**1、本章面试题**

            什么是 shell脚本，如何使用

rpm和yum是什么，如何使用

**2、知识点**

**2.1、课程回顾**

             ps top kill du df fdisk系统维护命令

vi vim编辑器

用户管理 (useradd usermod userdel)，组管理(groupadd,groupmod,groupdel),

sudo命令(sudo 命令名称 /etc/sudoers文件中，给非root加一行代码)

时间（硬件时间和系统时间 ）网络时间同步 ntpd 服务

**2.2、本章重点**

         shell编程 awk的使用

rpm和yum的使用（软件的维护工具）

**3、具体内容**

**3.1 Shell脚本简介**

**3.1.1 什么是shell脚本**

1.当命令不在命令行中执行，而是从一个文件中执行时，该文件就是shell脚本。

shell脚本是普通的文本文件，由流程控制逻辑和命令构成。

shell脚本通常以.sh作为后缀名，但不是必须的。

2.Shell是一种解释型编程语言，不需要编译，执行时也是按行执行。

3.Shell脚本是由解释器解释执行的，常见的解释器有：

bash dash ash  ksh  sh等

注意，每种Shell解释器对shell的语法解释并不完全相同(sh和bash解析器对ll的解释就不同)，但基本相同。

4. 脚本就是命令的堆砌

5. 脚本可以接受参数，也可以有变量，也可以有流程控制甚至还有函数

**3.1.2 shell脚本的编制、执行和调试**

1.编制：使用文本编辑器vi或者vim

2.执行：执行前必须为脚本文件添加可执行权限

chmod +x script-file

bash script-file   或者  ./script-file

3.调试

bash -x -n -v script-file

-n ：只读取shell脚本，但不会去执行它，常用来测试脚本语法错误

-x ：进入跟踪模式，显示所执行的每一行命令，常用来调试脚本逻辑

**3.1.3 shell版hello word**

1.Shell中的注释用‘#’

#这是注释，独占一行

echo  #这是注释，在脚本代码尾部，注意与代码用空格隔开

2. Shell第一行代码：#!/bin/sh 声明这个script使用的shell解释器

**3.1.4 备份并清空日志**

备份可变文件存放目录/var 下面的log目录，备份成功后，删除原来文件

该脚本存在很多弊端：

1，不能动态指定备份内容

2，备份后的名称不能随意指定

3，即使备份不成功，也会删除源文件

**3.2、变量的定义、引用、删除**

**3.2.1 变量**

1.shell中没有变量类型，本质上都是字符串

2.shell中有四种类型的变量

本地变量：仅在当前shell中有用，在shell外部或者子shell中都不可用

环境变量：当前及子shell有用

位置参数变量：脚本的参数或者函数的参数等

特殊变量：脚本内置的具有特殊用途的变量

**3.2.2 变量的定义及删除**

1.变量定义有以下方式

project= #定义空变量或者清空已有变量

project=/home/workspace   #定义变量并赋值

A=1

project=”/home/workspace${A}”  #定义变量并赋值，双引号对$ `  \无效

project='/home/workspace${A}'    #定义变量并赋值，单引号最强，引什么是什么

export project=/usr/redis    #将变量声明为环境变量，在子shell中也可以使用

2.变量的删除

变量删除使用语句为：unset project

**3.2.3 变量赋值及引用**

1.变量赋值

project=/home/work    #变量重新赋值了

2.变量引用

subproj=${project}/sub   #subproj引用了project，他俩的值一样了。

subproj=$project/sub     #等同于 subproj=${project}

**3.3、位置参数变量及特殊变量**

**3.3.1 位置参数变量**

1.位置参数是一组特殊的内置变量,跟在脚本名后面的用空格隔开的每个字符串

$1 表示第1个参数值，……，$9 表示第9个参数值

${10} 表示第10个参数值， ${11} 表示第11个参数值

2.位置参数的作用

从 shell 命令/脚本 的命令行接受参数

在调用 shell 函数时为其传递参数

示例：

#!/bin/bash

# ScriptName: var1.sh

# To test Positional Parameters & Special Parameters.

echo "Hello,$USER,the output of this script are as follows:"

echo "The script name is : $(basename $0)"

echo "The first param of the script is : $1"

echo "The second param of the script is : $2"

echo "The tenth param of the script is : ${10}"

echo "All the params you input are : $@"

echo "All the params you input are : $\*"

echo "The number of the params you input are: $#"

echo "The process ID for this script is : $$"

echo "The exit status of this script is : $?"

