**OceanArk V1.0.01安装部署手册**

版本：V1.0

共41页

(包括封面)

拟制

审核

**南京市中新赛克科技有限责任公司**

**目录**

**1 安装前说明 4**

**2 网关-gateway配置 4**

2.1 网关版本 4

2.2 修改网关目录下的配置文件application.yml 4

2.3 重启网关 6

*2.3.1 网关停止命令 6*

*2.3.2 启动网关 6*

**3 postgresql安装 7**

3.1 资源准备 7

3.2 数据库postgresql安装 7

3.3 数据库postgresql配置 8

3.4 数据库oeanark、knowledge初始化 11

**4 对接MPP（根据需要） 12**

4.1 确定CN所在服务器信息 12

4.2 创建oceanark用户 13

4.3 创建数据库并授权 13

4.4 刷数据库演示脚本 13

4.5 配置可访问数据库的网络地址 14

**5 Neo4j图库安装 14**

5.1 Neo4j安装 14

5.2 配置修改 15

5.3 启停与密码修改 16

5.4 服务器配置互信 16

5.5 数海方舟演示数据导入 16

**6 nginx安装 17**

6.1 获取nginx.tar.gz安装包 17

6.2 解压安装包 18

6.3 修改配置 18

6.4 启动nginx 18

6.5 停止nginx 18

6.6 重新加载nginx 19

6.7 配置文件服务器 19

**7 ES安装 20**

**8 nlp安装 20**

8.1 python安装 20

8.2 oceanark\_nlp安装部署 20

*8.2.1 起NER服务 21*

*8.2.2 csv文件入ES库 21*

*8.2.3 起雷达模型服务 21*

*8.2.4 起NLP解析服务 21*

**9 数海方舟-后台接口安装 22**

9.1 刷数据库脚本 22

9.2 ElasticSearch安装拼音插件 28

9.3 ElasticSearch安装IK分词插件 29

9.4 ElasticSearch创建索引 29

9.5 刷数据库演示脚本(根据需要) 30

9.6 修改配置文件application.yml 30

9.7 启动工程 34

9.8 模型后台安装 34

**10 数海方舟-前台安装 35**

10.1 基础安装 35

*10.1.1 获取前台系统文件夹oceanark\_web.V1.0.01.tar.gz 35*

*10.1.2 配置nginx 36*

*10.1.3 访问系统 37*

10.2 地图安装 38

10.3 演示音视频图片文档安装 39

10.4 知识库下载文件安装 40

10.5 配置文件说明 40

**11 附件1: ES安装手册 41**

**首先需要解压版本包，按照以下步骤安装配置启动：**

1. **安装前说明**

1、本系统基于用户权限系统，请先安装用户权限系统，版本信息如下

* 版本路径

/20 基线版本发布/04 大数据产品线/02 平台版本/OceanAuth/V1.0/V1.0.01.T6

* 文档列表

https://10.45.136.234/svn/BDC/OceanAuth/doc/V1.0/09 版本转测/V1.0.01.T6

2、以下系统配置文件中nacos服务地址配置为用户权限安装的地址。

3、postgresql和nginx可以复用用户权限已按照好的系统，也可以重新参考用户权限安装文档重新搭建。

4、因OceanAuthV1.0.01.T6版本中未包含OceanArk的初始化sql，故需要进行如下操作。

* 打开oceanauth\_increment.sql文件，执行里面的增量sql。
* 参照《OceanAuth V1.0.01.T6功能点界面操作使用手册.docx》为测试用户分配数海方舟系统的操作权限，并修改数海方舟应用地址。

5、Oceanark T4版本-【标签】模块依赖OceanMindV4.0.05 T3版本，且需要在该版本基础上打如下补丁，如果不需要【标签】模块，可忽略且在用户中心收回标签模块权限。



补丁路径：

https://art.sinovatio.com/artifactory/webapp/#/artifacts/browse/tree/General/bigdata-maven-virtual

1. **网关-gateway配置**
   1. **网关版本**

* 版本路径

/20 基线版本发布/04 大数据产品线/02 平台版本/OceanArk/V1.0.01/V1.0.01.T3

* 1. **修改网关目录下的配置文件application.yml**

进入网关安装目录，找到application.yml文件；

在配置文件application.yml添加红色字体部分即可，在security> permition的末尾添加 downLoadTXTFile。

|  |
| --- |
| server:  port: 9000  spring:  application:  name: oceanmind-gateway  cloud:  gateway:  discovery:  locator:  enabled: false  routes:  - id: oceanmind-datasource  uri: lb://oceanmind-datasource  predicates:  - Path=/connection/\*\*,/datamodel/\*\*,/rest/\*\*  - id: oceanmind-authority  uri: lb://oceanmind-authority  predicates:  - Path=/oauth/\*\*,/authority/\*\*  - id: oceanmind-service  uri: lb://oceanmind-dataservice  predicates:  - Path=/dataservice/\*\*  - id: oceanmind-datafactory  uri: lb://oceanmind-datafactory  predicates:  - Path=/datacollision/\*\*  - id: oceanmind-datashow  uri: lb://oceanmind-datashow  predicates:  - Path=/dataShow/\*\*,/dashboard/\*\*  - id: oceanmind-oplogmanage  uri: lb://oceanmind-oplogmanage  predicates:  - Path=/oplogmanage/\*\*  - id: oceanark  uri: lb://oceanark  predicates:  - Path=/oceanark/\*\*  nacos:  discovery:  server-addr: 10.45.150.187:8848  feign:  client:  config:  default:  connectTimeout: 5000  readTimeout: 5000  loggerLevel: basic  logging:  level:  root: info  org.springframework: info  org.springframework.cloud.gateway: info  security:  oauth:  header: Authorization  permition:  permitUrls:/rest/,/static/,/oauth/token,/exportFormData,/exportModelList,downLoadTXTFile |

* 1. **重启网关**
     1. **网关停止命令**

|  |
| --- |
| **#ps -ef | grep oceanmind-gateway-server | grep -v "grep" | awk '{print $2}' | xargs kill -9** |

* + 1. **启动网关**

|  |
| --- |
| **# nohup java -jar oceanmind-gateway-server-XXXX.jar -Dspring.config.location=application.yml >catalina.out 2>&1 &** |

注：xxxx对应着具体版本号，部署的时候根据具体版本修改。catalina.out生成的日志文件名，可以修改

1. **postgresql安装**
   1. **资源准备**
2. linux操作系统版本：

|  |
| --- |
| [root@rhino104 soft]**# cat /proc/version**  Linux version 3.10.0-862.el7.x86\_64 (builder@kbuilder.dev.centos.org) (gcc version 4.8.5 20150623 (Red Hat 4.8.5-28) (GCC) ) #1 SMP Fri Apr 20 16:44:24 UTC 2018 |

