**uwsgi entered RUNNING state, process has stayed up for > than 1 seconds (startsecs)报错**

**负载过大服务程序启动报错**

**解决：**

**修改somaxconn**

**内核参数默认值一般是128，对于负载大的服务程序不够，可修改大点**

**两种解决办法**

**1 echo 2048 > /proc/sys/net/core/somaxconn ,临时生效，重启后失效**

**2 /etc/sysctl.conf 中添加**

**net.core.somaxconn=2048**

**终端执行sysctl-p**

**uwsgi出现invalid request block size: 21573 (max 4096)...skip**

**uwsgi内部解析的数据包大小，默认4k。**

**如果准备接收大请求，你可以增长到64k。**

**因为请求大小为：21573，大于默认的4k，所有被丢弃**

**可以增大 buffer-size = 65536**

**[uwsgi]**

**socket = 127.0.0.1:9000**

**chdir = /project/web/test**

**wsgi-file = test/wsgi.py**

**master = true**

**processes = 1**

**buffer-size = 65536**

**chmod-socket = 664**

**vacuum = true**

**Redis编译安装**

Redis官网下载合适版本的压缩包上传服务器

解压缩

tar -xvzf redis\*\*\*

mv redis \*\*\* /usr/local/redis

Cd 目录make

设置密码

修改redis配置文件里面的requirepass，本来是注释的，将注释去掉，并将后面对应的字段设置成自己想要的密码，保存退出

配置远程访问

注释该行bind 127.0.0.1 -::1 或改为bind 0.0.0.0

配置后台运行

daemonize no改为yes

配置redis全局启动

vim ~/.bash\_profile

添加命令

source ~/.bash\_profile生效更改

**创建虚拟环境**

1.安装anaconda包(conda可以直接创建不同python版本的虚拟环境。而virtualenv只是指定创建不同python版本的虚拟环境，前提是已经安装了不同版本的python,与conda相比没有conda灵活。)

<https://mirrors.bfsu.edu.cn/anaconda/archive/>

Wget版本链接

Chmod修改权限后执行sh文件

添加环境变量，source ~/.bashrc

检查安装是否成功

conda -V

2.创建项目所需包版本的虚拟环境

conda create -n name python=x.x

Conda activate name

conda env remove -p 要删除的虚拟环境路径

conda env remove -n 需要删除的环境名 --all

conda info -e 查看虚拟环境

conda config --set auto\_activate\_base false

conda config --add channels <https://mirrors.tuna.tsinghua.edu.cn/anaconda/cloud/msys2/>

conda config --add channels <https://mirrors.tuna.tsinghua.edu.cn/anaconda/cloud/conda-forge/>

conda config --add channels <https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/free/>

conda config --set show\_channel\_urls yes

python安装

安装依赖yum install wget zlib-devel bzip2-devel openssl-devel ncurses-devel sqlite-devel readline-devel tk-devel gcc make zlib zlib-devel libffi-devel -y

编译安装路径

./configure --prefix=/usr/local/python3.11

make&&make install

**django.db.utils.OperationalError: (1040, 'Too many connections')**

**进数据库**

**django.db.utils.OperationalError: (1040, 'Too many connections')**

**SET GLOBAL max\_connections = <new\_value>;**

sudo **firewall**-cmd --zone=public --permanent --add-port=9000/tcp

#允许ip192.168.0.1访问9001端口

sudo firewall-cmd --permanent --add-rich-rule='rule family="ipv4" source address="10.248.72.0/24" port protocol="tcp" port="9000" accept'

#重新载入一下防火墙设置，使设置生效

sudo firewall-cmd --reload

#查看已设置规则

sudo firewall-cmd --zone=public --list-rich-rules

### 动态修改

ps aux | grep mongo 拿到PID数

cat /proc/{mongo进程数}/limits

系统默认是(soft)1024:(hard)4096

prlimit --pid 进程数 --nofile=65535:65535

如果已超过65535这个数，可考虑提高，或考虑分片集群

可使用 lsof -p 进程数 | wc -l 查看目前mongodb内的数据量

### 永久修改

sudo vi /etc/security/limits.conf

在末尾加上

mongodb soft nofile 65535

mongodb hard nofile 65535

保存后，再重启虚拟机``

**应用领域**

**请列举本技术适用的潜在或已知应用的产业、产品与技术领域及其相应之应用方式**

本技术适用于电子制造领域的新产品导入验证实验室，將实验室管理体系、流程与专案验证需求结合，建设实验室信息化管理平台，实现数据信息统一管理，设备状态跟踪预警，产品动态信息跟踪，测试数据收集/分析/處理，物料管理等功能， 通过二维报表加三维场景进行可视化展示