**3.3.2 特殊变量**

1.与命令行参数相关

$\*    将所有位置参量看成一个字符串（以空格间隔） 。String str= "1 2 3 4"

$@   将每个位置参量看成单独的字符串（以空格间隔）。

"1" "2 " "3" "4"

“$\*”   将所有位置参量看成一个字符串（以$IFS间隔）。

 “$@” 将每个位置参量看成单独的字符串（以空格间隔） 。

$0    命令行上输入的Shell程序名。

$#    表示命令行上参数的个数。

2.与进程状态相关

$?  表示上一条命令执行后的返回值

$$  当前进程的进程号

$!   显示运行在后台的最后一个作业的 PID

$\_  在此之前执行的命令或脚本的最后一个参数

位置参数和特殊变量举例，执行脚本

示例：

#!/bin/bash

# ScriptName: ifsargs.sh

#### Set the IFS to | ####

IFS='|'

echo "Command-Line Arguments Demo"

echo "\* All args displayed using \$@ positional parameter \*"

echo $@

echo "\* All args displayed using \$\* positional parameter \*"

echo $\*

echo '\* All args displayed using "$@" positional parameter \*'

echo "$@" #\*\*\* double quote added \*\*\*#

echo '\* All args displayed using "$\*" positional parameter \*'

echo "$\*" #\*\*\* double quote added \*\*\*#

**3.3.3 使用上面所学知识完善打包备份功能：**

vi tar2.sh

复制下面内容：

#!/bin/bash

#接受打包压缩后的文件名称

fn=$1

#接受想备份的目录名称

bd=$2

#打包，并且处理异常，打包备份成功时，删除原来的目录

# 2代表前面命名执行如果有错误 && 当前面一个命令执行成功时，执行后面命令

tar -cjf $fn.tar.bz2 $bd 2>error.log && rm -rf $bd

chmod u+x tar2.sh

./tar2.sh log3 /var/log3

vi tar2.sh 把tar命令改错误 tara

./tar2.sh log4 /var/log4

cat error.log 命令找不到的错误信息

ls /var 存在的，说明没有删

**3.4、常见状态和运算符，命令**

**3.4.1 退出/返回状态**

1.$?：返回上一条语句或脚本执行的状态

0：成功

1－255：不成功

2.exit 命令用于退出脚本或当前Shell

n 是一个从 0 到 255 的整数

0 表示成功退出，非零表示遇到某种失败

返回值 被保存在状态变量 $? 中

**3.4.2 常见的返回状态码**

0：

