行情web服务部署文档

日期： 2019-05-19

作者： 连少华

版本： v1.0

目录

[1. 安装python 4](#_Toc1747983)

[1.1. 安装python依赖包 4](#_Toc1747984)

[1.2. 安装python 3.7 4](#_Toc1747985)

[1.3. 安装virtualenv 4](#_Toc1747986)

[2. 安装nginx 5](#_Toc1747987)

[3. 部署web服务 5](#_Toc1747988)

[3.1. 安装依赖库 5](#_Toc1747989)

[3.2. 创建虚拟环境 5](#_Toc1747990)

[3.3. 部署 5](#_Toc1747991)

[4. 配置 6](#_Toc1747992)

[4.1. 配置web服务 6](#_Toc1747993)

[4.2. 配置uwsgi 7](#_Toc1747994)

[4.3. 配置nginx 9](#_Toc1747995)

[5. 整体测试 12](#_Toc1747996)

[6. 配置高可用 12](#_Toc1747997)

# 安装python

## 安装python依赖包

1. 执行命令（**如果已经安装了gcc、gcc-c++，则需要把下面命令行的gcc gcc-c++删除，否则可能会出现gcc版本被回退的问题**）：sudo yum install openssl-devel bzip2-devel expat-devel gdbm-devel readline-devel sqlite-devel gcc gcc-c++  openssl-devel libffi-devel  -y

## 安装python 3.7

1. 下载python 3.7，执行命令：

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

1. 安装

解压：tar -xzvf Python-3.7.0.tgz

进入目录：cd Python-3.7.0

配置：./configure --prefix=/usr --enable-optimizations

编译安装：make -j4 && sudo make install

1. 测试

执行命令：python3 --version

## 安装virtualenv

1. 执行命令：sudo pip3 install virtualenv
2. 测试：
   1. 创建env目录：cd ~;mkdir env
   2. 执行：virtualenv --no-site-packages env
   3. 激活：source env/bin/activate #如果能看到目录前带了”(env)”则说明安装成功
   4. 离开：deactivate