1. 安装postgresql数据库时，需要使用到root用户，并且需要新建postgres用户，用于起停postgresql数据库，数据库登录等数据库日常管理。

postgres用户新建与赋权：

postgresql 使用root用户无法启动，需要创建新的用户（postgres）来初始化启动。

|  |
| --- |
| **#cd /home**  **#useradd postgres**  **#mkdir -p /data01/pgsql/data**  **#chown -R postgres:postgres /data01/pgsql**  **#passwd postgres** |

输入新的密码并保存，例如：密码为postgres

* 1. **数据库postgresql安装**

1. 使用xshell工具上传postgresql数据库安装包（postgresql\_11.6.tar.gz）至/home /postgres目录，使用root用户进行上传操作。
2. 使用root登录至10.45.150.187，并切换至/home /postgres目录
3. 使用root账户进行登录,并输入密码登录

|  |
| --- |
| **#ssh root@10.45.150.187** |

1. 切换至home目录

|  |
| --- |
| **#cd/home/postgres** |

1. 解压安装文件至当前目录

|  |
| --- |
| **# tar -xvf postgresql\_11.6.tar.gz** |

* 1. **数据库postgresql配置**

1. postgres环境变量配置
2. 切换至postgre

|  |
| --- |
| **# su - postgres** |

1. 环境变量配置，在.bash\_profile的末尾添加如下项

|  |
| --- |
| **#vi ~/.bash\_profile** |

在.bash\_profile的末尾添加如下项

|  |
| --- |
| # postgres  PGPORT=5432  PGDATA=/data01/pgsql/data  export PGPORT PGDATA  export LANG=en\_US.utf8  export PGHOME=/home/postgres/postgresql  export LD\_LIBRARY\_PATH=$PGHOME/lib:/lib64:/usr/lib64:/usr/local/lib64:/lib:/usr/lib:/usr/local/lib:$LD\_LIBRARY\_PATH  export DATE=`date +"%Y%m%d%H%M"`  export PATH=$PGHOME/bin:$PATH  export MANPATH=$PGHOME/share/man:$MANPATH  export PGUSER=postgres  export PGHOST=127.0.0.1  export PGDATABASE=postgres |

1. 启用新配置的环境变量

|  |
| --- |
| **# source .bash\_profile** |

1. 查看环境变量是否设置成功

|  |
| --- |
| **# echo $PATH** |

查看输出结果，显示postgres用户环境变量配置成功。

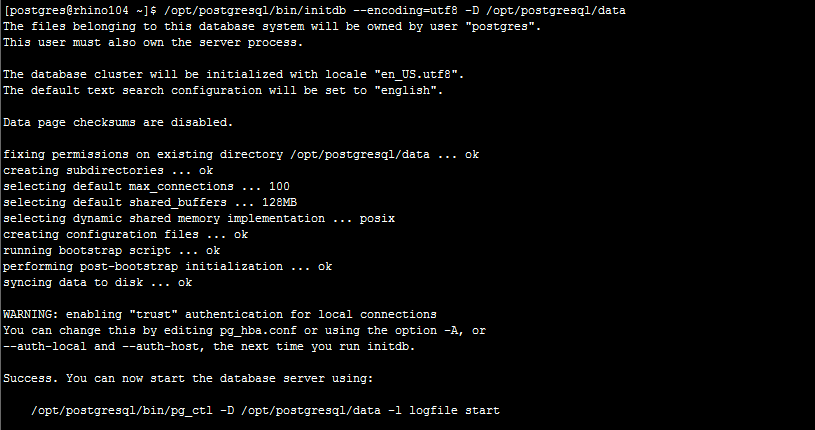
|  |
| --- |
| /home/postgres/postgresql/bin:/usr/local/bin:/bin:/usr/bin:/usr/local/sbin:/usr/sbin:/home/postgres/.local/bin:/home/postgres/bin:/home/postgres/.local/bin:/home/postgres/bin |

1. 初始化数据库

初始化数据库(会在data目录下生成配置文件)

|  |
| --- |
| /home/postgres/postgresql/bin/initdb --encoding=utf8 -D /data01/pgsql/data |

初始化成功



1. 启动数据库
2. 启动数据库脚本

|  |
| --- |
| **# /home/postgres/postgresql/bin/pg\_ctl -D /data01/pgsql/data -l logfile start** |

1. 关闭数据库脚本

|  |
| --- |
| **# /home/postgres/postgresql/bin/pg\_ctl -D /data01/pgsql/data -l logfile stop** |

1. 重启数据库脚本

|  |
| --- |
| **# /home/postgres/postgresql/bin/pg\_ctl -D /data01/pgsql/data -l logfile restart** |

1. 查看数据库状态脚本

|  |
| --- |
| **# /home/postgres/postgresql/bin/pg\_ctl -D /data01/pgsql/data -l logfile status** |

1. postgresql.conf文件配置，postgresql数据库的访问IP等配置
2. 编辑配置文件，配置可访问数据库的网络地址

|  |
| --- |
| **# vi /data01/pgsql/data/postgresql.conf** |

1. 添加一下内容，注意别忘了去掉#listen\_addresses=前面的#

|  |
| --- |
| listen\_addresses = '\*'  port = 5432  max\_connections = 10000  67 superuser\_reserved\_connections = 3  91 password\_encryption = on  101 tcp\_keepalives\_idle = 60  115 shared\_buffers = 6000MB  126 work\_mem = 4MB  151 vacuum\_cost\_delay = 10ms  159 bgwriter\_delay = 10ms  180 wal\_sync\_method = fsync  190 wal\_buffers = 128000kB  192 wal\_writer\_delay = 20ms  200 checkpoint\_timeout = 30min  201 checkpoint\_completion\_target = 0.9  321 logging\_collector = on  327 log\_directory = 'pg\_log'  329 log\_filename = 'postgresql-%Y-%m-%d\_%H%M%S.log'  333 log\_truncate\_on\_rotation = off  341 log\_rotation\_age = 0  343 log\_rotation\_size = 50MB  395 log\_min\_duration\_statement = 2000ms #跟踪哪些SQL执行时间长  434 log\_statement = 'ddl' #记录DDL语句, 一般用于跟踪数据库中的危险操作  407 log\_checkpoints = on #记录每一次checkpoint到日志中.  433 log\_lock\_waits = on #记录锁等待超过1秒的操作, 一般用于排查业务逻辑上的问题  558 deadlock\_timeout = 1s  451 track\_activity\_query\_size = 2048 #显示更长的SQL.  468 autovacuum = on  470 log\_autovacuum\_min\_duration = 0 #log\_autovacuum\_min\_duration = 0记录所有的autovacuum操作. |

