

**##Oracle 分析函数**

**实现 分组显示第一条的效果**

SELECT

--count(\*)

\*

FROM (SELECT **ROW\_NUMBER() OVER(PARTITION BY civilregionalismid,reportdate,bblx,isvalide,state ORDER BY registerdate DESC)** rn, hjcnhistory.\* FROM hjcnhistory)

WHERE rn > 1

and isvalide='01'

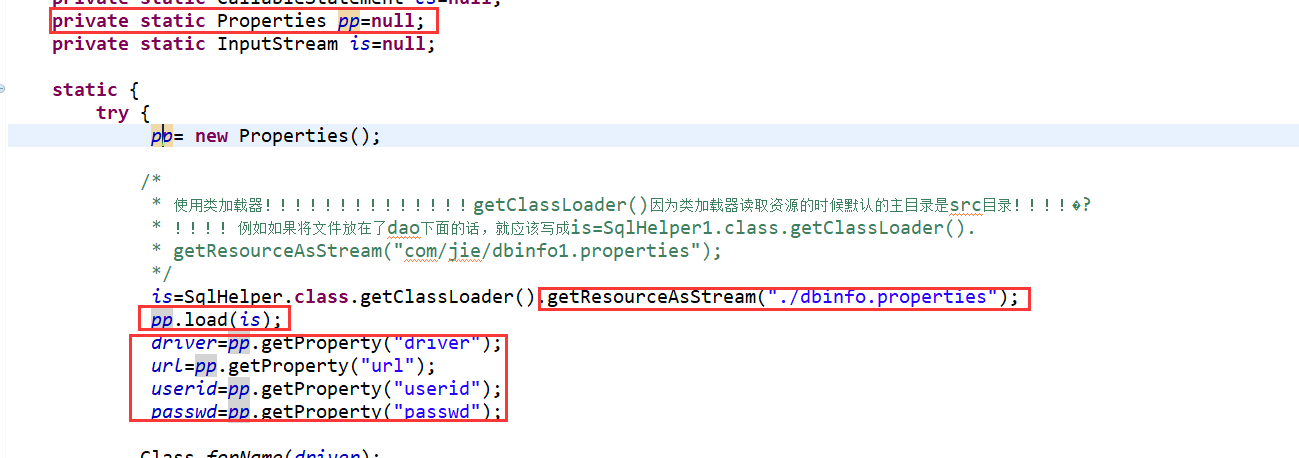
and state='4'

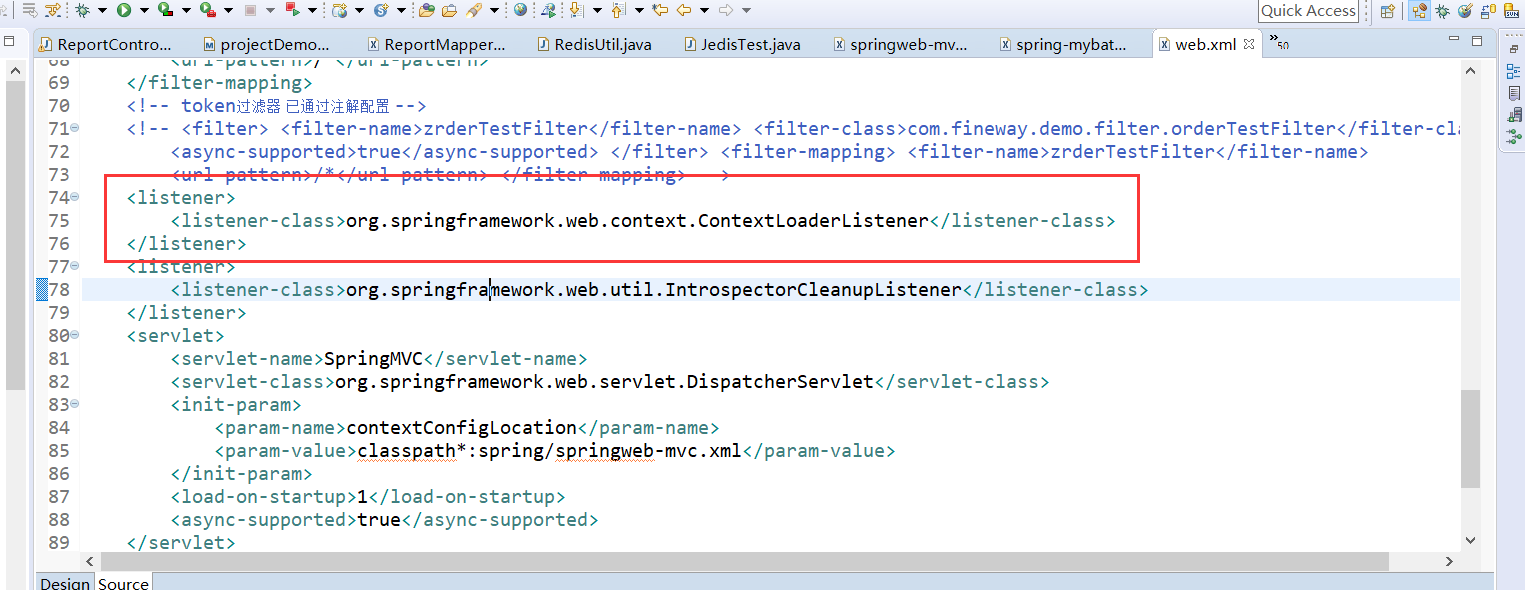
--and civilregionalismid='HJCN2017230011' and reportdate='2017-04'

