文件操作

文件编辑

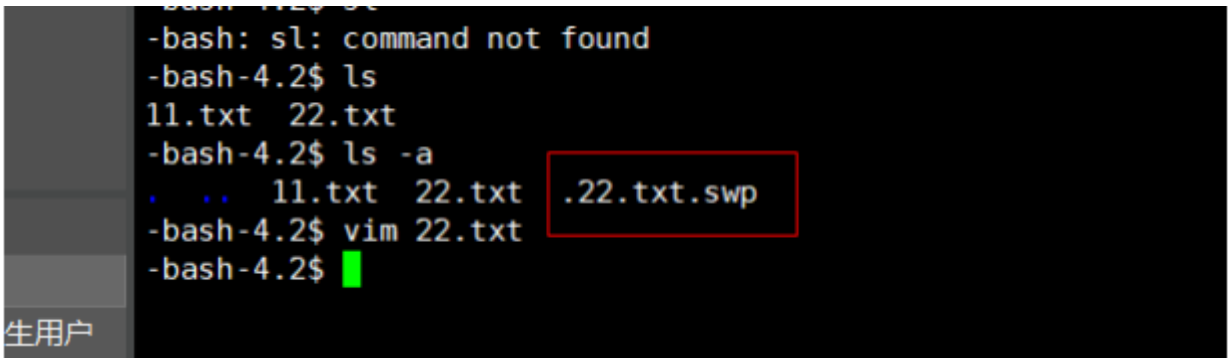
vi :文件编辑工具，相当于一个记事本，安装vim命令`yum install vim`

```
vim 文件    #使用vim打开一个文件用来编辑，如果文件不存在，则新建文件后编辑
vim 文件 +10    #使用vim打开一个文件用来编辑，并且光标自动在第10行
    #默认命令模式，只能接受命令，不能输入内容
        ZZ        #保存退出
        dd/2dd    #用来删除一行/2行数据
        u         #撤销
        yy/2yy    #复制一行/2行
        p         #粘贴
        G         #定位到最后一行
        gg/2gg    #定位到第一行/2行
        $         #定位到这一行的行尾
        0/^       #定位到这一行的行首
        x/2x      #删除光标右边一个/2个的数据
        X/2X      #删除光标左边一个/2个的数据
        ctrl+r    #反撤销
    #插入模式
        i         #在本行首插入数据
        I         #在本行第一个非空字符前面插入数据
        a         #在光标的右侧插入数据
        A         #在光标所在行的结尾插入数据
        s         #删除光标所在位置的文字，并插入数据
        S         #删除光标所在行的文字，并插入数据
        o         #在光标所在行的上一行插入数据
        O         #在光标所在行的下一行插入数据
        esc       #退出插入模式
    #底线命令模式
        :w        #保存修改
        :q        #退出
        :wq       #保存退出
        :q!       #不保存强制退出
        :e!       #放弃修改，回到最初状态，不退出
        :w文件名  #文件另存为
        :set nu #显示行号
        :n        #定位到n行
        :/A       #搜索文件内的所有A，n下一个匹配，N上一个匹配
        :%s/原内容/新内容       #默认只会替换每一行第一次匹配到的数据
        :%s/原内容/新内容/g     #替换所有数据
        :5,10/原内容/新内容     #默认只会替换文件5-10行中每一行第一次匹配到的数据
        :5,10/原内容/新内容/g #替换文件5-10行中所有数据
```

