Code Project Directory

|--- spiders : a distribution spider system controlled by Celery

| |---crawl\_worker : a scrapy project to get ccass data

| |---dockerfile : dockerfile to deploy docker

| |---queue : save queue data from Celery and Scrapy will read the data to crawl

| |---tasks : celery tasks and settings

| |---tools : some tools to deal with data

| ---crawl\_manager.py : celery task distributor

Docker Deploy

1. Manager:

Enter spiders/dockerfile/crawl\_manager folder

$ docker build -t crawl\_manager:0.1 .

$ docker run --name crawl\_manager -it -v ~/work:/home/work crawl\_manager:0.1 bash

2. Worker:

Enter spiders/dockerfile/crawl\_work

$ docker build -t crawl\_worker:0.1 .

$ docker run --name ccass\_1 -it -v ~/work:/home/work crawl\_worker:0.1

$ docker run --rm -it -v ~/work:/home/work crawl\_worker:0.1 # delete container after exit

$ docker run --rm -d -v ~/work:/home/work crawl\_worker:0.1 # run in background delete container after exit

3. load\_celery.sh

cd /home/work/spiders

celery --loglevel=info -A tasks.scrapy\_task worker --max-tasks-per-child 1 –n workername@%h

4. Distribute tasks

$ docker exec -it crawl\_manager bash

# cd /home/work/spiders

# python crawl\_manager.py -i "00006" -d 60

5. Load code

git clone https://github.com/easy00000000/spiders.git

Environment

1. MySQL – Use local docker mysql

$ docker run --name mysqllib -e MYSQL\_ROOT\_PASSWORD=toor -d mysql:5.6

$ docker run --name ccass\_1 -it -v ~/work:/home/work --link mysqllib:mysql crawl\_worker:0.1

2. MySQL - Use online mysql

Same as Docker Deploy

# check the database with - python check\_rdsmysql.py

Monitor

docker <http://172.17.0.1:9000/#/dashboard>

mq <http://67.209.179.247:15672/#/>

Monitor Celery at crawl\_manager

$ celery flower -A tasks.scrapy\_task --broker=amqp://guest:guest@67.209.179.247:5672// --port=5555

Monitor Celery at flower docker

$ git clone <https://github.com/mher/flower.git>

enter flower folder

$ docker build -t flower:1.0 .

$ docker run -d -p=49555:5555 --name flower flower:1.0 flower --broker=amqp://guest:guest@67.209.179.247:5672// --port=5555

Monitor @ 67.209.179.247:49555

RabbitMQ

docker run -d --hostname rabbit\_manager --name rabbit\_manager -p 15672:15672 -p 5672:5672 rabbitmq:3-management

Open at 0.0.0.0:15672

user: guest

passwd: guest

Schedule

python schedule\_crawl.py >> log.txt 2>&1

UTC+8=Shanghai

CMD [“bash”, “load\_schedule.sh”]

FRP

Server: ./frps -c ./frps.ini

frps.ini

[common]

bind\_port = 7000

[gw\_1]

listen\_token = 6001

auth\_token = 6001

[pb\_1]

listen\_token = 6002

auth\_token = 6002

[hm\_1]

listen\_token = 6003

auth\_token = 6003

Client gw\_1: ./frpc -c ./frpc.ini

frpc.ini

[common]

server\_addr = 67.209.179.247

server\_port = 7000

auth\_token = 6001

[gw\_1]

type = tcp

local\_ip = 10.0.2.15

local\_port = 22

remote\_port = 6001

PuTTY 67.209.179.247:6001

Client pb\_1: ./frpc -c ./frpc.ini

frpc.ini

[common]

server\_addr = 67.209.179.247

server\_port = 7000

auth\_token = 6002

[pb\_1]

type = tcp

local\_ip = 10.0.2.15

local\_port = 22

remote\_port = 6002

PuTTY 67.209.179.247:6002

Read existing data

Save data

Receive tasks

And working

Distribute tasks

Docker: crawl\_worker:0.1

Broker: rabbit\_mq

Worker n

Worker 3

Worker 2

Worker 1

Celery Manager

Docker: crawl\_manager:0.1