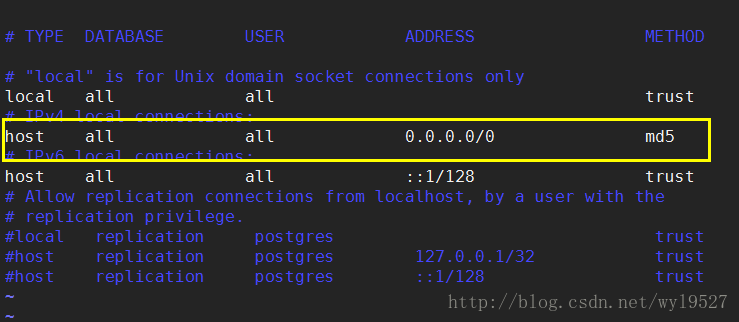
1. pg\_hba.conf数据库，postgresql数据库的访问配置

|  |
| --- |
| **# vi /data01/pgsql/data/pg\_hba.conf** |

添加一下内容

|  |
| --- |
| #to allow your client visiting postgresql server  host all all 0.0.0.0/0 trust |

其中Method 有trust（信任），md5（密码），首次使用postgres用户登录时，需要设置为trust，设置密码后，修改Method为md5，后期登录时，需要使用用户名和密码。



1. 所有配置配置成功后，需要重启数据库

|  |
| --- |
| **# /home/postgres/postgresql/bin/pg\_ctl -D /data01/pgsql/data -l logfile restart** |

* 1. **数据库oeanark、knowledge初始化**

创建用户oceanark，数据库oceanark，knowledge密码oceanark并且赋予权限

1. 输入psql登录

|  |
| --- |
| [postgres@rhino104 ~]$ **psql**  psql (11.6)  Type "help" for help.  postgres=# **create user "oceanark" with password 'oceanark';**  **postgres=# create database "knowledge" with owner="oceanark";**  postgres=# **create database "oceanark" with owner="oceanark";**  postgres=# **grant all privileges on database oceanark to oceanark;**  postgres=# **grant all privileges on database knowledge to oceanark;** |

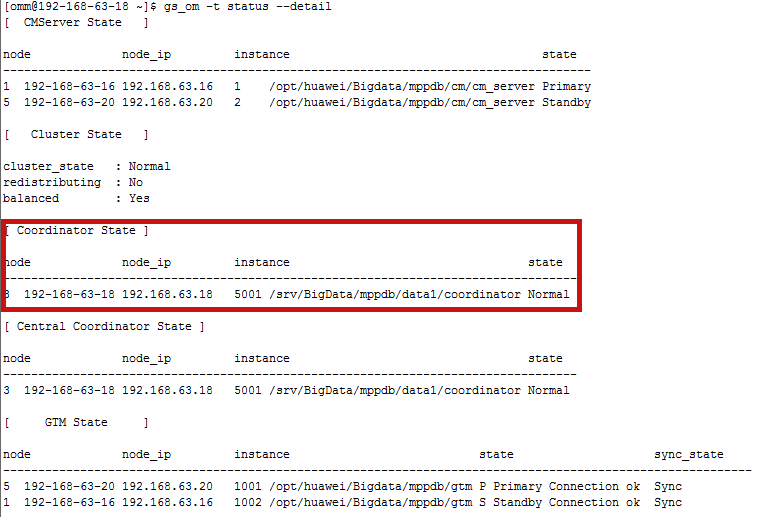
1. 退出当前登录

|  |
| --- |
| postgres=# **\q** |

1. **对接MPP（根据需要）**
   1. **确定CN所在服务器信息**

步骤1. 以omm用户身份登录LibrA集群任一节点。执行source ${BIGDATA\_HOME}/mppdb/.mppdbgs\_profile命令启动环境变量。如果不知道omm用户密码，可先登录到root用户然后使用su – omm命令切换至omm用户

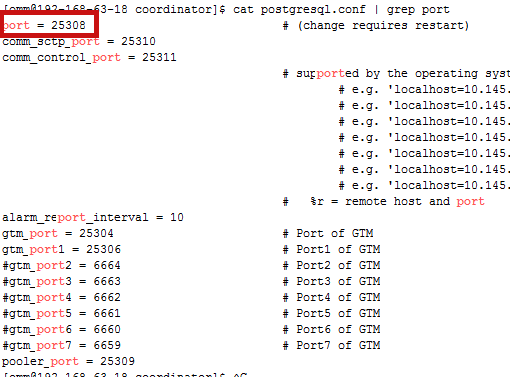
步骤2. 使用“gs\_om -t status --detail”命令查询集群各实例情况，确认CN所在的服务器的IP地址及数据路径（端口号查询备用）。结果如下：



图中红框内为CN所在服务器，CN数据路径为/srv/BigData/mppdb/data1/coordinator

步骤3. 在步骤 2查到的CN数据路径下的postgresql.conf文件中查看端口号信息。示例如下：

cat postgresql.conf | grep port



图中红框中25308为端口号信息

* 1. **创建oceanark用户**

步骤1. 使用omm用户登录到CN服务器上，执行source ${BIGDATA\_HOME}/mppdb/.mppdbgs\_profile命令启动环境变量。

步骤2. 连接数据库

**gsql -d** *postgres -***p** *25308*

步骤3 创建数据库用户

**CREATE USER** *oceanark* **WITHPASSWORD "***Bigdata@123***";**

* 1. **创建数据库并授权**

步骤1. 创建oceanark数据库

**CREATE DATABASE** oceanark;

步骤2. 将oceanark数据库授权给oceanark用户

**grant all on database** oceanark **to** oceanark**;**

步骤3. 退出

postgres=# **\q**

* 1. **刷数据库演示脚本**

步骤1. 在CN所在服务器的root用户主目录下使用**mkdir sql**命令新建sql目录，使用xshell工具上传演示数据脚本到新建的sql目录下，ftp路径为/02 平台版本/OceanArk/V1.0.01/V1.0.01.T3/V1.0.01.T3\_doc/OceanArk V1.0.01.T3 sql演示脚本

步骤2. 可根据注释修改execute\_sql\_mpp.conf

|  |
| --- |
| cd sql  使用chmod +x \* 授予执行权限  vi execute\_sql\_mpp.conf |

|  |
| --- |
| #!/bin/sh  #端口号  PG\_PORT=25308  #数据库名  DBNAME=oceanark  #用户名  USERNAME=oceanark  #密码  PASSWORD=Bigdata@123  SQLDIRPATH=`pwd`  source ${BIGDATA\_HOME}/mppdb/.mppdbgs\_profile  PGSQL\_COMMAND="gsql -d ${DBNAME} -p ${PG\_PORT} -U ${USERNAME} -W ${PASSWORD}"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/dl.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/dw.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/ods.sql" |

步骤3. 将sql目录拷贝到omm用户主目录下

cp -r ~/sql /home/omm/

步骤4. 执行刷库脚本execute\_sql\_mpp.conf

.使用su - omm 切换到omm用户

cd sql

执行脚本 ./execute\_sql\_mpp.conf

* 1. **配置可访问数据库的网络地址**

步骤1. 以omm用户身份登录CN所在服务器，执行source ${BIGDATA\_HOME}/mppdb/.mppdbgs\_profile命令启动环境变量。