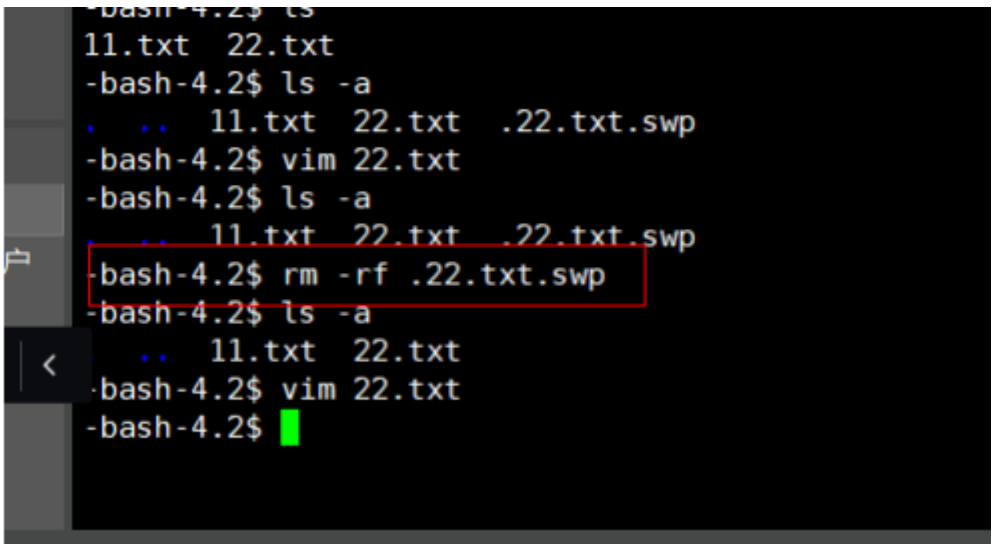
vim非法退出问题

```
E325: ATTENTION
Found a swap file by the name ".22.txt.swp"
          owned by: student   dated: Wed Mar 30 20:40:38 2022
         file name: ~student/test1/ttttt/22.txt
          modified: YES
         user name: student   host name: hecs-34902
        process ID: 17112
While opening file "22.txt"
             dated: Wed Mar 30 20:35:19 2022

(1) Another program may be editing the same file.  If this is the case,
    be careful not to end up with two different instances of the same
    file when making changes.  Quit, or continue with caution.
(2) An edit session for this file crashed.
    If this is the case, use ":recover" or "vim -r 22.txt"
    to recover the changes (see ":help recovery").
    If you did this already, delete the swap file ".22.txt.swp"
    to avoid this message.

Swap file ".22.txt.swp" already exists!
[O]pen Read-Only, (E)dit anyway, (R)ecover, (D)elete it, (Q)uit, (A)bort:
```

怎么看有这个问题呢 ls -a查看隐藏文件夹



或者xxx.swo把他们删除掉 使用 rm -rf



文件权限

chmod 修改文件权限

```
chmod o+w    #给文件其他权限增加写的权限
chmod o-w    #给文件其他权限取消写的权限
    o    #其他
    u    #所有者
    g    #所属组
    a    #所有
```

文件归属

```
sudo chgrp 组名 文件名    #将文件的所属组改为指定组
sudo chown 用户名 文件名    #将文件的所属者改为指定用户
```

文件解压缩

zip格式

```
zip a.zip 文件名    #将指定文件或文件夹压缩为a.zip
unzip 压缩包名      #将a.zip解压缩
```

gzip格式

```
gzip 文件名    #将指定文件替换压缩为gzip格式的压缩包
    -k   #保留原有的数据
    -r   #压缩文件夹内的没有个文件
gunzip 压缩包名 #将指定压缩文件替换解压
    -k   #保留原有数据
```

bzip2

```
bzip2 a.bzip2 文件名    #将指定文件或文件夹压缩为a.bzip2
```

tar

```
tar  #可以对tar后缀的压缩文件拆包或把文件压缩成tar文件
    c    #打包
    x    #拆包
    t    #不拆包查看内容
    v    #查看过程
    f    #指定文件
    z    #使用gzip压缩
    j    #使用bzip2压缩
    C    #指定解压路径
tar -xf test.tar 压缩文件 -C 路径    #将指定的压缩文件解压到指定的路径
tar -cf test.tar 文件名1 文件名2     #将指定文件或文件夹打包为test.tar
tar -xf test.tar           #将指定包解包
```

```
tar -tf test.tar          #不拆包查看指定打包文件
tar -zcvf test.tgz 文件1 文件1   将指定文件或文件夹压缩为test.tgz
tar -zxvf test.tgz        解压指定文件
tar -jcvf test.tbz 文件名   将指定文件或文件夹压缩为test.tbz
tar -jxvf test.tbz        解压指定文件
```

文件上传，下载[重要]

```
rz  # windwos 文件传送到linux
sz  # 文件 linux里面文件传送到windows
# 还有是工具 scp
```