order by civilregionalismid,bblx,registerdate

;

sh命令是调用的shell解析器，因此只要对文件的权限最低要求是可读就行了，而./则要求得更高一些，要有可读、写、执行权限。





不加此监听器会导致service注解@Autowired注入失败

东方通web服务器 后台启动+指定日志输出文件目录

nohup ./tongserver &>>/home/tongweb.log

使用ps –ef|grep java 查看TongWeb 的进程号

使用prstat –L –p 13516(注:13516为Java 进程号)查看Java 进程中每个线程占用的CPU



Hibernate指定字段查询 + 关联查询 + 自定义对象接收

grep -rn 字符串 .

查询当前目录下包含指定字符串的文件

Git上传大文件失败fatal: The remote end hung up unexpectedly

修改git config --global http.postBuffer 2428000

Git add .

Git commit -m ‘’

Git push

ls --full-time

grep -rn ‘字符串’ .

./startup.sh和catalina.sh run 区别：

startup.sh本质是执行catalina.sh start

项目根目录下 (Cd [项目名])

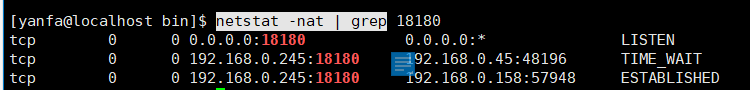
gradle clean

gradle build -i

(加 -i 输出信息 ；war生成在./build/lib下)

jvm内存排查监测 **jvisualvm jconsole**

netstat -nat | grep 【字符串】

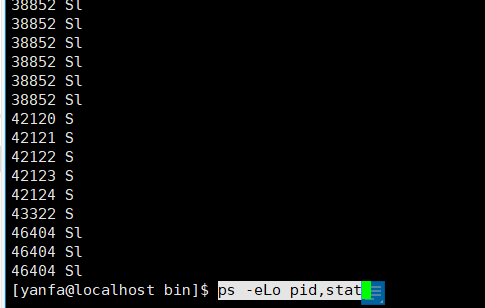


Window查看端口

netstat -aon | findstr :【关键字】

执行命令 ps -eLo pid,stat

显示



解释

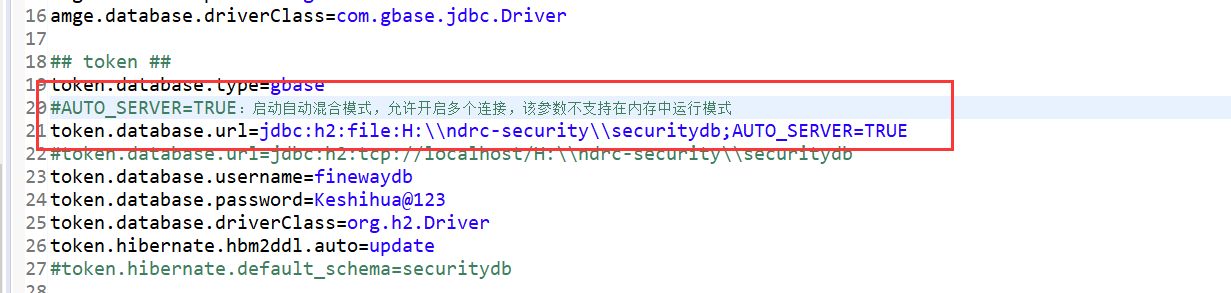
D    不可中断     Uninterruptible sleep (usually IO)  
    R    正在运行，或在队列中的进程  
    S    处于休眠状态  
    T    停止或被追踪  
    Z    僵尸进程  
    W    进入内存交换（从内核2.6开始无效）  
    X    死掉的进程  
  