步骤2. 配置客户端认证方式，允许客户端以“oceanark”用户连接到CN服务器，此处远程连接禁止使用“omm”用户。

例如，下面示例中配置允许IP地址为192.168.55.136的客户端访问CN服务器。（注意ip地址一定要与CN服务器网络互通,且oceanark后台部署在该服务器上）

gs\_guc set -Z coordinator -N all -I all -h "host all oceanark 192.168.55.136/32 sha256"

1. **Neo4j图库安装**
   1. **Neo4j安装**
2. 使用xshell工具上传Neo4j数据库安装包(neo4j-community-3.4.7-unix.tar.gz)至/home/rhino目录，使用rhino用户进行上传操作。
3. 使用rhino登录至服务器，并切换至/home/rhino目录。

|  |
| --- |
| **# cd /home/rhino** |

1. 解压安装文件至当前目录。

|  |
| --- |
| **# tar -zxvf neo4j-community-3.4.7-unix.tar.gz** |

1. 使用xshell工具上传Neo4j的算法包(apoc-3.4.0.7-all.jar、graph-algorithms-algo-3.4.12.7.jar)至/home/rhino/neo4j-community-3.4.7/plugins目录，使用rhino用户进行上传操作。
2. 将版本包中的import.sh脚本以rhino用户上传至Neo4j服务器/home/rhino/neo4j-community-3.4.7/bin目录，授予执行权限。

* import.sh脚本路径：

/20 基线版本发布/04 大数据产品线/02 平台版本/OceanArk/V1.0.01/V1.0.01.T3/V1.0.01.T3\_doc/OceanArk V1.0.01.T3 Neo4j演示数据及脚本/

|  |
| --- |
| **# chmod +x import.sh** |

修改import.sh脚本

|  |
| --- |
| **# vi import.sh** |

将红色部分修改为neo4j的安装目录，末尾不要加斜杠/。

|  |
| --- |
| #!/bin/sh  typeset NEO4J\_HOME='/home/rhino/neo4j-community-3.4.7'  typeset BIN\_HOME=${NEO4J\_HOME}/bin  typeset node=""  typeset edge="" |

* 1. **配置修改**

1. 将版本包中的updateNeo4jConf.sh脚本以rhino用户上传至Neo4j服务器/home/rhino目录，授予执行权限。

* updateNeo4jConf.sh脚本路径：

/20 基线版本发布/04 大数据产品线/02 平台版本/OceanArk/V1.0.01/V1.0.01.T3/V1.0.01.T3\_doc/OceanArk V1.0.01.T3 Neo4j演示数据及脚本/

|  |
| --- |
| **# chmod +x updateNeo4jConf.sh** |

1. 执行以下命令即可

|  |
| --- |
| **# ./updateNeo4jConf.sh /home/rhino/neo4j-community-3.4.7/conf/neo4j.conf** |

其中**/home/rhino/neo4j-community-3.4.7**为neo4j安装路径。

* 1. **启停与密码修改**

1. 启动Neo4j

|  |
| --- |
| **# cd /home/rhino/neo4j-community-3.4.7/bin; ./neo4j start** |

1. 将neo4j默认密码(neo4j)修改为123456

|  |
| --- |
| **# cd /home/rhino/neo4j-community-3.4.7/bin;./cypher-shell -u neo4j -p neo4j**  **# CALL dbms.changePassword('123456');** |

1. 停止Neo4j

|  |
| --- |
| **# cd /home/rhino/neo4j-community-3.4.7/bin; ./neo4j stop** |

* 1. **服务器配置互信**

说明:如果neo4j部署的服务器与oceanark部署服务器为同一台且部署在统一用户，可跳过该章节。

设置各台服务器的rhino用户无密码访问，执行如下命令，出现提示后按回车。

|  |
| --- |
| **# ssh-keygen -t rsa** |

执行如下命令，设置各台机器的无密码访问，注意每台服务器都要执行。

|  |
| --- |
| **# ssh-copy-id rhino@127.0.0.1** |

**127.0.0.1修改为真实IP。**

注意：配置rhino无密码访问在所有部署oceanark和部署neo4j服务器上都要执行。

* 1. **数海方舟演示数据导入**

说明:此数据为演示所用，如果使用现场真实数据，可跳过该章节。

1. 将版本包中的图库演示数据rhino用户上传至Neo4j服务器**/home/rhino/neo4j-community-3.4.7**/import目录。

其中**/home/rhino/neo4j-community-3.4.7**为neo4j安装路径。

* 演示数据路径：

/20 基线版本发布/04 大数据产品线/02 平台版本/OceanArk/V1.0.01/V1.0.01.T3/V1.0.01.T3\_doc/OceanArk V1.0.01.T3 Neo4j演示数据及脚本

* 演示数据清单

edge.dat

node.dat

1. 停止Neo4j，并清空data目录下数据。

|  |
| --- |
| **# cd /home/rhino/neo4j-community-3.4.7/bin; ./neo4j stop**  **# cd /home/rhino/neo4j-community-3.4.7/; rm -rf data/databases** |

1. 执行以下命令导入图库数据。

|  |
| --- |
| **# nohup ./bin/neo4j-admin import --nodes "/home/rhino/neo4j-community-3.4.7/import/node.dat" --relationships "/home/rhino/neo4j-community-3.4.7/import/edge.dat" --database=graph.db --delimiter='\t' --ignore-duplicate-nodes=true --ignore-missing-nodes=true --ignore-extra-columns=true &** |

1. 启动Neo4j

|  |
| --- |
| **# cd /home/rhino/neo4j-community-3.4.7/bin; ./neo4j start** |

1. **nginx安装**
   1. **获取nginx.tar.gz安装包**

将版本包解压后的nginx.tar.gz安装包,移动到/home/rhino/目录下（不支持其它目录或者解压后的nginx移动到/home/rhino/下）。

|  |
| --- |
| **# mv nginx.tar.gz /home/rhino/** |

注：由于提供的nginx安装包是已经编译完成后的压缩包已经指定的位置，如果需要自己指定需要下载源码包编译安装。

* 1. **解压安装包**

|  |
| --- |
| **# tar -xzvf nginx.tar.gz** |

* 1. **修改配置**

1. 进入nginx目录

|  |
| --- |
| **# cd nginx** |

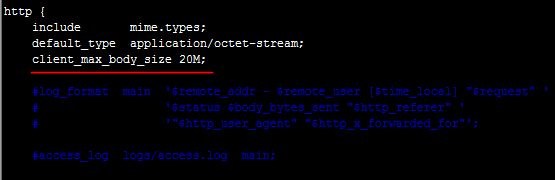
1. 进入conf目录

|  |
| --- |
| **# cd conf/** |

1. 打开nginx.conf文件

|  |
| --- |
| **# vim nginx.conf** |

在http{}里加如下配置：client\_max\_body\_size 20M;



