mysql

> 作者：郭彩军

> Email：987985143@qq.com

> 出自：兄弟连IT教育

> 注意:** 代表重点

第一章mysql表设计**

1.1 E-R图**

1. E-R图组成
    1. 长方形 实体 -》数据表
    2. 椭圆形 属性 -》字段
    3. 菱形　关系 -》实体和另外一个实体存在的关系

        (1)一对一 hasOne

        用户表-》用户详情表
        商品表-》商品详情表
        订单表-》订单详情表

        (2)一对多 hasMany

        用户表-》多个地址
        1把锁-》多把钥匙
        一个英雄-》多个皮肤
        一个栏目-》多个列表
        一个用户-》多个帖子

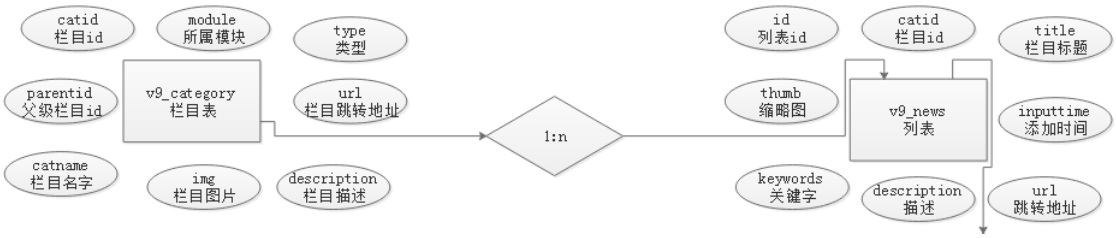
        (3)多对多ManyToMany

        帖子收藏
        一个用户可以收藏多个帖子
        一个帖子可以被多个用户收藏

        (4)属于belongTo

        多个皮肤属于英雄 多个列表属于当前栏目

    例如:



1.2 表的结构要满足三范式标准**

1. 原子性: 数据表字段达到不可再分状态,例如商品详情图,城市级联 省市县
2. 唯一性 : 数据表中只有唯一的一个记录，在一个表里,不要出现相同的记录，添加主键
3. 无冗余性 :如果一个字段可以从其他字段计算出来,这个字段不需要设计,例如商品总价...

1.3 选择合适的引擎(myisam和innodb的区别)**

1. 数据查询的速度

   myisam快

   innodb慢
2. 事务

   myisam不支持

   innodb支持
3. 全文索引

   myisam支持

   innodb不支持
4. 文件存储的区别

   (1)myisam有三个文件

   .frm 存储表结构 字段

   .myd 存储数据

   .myi 存储索引

   (2)innodb有两个文件

   .frm 存储表结构

   .ibd 表数据和索引
5. 总结

   频繁查询的表可以设置成myisam，对数据安全性比较高的设置成innodb

1.4 表字段的类型要设置合理**

1. 字段选择顺序 数字 -> 时间和枚举 -> 字符串 ,数字时间枚举要比字符串检索效率快
2. 尽量不要使用null类型

   null也会被当做检索数据源，增加检索时间

第二章mysql备份**

1.1普通备份**

1. 使用mysqldump备份单个数据库中的所有表

   mysqldump -u root -h localhost -p 数据库 >/tmp/caijun.sql
2. 使用mysqldump备份数据库中的某个表

   mysqldump -u root -h localhost -p caijun stu >/tmp/stu.sql
3. 使用mysqldump备份多个数据库

   mysqldump -u root -h localhost -p --databases 数据库1 数据库2 >/tmp/caijuno2o_13.sql

   注意:使用--databases参数之后，必须指定至少一个数据库的名称，多个数据库名称之间用空格隔开
4. 使用--all-databases参数备份系统中所有的数据库

   mysqldump -u root -h localhost -p --all-databases >/tmp/alls.sql

1.2 增量备份**

1. 增量备份介绍

   一种记录mysql操作的备份，mysql服务器会将每发一条的增删改语句写在二进制文件里
2. 操作

   查看配置文件my.cnf   vim /etc/my.cnf

   log-bin=mysql-bin 默认开启增量备份二级制文件

   expire_logs_days = 7 # 过期时间 单位为天

   binlog_format   = MIXED 日志格式
3. 查看

   增量备份文件位置:/usr/local/mysql/data=>mysql-bin.000001

   可以使用mysql的命令进行查看增量备份中的内容

   mysqlbinlog mysql-bin.000001
4. 数据恢复

   (1)通过时间

   mysqlbinlog --stop-datetime="2016-10-10 10:20:30" mysql-bin.000001 | mysql -u root -p