  
    <    高优先级  
    N    低优先级  
    L    有些页被锁进内存  
    s    包含子进程  
    +    位于后台的进程组；  
    l    多线程，克隆线程  multi-threaded (using CLONE\_THREAD, like NPTL pthreads do)

Eclipse启动web项目（h2数据库，嵌入式）报错：file is locked

（打包部署到tomcat正常启动）

解决办法：

添加url参数



JAVA\_OPTS='-server -Xms1536m -Xmx8192m -XX:PermSize=256m -XX:MaxPermSize=512m'

Maven命令行打包

mvn clean package -DskipTests

（-DskipTests 跳过单元测试）

Maven ：

Build 是编译

Package 是打包但不放进本地库

Install 是打包并放进本地库不放进远程库

Maven打包指定生成的war名称

<!-- 产生的构件的文件名，默认值是${artifactId}-${version}。 -->

<finalName>ShareData</finalName>

Maven指定jdk版本打包

<!-- compiler插件 -->

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.2</version>

<configuration>

<source>${jdk.version}</source>

<target>${jdk.version}</target>

<!-- <source>1.8</source>

<target>1.8</target> -->

<showWarnings>true</showWarnings>

<showDeprecation>true</showDeprecation>

<verbose>false</verbose>

<compilerArguments>

<extdirs>libs</extdirs>

</compilerArguments>

</configuration>

</plugin>

nohup command>/dev/null 2>&1 &

截取文件指定行

sed -n '432437,581294p' fgw\_keshihua120190401.sql > keshihualast2019\_5.sql



Gbase数据库命令行登录

gccli -ugbase -pgbase20110531

gbase数据库重启（全部服务重启）

service gcware restart

更新嵌套查询SQL

**update** t\_res t0 **set** t0.urlPath=(**select** **replace**(t1.urlPath,'192.168.57.101:8082','192.168.0.245:18180') **from** t\_res t1 **where** t0.code=t1.code) **where** t0.urlPath **like** '%:8082%';

Mysql不支持merge语法（可能my..存储引擎支持），使用

“ON DUPLICATE KEY” 语法代替

**insert** **into** t\_user\_photo(id,user\_level,upload\_time) **values**('ee85c3ad09d94925b5bdf0d001ef3437000',1,**now**()) **ON** DUPLICATE **KEY** **UPDATE** id=id;

linux防火墙

查看 firewall-cmd --list-all

永久开放端口 firewall-cmd --zone=public --add-port=54321/tcp --permanent

重新加载 firewall-cmd --reload

参考 <https://blog.csdn.net/u013514928/article/details/80411110>

Linux进程端口号查看

1. ps -ef |grep java
2. netstat -nltp |grep 80
3. 杀死进程 ：kill -9 【pid】

Windows进程查看

1. tasklist |findstr java
2. netstat -ano|findstr 80
3. taskkill -PID 【pid】 -F (-F:强制杀死进程)

idea整合springboot+jpa+docker

参考: <https://www.cnblogs.com/hsz-csy/p/9488469.html>

重点是pom.xml和dockerfile配置

Linux netstat参数

-a (all)显示所有选项，netstat默认不显示LISTEN相关

-t (tcp)仅显示tcp相关选项

-u (udp)仅显示udp相关选项

-n 拒绝显示别名，能显示数字的全部转化成数字。(重要)

-l 仅列出有在 Listen (监听) 的服務状态

-p 显示建立相关链接的程序名(macOS中表示协议 -p protocol)

-r 显示路由信息，路由表

-e 显示扩展信息，例如uid等

-s 按各个协议进行统计 (重要)

-c 每隔一个固定时间，执行该netstat命令。

参考：<https://www.cnblogs.com/echo1937/p/6677325.html>

JPA手动拼接SQL

1. 注入

@PersistenceContext  
private EntityManager entityManager;

2、

Query query = entityManager.createNativeQuery(sql);  
List list = query.getResultList();

(createNativeQuert可加第二个参数指定返回实体类.class)

