iVDG 部署升级手册

--Docker-Compose 版

修订历史

版本号	修订日期	修订人	章节	更改原因
V1.0	2021/7/25	时建	All	文档创建
V1.1	2021/7/27	张勇	All	文档完善
V1.2	2021/8/7	后欣	All	文档完善，适配 iVDG_V1.0.2 版本
V1.3	2021/9/7	后欣	All	文档完善，适配 iVDG_V1.0.3 版本
V1.4	2021/10/11	后欣	All	文档完善，适配 iVDG_V1.0.4 版本

目录

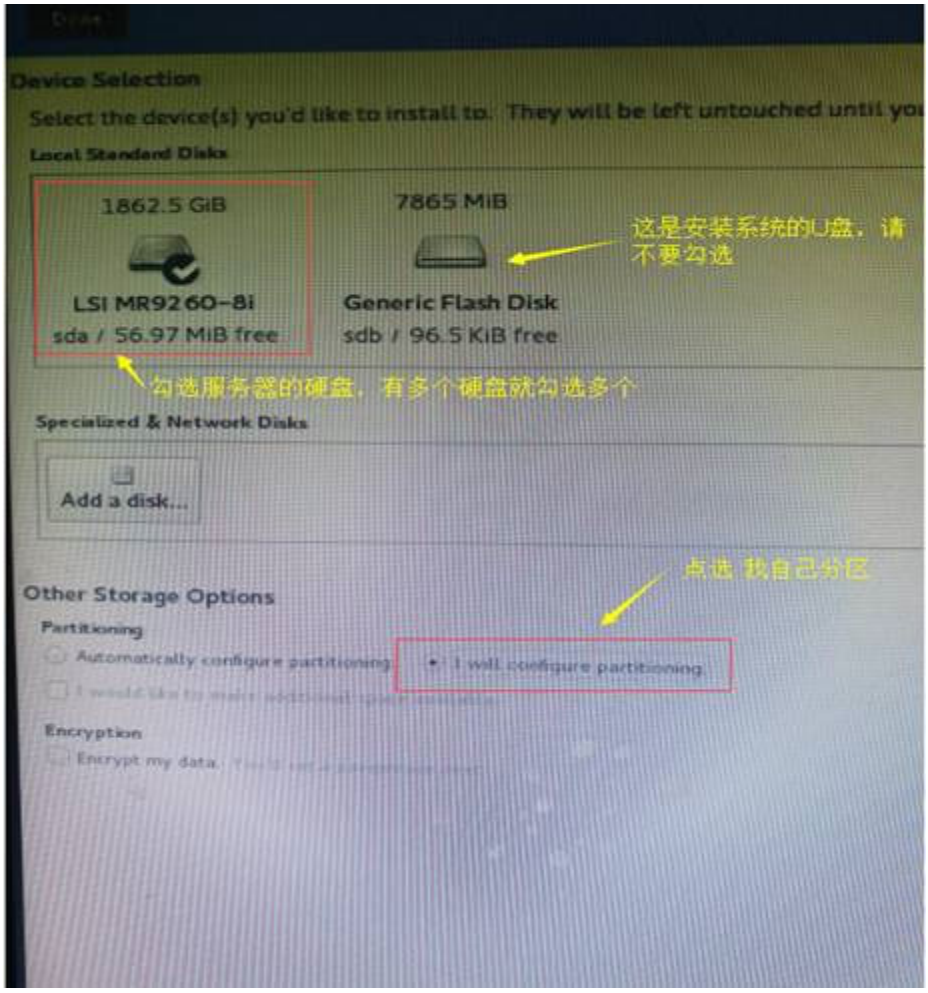
一、 操作系统安装

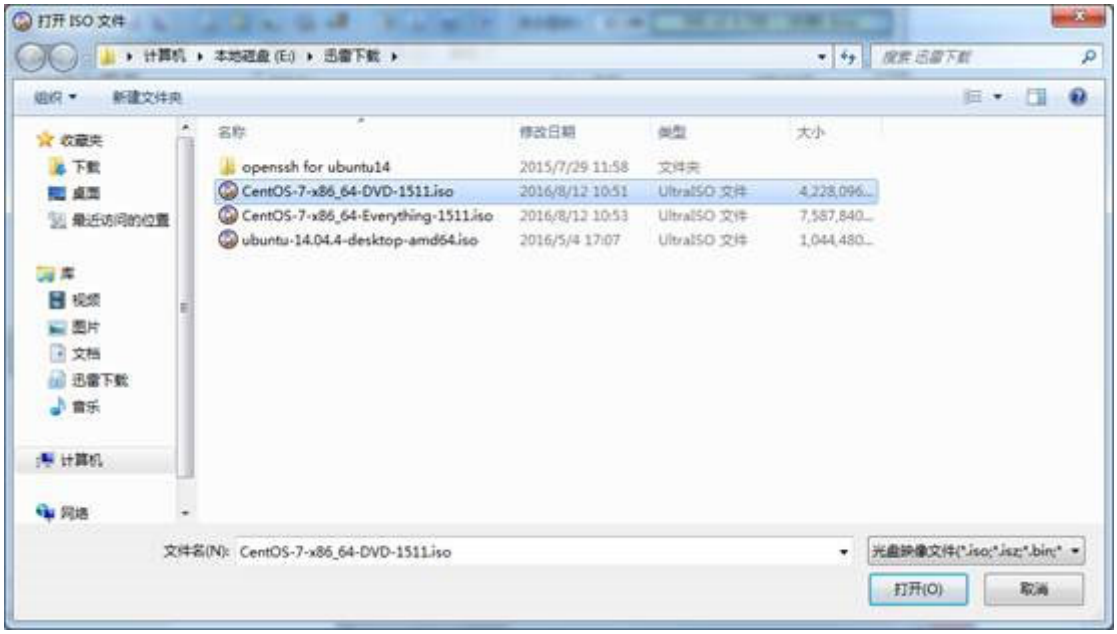
1.1. 制作 Centos 安装 u 盘

工具：UltraISO 软件，Centos 镜像文件(CentOS-7-x86_64)，U 盘。

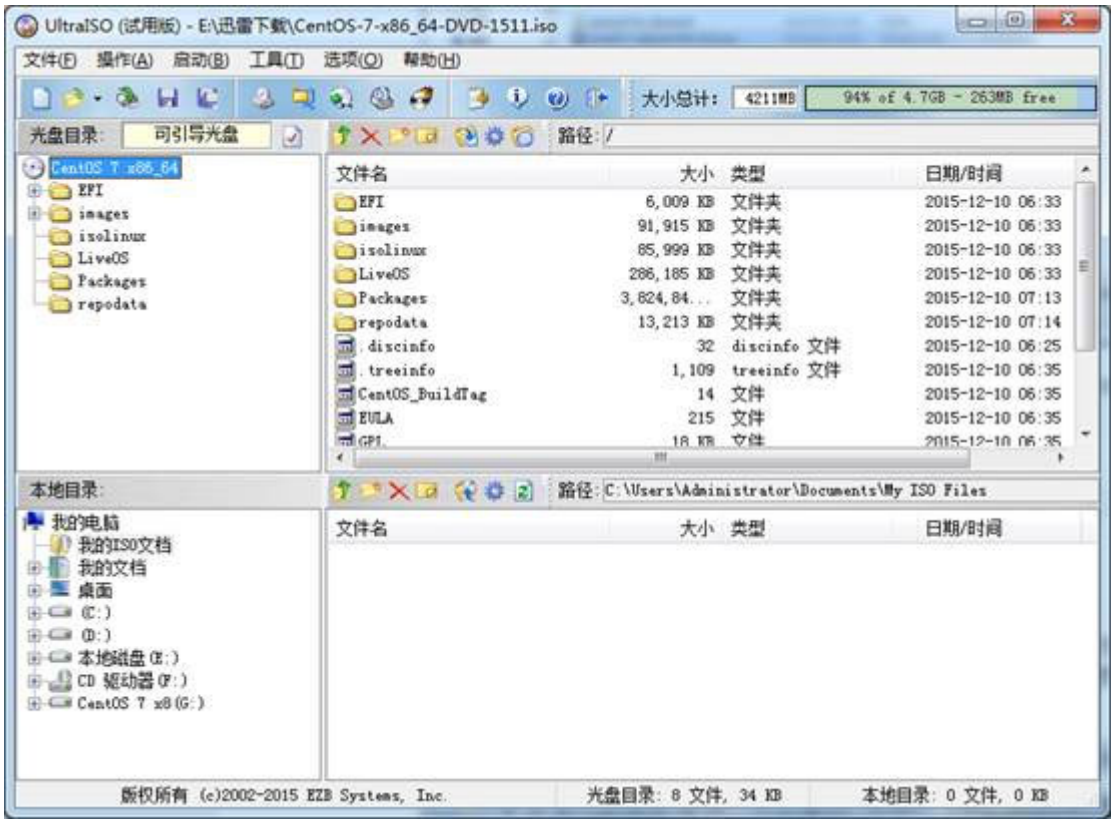
在制作启动盘前，备份 U 盘内的资料，软件在刻录前会对 U 盘进行格式化。

1、打开 UltraISO 软件，点击文件中的打开镜像工具，然后在打开的“打开 ISO 文件”对话框中找到 Centos 镜像文件，之后点右下方的“打开”按钮

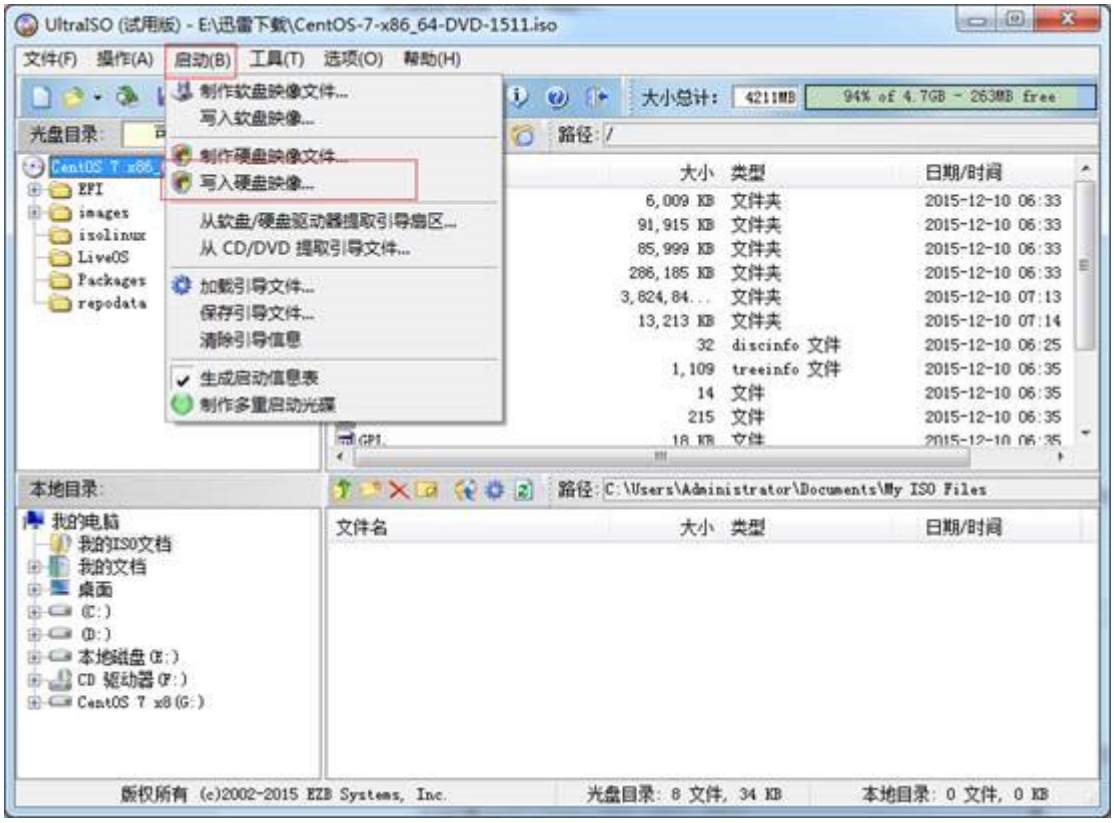




2、打开镜像文件之后，在上方的列表中就会出现对打开的镜像文件的预览左边显示的是具体的目录，右边显示的目录和具体文件

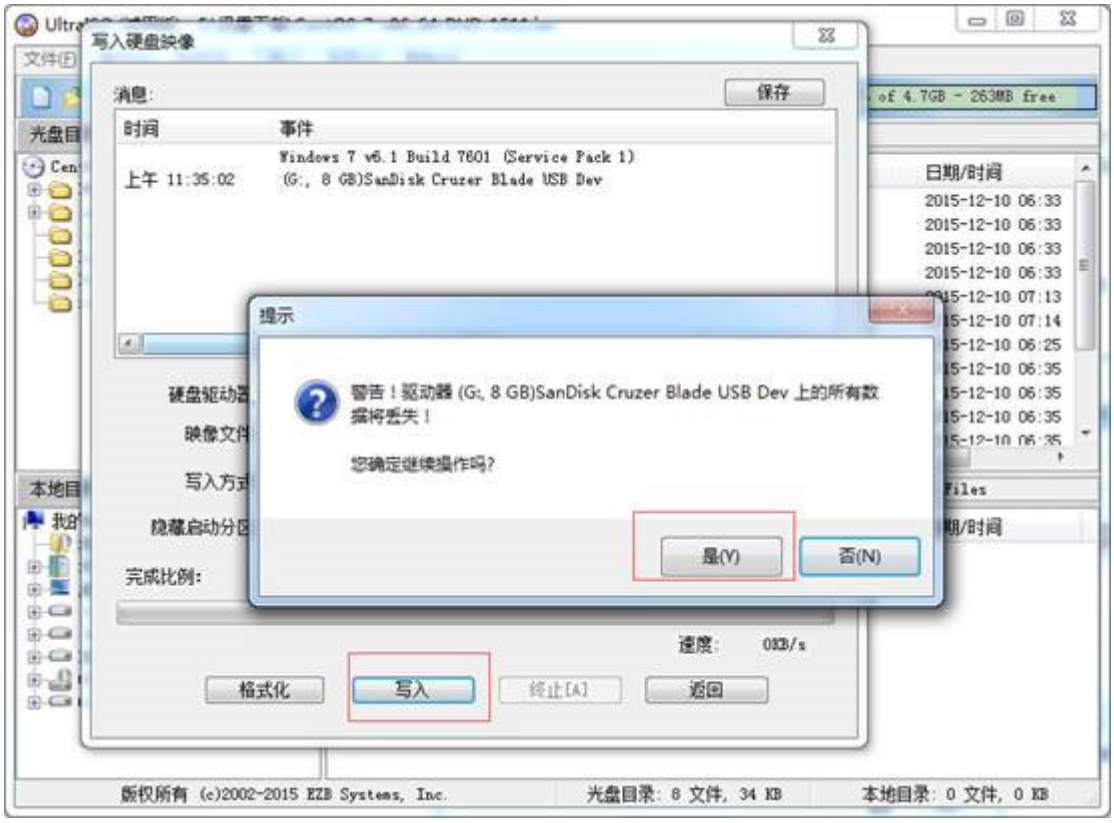