   (2)通过位置

   mysqlbinlog --stop-position="150" mysql-bin.000001 | mysql -u root -p

5. 重新记录增量备份

   在mysql下执行:reset master

1.3 计划任务linux操作**

1. 开启计划任务进程

   service crond start

2. 编辑

   crontab -e

3. 每隔2分钟执行一次右侧命令备份mysql数据到临时目录下o2o27.sql文件中 注意用户密码不能出现空格操作

   */2 * * * * /usr/local/mysql/bin/mysqldump -uroot -p123456 o2o_27 >>/tmp/o2o27.sql

4. 查看是否书写完成

   crontab -l

第三章mysql数据库优化**

1.1 定位慢语句**

1. 作用定位慢语句
2. 修改mysql 配置文件

   vim /etc/my.cnf

3. 在[mysqld]里加入如下代码

   log-slow-queries="/tmp/mysql-slow.log"

   long_query_time=1 （单位为秒,超过1秒代表慢语句）

4. 重启mysql
    1. mysql 关闭

       mysqladmin -u root -p shutdown

    2. mysql 启动

       mysqld_safe -u mysql &

5. 查看/tmp/mysql-slow.log

   vim /tmp/mysql-slow.log

1.2 增删改优化

1. 查看电脑的cpu使用和内存使用
    1. window打开任务管理器 ctrl+alt+delete

    2. linux top命令
    3. 如果cpu使用率高或者内存占有率大的话,及时释放内存
2. 查看表的内容是不是太大了

   select count(*) from test

   如果太大,可以考虑使用缓存

3. 检查表的索引是不是很多

   desc 表名

   如果表索引很多,及时减少索引,因为索引的创建也会消耗资源开销

4. mysql 主从服务器 读写分离

1.3 查询 优化**

1.3.1缓存优化**

原则:数据读的多写的少,类似于微博等，读的时候，先读缓存，缓存没有的话，那么就读数据库，然后取出数据后放入缓存，同时返回响应，可以使用redis,或者memcache

