... 广州

province:

id province

1 广东

2 湖南

3 湖北

.....

（1）写一条sql语句关系两个表，实现：显示城市的基本信息。？

select A.id,A.Cityname,B.province from city A,province B where A.provinceid=B.id

（2）如果要统计每个省份有多少个城市，请用group by 查询出来。？

显示字段：省份id ，省份名，包含多少个城市。

.select B.id,B.province,count(*) as num from city A,province B where A.provinceid=B.id group by B.id

9.请简述操作系统的线程与进程的区别

参考链接：http://mp.weixin.qq.com/s/hIawygtVQII9Q7Jo-M9eBg
一个进程可以包括多个线程；
一个进程的内存空间是共享的，每个线程都可以使用这些共享内存；
一个线程使用某些共享内存时，其他线程必须等它结束，才能使用这一块内存；

操作系统的设计，因此可以归结为三点：

（1）以多进程形式，允许多个任务同时运行；
（2）以多线程形式，允许单个任务分成不同的部分运行；
（3）提供协调机制，一方面防止进程之间和线程之间产生冲突，另一方面允许进程之间和线程之间共享资源。

10.请使用伪语言结合数据结构冒泡排序法对以下一组数据进行排序 10 2 36 14 10 25 23 85 99 45。

```
$a = array(10,2,36,14,10,25,23,85,99,45);
for($j=0 ; $j<9 ; $j++)
    for($i=0 ; $i<9-$j ; $i++)
        if($a[$i] > $a[$i+1]) {
            $t = $a[$i] ;
            $a[$i] = $a[$i+1] ;
            $a[$i+1] = $t ;
         }
echo var_dump($a);
```

11.COOKIE、SESSION的联系和区别，多台web服务器如何共享SESSION，禁用COOKIE后SESSION是否可用，为什么？
可以参考：http://blog.hlnet.com.cn/forum.php?mod=viewthread&tid=73

答：session的运行机制：
用户A访问站点Y，如果站点Y指定了session_start();(以下假设session_start()总是存在)那么会产生一个session_id,这个session_id一般会以COOKIE的形式保存到用户A（我们可以通过在php.ini里设置session.use_only_cookies为1，强制SESSIONID必须以COOKIE专递。）。这时候SESSIONID表现为$_COOKIE[‘PHPSESSID’];(PHPSESSID可用session_name()函数来下修改)
用户A接着访问，这个session id($_COOKID[‘PHPSESSID’])就会在A每次访问Y的时候传送到站点Y。在站点Y上，会有这么一个目录，是用来保存SESSION的实际数据的。站点Y接收到sessionid,然后通过session id,来获得与SESSION数据的关联，并返回SESSION数据。

答：session与cookie的区别：
SESSION存储在服务器端，用户无法进行修改，相对比较安全，COOKIE存储在客户端，用户通过手段可以进行修改，相对不安全。
Session会在一定时间内保存在服务器上，当访问增多，会比较占用服务器资源。
单个cookie在客户端的限制是3k，就是说一个站点在客户端存放的COOKIE不能超过3k。

答：多台服务器如何共享SESSION：
共享就是每台服务器公用一个，那显然要把这个session专门放到一个地方
比如存数据库，每台服务器都调这个数据库里的session
存memcache 一样的原理
答:可以，但是需要在传值的时候将Session_id写到URL中;因为禁用以后再传递参数的时候会以网址的形式参数接到后面传过去！

12.HTTp协议中的pOST和GET有何区别？

操作方式	数据位置	明文密文	数据安全	长度限制	应用场景
GET	HTTP包头	明文	不安全	长度较小	查询数据
POST	HTTP正文	可明可密	安全	支持较大数据传输	修改数据

13.一段php代码，写出输出结果:

```
$a=0;
$b=0;
if(($a=3)>0||($b=3)>0){
    $a++;
    $b++;
    echo $a ;
    echo $b; //输出b的值 ($a=4 ,$b=1) $b=3没有执行
}
```

14.reqiure的include都可包含文件，二者的区别何在？require/include 与 require_once/include_once区别何在?
当包含的文件不存在时,
include:代码继续往下执行,报一个warning,
而require则是报fatal error, 停止执行.

*_once只包含一次,即便多次调用.
而不加once,则调用一次,包含一次.