网络请求

查看网卡信息

```
ifconfig
```

测试网络连接情况

```
ping ip地址
    -c #ping的次数
    -l #每次ping的时间间隔
```

网络状态信息

```
netstat -anop #查看网络连接状态
    -t #列出所有tcp
    -l #只显示监听端口
    -n #以数字形式显示地址和端口号
    -p #显示进程的pid和名字
    -u #列出所有udp
    -a #打印所有的内容
```

jq命令

一般用于json数据的格式化

```
echo '{"a":11, "b":12}' | jq '.'    #格式优化
echo '{"foo":42, "bar":"less"}' | jq .foo  #提取键为foo的值
echo '[{"a":1, "b":2}, {"c":3, "d":4}]' | jq .[0] #提取数组中下标为0的数据
echo '[{"a":1, "b":2}, {"c":3, "d":4}]' | jq .[]  #从数组中提取所有数据
echo '[{"a":1, "b":2}, {"c":3, "d":4}, {"e":5, "f":6}]' | jq .[0, 1] #提取数组中下标为0，1的数
据
echo '{"a":1, "b":2, "c":3, "d":4}' | jq [.a, .b]  #把数组中键为a，b的值提取出来重组成数组
echo '{"a":1, "b":2, "c":3, "d":4}' | jq {"tmp":.a}  #把数组中键为a的值提取出来重组成数组
```

进程性能

进程管理

```
ps  #进程列表快照
    ps -ef  #获得所有进程列表
    ps aux  #获得所有进程列表，并提供更多数据
    ps --pid pidlist  #gnu风格参数
    ps aux -m   #查看进程中的线程，会比设置的多一个，因为有个主线程
    ps -o 指标 -M #获取所有进程的自定义输出指标，可以使用man ps查看所有的关键字指标
jobs    #列举后台的进程
top     #交互式进程观测，持续监控系统性能
    -d  #更新间隔时间
    -n  #获取多次CPU的执行情况
    -p  #获取指定进程的数据
    -b  #批处理模式
    -u  #指定用户
    top -b -n 3 -d 1  #每间隔1秒获取一次数据，共获取3次数据
kill    #结束进程
killall #结束所有进程
xargs   #使用awk或sed等命令，通过管道符将多个ID传给kill的时候，加上该命令，可以正确执行xargs kill，否则报错
fg      #将后台中的命令调至前台继续运行
bg      #将一个进程切换到后台
ctrl z  #将一个正在前台执行的命令放到后台，并且停止
&       #在所有的命令后面添加，可以把这个命令放到后台执行
w   #显示所有连接该服务器的地址
```

top信息

```
Tasks：进程总数
running：正在运行的进程数
sleeping：睡眠的进程数
stopped：停止的进程数
zombie：僵尸进程数
Cpu：
    us (user time)：用户空间占用 CPU 百分比
    sy (system time)：内核空间占用 CPU 百分比
    ni (nice)：改变过优先级的进程占用CPU的百分比
    id (idle)：空闲CPU百分比
    wa：IO等待占用CPU的百分比
    hi：硬中断（Hardware IRQ）占用CPU的百分比
    si：软中断（Software Interrupts）占用CPU的百分比
进程
    PR：进程优先级
    NI：nice值。负值表示高优先级，正值表示低优先级
    VIRT：进程使用的虚拟内存总量，单位kb。VIRT=SWAP+RES
    RES：进程使用的、未被换出的物理内存大小，单位kb
    SHR：共享内存大小，单位kb
    S：进程状态：D=不可中断的睡眠状态，R=运行，S=睡眠，T=跟踪/停止，Z=僵尸进程
    %CPU：上次更新到现在的CPU时间占用百分比
```

```
    %MEM：进程使用的物理内存百分比
    TIME+：进程使用的CPU时间总计，单位1/100秒
    COMMAND：命令名/命令行
```

CPU

```
cat /proc/cpuinfo    #查看CPU信息
vmstat  #展现给定时间间隔的服务器的状态值
```

内存

```
free    #查看内存信息
    -h  #使用更加贴近人类识别的格式显示，文件大小以KMGT显示
    -m  #用 M显示内容使用情况
```

字段说明

```
total 总物理内存
used 已经使用的物理内存
free 没有使用过的物理内存
shared 多进程共享内存
buff/cache 读写缓存内存，这部分内存是当空闲来用的，当free内存不足时，linux内核会将此内存释放
buffer是即将要被写入磁盘的，而cache是被从磁盘中读出来的。
available 还能被“应用程序"使用的物理内存
```

