数据库优化从4个方向去优化  
1、sql和索引，写出健壮的sql，索引不是越多越好  
2、数据表结构（存储引擎，字段大小，字段类型，索引，第三规范）  
3、系统配置（打开文件系统次数，文件安全性）  
4、硬件（更合适的cup,更大的内存，更快的io，cup并不是越大越好）

能用php 实现的就不用mysql

浏览器访问一个网页要访问apache 而apache和php是一个阵营的 一旦链接了mysql就要又产生至少一个通信； 通信是最浪费时间的地方；

第一节课:

mysql （两大部分 设计与操作）

|  |  |
| --- | --- |
| 表设计 | 数据库管理、表的管理、字段管理、索引管理、服务器管理 |
| 数据操作 | sql语句、单表、多表操作 |

Mysql管理数据库和表；

···永久存储 –> 存文件里；

数据库在mysql眼中是库，在我们眼中是目录

数据引擎 innedb （安全 速度快）

mess.frm 表字段

Mess.idb 部分数据 + 索引

还有一部分数据在ibdata1 文件中(共享存储文件/所有表数据都在这里)

Myisam 表引擎(功能不健全 )

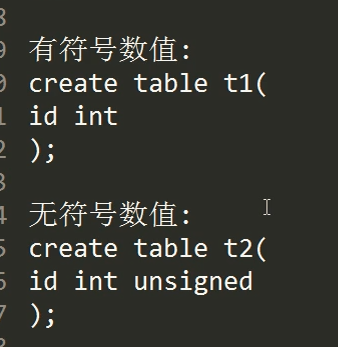
Frm 表字段

Myd 表数据

Myi 表索引

文件的效率 受磁盘的、转速、控制

第二节课:

\ 

数据库命令:

关闭mysql: net stop mysql

开启 mysql: net start msyql

登录并进入数据库 : mysql –u –p 数据库名称

退出mysql : exit;

表操作:

每创建一个表 磁盘上就会产生三个文件(表结构，表数据 ，表索引)

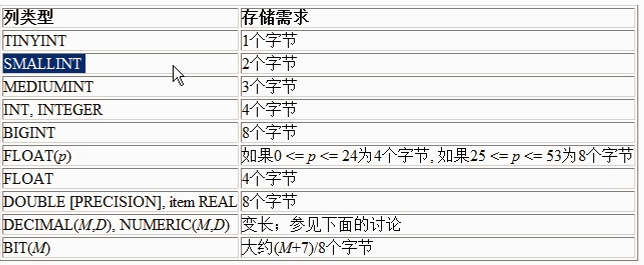
数据库设计：

1数据表引擎:

Myisam

innodb

2.数据值和列类型



相对字段 对应的存储需求

数值

Tinyint

无符号(正数)

0 到 255

有符号(有负数)

-128 到 127

Int(m)

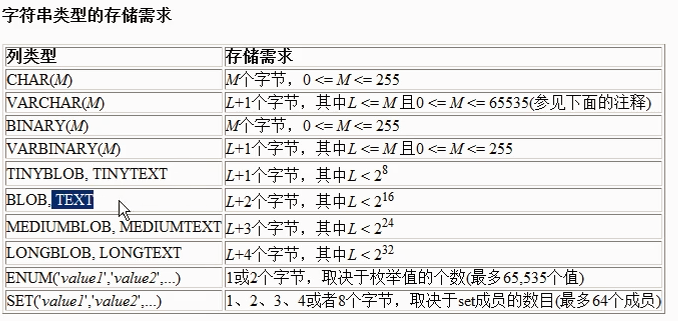
当int 为有符号的时候 m=11

无符号的时候 m=10

后面的其实没用。就算m=3 也能输入1234四位

还有 小int 中 int 大int

字符串



Char(M) 用于(定长的字符、比如：性别)

M = 0 – 255

如图 : 如果存“abc” 则浪费M个字节

优点 : 查的快 检索效率高

Varchar（m）

M = 0 - 65535

L = “ 存储的字符串的字节 ”

如图 : 如果存“abc” 则浪费L+1个字节

Text(m)

