|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**MarketCloud**

**GFlux中石化部署手册**

**目 录**

[1 油站数据获取 1](#_Toc421031887)

[1.1 手动拷贝原始数据库文件 1](#_Toc421031888)

[1.2 站级数据库增量上传 2](#_Toc421031889)

[1.2.1 运行环境说明 2](#_Toc421031890)

[1.2.2 工具文件创建 2](#_Toc421031891)

[1.2.3 执行过程 4](#_Toc421031892)

[1.3 省公司中心主动查询获取 4](#_Toc421031893)

[1.3.1 油品字段说明 4](#_Toc421031894)

[1.3.2 非油品字段说明 6](#_Toc421031895)

[1.3.3 上传的目录与文件名格式标准 6](#_Toc421031896)

[2 数据接收(FTP) 7](#_Toc421031897)

[2.1 接收原始数据库文件 7](#_Toc421031898)

[2.1.1 运行环境和FTP配置 7](#_Toc421031899)

[2.1.2 处理过程 7](#_Toc421031900)

[2.2 接收站级增量上传数据（txt文件） 9](#_Toc421031901)

[2.2.1 运行环境和FTP配置 9](#_Toc421031902)

[2.2.2 处理过程 10](#_Toc421031903)

[2.2.3 执行过程 11](#_Toc421031904)

[2.3 接收省中心指定格式上传数据 12](#_Toc421031905)

[3 Gfulx部署 13](#_Toc421031906)

[3.1 运行环境 13](#_Toc421031907)

[3.2 服务器程序打包 13](#_Toc421031908)

[3.3 服务器程序安装 14](#_Toc421031909)

[3.4 服务器配置与启动 14](#_Toc421031910)

[3.4.1 配置nginx 14](#_Toc421031911)

[3.4.2 配置PostgreSQL数据库 15](#_Toc421031912)

[3.4.3 配置Django 15](#_Toc421031913)

[3.4.4 启动服务器 16](#_Toc421031914)

[3.4.5 添加计划任务 16](#_Toc421031915)

[3.4.6 数据初始化 16](#_Toc421031916)

[4 Gflux数据导入 17](#_Toc421031917)

[4.1 手动数据导入 17](#_Toc421031918)

[4.2 自动增量数据导入 17](#_Toc421031919)

[4.3 自动从省中心服务器导入 17](#_Toc421031920)

[5 系统扩容 18](#_Toc421031921)

[5.1 数据库服务器扩容 18](#_Toc421031922)

[5.2 django服务器扩容 18](#_Toc421031923)

1 油站数据获取

根据与客户的讨论来确定油站数据获取方法，油站数据可以采取三种模式上传：手动拷贝原始数据库文件，增量上传和省公司获取模式。

1.1 手动拷贝原始数据库文件

用户直接手动拷贝站级服务器上存放数据库文件的目录下（一般是/smc20/database/）的gkdb.db和gkdb.log文件，将其使用FTP工具上传至[ftp.marketcloud.com.cn](ftp://ftp.marketcloud.com.cn/)上的指定用户和目录。

假定用户上传数据的目录为/home/cnpc\_jl/，需要创建一个新的目录比如puqing, 用来将对应站的数据放在一起。假定上传三个站的数据，站名分别为puqing, zhangsan, wangwu, 上传完成之后目录结构为:

/home/cnpc\_jl/

├─puqing

│ gkdb.db

│ gkdb.log

│

├─wangwu

│ gkdb.db

│ gkdb.log

│

└─zhansan

gkdb.db

gkdb.log

1.2 站级数据库增量上传

1.2.1 运行环境说明

中石化服务器环境为TurboLinux,所使用的数据库为Sybase SQL Anywhere 8.0。在命令行运行python以确保服务器上可以运行python命令。

1.2.2 工具文件创建

在站级服务器上添加如下两个文件：

（File.1）turbolinux\_auto\_upload\_sinopec\_settings.py

其需要手动配置的内容格式说明为：

1）'ftp\_host',ftp地址

2）'ftp\_user',ftp用户名

3）'ftp\_pwd',ftp密码

4）'db\_path',db文件存储路径

5）'dbisql\_bin', sybase路径

6）'save\_dir', 查询结果存储路径

7）'db\_user\_name', db用户名

8）'db\_pwd', db密码

9）'site', 站点名称

10）'location', 位置信息

一个真实的文件内容示例为：

1）'ftp\_host':'192.168.1.51'