3、下面就开始制作启动盘了，点击菜单栏的“启动”，然后再弹出才按中选择“写入硬盘映像...”，打开“写入硬盘映像”对话框

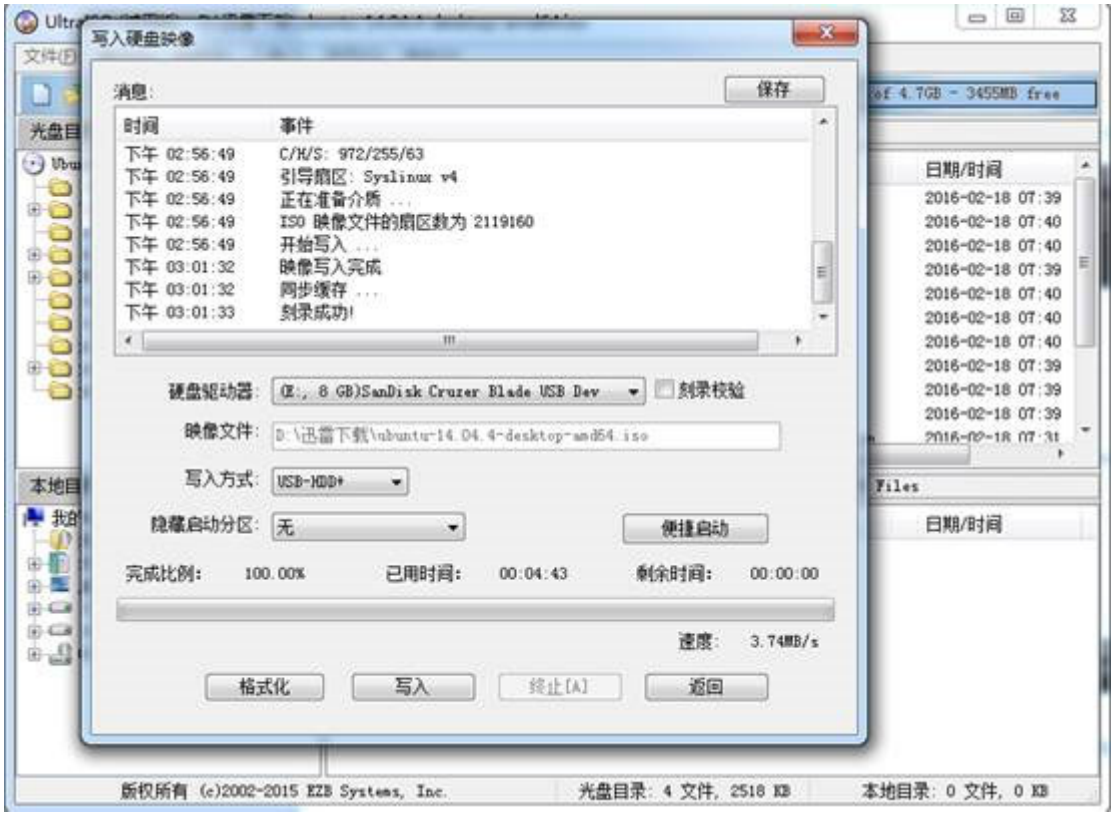


4、在写入硬盘映像对话框中，硬盘驱动器选择我们要写入的 U 盘，写入方式选择选择 USB-HDD+

5、点击下面的“写入”按钮，会弹出警告提示框，点击“是”就开始 U 盘安装盘的写入



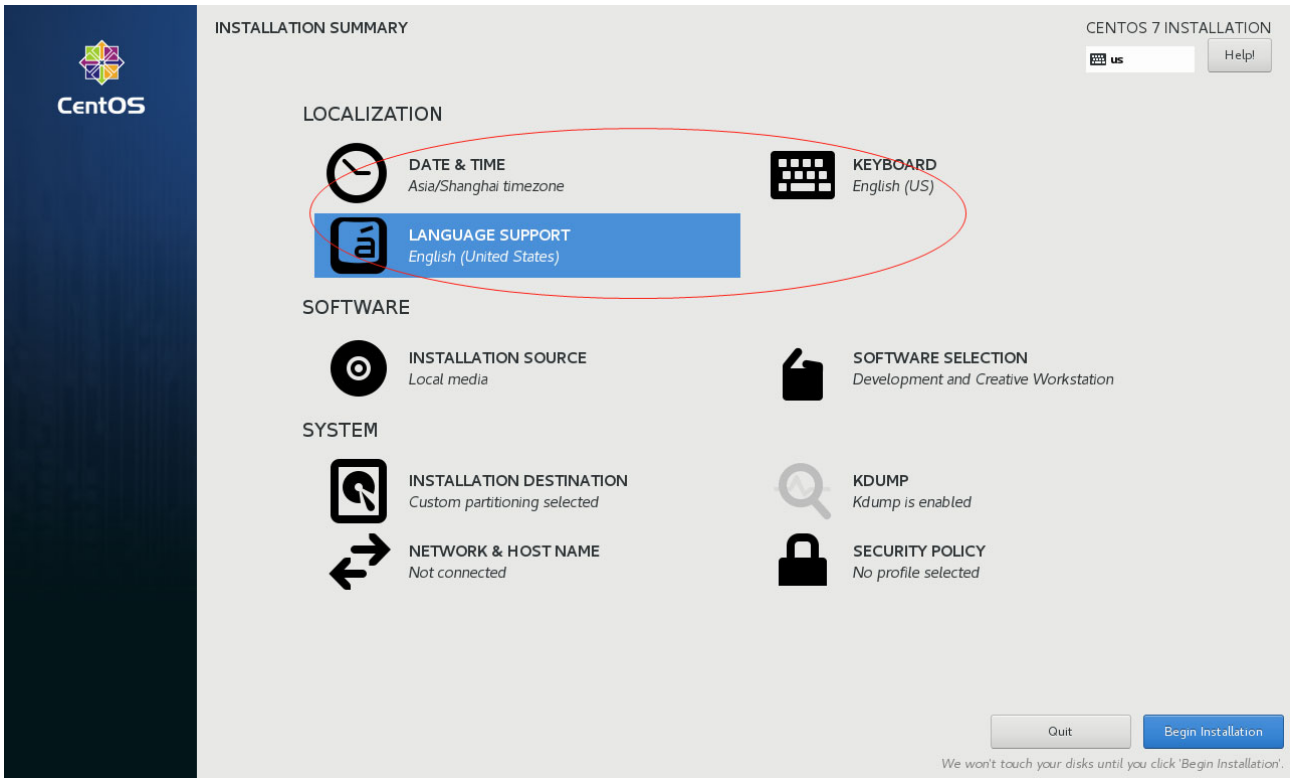
6、等待制作完成，安全弹出 U 盘



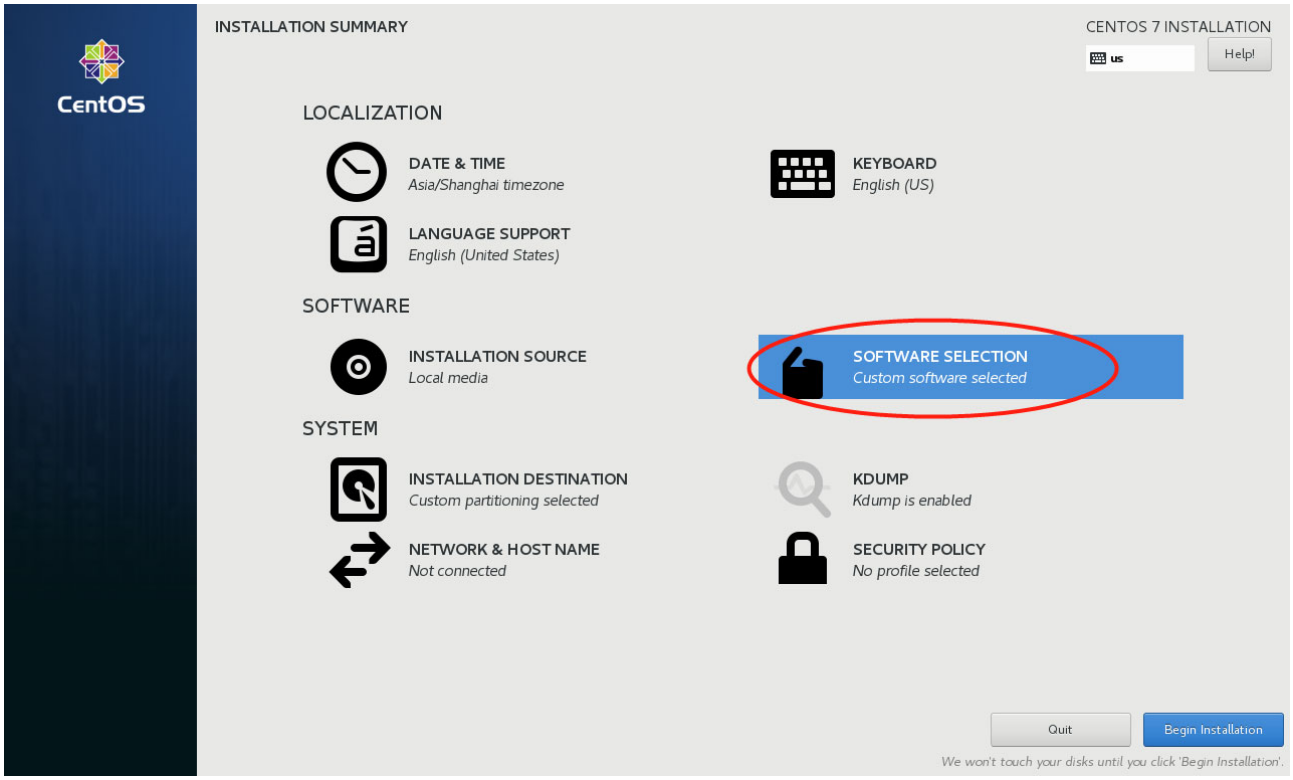
1.2. 安装 Centos

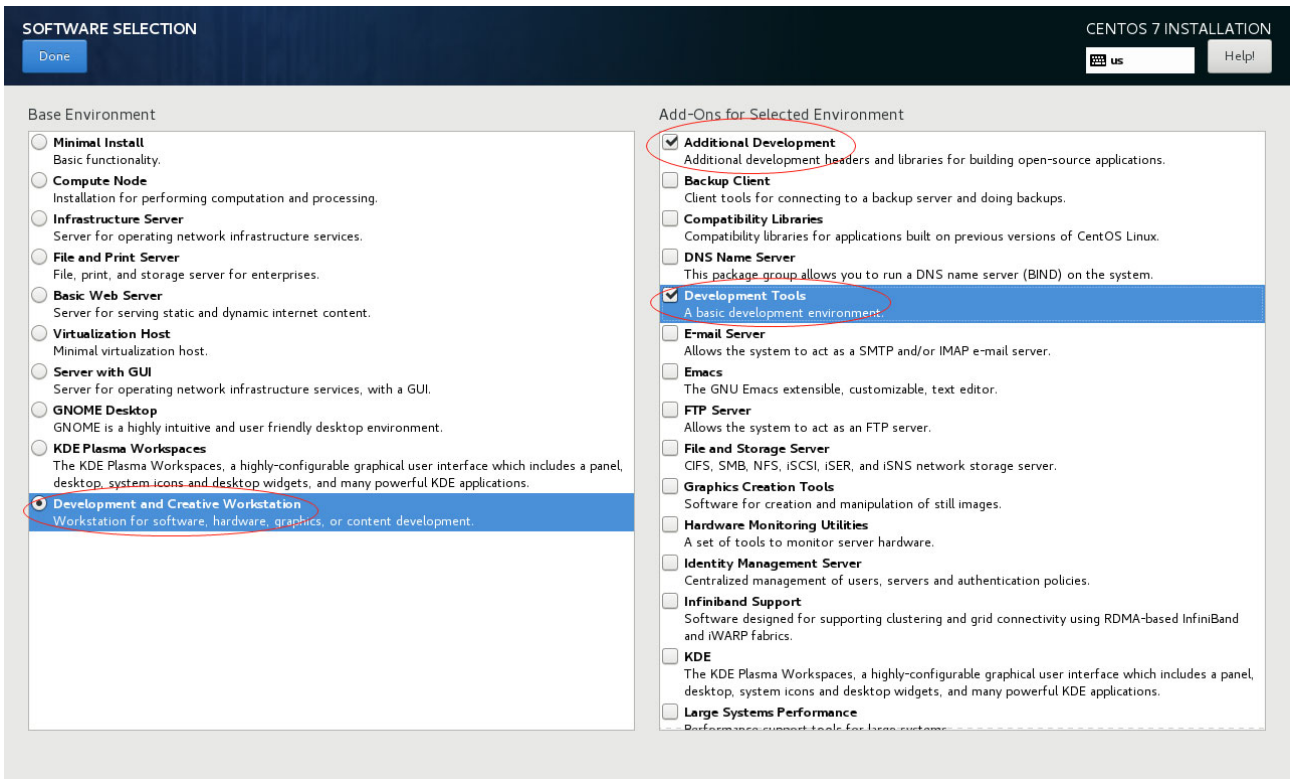
1、插入 U 盘，启动计算机，根据开机提示进入 BIOS 设置启动模式为 legacy，选择从 U 盘启动。稍等片刻即可进入下图所示的 Centos 安装界面。选择 Install Centos 7 图标进入下一界面，安装语言选择默认 English，选择 Continue。

2、选择时区为“Shanghai”,键盘和语言都为“美式英语”。



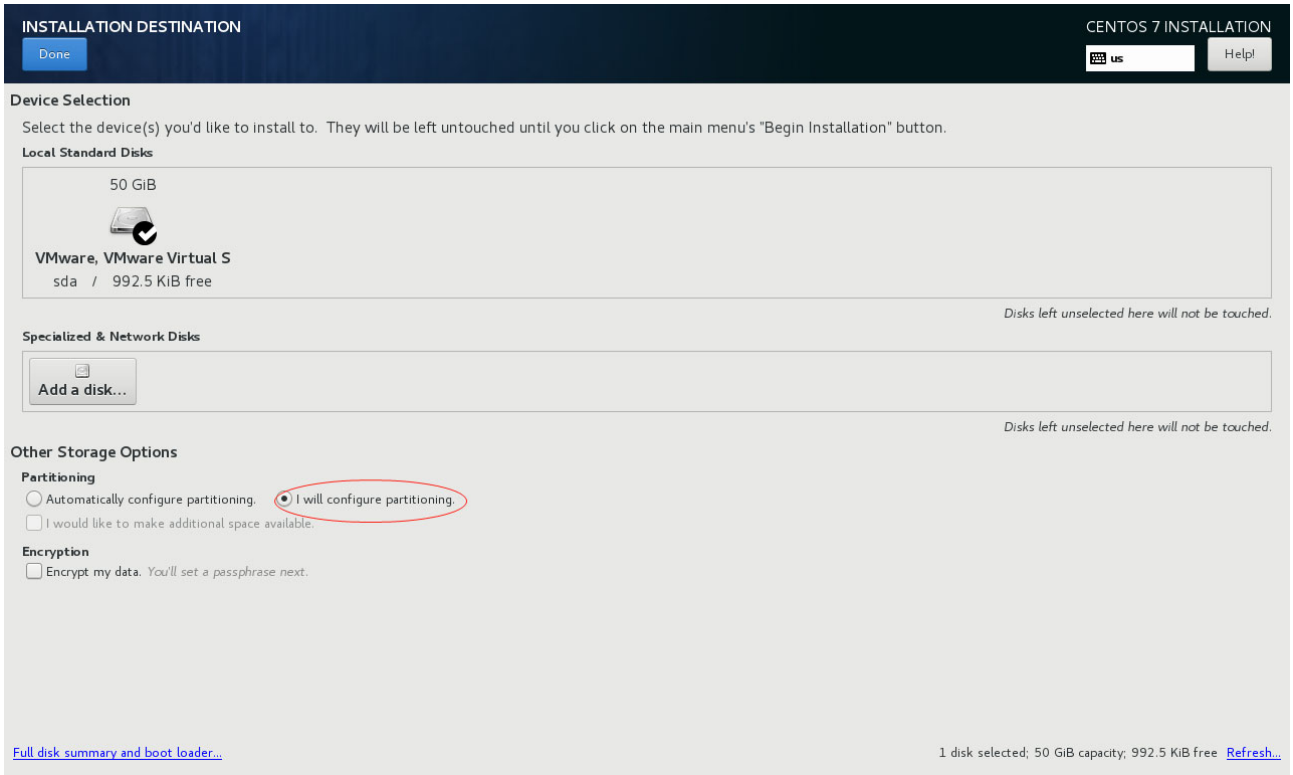
3、选择安装模式。



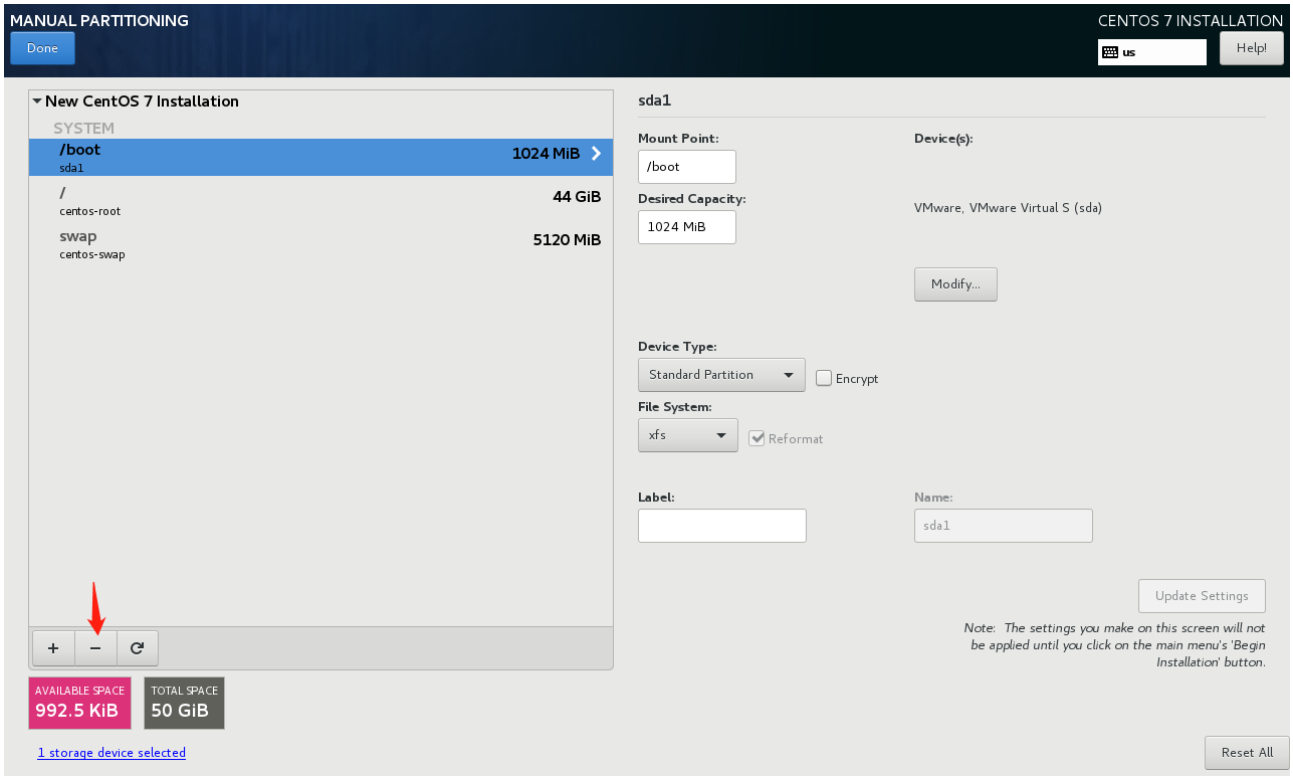


4、选择安装位置（磁盘分区）手动分区。

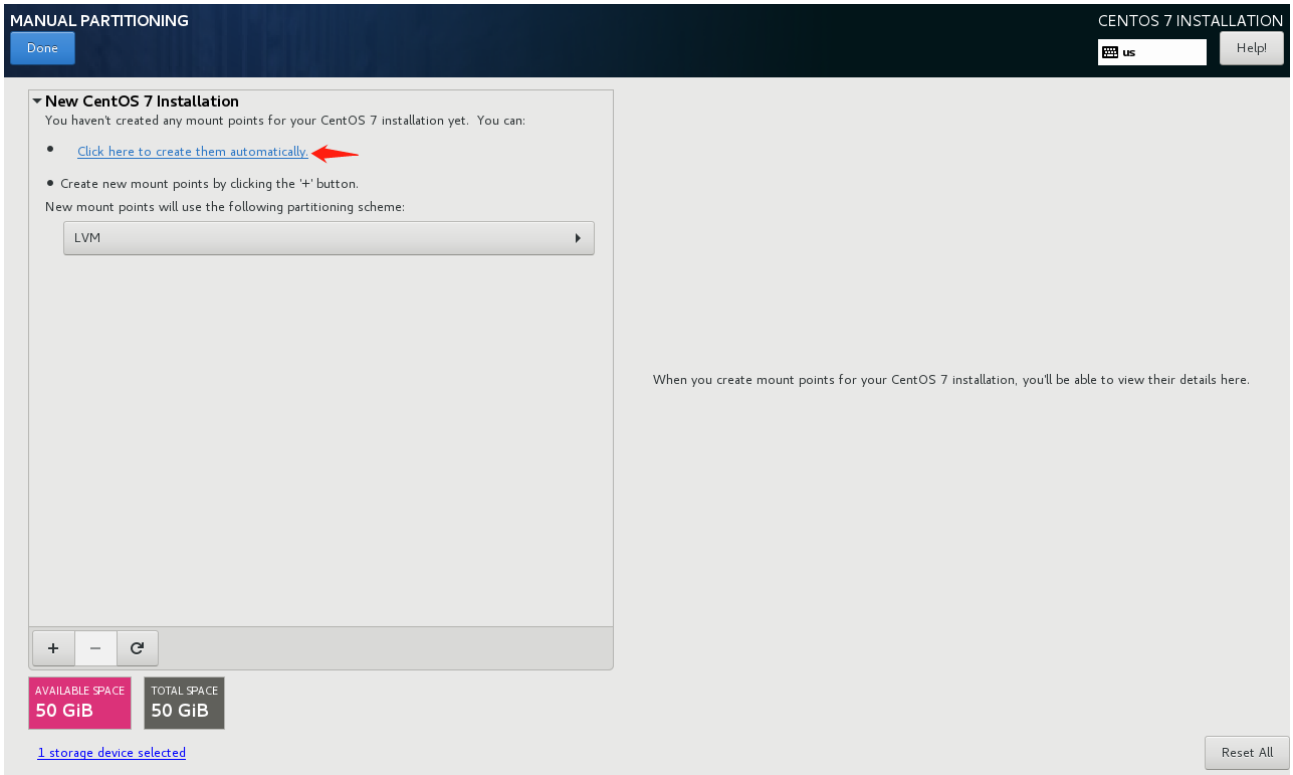
a. 选择“I will configure partitioning”，然后点击左上角的 “Done”按钮，开始分区