M = 0 -65535

…….还有更大的

日期

Int

存时间戳 因为时间戳在全球统一的，可以转变为各国的时间，它也能进行计算

3.数据字段属性

1. unsignd

无符号

2.zerofill

0填充

使用场景: int(4) 时候 存入33 表中数据为 0033

3．Auto\_increment

自增

4.null

是否能为空

5.not null

6.default

默认值

6.Mysql服务器的默认字符集

\s 查看mysql状态

在 my.ini 中可以修改默认字符集

Mysql 中utf8 么有杠

1. 建议客户端字符集是utf8;

Default – character – set = utf8

` 2. Mysql服务端字符集为utf8

Cahracter – set server=utf8

3.mysql服务端校验字符集

conllagtion – server = utf8\_general\_ci

字段管理

1添加

Alter 修改表结构 字段

Alter table user add age tinyint unsigned not null after id ;在id 后添加一个字段

Alter table user add pwd varchar(30) not null first 添加字段在第一位

Atter table user add age tinyint unsigned not null

2 删除

Alter table user drop password

3 修改

Modigy 只能改变 列类型 和列属性

Alter table user modify username varchar(30) not null

能修改名字 change

Alter table user change username user varchar(30) not null

4 查看

7.索引管理



中括号 可选 可不选 圆括号必须

1. 主键

包含了唯一索引 比如id字段里面不能出现两个相同的数字

一个表中只能有一个;

添加

创建表的时候 primary\_key

删除

先删自增 在删主键

、 因为自增是列属性 alter table user modify id int not null ; 修改就行了

然后删 alter table user drop primary key;

1. 唯一

不允许重复 只能唯一；

添加

Alter table user add unique u\_username(username);

删除

Alter table user drop index u\_username

查看

3.普通

添加

Alter table user index u\_username(username)

删除

查看

创建表的时候 要选用 not null

原因 null 会站四五个字节

而not null 不填数据的时候为空字符串 不站字节

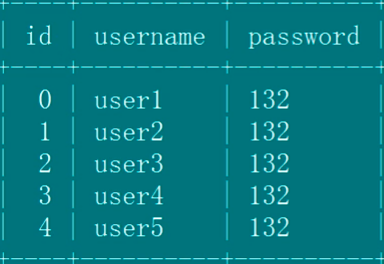
第三节

数据库操作:

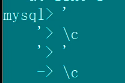
Mysql:: 触发器 ?????

Mysql:: 列可以进行数学运算

比如 select (id-1) id username,password from user;



\c 截断;



检测操作:: desc delete from user where id= 5 \G

mysql操作:: set @x = 1; 赋值语句

Select @x 输出语句

DCL 数据控制语言 grant commit rollback

服务器层面的东西

grant –> 授权新用户

commit -> 事务提交

rollback -> 事务回滚

DDL 数据定义语言 create drop alter

Create

Drop 数据库 表 级别

Alter 字段索引 级别的东西

DML 数据操作语言 insert update delete

DQL 数据查询语言select

查询一般优化:

Select \* form user; 与 select id form user 的区别

select id form user 速度快；(明确字段的话速度快)

查单列速度优于多列; 查id比查其他字段快

给字段起别名:

字段在php 查询的时候 会作为下标、所以字段过长会对php造成压力

Select username as name , password pwd from user;

Select username name, password pwd form user;

Distinct关键字的使用 (不重复)

Select distinct password from user;

查询控制null ???

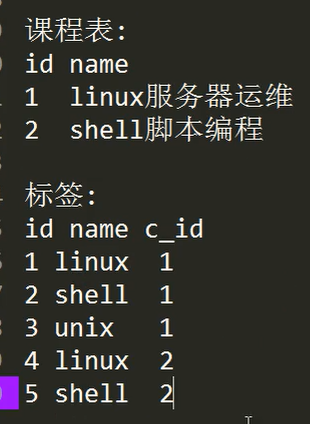
Select \* from user where age is null; 查询是null的行

Select \* from user where age is not null; 查询不是null的行

优化 之 分页 limit (好处提高查询速度)

Like 查找（效率低、会让索引失效）在数据多的情况下不合理

Like 是检索每一条数据 一条一条的查；慢



标签表的形式代替like查询

网站的搜索技术： (第九节课)