2）'ftp\_user':'lxyu'

3）'ftp\_pwd':'880127'

4）'db\_path':'/root/Develop/data/database/gkdb.db'

5）'dbisql\_bin':'/root/Gilbarco/gktest\_bak/gktest/sybase/'

6）'save\_dir':'/root/Develop/data/'

7）'db\_user\_name':'dba'

8）'db\_pwd':'sql'

9）'site':'heping'

10）'location':'JL\_CN'

在该文件中，会声明相应的查询语句，需要三个参数，分别为：结果输出路径，起始时间，结束时间。其中，输出路径为配置信息中的’save\_dir’的路径,起始时间和结束时间以当前日期的前一整天为范围，该文件中获取方式如下：

now\_time = datetime.datetime.now()

yes\_time = now\_time + datetime.timedelta(days=-1)

start\_time = yes\_time.strftime('%Y-%m-%d')+' 00:00:00'

end\_time = yes\_time.strftime('%Y-%m-%d')+' 23:59:59'

将参数传入SQL语句中：

SQL=\

"""

SELECT c.nodetag, a.gettime, a.takedate, a.tracode, a.cardno, a.paymode, a.balance, a.macno, b.oilname, a.price,

a.liter,a.amount,a.oilgunno,a.psam\_ttc,a.openo

FROM man\_oilvouch a,oiltype b, nodeinfor c

WHERE a.gettime>'%s' AND a.gettime<'%s' AND a.oilno=b.oilno

ORDER BY a.gettime ># %s

"""%(start\_time,end\_time,settings['save\_dir']+save\_name+'.txt')

（File.2）turbolinux\_sinopec\_auto\_upload.py

该文件首先通过turbolinux\_auto\_upload\_sinopec\_settings.py生成sql.sql查询脚本，通过os.chdir(dbisql\_bin\_dir)进入dbisql执行目录，调用dbisql查询输出语句：cmd='./dbisql -c "uid=%s;pwd=%s;dbf=%s;charset=utf-8" -nogui sql.sql'%(upload\_settings['db\_user\_name'], upload\_settings['db\_pwd'],dbf)，并执行

ret=os.system(cmd)

结果查询并输出到save.txt，同时生成一个以.sinopec为后缀的空文件，之后，连接ftp，将save.txt和.sinopec文件上传至ftp指定站点目录下，如/home/cnpc\_jl/puqing，ftp上文本命名规范为：站点名\_位置名\_年\_月\_日.sinopec,站点名\_位置名\_年\_月\_日.txt。

例如：puqing\_JN\_CN\_2015\_05\_30.sinopec，puqing\_JN\_CN\_2015\_05\_30.txt

1.2.3 执行过程

1）手工执行：

配置好相关信息后，执行python turbolinux\_auto\_upload.py即可。

2）自动执行：

例如每天凌晨2点执行脚本上传：

终端输入crontab -e，加入0 2 \* \* \* python /home/yulongxue/gflux/gflux/trunk/deployment/ trubolinux\_sinopec\_auto\_upload.py>>/home/yulongxue/log.log

命令解释：分钟 小时 日期 月份 周 命令

1.3 省公司中心主动查询获取

采取此种模式时，要求中石化省公司将每天的原始交易数据以指定格式将数据上传到内网的FTP服务器。

1.3.1 油品字段说明

中石化站级服务器获取油品交易记录数据的SQL查询语句如下（以2014.12.17日举例）：

SELECT c.nodetag, a.gettime, a.takedate, a.tracode, a.cardno, a.paymode, a.balance, a.macno, b.oilname, a.price,

a.liter, a.amount, a.oilgunno, a.psam\_ttc, a.openo

FROM man\_oilvouch a, oiltype b, nodeinfor c

WHERE a.gettime>'2014-12-17 00:00:00' AND a.gettime<'2014-12-17 23:59:59' AND a.oilno=b.oilno