1.3.2索引优化**

1. 索引的定义

   索引就类似于字典目录,使用索引可快速访问数据库表中的特定信息

2. 索引类型

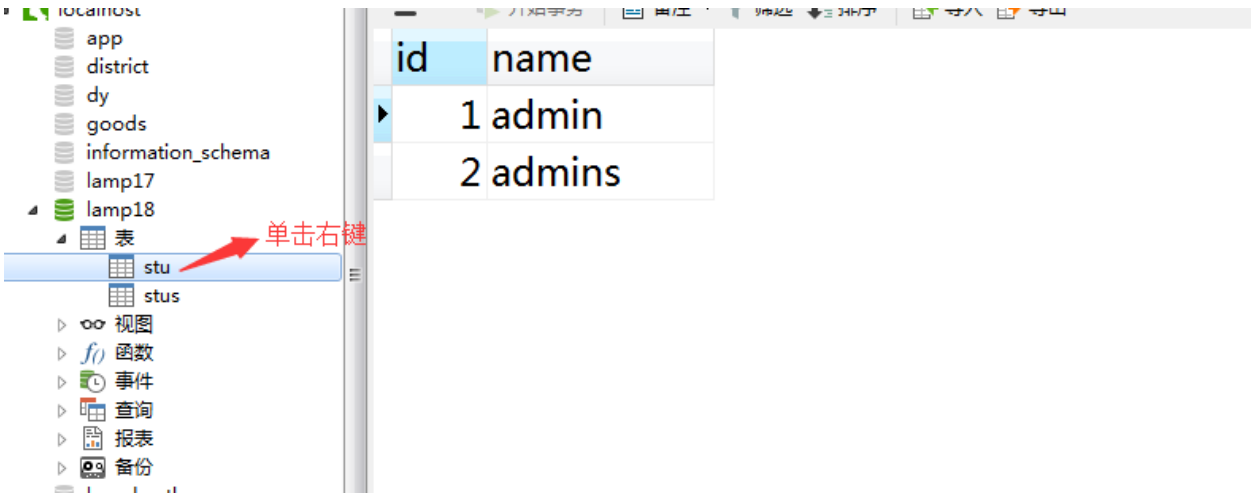
   (1)主键索引 它 是一种特殊的唯一索引，不允许有空值

   (2)唯一索引 与"普通索引"类似，不同的就是：索引列的值必须唯一，但允许有空值

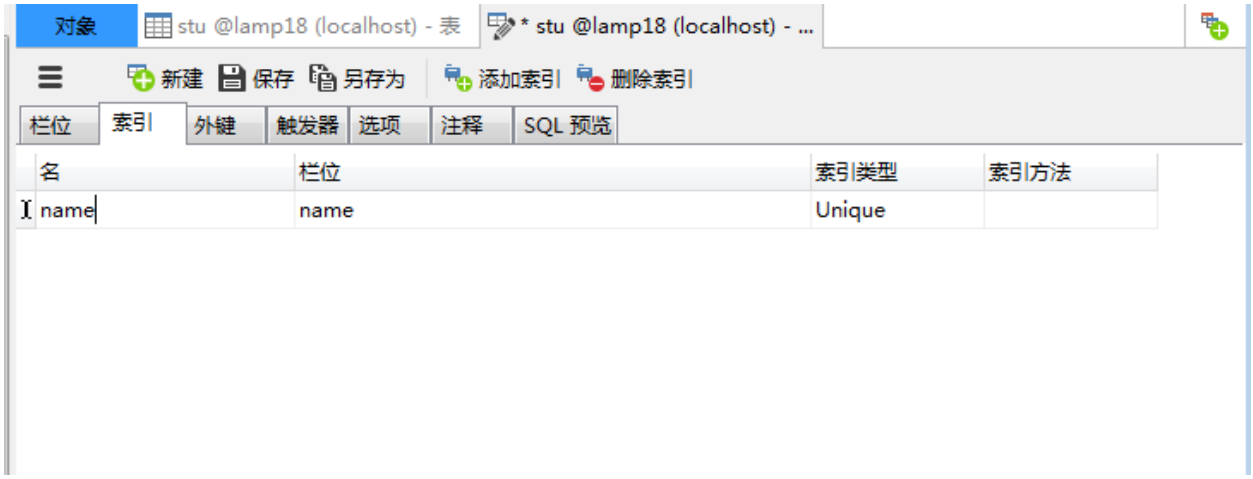
   (3)普通索引 最基本的索引，没有任何限制

3. 索引创建

   (1)通过数据库管理工具创建,例如:Navicat



   单击设计表



(2)命令创建

<1 主键索引:

建表时:create table test (id int primary key auto_increment, name char(30));

建表后 :alter table test add primary key (id)

<2 唯一索引

建表时:create table test (id int, name char(30), unique name(name))

建表后:alter table test add unique name (name)

<3 普通索引

建表时:create table test (id int, name char(30), index name(name))

建表后:alter table test add index name (name)

4. 索引删除

(1)Navicat 数据库管理工具删除

(2)命令删除

<1 主键索引

alter table test modify id int(10) 或者 alter table test drop primary key

<2 非主键索引

alter table test drop index name

5. 索引查看

show index from 表名或者 desc 表名

6. 注意点

索引并非是绝对的好, 索引的创建也是有资源开销的,比如磁盘空间的占用, 并且索引还会影响增删改语句的执行效率.