关于效率:
加once需要检测已加载的文件,效率稍低,
如果能确认不会多次包含,不必加once

'

请简述数据库设计的范式及应用。

请简述数据库设计的范式及应用

1 第一范式（1NF）

在任何一个关系数据库中，第一范式（1NF）是对关系模式的基本要求，不满足第一范式（1NF）的数据库就不是关系数据库。

所谓第一范式（1NF）是指数据库表的每一列都是不可分割的基本数据项，同一列中不能有多个值，即实体中的某个属性不能有多个值或者不能有重复的属性。如果出现重复的属性，就可能需要定义一个新的实体，新的实体由重复的属性构成，新实体与原实体之间为一对多关系。在第一范式（1NF）中表的每一行只包含一个实例的信息。例如，对于图3-2 中的员工信息表，不能将员工信息都放在一列中显示，也不能将其中的两列或多列在一列中显示；员工信息表的每一行只表示一个员工的信息，一个员工的信息在表中只出现一次。简而言之，第一范式就是无重复的列。

2 第二范式（2NF）

第二范式（2NF）是在第一范式（1NF）的基础上建立起来的，即满足第二范式（2NF）必须先满足第一范式（1NF）。第二范式（2NF）要求数据库表中的每个实例或行必须可以被惟一地区分。为实现区分通常需要为表加上一个列，以存储各个实例的惟一标识。如图3-2 员工信息表中加上了员工编号（emp_id）列，因为每个员工的员工编号是惟一的，因此每个员工可以被惟一区分。这个惟一属性列被称为主关键字或主键、主码。

第二范式（2NF）要求实体的属性完全依赖于主关键字。所谓完全依赖是指不能存在仅依赖主关键字一部分的属性，如果存在，那么这个属性和主关键字的这一部分应该分离出来形成一个新的实体，新实体与原实体之间是一对多的关系。为实现区分通常需要为表加上一个列，以存储各个实例的惟一标识。简而言之，第二范式就是非主属性非部分依赖于主关键字。

3 第三范式（3NF）

满足第三范式（3NF）必须先满足第二范式（2NF）。简而言之，第三范式（3NF）要求一个数据库表中不包含已在其它表中已包含的非主关键字信息。例如，存在一个部门信息表，其中每个部门有部门编号（dept_id）、部门名称、部门简介等信息。那么在员工信息表中列出部门编号后就不能再将部门名称、部门简介等与部门有关的信息再加入员工信息表中。如果不存在部门信息表，则根据第三范式（3NF）也应该构建它，否则就会有大量的数据冗余。简而言之，第三范式就是属性不依赖于其它非主属性。

范式说明

第一范式（1NF）：数据库表中的字段都是单一属性的，不可再分。这个单一属性由基本类型构成，包括整型、实数、字符型、逻辑型、日期型等。

例如，如下的数据库表是符合第一范式的：

字段1 字段2 字段3 字段4

而这样的数据库表是不符合第一范式的：

字段1 字段2 字段3 字段4
字段3.1 字段3.2

很显然，在当前的任何关系数据库管理系统（DBMS）中，傻瓜也不可能做出不符合第一范式的数据库，因为这些DBMS不允许你把数据库表的一列再分成二列或多列。因此，你想在现有的DBMS中设计出不符合第一范式的数据库都是不可能的。

第二范式（2NF）：数据库表中不存在非关键字段对任一候选关键字段的部分函数依赖（部分函数依赖指的是存在组合关键字中的某些字段决定非关键字段的情况），也即所有非关键字段都完全依赖于任意一组候选关键字。

假定选课关系表为SelectCourse(学号, 姓名, 年龄, 课程名称, 成绩, 学分)，关键字为组合关键字(学号, 课程名称)，因为存在如下决定关系：

(学号, 课程名称) → (姓名, 年龄, 成绩, 学分)

这个数据库表不满足第二范式，因为存在如下决定关系：

(课程名称) → (学分)

(学号) → (姓名, 年龄)