Windows退出telnet窗口

Ctrl+] 🡪 quit

Linux硬链接、软链接

创建：

硬： ln 真实文件绝对路径 链接绝对路径

（文件夹只能软链接）

软： ln -s 真实文件夹绝对路径 软链接地址

删除：

Rm 链接文件

（最后不能加/ ，否则为删除真实文件目录）

修改：

软：ln -snf 新文件路径 软链接地址

Mysql8修改了密码加密方式使用以前的工具无法登陆

修改加密方式

方法一：

ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql\_native\_password BY 'root';

方法二：

修改mysql配置文件 /etc/my.cnf

default\_authentication\_plugin=mysql\_native\_password

Docker启动mysql镜像

docker run --name mysql -p 3306:3306 -e MYSQL\_ROOT\_PASSWORD=123456 -d mysql

docker exec -it mysql /bin/bash

docker commit [OPTIONS] CONTAINER [REPOSITORY[:TAG]]

Git切换分支

git checkout 【分支名】

Git取消add 根据head匹配

git reset HEAD .

git创建分支包括本地和远程步骤：

git baranch [分支名] //创建新分支

git branch //查看分支

git checkout [分支名] //切到新分支

git add . git commit git push //代码上传

gitpush时会失败，提示push到新分支命令 git push --set-upstream [远程库名:origin] [分支名]

git修改远程仓库地址

先删除原地址，再添加新地址

git remote rm origin

git remote add origin [git@code.aliyun.com:fineway-ndrc-share/disaster-share.git](mailto:git@code.aliyun.com:fineway-ndrc-share/disaster-share.git)

git实现远程dev分支合并到master分支

先保证远程dev分支和master分支都是pull最新的

1、在本地切到master分支

Git checkout master

1. 把dev分支合进来

Git merge dev

1. 查看状态，确认无误后push到远程master分支

Git push

Springboot使用定时任务

第一步：在启动类添加注解@SpringBootApplication

第二步：注入类的方法上添加注解@Scheduled

使用cron表达式定义执行时间

注意：上述方法，如果有多个定时任务将在一个线程中执行，会有任务阻塞现象

Curl发送post请求实例（windows、linux通用）：

curl -H Content-Type:application/json;charset=UTF-8 -XPOST http://localhost:19002/jm/word/test -d "{\"id\":\"qwe\",\"name\":\"ls\",\"url\":\"XXX\",\"overview\":\"as\"}"

注意：1、-H 后面不能加引号；2、-d 后面必须为双引号;3、cmd中由于windows默认GBK编码所以不能携带中文

Tomcat部署两个相同的系统，只能启动一个

解决：

向其中一个系统Web.xml添加配置：

<context-param>

<param-name>webAppRootKey</param-name>

<param-value>project.root1</param-value>

</context-param>

（即重命名webAppRootKey）

Springboot启动修改jvm参数

java -jar -XX:MetaspaceSize=128m -XX:MaxMetaspaceSize=128m -Xms1024m -Xmx1024m -Xmn256m -Xss256k -XX:SurvivorRatio=8 -XX:+UseConcMarkSweepGC XXX.jar

参数说明：

-XX:MetaspaceSize=128m （元空间默认大小）

-XX:MaxMetaspaceSize=128m （元空间最大大小）

-Xms1024m （堆最大大小）

-Xmx1024m （堆默认大小）

-Xmn256m （新生代大小）

-Xss256k （棧最大深度大小）

-XX:SurvivorRatio=8 （新生代分区比例 8:2）

-XX:+UseConcMarkSweepGC （指定使用的垃圾收集器，这里使用CMS收集器）

-XX:+PrintGCDetails （打印详细的GC日志）

注意：

JDK8之后把-XX:PermSize 和 -XX:MaxPermGen移除了，取而代之的是

-XX:MetaspaceSize=128m （元空间默认大小）

-XX:MaxMetaspaceSize=128m （元空间最大大小）

JDK 8开始把类的元数据放到本地化的堆内存(native heap)中，这一块区域就叫Metaspace，中文名叫元空间。

使用本地化的内存有什么好处呢？最直接的表现就是java.lang.OutOfMemoryError: PermGen 空间问题将不复存在，因为默认的类的元数据分配只受本地内存大小的限制，也就是说本地内存剩余多少，理论上Metaspace就可以有多大（貌似容量还与操作系统的虚拟内存有关？这里不太清楚），这解决了空间不足的问题。不过，让Metaspace变得无限大显然是不现实的，因此我们也要限制Metaspace的大小：使用-XX:MaxMetaspaceSize参数来指定Metaspace区域的大小。JVM默认在运行时根据需要动态地设置MaxMetaspaceSize的大小。