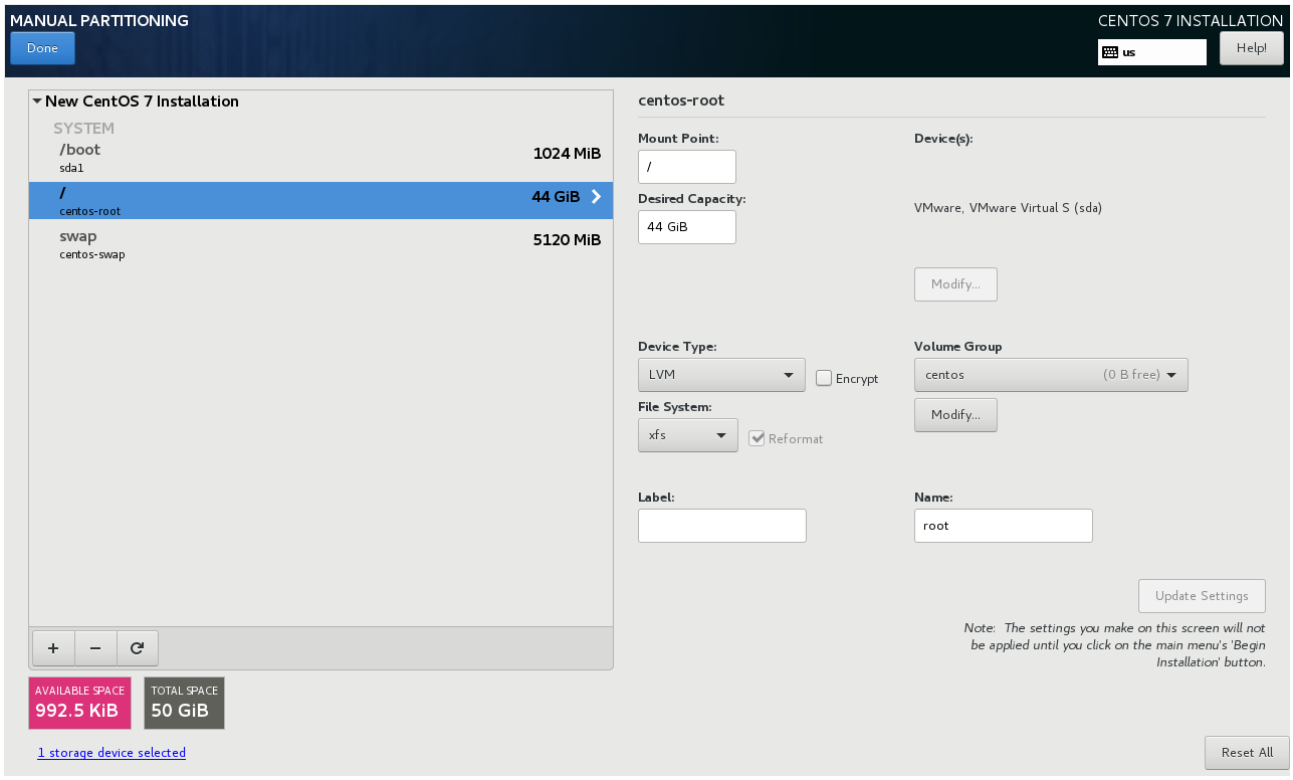
b. 点击已有的文件系统，再点击左下方的“-”。把旧文件系统全部删除。



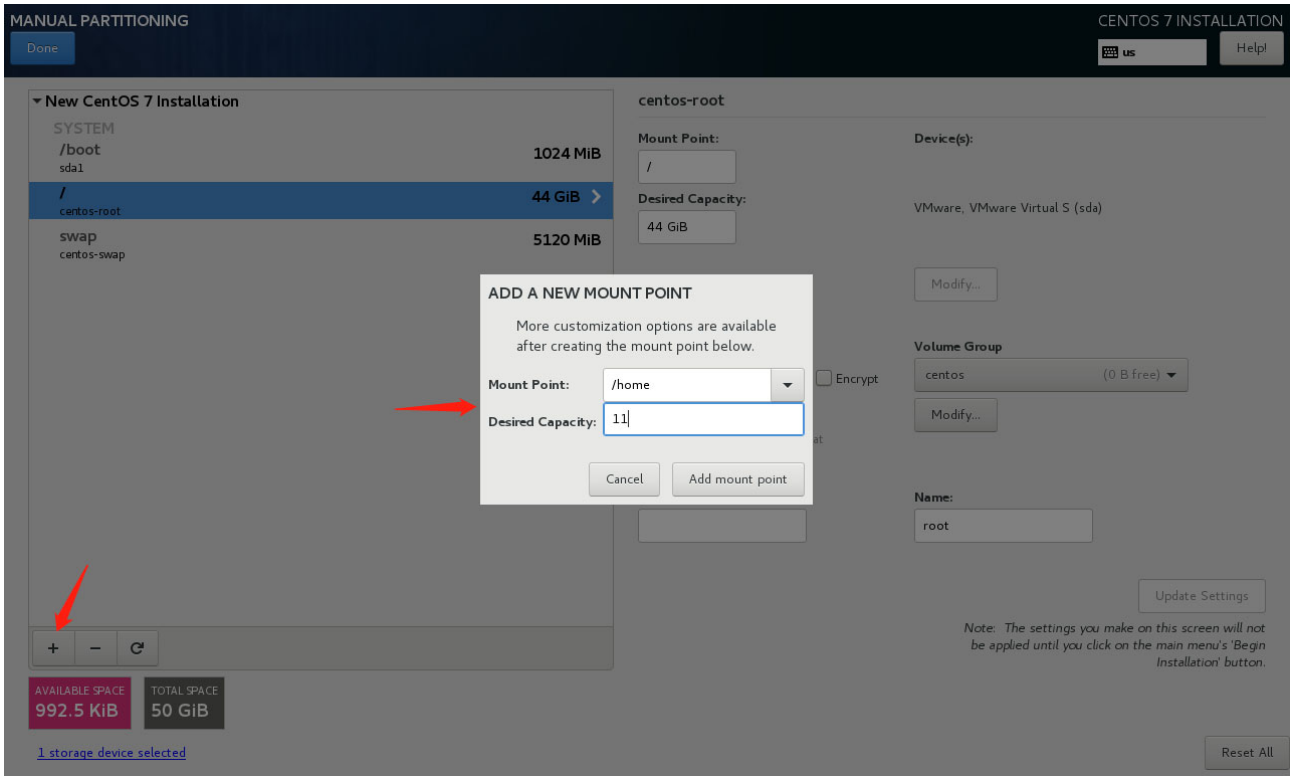
c. 删除成功，显示如下图界面，然后点击下图红色框标定处：



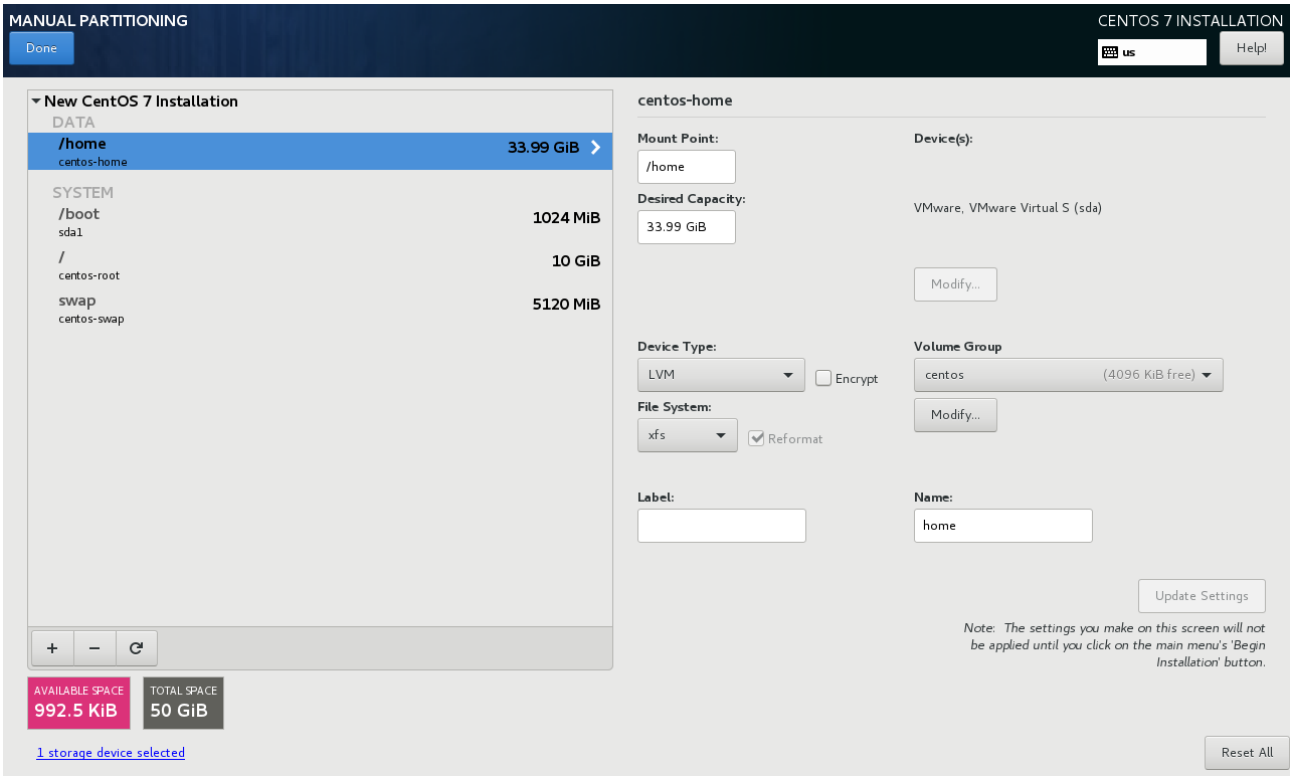
e. 点击后，显示如下：



f. 点击左下方的“+”，新增/home 挂载点

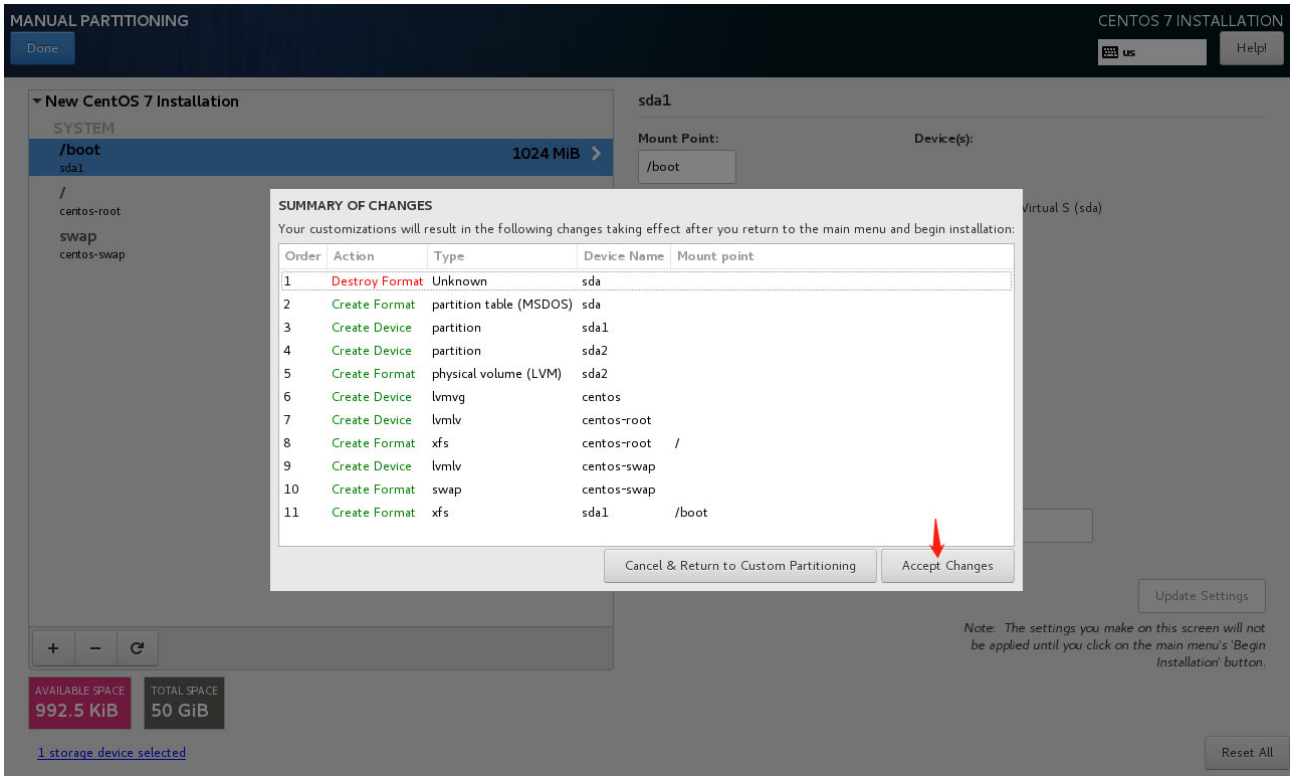


g. 调整根目录和 HOME 目录的大小：建议根目录保留最大的存储空间，将少量的存储空间放到 HOME 目录

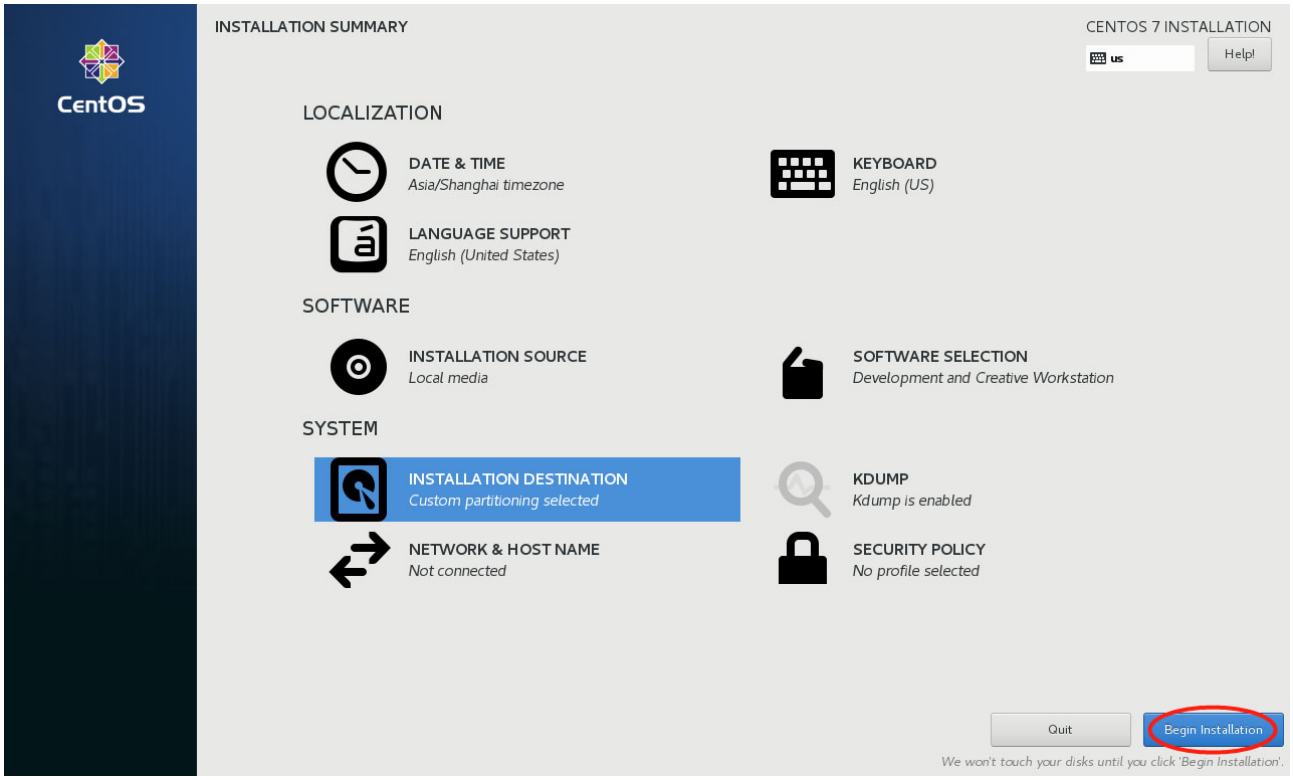


h. 点击上方的“Done”按钮，完成：

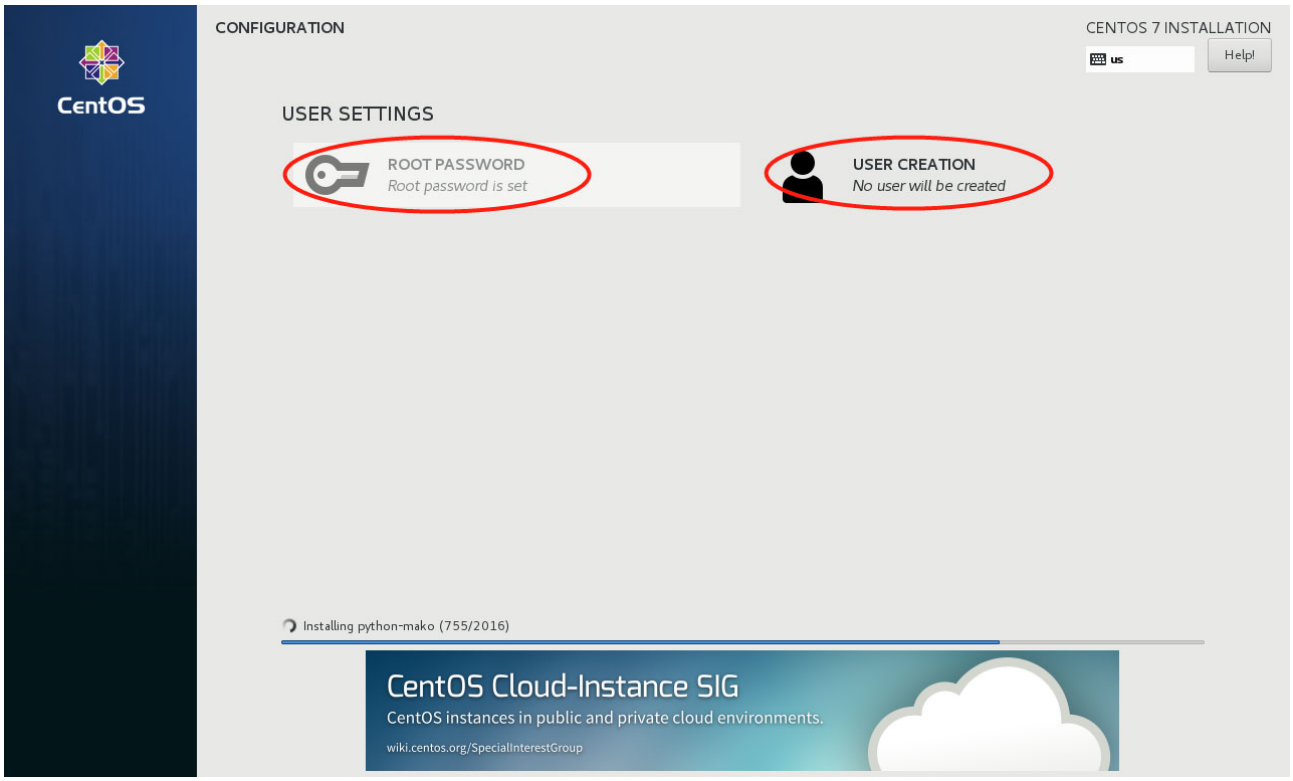
i. 点击“Accept Changes”：



5、配置完成后，点击开始安装。



6、安装时设置 root 密码，如有需求，可以另外创建普通用户。



7、等待安装完成，单击重新启动，重启后接受许可协议，至此系统安装完成。

8、配置固定 ip：进入/etc/sysconfig/network-scripts 目录，找到网口的配置文件，例如：ifcfg-enp0s3


```
[root@centos7 network-scripts]# pwd
/etc/sysconfig/network-scripts
[root@centos7 network-scripts]# ls
ifcfg-enp0s3  ifdown-ppp      ifup-eth     ifup-sit
ifcfg-lo      ifdown-routes   ifup-ippp    ifup-Team
ifdown        ifdown-sit      ifup-ipv6    ifup-TeamPort
ifdown-bnep   ifdown-Team     ifup-isdn    ifup-tunnel
ifdown-eth    ifdown-TeamPort ifup-plip    ifup-wireless
ifdown-ippp   ifdown-tunnel   ifup-plusb   init.ipv6-global
ifdown-ipv6   ifup            ifup-post    network-functions
ifdown-isdn   ifup-aliases    ifup-ppp     network-functions-ipv6
ifdown-post   ifup-bnep       ifup-routes
[root@centos7 network-scripts]#
```

打开配置文件并编辑以下变量：

BOOTPROTO="static" #dhcp 改为 static

ONBOOT="yes" #开机启用本配置

IPADDR=192.168.0.17 #静态 IP

GATEWAY=192.168.0.255 #默认网关

NETMASK=255.255.255.0 #子网掩码

DNS1=192.168.7.1 #DNS 配置

保存修改，然后使用以下命令来重启网络服务：

systemctl restart network.service

现在验证接口是否配置正确：ip add

```
[root@centos7 network-scripts]# systemctl restart network.service
[root@centos7 network-scripts]# ip add
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP
qlen 1000
    link/ether 08:00:27:f3:d4:74 brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.17/24 brd 192.168.0.255 scope global enp0s3
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fef3:d474/64 scope link
       valid_lft forever preferred_lft forever
[root@centos7 network-scripts]# cat /etc/redhat-release
CentOS Linux release 7.0.1406 (Core)
[root@centos7 network-scripts]#
```