ORDER BY a.gettime >#result.txt

该sql语句执行完成之后会生成一个txt文件，中石化将该txt文件上传到ftp服务器即可。

需要在省公司中心执行类似SQL语句导出数据。

上述每行数据意义如下：

1）加油站名称；

2）交易进行时间；

3）交易开始时间；

4）交易编码（此处交易编码为系统计算）；

5）卡号；

6）交易类型（此处根据卡号计算为现金或刷卡）；

7）卡余额；

8）油枪；

9）油品名称；

10）单价；

11）数量；

12）消费金额；

13）油枪号；

14）PSAM 序列号；

15）员工号。

数据实例：

'江南黄泥磅加油站', 2014-12-17 23:09:42.0,2014-12-17 23:09:39.0, '08', 00000000000000 00000', '00',0.00,7, '93号车用汽油（Ⅳ）',0.00,0.00,0.00,'013',0,'00'

'江南黄泥磅加油站', 2014-12-17 23:09:42.0,2014-12-17 23:09:40.0,'08','00000000000000 00000','00',0.00,2,'97号车用汽油（Ⅳ）',0.00,0.00,0.00,'003',0,'00'

'江南黄泥磅加油站', 2014-12-17 23:25:35.0,2014-12-17 23:25:28.0,'00','10004150000000 04902','00',40000000.00,2,'93号车用汽油（Ⅳ）',6.57,0.00,0.00,'004',0,'06'

1.3.2 非油品字段说明

中石化暂未提供非油品，仿照中石油提供的非油品数据，建议数据格式如下：

1167393,9030270001862570,2015-4-24 19:6:49,70006540,1,0.0,福临门 大豆油 5L,1.0, 69.5, 69.5, 桶, 0

1167393,9030270001639604,2015-4-24 19:6:49,70061099,1,0.0,福临门 AE营养一级大豆油 5L,1.0,70.5,70.5,桶,0

数据意义如下：

交易号，卡号，时间，条形码，数量[非油品有效]，金额[油品有效，非油品为0]，商品名，体积，商品单价，金额[非油品有效]，商品单位，油枪号[油品有效，非油品为0]。

1.3.3 上传的目录与文件名格式标准

站级服务器将查询结果的每一行以“,”分开，文件命名为：站点\_位置\_年\_月\_日.txt，同时上传一个以站点\_位置\_年\_月\_日.sinopec为命名格式的空文件，例如：puqing\_JL\_CN\_2015\_05\_30.txt和puqing\_JL\_CN\_2015\_05\_30.sinopec

上传到ftp的相应的站点文件夹下，如：/home/sinopec\_jl/puqing

2 数据接收(FTP)

上一节上传的数据，在省公司采取FTP服务器的方式进行接收和处理。与上一节的数据相对应，有三种对应的接收模式：

2.1 接收原始数据库文件

接收原始数据库文件(.db文件)在[ftp.marketcloud.com.cn](ftp://ftp.marketcloud.com.cn/)上面，每个用户都将数据上传到此目录。

2.1.1 运行环境和FTP配置

采用此种接受方式的FTP服务器运行环境为windows xp，安装好sysbase数据库：PB90Ent下setup安装即可。

2.1.2 处理过程

需要创建两个python文件：

（File.1）sinopec\_auto\_upload\_settings.py

其需要手动配置的内容格式说明为：

'db\_dir',数据库文件存放路径

'tmp\_sql\_dir',sql查询脚本存放路径

'save\_dir'，查询结果的保存路径

'dbisql\_dir'，dbisql的路径

'db\_user'，db用户名

'db\_password',db密码

'remote\_server'，部署gflux服务器的地址

'remote\_token'',部署gflux服务器的密码

'remote\_store\_dir',处理好的数据上传到gflux服务器的路径

'main\_project\_path'，gflux执行路径

11）'uploaded\_flag' 判断是否有需要上传文件的标识，即文件的后缀名

例如：

1）'db\_dir'：'C:\datas\database\gkdb.db'

2）'tmp\_sql\_dir'：'c:/tmp/sql.sql'

3）'save\_dir'：'c:/tmp/',

4）'dbisql\_dir'：'C:\Program Files (x86)\Sybase\SQL Anywhere 8\win32'

5）'db\_user'：'dba'

6）'db\_password'：'sql'

7）'remote\_server':'work@gflux3.tmlsystem.com',

8）'remote\_token':'plm!@#qaz'

9）'remote\_store\_dir'：'/home/work/data/'

10）'main\_project\_path'：'/home/work/develop/gflux/trunk/gflux'

11）'uploaded\_flag'：'.sinopec'