到此，python环境安装完成；

# 安装nginx

1. 执行命令：sudo yum install nginx -y
2. 测试：nginx -v

# 部署web服务

## 安装依赖库

1. 执行命令：sudo yum install python-devel mysql-devel gcc  MariaDB-shared -y

## 创建虚拟环境

1. 进入home目录
2. 创建目录：mkdir env
3. 执行：virtualenv --no-site-packages env

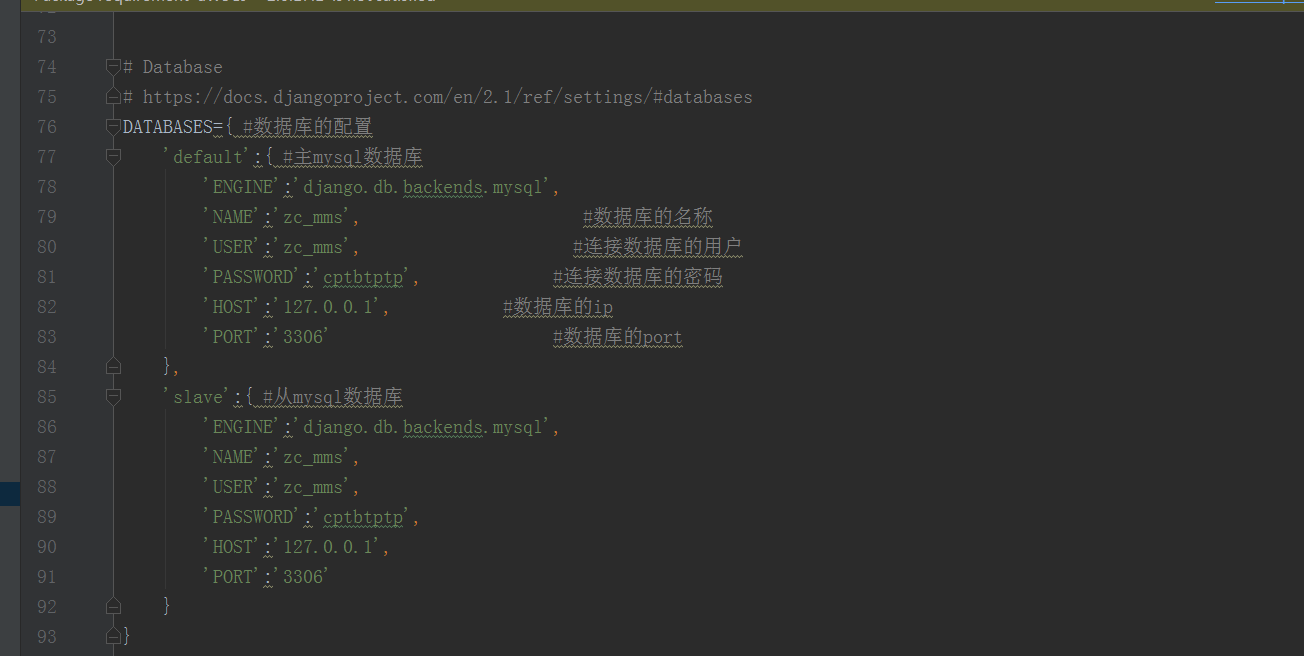
## 部署

1. 把安装包copy到~目录，解压(解压后的目录为webserver)
2. 激活虚拟环境：source env/bin/activate
3. 进入webserver目录，执行命令安装webserver需要的依赖库：pip install -r requirements.txt
4. 测试：
   1. 在虚拟环境中执行：python
   2. 执行：import django
   3. 执行：django.version
   4. 如果参够输出django的版本号：(2,1,4,’final’,0)则依赖库安装正确
   5. 执行：exit() 退出python交互环境

# 配置

## 配置web服务

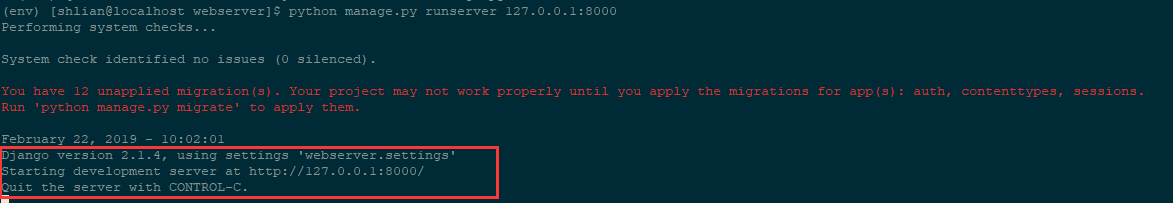
1. 进入webserver目录[后面所有的操作，如果没有特别说明，均以~/webserver目录为当前目录]
2. 打开webserver/settings.py文件，如下图所示：





**注意事项：一定不能按tab键对齐，可使用空格键代替。**

1. 测试配置的正确性：
   1. 在当前目录，激活virtualenv: source ../env/bin/activate
   2. 执行：python manage.py runserver 127.0.0.1:8000，如果运行成功，则会出现如下信息：

