Redash安装说明

# 一、准备环境

## 1、安装python3

sudo apt-get install python3-dev

sudo apt-get install python3-pip

sudo apt-get install python3-venv

## 2、安装PostgreSQL

sudo apt-get install postgresql

### 2.1、修改密码

sudo -u postgres psql（或先su - postgres再执行psql），出现提示postgres=#

postgres=# ALTER USER postgres WITH PASSWORD '123456';

postgres=# \q

### 2.2、修改配置

#### 2.2.1、为了支持远程访问，修改以下2个文件：

* /etc/postgresql/10/main/pg\_hba.conf

在合适的地方加上

host all all 10.15.101.1/24 md5

* /etc/postgresql/10/main/postgresql.conf

找到这一行#listen\_addresses=’localhost

修改成listen\_addresses=’\*'

端口是5432，根据需要修改

#### 2.2.2、修改完重启

sudo /etc/init.d/postgresql restart

### 2.3、其他信息（不需要执行）

#### 2.3.1、启动停止命令行

* 启动

sudo /etc/init.d/postgresql start

* 停止

sudo /etc/init.d/postgresql stop

* 重启

sudo /etc/init.d/postgresql restart

* 重新加载配置

sudo /etc/init.d/portgresql reload

#### 2.3.2、postgresql基本操作

* 进入数据库

sudo -u postgres psql（或先su - postgres再执行psql）

* 列出当前库中的所有表

\d

* 创建新的超级用户，账号root，密码123456

CREATE ROLE root SUPERUSER PASSWORD '123456' LOGIN;

* 退出

\q

#### 2.3.3、安装Windows可视化工具pgAdmin

## 3、安装Redis

sudo apt-get install redis-server

### 检查

redis-cli

info

exit

## 4、安装Node.js

sudo apt-get install nodejs

sudo apt-get install npm

**以下3句是为了安装满足要求的最新版本**

sudo npm config set registry https://registry.npm.taobao.org

sudo npm install n -g

sudo n stable

**重启控制台，确保版本正确！！！！！**

node -v

npm -v

## 5、其他

sudo apt-get install libpq-dev

sudo apt-get install build-essential libssl-dev libffi-dev libsasl2-dev libldap2-dev libmysqlclient-dev unixodbc-dev【安装requirements\_all\_ds.txt时需要】

npm update caniuse-lite browserslist【忽略，npm run build时提示版本旧，执行后版本还是旧】

# 二、安装redash(假设当前在~目录)

## 1、创建python环境

python3 -m venv vredash

## 2、进入python环境

. vredash/bin/activate

## 3、得代码

cd vredash

git clone <https://github.com/CFHH/redash>

cd redash

## 4、安装redash（时间较长）

pip3 install wheel

pip3 install -r requirements.txt

pip3 install -r requirements\_dev.txt

npm install

npm run build

## 5、redash建数据库

### 5.1、新建数据库

./manage.py database create\_tables

./manage.py database setup\_admin

#### 错误处理

如果create\_tables报错，提示没有类似account这样的用户名【这里的account是当前linux的用户名，如果是root不会报错】，则执行

sudo su - postgres -c "createuser -s account"

### 5.2、文件拷贝式恢复数据库

* 先停掉postgresql

sudo /etc/init.d/postgresql stop

* 把别处机器上/var/lib/postgresql/10/main下的所有数据拷贝下来，替换该机器的相应文件夹（原文件夹建议重命名，不要删掉），执行以下命令：

chown -R postgres.postgres main/

* 再启动

sudo /etc/init.d/postgresql start

#### 错误处理

如果发现有以下错误

FATAL: database locale is incompatible with operating system DETAIL:

The database was initialized with LC\_COLLATE "zh\_CN.UTF-8", which is not recognized by setlocale().

HINT: Recreate the database with another locale or install the missing locale.