7. explain语句分析

(1)示例

explain select * from user\G

(2)根据返回参数来判定是否需要加索引

<1 row 查询次数

<2 type(连接类型)

《1 好坏顺序

system > const > eq_ref > ref > fulltext > ref_or_null > index_merge > unique_subquery > index_subquery > range > index > ALL

《2 const

表中满足条件的记录最多一条,通常会出现在主键和unique索引中 ,例如:

explain select * from test where id = 1\G;

《3 eq_ref

某一列等于带索引的列 ,例如:

explain select a.,*b.* from goods as a,cate as b where a.cate_id = b.id and a.id > 5000\G;

《4 ref

使用普通索引进行查询

select * from test where nickname = 'xiaohigh';

《5 ref_or_null

通过普通索引检索,并且会检索null值

explain select * from test_2 where nickname = 'xiaohigh' or nickname = null\G

《6 range

explain select * from test where id > 3;

《7 index

跟all一样,不过只会扫描索引.

explain select id from test

《8 all

全表扫描

explain select * from test

《9 注意

查询时一定不能出现all类型 (全表扫描,速度是最慢的) ,如果在一个数据很多的数据表里分析sql语句的时候,返回类型出现index或者all,那这个时候必须加索引

<3 key(搜索使用的索引)

1.3.3mysql主从服务器搭建,读写分离优化**

1. 目的

   主服务器写数据,从服务器读取数据

2. 主从服务器搭建

   (1)在两台服务器环境下安装mysql数据库 iptables -F(关闭防火墙)

   (2)主服务器

   <1 修改主服务器配置文件

《1 vi /etc/my.cnf

《2 [mysqld] log-bin=mysql-bin (51行) //[必须]启用二进制日志

《3 server-id=1(59行) (一般来说默认开启)

《4 重启mysql服务

mysql 关闭

mysqladmin -u root -p shutdown

mysql启动

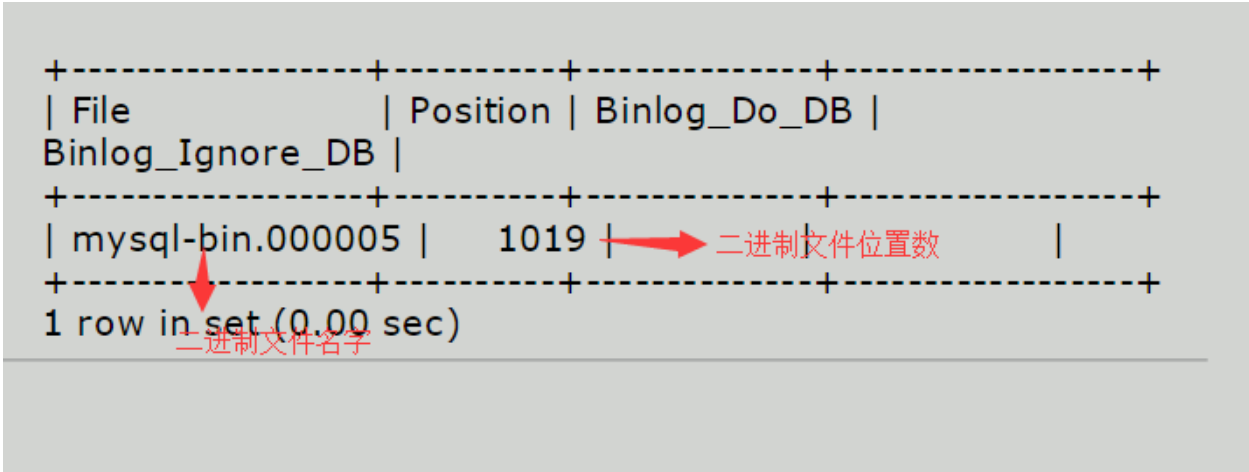
mysqld_safe -u mysql &

<2 主服务器创建mysql用户

```
mysql> GRANT REPLICATION SLAVE ON *.* to 'slaver'@'%' identified by '123456'
```