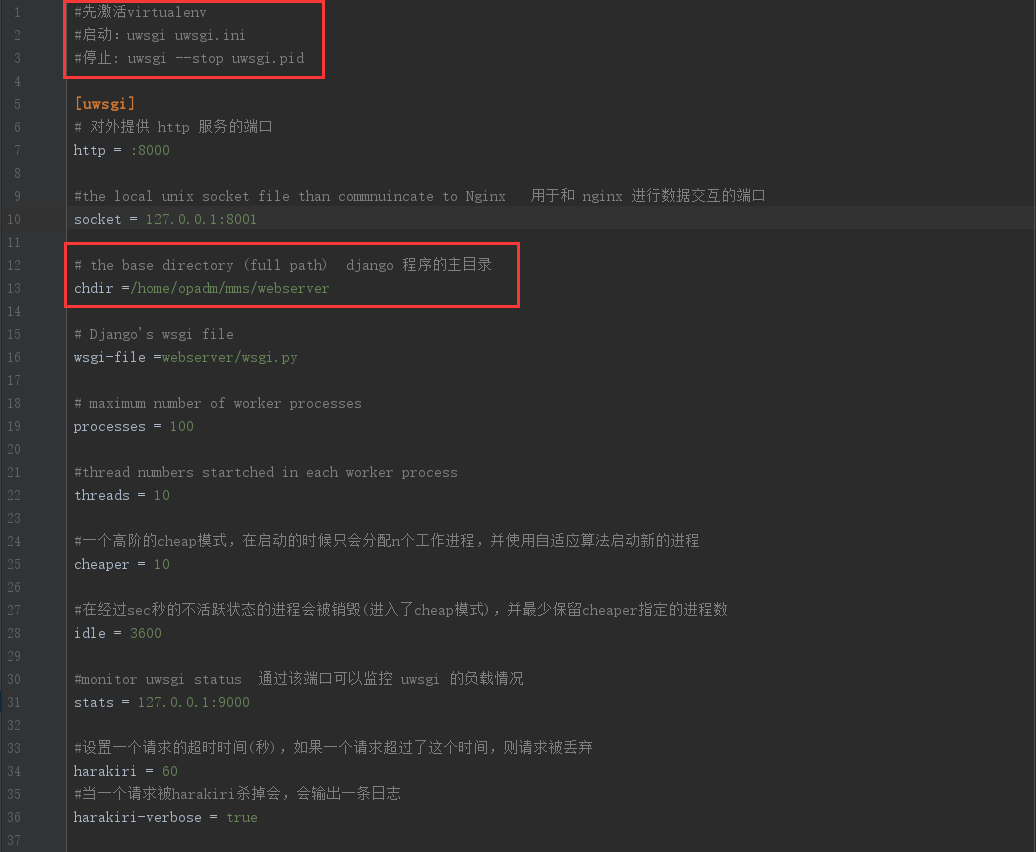


否则会有详细的报错信息。

* 1. 向127.0.0.1:8000发送命令：curl <http://127.0.0.1:8000/help>，如果没有报错，则证明webserver运行正常
  2. 按Ctl+C结束webserver
  3. 输入deactivate退出虚拟环境

## 配置uwsgi

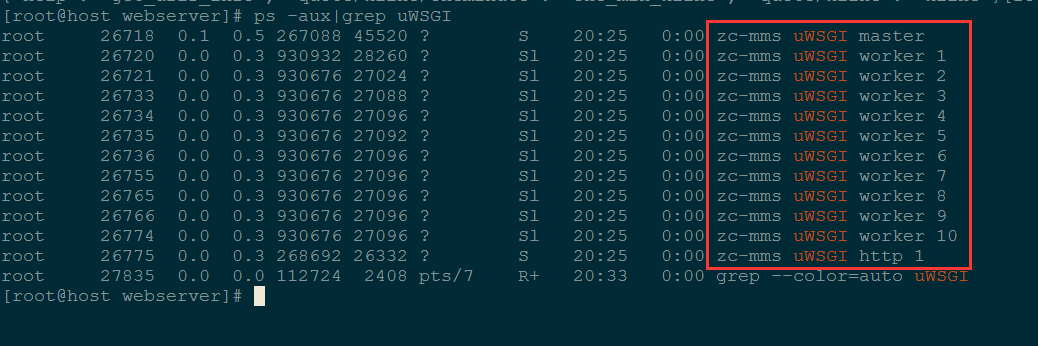
1. 进入webserver目录：cd ~/webserver
2. 打开uwsgi.ini文件，如下图所示：