* 1. **启动nginx**

1. 进入nginx/sbin目录

|  |
| --- |
| **# cd /home/rhino/nginx/sbin** |

1. 启动

|  |
| --- |
| **# ./nginx** |

* 1. **停止nginx**

1. 进入nginx/sbin目录

|  |
| --- |
| **# cd /home/rhino/nginx/sbin** |

1. 停止

|  |
| --- |
| **# ./nginx -s quit** |

* 1. **重新加载nginx**

1. 进入nginx/sbin目录

|  |
| --- |
| **# cd /home/rhino/nginx/sbin** |

1. 重新加载配置文件

|  |
| --- |
| **# ./nginx -s reload** |

* 1. **配置文件服务器**

1. 进入nginx/conf目录

|  |
| --- |
| **# cd /home/rhino/nginx/conf** |

1. 打开nginx.conf文件

|  |
| --- |
| **# vim nginx.conf** |

1. 在原来的nginx.conf里加一个server配置

说明：配置写在http { } 里面，与其他server 同级别。

|  |
| --- |
| server {  listen 8001;  server\_name 10.45.150.187;  location / {  root /var/www/html/;  autoindex on;  autoindex\_exact\_size off;  autoindex\_localtime on;  charset utf-8;  }  error\_page 500 502 503 504 /50x.html;  location = /50x.html {  root html;  }  } |

注：往指定服务器路径上传图片，视频，音频，文档用于关联媒体信息展示时，需要在文件->选项中勾选使用UTF-8编码，否则word文档再下载后可能出现乱码

1. 进入nginx/sbin目录，重新加载nginx

|  |
| --- |
| **# cd /home/rhino/nginx/sbin**  **# ./nginx -s reload** |

1. **ES安装**

参照11附件1: ES安装手册

1. **nlp安装**
   1. **python安装**

注：如果当前环境已安装oceanark的python包，则跳过此步骤

1. 获取python安装包 python3.zip，至/home/rhino目录下。
2. 切换到root用户，解压压缩包到指定路径：unzip python3.zip -d /usr/local/lib
3. 添加软链接：ln -s /usr/local/lib/python3/bin/python3 /usr/bin/python3

注：如果已有python3的软链接，则删除以前的软链接，重新设置。

1. 测试，输入python3，能够正常进入说明python安装正确
   1. **oceanark\_nlp安装部署**
2. 检查服务及端口占用情况：ps -ef|grep python3|grep -v grep，如果存在8081，8082，8084的端口占用，则kill掉相关进程



1. cd /home/rhino
2. 如/home/rhino/下存在oceanark\_nlp目录，运行rm -rf oceanark\_nlp删除该文件夹
3. 新建oceanark\_nlp目录：mkdir oceanark\_nlp
4. cd oceanark\_nlp
5. 获取data\_radarV1.0.01.T3.zip,解压至当前目录：unzip data\_radarV1.0.01.T3.zip
6. 获取data\_radar\_modelsV1.0.01.T3.zip，解压至当前目录：unzip data\_radar\_modelsV1.0.01.T3.zip
7. 获取entityextraction.zip，解压至当前目录：unzip entityextraction.zip
   * 1. **起NER服务**
8. cd /home/rhino/oceanark\_nlp/EntityExtraction，运行./start\_service.sh ip地址端口号

ip地址：起NLP解析的服务器地址

端口号：自行设置没有占用的端口（默认为8081）

注：如果端口被占用，则设置其它未被占用的端口。

1. 查看端口号对应的out文件，确认ner服务启动成功。

注：如服务的端口号为8081，则查看8081.out文件。

* + 1. **csv文件入ES库**

1. cd /home/rhino/oceanark\_nlp/data\_radar\_models/resource
2. 编辑setting.py文件，配置es\_host，es\_port为可用的ES数据库ip，端口，如需要可自行配置es\_index\_name参数
3. 运行es.py文件：python3 es.py,等待入库完成
   * 1. **起雷达模型服务**
4. cd /home/rhino/oceanark\_nlp/data\_radar
5. 执行./start\_radar\_model\_service.sh ip地址端口号

ip地址：当前服务器ip地址

端口号：自行设置没有占用的端口（默认为8084）

1. 查看端口号对应的out文件，确认雷达模型服务启动成功。

注：如服务的端口号为8084，则查看8084.out文件

* + 1. **起NLP解析服务**

1. cd /home/rhino/oceanark\_nlp/data\_radar，修改setting.py文件:

设置ner\_ip参数为当前服务器ip，设置ner\_port参数为2.1中设置的端口号，如2.1中端口号为8081，则无需改动；

设置es\_host，es\_port，es\_index\_name为2.2中设置的es\_host，es\_port，es\_index\_name；

设置radar\_id ，radar\_port为2.3中设置的ip地址，端口号;

设置theme\_db\_name为当前主题库的库名，resource\_db\_name为当前资源库中间表库名

1. 在oceanark\_nlp/data\_radar路径下，执行./start\_service.sh ip地址端口号

ip地址：起NLP解析的服务器地址

端口号：自行设置没有占用的端口（默认为8082）

1. 查看端口号对应的out文件，确认nlp解析服务启动成功。

注：如服务的端口号为8082，则查看8082.out文件。

1. **数海方舟-后台接口安装**
   1. **刷数据库脚本**
2. sql目录下是需要执行的数据库脚本，执行chmod +x \* 赋予所有文件执行权限

|  |
| --- |
| # cd sql  # vim execute\_sql.conf |

可根据注释修改刷库脚本中各项配置

|  |
| --- |
| #!/bin/sh  #端口号  PG\_PORT=5432  #数据库ip  PG\_IP=10.45.150.187  #数据库用户名  USERNAME=oceanark  #数据库密码  PASSWORD=oceanark  #数据库名  DBNAME=oceanark  #主题模式  SCHEMAZT=dw  #资源模式  SCHEMAZY=dl  export PGPASSWORD=${PASSWORD}  SQLDIRPATH=`pwd`  PGSQL\_COMMAND="psql -U ${USERNAME} -h ${PG\_IP} -d ${DBNAME} -p ${PG\_PORT}"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/create\_schema.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/ark\_base\_column\_expand.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/ark\_base\_errorcode.sql"  ${PGSQL\_COMMAND} -v SCHEMAZT="dw" -v SCHEMAZY="dl" -f "${SQLDIRPATH}/ark\_base\_neo4j.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/ark\_base\_table\_expand.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/ark\_base\_tag\_value.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/ark\_base\_taskinfo.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/ark\_interest\_change.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/ark\_interest\_change\_detail.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/ark\_interest\_change\_readstatus.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/ark\_interest\_info.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/ark\_interest\_version.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/ark\_radar\_search\_history.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/ark\_resource\_table\_expand.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/ark\_share\_snapshot\_expand.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/ark\_share\_snapshot\_info.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/ark\_share\_snapshot\_users.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/ark\_sync\_dictionary.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/ark\_sync\_standard\_info.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/ark\_sync\_standard\_rule.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/ark\_sync\_tag\_tree.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/ga\_kz\_gxlx.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/ga\_kz\_zdnr.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/ga\_kz\_zdxw.sql" |