Redis命令记录：

auth [密码] //密码登录

info //查看使用情况、内存、所以库key数量

select [db] //切换db分区0-15，默认0

keys [通配符\*] //查询复合的keys，生产环境禁用

scan [游标:0] {match [通配符]} //游标查询key

set [key] [value] //string类型设置key-value

get [key] //string类型获取value

lrange [key] [开始下标] [结束下标] //下标为负，表示从后往前（从-1开始）

llen [key] //查看list长度

rpush [key] [value] //list尾部添加

lpush [key] [value] //list头部添加

lindex [key] [index] //查询list指定索引的value

lset [key] [index] [value] //修改list指定索引的value

lpop [key] //删除list类型key的第一条value

rpop [key] //删除list类型key的最后一条value

lrem [key] [个数] [value] //个数正从左数、负从右数、0所有,删除list中指定value

dbsize //统计当前库key个数

redis批量删除（生产禁用keys）

redis-cli --scan --pattern "key前缀\*" | xargs -L 1000 redis-cli del

mysql生成uuid

**select** **REPLACE**(uuid(),'-','');

解决windows oracle 忘记密码：

进入product下至bin目录中打开cmd

输入：

1. sqlplus /nolog
2. conn /as sysdba
3. alter user sys identified by 【新密码】
4. exit

退出，此时管理员用户sys密码已修改

登录：

1. sqlplus
2. sys /as sysdba
3. 【新密码】

此时即登录成功

关闭启动oracle命令(需以管理员身份执行)：

lsnrctl status //查看oracle服务状态

lsnrctl stop //关闭oracle

lsnrctl start //开启oracle

oracle删除用户及其下的所有数据

drop user【用户名】cascade;

Linux安装nginx

<https://www.cnblogs.com/wyd168/p/6636529.html>

解决安装报错:

./configure --prefix=/usr/local/nginx --with-pcre=../pcre-8.33 --with-zlib=../zlib-1.2.11 --with-openssl=../openssl-fips-2.0.9

/usr/local/bin/ 下的可执行程序是 可以在任意目录执行

Tomcat启动慢问题解决：(亲测可用)

-Djava.security.egd=file:/dev/urandom



原理：

Tomcat 7和Tomcat 8在启动的时候会调用org.apache.catalina.util.SessionIdGeneratorBase.createSecureRandom来产生一串安全随机数。

在Linux（CentOS）环境下，随机数可以从两个特殊的文件中产生，一个是/dev/urandom，另外一个是/dev/random。

它们产生随机数的原理是利用当前系统的熵池来计算出固定一定数量的随机比特，然后将这些比特作为字节流返回。熵池就是当前系统的环境噪音，熵指的是一个系统的混乱程度，系统噪音可以通过很多参数来评估，如内存的使用，文件的使用量，不同类型的进程数量等等。

/dev/random在不能产生新的随机数时会阻塞程序，直到根据熵池产生新的随机字节之后才返回；而/dev/urandom不会（ublock），当然，产生的随机数效果也不太好。

所以我们强制Tomcat使用/dev/urandom而不是/dev/random来产生随机数，速度就会大幅提升——由好几分钟到只有几秒钟。

lasticsearch要求的最小值是262144，小于此值elasticsearch无法启动。

sysctl -w vm.max\_map\_count=262144

Linux环境修改docker镜像源

方法一（有问题）：curl -sSL https://get.daocloud.io/daotools/set\_mirror.sh | sh -s <http://f1361db2.m.daocloud.io>

sudo systemctl restart docker

方法二（注意镜像源能否访问）：

修改或新增 /etc/docker/daemon.json

# vi /etc/docker/daemon.json

{

"registry-mirrors": ["http://hub-mirror.c.163.com"]

}

systemctl restart docker.service

Docker运行mysql端口映射并挂载数据目录：

docker run --name nginx-mysql -p 3306:3306 -e MYSQL\_ROOT\_PASSWORD=123456 -v /opt/docker/mysql/data:/var/lib/mysql -d mysql:5.7