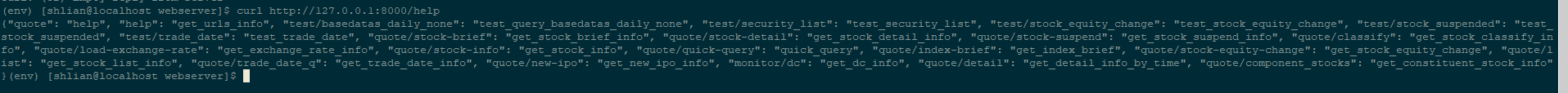


一般情况，建议只修改标红的地方即可，其他优化项根据情况配置。

1. 测试配置的正确性：
   1. 激活virtualenv:source ../env/bin/activate
   2. 执行命令：uwsgi uwsgi.ini



* 1. 执行命令：curl <http://127.0.0.1:8000/help>

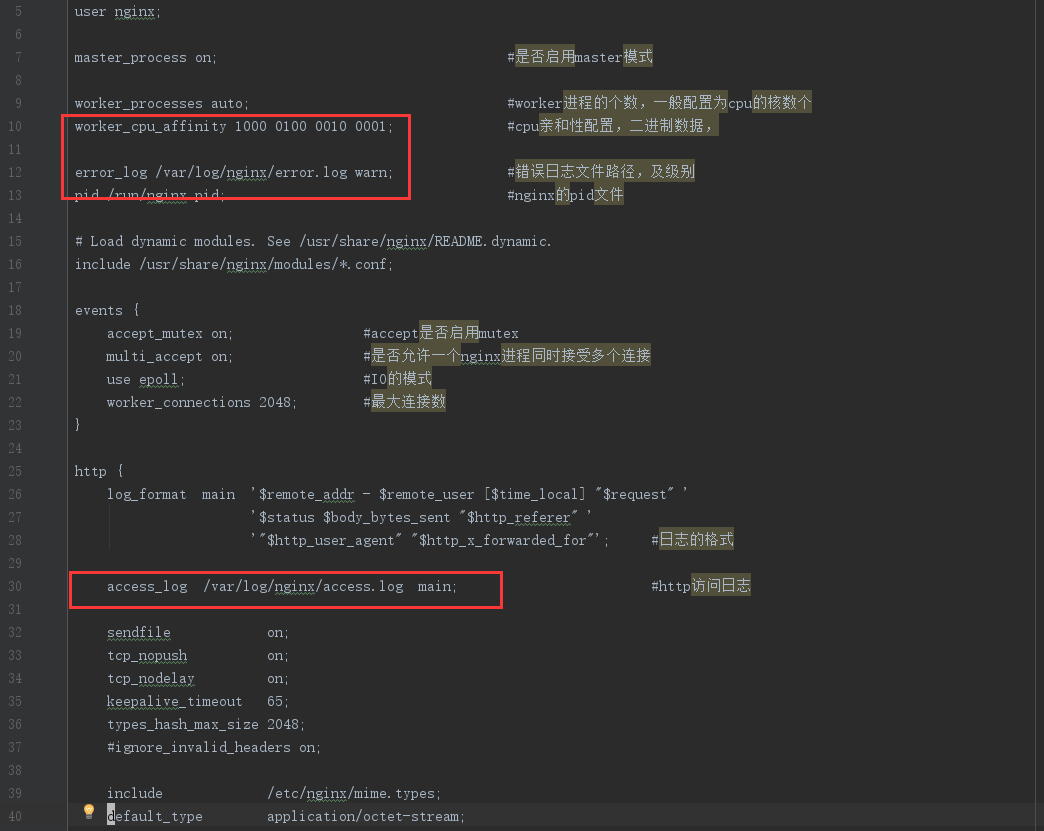


没有报错并且有返回，则执行成功。

* 1. 执行命令：uwsgi --stop uwsgi.pid停止服务