1. 执行

|  |
| --- |
| **# ./execute\_sql.conf** |

步骤3

|  |
| --- |
| # cd sql  #cd knowledge  # vim execute\_sql.conf |

可根据注释修改知识库相应的配置

|  |
| --- |
| #!/bin/sh  #端口号  PG\_PORT=5432  #数据库ip  PG\_IP=10.45.150.187  #数据库用户名  USERNAME=oceanark  #数据库密码  PASSWORD=oceanark  #数据库名  DBNAME=knowledge  export PGPASSWORD=${PASSWORD}  SQLDIRPATH=`pwd`  PGSQL\_COMMAND="psql -U ${USERNAME} -h ${PG\_IP} -d ${DBNAME} -p ${PG\_PORT}"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/create\_schema.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_app/k\_app.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_bdk/k\_bdk.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_car/k\_car\_beidonganquan.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_car/k\_car\_biansuxiang.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_car/k\_car\_bolihoushijing.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_car/k\_car\_carboby.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_car/k\_car\_chelun.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_car/k\_car\_denguangpeizhi.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_car/k\_car\_dipan.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_car/k\_car\_duomeiti.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_car/k\_car\_fadongji.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_car/k\_car\_fangdao.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_car/k\_car\_jiashifuzhu.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_car/k\_car\_kongtiao.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_car/k\_car\_neibupeizhi.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_car/k\_car\_peizhi.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_car/k\_car\_waibupeizhi.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_car/k\_car\_yueye.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_car/k\_car\_zhudonganquan.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_car/k\_car\_zuoyipeizhi.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_chanyequ/k\_chanyequ.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_ershouwupin/spl\_tb\_ganji\_ershouwupin.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_fgwzcwj/k\_fgwjs.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_fgwzcwj/k\_fgwnj.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_gjbz/gjbz\_gb688.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_gxcqywyl/k\_gxcqywyl.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_gxtzcwj/k\_gxt.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_gxtzcwj/k\_jxwnj.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_huaxuewupin/spl\_tb\_aladdin\_chemicals.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_huochezhan/jcl\_tb\_hczxx.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_imei/k\_imei.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_ip/k\_ip.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_jrjgzcwj/k\_jrbnj.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_jrjgzcwj/k\_jsjrb.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_jsip/k\_jsip.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_kjtjzcwj/k\_kwnj.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_kjtjzcwj/k\_kxjst.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_loudongxinxi/jcl\_tb\_loudongxinxi.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_menjin/spl\_tb\_jingdong\_anfang.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_nynczcwj/k\_coajs.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_nynczcwj/k\_nyncjnj.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_poi/k\_anhui.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_poi/k\_aomen.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_poi/k\_chongqing.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_poi/k\_gansu.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_poi/k\_guizhou.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_poi/k\_hainan.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_poi/k\_henan.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_poi/k\_jilin.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_poi/k\_ningxia.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_poi/k\_qinghai.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_poi/k\_taiwan.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_poi/k\_tianjin.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_poi/k\_xianggang.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_poi/k\_xinjiang.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_poi/k\_xizang.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_qianlima/qianlima01.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_shenjilei/guojiashenji.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_shenjilei/k\_caizheng.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_shenjilei/k\_renshebu.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_sjdq/k\_sjdq.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_suoju/spl\_tb\_jingdong\_locks.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_swzcwj/k\_docjs.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_swzcwj/k\_xzspcjs.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_wbflywxwyl/k\_wbflywxwyl.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_xiangji/spl\_tb\_jingdong\_camera.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_xiaoyuzhong/k\_xiaoyuzhong.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_yuming/k\_yuming.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_zfcgxx/k\_anhui.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_zfcgxx/k\_chongqing.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_zfcgxx/k\_fujian.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_zfcgxx/k\_guangdong.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_zfcgxx/k\_jiangsu.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_zfcgxx/k\_tianjin.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_zhaopin/job\_58.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_zhenguominsu/k\_zhenguominsu.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_zscqzcfvfg/k\_jsip.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/knowledge\_phonehome/dic\_code\_phonehome.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/knowledge/oceainfo/info\_count.sql" |

* 1. **ElasticSearch安装拼音插件**

1. 根据安装的es版本去下载拼音分析插件（以6.1.3版本路径为例: https://github.com/medcl/elasticsearch-analysis-pinyin/releases/download/v6.1.3/elasticsearch-analysis-pinyin-6.1.3.zip）
2. 下载后解压zip到es的plugins路径下（解压后路径层次为<ES\_HOME>/plugins/<这是一层文件夹>/elasticsearch-analysis-pinyin-6.1.3.jar）
3. plugins路径下只允许有插件解压后的东西，zip不能放plugins路径下！！！
4. 然后重启elasticsearch
   1. **ElasticSearch安装IK分词插件**
5. 根据安装的es版本去下载ik分词插件（以6.1.3版本路径为例: https://github.com/medcl/elasticsearch-analysis-pinyin/releases/download/v6.1.3/elasticsearch-analysis-ik-6.1.3.zip）
6. 下载后解压zip到es的plugins路径下（解压后路径层次为<ES\_HOME>/plugins/<这是一层文件夹>/elasticsearch-analysis-ik-6.1.3.jar）
7. plugins路径下只允许有插件解压后的东西，zip不能放plugins路径下！！！
8. 然后重启elasticsearch
   1. **ElasticSearch创建索引**

192.168.55.137:9200修改为ElasticSearch的ip地址和端口。

|  |
| --- |
| # curl -XPUT "http://192.168.55.137:9200/ark\_search" -d '{"mappings":{"ark\_search":{"properties":{"insert\_time":{"type":"long"},"search\_column":{"type":"text","analyzer":"ik\_max\_word"},"theme\_type":{"type":"keyword"},"zjlid":{"type":"keyword"},"theme\_big\_sort":{"type":"keyword"},"label":{"type":"text"}}}}}' -H 'Content-Type:application/json' |

|  |
| --- |
| # curl -XPUT "http://192.168.55.137:9200/ark\_recommend" -d '{"mappings":{"ark\_recommend":{"properties":{"recommend\_column":{"type":"completion","analyzer":"pinyin\_analyzer"}}}}, "settings":{"analysis":{"analyzer":{"pinyin\_analyzer":{"tokenizer":"my\_pinyin"}},"tokenizer":{"my\_pinyin":{"type":"pinyin","keep\_separate\_first\_letter":true,"keep\_original":true}}}}}' -H 'Content-Type:application/json' |

|  |
| --- |
| # curl -XPUT http://192.168.55.137:9200/ark\_position -H 'Content-Type:application/json' -d '{"mappings": {"ark\_position":{"properties":{"insert\_time":{"type":"long"},"zjl\_position":{"type":"geo\_point"},"theme\_big\_sort":{ "type":"keyword"},"zjlid":{ "type":"keyword"},"theme\_type":{ "type":"keyword"}}}}}' |