即存在组合关键字中的字段决定非关键字的情况。

由于不符合2NF，这个选课关系表会存在如下问题：

(1) 数据冗余：

同一门课程由n个学生选修，"学分"就重复n-1次；同一个学生选修了m门课程，姓名和年龄就重复了m-1次。

(2) 更新异常：

若调整了某门课程的学分，数据表中所有行的"学分"值都要更新，否则会出现同一门课程学分不同的情况。

(3) 插入异常：

假设要开设一门新的课程，暂时还没有人选修。这样，由于还没有"学号"关键字，课程名称和学分也无法记录入数据库。

(4) 删除异常：

假设一批学生已经完成课程的选修，这些选修记录就应该从数据库表中删除。但是，与此同时，课程名称和学分信息也被删除了。很显然，这也会导致插入异常。

把选课关系表SelectCourse改为如下三个表：

学生：Student(学号, 姓名, 年龄)；

课程：Course(课程名称, 学分)；

选课关系：SelectCourse(学号, 课程名称, 成绩)。

这样的数据库表是符合第二范式的， 消除了数据冗余、更新异常、插入异常和删除异常。

另外，所有单关键字的数据库表都符合第二范式，因为不可能存在组合关键字。

第三范式（3NF）：在第二范式的基础上，数据表中如果不存在非关键字段对任一候选关键字段的传递函数依赖则符合第三范式。所谓传递函数依赖，指的是如果存在"A → B → C"的决定关系，则C传递函数依赖于A。因此，满足第三范式的数据库表应该不存在如下依赖关系：

关键字段 → 非关键字段x → 非关键字段y

假定学生关系表为Student(学号, 姓名, 年龄, 所在学院, 学院地点, 学院电话)，关键字为单一关键字"学号"，因为存在如下决定关系：

(学号) → (姓名, 年龄, 所在学院, 学院地点, 学院电话)

这个数据库是符合2NF的，但是不符合3NF，因为存在如下决定关系：

(学号) → (所在学院) → (学院地点, 学院电话)

即存在非关键字段"学院地点"、"学院电话"对关键字段"学号"的传递函数依赖。

它也会存在数据冗余、更新异常、插入异常和删除异常的情况，读者可自行分析得知。

把学生关系表分为如下两个表：

学生：(学号, 姓名, 年龄, 所在学院)；

学院：(学院, 地点, 电话)。

这样的数据库表是符合第三范式的，消除了数据冗余、更新异常、插入异常和删除异常。

鲍依斯-科得范式（BCNF）：在第三范式的基础上，数据库表中如果不存在任何字段对任一候选关键字段的传递函数依赖则符合第三范式。

假设仓库管理关系表为StorehouseManage(仓库ID, 存储物品ID, 管理员ID, 数量)，且有一个管理员只在一个仓库工作；一个仓库可以存储多种物品。这个数据库表中存在如下决定关系：

(仓库ID, 存储物品ID) →(管理员ID, 数量)

(管理员ID, 存储物品ID) → (仓库ID, 数量)

所以，(仓库ID, 存储物品ID)和(管理员ID, 存储物品ID)都是StorehouseManage的候选关键字，表中的唯一非关键字段为数量，它是符合第三范式的。但是，由于存在如下决定关系：

(仓库ID) → (管理员ID)

(管理员ID) → (仓库ID)

即存在关键字段决定关键字段的情况，所以其不符合BCNF范式。它会出现如下异常情况：

(1) 删除异常：

当仓库被清空后，所有"存储物品ID"和"数量"信息被删除的同时，"仓库ID"和"管理员ID"信息也被删除了。

(2) 插入异常：

当仓库没有存储任何物品时，无法给仓库分配管理员。

(3) 更新异常：

如果仓库换了管理员，则表中所有行的管理员ID都要修改。

把仓库管理关系表分解为二个关系表：

仓库管理：StorehouseManage(仓库ID, 管理员ID)；

仓库：Storehouse(仓库ID, 存储物品ID, 数量)。

这样的数据库表是符合BCNF范式的，消除了删除异常、插入异常和更新异常。

范式应用

我们来逐步搞定一个论坛的数据库，有如下信息：

（1）用户：用户名，email，主页，电话，联系地址

（2）帖子：发帖标题，发帖内容，回复标题，回复内容

第一次我们将数据库设计为仅仅存在表：

用户名 email 主页 电话 联系地址 发帖标题 发帖内容 回复标题 回复内容

这个数据库表符合第一范式，但是没有任何一组候选关键字能决定数据库表的整行，唯一的关键字段用户名也不能完全决定整个元组。我们需要增加"发帖ID"、"回复ID"字段，即将表修改为：

用户名 email 主页 电话 联系地址 发帖ID 发帖标题 发帖内容 回复ID 回复标题 回复内容

这样数据表中的关键字(用户名，发帖ID，回复ID)能决定整行：

(用户名,发帖ID,回复ID) → (email,主页,电话,联系地址,发帖标题,发帖内容,回复标题,回复内容)