  2. 执行命令：deactivate退出虚拟环境

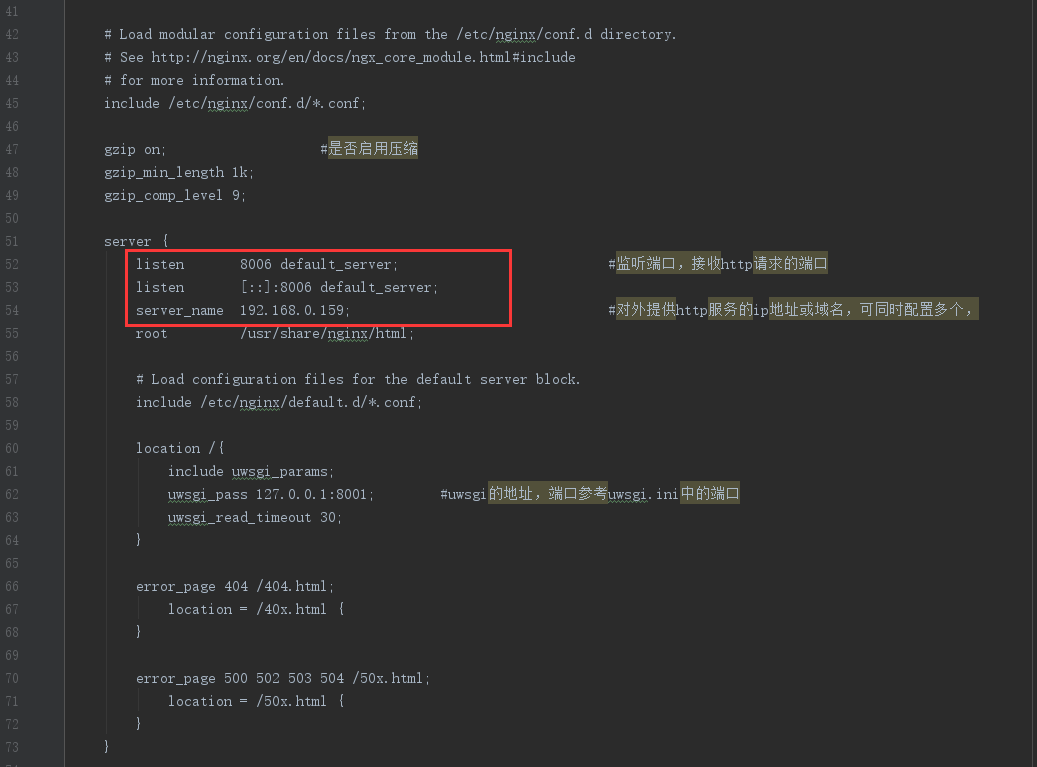
## 配置nginx

1. 拷贝nginx配置文件：sudo cp ~/webserver/config/nginx.conf /etc/nginx/
2. 进入/etc/nginx目录，打开nginx.conf配置文件：



说明：

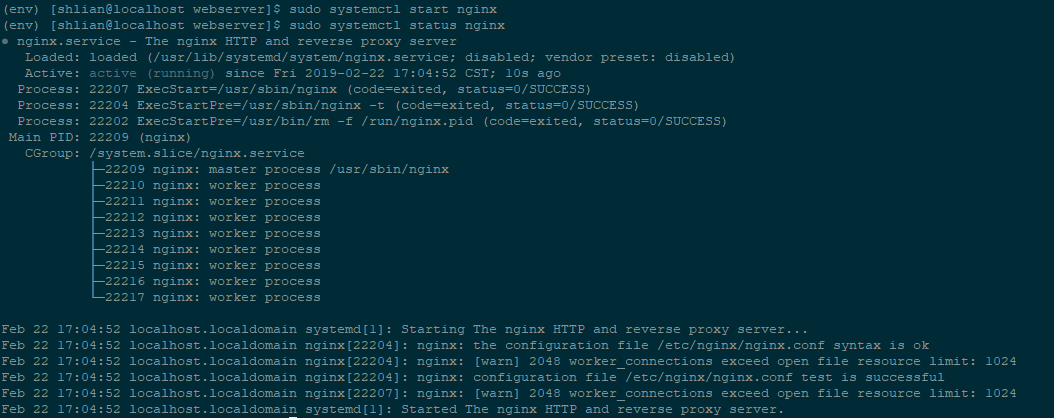
* 更改日志配置：error\_log和assess\_log
* 更改cpu的亲和性，如果是四核，则配置不变，如果是八核，则配置0~7的二进制数字（见上图10行）