重定向

重定向，可以将结果重定向到指定地

```
ps -ef > p.txt   #将任务监听结果存到p.txt文件内，如果文件存在，则覆盖内容
ps -ef 1> p.txt  #将任务监听结果存到p.txt文件内，指令报错，则不进行重定向，如果文件存在，则覆盖内容
ps -ef 2> p.txt  #将错误结果存到p.txt文件内，指令不报错，则不进行重定向，如果文件存在，则覆盖内容
ps -ef &> p.txt  #将所有结果存到p.txt文件内，如果文件存在，则覆盖内容
cat 文件1 文件2 > 文件3    #把文件1和文件2的内容合并到文件3中
ps -ef >> p.txt  #将任务监听结果存到p.txt文件内，如果文件存在，则追加内容
read a < /tmp/1  #将1文件中的内容重定向到a文件中
ls not_exist_dir > /tmp/1 2>&1  #将错误输出重定向到1文件中，2是错误输出
```

三剑客和管道符

grep

参数

```
grep pattern filename
    -v #显示不被匹配到的行
    -i #忽略字符大小写
    -n #显示匹配的行号
```

```
    -c #同级匹配的行数
    -o #仅显示匹配到的字符串
    -E #使用ERE，相当于egrep
    -A3 #显示符合条件的行的同时，还要显示之后的3行
    -B2 #显示符合条件的行的同时，还要显示之前的2行
    -C2 #显示符合条件的行的同时，还要显示前后的2行
    -r #搜索的目标为目录的时候，必须加-r
    -h #搜索的目标为目录时，不显示所属的文件，只显示内容
    -l #搜索的目标为目录时，显示所属的文件
    -L #搜索的目标为目录时，显示不包含搜索内容所属的文件
    --line-buffered #可以打印实时的内容
        top -p 进程号 -b -d 1 -n 20 | grep --line-buffered -i aliyun | grep -i dun
```

示例

```
grep -r leo . --include "*.java" #在当前目录及所有子目录查找所有java文件中查找leo
grep -rni -C5 'link_params' --include '*.jsp' --exclude-dir={.git,vendor,log}/data/web #在/data/web目录下，递归所有扩展名为.jsp的文件，排除 .git,vendor,log 这三个目录，查找所有包含字符串'link_params'的文件及文件内容（不区分大小写），并展示相应行的上下5行内容，以及行号
```

sed

```
sed [-hnV][-e<script>][-f<script文件>][文本文件]
    -h #显示帮助
    -n #仅显示表达式处理后的结果
        sed -n '2p' #打印第二行
        sed -n '3,32p' #打印第3-32行
        sed -n '/c/p' #打印匹配到c的行
    -e #以选项中指定的表达式来处理输入的文本文件
        #匹配
        /pattern/    #正则匹配
        /a/,/b/      #区间匹配
    -f #以选项中指定的表达式文件来处理输入的文本文件
    -i #sed默认不会修改文本内容，只会修改读取到内存的内容，加上-i后可以修改源文件
    -E #扩展表达式
    -debug #调试
    a #追加 sed -e '4 a newline'，在第4行追加内容newline
    c #取代 sed -e '2,5c helloword'，2-5行用新内容helloword取代
    d #删除 sed -e '2,5d'，删除2-5行
    ! #取反 sed -e '2,5!d'，获取除了2-5行外的所有内容
    $ #最后一行
    i #插入内容newline到匹配行之前 sed -e '2i newline'
    e #执行命令
    p #打印 sed -n '/root/p'
    s #替换 sed -e 's/正则/替换内容/g'
    & #原始数据，可通过该符号指定替换内容的前后插入数据
        sed -e 's/:/123&/g' #在：前添加123
sed 's#([0-9])|([a-z])#\2\1#' #反向引用，分组匹配0-9和a-z的内容，然后a-z放在前面，0-9放在后面，\2匹配第二个，\1匹配第一个
```