但是，这样的设计不符合第二范式，因为存在如下决定关系：

(用户名) → (email,主页,电话,联系地址)

(发帖ID) → (发帖标题,发帖内容)

(回复ID) → (回复标题,回复内容)

即非关键字段部分函数依赖于候选关键字段，很明显，这个设计会导致大量的数据冗余和操作异常。

我们将数据库表分解为（带下划线的为关键字）：

（1）用户信息：用户名，email，主页，电话，联系地址

（2）帖子信息：发帖ID，标题，内容

（3）回复信息：回复ID，标题，内容

（4）发贴：用户名，发帖ID

（5）回复：发帖ID，回复ID

这样的设计是满足第1、2、3范式和BCNF范式要求的，但是这样的设计是不是最好的呢？

不一定。

观察可知，第4项"发帖"中的"用户名"和"发帖ID"之间是1：N的关系，因此我们可以把"发帖"合并到第2项的"帖子信息"中；第5项"回复"中的"发帖ID"和"回复ID"之间也是1：N的关系，因此我们可以把"回复"合并到第3项的"回复信息"中。这样可以一定量地减少数据冗余，新的设计为：

（1）用户信息：用户名，email，主页，电话，联系地址

（2）帖子信息：用户名，发帖ID，标题，内容

（3）回复信息：发帖ID，回复ID，标题，内容

数据库表1显然满足所有范式的要求；

数据库表2中存在非关键字段"标题"、"内容"对关键字段"发帖ID"的部分函数依赖，即不满足第二范式的要求，但是这一设计并不会导致数据冗余和操作异常；

数据库表3中也存在非关键字段"标题"、"内容"对关键字段"回复ID"的部分函数依赖，也不满足第二范式的要求，但是与数据库表2相似，这一设计也不会导致数据冗余和操作异常。

由此可以看出，并不一定要强行满足范式的要求，对于1：N关系，当1的一边合并到N的那边后，N的那

边就不再满足第二范式了，但是这种设计反而比较好！

对于M：N的关系，不能将M一边或N一边合并到另一边去，这样会导致不符合范式要求，同时导致操作异常和数据冗余。
对于1：1的关系，我们可以将左边的1或者右边的1合并到另一边去，设计导致不符合范式要求，但是并不会导致操作异常和数据冗余。

结论

满足范式要求的数据库设计是结构清晰的，同时可避免数据冗余和操作异常。这并意味着不符合范式要求的设计一定是错误的，在数据库表中存在1：1或1：N关系这种较特殊的情况下，合并导致的不符合范式要求反而是合理的。

在我们设计数据库的时候，一定要时刻考虑范式的要求。

神经蛮人简单的三范式

第一 每个字段要是最小单位了不能再分

第二 其它字段都要和主键有关联

第三 其它字段要和主键是直接的相关

单选题

1.有关PHP字符串的说法，不对的是：
A. 如果一个脚本的编码是ISO-8859-1，则其中的字符串也会被编码为 ISO-8859-1。
B. PHP的字符串在内部是字节组成的数组，用花括号访问或修改字符串对多字节字符集很不安全。
C. substr()、strpos()、strlen()、htmlentities()处理字符串时依据的编码方式是相同的。
D. 一个布尔值 Boolean 的 true 被转换成 string 的 “ 1” ，false 被转换成空字符串。
答案：c
substr多字节字符会乱码

10个最重大的Web应用风险与攻防（OWASP Top 10十大风险）

原文资料链接 https://mp.weixin.qq.com/s/SWW_pkG_c-9kebDDpQkamw

A1. SQL注入

防止注入漏洞需要将不可信数据从命令及查询中区分开。

1.最佳选择是使用安全的API，完全避免使用解释器或提 供参数化界面的API。但要注意有些参数化的API，比 如存储过程（stored procedures），如果使用不当，仍然可以引入注入漏洞。

2.如果没法使用一个参数化的API，那么你应该使用解释器具体的escape语法来避免特殊字符。OWASP的ESAPI 就有一些escape例程。

3.使用正面的或“白名单”的具有恰当的规范化的输入验证方法同样会有助于防止注入攻击。但由于很多应用在输入中需要特殊字符，这一方法不是完整的防护方法。 OWASP的ESAPI中包含一个白名单输入验证例程的扩展库。