* 1. **刷数据库演示脚本(根据需要)**

1. demo目录下是需要执行的数据库演示脚本，执行chmod +x \* 赋予所有文件执行权限

|  |
| --- |
| # cd sql  # vim execute\_sql.conf |

可根据注释修改刷库脚本中各项配置

|  |
| --- |
| #!/bin/sh  #端口号  PG\_PORT=5432  #数据库ip  PG\_IP=10.45.150.187  #数据库用户名  USERNAME=oceanark  #数据库密码  PASSWORD=oceanark  #数据库名  DBNAME=oceanark  export PGPASSWORD=${PASSWORD}  SQLDIRPATH=`pwd`  PGSQL\_COMMAND="psql -U ${USERNAME} -h ${PG\_IP} -d ${DBNAME} -p ${PG\_PORT}"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/dl.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/dw.sql"  ${PGSQL\_COMMAND} -f "${SQLDIRPATH}/ods.sql" |

1. 执行

|  |
| --- |
| **# ./execute\_sql.conf** |

* 1. **修改配置文件application.yml**

安装包解压路径/V1.0/ oceanark\_server/conf/

|  |
| --- |
| **# vim application.yml** |

(标红的息需要对应修改)

|  |
| --- |
| server:  port: 8908  servlet:  context-path: /oceanark  spring:  application:  name: oceanark  cloud:  nacos:  discovery:  server-addr: localhost:8848  postgresql:  datasource:  driver-class-name: org.postgresql.Driver  jdbc-url: jdbc:postgresql://10.45.150.86:5432/oceanark  username: oceanark  password: oceanark  warehouse:  datasource:  driver-class-name: org.postgresql.Driver  jdbc-url: jdbc:postgresql://10.45.150.86:5432/oceanark  username: oceanark  password: oceanark  knowledge:  datasource:  driver-class-name: org.postgresql.Driver  jdbc-url: jdbc:postgresql://10.45.150.86:5432/knowledge  username: oceanark  password: oceanark  #mybatis:  # type-aliases-package: com.sinovatio.oceanark.cache  # configuration:  # map-underscore-to-camel-case: true  # mapper-locations: classpath\*:/mapper/\*.xml  #数据抽取任务定时策略(配置方式按照每天执行一次的方式来配置,如下的方式)  task:  isPrimary: 0 #是否微服务实例群的主节点 0:是 1:否  extract:  neo4j: "0 0 1 \* \* ?" #每天1点下达图库数据抽取任务  ark-search: "0 20 1 \* \* ?" #每天1点20下发抽取ark\_search的任务  ark-recommend: "0 20 2 \* \* ?" #每天2点20下发抽取ark\_recommend的任务  ark-zt-position: "0 20 3 \* \* ?" #每天3点20下发抽取ark\_zt\_position的任务  focus-update: "0 0 4 \* \* ?" #每天4点下发关注更新的任务  step:  base: 10000 #数据雷达和位置模块抽取步长  logging:  level:  root: info  org.springframework: info  com.sinovatio.oceanark.object.warehouse.mapper: debug  security:  oauth2:  resource:  jwt:  key-value: jwtSecret  elasticsearch:  clusternode: 10.45.150.86:9200  clustername: fts #默认为elasticsearch  clusterid: 42 #conn\_param\_list的id值  repositories-enabled: true  isESSecurity: false # true华为安全模式  isC80: true # 是否是C80环境，如果是认证时加HTTP/  #schema: http  esAuthPath: /home/rhino/oceanark/auth/ #路径以'/'结尾  securityUsername: saikedaas  connectTimeOut: 1000 # 连接超时时间  socketTimeOut: 30000  connectionRequestTimeOut: 500  maxConnectNum: 100 #最大连接数  maxConnectPerRoute: 100 #最大路由连接数=======  neo4j:  url: bolt://10.45.150.86:7687 #neo4j地址  username: neo4j #neo4j用户名  password: 123456 #neo4j密码  datapath: /home/rhino/neo4j-community-3.4.7/import/ #neo4j数据目录  user: rhino #neo4j安装用户  homepath: /home/rhino/neo4j-community-3.4.7/ #neo4j家目录  host: 10.45.150.86 #neo4j安装服务器的地址  radar:  nlp:  switch: 0 # 雷达搜索nlp开关， 0 表示关(只ElasticSearch) 1表示只开NLP， 2表示NLP + ElasticSearch  url: http://10.45.150.187:8082/demo  time\_function: to\_date  time\_format: "YYYY-MM-DD HH24:MI:SS"  media:  uri: http://10.45.150.187:8001/ #配置文件服务器的ip和port  knowledge:  uri: http://10.45.150.187:8001/ #配置文件服务器的ip和port  username: rhino  password: rhino  remoteip: 10.45.150.187  port: 22  absolute\_parent\_path: /var/www/html/  schema:  zt: dw  zy: dl  #数据源类型,如mpp、pg、hive等，用于扩展针对不同的数据源使用不同的sql语句  warehouse:  type: mpp |

* 1. **启动工程**

cd到版本包/sbin的目录下，执行命令

|  |
| --- |
| **vim start.sh**  **#修改运行脚本**  **nohup java -jar -Dloader.path=../libs -Dspring.config.location=/home/rhino/oceanark/oceanark\_server/conf/application.yml oceanark-server-xxxx.jar >../logs/nohup.out 2>&1 &** |

注：xxxx对应着具体版本号，部署的时候根据具体版本修改。

* 1. **模型后台安装**

模型后台复用数据碰撞的后台，包括接口后台（datafactory）和大数据后台（shark-jetty、shark-tomcat）

* 版本路径：

/20 基线版本发布/04 大数据产品线/02 平台版本/OceanMind/001 版本包/小工具/V1/DataCollision/V1.0.1 T6

* 后台安装

参考数据碰撞安装文档，ftp路径：

/20 基线版本发布/04 大数据产品线/02 平台版本/OceanMind/001 版本包/小工具/V1/DataCollision/V1.0.1 T6/OceanMind\_Tools\_dataCollision V1.0.1 T6\_doc

说明：模型后台基于用户权限系统T4版本(及以上版本),数据源管理系统T2版本(及以上版本),daas系统V4.1.02 T2(及以上版本)，请先安装用户权限T4版本(及以上版本),数据源管理系统T2版本(及以上版本), daas系统V4.1.02 T2(及以上版本)。

**重要说明（关于数据源管理系统）：**

dataservice取华为版本的外发补丁包，ftp路径：/20 基线版本发布/04 大数据产品线/02 平台版本/OceanMind/001 版本包/小工具/V1/dataSource/V1.0.1 T3/外发补丁/T3.31511.HW的sparksql连接创建不成功/后台/dataservice

请参考补丁说明文档安装，ftp路径：/20 基线版本发布/04 大数据产品线/02 平台版本/OceanMind/001 版本包/小工具/V1/dataSource/V1.0.1 T3/外发补丁/T3.31511.HW的sparksql连接创建不成功