执行正确

1：

通用错误

126：

命令或脚本没有执行权限

127：

命令没找到

**3.4.3 数据流重定向**

标准输入：代码0，使用 < 或者 <<

标准输出：代码1，使用 > 或者 >>

错误输出：代码2，使用 > 或者 >>

> 表示 写入； >>表示追加

<表示 读入；<<表示结束读入

#!/bin/sh

#这段代码会往hosts文件里追加两行域名的解析

cat >> /etc/hosts << EOF

172.164.118.110 www.baidu.com

154.170.124.121 www.sohu.com

EOF

**3.4.4 命令的连续执行 &&   ||**

1.脚本中经常有多条相关命令，比如：

执行备份日志成功后删除日志

#!/bin/bash

#接受打包压缩后的文件名称

fn=$1

#接受想备份的目录名称

bd=$2

#打包，并且处理异常，打包备份成功时，删除原来的目录

# 2代表前面命名执行如果有错误 && 当前面一个命令执行成功时，执行后面命令

tar -cjf $fn.tar.bz2 $bd 2>error.log && rm -rf $bd

从服务器下载文件失败后从备份服务器下载

2. &&   连起来的命令，后面的命令只有在前一命令执行成功后才执行

3. ||   连起来的命令，后面的命令只有在前一命令执行失败后才执行

**#!/bin/sh**

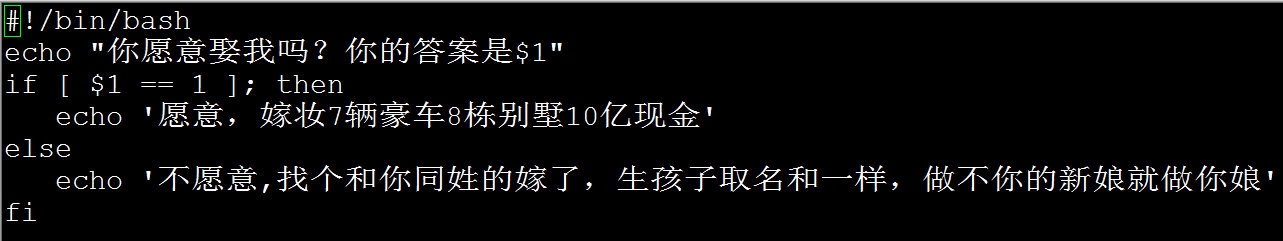
**master\_ip=”**[**www.oracle.com**](http://www.oracle.com)**111” #主服务器IP地址变量**

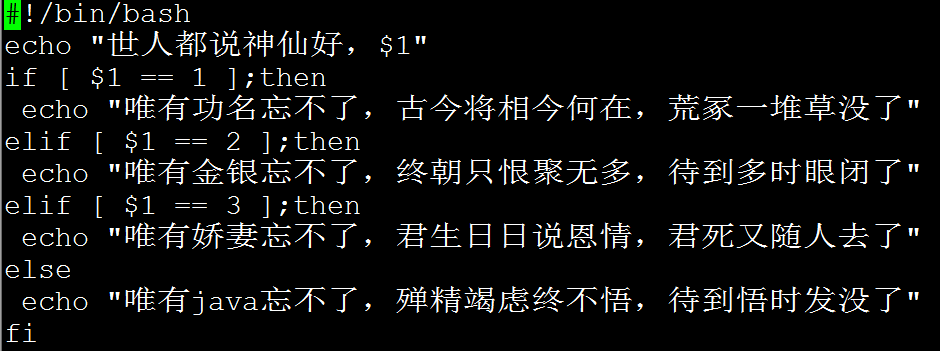
**bak\_ip=”**[**www.oracle.com**](http://www.oracle.com)**” #备份服务器IP地址变量**

**uri=”/otn/java/jdk/8u231-b11-demos/5b13a193868b4bf28bcb45c792fce896/jdk-8u231-linux-x64-demos.tar.gz” #要下载的文件**

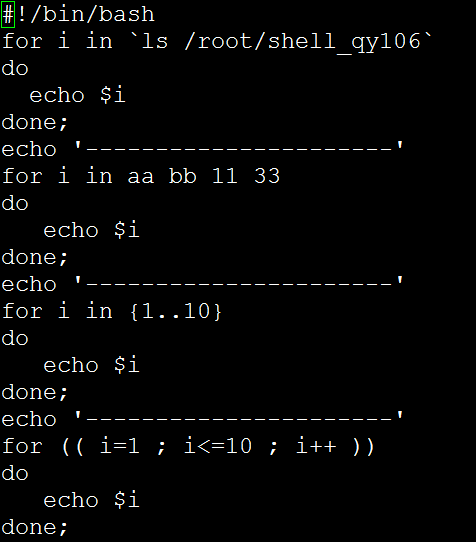
**curl -O “https://${master\_ip}${uri}” || curl -O “https://${bak\_ip}${uri}”**

**3.4.5 if else if elseif for while**

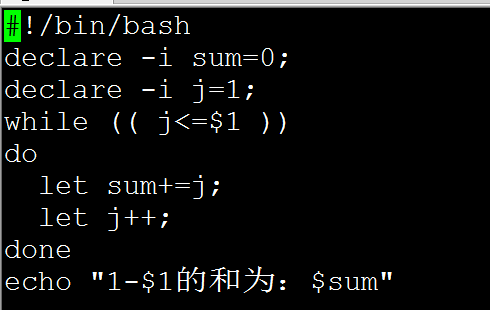




间隔符 ````````` esc下面 1左边的键



**sum+=j 等同于 sum=sum+j**



**3.4.5命令连接常用命令-awk简介**

1.awk是一个非常棒的数据处理程序，它比较倾向于将一行分成多个‘字段’来处理。

类似 String[] strArray = "a,b,c,d".split(","); strArray[0] strArray[3]