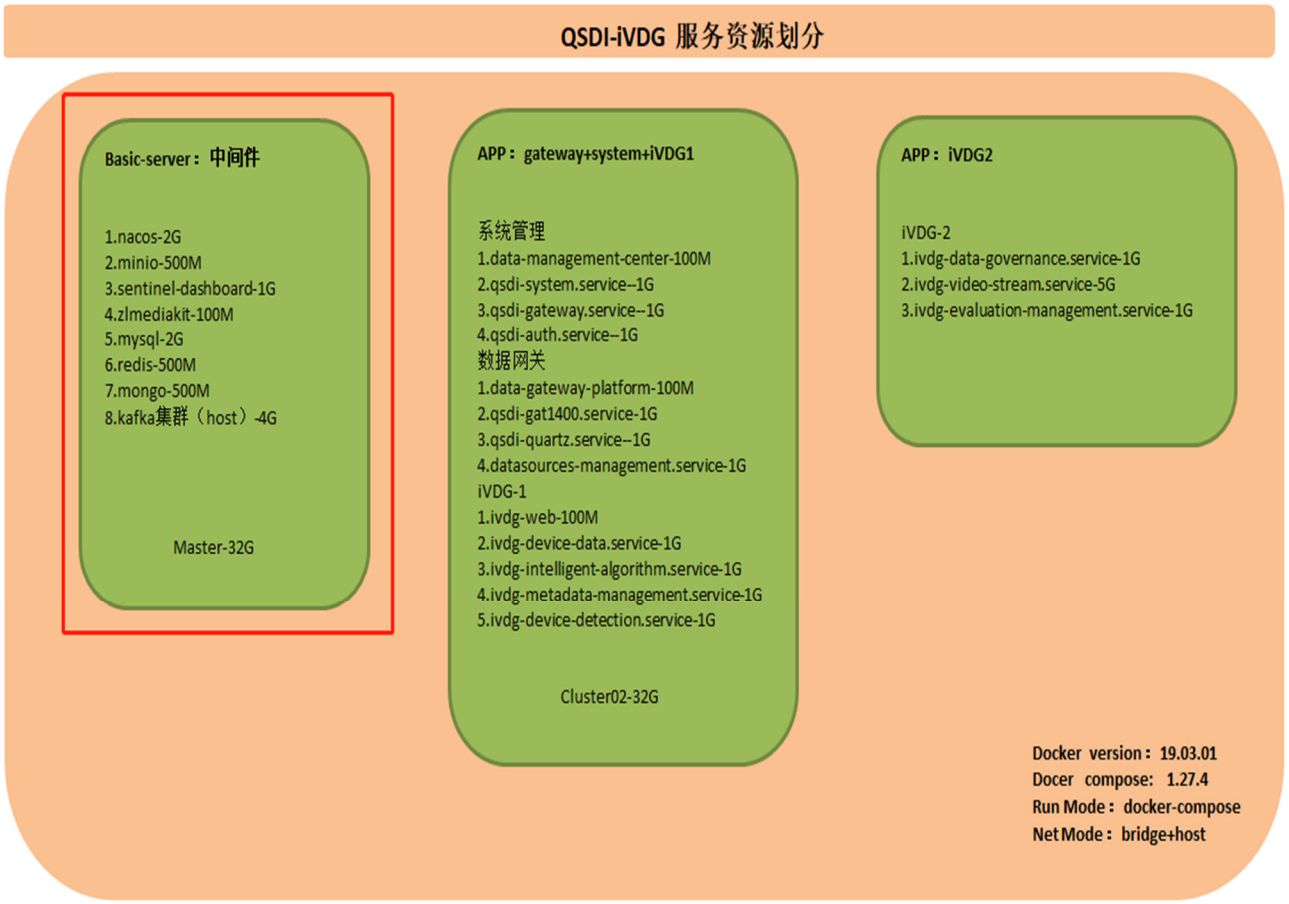
11、同步网络时间

如果系统时间和网络时间不一致，执行以下命令同步网络时间：

ntpdate time.nuri.net

二、 全新部署

2.1. 服务器资源分配说明



2.2. 操作系统环境配置

a) 关闭防火墙：

以下命令可以永久关闭防火墙：

```
[root@localhost qsdi]# systemctl disable firewalld
Removed symlink /etc/systemd/system/multi-user.target.wants/firewalld.service.
Removed symlink /etc/systemd/system/dbus-org.fedoraproject.FirewallD1.service.
```

b) 关闭 SELinux 状态：

- 临时关闭：

```
[root@localhost qsdi]# set enforce 0 #设置 SELinux 成为 permissive 模式
#set enforce 1 设置 SELinux 成为 enforcing 模式
```

- 永久关闭：修改/etc/selinux/config 文件，将 SELINUX=enforcing 改为 SELINUX=disabled，保存之后，重启服务器
- “临时关闭”只是保证当前不通过重启来修改配置，“永久关闭”这一步一定要做

c) 修改 linux 的最大进程数和最大文件打开数

- 临时修改：ulimit -n 204800
- 永久修改：

1.修改 vim /etc/security/limits.conf 文件，在文件末尾添加：

```
* soft nofile 204800
* hard nofile 204800
* soft nproc 204800
* hard nproc 204800
```

2.修改文件 vim /etc/security/limits.d/20-nproc.conf，在文件尾添加：

```
* soft nproc 204800
* hard nproc 204800
```

不需要重启服务器就可以生效

centos6 需修改的文件路径为 /etc/security/limits.d/90-nproc.conf

centos7 需修改的文件路径为 /etc/security/limits.d/20-nproc.conf

2.3. Docker 以及 Docker-compose 安装

a) 将版本包中的 docker-install-offline-tar 目录上传至/home/qsdi/install 目录下

b) 运行 docker-install-offline-tar 中的 qsdi-docker.sh 脚本，脚本会先卸载宿主机上已经安装的 docker，我们要求 docker 版本为 19.03.1，如果宿主机安装的 docker 符合要求可略过此步骤

```
[root@localhost docker-install-offline-tar]# pwd
/home/qsdi/install/docker-install-offline-tar
[root@localhost docker-install-offline-tar]# sh qsdi-docker.sh
输入镜像存储路径:例如/home/docker
说明：默认 Docker 的镜像文件是安装在系统盘根目录，需要重新指定目录，才能存放更多的镜像
Installing docker...
下略
----------docker-compose----------



  Status: installed!
docker-compose version 1.27.4, build 40524192
[root@localhost docker-install-offline-tar]#
```

c) 安装成功后检查命令

```
[root@localhost docker-install-offline-tar]# docker version
Client: Docker Engine - Community
 Version:           19.03.1
 API version:       1.40
 Go version:        go1.12.5
 Git commit:        74b1e89e8a
 Built:             Thu Jul 25 21:17:37 2019
 OS/Arch:           linux/amd64
 Experimental:      false

Server: Docker Engine - Community
 Engine:
  Version:          19.03.1
下略
[root@localhost docker-install-offline-tar]# docker-compose version
docker-compose version 1.27.4, build 40524192
docker-py version: 4.3.1
CPython version: 3.7.7
```


OpenSSL version: OpenSSL 1.1.0l 10 Sep 2019

2.4. 基础服务部署

2.4.1. 中间件服务列表

服务名称	版本号	网络模式	部署方式
kafka 集群： kafka-manager kafka1,kafka2,kafka3 zookeeper	 1.3.1.8 2.6.0 3.4	host	docker-compose
MINIO	RELEASE.2021-04-18T19-26-29Z	bridge	docker-compose
MYSQL-MARIADB	10.6.4	bridge	docker-compose
NACOS	1.4.1	bridge	docker-compose
REDIS	6.2.2	bridge	docker-compose
MONGO	4.4	bridge	docker-compose
SENTINEL-DASHBOARD	18.09.1	bridge	docker-compose
ZLMEDIAKIT	-	bridge	docker-compose
qsdi-xxl-job-admin	2.3.0	bridge	docker-compose

2.4.2. kafka 集群部署

a) 启动前确认 9090,9091,9092,2183,2184,2185 端口是否占用，如占用请更改映射端口

b) 将版本包中的 kafka 目录上传至/home/qsdi/install 目录下

c) 需要根据实际环境修改.env 配置文件

```
[root@localhost kafka]# vi .env
#镜像仓库地址
DOCKER_REPOSITOR=192.168.1.123:81   #保持不变
KAFKA_IP=192.168.206.3
```

d) 加载镜像

```
[root@localhost kafka]# pwd
/home/qsdi/install/kafka
[root@localhost kafka]# sh load_docker_image.sh
search path:/home/qsdi/install/kafka/pkg_image
Find match files:2
/home/qsdi/install/kafka/pkg_image/kafka.tar
/home/qsdi/install/kafka/pkg_image/zookeeper.tar\
下略
```

e) 安装 kafka 集群

```
[root@localhost kafka]# pwd
/home/qsdi/install/kafka
[root@localhost kafka]# docker-compose up -d
Creating network "kafka_default" with the default driver
Creating kafka2 ... done
Creating zoo1      ... done
Creating kafka1 ... done
Creating kafka3 ... done
```

2.4.3. 其他基础服务

a) 执行前确认端口 3306，9002，8080，8848，6379，27017，8858，31080 是否占用

b) 将版本包中的 basic 目录上传至/home/qsdi/install 目录下

c) 需要根据实际环境修改.env 配置文件

```
[root@localhost basic]# vi .env
#镜像仓库地址
 DOCKER_REPOSITOR=192.168.1.123:81   #保持不变
#本地 IP 地址
 LOCAL_IP=192.168.206.3   #根据实际配置修改
#docker 网络模式
 #DOCKER_MODE="host"
 DOCKER_MODE="bridge"
```

d) 加载镜像

```
[root@localhost basic]# pwd
/home/qsdi/install/basic
[root@localhost basic-server]# sh load_image.sh
下略
```

e) 启动所有中间件服务

```
[root@localhost basic]# docker-compose up -d
Creating mysql.basic        ... done
Creating minio.basic        ... done
Creating mongo.basic        ... done
Creating nacos.basic        ... done
Creating redis.basic        ... done
Creating sentinel.basic ... done
Creating zlmediakit.basic    ... done
Creating qsdi-xxl-job-admin.basic    ... done
```

f) 如果需要启动单个服务：

```
#启动单个服务，例如：REDIS                执行命令：docker-compose up -d redis
#启动单个服务，例如：MINIO                执行命令：docker-compose up -d minio
#启动单个服务，例如：NACOS                执行命令：docker-compose up -d nacos
#启动单个服务，例如：MONGO                执行命令：docker-compose up -d mongo
#启动单个服务，例如：SENTINEL             执行命令：docker-compose up -d sentinel
#启动单个服务，例如：MYSQL                执行命令：docker-compose up -d mysql
#启动单个服务，例如：ZLMediaKit           执行命令：docker-compose up -d zlmediakit
#启动单个服务，例如：qsdi-xxl-job-admin 执行命令：docker-compose up -d qsdi-xxl-job-admin
```

2.5. 应用服务部署

2.5.1. 基础服务列表

系统管理	服务名称	版本	端口
1	system-view	1.0.4-RELEASE（以实际发包为准）	81
2	qsdi-system-service	1.0.4-RELEASE（以实际发包为准）	8885
3	qsdi-gateway	1.0.4-RELEASE（以实际发包为准）	8888
4	qsdi-auth	1.0.4-RELEASE（以实际发包为准）	8887
5	qsdi-quartz-service	1.0.4-RELEASE（以实际发包为准）	8897

数据网关	服务名称	版本	端口
1	getway-view	1.0.4-RELEASE（以实际发包为准）	82

2	qsdi-gat1400-service	1.0.4-RELEASE（以实际发包为准）	8878
3	datasources-management-service	1.0.4-RELEASE（以实际发包为准）	8886
4	qsdi-data-sharing-service	1.0.4-RELEASE（以实际发包为准）	8870

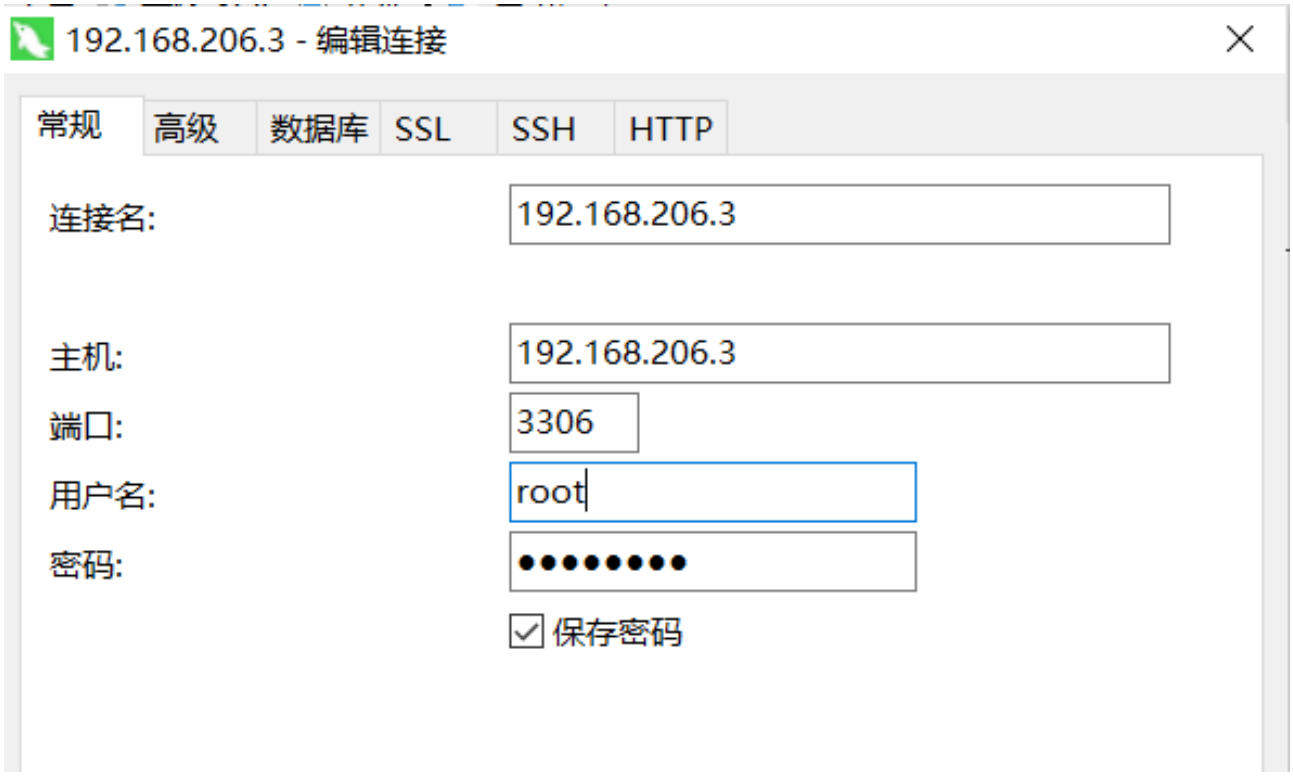
数据治理	服务名称	版本	端口
1	ivdg-view	1.0.4-RELEASE（以实际发包为准）	80
2	ivdg-evaluation-management-service	1.0.4-RELEASE（以实际发包为准）	8810
3	ivdg-device-data-service	1.0.4-RELEASE（以实际发包为准）	8815
4	ivdg-data-governance-service	1.0.4-RELEASE（以实际发包为准）	8806
5	ivdg-intelligent-algorithm-service	1.0.4-RELEASE（以实际发包为准）	8808
6	ivdg-metadata-management-service	1.0.4-RELEASE（以实际发包为准）	8803
7	ivdg-device-detection-service	1.0.4-RELEASE（以实际发包为准）	8820
8	ivdg-asert-center-service	1.0.4-RELEASE（以实际发包为准）	8816
9	ivdg-examination-app	1.0.4-RELEASE（以实际发包为准）	8808
10	ivdg-examination-service	1.0.4-RELEASE（以实际发包为准）	8809
11	ivdg-video-info-service	1.0.4-RELEASE（以实际发包为准）	8818
12	ivdg-gb28181-service	1.0.4-RELEASE（以实际发包为准）	8817
13	ivdg-data-sync-service	1.0.4-RELEASE（以实际发包为准）	8830

：

采集 APP	服务名称	版本	端口
1	qsdi-app-manage-view	1.0.4-RELEASE（以实际发包为准）	91
2	qsdi-app-manage-service	1.0.4-RELEASE（以实际发包为准）	8825
3	qsdi-apph5-view	1.0.4-RELEASE（以实际发包为准）	92
4	ivdg-mobile-server-service	1.0.4-RELEASE（以实际发包为准）	8900

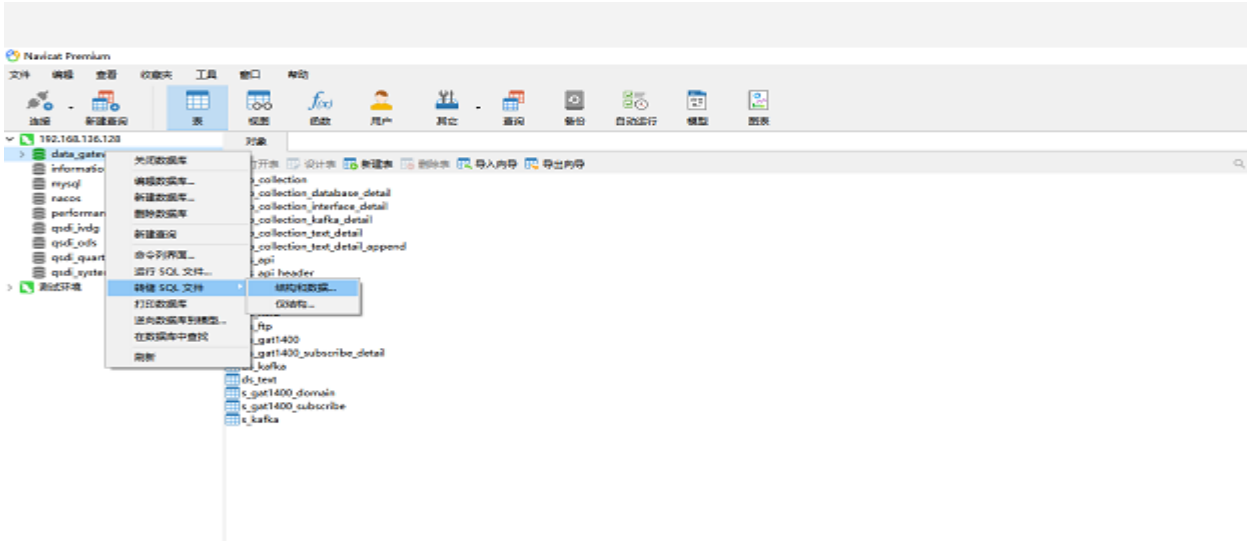
2.5.2. 数据库脚本

a) 使用工具登录 mysql，IP 为 MYSQL 所在服务器 IP，端口为 3306，用户名 root，密码为 qsdi（以上信息是在/basic-server/docker-compose.yml 中配置的）



b) 备份数据库（首次部署可忽略该步骤直接操作 c 步骤）

将 6 个数据库通过 navicat 工具进行备份，操作如下



c) 首次导入全量的数据库脚本

注意事项：
一。注意事项：
1. 执行脚本时，将脚本内容复制出来手动运行，不要用导入的方式执行（会报错或直接略过了报错的脚本）；

2. 执行 update 脚本时，要先选择对应的数据库；
3. 对于 nacos 的脚本，可以首先将脚本中的 192.168.1.110 批量替换为现场基础服务所在主机的 IP，以减少配置工作量

二。V1.0.4 全新部署：
1. 执行 V1.0.4\total 下所有脚本；1.0.1_upgrade_qsdi_system_alert.sql 在 qsdi_system 库中最后执行；
2. 按照时间顺序执行 V1.0.4 目录下的其他脚本

三。从 V1.0.3 升级到 V1.0.4
1. 确认现场最后升级 V1.0.3 版本的日期，然后执行 V1.0.3 目录下未执行的脚本
2. 按照时间顺序执行 V1.0.4 目录下，除 total 之外的其他脚本

2.5.3. Nacos 配置

a) 使用工具登录 mysql，执行\1.0.4\基础服务\basic\nacos.sql 脚本（可以首先将脚本中的 192.168.1.110 批量替换为现场基础服务所在主机的 IP，以减少配置工作量