Like :数据很少的时候可以用like

标签 :量大 并且精确 ；

搜索引擎 :如果量大并且及时精确并且在极端的时间内搜索到

百度为什么搜索那么快 ? 会牵扯到搜索引擎技术 、 爬虫技术 、 缓存等 Sphinx 搜索引擎

Mysql 里面有个计数器 用来记录id ； 所以会出现1，2,3,5,6

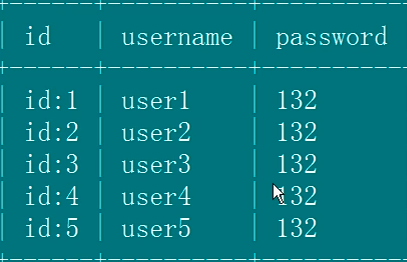
Delete 与 truncate的区别

1. delete 清空表数据，但不会清楚计数器(自增) 能删除某几条
2. truncate清空表数据，同时会清楚计数器(自增) 全删
3. 清空整个表 truncate 会快 并从id =1开始； delete是一条一条删，truncate是一页一页删

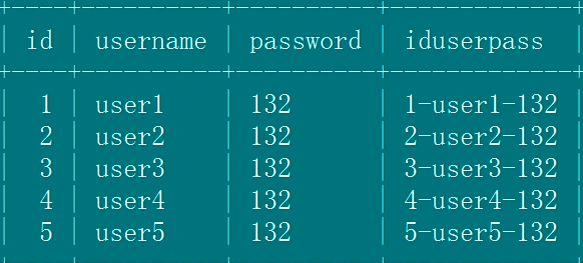
Mysql 函数:

Concat() 连接;

Select concat(“id:”,id) id,username,password form user;



Select id,username,password concat(id,”-”,username,’-’,password) iduserpass from user;



网络监控

数据数rand函数:

Select \* from user order by rand() limit 1 （用于抽奖）

Rand随机排序；

统计个数count函数

Select count(\*) form user; 打印出总行数;

Count(\*) 和 count(主键字段) 统计最快；

求和sum函数  
 select sun(age) from user;

平均值avg函数

Select avg(id) from user;

最大值max函数

Select max(id) from user;

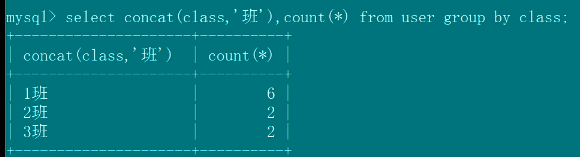
最小值 min 函数

Select min（id ）from user;

Group by 分组聚合的使用（两个关键动作 分组 、聚合）

使用场景（分组统计个数、分组获取最大值/最小值、分组获取总成绩、分组获取平均值。等）

Select count(\*) ,class from user group by class



Linux 无版权 免费 开源 、开源不开源的产品一开始都在linux发布;

黑屏系统 没有桌面; 极大节约系统资源可以最大化节省资源和开支；

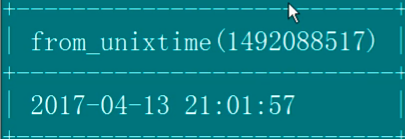
Mysql中查看未来的自增数 show create table user；

查函数 –》 ？functions

Mysql 中时间戳: unix\_timestamp() 获取当前时间戳

insert into class(name,ctime) values(‘1\_class’,unix\_timestamp())

mysql 中时间戳:from\_unixtime(unix\_timestamp)转换成时间格式



索引不能乱加;索引所占的大小有可能比列下面的值都大；

太站空间了,但是当空间够大 也可以多加索引；

Mysql 多表查询

1. 普通多表查询:

Select \* from user，class where 条件 ；

这个查询 查询的次数是两个表条数的乘积

1. 嵌套查询(子查询)