awk

参数

```
awk 'pattern{action}' [filename]
    -pattern #匹配规则
        awk 'BEGIN{}END{}'  #开始和结束
        ! #取反
        #正则匹配
            awk '/Running/' 匹配到指定内容时，打印整行
            awk '$1>2,$2=="b"'  比较匹配
            awk '$2~/xxx/'  字段匹配，指定列匹配时必须带~
        #行数表达式
            awk 'NR==2' 取第二行
            awk 'NR>1'  去掉第一行
        #区间选择
            awk '/aa/,/bb/' 取aa-bb的所有行数据
            awk '/1/,NR==2' 取1和之后的数据，直到第2行结束
    -action #对匹配到的内容执行的命令
        {print $0} {print $2}   #打印整行和每行的第二列
        {$1='abc'}  #赋值，修改指定内容
        {$1=$1;print $0}  #更新后内容
    fflush #可以打印实时的内容
    FILENAME #awk浏览的文件名
    BEGIN #处理文本之前要执行的操作
    END #处理文本之后要执行的操作
    FS #设置输入域分隔符
    NF #浏览记录的域的列数
    NR #已读的记录数（行数）
    OFS #输出域分隔符
    ORS #输出记录分隔符
    RS #控制记录分隔符,改变换行的符号
    $0 #整条记录
    $1 #表示当前行的第一个列内容.....以此类推
    -F #指定字段分隔符，可以用|指定多个-多分隔符-F'<|>'
    BEGIN{FS="_"} #也可以表示分隔符
    $NF #代表最后一个字段
    $(NF-1) #代表倒数第二个字段
```

示例

```
cat service.log -n | grep 11408 | awk '{print $8}' | awk 'BEGIN{FS="|"}{print $4,$5}' | jq
'.'

#获取a,b三个月的总数
echo 'a, 1, 10
a, 2, 20
a, 3, 30
b, 1, 5
b, 2, 6
b, 3, 7' | awk '{data[$1]+=$3}END{for(k in data) print k,data[k]}'

#获取a,三个月的平均数
echo 'a, 1, 10
```

```
a, 2, 20
a, 3, 30
b, 1, 5
b, 2, 6
b, 3, 7' | awk '{data[$1]+=$3;count[$1]+=1;}END{for(k in data) print k,data[k]/count[k]}'
```

管道符

```
ls | ps -ef #将前一个命令结果传给后一个命令执行
    echo 1 | {read a; echo input is $a}  #将内容1，传给下个命令,管道符后面有两个命令，要用{}括起来，否则只执行第一个命令
    echo 1 | while read a; do echo $a; done  #使用while read组合可以不使用{}
```

环境配置

path变量

```
which python   #查询python在系统的环境变量路径
```

PATH变量是一个路径列表，以：隔开

如果可执行程序所在的目录在PATH变量的路径列表里，那么输入命令时可省略路径

路径列表前面的路径为优先匹配路径，可以用来实现新老版本的命令更换

$HOME、$PATH中有系统自带的环境变量路径

定义一个环境变量，可以在Linux命令行直接使用echo $x的释放打印值，定义完后需要source刷新文件

```
export x=100
```

如果有命令在其他文件夹，需要添加到/usr/bin中，可以使用环境变量定义,这个配置是临时的，如需永久生效，在~/.bashrc内添加

```
export PATH=$PATH:命令所在的绝对路径   # 定义一个环境变量PATH，引用系统的环境变量$PATH
```

软件管理

Ubuntu

安装包后缀是.deb

```
dpkg #下载离线安装包
apt  #直接在线安装
```

CentOS

rpm 下载离线安装包

```
rpm -ivh 包名.rpm #安装.rpm后缀名的软件，并显示安装详情
rpm -qa #列出服务器所有安装软件
rpm -e 包名    #删除安装软件
```

yum 直接在线安装

```
 yum seach 包名   #查找对应的安装包
 yum install 包名.rpm #安装.rpm后缀名的软件
 yum list install    #列出服务器所有安装软件
 yum remove 包名   #删除安装软件
 yum check-update    #检查更新
 yum update 包名   #更新指定的软件
```

wget 下载

```
wget 网址
```

通用

下载.tgz文件，直接解压缩，配置环境变量使用

node.js安装

```
下载源码
tar -zxvf node.tgz
cd node
./configure & make
make install
cd node
```

java安装的步骤

安装方式一

1、通过filezilla这个工具，连接上Linux服务器，然后将我们准备好的Java和tomcat的安装包传输到服务器中。

2、对jdk进行解压，命令是 tar zxvf 文件名

3、在根目录的usr这个文件夹里面创建一个叫java的文件夹。

4、将我们解压后出现的那个文件夹移动到上一步创建的Java文件夹中。

5、进入到根目录下面的etc文件夹中，使用vi命令 编辑 profile 这个文件。

6、讲这段内容写到profile这个文件里面的done这个一行下面。

```
export JAVA_HOME=/usr/java/jdk1.8.0_211
export CLASSPATH=.:$JAVA_HOME/jre/lib/rt.jar:$JAVA_HOME/lib/dt.jar:$JAVA_HOME/lib/tools.jar
export PATH=$PATH:$JAVA_HOME/bin
```