2.数据可以来自标准输入、一个或者多个文件、其他进程的输出

3.awk从第1行到最后一行逐行扫描，并执行匹配的操作

4.awk的默认操作是输出匹配行即打印匹配行到标准输出

**3.4.6  awk语法**

1.awk 'BEGIN{语句1;语句2;...}  END{语句1;语句2;...}  模式1{语句1;语句2;...}  模式2{语句1;语句2;...}  模式n...'    [文件名]

2.如果没有文件名，则数据来自标准输入

BEGIN{...} 是awk开始运作前执行的语句；END{...}是awk最终结束后执行的语句，他们可以省略

2. awk默认分隔符是空格和tab，可以在BEGIN语句中为FS内置变量赋值重新定义分隔符

脚本举例如下：

touch tomcat.log

date >> tomcat.log 执行多次

cat tomcat.log

awk 'BEGIN{print "文件开始";} END{print "文件结束";} {print "执行行" NR}' tomcat.log

echo “1 2 3 4” | awk '{print $1$4}'    #$1代表第1列，$4代表第4列

echo “1 2 3 4” | awk '{print $1A$4}'      #中间加入的字符A并不能输出

echo “1 2 3 4” | awk '{print “$1A$4”}'   #参数变量不能加入双引号

echo “1 2 3 4” | awk '{print $1“A”$4}'    #正确方式

echo “1 2 3 4” | awk '{print $1,”A”,$4}'   #注意与上式比较

**3.4.6  awk内置变量及使用**



**1.以冒号为分隔符进行划分字段**

**echo “A:B:C:D” | awk 'BEGIN{FS=”:”} {print $1,$2,$3,$4}'**

**2.分隔符还可以是字符串**

**echo “A--B--C--D” | awk 'BEGIN{FS=”--”} {print $1,$2,$3,$4}'**

**3.分隔符还可以是正则表达式**

**echo “A:B:a-b-c-d” | awk 'BEGIN{FS=”[:-]”}{print $1,$2,$3,$4}'**

**4.忽略第一行数据**

**awk 'NR>1{print $0}' /etc/hosts**

**3.5、软件的安装和维护（RPM）**

**3.5.1 RPM软件包管理及概述**

**RPM是RedHat Package Manager（RedHat软件包管理工具）的缩写，这一文件格式名称虽然打上了RedHat的标志，但是其原始设计理念是开放式的，现在包括RedHat、CentOS、SUSE等Linux的分发版本都有采用，可以算是公认的行业标准了。RPM文件在Linux系统中的安装最为简便。**

**RPM 最早是由 Red Hat 公司提出的软件包管理标准，最初的全称是 Red Hat Package Manager。后来随着版本的升级又融入了许多其他的优秀特性， 成为了Linux中公认的软件包管理标准。被许多Linux发行使用，如：RHEL/CentOS/Fedora, SLES/openSUSE 等。如今RPM是RPM Package Manager的缩写，由RPM社区（http://www.rpm.org/）负责维护。**

**RPM的有点在于1.易于安装、升级便利；2.丰富的软件包查询功能；3.软件包内容校验功能；4.支持多种硬件平台**

**3.5.2 RPM五大功能及组件介绍**

**1.五大功能为：**

**安装——将软件从包中解出来，并安装到硬盘。**

**卸载——将软件从硬盘清除。**

**升级——替换软件的旧版本。**

**查询——查询软件包的信息。**

**验证——检验系统中的软件与包中软件的区别。**

**2.RPM组件：**

**本地数据库**

**rpm及其相关的可执行文件**

**RPM 前端工具，如 yum**

**软件包文件**

**3.5.3 RPM命令的使用**

**rpm的常用选项：**

**i：安装应用程序（install）**

**e：卸载应用程序（erase）**

**vh：显示安装进度（verbose hash）**

**U：升级软件包；（update）**

**qa: 显示所有已安装软件包（query all）**

**3.5.4 RPM命令安装、升级和删除软件**

**1.输出选项：**

**-v：安装时显示软件名称**

**-h：使用“#”显示进度**

**2.rpm file 的URL支持**

**ftp://**

**http://**

rpm -ivh https://download.oracle.com/otn/java/jdk/8u231-b11/5b13a193868b4bf28bcb45c792fce896/jdk-8u231-linux-x64.rpm