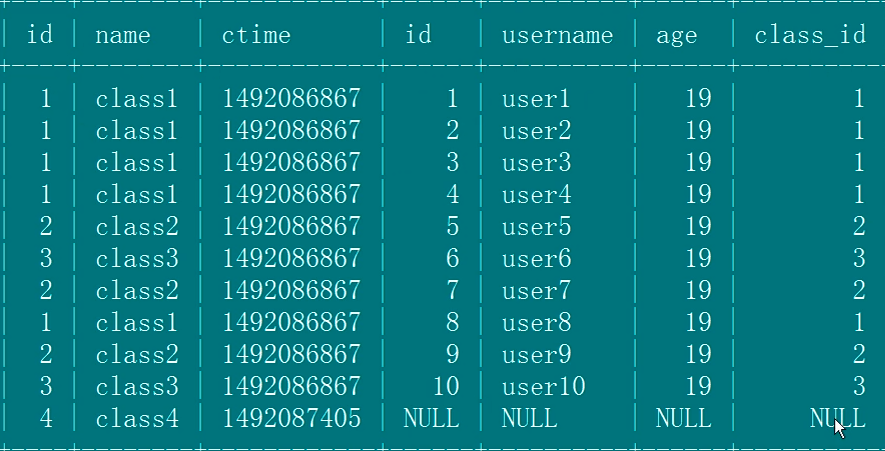
缺点: 查的信息只能是某一个表的信息(最外层的sql查询的表)  
 select distinct class\_id from user; // 去重获取

Select \* from class where id in (select distinct class\_id from user)

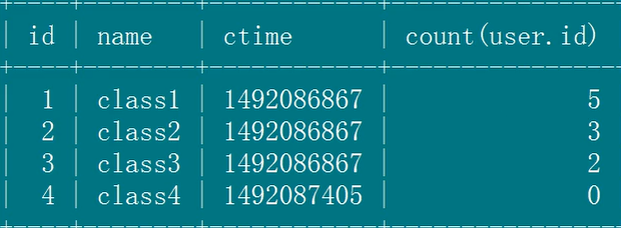
1. 链接查询
2. 左链接(作用: 谁在左边谁的数据都出来)

Select \* from class left join user on class.id = user.class\_id;

Select class.\* from class left join user on class.id = user.class\_id



Select class.\*,count(id) from class left join user on class.id = user.class\_id group by id;



Count(id) 因为最后一行id 为 大写 NULL 不是字符串! 所以统计为0；

1. 右链接

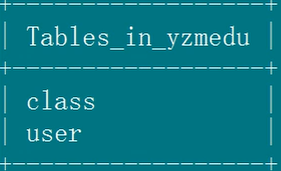
和左链接一i样

3.内链接

和普遍链接一样

、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、

优化篇  
 1.mysql基础操作



Mysql 表复制

1.复制表结构

Create table student like user;//创建student表并将表结构给了student

2.复制表内容

Insert info student select \* from user;//查出来 插入

Mysql 索引

Alter

1创建索引

主键:alter

唯一 cerate unique index u\_username on user(age);

普通 create index i\_age on user(age);

2删除索引

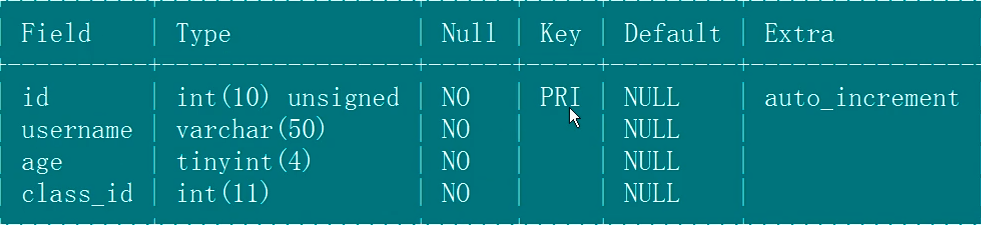
主键:alter

唯一 drop index u\_username on user;

普通 drop index i\_age on user;

3查看索引

Desc 表名;



Show index form user\G

Mysql 视图

特性:当表中数据发生变化时 视图也会发生变化

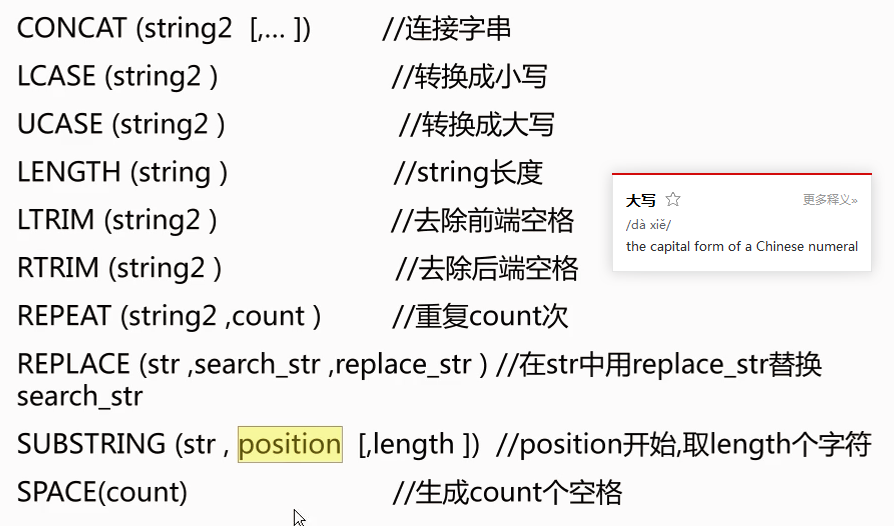
创建 create view :