<3 查看当前主服务器信息

show master status



(3)从服务器

<1修改主服务器的mysql配置文件

vi /etc/my.cnf

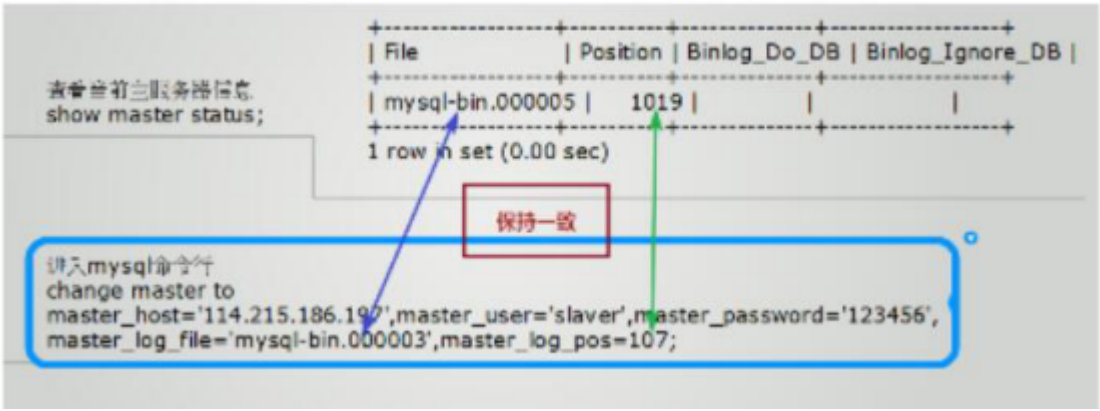
<2 修改配置文件信息

[mysqld] 开启server-id=2(96行)(默认没有开启) 重启mysql服务

<3 进入从服务器 mysql命令行 ,执行如下命令

change master to master_host='114.215.186.197',master_user='slaver',master_password='123456',master_log_file='mysql-bin.000003',master_log_pos=107;

注意:master_host 主服务器ip master_user 主服务器创建的用户名 master_password 密码 master_log_file 值必须和主服务器相同 master_log_pos 必须和主服务器相同

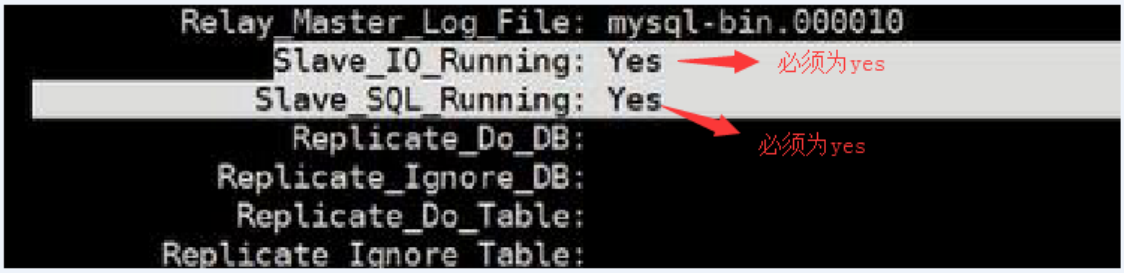


<4 开启

start slave;

<5 查看状态

show slave status\G



返回的两个参数全部为yes才代表,主从服务器搭建成功

3. 代码演示

在localhost本地连接 主从服务器,做读写分离

<1 zhu.php 主服务器代码 做数据写入

```php
<?php
//主服务器 数据的插入
$pdo=new PDO("mysql:host=主服务器ip;dbname=o2o_27","root","123456");
//设置字符集
$pdo->exec("set names utf8");
$pdo->exec("insert into stu(name)value('admins')");
 ?>
```

<2 cong.php 从服务器代码 做数据的读取

```php
<?php
//从服务器做数据的读取
$pdo=new PDO("mysql:host=主服务器ip;dbname=o2o_27","root","123456");
$pdo->exec("set names utf8");
//获取数据
$list=$pdo->query("select * from stu");
//获取结果集
$data=$list->fetchAll(PDO::FETCH_ASSOC);
echo "<pre>";
var_dump($data);
 ?>
```