可能需要执行类似下面这个命令：

sudo locale-gen zh\_CN.UTF-8

# 三、启动与停止

## 1、简单脚本

. vredash/bin/activate

cd vredash/redash/

./stop.sh

./xstart.sh

在~/vredash/redash目录下有start.sh、stop.sh、xstart.sh三个文件

start.sh只启动一个worker进程

xstart.sh启动20个worker进程

stop.sh关闭所有redash相关进程

## 2、命令行

* 启动http进程（只能一个）

./manage.py runserver --host 0.0.0.0 & 【--port 5000】

* 启动调度进程（只能一个）

./manage.py rq scheduler &

* 启动万能worker进程（可以多个）

./manage.py rq worker &

* 启动指定功能的worker进程（可以多个）

./manage.py rq worker scheduled\_queries periodic emails default schemas &

./manage.py rq worker queries &

各种页面的数据请求最终都在queries队列里

## 3、其他

* 列举进程

ps aux | grep manage.py

* 杀死进程

kill –s 9 PID

kill -9 $(ps -ef|grep manage.py|grep -v grep|awk '{print $2}')

# 四、Redis缓存查看

7、redis的一些操作

查看job数量 zcount rq:scheduler:scheduled\_jobs 0 999999999999999

worker进程 smembers rq:workers

keys \*rq:worker:\*

queue队列 keys \*rq:workers:\*

worker当前job hget rq:worker:cb8068f4dccc4024a79e83db00be76d5 current\_job

7、缓存信息

（一）SCHEDULE

keys \*schedule\*

查看schedule相关的缓存信息

（二）WORKER

keys \*worker\*

查看worker相关的缓存信息，rq:worker:开头的都是具体的一个worker

smembers rq:workers

查看所有的worker

smembers rq:workers:queries

查看为名为queries的QUEUE工作的worker

hgetall rq:worker:cb8068f4dccc4024a79e83db00be76d5

hget rq:worker:cb8068f4dccc4024a79e83db00be76d5 current\_job

查看某个具体worker的current\_job, state, successful\_job\_count, failed\_job\_count, total\_working\_time, queues, pid, birth, death, last\_heartbeat, hostname，举例如下

"birth" "2020-08-21T07:27:25.528026Z"

"last\_heartbeat" "2020-08-25T06:16:32.496566Z"

"queues" "scheduled\_queries,queries,periodic,emails,default,schemas"

"pid" "20770"

"hostname" "ubuntu"

"state" "idle"

"successful\_job\_count" "21872"

"total\_working\_time" "22004.33879800000005744"

（三）QUEUE

smembers rq:queues

列举当前的QUEUE

rq:queues:queries类似这样的key的内容是job\_id的List，缓存时间180秒

（四）JOB

keys \*rq:job:\*

列举所有job

（五）其他

keys \*registr\*

1) "rq:clean\_registries:default"

2) "rq:clean\_registries:schemas"

3) "rq:clean\_registries:periodic"

4) "rq:clean\_registries:scheduled\_queries"

5) "rq:clean\_registries:emails"

6) "rq:clean\_registries:queries"

hgetall rq:job:8469d2ca-2296-4475-a120-0b9ca021c068

hget rq:job:8469d2ca-2296-4475-a120-0b9ca021c068 status

查看某个具体worker的信息，举例如下

"enqueued\_at" "2020-08-25T05:48:43.407277Z"

"started\_at" "2020-08-25T05:48:43.422264Z"

"ended\_at" "2020-08-25T05:48:44.289082Z"

"created\_at" "2020-08-25T05:48:43.407119Z"