7、使用 `source profile` 这个命令。让Java的环境变量的配置生效。

8、检查Java是否安装成功。命令有2个。 `java -version` 和 `javac -version`

安装方式二

```
yum search jdk
yum install -y java-11-openjdk
```

查看当前系统所有Java程序并指定使用

```
sudo alternatives --list    列出当前系统安装的所有Java程序
sudo alternatives --config java 选择系统中想用的Java程序
```

maven环境配置

官网下载压缩包

解压

编辑bash_profile文件

```
vim ~/.bash_profile
    MAVEN_HOME=解压后的bin路径
    export PATH=$MAVEN_HOME/bin:$PATH
```

maven打包运行流程

```
git中clone项目
cd 下载的项目中/complete
`mvn clean package -DskipTests`  清理项目，并重新编译打包项目，不加-DskipTests会执行单元测试
`java -Dserver.port=8888 -jar` target/jar包  # 启动项目
```

maven常用命令

```
mvn clean test
mvn package install
mvn test -DskipTests
mvn clean org.jacoco:jacoco-maven-plugin:0.8.5:prepare-agent test \
    jacoco-maven-plugin:0.8.5:report \
    -Dmaven.test.failure.ignore=true \
    -Dmaven.test.skip=false
```

tomcat的安装步骤

1、通过filezilla这个工具，连接上Linux服务器，然后将我们准备好的Java和tomcat的安装包传输到服务器中。

2、对tomcat进行解压，命令是 `tar zxvf` 文件名

3、在根目录的usr这个文件夹里面创建一个叫tomcat的文件夹。

4、将我们解压后出现的那个文件夹移动到上一步创建的tomcat文件夹中。

5、对tomcat进行配置，进入到tomcat下面的那个什么apache...的文件夹里的bin目录中，使用vi命令，对setclasspath.sh这个文件进行编辑。

6、在setclasspath.sh这个文件的第二排写入以下内容：

```
export JAVA_HOME=/usr/java/jdk1.8.0_211
export JAVA_JRE=/usr/java/jdk1.8.0_211/bin
```

7、启动tomcat。在bin目录下运行./startup.sh这个文件。

8、在阿里云中，打开端口号。（一般这个步骤可以不做，除非你的服务器的8080端口没有打开。）

9、在浏览器中访问自己的tomcat的页面。

MySQL的离线rpm包安装步骤

1、通过filezilla这个工具，连接上Linux服务器，然后将我们准备好的mysql的安装包传输到服务器中。

2、对mysql进行解压，命令是 tar zf（xf?） 文件名

3、安装numactl（必要组件，不安装会导致后面的步骤出现依赖的问题。）

```
yum -y install numactl
```

3.1、卸载mariadb（这是系统自带的数据库，不卸载会导致MySQL安装失败。）

```
rpm -qa | grep -i mariadb
rpm -e --nodeps mariadb-libs-5.5.60-1.el7_5.x86_64   #这个文件名字是上一步查出来的
```

3.2、安装mysql，按顺序安装下面4个rpm（版本号可能不一样）。

```
rpm -ivh mysql-community-common-5.7.23-1.el7.x86_64.rpm
rpm -ivh mysql-community-libs-5.7.23-1.el7.x86_64.rpm
rpm -ivh mysql-community-client-5.7.23-1.el7.x86_64.rpm
rpm -ivh mysql-community-server-5.7.23-1.el7.x86_64.rpm
```

4、等待安装结束后，启动数据库。命令：`systemctl start mysqld.service`

5、检查数据库是否运行成功，命令：`systemctl status mysqld.service`

6、数据库安装成功后，先生成一个默认密码，查看密码的命令：`cat /var/log/mysqld.log | grep password`

7、使用上一步获取的密码连接数据库，`mysql -u root -p` 这个步骤你们肯定没问题的。

8、进入数据库后，必须修改密码才能做其他的操作，所以修改密码为1qaz!QAZ，命令：`ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password BY '1qaz!QAZ';`

9、然后退出数据库，用新密码重新登录。

10、然后创建一个具有远程访问权限的账号。有3条语句，如下：

```
create user 'root'@'%' identified with mysql_native_password by '1qaz!QAZ';
grant all privileges on *.* to 'root'@'%' with grant option;
flush privileges;
```