<3 在浏览器里执行zhu.php

直接报错-》mysql Access denied for user 'root'@'localhost'

原因:localhost 远程连接linux下mysql root没有权限

解决方法:登录主服务器mysql 执行如下命令:

```
GRANT ALL PRIVILEGES ON *.* TO 'root'@'%' IDENTIFIED BY '你当前数据库的密码' WITH GRANT OPTION;
```

执行完命令,重新执行zhu.php ,这下不报错了,数据做了写入

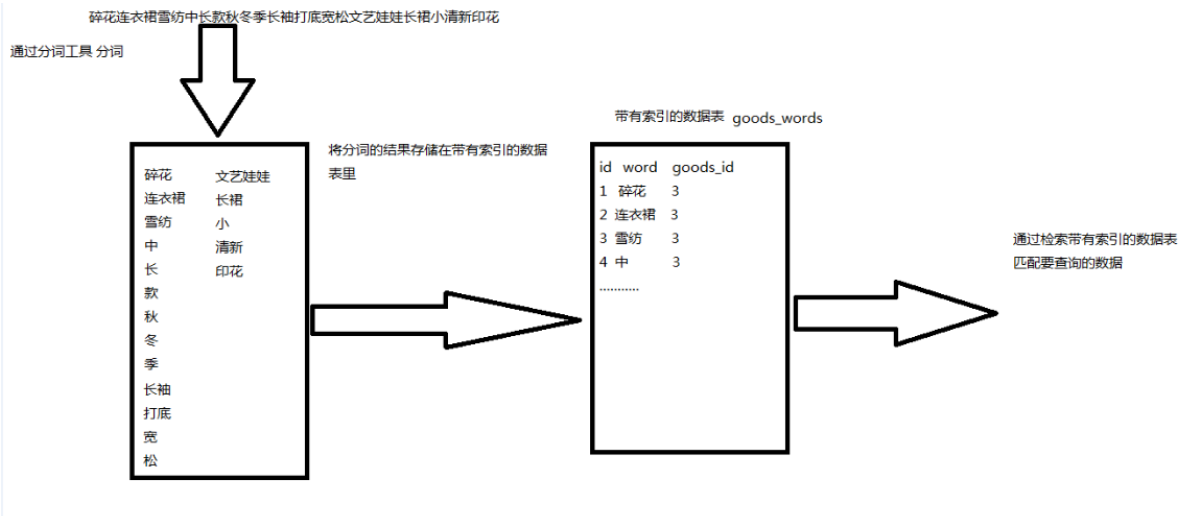
<4 在浏览器里执行cong.php 读取主服务器写入的数据

1.3.4 中文分词优化**

1. 使用中文分词背景

   普通的模糊搜索，检索数据的时候,用不到索引，而是全文检索,全文检索速率很慢,尤其是在千万级的数据里,这个时候我们考虑中文分词优化查询,其核心使其用到索引,提高数据的检索效率

2. 中文分词优化原理



3. 分词工具使用

```php
<?php
//test.php
//
// Usage on command-line: php test.php <file|textstring>
// Usage on web:
header('content-type:text/html;charset=utf-8');

//声明字符串 分词的数据源
$text = <<<EOF
怎么会爱上了他，还不是因为眼瞎，放弃了一片森林，却被一颗破草拌倒了。
EOF;
//导入导入中文分词工具包分词类
require './pscws4/pscws4.class.php';
// 实例化
$cws = new PSCWS4();
//设置字符集
$cws->set_charset('utf8');
//设置中华词典（分词的规则）
$cws->set_dict('./pscws4/etc/dict.utf8.xdb');
//设置utf8规则
$cws->set_rule('./pscws4/etc/rules.utf8.ini');
//忽略标点符号
$cws->set_ignore(true);
//传递字符串
$cws->send_text($text);
//获取权重最高的前十个词
$ret = $cws->get_tops(5);
//获取全部的分词结果
// $data=$cws->get_result();
//打印
echo "<pre>";
var_dump($ret);
//关闭
$cws->close();
?>
```