在该文件中，会声明相应的查询语句，需要三个参数，分别为：结果输出路径，起始时间，结束时间。其中，输出路径为配置信息中的’save\_dir’的路径,起始时间和结束时间以当前日期的前一整天为范围，该文件中获取方式如下：

now\_time = datetime.datetime.now()

yes\_time = now\_time + datetime.timedelta(days=-1)

start\_time = yes\_time.strftime('%Y-%m-%d')+' 00:00:00'

end\_time = yes\_time.strftime('%Y-%m-%d')+' 23:59:59'

将参数传入SQL语句中：

SQL=\

"""

SELECT c.nodetag,a.gettime,a.takedate,a.tracode,a.cardno,a.paymode,a.balance,a.macno,b.oilname,a.price,

a.liter,a.amount,a.oilgunno,a.psam\_ttc,a.openo

FROM man\_oilvouch a,oiltype b, nodeinfor c

WHERE a.gettime>'%s' AND a.gettime<'%s' AND a.oilno=b.oilno

ORDER BY a.gettime ># %s

"""%(start\_time,end\_time,settings['save\_dir'])

（File.2）sinopec\_auto\_upload.py

该文件首先通过sinopec\_auto\_upload\_settings.py生成相应的sql查询脚本，通过os.chdir (dbisql\_bin\_dir)进入dbisql执行路径，调用cmd='dbisql -c "uid=%s; pwd=%s; dbf=%s; charset= utf-8" READ %s -nogui -onerror exit'%(uid,pwd,dbf,sql\_dir)和ret=os.system(cmd)将结果输出到指定路径的文本下（上小节中配置的save\_dir），通过fabric将文本上传至gflux服务器的指定目录下（settings文件下的remote\_store\_dir），并调用导入脚本导入数据，命令如下：

django\_path=settings['main\_project\_path']

import\_data\_command='python manage.py submit\_sinopec\_task --user tao --file %s --site %s --location\_name %s '%(dir\_path,site,location)

with cd(django\_path):

run(import\_data\_command)

Fabric的安装命令为sudo pip install fabric,通过在终端运行python-c"from fabric.api import \* ; print env.version"检查是否安装成功。

2.2 接收站级增量上传数据（txt文件）

2.2.1 运行环境和FTP配置

FTP服务器运行环境为Ubuntu或centos系统，配置为可写入文件。FTP服务器配置过程如下：

1）sudo apt-get install vsftpd 安装ftp

2）sudo service vsftpd restart查看是否安装成功

3）sudo mkdir /home/yulongxue/sinopec创建一个目录作为ftp上传目录

4）打开"终端窗口"，输入"sudo useradd -d /home/yulongxue/sinopec -s /bin/bash uftp"-->回车-->用户新建成功-->输入"sudo passwd uftp"设置uftp用户的密码-->回车-->输入两次密码-->回车-->密码设置成功.

5）打开终端，输入"sudo gedit /etc/vsftpd.conf"-->回车-->打开了vsftpd.conf文件，向文件中添加"userlist\_deny=NO，userlist\_enable=YES userlist\_file=/etc/allowed\_users"和"seccomp\_sandbox=NO"，修改文件中的"local\_enable"为"local\_enable=YES" 设置ftp权限为可访问可写入

6）打开"终端窗口"，输入"sudo gedit /etc/allowed\_users"-->回车-->输入你设置的用户名（例如我们上面设置的是uftp）-->保存 设置可访问用户

7）打开"终端窗口"，输入"sudo gedit /etc/ftpusers"-->回车-->打开这个文件后，看一看有没有uftp（上面设置的用户名）这个用户名，如果没有，就直接退出。如果有就删除uftp（上面设置的用户名）,因为这个文件中记录的是不能访问FTP服务器的用户清单。 设置用户黑名单

2.2.2 处理过程

中石化在ftp服务器上的处理过程为文本处理、数据上传。

需要先添加两个文件：

（File.1）sinopec\_auto\_upload\_settings\_by\_txt.py

首次需要进行setting（sinopec\_upload\_settings\_by\_txt）的配置，内容如下：

'data\_dir' 需要处理的文件的目录路径

'uploaded\_flag' 判断是否有需要上传文件的标识，即文件的后缀名

'remote\_server', django服务器地址

'remote\_token', django密码

'remote\_store\_dir'文本上传到部署gflux服务器的路径

'main\_project\_path'，gflux运行路径

例如：

1) 'data\_dir':'/home/sinopec\_jl/puqing/'