删除 drop view userclass

查看 select \* from userclass;

Mysql 内置函数

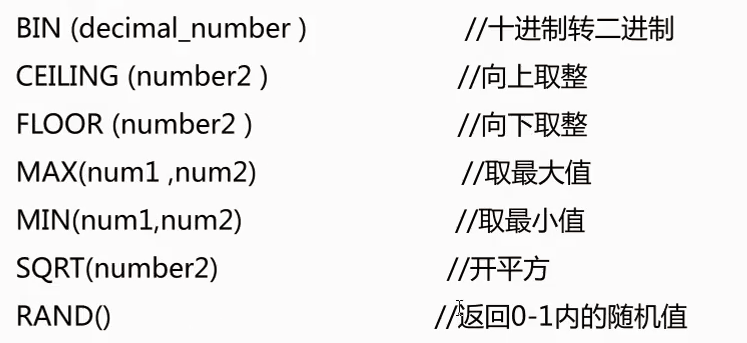
字符串函数:



Space 生成空格；

Select concat(space(4).”a”);

数学函数



日期函数



重排auto\_increment值

1 truncate tablename;

优点速度快 缺点不知道删了几行

2将数据清空后 修改;

Alter tbale student auto\_increment=1;

常用sql 技巧和常见问题:

正则表达式的使用: (不建议使用)

select \* from user where username regexp ‘php&’;

巧用rand()

随机取三行

Select \* from user order by rand() limit 3;

Mysql中获取帮助

简单

？show

更多

？sh%;

Sql优化:

书籍: 网易专家DBA 《深入浅出mysql》

看书的计划 : 先看书 再看视频;有计划的看；

优化sql语句的一般步骤:

查看各种sql执行频率

show session status（默认）（当前链接）

show global status （从mysql开启到至今）

检查服务器增删改查的使用频次(本次链接)

Show status like “%Com\_select%”

Show status like “%Com\_update%”

Show status like “%Com\_delete%”

Show status like “%Com\_insert%”

检查服务器增删改查的使用频次(服务器开启到至今)

Show global status like “%Com\_select%”

Show global status like “%Com\_update%”

Show global status like “%Com\_delete%”

Show global status like “%Com\_insert%”

查看innodb 的增删改查 操作的行数

Show status like “%InnoDB\_rows%”

定位执行效率低的sql

1.Explain 或 desc 定位一条sql语句的影响行数（如果知道sql）

2.查看mysql 的慢查询日志（不知道的话）

慢查询日志（默认开启）:

my.ini 可以开启日志

在sql执行一定时间时候会被记入到日志中

1. 查看慢查询日志是否开启和慢查询日志的名字

show variables like "%query%";

slow\_query\_log | OFF //开启状态

slow\_query\_log\_file | DESKTOP-3D8F9D5-slow.log//存放的文件

long\_query\_time | 10.000000 // 超过10秒才会被记载

1. 查看慢查询的次数

Show status like “%quer%”