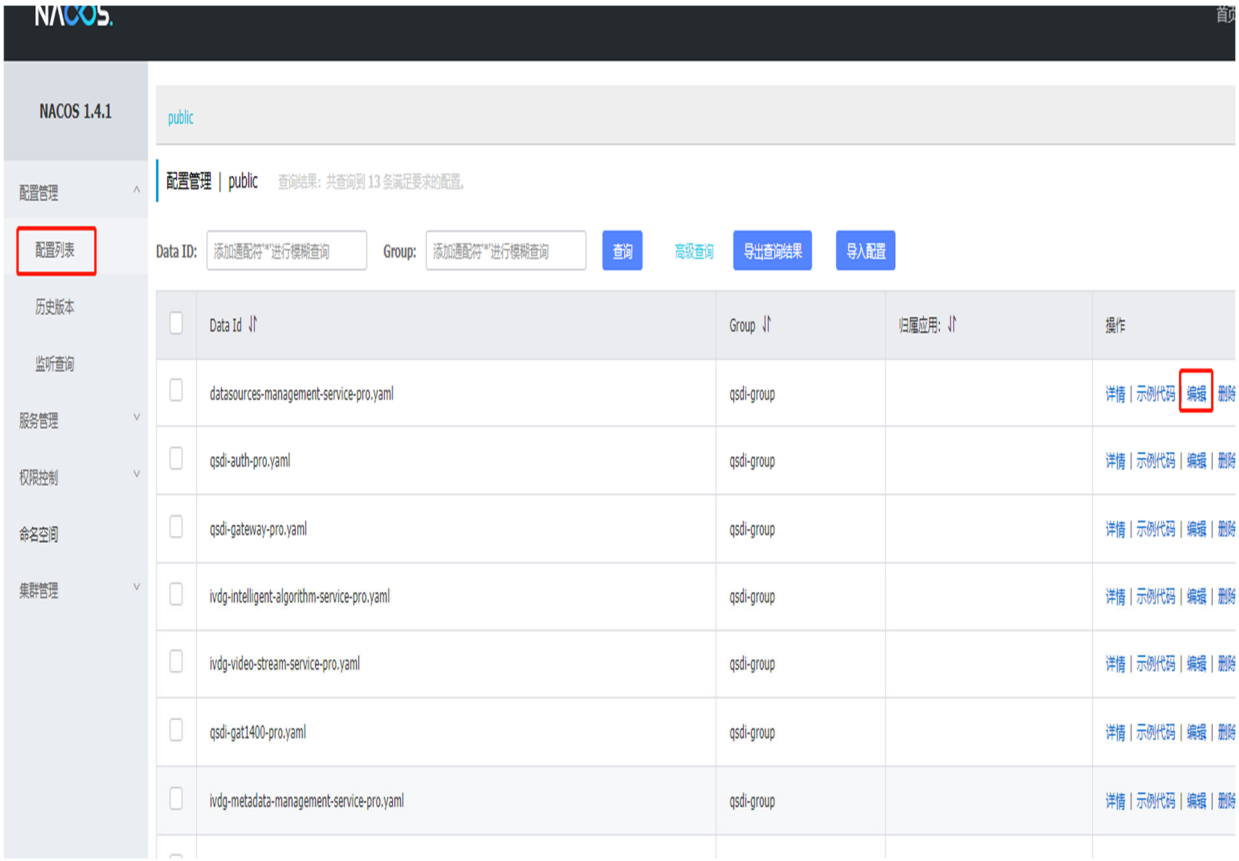
b) 重启 nacos

```
[root@localhost basic]# pwd
/home/qsdi/install/basic
[root@localhost basic]# docker restart nacos.basic
```

c) 访问 nacos 控制台检查：

http://ip:8848/nacos/index.html，默认用户名/密码：nacos/nacos

如下：配置管理-配置列表，会加载 nacos.config_info 表初始化数据



d) 在 nacos 上修改每个应用服务的配置，主要是：host、数据库 ip、dashboard、bootstrao-servers(ip:port1,ip:port2,ip:port3)等。一定要注意 yml 文件的缩进。版本包的 nacos-config 目录下提供了样例文档，供参考。

e) 以下样例文件中需要修改的地方已经用黄色标注出。

2.5.3.1. 系统管理相关

a) qsdi-auth-pro.yaml

```yaml
server:
  port: 8887

spring:
  application:
    name: qsdi-auth
  redis: # spring.factories
    database: 10
    host: 192.168.1.110
```

```
    port: 6379
    password: qishudi

  datasource:
    url:
jdbc:mysql://192.168.1.110:3306/qsdi_system?useUnicode=true&characterEncoding=UTF-8
&useSSL=false&autoReconnect=true&serverTimezone=Asia/Shanghai
    username: root
    password:  qsdi
    driver-class-name: com.mysql.cj.jdbc.Driver

  cloud:
    sentinel:
      transport:
        dashboard: 192.168.1.110:8858
      eager: true
```

b) qsdi-gateway-pro.yaml

```
# Tomcat
server:
  port: 8888
  servlet:
    context-path: /
spring:
  application:
    name: qsdi-gateway
  redis: # spring.factories
    database: 10
    host: 192.168.1.110
    port: 6379
    password: qishudi
    pool:
      maxIdle: 10
      maxTotal: 30
      minEvictableIdleTimeMillis: 30000
      minIdle: 5
      testOnBorrow: true
      timeBetweenEvictionRunsMillis: 30000
  cloud:
    sentinel:
```

```
    transport:
      dashboard: 192.168.1.110:8858
    eager: true
  datasource:
    type: com.alibaba.druid.pool.DruidDataSource
    druid:
      driver-class-name: com.mysql.cj.jdbc.Driver
      url:
jdbc:mysql://192.168.1.110:3306/qsdi_ivdg?allowMultiQueries=true&useUnicode=true&ch
aracterEncoding=UTF-8&useSSL=false&serverTimezone=Asia/Shanghai
      username: root
      password: qsdi
      initial-size: 10
      max-active: 100
      min-idle: 10
      max-wait: 60000
      pool-prepared-statements: true
      max-pool-prepared-statement-per-connection-size: 20
      time-between-eviction-runs-millis: 60000
      min-evictable-idle-time-millis: 300000
      #Oracle 需要打开注释
      #validation-query: SELECT 1 FROM DUAL
      test-while-idle: true
      test-on-borrow: false
      test-on-return: false
      stat-view-servlet:
        enabled: true
        url-pattern: /druid/*
        #login-username: admin
        #login-password: admin
      filter:
        stat:
          log-slow-sql: true
          slow-sql-millis: 1000
          merge-sql: false
        wall:
          config:
            multi-statement-allow: true
```

c) qsdi-system-service-pro.yaml

```yaml
spring:
  redis: # spring.factories
    database: 1
    host: 192.168.1.110
    port: 6379
    password: qishudi

  datasource:
    type: com.alibaba.druid.pool.DruidDataSource
    druid:
      driver-class-name: com.mysql.cj.jdbc.Driver
      url:
jdbc:mysql://192.168.1.110:3306/qsdi_system?allowMultiQueries=true&useUnicode=true&
characterEncoding=UTF-8&useSSL=false&serverTimezone=Asia/Shanghai
      username: root
      password: qsdi
      initial-size: 10
      max-active: 100
      min-idle: 10
      max-wait: 60000
      pool-prepared-statements: true
      max-pool-prepared-statement-per-connection-size: 20
      time-between-eviction-runs-millis: 60000
      min-evictable-idle-time-millis: 300000
      #Oracle 需要打开注释
      #validation-query: SELECT 1 FROM DUAL
      test-while-idle: true
      test-on-borrow: false
      test-on-return: false
      stat-view-servlet:
        enabled: true
        url-pattern: /druid/*
        #login-username: admin
        #login-password: admin
      filter:
        stat:
          log-slow-sql: true
          slow-sql-millis: 1000
          merge-sql: false
        wall:
          config:
```

```
          multi-statement-allow: true

  cloud:
    sentinel:
      transport:
        dashboard: http://192.168.1.110:8858
      eager: true

# Minio 配置
minio:
  url: http://192.168.1.110:9002
  accessKey: qsdi@2021
  secretKey: qsdi@2021
  bucketName: qsdi

security:
  oauth2:
    ignore:
      urls:
        - /token/**
        - /system/region/queryForList
        - /user/orgsTree
        - /system/org/orgsTree
        - /system/org/orgsTreeNew
    client:
      client-id: ENC(sAx6e/5ULrJy+rJ738HL8g==)
      client-secret: ENC(sAx6e/5ULrJy+rJ738HL8g==)
      scope: server
    resource:
      id: qsdi-system
      token-info-uri: http://qsdi-auth/oauth/check_token
      prefer-token-info: true
```

d) qsdi-quartz-pro.yaml

```
spring:
  jackson:
    default-property-inclusion: non_null
  main:
    allow-bean-definition-overriding: true
  datasource:
```



```
    type: com.alibaba.druid.pool.DruidDataSource
    druid:
      driver-class-name: com.mysql.cj.jdbc.Driver
      url:
jdbc:mysql://192.168.1.110:3306/qsdi_quartz?allowMultiQueries=true&useUnicode=true&
characterEncoding=UTF-8&useSSL=false&serverTimezone=Asia/Shanghai
      username: root
      password: qsdi
      initial-size: 10
      max-active: 100
      min-idle: 10
      max-wait: 60000
      pool-prepared-statements: true
      max-pool-prepared-statement-per-connection-size: 20
      time-between-eviction-runs-millis: 60000
      min-evictable-idle-time-millis: 300000
      #Oracle 需要打开注释
      #validation-query: SELECT 1 FROM DUAL
      test-while-idle: true
      test-on-borrow: false
      test-on-return: false
      stat-view-servlet:
        enabled: true
        url-pattern: /druid/*
        #login-username: admin
        #login-password: admin
      filter:
        stat:
          log-slow-sql: true
          slow-sql-millis: 1000
          merge-sql: false
        wall:
          config:
            multi-statement-allow: true

  redis: # spring.factories
    database: 1
    host: 192.168.1.110
    port: 6379
    password: qishudi
    pool:
      maxIdle: 10
      maxTotal: 30
      minEvictableIdleTimeMillis: 30000
```



```
    minIdle: 5
    testOnBorrow: true
    timeBetweenEvictionRunsMillis: 30000
  cloud:
    sentinel:
      transport:
        dashboard: http://192.168.1.110:8858
      eager: true

  data:
    mongodb:
      database: qsdi
      uri: mongodb://root:qsdi@192.168.1.110:27017/?authSource=admin
```

2.5.3.2. 数据网关相关

a) datasources-management-service-pro.yaml

```
spring:
  kafka:
    bootstrap-servers: 192.168.1.110:9090,192.168.1.110:9091,192.168.1.110:9092
    producer:
      acks: 1
      # 重试次数
      retries: 0
      # 批量发送的消息数量
      batch-size: 16384
      # 32MB 的批处理缓冲区
      buffer-memory: 33554432
      client-id: datasource-management
  ods:
    jdbc:
      driver-class-name: com.mysql.cj.jdbc.Driver
      url:
jdbc:mysql://192.168.1.110:3306/qsdi_ods?serverTimezone=Asia/Shanghai&allowMultiQue
ries=true&useUnicode=true&characterEncoding=UTF-8&useSSL=false
      username: root
      password: qsdi
      initialSize: 10
```



```
      maxActive: 30
      minIdle: 10
      maxWait: 60000
    host: 192.168.1.110
  mongodb:
    username: ods
    password: qsdi
    database: ods
    host: 192.168.1.110
    port: 27017
  datax:
    host: 192.168.1.110
    username: root
    password: root
  ftp:
    host: 192.168.1.110
    port: 21
    username: root
    password: qsdi
  redis:
    host: 192.168.1.110
    port: 6379
    password: qishudi
    client-name: datasource-management
    database: 0
  jackson:
    default-property-inclusion: non_null
  main:
    allow-bean-definition-overriding: true
  datasource:
    type: com.alibaba.druid.pool.DruidDataSource
    druid:
      driver-class-name: com.mysql.cj.jdbc.Driver
      url:
jdbc:mysql://192.168.1.110:3306/data_gateway?serverTimezone=Asia/Shanghai&allowMult
iQueries=true&useUnicode=true&characterEncoding=UTF-8&useSSL=false
      username: root
      password: qsdi
      initial-size: 10
      max-active: 30
      min-idle: 10
      max-wait: 60000
      pool-prepared-statements: true
      max-pool-prepared-statement-per-connection-size: 20
```



```
      time-between-eviction-runs-millis: 60000
      min-evictable-idle-time-millis: 300000
      #Oracle 需要打开注释
      #validation-query: SELECT 1 FROM DUAL
      test-while-idle: true
      test-on-borrow: false
      test-on-return: false
      stat-view-servlet:
        enabled: true
        url-pattern: /druid/*
        #login-username: admin
        #login-password: admin
      filter:
        stat:
          log-slow-sql: true
          slow-sql-millis: 1000
          merge-sql: false
        wall:
          config:
            multi-statement-allow: true
  #  profiles:
  #    active: dev
  cloud:
    sentinel:
      transport:
        dashboard: 192.168.1.110:8858
      eager: true
feign:
  sentinel:
    enabled: true
#mybatis
mybatis-plus:
  type-enums-package: com.qsdi.datasources.api.enumeration
  mapper-locations: classpath*:/mapper/**/*.xml
  #实体扫描，多个 package 用逗号或者分号分隔
  typeAliasesPackage: com.qsdi.datasources.api.entity
  global-config:
    #数据库相关配置
    db-config:
      #主键类型  AUTO:"数据库 ID 自增", INPUT:"用户输入 ID", ID_WORKER:"全局唯一 ID (数字类型唯一 ID)", UUID:"全局唯一 ID UUID";
      id-type: INPUT
      #字段策略 IGNORED:"忽略判断",NOT_NULL:"非 NULL 判断"),NOT_EMPTY:"非空判断"
      field-strategy: NOT_NULL
```



```
      #驼峰下划线转换
      column-underline: true
      logic-delete-value: -1
      logic-not-delete-value: 0
    banner: false
  #原生配置
  configuration:
    #    log-impl: org.apache.ibatis.logging.stdout.StdOutImpl
    default-enum-type-handler: org.apache.ibatis.type.EnumOrdinalTypeHandler
    map-underscore-to-camel-case: true
    cache-enabled: false
    call-setters-on-nulls: true
    jdbc-type-for-null: 'null'
    # 是否将 sql 打印到控制面板(该配置会将 sql 语句和查询的结果都打印到控制台)
    # log-impl: org.apache.ibatis.logging.stdout.StdOutImpl
```

b) qsdi-gat1400-pro.yaml

```
spring:
  quartz:
    jdbc:
      driverClassName: com.mysql.cj.jdbc.Driver
      url:
jdbc:mysql://192.168.1.110:3306/qsdi_quartz?serverTimezone=Asia/Shanghai&allowMulti
Queries=true&useUnicode=true&characterEncoding=UTF-8&useSSL=false
      username: root
      password: qsdi
    scheduler:
      instanceName: qsdi-gat1400-scheduler
  kafka:
    bootstrap-servers: 192.168.1.110:9090,192.168.1.110:9091,192.168.1.110:9092
    producer:
      acks: 1
      # 重试次数
      retries: 0
      # 批量发送的消息数量
      batch-size: 2097152
      # 32MB 的批处理缓冲区
      buffer-memory: 335544320
      client-id: datasource-management
  ods:
```



```
    jdbc:
      driver-class-name: com.mysql.cj.jdbc.Driver
      url:
jdbc:mysql://192.168.1.110:3306/qsdi_ods?serverTimezone=Asia/Shanghai&allowMultiQue
ries=true&useUnicode=true&characterEncoding=UTF-8&useSSL=false
      username: root
      password: qsdi
      initialSize: 10
      maxActive: 30
      minIdle: 10
      maxWait: 60000
  redis:
    host: 192.168.1.110
    port: 6379
    password: qishudi
    client-name: datasource-management
    database: 0
  jackson:
    default-property-inclusion: non_null
  main:
    allow-bean-definition-overriding: true
  #  profiles:
  #    active: dev
  cloud:
    sentinel:
      transport:
        dashboard: 192.168.1.110:8858
      eager: true
feign:
  sentinel:
    enabled: true
digest:
  key:
AAAAB3NzaC1yc2EAAAADAQABAAABgQCziIXKqsS+0HoVZLvTq3Eg9f6vruCZ5yBvQkrk3qgjLM5+AukwTDA
SArS++4/iE1zCd9uXnQnnQxPc7rywfMVmb7/HaOc04MC
```

c) qsdi-data-sharing-pro.yaml

```
spring:
  quartz:
    jdbc:
```

```
    driverClassName: com.mysql.cj.jdbc.Driver
    url:
jdbc:mysql://192.168.1.110:3306/qsdi_quartz?serverTimezone=Asia/Shanghai&allowMulti
Queries=true&useUnicode=true&characterEncoding=UTF-8&useSSL=false
    username: root
    password: qsdi
  scheduler:
    instanceName: qsdi-data-sharing
 kafka:
  bootstrap-servers: 192.168.1.110:9090,192.168.1.110:9091,192.168.1.110:9092
  producer:
    acks: 1
    # 重试次数
    retries: 0
    # 批量发送的消息数量
    batch-size: 16384
    # 32MB 的批处理缓冲区
    buffer-memory: 33554432
    client-id: datasource-management
 mongodb:
  username: ods
  password: qsdi
  database: ods
  host: 192.168.1.110
  port: 27017
 #hive: #hive 数据源
 #  url: jdbc:hive2://192.168.1.120:21050/ods;auth=noSasl
 #  jdbcUrl: jdbc:hive2://192.168.1.120:10000/ods
 #  username: hive
 #  password: hive
 #  driver-class-name: org.apache.hive.jdbc.HiveDriver
 ods:
  jdbc:
    driver-class-name: com.mysql.cj.jdbc.Driver
    url:
jdbc:mysql://192.168.1.110:3306/qsdi_ods?serverTimezone=Asia/Shanghai&allowMultiQue
ries=true&useUnicode=true&characterEncoding=UTF-8&useSSL=false&serverTimezone=Asia/
Shanghai
    username: root
    password: qsdi
    initialSize: 10
    maxActive: 30
    minIdle: 10
    maxWait: 60000
```



```
  ftp:
    host: 192.168.1.110
    port: 21
    username: root
    password: qsdi
  redis:
    host: 192.168.1.110
    port: 6379
    password: qishudi
    client-name: datasource-management
    database: 0
  jackson:
    default-property-inclusion: non_null
  main:
    allow-bean-definition-overriding: true
  #  profiles:
  #    active: dev
  cloud:
    sentinel:
      transport:
        dashboard: 192.168.1.110:8858
      eager: true
feign:
  sentinel:
    enabled: true
```

2.5.3.3. 数据治理相关

a) ivdg-intelligent-algorithm-service-pro.yaml

```
spring:
  application:
    name: ivdg-intelligent-algorithm-service
  datasource:
    type: com.alibaba.druid.pool.DruidDataSource
    druid:
      driver-class-name: com.mysql.cj.jdbc.Driver
      url:
jdbc:mysql://192.168.1.110:3306/qsdi_ivdg?allowMultiQueries=true&useUnicode=true&ch
aracterEncoding=UTF-8&useSSL=false&serverTimezone=Asia/Shanghai
      username: root
```



```
    password: qsdi
    initial-size: 10
    max-active: 100
    min-idle: 10
    max-wait: 60000
    pool-prepared-statements: true
    max-pool-prepared-statement-per-connection-size: 20
    time-between-eviction-runs-millis: 60000
    min-evictable-idle-time-millis: 300000
    #Oracle 需要打开注释
    #validation-query: SELECT 1 FROM DUAL
    test-while-idle: true
    test-on-borrow: false
    test-on-return: false
    stat-view-servlet:
      enabled: true
      url-pattern: /druid/*
      #login-username: admin
      #login-password: admin
    filter:
      stat:
        log-slow-sql: true
        slow-sql-millis: 1000
        merge-sql: false
      wall:
        config:
          multi-statement-allow: true

cloud:
  sentinel:
    transport:
      dashboard: http://192.168.1.110:8858
    eager: true
redis: # spring.factories
  database: 0
  host: 192.168.1.110
  port: 6379
  password: qishudi
  pool:
    maxIdle: 10
    maxTotal: 30
    minEvictableIdleTimeMillis: 30000
    minIdle: 5
    testOnBorrow: true
```

```
    timeBetweenEvictionRunsMillis: 30000

##安全配置##
#security:
#  oauth2:
#    resource:
#      id: example-service
#      ##如果配置了 user-info-uri，该资源服务器使用 userInfoTokenServices 远程调用认证中心
接口，通过认证中心的 OAuth2AuthenticationProcessingFilter 完成验证工作，一般设置
user-info-uri 即可
#      user-info-uri: http://qsdi-gateway.basic:8888/qsdi-auth/oauth/user
#      prefer-token-info: false
#宇视博观算法
bs:
  face:
    detect:
      batchDataSize: 16
      score: 90
      server: http://192.168.1.113:4001
      threadPoolSize: 6
  vehicle:
    detect:
      batchDataSize: 16
      score: 90
      server: http://192.168.1.113:4000
      threadPoolSize: 6
  #博观 1：N 算法
  search:
    server: http://192.168.1.113:18040
    algorithmVersion: test
#深网人脸算法
sn:
  face:
    detect:
      batchDataSize: 16
      score: 90
      server: http://192.168.2.23:9001
      threadPoolSize: 6
#深网 fse 配置
fse:
  server:
    capacity: 100000000
    data_type: 2
    feat_dim: 384
```



```
    featureUrl: /fse/feature
    host: http://192.168.2.16:9900
    mutilSearchUrl: /fse/mutil_search
    op: 1
    repoUrl: /fse/repo
    searchUrl: /fse/search
    threshold: 0.95
    top: 1
    type: 1

#旷视人脸算法 url
ks:
  face:
    #创建人脸库
    faceGroupUrl: /api/armour/v1/faceGroups
    #添加人脸到人脸库
    addFaceUrl: /api/armour/v1/faceGroups/{faceGroupId}/faces
    #1:1
    one2OneUrl: /api/armour/v1/action/face/verify
    #1:N
    one2NUrl: /api/armour/v1/action/face/query
    #人脸结构化
    faceStructuredUrl: /api/armour/v1/action/analyze

#系统参数配置
system:
  pram:
    #线程池配置参数
    thread:
      threadPoolSize: 5
      threadMaxPoolSize: 10
      threadAliveTime: 20

#车辆算法接口配置
vehicle:
  algorithm:
    vendor:
      #博观车辆结构化 url
      bgStructuredUrl: /V1/Task/SyncPicSearch
      #海康车辆结构化 url
      hkStructuredUrl: /api/aibasic/v1/vehicle/imageDetectionAndModeling
      #旷视车辆结构化 url
      ksStructuredUrl: /api/armour/v1/action/analyze
      #以萨车辆结构化 url
```



```
    ysStructuredUrl: /api/image/process

#以萨算法参数配置
ys:
  client_id: 1006
  client_secret: 2010b5c5f94dd15207d5d1629edf6a1e
```