**技术背景**

**本技术之发明目的及(或)所欲解决之问题；若本技术存在相应的最相近现有技术、习知技艺，请说明该现有技术、习知技艺，并说明现有技术之技术问题及其缺陷。**

**1.现有技术描述**

通过物联网、API、扫描枪、手动录入等方式，获取终端设备、SFC系统、机台、文件等数据，利用Python、Django ORM、HTTP、H5、Javascript、CSS完成前后端开发与数据分析，通过图表可视化、数据查询、报表生成等方式支持实验室系统管理业务展开，如专案管理、产品测试管理、设备管理、机台管理、安全管理、人员管理、岗位职能培训、人员培育

**2.现有技术问题及其缺陷描述**

现有技术暂时由于资源限制和开发限制，无法第一时间获取数据，无法全面获取到产品流转中的信息

少量数据未能实现实时采集

**3.技术发明目的、所欲解决之问题、能提升的功效**

通过统一建立实验室信息化管理平台，实现数据信息统一管理，以及设备状态、产品动态信息跟踪，并且对测试数据进行收集/分析/处理，解决以下问题：

1. 数据分散，无统一指标体系管理

2. 信息传递不及时，目前较多信息、工单等仍依靠手写记录，无法及时进行文件信息更新同步，协同工作效率低

3. 实验室分散在各个不同区域，无法精准快速得到产品位置信息、状态信息

4.设备机台维护仍需人员进行状态跟踪

5.物料需人工管理，对耗材等信息未进行统一管理

**技术关键点**

**请说明本技术欲保护的技术关键点；若本技术存在现有技术，则需说明与现有技术的关键区别点(如制程简化、组件简化、材料配方替换等)**

**1.与现有技术的关键技术特征区别**

实时记录系统当前的信息，判断异常情况，针对异常情况及时提醒

1. 將实验室管理体系、流程与专案验证需求结合，建设实验室信息化管理平台，实现数据信息统一管理

**2.本技术欲保护的技术关键点**

1. 系统的功能框架、业务框架与页面设计

2. 数据的应用、分析方式

**技术内容说明**

**请说明本技术的技术内容(如结构、制程、方法、配方)等技术特点；以及具体实施例说明（如分析数据、附图和文字等说明）。**

**1.1.请结合附图或流程详细描述本技术的工作方式**

1)技术特征（如应用产品或领域之配方或化合物、制程或方法、系统或装置）

將实验室管理体系、流程与专案验证需求结合，应用Python、Django ORM、HTTP、H5、Javascript、CSS、Backend完成前后端开发与数据分析，建立实验室信息化管理平台

(2)技术流程图/产品图/方块图标中，每一图框或相邻图框间连接关系的技术原理、定义及功效

业务流程包含4个部分，基础数据作为业务的基础支撑，包含了专案管理、设备管理、文件管理、位置管理、权限管理部分；测试管理包含了测试矩阵、测试计划、样品管控、故障模式、人力保障、设备运维六部分，为实验室测试作业管理提供职能支持；实验室测试作业管理包含机台验证所需进行整个管控流程，测试管控中心进行监控、调度、干预等远程管理

# #######################################################################################

## 构建基础镜像 dqvp-base

## 生成一个有python 和nginx环境的镜像 dqvp-base:latest

## 生成镜像指令 docker build -t dqvp-base:latest .

# 设置代理

# oracle数据库

# 安装Python运行环境

# #######################################################################################

#FROM nginx

#COPY sources.list /etc/apt/sources.list

#COPY dqvp/requirements.txt /root/requirements.txt

#RUN export http\_proxy="http://F7688609:Ai&Data852.@10.191.131.21:3128" \

#&& export https\_proxy="http://F7688609:Ai&Data852.@10.191.131.21:3128" \

#&& apt-get update \

#&& apt-get install -y --no-install-recommends apt-utils \

#&& apt-get install -y python3 \

#&& apt-get install -y python3-pip \

#&& apt-get install -y libaio1 \

#&& apt-get install -y gettext \

#&& pip install -r /root/requirements.txt \

#&& pip install uwsgi \

#&& pip install pymysql \

#&& pip install supervisor

# #######################################################################################

## 构建项目镜像 docker build -t dqvp:<version> .

# 配置文件

# 项目文件

# #######################################################################################

FROM kb-base

# 前端配置

COPY dqvp/web\_dist /usr/share/nginx/html/dqvp-web-dist

# COPY web.conf /etc/nginx/conf.d/web.conf

# 后端配置

COPY instantclient\_21\_8/ /opt/oracle/instantclient\_21\_8

COPY dqvp /root/dqvp

COPY .env /root/dqvp/.env

COPY uwsgi.ini /root/dqvp/uwsgi.ini

# COPY server.conf /etc/nginx/conf.d/server.conf

COPY default.conf /etc/nginx/conf.d/default.conf

COPY start.sh /start.sh

COPY supervisord.conf /etc/supervisor/supervisord.conf

COPY docker-entrypoint.sh /docker-entrypoint.sh

RUN chmod 777 /docker-entrypoint.sh \

&& chmod 777 /start.sh \

&& nginx -c /etc/nginx/nginx.conf

EXPOSE 7000