10.1、为了让数据库的密码能修改为123456，所以我们需要对数据库进行一些配置。命令如下，完成后，就可以通过navicat来随意修改密码了。

```
SHOW VARIABLES LIKE 'validate_password%';  #查看数据库的密码规则
set global validate_password_policy=LOW;  #修改密码强度要求
set global validate_password_length=6;  #修改密码长度要求
```

11、好了数据库的设置结束了，你现在可以尝试能不能用navicat来连接了。

12、如果不能，那八成是端口的问题。所以检查阿里云的控制台的安全组是否开放端口。

13、通过命令查看当前已经开放的端口： `netstat -ntlp`

14、如果不存在3306，那么通过以下2个命令打开3306端口号。

```
#将3306端口添加到防火墙例外并重启:
firewall-cmd --zone=public --add-port=3306/tcp --permanent
firewall-cmd --reload
```

15、再次尝试navicat能连接了不。

MySQL的在线安装步骤

1、下载mysql官方的yum工具。备注：yum可以理解成是一个应用市场。

```
wget -i -c http://dev.mysql.com/get/mysql57-community-release-el7-10.noarch.rpm
```

2、用yum安装MySQL的基础配置。

```
yum -y install mysql57-community-release-el7-10.noarch.rpm
```

如果默认安装的MySQL版本是8.0或其他，一般开发用5.7，使用 `vi /etc/yum.repos.d/mysql-community.repo` 修改mysql-community.repo文件，将文件中mysql80的enable的值改为0，mysql57的enable值改为1

3、用yum下载并安装mysql的核心服务。

```
yum -y install mysql-community-server
```

4、等待安装结束后，启动数据库。命令： `systemctl start mysqld.service`

5、检查数据库是否运行成功，命令： `systemctl status mysqld.service` **或** `ps -aux|grep -v grep|grep mysql`

6、数据库安装成功后，先生成一个默认密码，查看密码的命令： `cat /var/log/mysqld.log | grep password`

7、使用上一步获取的密码连接数据库， `mysql -u root -p` **这个步骤你们肯定没问题的。**

8、进入数据库后，必须修改密码才能做其他的操作，所以修改密码为1qaz!QAZ，命令： `ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password BY '1qaz!QAZ';`

9、然后退出数据库，用新密码重新登录。

修改密码：`update user set authentication_string=password('你的密码') where user='root'` 或者

`ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password BY '1qaz!QAZ';`

10、然后创建一个具有远程访问权限的账号。有3条语句，如下：

```
create user 'root'@'%' identified with mysql_native_password by '1qaz!QAZ';
grant all privileges on *.* to 'root'@'%' with grant option;  # all privileges全部权限，也可以指定select|insert|update等
flush privileges;  # 刷新使权限生效
```

修改mysql配置文件（不同操作系统，不同sql版本，配置文件存储位置不同）

```
sudo vim /etc/my.cnf
```

将bind-address=127.0.0.1注释掉（如果这段代码存在的话），让计算机允许mysql远程登陆

删除用户：user mysql

`select host,user from user` # 查看表里现在的用户

`drop user 用户名@'%'`

查看权限：`show grants` # 查看当前用户权限

`show grants for 'root'@'localhost'` # 查看指定用户权限

回收权限：`revoke all privileges on *.* from 'root'@'localhost'` # 回收指定用户所有权限

`revoke grants option on *.* from 'root'@'localhost'` # 回收指定用户权限的传递

10.1、为了让数据库的密码能修改为123456，所以我们需要对数据库进行一些配置。命令如下，完成后，就可以通过navicat来随意修改密码了。

```
SHOW VARIABLES LIKE 'validate_password%';  #查看数据库的密码规则
set global validate_password_policy=LOW;  #修改密码强度要求
set global validate_password_length=6;  #修改密码长度要求
```

11、好了数据库的设置结束了，你现在可以尝试能不能用navicat来连接了。

12、如果不能，那八成是端口的问题。所以检查阿里云的控制台的安全组是否开放端口。

13、通过命令查看当前已经开放的端口： `netstat -ntlp`

14、如果不存在3306，那么通过以下2个命令打开3306端口号。

将3306端口添加到防火墙例外并重启：

```
firewall-cmd --zone=public --add-port=3306/tcp --permanent
firewall-cmd --reload
```