b) ivdg-metadata-management-service-pro.yaml

```
spring:
  datasource:
    type: com.alibaba.druid.pool.DruidDataSource
    druid:
      driver-class-name: com.mysql.cj.jdbc.Driver
      url:
jdbc:mysql://192.168.1.110:3306/qsdi_ivdg?allowMultiQueries=true&useUnicode=true&ch
aracterEncoding=UTF-8&useSSL=false&serverTimezone=Asia/Shanghai
      username: root
      password: qsdi
      initial-size: 10
      max-active: 100
      min-idle: 10
      max-wait: 60000
      pool-prepared-statements: true
      max-pool-prepared-statement-per-connection-size: 20
      time-between-eviction-runs-millis: 60000
      min-evictable-idle-time-millis: 300000
      #Oracle 需要打开注释
      #validation-query: SELECT 1 FROM DUAL
      test-while-idle: true
      test-on-borrow: false
      test-on-return: false
      stat-view-servlet:
        enabled: true
        url-pattern: /druid/*
        #login-username: admin
        #login-password: admin
      filter:
        stat:
          log-slow-sql: true
          slow-sql-millis: 1000
```

```
        merge-sql: false
      wall:
        config:
          multi-statement-allow: true

cloud:
  sentinel:
    transport:
      dashboard: http://192.168.1.110:8858
    eager: true
```

c) ivdg-data-governance-service-pro.yaml

```
spring:
  main:
    allow-bean-definition-overriding: true
  datasource:
    type: com.alibaba.druid.pool.DruidDataSource
    druid:
      driver-class-name: com.mysql.cj.jdbc.Driver
      url:
jdbc:mysql://192.168.1.110:3306/qsdi_ivdg?serverTimezone=Asia/Shanghai&allowMultiQu
eries=true&useUnicode=true&characterEncoding=UTF-8&useSSL=false
      username: root
      password: qsdi
      initial-size: 10
      max-active: 100
      min-idle: 10
      max-wait: 60000
      pool-prepared-statements: true
      max-pool-prepared-statement-per-connection-size: 20
      time-between-eviction-runs-millis: 60000
      min-evictable-idle-time-millis: 300000
      #Oracle 需要打开注释
      #validation-query: SELECT 1 FROM DUAL
      test-while-idle: true
      test-on-borrow: false
      test-on-return: false
      stat-view-servlet:
        enabled: true
        url-pattern: /druid/*
```



```
      #login-username: admin
      #login-password: admin
    filter:
      stat:
        log-slow-sql: true
        slow-sql-millis: 1000
        merge-sql: false
      wall:
        config:
          multi-statement-allow: true
  quartz:
    jdbc:
      driver-class-name: com.mysql.cj.jdbc.Driver
      url:
jdbc:mysql://192.168.1.110:3306/qsdi_quartz?allowMultiQueries=true&useUnicode=true&
characterEncoding=UTF-8&useSSL=false&serverTimezone=Asia/Shanghai
      username: root
      password: qsdi
    scheduler:
      instanceName: FaceScheduler

  redis: # spring.factories
    database: 1
    host: 192.168.1.110
    port: 6379
    password: qishudi
    pool:
      maxIdle: 10
      maxTotal: 30
      minEvictableIdleTimeMillis: 30000
      minIdle: 5
      testOnBorrow: true
      timeBetweenEvictionRunsMillis: 30000

  cloud:
    sentinel:
      transport:
        dashboard: 192.168.1.110:8858
      eager: true

##安全配置##
security:
  oauth2:
    ignore:
```

```
      urls: /**
    client:
      client-id: qsdi
      client-secret: qsdi
      scope: server
    resource:
      id: ivdg-data-governance-service
      token-info-uri: http://qsdi-auth/oauth/check_token
      prefer-token-info: true

kafka:
  consumer:
    auto:
      commit:
        interval: 50
      offset:
        reset: earliest
    concurrency: 6
    connect: 192.168.1.110:9090,192.168.1.110:9091,192.168.1.110:9092
    enable:
      auto:
        commit: false
    group: big_lib_compare_group
    heartbeat:
      interval:
        ms: 5000
    max:
      partition:
        fetch:
          bytes: 10485760
      poll:
        interval:
          ms: 30000
        records: 1
    servers: 192.168.1.110:9090,192.168.1.110:9091,192.168.1.110:9092
    session:
      timeout: 30000
  producer:
    acks: 0
    batch:
      size: 0
    buffer:
      memory: 99999999
    linger: 0
```



```
    max:
      request:
        size: 10485760
    request:
      timeout: 60000
    retries: 0
    servers: 192.168.1.110:9090,192.168.1.110:9091,192.168.1.110:9092

#线程处理数量
taskNum: 1000
deviceNum: 1000
personLibNum: 50
personBaseNum: 1000
faceNum: 30
personNum: 3
deviceKafka: UNVIEW_ACCESS_POOL_DEVICE_DOC_INFO
deviceStandardKafka: UNVIEW-COLLECTION_DEVICE_CATEGORY
deviceTempKafka: UNVIEW_ACCESS_POOL_T_DEVICE_INFO_TEMPORARY
faceKafka: UNVIEW-FACE,HIKVISION-FACE
vehicleKafka: UNVIEW-MOTOR_VEHICLE
importantPersonEsKafka: FOCUS_PERSON_TOPIC
importantPersonBaseKafka: HIKVISION_IMPORTANT_PERSON_TOPIC
importantPersonLibKafka: HIKVISION_IMPORTANT_PERSON_LIB_TOPIC

# wt 字幕指标识别方式 sdk or ocr
osd:
  model: ocr
  # ocr 字幕图片所需的临时路径 wt
  file:
    path: D:\\osdfile
thread:
  corePoolSize: 5
  maxPoolSize: 10
  keepAliveSeconds: 200
  queueCapacity: 200

data-governance:
  vehicleSize: 10

#ntp 的服务器时间 wt
ntp:
  ip: 202.120.2.101

# Minio 配置
```



```
minio:
  url: http://192.168.1.110:9002
  accessKey: qsdi@2021
  secretKey: qsdi@2021
  bucketName: qsdi
```

d) ivdg-device-data-service-pro.yaml

```
spring:
  application:
    name: ivdg-device-data-service
  datasource:
    type: com.alibaba.druid.pool.DruidDataSource
    druid:
      driver-class-name: com.mysql.cj.jdbc.Driver
      url:
jdbc:mysql://192.168.1.110:3306/qsdi_ivdg?allowMultiQueries=true&useUnicode=true&ch
aracterEncoding=UTF-8&useSSL=false&serverTimezone=Asia/Shanghai
      username: root
      password: qsdi
      initial-size: 10
      max-active: 100
      min-idle: 10
      max-wait: 60000
      pool-prepared-statements: true
      max-pool-prepared-statement-per-connection-size: 20
      time-between-eviction-runs-millis: 60000
      min-evictable-idle-time-millis: 300000
      #Oracle 需要打开注释
      #validation-query: SELECT 1 FROM DUAL
      test-while-idle: true
      test-on-borrow: false
      test-on-return: false
      stat-view-servlet:
        enabled: true
        url-pattern: /druid/*
        #login-username: admin
        #login-password: admin
      filter:
        stat:
          log-slow-sql: true
```



```
        slow-sql-millis: 1000
        merge-sql: false
      wall:
        config:
          multi-statement-allow: true

  redis: # spring.factories
    database: 1
    host: 192.168.1.110
    port: 6379
    password: qishudi
    pool:
      maxIdle: 10
      maxTotal: 30
      minEvictableIdleTimeMillis: 30000
      minIdle: 5
      testOnBorrow: true
      timeBetweenEvictionRunsMillis: 30000

  cloud:
    sentinel:
      transport:
        dashboard: 192.168.1.110:8858
      eager: true

  quartz:
    jdbc:
      driver-class-name: com.mysql.cj.jdbc.Driver
      url:
jdbc:mysql://192.168.1.110:3306/qsdi_quartz?allowMultiQueries=true&useUnicode=true&
characterEncoding=UTF-8&useSSL=false&serverTimezone=Asia/Shanghai
      username: root
      password: qsdi
    scheduler:
      instanceName: FaceScheduler

kafka:
  consumer:
    auto:
      commit:
        interval: 50
      offset:
        reset: earliest
    concurrency: 6
```



```
    connect: 192.168.1.110:9090,192.168.1.110:9091,192.168.1.110:9092
    enable:
      auto:
        commit: false
    group: big_lib_compare_group
    heartbeat:
      interval:
        ms: 5000
    max:
      partition:
        fetch:
          bytes: 10485760
      poll:
        interval:
          ms: 30000
        records: 1
    servers: 192.168.1.110:9090,192.168.1.110:9091,192.168.1.110:9092
    session:
      timeout: 30000
  producer:
    acks: 0
    batch:
      size: 0
    buffer:
      memory: 99999999
    linger: 0
    max:
      request:
        size: 10485760
    request:
      timeout: 60000
    retries: 0
    servers: 192.168.1.110:9090,192.168.1.110:9091,192.168.1.110:9092

initDeviceCount: 300
initFaceCount: 200
taskNum: 100
#抽样配置  0 正常抽样车辆  1 抽样卡口
dataSamplingConfig: 0

##########多线程线程池配置############
thread:
  core:
    pool:
```



```
    size: 5
  max:
    pool:
      size: 10
  queue:
    capacity: 20
  keepAlive:
    seconds: 200
ntp:
  ip: 202.120.2.101
```

e) ivdg-evaluation-management-service-pro.yaml

```
spring:
  main:
    allow-bean-definition-overriding: true
  datasource:
    type: com.alibaba.druid.pool.DruidDataSource
    druid:
      driver-class-name: com.mysql.cj.jdbc.Driver
      url:
jdbc:mysql://192.168.1.110:3306/qsdi_ivdg?allowMultiQueries=true&useUnicode=true&ch
aracterEncoding=UTF-8&useSSL=false&serverTimezone=Asia/Shanghai
      username: root
      password: qsdi
      initial-size: 10
      max-active: 100
      min-idle: 10
      max-wait: 60000
      pool-prepared-statements: true
      max-pool-prepared-statement-per-connection-size: 20
      time-between-eviction-runs-millis: 60000
      min-evictable-idle-time-millis: 300000
      #Oracle 需要打开注释
      #validation-query: SELECT 1 FROM DUAL
      test-while-idle: true
      test-on-borrow: false
      test-on-return: false
      stat-view-servlet:
        enabled: true
        url-pattern: /druid/*
```



```
      #login-username: admin
      #login-password: admin
    filter:
      stat:
        log-slow-sql: true
        slow-sql-millis: 1000
        merge-sql: false
      wall:
        config:
          multi-statement-allow: true

  # cloud:
  #   nacos:
  #     discovery:
  #       server-addr: http://192.168.1.121:8848
  #       username: nacos
  #       password: nacos
  #       group: qsdi-group
  #     config:
  #       server-addr: http://192.168.1.121:8848
  #       file-extension: yaml
  #       group: qsdi-group
  redis: # spring.factories
    database: 1
    host: 192.168.1.110
    port: 6379
    password: qishudi
    pool:
      maxIdle: 10
      maxTotal: 30
      minEvictableIdleTimeMillis: 30000
      minIdle: 5
      testOnBorrow: true
      timeBetweenEvictionRunsMillis: 30000


##安全配置##
security:
  oauth2:
    ignore:
      urls: /**
    client:
      client-id: qsdi
      client-secret: qsdi
```

```
    scope: server
  resource:
    id: ivdg-data-governance-service
    token-info-uri: http://qsdi-auth/oauth/check_token
    prefer-token-info: true


kafka:
  consumer:
    auto:
      commit:
        interval: 50
      offset:
        reset: earliest
    concurrency: 6
    connect: 192.168.1.110:9090,192.168.1.110:9091,192.168.1.110:9092
    enable:
      auto:
        commit: false
    group: big_lib_compare_group
    heartbeat:
      interval:
        ms: 5000
    max:
      partition:
        fetch:
          bytes: 10485760
      poll:
        interval:
          ms: 30000
        records: 1
    servers: 192.168.1.110:9090,192.168.1.110:9091,192.168.1.110:9092
    session:
      timeout: 30000
  producer:
    acks: 0
    batch:
      size: 0
    buffer:
      memory: 99999999
    linger: 0
    max:
      request:
        size: 10485760
```



```
    request:
      timeout: 60000
    retries: 0
    servers: 192.168.1.110:9090,192.168.1.110:9091,192.168.1.110:9092
face:
  detect:
    batchDataSize: 16
    server: http://192.168.1.110:9001
    threadPoolSize: 6
    url: /face/detect?attr=1&getfeature=1&priority=0&from=net
ribbon:
  ConnectTimeout: 100000
  ReadTimeout: 100000


# Minio 配置
minio:
  url: http://192.168.1.110:9002
  accessKey: qsdi@2021
  secretKey: qsdi@2021
  bucketName: qsdi

#ntp 的服务器时间 wt
ntp:
  ip: 202.120.2.101

#查看多少天之前的历史录像 wt
video:
  day: 15

# wt 字幕指标识别方式 sdk or ocr
osd:
  model: ocr
  # ocr 字幕图片所需的临时路径 wt
  file:
    path: D:\\osdfile
```

f) ivdg-device-detection-service-pro.yaml

```
spring:
  security:
    user:
      name: admin
```



```
    password: admin
  main:
    allow-bean-definition-overriding: true
  datasource:
    type: com.alibaba.druid.pool.DruidDataSource
    druid:
      driver-class-name: com.mysql.cj.jdbc.Driver
      url:
jdbc:mysql://192.168.1.110:3306/qsdi_ivdg?allowMultiQueries=true&useUnicode=true&ch
aracterEncoding=UTF-8&useSSL=false&serverTimezone=Asia/Shanghai
      username: root
      password: qsdi
      initial-size: 10
      max-active: 100
      min-idle: 10
      max-wait: 60000
      pool-prepared-statements: true
      max-pool-prepared-statement-per-connection-size: 20
      time-between-eviction-runs-millis: 60000
      min-evictable-idle-time-millis: 300000
      #Oracle 需要打开注释
      #validation-query: SELECT 1 FROM DUAL
      test-while-idle: true
      test-on-borrow: false
      test-on-return: false
      stat-view-servlet:
        enabled: true
        url-pattern: /druid/*
        #login-username: admin
        #login-password: admin
      filter:
        stat:
          log-slow-sql: true
          slow-sql-millis: 1000
          merge-sql: false
        wall:
          config:
            multi-statement-allow: true
  redis: # spring.factories
    database: 1
    host: 192.168.1.110
    port: 6379
    password: qishudi
    pool:
```



```
    maxIdle: 10
    maxTotal: 30
    minEvictableIdleTimeMillis: 30000
    minIdle: 5
    testOnBorrow: true
    timeBetweenEvictionRunsMillis: 30000

  cloud:
    sentinel:
      transport:
        dashboard: 192.168.1.110:8858
      eager: true
```

g) ivdg-gb28181-service-pro.yaml

```
server:
  ssl:
    enabled: false
spring:
  datasource:
    type: com.alibaba.druid.pool.DruidDataSource
    druid:
      driver-class-name: com.mysql.cj.jdbc.Driver
      url:
jdbc:mysql://192.168.1.110:3306/qsdi_ivdg?allowMultiQueries=true&useUnicode=true&ch
aracterEncoding=UTF-8&useSSL=false
      username: root
      password: qsdi
      initial-size: 10
      max-active: 100
      min-idle: 10
      max-wait: 60000
      pool-prepared-statements: true
      max-pool-prepared-statement-per-connection-size: 20
      time-between-eviction-runs-millis: 60000
      min-evictable-idle-time-millis: 300000
      #Oracle 需要打开注释
      #validation-query: SELECT 1 FROM DUAL
      test-while-idle: true
      test-on-borrow: false
      test-on-return: false
```



```
      stat-view-servlet:
        enabled: true
        url-pattern: /druid/*
        #login-username: admin
        #login-password: admin
      filter:
        stat:
          log-slow-sql: true
          slow-sql-millis: 1000
          merge-sql: false
        wall:
          config:
            multi-statement-allow: true
  redis: # spring.factories
    database: 0
    host: 192.168.1.110
    port: 6379
    password: qishudi
  cache:
    type: redis

  cloud:
    sentinel:
      transport:
        dashboard: http://192.168.1.110:8858
      eager: true



##安全配置##
security:
  oauth2:
    ignore:
      urls: /token/**
    client:
      client-id: qsdi
      client-secret: qsdi
      scope: server
    resource:
      id: qsdi-system
      token-info-uri: http://qsdi-auth/oauth/check_token
      prefer-token-info: true

mybatis-plus:
```

```yaml
  type-enums-package: com.qsdi.gb.api.enums

kafka:
  consumer:
    auto:
      commit:
        interval: 50
      offset:
        reset: earliest
    concurrency: 6
    connect: 192.168.1.110:9090,192.168.1.110:9091,192.168.1.110:9092
    enable:
      auto:
        commit: false
    group: big_lib_compare_group
    heartbeat:
      interval:
        ms: 5000
    max:
      partition:
        fetch:
          bytes: 10485760
      poll:
        interval:
          ms: 30000
        records: 5
    servers: 192.168.1.110:9090,192.168.1.110:9091,192.168.1.110:9092
    session:
      timeout: 30000
  producer:
    acks: 0
    batch:
      size: 0
    buffer:
      memory: 99999999
    linger: 0
    max:
      request:
        size: 10485760
    request:
      timeout: 60000
    retries: 0
    servers: 192.168.1.110:9090,192.168.1.110:9091,192.168.1.110:9092
```



```
#zlm 服务器配置
media:
  nodes:
    - ip: 192.168.1.110
  # [可选] zlm 服务器的公网 IP，内网部署置空即可
  wan-ip:
  # [可选] 服务网关 IP
  hook-ip: 192.168.1.110
  # 服务网关端口
  hook-port: 8888
  # [必须修改] zlm 服务器的 http.port
  http-port: 31080
  # [可选] zlm 服务器的 http.sslport，置空使用 zlm 配置文件配置
  http-ssl-port: 31443
  # [可选] zlm 服务器的 rtmp.port，置空使用 zlm 配置文件配置
  rtmp-port: 31935
  # [可选] zlm 服务器的 rtmp.sslport，置空使用 zlm 配置文件配置
  rtmp-ssl-port: 31936
  # [可选] zlm 服务器的 rtp_proxy.port，置空使用 zlm 配置文件配置
  rtp-proxy-port: 31000
  # [可选] zlm 服务器的 rtsp.port，置空使用 zlm 配置文件配置
  rtsp-port: 31554
  # [可选] zlm 服务器的 rtsp.sslport，置空使用 zlm 配置文件配置
  rtsp-ssl-port: 31332
  # [可选] 是否自动配置 ZLM，如果希望手动配置 ZLM，可以设为 false
  auto-config: true
  # [可选] zlm 服务器的 hook.admin_params=secret
  secret: 035c73f7-bb6b-4889-a715-d9eb2d1925cc
  # [可选] zlm 服务器的 general.streamNoneReaderDelayMS
  stream-none-reader-delay-ms:  20000  # 无人观看多久自动关闭流，-1 表示永不自动关闭,即关闭按需拉流
  # 启用多端口模式，多端口模式使用端口区分每路流，兼容性更好。 单端口使用流的 ssrc 区分， 点播超时建议使用多端口测试
  rtp:
    # [可选] 是否启用多端口模式，开启后会在 portRange 范围内选择端口用于媒体流传输
    enable: true
    # [可选] 在此范围内选择端口用于媒体流传输，
    port-range: 32000,32500 # 端口范围
userSettings:
  # [可选] 自动点播， 使用固定流地址进行播放时，如果未点播则自动进行点播，需要 rtp.enable=true
  autoApplyPlay: false
  # [可选] 部分设备需要扩展 SDP，需要打开此设置
  seniorSdp: false
```



```
  # 保存移动位置历史轨迹：true:保留历史数据，false:仅保留最后的位置(默认)
  savePositionHistory: false
  # 点播等待超时时间,单位：毫秒
  playTimeout: 15000
  # 异步请求通用超时时间
  asynRequestTimeout: 15000
  # 等待音视频编码信息再返回， true： 可以根据编码选择合适的播放器，false： 可以更快点播
  waitTrack: false
  # 是否开启接口鉴权
  interfaceAuthentication: true
  # 是否新增数据到 device_info
  addDeviceInfo: true
  addDeviceType: 131,132
feign:
  sentinel:
    enabled: true
  client:
    config:
      default:
        connectTimeout: 550000
        readTimeout: 550000

# 日志配置
logging:
  level:
    com.qsdi: DEBUG
    org.springframework: INFO
    org.spring.springboot.dao: INFO
    org.springframework.jdbc.core.JdbcTemplate: DEBUG
```