**3.rpm包的安装**

**mkdir soft**

**cd soft**

**ls**

**java -version 查看jdk版本 错误没有找到命令**

**rpm -ivh jdk-8u151-linux-x64.rpm**

**java -version**

**rpm -qa | grep jdk**

**配置前：**

**echo $JAVA\_HOME**

**配置环境变量：**

**查看软件安装位置：**

**rpm -ql jdk1.8-1.8.0\_151-fcs.x86\_64 | tail -1**

**/usr/java/jdk1.8.0\_151/src.zip**

/etc/profile环境变量配置文件位置：

vim /etc/profile

环境变量配置：（路径为JDK 的解压路径）

export JAVA\_HOME=/usr/java/jdk1.8.0\_151

export PATH=$JAVA\_HOME/bin:$PATH

export ClASSPATH=.:$JAVA\_HOME/lib/dt.jar:$JAVA\_HOME/lib/tools.jar

//让配置生效

source /etc/profile

//查看配置结果

echo $JAVA\_HOME

**4.rpm包的卸载**

**rpm -e jdk1.8-1.8.0\_151-fcs.x86\_64**

**java -version**

**3.5.5 RPM的基本查询**

**1.查询已安装的所有软件包**

**rpm -qa**

**2.查询软件包是否安装并查看软件包的版本**

**rpm -q <package\_name>**

**rpm -qa | grep jdk**

**rpm -q jdk1.8-1.8.0\_151-fcs.x86\_64**

**3.查询软件包信息**

**rpm -qi <package\_name>**

**rpm -qip <package\_file\_path\_name>**

**rpm -qi jdk1.8-1.8.0\_151-fcs.x86\_64**

**4.查询软件包中所有文件的名称**

**rpm -ql <package\_name>**

**rpm -qlp <package\_file\_path\_name>**

**rpm -ql jdk1.8-1.8.0\_151-fcs.x86\_64**

**5.查询磁盘上的文件是从何软件包安装的**

**rpm -qf <path\_name>**

**rpm -qf /usr/java/jdk1.8.0\_151/**

**3.5.5 RPM校验**

**1.校验有已安装的所有软件包**

**rpm -Va**

**验证内容中的8个信息的具体内容如下：**

S    文件大小是否改变

M    文件的类型或文件的权限（rwx）是否变改变

5    文件MD5校验和是否改变（可以看作文件内容是否改变）

D    装置的主/次代码已经改变

L    Link路径已被改变

U    文件的所属人已被改变

G    文件的所属群组已被改变

T    文件的创建时间忆被改变

**文件类型：**

c    配置文件（config file）

d    普通文件（documentation）

g    "鬼"文件（ghost file），很少发生，通常是指该文件不被软件包所包含

l    授权文件（license file）

r    描述文件（read me）

**2.校验指定的软件包**

**rpm -V <package\_name>**

rpm -V jdk1.8-1.8.0\_151-fcs.x86\_64(删除安装目录下任意文件，再次检查)

rpm -V jdk1.8-1.8.0\_151-fcs.x86\_64(改变任意文件的权限，再次检查)

**3.验证包含指定文件的软件包**

**rpm -Vf <path\_name>**

rpm -Vf /usr/java/jdk1.8.0\_151/

**3.6、软件的安装和维护（yum）**

**3.6.1 yum概述**

**RPM软件包一般非为两大类：1.yy-x.x.rpm 软件包；2.yy-devel-x.x.rpm软件开发包，供别的软件包引用的库包。由此，RPM软件包安装时经常会出现缺少依赖库的现象，有时会产生连锁反应。yum就是为了解决软件依赖问题而生。它的原理是将所有的软件包和软件开发包都放到服务器上，当安装软件包时向服务器请求，由服务器分析出依赖关系，从而提供一整套安装包。**