"description" "redash.tasks.queries.execution.execute\_query('X{\\r\\n \"tables\": [\\r\\n\\r\\n ],\\r\\n \"sub\_queries\":[\\r\\n {\\r\\n \"name\": \"time\_1\",\\r\\n \"query\": \"SELECT dates,COUNT(ANALYSIS\_ORDER\_NO) AS NUM FROM(SELECT ANALYSIS\_ORDER\_NO,DATE\_TRUNC(\\'hour\\',\_\_time) AS dates FROM medical\_treatment\_hospital\_analysis WHERE \_\_time>=\\'2018-07-12T00:00:00\\' AND \_\_time<\\'2018-07-13T00:00:00\\' GROUP BY ANALYSIS\_ORDER\_NO,DATE\_TRUNC(\\'hour\\',\_\_time) ORDER BY dates) GROUP BY dates\"\\r\\n },\\r\\n {\\r\\n \"name\": \"time\_2\",\\r\\n \"query\": \"SELECT dates,COUNT(ANALYSIS\_ORDER\_NO) AS NUM FROM(SELECT ANALYSIS\_ORDER\_NO,DATE\_TRUNC(\\'hour\\',\_\_time) AS dates FROM medical\_treatment\_hospital\_analysis WHERE \_\_time>=\\'2018-07-11T00:00:00\\' AND \_\_time<\\'2018-07-12T00:00:00\\' GROUP BY ANALYSIS\_ORDER\_NO,DATE\_TRUNC(\\'hour\\',\_\_time) ORDER BY dates) GROUP BY dates\"\\r\\n },\\r\\n {\\r\\n \"name\": \"time\_3\",\\r\\n \"query\": \"SELECT dates,COUNT(ANALYSIS\_ORDER\_NO) AS NUM FROM(SELECT ANALYSIS\_ORDER\_NO,DATE\_TRUNC(\\'hour\\',\_\_time) AS dates FROM medical\_treatment\_hospital\_analysis WHERE \_\_time>=\\'2018-07-10T00:00:00\\' AND \_\_time<\\'2018-07-11T00:00:00\\' GROUP BY ANALYSIS\_ORDER\_NO,DATE\_TRUNC(\\'hour\\',\_\_time) ORDER BY dates) GROUP BY dates\"\\r\\n },\\r\\n {\\r\\n \"name\": \"time\_4\",\\r\\n \"query\": \"SELECT dates,COUNT(ANALYSIS\_ORDER\_NO) AS NUM FROM(SELECT ANALYSIS\_ORDER\_NO,DATE\_TRUNC(\\'hour\\',\_\_time) AS dates FROM medical\_treatment\_hospital\_analysis WHERE \_\_time>=\\'2018-07-09T00:00:00\\' AND \_\_time<\\'2018-07-10T00:00:00\\' GROUP BY ANALYSIS\_ORDER\_NO,DATE\_TRUNC(\\'hour\\',\_\_time) ORDER BY dates) GROUP BY dates\"\\r\\n },\\r\\n {\\r\\n \"name\": \"time\_5\",\\r\\n \"query\": \"SELECT dates,COUNT(ANALYSIS\_ORDER\_NO) AS NUM FROM(SELECT ANALYSIS\_ORDER\_NO,DATE\_TRUNC(\\'hour\\',\_\_time) AS dates FROM medical\_treatment\_hospital\_analysis WHERE \_\_time>=\\'2018-07-08T00:00:00\\' AND \_\_time<\\'2018-07-09T00:00:00\\' GROUP BY ANALYSIS\_ORDER\_NO,DATE\_TRUNC(\\'hour\\',\_\_time) ORDER BY dates) GROUP BY dates\"\\r\\n },\\r\\n {\\r\\n \"name\": \"time\_6\",\\r\\n \"query\": \"SELECT dates,COUNT(ANALYSIS\_ORDER\_NO) AS NUM FROM(SELECT ANALYSIS\_ORDER\_NO,DATE\_TRUNC(\\'hour\\',\_\_time) AS dates FROM medical\_treatment\_hospital\_analysis WHERE \_\_time>=\\'2018-07-07T00:00:00\\' AND \_\_time<\\'2018-07-08T00:00:00\\' GROUP BY ANALYSIS\_ORDER\_NO,DATE\_TRUNC(\\'hour\\',\_\_time) ORDER BY dates) GROUP BY dates\"\\r\\n },\\r\\n {\\r\\n \"name\": \"time\_7\",\\r\\n \"query\": \"SELECT dates,COUNT(ANALYSIS\_ORDER\_NO) AS NUM FROM(SELECT ANALYSIS\_ORDER\_NO,DATE\_TRUNC(\\'hour\\',\_\_time) AS dates FROM medical\_treatment\_hospital\_analysis WHERE \_\_time>=\\'2018-07-06T00:00:00\\' AND \_\_time<\\'2018-07-07T00:00:00\\' GROUP BY ANALYSIS\_ORDER\_NO,DATE\_TRUNC(\\'hour\\',\_\_time) ORDER BY dates) GROUP BY dates\"\\r\\n }\\r\\n ]\\r\\n}', 1, {'Username': 'admin@playblock.com', 'Query ID': '252', 'Queue': 'queries', 'Enqueue Time': 1598334523.4070466}, is\_api\_key=False, scheduled\_query\_id=None, user\_id=1)"