h) ivdg-video-info-service-pro.yaml

```
spring:
  datasource:
    type: com.alibaba.druid.pool.DruidDataSource
    druid:
      driver-class-name: com.mysql.cj.jdbc.Driver
      url:
jdbc:mysql://192.168.1.110:3306/qsdi_ivdg?allowMultiQueries=true&useUnicode=true&ch
aracterEncoding=UTF-8&useSSL=false
```



```
    username: root
    password: qsdi
    initial-size: 10
    max-active: 100
    min-idle: 10
    max-wait: 60000
    pool-prepared-statements: true
    max-pool-prepared-statement-per-connection-size: 20
    time-between-eviction-runs-millis: 60000
    min-evictable-idle-time-millis: 300000
    #Oracle 需要打开注释
    #validation-query: SELECT 1 FROM DUAL
    test-while-idle: true
    test-on-borrow: false
    test-on-return: false
    stat-view-servlet:
      enabled: true
      url-pattern: /druid/*
      #login-username: admin
      #login-password: admin
    filter:
      stat:
        log-slow-sql: true
        slow-sql-millis: 1000
        merge-sql: false
      wall:
        config:
          multi-statement-allow: true
  redis: # spring.factories
    database: 0
    host: 192.168.1.110
    port: 6379
    password: qishudi
  cache:
    type: redis

  cloud:
    sentinel:
      transport:
        dashboard: http://192.168.1.110:8858
      eager: true
  resources:
    static-locations: file:${picture.image-path}
# Minio 配置
```



```yaml
minio:
  url: http://192.168.1.110:9000
  accessKey: qsdi@2021
  secretKey: qsdi@2021
  bucketName: qsdi


##安全配置##
security:
  oauth2:
    ignore:
      urls: /token/**
    client:
      client-id: qsdi
      client-secret: qsdi
      scope: server
    resource:
      id: qsdi-system
      token-info-uri: http://qsdi-auth/oauth/check_token
      prefer-token-info: true

mybatis-plus:
  type-enums-package: com.qsdi.video.api.enums
picture:
  image-path: /pictureTemp
#ntp 的服务器时间 wt
ntp:
  ip: 202.120.2.101
userSettings:
  media:
    playingThreadCount: 10
    playbackThreadCount: 10
    # 查询状态模式-status,不拉流模式-only_sip，拉流模式-pull
    playingMode: pull
    # 查询目录模式-record,不拉流模式-only_sip,拉流模式-pull
    playbackMode: pull
    failureRetryCount: 1
  sdk:
    macUp: true

feign:
  sentinel:
    enabled: true
  client:
```



```
    config:
      default:
        connectTimeout: 550000
        readTimeout: 550000

# 日志配置
logging:
  level:
    com.qsdi: DEBUG
    org.springframework: INFO
    org.spring.springboot.dao: INFO
    org.springframework.jdbc.core.JdbcTemplate: DEBUG
```

i) ivdg-examination-service-pro.yaml

```
spring:
  main:
    allow-bean-definition-overriding: true
  datasource:
    type: com.alibaba.druid.pool.DruidDataSource
    druid:
      driver-class-name: com.mysql.cj.jdbc.Driver
      url:
jdbc:mysql://192.168.1.110:3306/qsdi_ivdg?allowMultiQueries=true&useUnicode=true&ch
aracterEncoding=UTF-8&useSSL=false&serverTimezone=Asia/Shanghai
      username: root
      password: qsdi
      initial-size: 10
      max-active: 100
      min-idle: 10
      max-wait: 60000
      pool-prepared-statements: true
      max-pool-prepared-statement-per-connection-size: 20
      time-between-eviction-runs-millis: 60000
      min-evictable-idle-time-millis: 300000
      #Oracle 需要打开注释
      #validation-query: SELECT 1 FROM DUAL
      test-while-idle: true
      test-on-borrow: false
      test-on-return: false
      stat-view-servlet:
```

```
      enabled: true
      url-pattern: /druid/*
      #login-username: admin
      #login-password: admin
    filter:
      stat:
        log-slow-sql: true
        slow-sql-millis: 1000
        merge-sql: false
      wall:
        config:
          multi-statement-allow: true
  cloud:
    sentinel:
      transport:
        dashboard: http://sentinel.basic:8858
      eager: true

xxl:
  job:
    admin:
      addresses: http://192.168.1.110:8080/xxl-job-admin
    executor:
      appname: ivdg-examination-service
      address:
      ip:
      port: 0
      logpath: /data/ivdg/logs/ivdg-examination-service/xxl-job/jobhandler
      logretentiondays: 30
    accessToken:

system:
  applicationCode: "00000002"
```

j) ivdg-examination-app-pro.yaml

```
server:
  port: 8807

spring:
# mysql 数据库配置
```



```
  datasource:
    type: com.alibaba.druid.pool.DruidDataSource
    druid:
      driver-class-name: com.mysql.cj.jdbc.Driver
      url:
jdbc:mysql://192.168.1.110:3306/qsdi_ivdg?allowMultiQueries=true&useUnicode=true&ch
aracterEncoding=UTF-8&useSSL=false&serverTimezone=Asia/Shanghai
      username: root
      password: qsdi
      initial-size: 10
      max-active: 100
      min-idle: 10
      max-wait: 60000
      pool-prepared-statements: true
      max-pool-prepared-statement-per-connection-size: 20
      time-between-eviction-runs-millis: 60000
      min-evictable-idle-time-millis: 300000
      #Oracle 需要打开注释
      #validation-query: SELECT 1 FROM DUAL
      test-while-idle: true
      test-on-borrow: false
      test-on-return: false
      stat-view-servlet:
        enabled: true
        url-pattern: /druid/*
        #login-username: admin
        #login-password: admin
      filter:
        stat:
          log-slow-sql: true
          slow-sql-millis: 1000
          merge-sql: false
        wall:
          config:
            multi-statement-allow: true
  # sentinel 配置
  cloud:
    sentinel:
      transport:
        dashboard: http://192.168.1.110:8858
      eager: true

xxl:
  job:
```

```
    admin:
      addresses: http://192.168.1.110:8080/xxl-job-admin
    executor:
      appname: ivdg-examination-app
      address:
      ip:
      port:
      logpath: /
      logretentiondays: 30
    accessToken:
```

k) ivdg-data-sync-service.yaml

```
spring:
  main:
    allow-bean-definition-overriding: true
  datasource:
    type: com.alibaba.druid.pool.DruidDataSource
    druid:
      driver-class-name: com.mysql.cj.jdbc.Driver
      url:
jdbc:mysql://192.168.1.110:3306/qsdi_ivdg?allowMultiQueries=true&useUnicode=true&ch
aracterEncoding=UTF-8&useSSL=false
      username: root
      password: qsdi
      initial-size: 10
      max-active: 100
      min-idle: 10
      max-wait: 60000
      pool-prepared-statements: true
      max-pool-prepared-statement-per-connection-size: 20
      time-between-eviction-runs-millis: 60000
      min-evictable-idle-time-millis: 300000
      #Oracle 需要打开注释
      #validation-query: SELECT 1 FROM DUAL
      test-while-idle: true
      test-on-borrow: false
      test-on-return: false
      stat-view-servlet:
        enabled: true
        url-pattern: /druid/*
```

```yaml
      #login-username: admin
      #login-password: admin
    filter:
      stat:
        log-slow-sql: true
        slow-sql-millis: 1000
        merge-sql: false
      wall:
        config:
          multi-statement-allow: true

  cloud:
    sentinel:
      transport:
        dashboard: http://192.168.1.110:8858
      eager: true
    nacos:
      discovery:
        server-addr: ${NACOS_IP:192.168.1.110}:8848
        username: nacos
        password: nacos
        group: qsdi-group
      config:
        server-addr: ${NACOS_IP:192.168.1.110}:8848
        file-extension: yaml
        group: qsdi-group


ftp:
  # 1 2 3 都用
  host: 192.168.1.123
  # 1 2 3 都用
  port: 21
  # 1 2 3 都用
  user: test
  # 1 2 3 都用
  pwd: test
  # 1 2 3 都用
  path: /
  #每次从表中抽取一万条 针对 task.type 为 1 时这个值生效
  size: 10000
  file:
    # 针对 task.type 为 3 时这个值生效  每次取多少个文件
    size: 100
```



```
    sample:
      # 针对 task.type 为 3 时这个值生效 意思是 对抽取出来的文件进行过滤 需不需要根据指定的区过滤 截取前六位判断 0 为不需要过滤
      org_codes: 370102,370104
      # 针对 task.type 为 3 时这个值生效 意思是 每个文件里面满足 org_codes 的情况下抽取多少条数据 为 0 时 全部抽取
      size: 2
# 1 视频流表数据解析并上传 ftp 2 解析 ftp 视频流表数据并入库 3 解析浪潮相关数据信息
zip:
  file:
    # 针对 task.type 为 1 和 2 时这个值生效 意思是 压缩文件放在那里
    path: /home/video/up/zipfile/
# 1 视频流表数据解析并上传 ftp 2 解析 ftp 视频流表数据并入库 3 解析浪潮相关数据信息
task:
  type: 1
  up:
    # 针对 task.type 为 1 时这个值生效 所需要解析的表 按照依赖顺序 排列 即 后一个依赖前一个
    table:
t_evaluation_scheme,t_evaluation_scheme_index,t_evaluation_task,t_evaluation_result
,t_evaluation_result_index,t_evaluation_video_device_detail
error:
  file:
    # 针对 task.type 为 1、2、3 时这个值生效 文件名根据对应 type 进行更改 1 2 video 3 langchao
    path: /home/video/up/error/
    #错误文件是否保存 true 保存 false 不保存
    save: true

# Minio 配置
minio:
  url: http://192.168.1.110:9002
  accessKey: qsdi@2021
  secretKey: qsdi@2021
  bucketName: qsdi
```

l) ivdg-asert-center-service-pro.yaml

```
spring:
  main:
    allow-bean-definition-overriding: true
  datasource:
    type: com.alibaba.druid.pool.DruidDataSource
```



```
    druid:
      driver-class-name: com.mysql.cj.jdbc.Driver
      url:
jdbc:mysql://192.168.1.110:3306/qsdi_ivdg?allowMultiQueries=true&useUnicode=true&ch
aracterEncoding=UTF-8&useSSL=false&serverTimezone=Asia/Shanghai
      username: root
      password: qsdi
      initial-size: 10
      max-active: 100
      min-idle: 10
      max-wait: 60000
      pool-prepared-statements: true
      max-pool-prepared-statement-per-connection-size: 20
      time-between-eviction-runs-millis: 60000
      min-evictable-idle-time-millis: 300000
      #Oracle 需要打开注释
      #validation-query: SELECT 1 FROM DUAL
      test-while-idle: true
      test-on-borrow: false
      test-on-return: false
      stat-view-servlet:
        enabled: true
        url-pattern: /druid/*
        #login-username: admin
        #login-password: admin
      filter:
        stat:
          log-slow-sql: true
          slow-sql-millis: 1000
          merge-sql: false
        wall:
          config:
            multi-statement-allow: true
  cloud:
    sentinel:
      transport:
        dashboard: http://sentinel.basic:8858
      eager: true
  redis: # spring.factories
    database: 1
    host: 192.168.1.110
    port: 6379
    password: qishudi
    pool:
```

```
      maxIdle: 10
      maxTotal: 30
      minEvictableIdleTimeMillis: 30000
      minIdle: 5
      testOnBorrow: true
      timeBetweenEvictionRunsMillis: 30000
# Minio 配置
minio:
  url: http://192.168.1.110:9002
  accessKey: qsdi@2021
  secretKey: qsdi@2021
  bucketName: qsdi
#安全配置##
security:
  oauth2:
    ignore:
      urls:
        - /receiveDeviceDetail/receiveReport
        - /reportDeviceFallback/VIID/Feedbacks
    client:
      client-id: qsdi
      client-secret: qsdi
      scope: server
    resource:
      id: ivdg-asert-center-service
      token-info-uri: http://qsdi-auth/oauth/check_token
      prefer-token-info: true
# wt 字幕指标识别方式 sdk or ocr
osd:
  model: ocr
  # ocr 字幕图片所需的临时路径 wt
  file:
    path: /home/osdfile
  sdk:
    split:
    # sdk 的时间切图 时间字幕可以往左偏  数字代表 数字分之一  比如 2 就是 2 分之一 3 就是 3 分
之一
      num: 2
```


2.5.4. 应用服务部署

注：应用服务支持部署在多台服务器上，需要修改docker-compose/docker-compose.yml，仅保留当前服务器上需要启动的应用服务即可。

a) 创建目录

```
mkdir -p /home/qsdi/projects/ #对应的平台应用放入该目录下
```

b) 微盘下载：版本发布/iVDG_V1.0.4/deploy-01(compose)

清单如下:

①配置文件：.env，docker-compose.yml，

②加载镜像脚本：load_image.sh

③.dockerignore 文件，resource 文件夹

将 data-gateway，ivdg，system-center，run-conf-01.sh 上传到 projects 目录下

c) 参考“2.5.1 基础服务列表”章节，将版本镜像包分别上传到应用的 pkg_images 目录下(如果没有 pkg_images，则手工新建)

d) 加载镜像

```
[root@localhost data-gateway]# cd /home/qsdi/projects/data-gateway/docker-compose/
[root@localhost docker-compose]# sh load_image.sh
下略
[root@localhost docker-compose]# cd /home/qsdi/projects/system-center/docker-compose/
[root@localhost docker-compose]# sh load_image.sh
下略
[root@localhost docker-compose]# cd /home/qsdi/projects/ivdg/docker-compose/
[root@localhost docker-compose]# sh load_image.sh
下略
```

e) 修改 resource 目录里的 default.conf 文件


```
[root@localhost data-management-center]# pwd
/home/qsdi/projects/system-center/docker-compose/resource/data/system-view
[root@localhost data-management-center]# vi default.conf
修改每个 location 的 IP 为 qsdi-gateway.service 服务所在的 ip

[root@localhost data-getway-platform]# pwd
/home/qsdi/projects/data-gateway/docker-compose/resource/data/data-getway-platform/
[root@localhost data-getway-platform]# vi default.conf
修改每个 location 的 IP 为 qsdi-gateway.service 服务所在的 ip

[root@localhost ivdg]# pwd
/home/qsdi/projects/ivdg/docker-compose/resource/data/ivdg/
[root@localhost data-getway-platform]# vi default.conf
修改每个 location 的 IP 为 qsdi-gateway.service 服务所在的 ip
```

f) 修改 docker-compose 目录中的.env 文件

```
[root@localhost docker-compose]# pwd
/home/qsdi/projects/system-center/docker-compose
[root@localhost docker-compose]# vi .env
修改每个变量为正确的 ip

[root@localhost docker-compose]# pwd
/home/qsdi/projects/data-gateway/docker-compose
[root@localhost docker-compose]# vi .env
修改每个变量为正确的 ip

[root@localhost docker-compose]# pwd
/home/qsdi/projects/ivdg/docker-compose
[root@localhost docker-compose]# vi .env
修改每个变量为正确的 ip
```

g) 执行命令：sh run-conf-01.sh

```
[root@localhost projects]# pwd
/home/qsdi/projects
[root@localhost projects]# sh run-conf.sh-01.sh
---
初始环境配置并启动所有服务！
nacos ip: 192.168.136.128 --此处输入实际配置
qsdi-gateway ip: 192.168.136.128--此处输入实际配置
回车后，打印如下
s/nacos.basic/192.168.136.128/g /home/qsdi/projects/data-gateway/docker-compose/.env
```

```
s/nacos.basic/192.168.136.128/g /home/qsdi/projects/ivdg/docker-compose/.env
s/nacos.basic/192.168.136.128/g /home/qsdi/projects/system-center/docker-compose/.env
s/gateway.basic/192.168.136.128/g
/home/qsdi/projects/data-gateway/docker-compose/resource/data/data-getway-platform/default.conf
s/gateway.basic/192.168.136.128/g /home/qsdi/projects/ivdg/docker-compose/resource/data/ivdg/default.conf
s/gateway.basic/192.168.136.128/g
/home/qsdi/projects/system-center/docker-compose/resource/data/data-management-center/default.conf
系统管理版本：1.0.4-RELEASE #输入本次版本
sed -i s/1.0.4-RELEASE/1.0.4/g /home/qsdi/projects/system-center/docker-compose/.env
数据网关版本：1.0.4-RELEASE #输入本次版本
sed -i s/1.0.4-RELEASE 1.0.1-RELEASE/1.0.4/g /home/qsdi/projects/system-center/docker-compose/.env
iVDG 版本：1.0.4-RELEASE #输入本次版本
sed -i s/1.0.1-RELEASE 1.0.4-RELEASE/1.0.4/g /home/qsdi/projects/ivdg/docker-compose/.env
```

h) 启动容器

```
[root@localhost system-center]# cd /home/qsdi/projects/system-center/docker-compose/
[root@localhost docker-compose]# docker-compose up -d
下略
[root@localhost system-center]# cd /home/qsdi/projects/data-gateway/docker-compose/
[root@localhost docker-compose]# docker-compose up -d
下略
[root@localhost system-center]# cd /home/qsdi/projects/ivdg/docker-compose/
[root@localhost docker-compose]# docker-compose up -d
下略
```

2.5.5. 初始化 admin,root 的数据权限

a) 系统管理模块

系统管理执行完脚本后在应用部署的服务器上调用以下两个请求
说明：用于初始化 admin,root 的数据权限
方法 1：

操作：curl -X POST "http://localhost:8885/token/initSuperUser" -H "accept: */*"

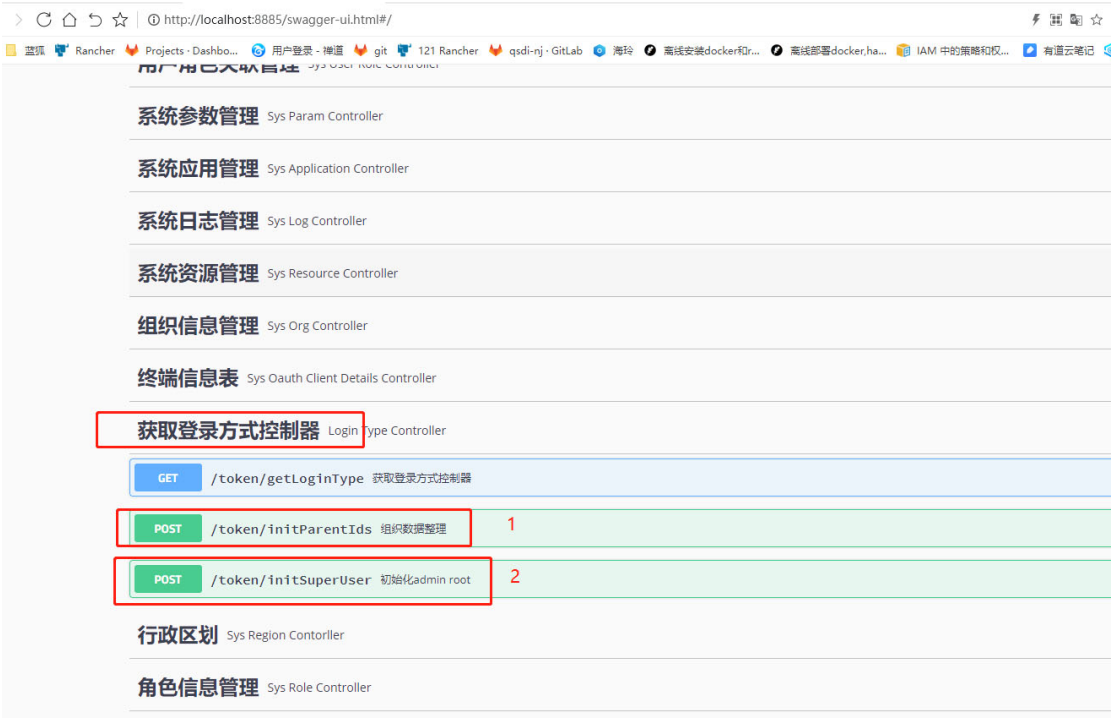
```
[root@qsdi111 ~]# curl -X POST "http://localhost:8885/token/initSuperUser" -H "accept: */*"
{"code":200,"msg":"成功","data":"初始化admin、root权限信息成功"}[root@qsdi111 ~]#
```

操作：curl -X POST "http://localhost:8885/token/initParentIds" -H "accept: */*"