说明：

* 由于行情提供的web服务默认是8006端口，所以只更改server\_name为公网的ip或域名即可

1. 测试配置的正确性：
   1. 启动nginx：sudo systemctl start nginx
   2. 查看状态：sudo systemctl status nginx，如下图：

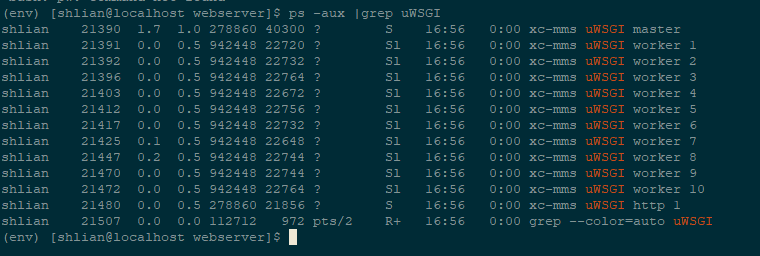


* 1. 停止nginx:sudo systemctl stop nginx

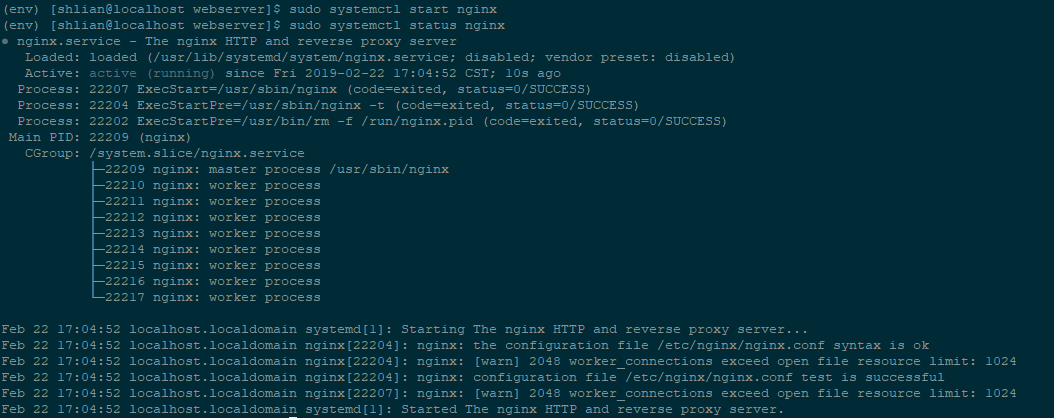
如果启动失败，则可以使用sudo journalctl -xe命令查看详细的错误日志，此处常见的两种错误：要么是nginx.conf配置错误，要么是开启了selinux所致。

# 整体测试

1. 启动uwsgi
   1. 进入webserver目录:cd ~/webserver
   2. 激活虚拟环境：source ../env/activate
   3. 启动uwsgi:uwsgi uwsgi.ini
   4. 检查一下uwsgi的运行状态：



1. 启动nginx
   1. 执行命令：sudo systemctl start nginx
   2. 查看状态：sudo systemctl status nginx，如下图：



1. 在公网，验证接口：
   1. <http://ip:8006/help>
   2. 其他接口请测试逐个验证

# 配置高可用

LVS的配置，略。