A2. 失效的身份认证和会话管理

常见的会话劫持 就是用户的身份凭证没有哈希和加密、会话ID暴露在URL里（例如, URL重写）。

对企业最主要的建议是让开发人员使用如下资源：

1.一套单一的强大的认证和会话管理控制系统。这套控 制系统应:

a) 满足OWASP的应用程序安全验证标准（ASVS） 中V2（认证）和V3（会话管理）中制定的所 有认证和会话管 理的要求。

b) 具有简单的开发界面ESAPI认证器和用户 API 是可以仿照、使用或扩展的好范例。

2. 企业同样也要做出巨大努力来避免跨站漏洞，因为这 一漏洞可以用来盗窃用户会话ID。

A3、跨站脚本 (XSS)攻击

A4、失效的访问控制

注解：1 代码分析工具：http://bobao.360.cn/learning/detail/108.html

网络性能优化的8个最佳技巧

文章资料来源：http://mp.weixin.qq.com/s/Gbg_Zocu8Vt5Q7xbw-lyfw
CDN(Content Delivery Network)即内容分发网络，将源站内容分发至全国所有的节点，缩短用户查看对象的延迟，提高用户访问网站的响应速度与网站的可用性，解决网络带宽小、用户访问量大、网点分布不均等问题。

技巧一：首屏一秒渲染原则

技巧二：加载优化

技巧三：渲染优化

技巧四：动画优化

技巧五：CSS优化

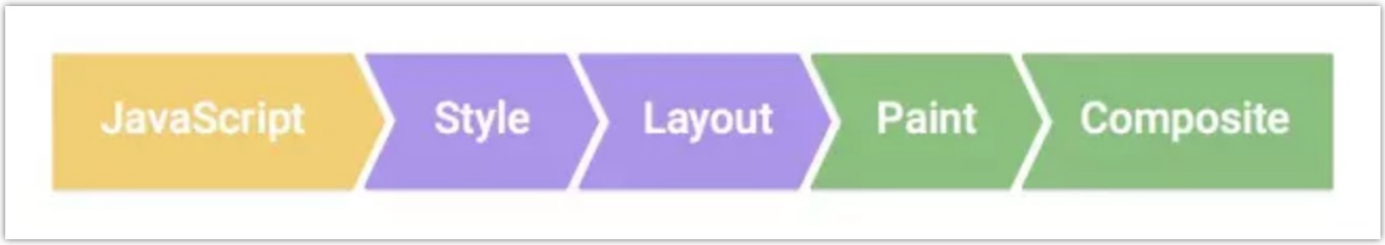
技巧六：JavaScript优化

技巧七：图片优化

技巧八：CDN优化

如何避免页面卡顿

文章链接来源：http://mp.weixin.qq.com/s/FspD-d8JYWn9SWCBVlJeLA
页面卡顿就是失帧或者掉帧了。
浏览器渲染的流程：



首先加载js做的逻辑，触发了样式变化，style把应用的样式规则计算好之后，把影响到的页面元素进行重新布局，叫做layout（布局），再把它画到内存的一个画布里面，paint（颜料）成了像素，最后把这个画布刷到屏幕上去，叫做composite（合成物），形成一帧。

mysql数据库的优化

资料来源链接：http://www.jianshu.com/p/dac715a88b44

首先肯定要符合数据库的三范式

1、选取最适用的字段属性

一般来说，数据库中的表越小它的查询速度越快，所以为了获得更好的性能，我们创建表的时候，我们可以将表中字段的宽度尽量设的小点，就是合理的设置属性的类型，例如char,varchar,data,datatime.....而且尽量把字段设置为NOT NULL,将来查询的时候就不用去比较NULL值了

2、使用索引

索引是提高数据库性能的常用方法，它可以令数据库服务器以比没有索引快得多的速度检索特定的行，尤其是在查询语句当中包含有MAX(),MIN()和ORDERBY这些命令的时候，性能提高更为明显。

那该对哪些字段建立索引呢？

一般说来，索引应建立在那些将用于JOIN,WHERE判断和ORDERBY排序的字段上。尽量不要对数据库中某个含有大量重复的值的字段建立索引。对于一个ENUM类型的字段来说，出现大量重复值是很有可能的情况

3、优化的查询语句

4、设置主键和合适的索引

构建PHP框架的步骤

1、确定好目录结构
2、创建入口文件
3、确定好MVC架构