Es启动报错 65535

解决办法：

在/etc/sysctl.conf文件最后添加一行

vm.max\_map\_count=262144

Docker启动es:

docker run -d -e ES\_JAVA\_OPTS="-Des.insecure.allow.root=true" -e ES\_JAVA\_OPTS="-Xms256m -Xmx256m" --name es-node01 -p 9200:9200 -p 9300:9300 docker.elastic.co/elasticsearch/elasticsearch:6.3.2

docker启动kibana:

docker run --name kibana -p 5601:5601 --link es-node01:elasticsearch -d docker.elastic.co/kibana/kibana:6.3.2

docker启动redis

docker run -p 宿主机端口:6379 -v 宿主机/data:/data -d docker.io/redis redis-server --appendonly yes –requirepass “password”

Tomcat启动log显示乱码修改conf目录下的logging.properties文件，修改编码格式对应为当前服务器编码格式

Java安装gdal参考：

<https://blog.csdn.net/wm6752062/article/details/80897411>

spring线程池

当一个任务通过 execute(Runnable) 方法欲添加到线程池时，线程池采用的策略如下：

1. 如果此时线程池中的数量小于 corePoolSize ，即使线程池中的线程都处于空闲状态，也要创建新的线程来处理被添加的任务。

2. 如果此时线程池中的数量等于 corePoolSize ，但是缓冲队列 workQueue 未满，那么任务被放入缓冲队列。

3. 如果此时线程池中的数量大于 corePoolSize ，缓冲队列 workQueue 满，并且线程池中的数量小于maximumPoolSize ，建新的线程来处理被添加的任务。

4. 如果此时线程池中的数量大于 corePoolSize ，缓冲队列 workQueue 满，并且线程池中的数量等于maximumPoolSize ，那么通过 handler 所指定的策略来处理此任务。

处理任务的优先级为：

核心线程 corePoolSize 、任务队列 workQueue 、最大线程 maximumPoolSize ，如果三者都满了，使用 handler处理被拒绝的任务。当线程池中的线程数量大于 corePoolSize 时，如果某线程空闲时间超过 keepAliveTime ，线程将被终止。这样，线程池可以动态的调整池中的线程数。

理解了上面关于 ThreadPoolExecutord 的介绍，应该就基本能了解它的一些使用，不过在 ThreadPoolExocutor里面有个关键的 Worker 类，所有的线程都要经过 Worker 的包装。这样才能够做到线程可以复用而无需重新创建线程。

@Autowired先根据type注入，@Resource先根据name注入

Mysql创建库指定默认编码为utf8mb4

CREATE DATABASE `wordpress` DEFAULT CHARACTER SET utf8mb4 COLLATE utf8mb4\_unicode\_ci;

Msql建表时设置时间字段修改时更新当前时间

CREATE TABLE `user`  (

  `id` int(10) UNSIGNED NOT NULL AUTO\_INCREMENT,

  `gmt\_create` datetime(3) NULL DEFAULT CURRENT\_TIMESTAMP(3),

  `gmt\_modified` datetime(3) NULL DEFAULT CURRENT\_TIMESTAMP(3) ON UPDATE CURRENT\_TIMESTAMP(3),

  `name` varchar(255) CHARACTER SET utf8mb4 COLLATE utf8mb4\_0900\_ai\_ci NULL DEFAULT '',

  PRIMARY KEY (`id`) USING BTREE

) ENGINE = InnoDB CHARACTER SET = utf8mb4 COLLATE = utf8mb4\_0900\_ai\_ci ROW\_FORMAT = Dynamic;

Centos使用yum安装jdk1.8

yum install java-1.8.0-openjdk

linux配置ssh免登陆

<https://www.cnblogs.com/linwenbin/p/11612387.html>

Java 自定义注解支持的参数类型：

1.所有基本数据类型（int,float,boolean,byte,double,char,long,short)

2.String类型

3.Class类型

4.enum类型

5.Annotation类型

6.以上所有类型的数组

Oracle查询表的索引

**select** **\*** **from** user\_indexes **where** table\_name**=**upper('表名');

mysql查看表的索引

**show** **index** **from** 表名;

centos索buff/cache使用过高，解决办法

echo 1 > /proc/sys/vm/drop\_cach

drop\_caches的值可以是0-3之间的数字，代表不同的含义：  
            0：不释放（系统默认值）  
            1：释放页缓存  
            2：释放dentries和inodes  
            3：释放所有缓存

Oracle递归查询

START WITH CONNECT BY PRIOR

Mysql参考：<https://www.jianshu.com/p/5f8bf91376b9>

windows使用cmd卸载安装程序：

1、输入wmic进入\*\*模式

2、输入product list，等待一会

3、获取程序名称

4、另开一个cmd窗口，输入

wmic product where '(name="程序名称")' call uninstall