2) 'uploaded\_flag':'.sinopec'

3) 'remote\_server':'work@gflux3.tmlsystem.com', django服务器地址

4) 'remote\_token':plm!@#qaz', django密码

5) 'remote\_store\_dir':'/home/work/data/puqing',

6) 'main\_project\_path':'/home/work/develop/gflux/trunk/gflux',

（File.2）sinopec\_auto\_upload\_by\_txt.py

该文件需要首先查询是否有需要上传的数据，根据上小节中配置的上传标志识别，如果有的话，通过fabric远程上传文本并调用导入数据脚本，该文件中具体实现过程如下：

from fabric.api import put,env,run,cd

env['host\_string']=settings['remote\_server']

env['passwords'] = {

env['host\_string']:settings['remote\_token']

}

put(file\_path,settings['remote\_store\_dir']+file\_name+'.txt')

django\_path=settings['main\_project\_path']

import\_data\_command='python manage.py submit\_sinopec\_task --user tao --file %s --site %s --location\_name %s '%(dir\_path,site,location)

with cd(django\_path):

run(import\_data\_command)

执行后，登录gflux的服务器上，通过定期查看process.log，查看导入过程是否有异常存在。

2.2.3 执行过程

1）手动运行

运行python sinopec\_auto\_upload\_by\_txt.py即可。

2）自动运行

在部署着ftp的服务器上，通过crontab定时执行查询处理脚本sinopec\_auto\_upload\_ by\_txt.py，步骤如下：

终端运行crontab -e，编辑配置文件

选择要使用的编辑器，如vi

写配置文件

例如：

# m h dom mon dow command

0 2 \* \* \* python /home/yulongxue/gflux/sinopec\_auto\_upload\_by\_txt.py

2.3 接收省中心指定格式上传数据

省中心上传的文件命名规范如下：站名\_位置\_年\_月\_日.txt，同时要上传一个站名\_位置\_年\_月\_日.sinopec的空文件，作为需要导入数据的标志。

例如puqing\_JL\_CN\_ 2015\_05\_30 .txt和puqing\_JL\_CN\_2015\_05\_30.sinopec。在FTP服务器的执行过程同2.2.3小节相同。

3 Gfulx部署

3.1 运行环境

此系统要求运行于Ubuntu 12.0.4 LTS操作系统之上，要求硬件服务器具有2GB以上内存和60GB以上的数据盘空间。

在缺省安装上述操作系统之后，运行python命令检查语言版本，必须为2.7.3以上、3.0以下版本。

用如下命令安装必须软件库：

sudo apt-get install build-essential,gearmand,memcached,postgresql

然后用如下命令安装Python库：

sudo pip install django==1.6 gearman python-memcached django-localeurl sqlalchemy psycopg2 python-dateutil flup xlrd

sudo yum install gcc-gfortran libgfortran blas-devvel blas lapack lapack-devel gcc-c++ libuuid-devel gperf boost-devel memcached memcached-devel freetype-devel libpng-devel

sudo pip install numpy

sudo pip install scipy matplotlib scikit-learn

3.2 服务器程序打包

使用:

import compileall

compileall.compile\_dir(r'\*××××××××')

例如：

python -c "import compileall; import re; compileall.compile\_dir('gflux', rx=re.compile (r'[/\\][.]svn'), force=True)"

将gflux目录下的代码文件编译，然后删除gcustomer/gadvertise以及其它不需要的部分，然后将文件打包，去除所有的py（除了mange.py）和.txt文件并打包成gflux.tar.gz。

具体的打包命令如下：

tar zcvf gflux.tar.gz

3.3 服务器程序安装

服务器程序软件包为gflux.tar.gz，将其在Linux系统解压缩的命令如下：

tar zxvf gflux.tar.gz

假定解压缩之后程序包的路径为：/home/user/gflux/，里面应含manage.py文件和gflux子目录。称该路径为GFLUX\_DIR。

3.4 服务器配置与启动

假定服务器所在的域名为XXX.COM, 服务器程序被部署之后的URL地址为[http://xxx.com/](http://abc.yunji.com.cn/yunji/)gflux/，称xxx.com为GFLUX\_DOMAIN。

3.4.1 配置nginx

在nginx的配置文件中加入：

upstream gflux\_fastcgi {

server 127.0.0.1:9091 weight=1 max\_fails=3 fail\_timeout=300s;