**重要说明（关于datafactory）：**

datafacotry需要将碰撞版本包中的oceanmind-datafactory-server-1.0.1.6.jar替换为oceanark版本包中V1.0/datafactory中的oceanmind-datafactory-server-1.0.1.6.jar

**重要说明（关于shark-jetty）：**

shark-jetty需要将部署好的.shark\_collision/jetty/WEB-INF/lib/shark-jetty.jar替换为oceanark版本包中V1.0/shark中的shark-jetty.jar，完成之后切换到.shark\_collision目录下执行如下命令压缩shark-jetty：

./wartool\_huawei.sh c jetty

压缩完成之后删掉已存在的Jetty进程

1. **数海方舟-前台安装**
   1. **基础安装**
      1. **获取前台系统文件夹oceanark\_web.V1.0.01.tar.gz**
2. 获取oceanark\_web.V1.0.01.tar.gz文件，解压文件

|  |
| --- |
| **# tar zxvfoceanark\_web.V1.0.01.tar.gz** |

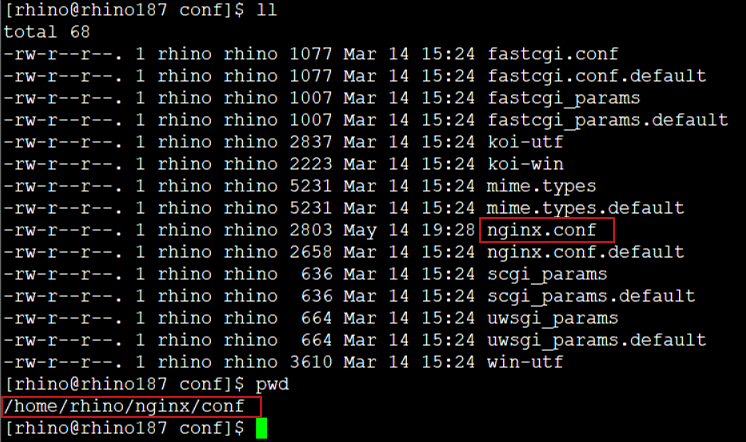
1. 将解压后的oceanark\_web文件，放置在/home/rhino/oceanark/文件夹下。（/home/rhino实际目录地址）

|  |
| --- |
| **# pwd**  /home/rhino/oceanark/oceanark\_web |

* + 1. **配置nginx**

1. 进入nginx文件夹：

|  |
| --- |
| **# cd /home/rhino/nginx/conf** |



1. 编辑配置文件，vim nginx.conf:

拷贝server配置段中内容，在server闭包下粘贴拷贝的内容，修改如下配置(如果是新的nginx直接修改server端对应配置)：

注：nginx每个配置以;结尾

|  |
| --- |
| server {  listen 8000; #替换为新的端口号  server\_name 10.45.150.187; #当前机器ip  location / {  root /home/rhino/oceanark/oceanark\_web; #实际路径  index index.html index.htm;  }  location ~ ^/(oauth|authority|oceanark|datacollision|dataservice) {  proxy\_pass http://10.45.150.187:9000; #配置网关地址  proxy\_connect\_timeout 10s;  proxy\_read\_timeout 600s;  proxy\_send\_timeout 10s;  proxy\_set\_header Upgrade $http\_upgrade;  proxy\_set\_header Connection "upgrade";  }  location ~ ^/(map) {  proxy\_pass http://10.45.150.187:8001;#配置文件服务器的ip和port  }  error\_page 500 502 503 504 /50x.html;  location = /50x.html {  root html;  }  } |

1. 进入nginx/sbin目录，重新加载nginx

|  |
| --- |
| **# cd /home/rhino/nginx/sbin**  **# ./nginx -s reload** |

* + 1. **访问系统**

浏览器输入地址(其中ip部分为nginx服务器地址，端口为nginx中配置的监听端口)：

http://10.45.150.187:8000/

安装成功后首页：

默认用户名\密码

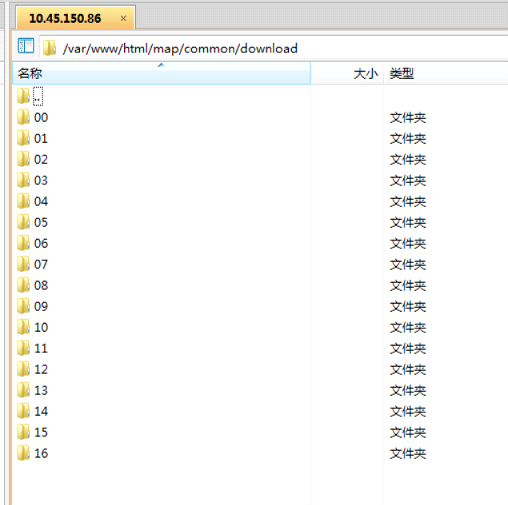
admin\admin

* 1. **地图安装**

1. 在/var/www/html路径下；

/var/www/html路径为章节5.7配置文件服务器中的文件服务器地址

1. 创建一个文件目录map/common/download,如/var/www/html/map/common/download，可将10.45.150.86服务器上，/var/www/html/map/common/download目录上的文件拷贝至实际目录地址中。

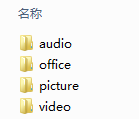


* 1. **演示音视频图片文档安装**

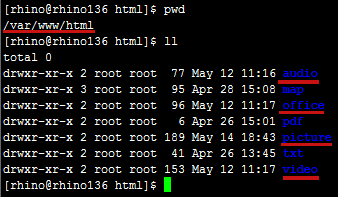
1. 进入现场实际服务器，在根目录下新建/var/www/html/目录；
2. 然后在/html目录下新建picture、office、audio、video目录，分别用于存放图片、文档、音频、视频；
3. 从版本包中取出“关联媒体样例数据”文件夹，拷贝到现场服务器的实际目录地址中；

**注意：**

（1）关联媒体接口ip和端口配置，参见后台部署文件



上图展示的是的“关联媒体样例数据”文件夹情况



* 1. **知识库下载文件安装**

1. 进入现场实际服务器，在根目录下新建/var/www/html/目录；

2.. 在版本包中取出“知识库下载文件”文件夹中的pdf.tar.gz，放到/var/www/html目录中

执行

tar -zxvf pdf.tar.gz

**注意：**

关联媒体接口ip和端口配置，参见后台部署文件

* 1. **配置文件说明**

1. 配置文件位置：oceanark\_web/static/settings.json
2. 各参数说明如下：

|  |  |  |
| --- | --- | --- |
| 参数名 | 类型 | 备注 |
| map\_center | 数组，[114.52,38.05] | 地图中心点经纬度 |
| map\_zoom | 数值，8 | 地图默认层级 |
| map\_minZoom | 数值，0 | 最小缩放层级 |
| map\_maxZoom | 数值，16 | 最大缩放层级 |
| timeout | 数值，单位为毫秒，5000 | 超时时间，模型那块需要前台将超时时间设置为120000 |