4. 中文分词优化代码

```php
<?php
//导入导入中文分词工具包分词类
require './pscws4/pscws4.class.php';
    class Chinese{
        //成员属性
        //成员方法  $str 分词的数据源
        public static function fenci($str){
            // 实例化
            $cws = new PSCWS4();
            //设置字符集
            $cws->set_charset('utf8');
            //设置中华词典（分词的规则）
            $cws->set_dict('./pscws4/etc/dict.utf8.xdb');
            //设置utf8规则
```

```
            $cws->set_rule('./pscws4/etc/rules.utf8.ini');
            //忽略标点符号
            $cws->set_ignore(true);
            //传递字符串
            $cws->send_text($str);
            //获取权重最高的前十个词
            // $ret = $cws->get_tops(5);
            //获取全部的分词结果
            $data=$cws->get_result();
            //打印
            // echo "<pre>";
            // var_dump($ret);
            //关闭
            $cws->close();
            //返回结果
            return $data;
        }
    }
    //连接数据库
    $pdo=new PDO("mysql:host=localhost;dbname=lamp","root","123456");
    //设置字符集
    $pdo->exec("set names utf8");
    //准备sql
    $sql="select title,id from goods";
    //执行sql
    $list=$pdo->query($sql);
    //获取结果集
    $data1=$list->fetchAll(PDO::FETCH_ASSOC);
    // echo "<pre>";
    // var_dump($data1);
    //遍历
    foreach($data1 as $key=>$value){
        //分词数据源 $value['title']
        $data2=Chinese::fenci($value['title']);
        // echo "<pre>";
        // var_dump($data2);666
        //遍历
        foreach($data2 as $k=>$v){
            //把分词的结果 id存储在goods_words表
            $pdo->exec("insert into goods_words
(word,goods_id)values('".$v['word']."','".$value['id']."')");
        }
    }



 ?>
```

5. 中文分词优化mysql搜索实例

    1. fenci.php

```
<?php
```

```php
//判断是否具有搜索关键词
if(empty($_GET['title'])){
    //接数据库
    $pdo=new PDO("mysql:host=localhost;dbname=lamp","root","123456");
    $pdo->exec("set names utf8");
    //；获取goods表数据
    $sql="select id,title from goods";
    //执行sql
    $list=$pdo->query($sql);
    //获取结果集
    $data1=$list->fetchAll(PDO::FETCH_ASSOC);
}else{
    //包含form.php
    require("./form.php");
}

 ?>
 <!doctype html>
 <html>
 <head>
    <meta charset="utf-8">
    <title>前台首页</title>
 </head>
 <body>
    <center>
        <form action="fenci.php" method="get">
            搜索:<input type="text" name="title">
            <input type="submit" value="搜索">
        </form>

        <table border="1px" width="400px">
            <tr>
                <th>ID</th>
                <th>TITLE</th>

            </tr>
            <?php foreach($data1 as $row){ ?>
            <tr>
                <td><?php echo $row['id'] ?></td>
                <td><?php echo $row['title'] ?></td>
            </tr>
            <?php } ?>
        </table>
    </center>
 </body>
 </html>
```

2. form.php

```php
<?php
//获取搜索的关键词
```

```php
$title=isset($_GET['title'])?$_GET['title']:'';
$pdo=new PDO("mysql:host=localhost;dbname=lamp","root","123456");
$pdo->exec("set names utf8");
//从goods_words 表查询数据
$sql1="select word,goods_id from goods_words where word='{$title}'";
//执行
$list1=$pdo->query($sql1);
//获取结果集
$data2=$list1->fetchAll(PDO::FETCH_ASSOC);
// echo "<pre>";
// var_dump($data2);
//存储goods_id
$arr=array();
//遍历数据
foreach($data2 as $key=>$value){
    $arr[$key]=$value['goods_id'];
}
//把数组转换为字符串
$id=implode(",",$arr);
// echo "<pre>";
// var_dump($arr);
//通过id匹配数据
//准备sql
$sql2="select id,title from goods where id in({$id})";
//执行sql
$list3=$pdo->query($sql2);
//获取结果集
$data1=$list3->fetchAll(PDO::FETCH_ASSOC);
// echo "<pre>";
// var_dump($data1);
return $data1;
 ?>
```