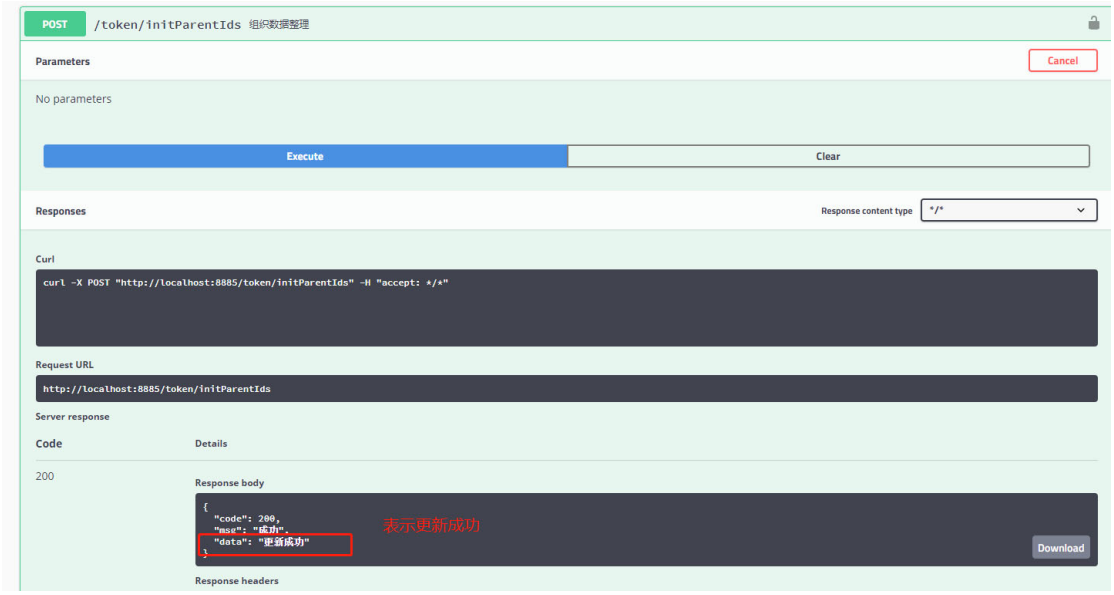
```
{"code":200,"msg":"成功","data":"初始化admin、root权限信息成功"}[root@qsdi111 ~]# curl -X POST "http://localhost:8885/token/initParentIds" -H "accept: */*"
{"code":200,"msg":"成功","data":"更新成功"}[root@qsdi111 ~]#
```

方法 2：如果服务器不支持 curl 指令

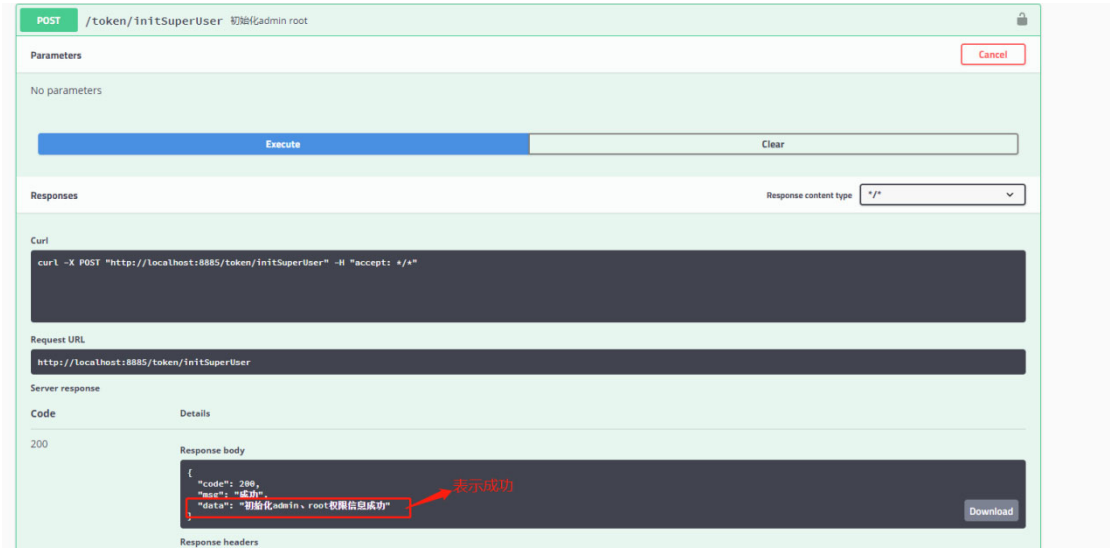
http://ip:8885/swagger-ui.html



点击 Try it out 再点击 Execute，出现下图信息表示【组织数据整理】方法执行成功

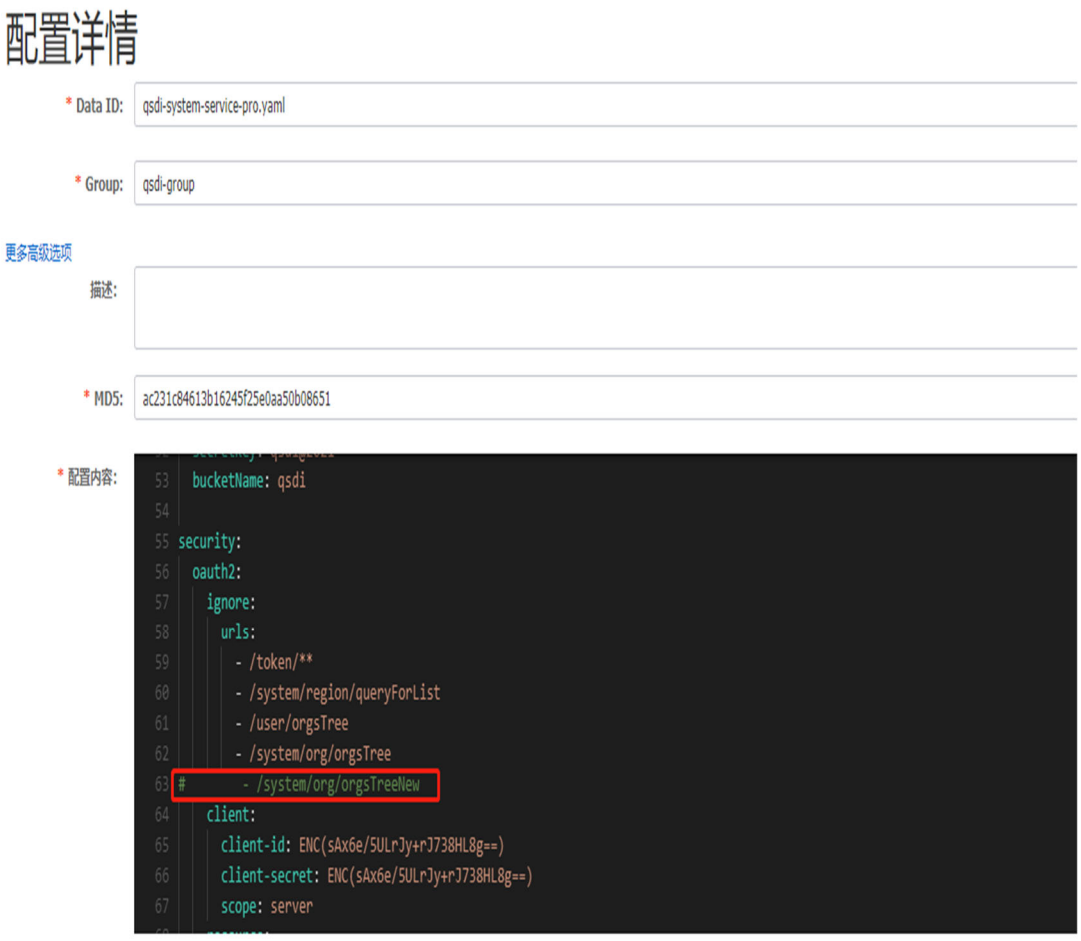


点击 Try it out 再点击 Execute，出现下图信息表示【初始化 admin root】方法执行成功



b) nacos 配置修改

qsdi-system-service-pro.yaml 配置文件
删除或注释：/system/org/orgsTreeNew
说明：解决更新后组织管理不显示组织的问题，浏览器缓存需要清下



三、 版本升级

注意：部署基础服务的服务器，如果对接过第三方，那么不管是升级还是替换服务，这台机器不能动。

3.1. 基础服务更新升级

3.1.1. kafka 集群升级

a) 如果版本包中包含 kafka 包，则需要升级 kafka

b) 升级前确认 kafka 中所有数据已经被消费掉

c) 在原安装目录中，卸载 kafka 集群

```
[root@localhost Kafka-overlay]# sh set-kafka.sh
----------Option----------
[1]: Uninstall-all-clusters
[2]: Stop-all-clusters
[3]: Start-all-clusters
[4]: Restart-all-clusters
[5]: Exit-SetUp
--------------------------

Type[Num]: 1
Uninstall all clusters are deleted?[y/n]: y
[1]--uninstall-all-clusters
remove-clusters-of: 1c56b91f4bdb
1c56b91f4bdb
exec success!
remove-clusters-of: d66940ae7515
d66940ae7515
exec success!
```

```
remove-clusters-of: 9255fa0e27a9
9255fa0e27a9
exec success!
```

d) 执行 docker ps -a，检查是否有 kafka/zk 的镜像残留，如有，则 docker rm [containerID]删掉

e) 参考 2.4.2 个章节，将新的版本包上传到新的目录，解压，然后进行重新部署

3.1.2. Mysql 基础服务升级

a) 如果版本包中包含 mariadb 包，则需要升级 mysql

b) 停止所有业务镜像之后，做好 mysql 数据的数据备份

```
--进入 mysql 容器
docker ps|grep mysql   #查 mysql 的容器 id

sudo docker exec -it mysql 的容器 id /bin/sh

--备份所有库
mysqldump -uroot -pqsdi --all-databases > /tmp/all_backup.db   #备份所有数据库到/tmp 目录下

--备份指定库
mysqldump -uroot -pqsdi qsdi_ivdg > /tmp/qsdi_ivdg_backup.db   #备份 qsdi_idvg 库到/tmp 目录下

--退出容器之后，将备份文件拷贝到物理服务器上
docker cp mysql.basic:/tmp/all_backup.db  /home/


--后续如果要恢复备份数据库
--恢复所有库
mysql -uroot -pqsdi < /tmp/all_backup.db   #恢复所有库

--恢复指定库
mysql -uroot -pqsdi qsdi_ivdg < /tmp/qsdi_ivdg_backup.db   #恢复 qsdi_idvg 库数据
```

c) 参考 2.4.3 章节，将新的版本包上传到原目录，解压，然后进行重新部署

3.1.3. 其他基础服务升级

a) 如果版本包中还包含其他的基础服务包，则需要升级

b) 在原安装目录中，停掉对应的容器，例如停掉 nacos 容器：

```
root@localhost basic-server]# docker-compose down nacos
Removing nacos.basic        ... done
```

c) 参考 2.4.3 章节，将新的版本包上传到原目录，解压，然后进行重新部署

3.2. 应用服务更新升级

3.2.1. 数据库脚本升级

a) 停止所有业务镜像之后，做好 mysql 数据的数据备份

b) 更新脚本

一。注意事项：
1. 执行脚本时，将脚本内容复制出来手动运行，不要用导入的方式执行（会报错或直接略过了报错的脚本）；
2. 执行 update 脚本时，要先选择对应的数据库；

二。V1.0.4 全新部署：
1. 执行 V1.0.4\total 下所有脚本；1.0.1_upgrade_qsdi_system_alert.sql 在 qsdi_system 库中最后执行；
2. 按照时间顺序执行 V1.0.4 目录下的其他脚本

三。从 V1.0.3 升级到 V1.0.4
1. 确认现场最后升级 V1.0.3 版本的日期，然后执行 V1.0.3 目录下未执行的脚本
2. 按照时间顺序执行 V1.0.4 目录下，除 total 之外的其他脚本

3.2.2. 配置升级

a) 参考版本包中的 ReleaseNotes.docs，进行手工配置

3.2.3. 应用服务升级

3.2.3.1. 同版本的应用服务升级

a) 如果版本包中提供了 deploy(compose)目录，则需要更新至服务器，并修改 resource 目录里的 default.conf 文件和 docker-compose/.env 文件，下表为参考：

```
[root@localhost data-management-center]# pwd
/home/qsdi/projects/system-center/docker-compose/resource/data/data-management-center
[root@localhost data-management-center]# vi default.conf
修改每个 location 的 gateway.basic 变量为正确的 ip

[root@localhost docker-compose]# pwd
/home/qsdi/projects/system-center/docker-compose
[root@localhost docker-compose]# vi .env
修改每个变量为正确的 ip
```

b) 将应用服务的镜像文件上传至所属模块的 pkg_image 目录下

c) 停止容器，删除容器，删除镜像（以 qsdi-system 服务为例）

```
[root@localhost docker-compose]# pwd
/home/qsdi/install2/deploy-01(compose)/system-center/docker-compose
[root@localhost docker-compose]# docker-compose stop qsdi-system
[root@localhost docker-compose]# docker-compose rm qsdi-system
Going to remove qsdi-system.service
Are you sure? [yN] y
Removing qsdi-system.service ... done
[root@localhost docker-compose]# docker images   #查询旧的镜像 ID
[root@localhost docker-compose]# docker rmi  镜像 ID
```

d) 加载镜像

```
[root@localhost system-center]# cd /home/qsdi/projects/system-center/docker-compose/
[root@localhost docker-compose]# sh load__image.sh
```



```
下略
```

e) 启动容器

```
[root@localhost system-center]# cd /home/qsdi/projects/system-center/docker-compose/
[root@localhost docker-compose]# docker-compose up qsdi-system
下略
```

3.2.3.2. 跨版本的应用服务升级

a) 如果版本包中提供了 deploy(compose)目录，则需要更新至服务器，并修改 resource 目录里的 default.conf 文件和 docker-compose/.env 文件，下表为参考：

```
[root@localhost data-management-center]# pwd
/home/qsdi/projects/system-center/docker-compose/resource/data/data-management-center
[root@localhost data-management-center]# vi default.conf
修改每个 location 的 gateway.basic 变量为正确的 ip

[root@localhost docker-compose]# pwd
/home/qsdi/projects/system-center/docker-compose
[root@localhost docker-compose]# vi .env
修改每个变量为正确的 ip
```

b) 将应用服务的镜像文件上传至所属模块的 pkg_image 目录下

c) 停止容器，删除容器，删除镜像（以 qsdi-system 服务为例）

```
[root@localhost docker-compose]# pwd
/home/qsdi/install2/deploy-01(compose)/system-center/docker-compose
[root@localhost docker-compose]# docker-compose stop qsdi-system
[root@localhost docker-compose]# docker-compose rm qsdi-system
Going to remove qsdi-system.service
Are you sure? [yN] y
Removing qsdi-system.service ... done
[root@localhost docker-compose]# docker images   #查询旧的镜像 ID
[root@localhost docker-compose]# docker rmi  镜像 ID
```

d) 加载镜像

```
[root@localhost system-center]# cd /home/qsdi/projects/system-center/docker-compose/
[root@localhost docker-compose]# sh load_docker_image.sh
下略
```


e) 删除/home/qsdi/projects/目录下的.sys 文件

f) 执行命令：sh run-conf-01.sh

```
[root@localhost projects]# pwd
/home/qsdi/projects
[root@localhost projects]# sh run-conf.sh-01.sh
---
初始环境配置并启动所有服务！
nacos ip: 192.168.136.128   #此处输入实际配置
qsdi-gateway ip: 192.168.136.128   #此处输入实际配置
回车后，打印如下
s/nacos.basic/192.168.136.128/g /home/qsdi/projects/data-gateway/docker-compose/.env
s/nacos.basic/192.168.136.128/g /home/qsdi/projects/ivdg/docker-compose/.env
s/nacos.basic/192.168.136.128/g /home/qsdi/projects/system-center/docker-compose/.env
s/gateway.basic/192.168.136.128/g
/home/qsdi/projects/data-gateway/docker-compose/resource/data/data-getway-platform/default.conf
s/gateway.basic/192.168.136.128/g /home/qsdi/projects/ivdg/docker-compose/resource/data/ivdg/default.conf
s/gateway.basic/192.168.136.128/g
/home/qsdi/projects/system-center/docker-compose/resource/data/data-management-center/default.conf
系统管理版本：1.0.4-RELEASE #输入本次版本
sed -i s/1.0.4-RELEASE/1.0.4/g /home/qsdi/projects/system-center/docker-compose/.env
数据网关版本：1.0.4-RELEASE #输入本次版本
sed -i s/1.0.4-RELEASE 1.0.1-RELEASE/1.0.4/g /home/qsdi/projects/system-center/docker-compose/.env
iVDG 版本：1.0.4-RELEASE #输入本次版本
sed -i s/1.0.1-RELEASE 1.0.4-RELEASE/1.0.4/g /home/qsdi/projects/ivdg/docker-compose/.env
```

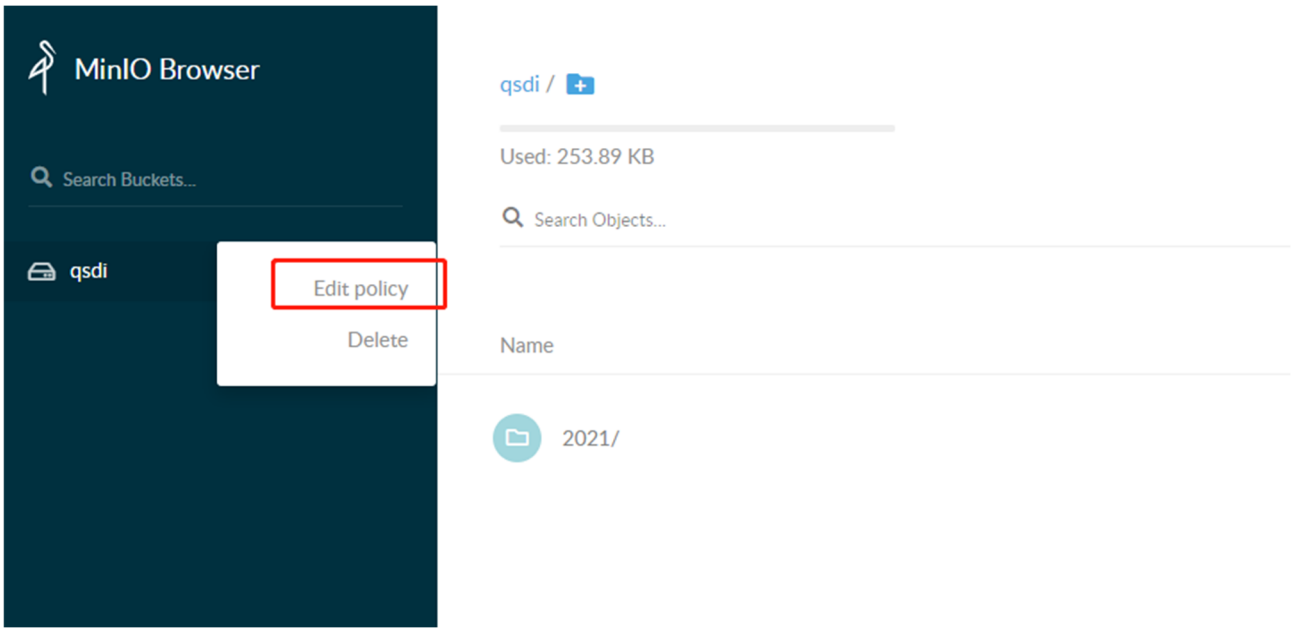
g) 启动容器

```
[root@localhost system-center]# cd /home/qsdi/projects/system-center/docker-compose/
[root@localhost docker-compose]# docker-compose up qsdi-system
下略
```


四、 Q&A

4.1. 服务datasources-management-service报文件解析失败

解决：登录 minIO，编辑 qsdi,增加策略，如下图



4.2. Kafka Tool 无法连接 kafka

解决：

C:\Windows\System32\drivers\etc\hosts

增加：

192.168.1.110 kafka1

192.168.1.111 kafka2

192.168.1.112 kafka3

其中 192.168.1.110、111、112 为 kafka 各节点所在服务器的 IP

4.3. 修改 MySQL 最大连接数

目前系统默认的 MySQL 最大连接数偏小：在 Navicat 中执行：
show variables like 'max_connections'; -- 查询当前最大连接数，应该是 151

需要改大，步骤如下：
1. 登录服务器
2. docker ps 查找 mysql 容器的 ID，比如为 b60187d7d645
3. docker cp b60187d7d645:/etc/mysql/mariadb.cnf ./
4. vi mariadb.cnf，在[mariadbd]下新增：max_connections = 1000，保存
5. docker cp mariadb.cnf b60187d7d645:/etc/mysql/mariadb.cnf
6. docker restart b60187d7d645

重启完成之后，在 Navicat 中执行： show variables like 'max_connections'; -- 查询当前最大连接数，应该是 1000