15、再次尝试navicat能连接了不。

16、安装mycli，支持语法提示和高亮

```
sudo pip3 install mycli
```

修改配置文件可以换行

```
vim ~/.myclirc
```

multi_line从False改为True

sql连接时的语句`mysql -u root -p`改为`mycli -u root -p`

python3和pip3的安装

第一种安装方式

1、安装各种需要的依赖包和组件，有两个命令。

```
yum install zlib-devel bzip2-devel openssl-devel ncurses-devel sqlite-devel readline-devel
tk-devel gcc make
yum install libffi-devel
```

2、下载python

```
wget https://www.python.org/ftp/python/3.7.0/Python-3.7.0.tgz
```

3、解压安装，四个步骤。

```
tar -zxvf Python-3.7.0.tgz
cd Python-3.7.0
./configure
make && make install
```

4、配置环境变量

```
mv /usr/bin/python /usr/bin/python.bak
ln -s /usr/local/bin/python3 /usr/bin/python
mv /usr/bin/pip /usr/bin/pip.bak
ln -s /usr/local/bin/pip3 /usr/bin/pip
```

5、验证是否安装成功，两个命令。

```
python -V
pip -V
```

6、配置yum，不然这个就不能用了。

```
vi /usr/libexec/urlgrabber-ext-down
```

把第一行的python改为python2.7

```
vi /usr/bin/yum
```

把第一行的python改为python2.7

第二种安装方式

centeOS自带Python2，直接安装Python3会报错

使用EPEL安装

使用`sudo yum install epel-release`安装EPEL

使用`sudo yum install python3`安装Python3

使用源码安装和安装nginx步骤一致

查到Python快捷方式可删除`sudo rm -rf python`

重新指定快捷方式指定执行的文件`sudo ln -s python3.6 python`

Linux管理虚拟环境

使用`sudo pip3 install virtualenv`安装virtualenv

使用`sudo pip3 install virtualenvwrapper`安装virtualenvwrapper

在家目录创建一个文件夹用于存放虚拟环境`mkdir .virtualenvs`

编辑`sudo vim ~/.bashrc`

文件最后添加

```
export VIRTUALENVWRAPPER_PYTHON=/usr/bin/Python3.6  # 指定新虚拟环境使用的Python版本
export WORKON_HOME=$HOME/.virtualenvs  # 指定创建新虚拟环境保存在哪个文件夹下
source /usr/local/bin/virtualenvwrapper.sh  # 执行virtualenvwrapper.sh脚本
```

刷新配置文件`source .bashrc`

创建虚拟环境`mkvirtualenv spider`，**放在了.virtualenvs里**

使用`workon 虚拟环境名`切换虚拟环境

退出当前虚拟环境`deactivate`

删除虚拟环境`rmvirtualenv 虚拟环境名`

导出项目涉及的所有包

`pip freeze > 文件名.txt`

安装所有包

`pip3 install -r 文件名.txt`

源码安装

```
curl -O 包地址
tar -zxvf Python-3.10.0.tgz
cd Python-3.10.0
./configure & make
make install
配置环境变量
```

Nginx的安装步骤

第一种

安装:

```
sudo yum install nginx
```

查看nginx的版本

```
nginx -v
```

查看nginx是否安装成功

```
ps -aux|grep nginx
```

查找和nginx相关的文件

```
whereis nginx
```

启动

```
sudo systemctl start nginx.service
```

开机自启

```
systemctl enable nginx.service
```

查看启动状态

```
systemctl status nginx.service
```

关闭防火墙

```
systemctl stop firewalld.service
```

配置文件

```
vi /etc/nginx/nginx.conf
```

搜索html，找到root开头的行，修改静态页面的存放路径

listen 监听端口

卸载nginx

```
yum remove nginx
```

第二种

使用wget 网址 从官网下载nginx

解压

找到nginx内的configure执行 `./configure --prefix=/usr/local/nginx` **将nginx安装到指定路径**

安装相关依赖 `sudo yum install gcc-c++ pcre-devel zlib zlib-devel openssl openssl-devel`

再次执行 `./configure --prefix=/usr/local/nginx`

执行命令编译安装 `sudo make&&sudo make install`

进入到安装目录,执行 `nginx sudo ./nginx` **或使用** `sudo systemctl start nginx.service` **启动**