1.3.5其他优化小技巧**

1. 尽量使用一条语句插入, 避免循环插入.
2. 不要将图片存入到数据库中,用路径代替
3. 尽量避免使用select * ,效率较低
4. 分组时添加group by id order by null 及时释放资源
5. 使用or语句要注意, 两侧的语句都有索引才会使用索引.
6. 针对myisam的表要定期执行命令 optimize table test　　合并表空间产生碎片

第四章mysql 权限操作**

1.1 用户管理**

1. 必须在root下创建用户

2. 创建用户

   insert into mysql.user(Host,User,Password)values(“localhost”,用户名,password(密码));

   创建完毕后:flush privileges刷新权限

3. 查看用户列表

   select User from mysql.user group by User;

1.2 权限分配**

1. 必须在root下做权限分配
2. 分配权限

```
grant 权限 on *.* to xx@localhost identified by '用户密码';
```

3. 授权完毕之后要刷新权限 flush privileges;
4. 权限列表

   create 建库建表权限

   drop 删库删表权限

   insert 数据插入权限

   delete 数据删除权限

   alter 数据修改权限

   select 数据读取权限

   index 索引操作权限
5. *. * 代表数据库 数据表
6. xx 用户名
7. 查看权限
   1. 登录当前用户查看当前用户（自己）权限

      show grants;
   2. 登录root查看其他 MySQL 用户权限

      show grants for 用户@localhost;

1.3权限夺回**

1. 必须在root下做权限夺回
2. 权限夺回命令

   revoke 权限 on 数据库.数据表 from 用户名@localhost;
3. 授权完毕之后要刷新权限 flush privileges;

1.4 注意**

1. Test information_schema数据库和test前缀的数据库不受权限控制
2. 授权完毕之后要刷新权限 flush privileges;

第五章mysql日志操作

1.1 日志介绍

Mysql 日志操作可以快速的记录mysql的操作信息,

1.2 日志操作

修改mysql 配置文件

vim /etc/my.cnf

1.3 错误日志操作

1. 在启动或者关闭数据库信息的时候,出现错误,会记录一些日志信息
2. 设置
    1. 修改mysql 配置文件
       vim /etc/my.cnf
    2. 在[mysqld]里加入如下代码
       log-error="/tmp/mysql-error.log"
    3. 重启mysql
        1. mysql 关闭
           mysqladmin -u root -p shutdown
        2. mysql 启动
           mysqld_safe -u mysql &
    4. 查看/tmp/mysql-error.log 文件
       vim /tmp/mysql-error.log

1.4 查询日志 操作

1. 在执行增删改查的时候,日志信息记录
2. 修改mysql 配置文件

vim /etc/my.cnf

3. 在[mysqld]里加入如下代码

   log="/tmp/mysql-query.log"

4. 重启mysql

   1. mysql 关闭

      mysqladmin -u root -p shutdown

   2. mysql 启动

      mysqld_safe -u mysql &

5. 查看/tmp/mysql-query.log 文件

   vim /tmp/mysql-query.log

1.5慢查询日志 操作

1. 作用定位慢语句
2. 修改mysql 配置文件

   vim /etc/my.cnf

3. 在[mysqld]里加入如下代码

   log-slow-queries="/tmp/mysql-slow.log"

   long_query_time=1 （单位为秒,超过1秒代表慢语句）

4. 重启mysql

   1. mysql 关闭

      mysqladmin -u root -p shutdown

   2. mysql 启动

      mysqld_safe -u mysql &

5. 查看/tmp/mysql-slow.log

   vim /tmp/mysql-slow.log

1.6 二进制日志 操作

二进制日志 默认开启