**Yum（全称为 Yellow dog Updater, Modified）是一个在Fedora和RedHat以及SUSE、CentOS中的Shell前端软件包管理器。基於RPM包管理，能够从指定的服务器自动下载RPM包并且安装，可以自动处理依赖性关系，并且一次安装所有依赖的软件包，无须繁琐地一次次下载、安装。**

**3.6.2 yum的功能介绍**

**1.安装——将软件从包中解出来，并安装到硬盘。**

**2.卸载——将软件从硬盘清除。**

**3.升级——替换软件的旧版本。**

**4.查询——查询软件包的信息。**

**3.6.3 yum源的配置文件**

**[root@localhost yum.repos.d]# vi /etc/yum.repos.d/CentOS-Base.repo**

**[base] 容器名称，一定要放在[]中**

**name 容器说明，可以自己随便写**

**mirrorlist 镜像站点，这个可以注释掉**

**baseurl 我们的yum源服务器的地址。默认是CentOS官方的yum源服务器，是可以使用的，如果你觉得慢可以改成你喜欢的yum源地址**

**enabled 此容器是否生效，如果不写或写成enable=1都是生效，写成enable=0就是不生效**

**gpgcheck 如果是1是指RPM的数字证书生效，如果是0则不生效**

**gpgkey 数字证书的公钥文件保存位置。不用修改**

**3.6.4 yum源的配置**

**方法一： 配置本地yum源：**

**1.列出配置源**

**yum repolist**

**2.配置本地源**

**第一步：**

**从CentOS的官网下载CentOS的完整版iso文件，并上传到Linux文件系统中，例如/opt/tools/CentOS-6.9-x86\_64-bin-DVD1.iso**

**第二步：挂载这个iso文件到/media/CentOS**

**mkdir /media/vcdrom**

**mount -o loop /opt/tools/CentOS-6.9-x86\_64-bin-DVD1.iso /media/vcdrom**

**安装时使用参数enablerepo=c6-media**

**yum --enablerepo=c6-media -y install mysql-server**

方法二：配置阿里源

**1) 安装wget**

**yum install -y wget**

**2) 备份/etc/yum.repos.d/CentOS-Base.repo文件**

**cd /etc/yum.repos.d/**

**mv CentOS-Base.repo CentOS-Base.repo.back**

**3) 下载阿里云的Centos-6.repo文件**

**wget -O CentOS-Base.repo http://mirrors.aliyun.com/repo/Centos-6.repo**

**4) 重新加载yum**

**yum clean all**

**yum makecache**

**3.6.5 yum的基本查询**

**1.查询jdk环境应该安装哪个具体的包名**

**yum search jdk**

**2.查询软件包的功能**

**yum search curl**

**3.查询服务器上所有的已经安装和可以安装的程序包**

**yum list**

yum list rpm

yum list jdk1.8-1.8.0\_151-fcs.x86\_64

**4.查询有哪些软件需要升级**

**yum list updates**

**3.6.6 yum的安装、卸载与升级**

**1.安装**

centos6安装mysql:

**yum install mysql-server #中间会提示输入y/n来确认安装**

**yum -y install mysql-server #不用提示，直接安装**

**centos7安装mysql5.7:**

**一、安装YUM Repo**

**1、由于CentOS 的yum源中没有mysql，需要到mysql的官网下载yum repo配置文件。**

**下载命令：**

**wget https://dev.mysql.com/get/mysql57-community-release-el7-9.noarch.rpm**

**2、然后进行repo的安装：**

**rpm -ivh mysql57-community-release-el7-9.noarch.rpm**

**执行完成后会在/etc/yum.repos.d/目录下生成两个repo文件mysql-community.repo mysql-community-source.repo**

**二、使用yum命令即可完成安装和使用**

**1、安装命令：**

**yum install mysql-server**

**2、启动msyql：**

**systemctl start mysqld #启动MySQL**

**3、获取安装时的临时密码（在第一次登录时就是用这个密码）：**

**grep 'temporary password' /var/log/mysqld.log**

**4 登录**

**mysql -u root -p 回车，复制临时密码**

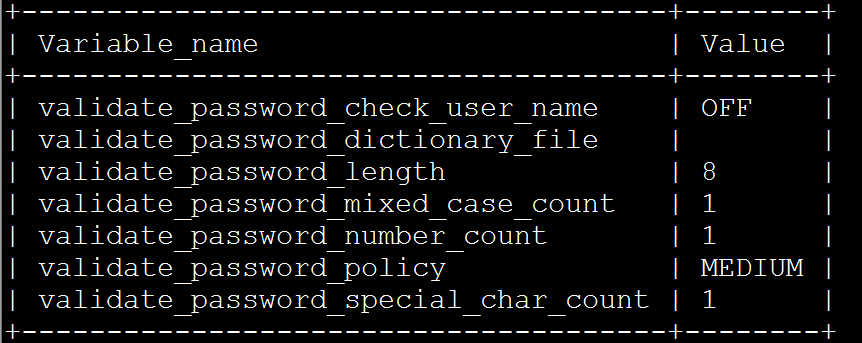
**5，修改密码**

**set password=password('tigeradmin');**

**ERROR 1819 (HY000): Your password does not satisfy the current policy requirements**

**6，查看密码策略，然后再修改**

**SHOW VARIABLES LIKE 'validate\_password%';**



7，或者修改为低等级验证

 set global validate\_password\_policy=LOW;

set password=password('tiger1314');

8，开启远程可以访问：

连接服务器: mysql -u root -p

　 看当前所有数据库：show databases;

　　 进入mysql数据库：use mysql;

　 查看mysql数据库中所有的表：show tables;

　　 查看user表中的数据：select Host, User,Password from user;

　 修改user表中的Host:   update user set Host='%' where User='root';

          说明： % 代表任意的客户端,可替换成具体IP地址。

　 最后刷新一下：flush privileges;

**centos7安装mysql5.6:**

查看已经安装mysql组件：

yum list installed | grep mysql

卸载：

yum -y remove mysql-libs.x86\_64

下载yum源

wget http://repo.mysql.com/mysql-community-release-el6-5.noarch.rpm

安装yum源

rpm -ivh mysql-community-release-el6-5.noarch.rpm

查看yum源中mysql有效版本

yum repolist all | grep mysql

使用下载yum源安装

yum install mysql-community-server -y

查看mysqld服务是否开机启动

chkconfig --list mysqld centos6

systemctl list-unit-files 查看开机启动列表(centos7)

让mysqld开机启动

chkconfig mysqld on centos6 不起作用

systemtcl enable mysqld centos7 不起作用

利用下面方法解决：

vi /etc/rc.d/rc.local

加入：/etc/rc.d/init.d/mysqld start

chmod +x /etc/rc.d/rc.local

查看是否启动

service mysqld status

启动mysql服务

service mysqld start

如果启动不了服务，删除/var/lib/mysql/，再启动服务

rm -rf /var/lib/mysql/\*

**2.卸载**

**yum remove pam-devel**

**3.升级**

**升级指定软件包**

**yum update kernel #升级内核，小心。**

**全系统升级**

**yum update**

**3.6.7 的软件组功能**

**1.什么是软件组？**

**软件组就是一组相关的软件，比如gnome、比如gcc等，一个软件组名字会包含很多个软件。**

**2.查询软件组有哪些**

**yum grouplist**

**可用环境组：**

**最小安装**

**计算节点**

**基础结构服务器**

**文件和打印服务器**

**基本Web服务器**

**虚拟化主机**

**带图形用户界面的服务器**

**GNOME桌面**

**KDE Plasma工作区**

**开发创意工作站**

**可用组：**

**兼容性库**

**控制台Internet工具**

**开发工具**

**图形管理工具**

**旧式UNIX兼容性**

**科学支持**

**安全工具**

**智能卡支持**

**系统管理工具**

**系统管理**

**3.查询软件组gcc的信息**

**yum groupinfo "Development Tools"**

**4、本章总结**

**4.1 总结本章知识点**

**4.2 面试题答案**

**4.3 预习下一章重点**