**MySQL数据库优化专题**

# MySQL如何优化

表的设计合理化(符合3NF)

添加适当索引(index) [四种: 普通索引、主键索引、唯一索引unique、全文索引]

SQL语句优化

分表技术(水平分割、垂直分割)

读写[写: update/delete/add]分离

存储过程 [模块化编程，可以提高速度]

对mysql配置优化 [配置最大并发数my.ini, 调整缓存大小 ]

mysql服务器硬件升级

定时的去清除不需要的数据,定时进行碎片整理(MyISAM)

# 数据库设计

## 什么是数据库范式

为了建立冗余较小、结构合理的数据库，设计数据库时必须遵循一定的规则。在关系型数据库中这种规则就称为范式。范式是符合某一种设计要求的总结。要想设计一个结构合理的关系型数据库，必须满足一定的范式。

## 数据库三大范式

第一范式：1NF是对属性的原子性约束，要求属性(列)具有原子性，不可再分解；(只要是关系型数据库都满足1NF)

第二范式：2NF是对记录的惟一性约束，表中的记录是唯一的, 就满足2NF, 通常我们设计一个主键来实现，主键不能包含业务逻辑。

第三范式：3NF是对字段冗余性的约束，它要求字段没有冗余。 没有冗余的数据库设计可以做到。

但是，没有冗余的数据库未必是最好的数据库，有时为了提高运行效率，就必须降低范式标准，适当保留冗余数据。具体做法是： 在概念数据模型设计时遵守第三范式，降低范式标准的工作放到物理数据模型设计时考虑。降低范式就是增加字段，允许冗余。

# 分表分库

## 垂直拆分

垂直拆分就是要把表按模块划分到不同**数据库**表中（当然原则还是不破坏第三范式），这种拆分在大型网站的演变过程中是很常见的。当一个网站还在很小的时候，只有小量的人来开发和维护，各模块和表都在一起，当网站不断丰富和壮大的时候，也会变成多个子系统来支撑，这时就有按模块和功能把表划分出来的需求。其实，相对于垂直切分更进一步的是服务化改造，说得简单就是要把原来强耦合的系统拆分成多个弱耦合的服务，通过服务间的调用来满足业务需求看，因此表拆出来后要通过服务的形式暴露出去，而不是直接调用不同模块的表，淘宝在架构不断演变过程，最重要的一环就是服务化改造，把用户、交易、店铺、宝贝这些核心的概念抽取成独立的服务，也非常有利于进行局部的优化和治理，保障核心模块的稳定性

垂直拆分用于分布式场景。

## 水平拆分

上面谈到垂直切分只是把表按模块划分到不同数据库，但没有解决单表大数据量的问题，而水平切分就是要把一个表按照某种规则把数据划分到不同表或数据库里。例如像计费系统，通过按时间来划分表就比较合适，因为系统都是处理某一时间段的数据。而像SaaS应用，通过按用户维度来划分数据比较合适，因为用户与用户之间的隔离的，一般不存在处理多个用户数据的情况，简单的按user\_id范围来水平切分

通俗理解：水平拆分行，行数据拆分到不同表中， 垂直拆分列，表数据拆分到不同表中

### 水平分割案例

思路:在大型电商系统中，每天的会员人数不断的增加。达到一定瓶颈后如何优化查询。

可能大家会想到索引，万一用户量达到上亿级别，如何进行优化呢？

使用水平分割拆分数据库表。

### 如何使用水平拆分数据库

使用水平分割拆分表，具体根据业务需求，有的按照注册时间、取摸、账号规则、年份等。

### 使用取摸方式分表

首先我创建三张表 user0 / user1 /user2 , 然后我再创建 uuid表，该表的作用就是提供自增的id。

|  |
| --- |
| create table user0(  id int unsigned primary key ,  name varchar(32) not null default '',  pwd varchar(32) not null default '')  engine=myisam charset utf8;  create table user1(  id int unsigned primary key ,  name varchar(32) not null default '',  pwd varchar(32) not null default '')  engine=myisam charset utf8;  create table user2(  id int unsigned primary key ,  name varchar(32) not null default '',  pwd varchar(32) not null default '')  engine=myisam charset utf8;  create table uuid(  id int unsigned primary key auto\_increment)engine=myisam charset utf8; |