}

server {

………………………………..

location /yunji {

fastcgi\_pass gflux\_fastcgi;

fastcgi\_param PATH\_INFO $fastcgi\_script\_name;

fastcgi\_param REQUEST\_METHOD $request\_method;

fastcgi\_param QUERY\_STRING $query\_string;

fastcgi\_param CONTENT\_TYPE $content\_type;

fastcgi\_param CONTENT\_LENGTH $content\_length;

fastcgi\_param SERVER\_PROTOCOL $server\_protocol;

fastcgi\_param SERVER\_PORT $server\_port;

fastcgi\_param SERVER\_NAME $server\_name;

fastcgi\_param REMOTE\_ADDR $remote\_addr;

fastcgi\_param REMOTE\_PORT $remote\_port;

fastcgi\_pass\_header Authorization;

fastcgi\_intercept\_errors off;

}

………………………………..

}

此项配置将/gflux目录以fastcgi的模式导向本地的9091服务端口。

3.4.2 配置PostgreSQL数据库

sudo -u postgres psql

#create user tao with password 'pku123';

#alter role tao with superuser;

#create database gflux;

sudo service postgresql stop

sudo vi /etc/postgresql/9.\*/main/pg\_hba.conf

modify peer to md5 for local

modify add record host like below:

host all all 0.0.0.0/0 md5

sudo vi /etc/postgresql/9.\*/main/postgresql.conf

modify listen to \*

sudo service postgresql start

此项配置创建了一个数据库以及数据库用户，最后将数据库开放给所有ip段访问。

3.4.3 配置Django

在local\_settings.py中配置PYTHON\_BIN，DATABASES和SQL\_BACKENDS，其中PYTHON\_BIN是python可执行文件的路径。

运行如下命令创建表格和初始化数据:

python manage.py syncdb

python manage.py init\_gflux\_dev\_env

3.4.4 启动服务器

启动django服务：

python collectstatic\_manage.py collectstatic --noinput

python manage.py runfcgi daemonize=True host=127.0.0.1 port=9091

启动nginx服务：

service nginx restart

使用supervise启动django服务器：

supervise是一个小巧的进程守护工具，它能在进程崩溃退出的时候自动重启进程，在启动时只需传入文件夹路径，supervise就能自动运行run文件，运行如下命令启动

nohup ./supervise run\_fcgi/ &

3.4.5 添加计划任务

运行命令 crontab crontab.conf

里面有如下的计划：

1）compute\_gilbarco\_index 离线计算网感指数的脚本；

2）compute\_item\_assoc 计算相关性；

3）gearman\_worker\_compute\_fuel\_daybatch 离线计算油品每天销售；

4）gearman\_worker\_compute\_station\_daybatch 计算站点的逻辑天。

3.4.6 数据初始化

采取如下方式往数据库中导入样本数据进行开发调试：

1）启动django 服务器

2）启动gearman\_worker\_import\_data.py

3）打开上传网页，选择中油，使用data/all1000.txt 和 data/card1000.txt 上传。

4 Gflux数据导入

4.1 手动数据导入

油站人员手动上传gkdb.db和gkdb.log到FTP服务器，在FTP服务器上开启一个定时任务，定时查看是否有以.sinopec为后缀的文件，如果有，证明有数据需要导入，则可以根据该文件名找到相对应的txt文件，远程上传至Gflux上，并调用导入脚本导入数据。

导入命令举例：

python manage.py submit\_sinopec\_task --user tao --file /home/data/puqing\_JL\_CN\_ 2015\_05\_30.txt --site puqing --location\_name JN\_CN

4.2 自动增量数据导入

在站级服务器上，开启一个定时任务，查询前一天的所有数据并输出到一个文本，将该文本上传至FTP服务器对应的站点文件夹下；在FTP服务器上，同样开启一个定时任务，定时查看是否有以.sinopec为后缀的文件，如果有，证明有数据需要导入，则可以根据该文件名找到相对应的txt文件，远程上传至Gflux上，并调用导入脚本导入数据。

导入命令举例：

python manage.py submit\_sinopec\_task --user tao --file /home/data/puqing\_JL\_CN\_ 2015\_05\_30.txt --site puqing --location\_name JN\_CN