"meta" "\x80\x04\x95M\x00\x00\x00\x00\x00\x00\x00}\x94(\x8c\x0edata\_source\_id\x94K\x01\x8c\x06org\_id\x94K\x01\x8c\tscheduled\x94\x89\x8c\bquery\_id\x94\x8c\x03252\x94\x8c\auser\_id\x94K\x01u."

"result" "\x80\x04\x95\x04\x00\x00\x00\x00\x00\x00\x00M\x1a\x87."

"timeout" "-1"

"data" "x\x9c\xdd\x96Ok\xd4@\x18\xc6WTZ\xa2A\xd0/0\xe4\xb2YHC\x92\xda\xee\xbaX1\xee\xc6Zl\x13\xcd&\xd8Rd\x98M\x066l\xfel3\x13p\x91\x05o\"\x04O\xe3\a\xf3$\xf8i\xcc?h\xc5C\xf0\xa2K\x86\x1c\x92\xe7}\x9f\xf7y\xf9\xe52\x9f\xee|{p\xafW\x1d1\x97R\xec#\xb2\x90)\"K\"\_e8\r0\x91\xf1\a\xece4H\xe2\xe6\r\xc3\xb2\xb2f\xe6\xf9\x11\xd7\xeb\x9d\x7f\xe49P\x1c\x81\xa2y\x88\x890\x06\x97<WK\xef\xa5\xa6D\xb29l\xa6\t\xe3\xcbZllU=F\x11.\x8c\x02\r\"\x0cUA\xbaQ\xaa\xa2\xca\xda\xcc85&\x0e\xf0\x11\xc5D\x9aX\xae\xe9\x88\xba\xa9\x9f^\xccNf\xd0\xb2\xa7\x86\rMk\x00\xf4\x190\xdd3\xf0\xd2\xb6\xce\xc4\xc6\xf1G\x974\xd5\x1d\x03:\xb6kN\xc4\xfe\"\xc9\xd2\xbe\x04a\x19]\xd9\xab\x80j\x00\x88\xb0\x1fx(\x844\xc5\x88F8\xa6p\x91\x90U@\x0b\t\xc5(\\\x93\x80\x80w\xaf\x0c\xdb\x00\xb5\xff\xd9Q\_S\xd4\xd1\x9e2\xdcS5GQ\xc6\xd5\xd3/6\x986\x1dO\xaf\x1b\xf6o4\x1c\xdb\x96\xfb\x06\xbc\xb8\xf8\xab]\xab\x96\xd2Tm<\xb8\x1eR}\x0b5\xc3\x8d\xd4B[\xeb\x04m\xb5\x8d\xb6\xb6\x1d\xb4\xf7;A[i\xa3\xadn\a\xed\xc7]\xa0\xad<i\xa3\xadl\a\xed\x83N\xd0\x1e\xb5\xd0\xfe\xedw\xfcG\xda\x87\x9d\xa0=l\xa3=\xda\x0e\xda\xc3N\xd0>l\xa3=\xfc7\xb4\x9b\xdb!\xcfm\xd8\xeb[\x1b&\xe6\xbb.\xc1i\t\x9c\xe5\x8f\x90\x1f\x05\xf1\xf3U\x88\xd6\xf30\xf1\x96\xb2\x97D,\xdf}[\"\a'S\x96\xdf\xd6\x0e4\x96\xdf-\x84\xach\xdfi\xae\x95,\xbfo\xc4W\xa5\x06\x9cb\x13v\xac\xff\xf8>\xf8\xfa\x93\xe7\xb3\xcf\xacL\xd8\xc9\x8a\x04\x18\xf8E`\xfe\x90x\x0b\xecg!\xf6\xeb\x8bl)\x9b9\x17\x10\x88V\x01\\\xe25\xfb\x92Q&\xff\x02=\xa01m"

"status" "finished"

"origin" "queries"

"result\_ttl" "1800"