### 创建一个demo项目

#### POM文件

|  |
| --- |
| **<parent>**  **<groupId>org.springframework.boot</groupId>**  **<artifactId>spring-boot-starter-parent</artifactId>**  **<version>1.3.3.RELEASE</version>**  **</parent>**  **<dependencies>**  **<dependency>**  **<groupId>org.springframework.boot</groupId>**  **<artifactId>spring-boot-starter-jdbc</artifactId>**  **</dependency>**  **<dependency>**  **<groupId>org.springframework.boot</groupId>**  **<artifactId>spring-boot-starter</artifactId>**  **</dependency>**  **<dependency>**  **<groupId>org.springframework.boot</groupId>**  **<artifactId>spring-boot-starter-test</artifactId>**  **<scope>test</scope>**  **</dependency>**  **<dependency>**  **<groupId>mysql</groupId>**  **<artifactId>mysql-connector-java</artifactId>**  **</dependency>**  **<dependency>**  **<groupId>org.springframework.boot</groupId>**  **<artifactId>spring-boot-starter-web</artifactId>**  **</dependency>**  **</dependencies>** |

#### Service代码

|  |
| --- |
| **@Service**  **public class UserService {**  **@Autowired**  **private JdbcTemplate jdbcTemplate;**  **public String regit(String name, String pwd) {**  **// 1.先获取到 自定增长ID**  **String idInsertSQL = "INSERT INTO uuid VALUES (NULL);";**  **jdbcTemplate.update(idInsertSQL);**  **Long insertId = jdbcTemplate.queryForObject("select last\_insert\_id()", Long.class);**  **// 2.判断存储表名称**  **String tableName = "user" + insertId % 3;**  **// 3.注册数据**  **String insertUserSql = "INSERT INTO " + tableName + " VALUES ('" + insertId + "','" + name + "','" + pwd**  **+ "');";**  **System.*out*.println("insertUserSql:" + insertUserSql);**  **jdbcTemplate.update(insertUserSql);**  **return "success";**  **}**  **public String get(Long id) {**  **String tableName = "user" + id % 3;**  **String sql = "select name from " + tableName + " where id="+id;**  **System.*out*.println("SQL:" + sql);**  **String name = jdbcTemplate.queryForObject(sql, String.class);**  **return name;**  **}**  **}** |

#### Controller

|  |
| --- |
| **@RestController**  **public class UserController {**  **@Autowired**  **private UserService userService;**  **@RequestMapping("/regit")**  **public String regit(String name, String pwd) {**  **return userService.regit(name, pwd);**  **}**  **@RequestMapping("/get")**  **public String get(Long id) {**  **String name = userService.get(id);**  **return name;**  **}**  **}** |

#### 配置文件

|  |
| --- |
| spring.datasource.url=jdbc:mysql://localhost:3306/test  spring.datasource.username=root  spring.datasource.password=root  spring.datasource.driver-class-name=com.mysql.jdbc.Driver |

# SQL优化

如何从一个大项目中，迅速的定位执行速度慢的语句. (定位慢查询)

### show status

使用show status使用show status查看MySQL服务器状态信息

常用命令

|  |
| --- |
| --mysql数据库启动了多少时间  show status like 'uptime'; |
| show stauts like 'com\_select' show stauts like 'com\_insert' ...类推 update delete（显示数据库的查询，更新，添加，删除的次数） |
| show [session|global] status like .... 如果你不写 [session|global] 默认是session 会话，指取出当前窗口的执行，如果你想看所有(从mysql 启动到现在，则应该 global) |
| //显示到mysql数据库的连接数  show status like 'connections '; |
| //显示慢查询次数  show status like 'slow\_queries'; |

### 慢查询

#### 什么是慢查询

MySQL默认10秒内没有响应SQL结果,则为慢查询

可以去修改MySQL慢查询默认时间

#### 如何修改慢查询

|  |
| --- |
| --查询慢查询时间  show variables like 'long\_query\_time';  --修改慢查询时间  set long\_query\_time=1; ---但是重启mysql之后，long\_query\_time依然是my.ini中的值 |