4.3 自动从省中心服务器导入

省中心以：站名\_位置\_年\_月\_日.txt和站名\_位置\_年\_月\_日.sinopec为命名规范，将结果文本上传到FTP服务器上，在FTP服务器上开启一个定时任务，定时查看是否有以.sinopec为后缀的文件，如果有，证明有数据需要导入，则可以根据该文件名找到相对应的txt文件，远程上传至Gflux上，并调用导入脚本导入数据。

导入命令举例：

python manage.py submit\_sinopec\_task --user tao --file /home/data/puqing\_JL\_CN\_ 2015\_05\_30.txt --site puqing --location\_name JN\_CN

5 系统扩容

5.1 数据库服务器扩容

根据3.4.2所述，新增数据库服务器后，在local\_settings.py中，在SQL\_BACKENDS中增加相关信息，例如：

{

'db\_name': 'first db',

'db\_url': 'postgresql://tao:pku123@127.0.0.1/gflux',

'db\_pool\_size': 15,

'db\_charset': 'utf-8',

'db\_pool\_recycle': 20000

}

5.2 django服务器扩容

如果在现有加油中添加一台新的django服务器，需要使用haproxy来配置。

haproxy安装过程如下：

sudo apt-get install haproxy

sudo nano /etc/default/haproxy ，将ENABLED=0修改为ENABLED=1

vi /etc/haproxy/haproxy.cfg添加监听状态

例如：

listen haproxy-status 127.0.0.1:8888

mode http

option httplog

stats uri /haproxy-status

stats realm Global\ statistics

stats auth admin:123456

六 服务器空间清理

6.1 磁盘空间查询（ubuntu）

在ubuntu环境下，需要定期查看磁盘空间，查询命令为:df -hl，例如在终端输入 df -hl，得到如下结果：

文件系统 容量 已用 可用 已用% 挂载点

/dev/sda1 451G 78G 351G 19% /

none 4.0K 0 4.0K 0% /sys/fs/cgroup

udev 3.8G 4.0K 3.8G 1% /dev

tmpfs 766M 1.3M 765M 1% /run

none 5.0M 0 5.0M 0% /run/lock

none 3.8G 105M 3.7G 3% /run/shm

none 100M 56K 100M 1% /run/user

通过查看已用百分比可以获悉磁盘空间使用情况，如果需要查看某个文件所占空间，则使用 du -k nohup.out命令，例如输出结果：

980840 nohup.out

6.2 磁盘空间清理

当磁盘空间占用率很高时则需要对空间进行清理 ，可以分为以下几个方面：

1）所有阿里云的机器我们都有第二块磁盘/data，如果是较大的数据的话，将数据存放在/data下。如果习惯在home目录下操作， 则创建一个软链接，比如现在work下的 data目录连着/data/gflux\_data目录，这样虽然是在写data目录，但实际上是在写/data所在的磁盘。

2）经常出现没有多少文件使用量(用df -m --max-depth=1可以看到)，这种情况下，一般是某个 正在运行的进程输出 文件占用了磁盘 空间，之所以看不到这个文件，是因为这个文件虽然被删除了，但在回收站里还没被系统资源回 收。用lsof | grep deleted可以看到所有磁盘上已经被删除，但 实际上被删除文件仍然在回收站里占用磁盘 空间的文件。采取两种方式来解决 被删除文件仍然在回收站里占用 空间的情况：

a. lsof | grep deleted，可以找到对应的进程号，杀掉该进 程，则它的文件被系统回收。简 单一点，我们杀掉所有旧的 python和 supervise进程，重启 它们，则回收站 被清空  
b. 杀掉旧 的进程并重启仍然无法避免将来 重新生成大的回收站文件，因此 我们在nohup时在后面加上 >/dev/null 2>&1 ，表示将标准输出和错误输出都直接扔掉。

6.3 定时清理nohup.out文件

通过对磁盘空间使用量的查看，可以看到nohup.out占有的空间会很大，因此我们需要定期去清空nohup.out文件，在目录/home/work/develop/gilbarco/trunk/scripts下，脚本clear\_disk\_space.py为清空nohup.out文件的脚本，设定每天凌晨三点钟定时执行该脚本清空文件，设置如下：

1）crontab -e编辑定时任务

2）加入0 3 \* \* \* python /home/work/develop/gilbarco/trunk/scripts/clear\_disk\_space.py,保存退出

3) crontab -l查看任务是否已经加入成功