Slow\_queries | 0//次数

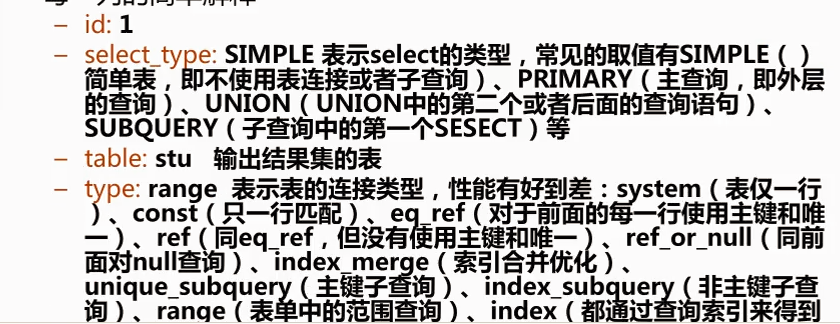
1. 修改慢查询的时间(my.ini)（记得重启mysql）

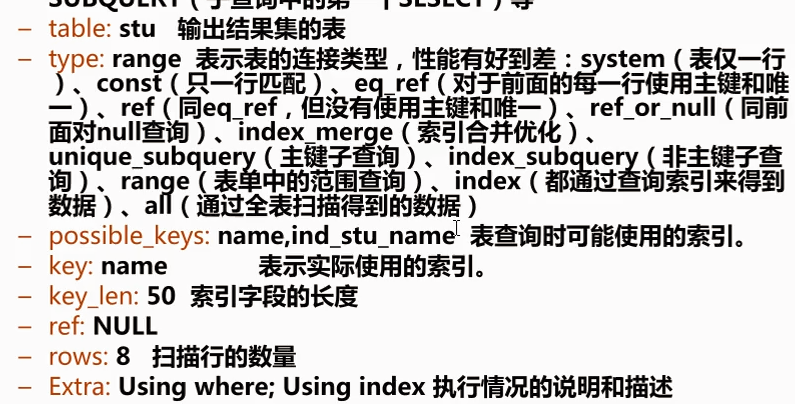
Long\_query\_time = 5

1. 快速填充量级数据

Insert into text (‘name’,’age’) select name,age from test;

查自己 , 在往自己里面添加，下一次再执行就是2倍





Linux 计划任务

Optimize

优化表空间

Optimize table sales

1. myisam表没有问题
2. innodb 表ibdata1 文件无法回收空间

ibdata1应为所有的数据库都在用，所以这个空间无法优化，但是可以“瘦身”先导出 然后删除 在导入

索引优化(最有用)

复合索引(创建多列索引)

Create index username on user(name);

创建个索引username 给 user表中的name 字段

当复合索引中的列只被用一个的时候:

用最左边的列 则索引有用，用到其他列则失效；

Like关键字

当时用like关键字进行搜索时 “%name”前面有百分号则索引失效

Null 判断

Desc select \* from user where username is null;

当判断null 值时会使用username这一列的索引

Or关键字

Desc select \* from user where username=”user7” or age = 15\G

在使用or的情况下两边的索引都有可能失效

字段和值类型不同

下面两条语句都能查到值，但有区别

Select \* from user where name =’200’；//这条用到了索引因为200是字符串200

Select \* from user where name =200//这条没有用到索引 因为值得类型和表中数据类型不同

存在索引但不适用索引

表中数据有10条

Select \* from user where id > 1 and id<=10;这条语句查询全部数据，这时候用索引就不好了，所以这时候可能会不使用索引

查看索引的使用频次

Handler\_read\_key 越高效率越高

Handler\_read\_rnd\_next 越高效率越低效

如何查？

Show status like “ %Handler\_read% ”;

多表操作

Mysql 服务器 优化

Root 密码破解

1. 密码没忘但是要改密码

Set password=password(123);

1. 密码忘记了我要破解密码

1)关闭mysql服务

net stop mysql57

2)越过user表启动mysql

a.越过授权表启动mysql母进程

cd AppServ\MySQL\bin

mysqld.exe --skip-grant-tables &

b.无密码登录

mysql -uroot

c.刷新管理员权限

flush privileges;

d.有权限后改密码

alter user 'root'@'localhost' identified by '789';

3)关闭win下任务管理器中mysqld进程

4)重新启动mysql服务

net start mysql57

Check与optimize使用方法(检测表是否损坏)

优化表空间

Optimize table sales

1.myisam表没有问题

2.innodb 表ibdata1 文件无法回收空间

ibdata1应为所有的数据库都在用，所以这个空间无法优化，但是可以“瘦身”先导出 然后删除 在导入

检测表

Check table 表名；

常用sql的优化

默认导出 中

1不含有建库语句 大量注释 一些表锁

--no-defaults 去除凤鸣 -B yzmedu 建库

--skip—add-locks 干掉写锁 --compact 去除注释

导入 source 路径