#### 如何定位慢查询

#### 初始化测试数据

##### 创建表结构

|  |
| --- |
| /\*部门表\*/  CREATE TABLE dept(  deptno MEDIUMINT UNSIGNED NOT NULL DEFAULT 0, /\*编号\*/  dname VARCHAR(20) NOT NULL DEFAULT "", /\*名称\*/  loc VARCHAR(13) NOT NULL DEFAULT "" /\*地点\*/  ) ENGINE=MyISAM DEFAULT CHARSET=utf8 ;  /\*员工表\*/  CREATE TABLE emp  (empno MEDIUMINT UNSIGNED NOT NULL DEFAULT 0, /\*编号\*/  ename VARCHAR(20) NOT NULL DEFAULT "", /\*名字\*/  job VARCHAR(9) NOT NULL DEFAULT "",/\*工作\*/  mgr MEDIUMINT UNSIGNED NOT NULL DEFAULT 0,/\*上级编号\*/  hiredate DATE NOT NULL,/\*入职时间\*/  sal DECIMAL(7,2) NOT NULL,/\*薪水\*/  comm DECIMAL(7,2) NOT NULL,/\*红利\*/  deptno MEDIUMINT UNSIGNED NOT NULL DEFAULT 0 /\*部门编号\*/  )ENGINE=MyISAM DEFAULT CHARSET=utf8 ;  /\*薪水\*/  CREATE TABLE salgrade  (  grade MEDIUMINT UNSIGNED NOT NULL DEFAULT 0,  losal DECIMAL(17,2) NOT NULL,  hisal DECIMAL(17,2) NOT NULL  )ENGINE=MyISAM DEFAULT CHARSET=utf8;  /\*测试数据\*/  INSERT INTO salgrade VALUES (1,700,1200);  INSERT INTO salgrade VALUES (2,1201,1400);  INSERT INTO salgrade VALUES (3,1401,2000);  INSERT INTO salgrade VALUES (4,2001,3000);  INSERT INTO salgrade VALUES (5,3001,9999); |

##### 创建函数

|  |
| --- |
| create function rand\_string(n INT)  returns varchar(255) #该函数会返回一个字符串  begin  #chars\_str定义一个变量 chars\_str,类型是 varchar(100),默认值'abcdefghijklmnopqrstuvwxyzABCDEFJHIJKLMNOPQRSTUVWXYZ';  declare chars\_str varchar(100) default  'abcdefghijklmnopqrstuvwxyzABCDEFJHIJKLMNOPQRSTUVWXYZ';  declare return\_str varchar(255) default '';  declare i int default 0;  while i < n do  set return\_str =concat(return\_str,substring(chars\_str,floor(1+rand()\*52),1));  set i = i + 1;  end while;  return return\_str;  end  create FUNCTION rand\_num()  RETURNS int(5)  BEGIN  DECLARE i int default 0;  set i =floor(10+RAND()\*500);  return i;  END |

##### 创建存储过程

|  |
| --- |
| delimiter $$  create procedure insert\_emp(in start int(10),in max\_num int(10))  begin  declare i int default 0;  #set autocommit =0 把autocommit设置成0  set autocommit = 0;  repeat  set i = i + 1;  insert into emp values ((start+i) ,rand\_string(6),'SALESMAN',0001,curdate(),2000,400,rand\_num());  until i = max\_num  end repeat;  commit;  end $$  执行存储过程  call insert\_emp (100001,40000000); |

#### 如何将慢查询定位到日志中

在默认情况下，我们的mysql不会记录慢查询，需要在启动mysql时候，指定记录慢查询才可以

bin\mysqld.exe --safe-mode --slow-query-log [mysql5.5 可以在my.ini指定]（安全模式启动，数据库将操作写入日志，以备恢复）

bin\mysqld.exe –log-slow-queries=d:/abc.log [低版本mysql5.0可以在my.ini指定]

先关闭mysql,再启动, 如果启用了慢查询日志，默认把这个文件放在

my.ini 文件中记录的位置

#Path to the database root

datadir=" C:/ProgramData/MySQL/MySQL Server 5.5/Data/"