

# 泛微平台系统部署

本文主要描述了泛微平台环境部署的几种方案，包括 `window / linux` 环境下的 `e-cology`，`e-mobile` 部署，集群部署，docker 微服务部署等。

## 1. 单机部署

通常情况下，如果客户的服务器系统是 `window server`，那么在部署 `e-cology` 时，数据库一般选用 `sql server` (版本建议在2014以上)

### 1.1 SQL Server 部署

SQL Server 的安装包可以在 <https://msdn.itellyou.cn/> 中获取，在安装包的选择上需要注意的是：

`sql_server_xx_express_with_tools` = `sql_server_xx_express` + `sql_server_xx_management_studio`

- `sql_server_xx_express`：表示企业版数据库
- `sql_server_xx_management_studio`：数据库管理工具软件。

这里我们选择的是 `cn_sql_server_2014_express_with_tools_x64`，因为该安装包中包含数据库管理工具。

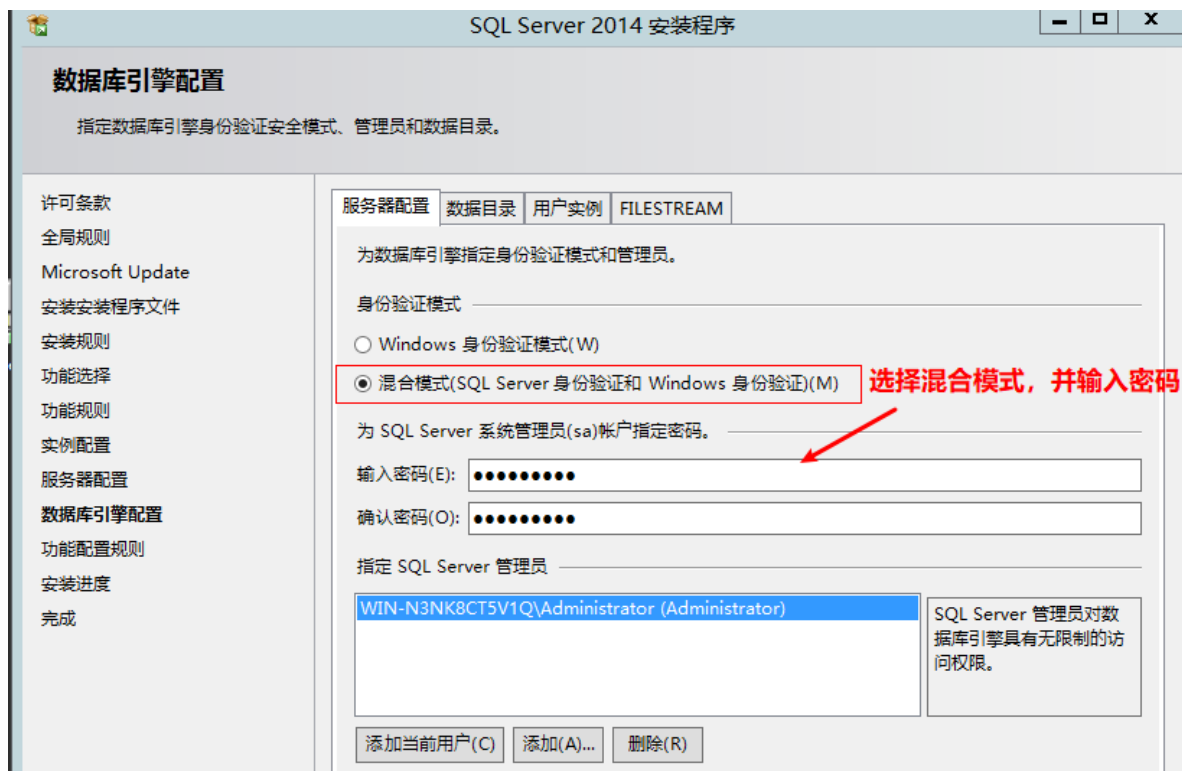


安装 `SQL Server 2014`

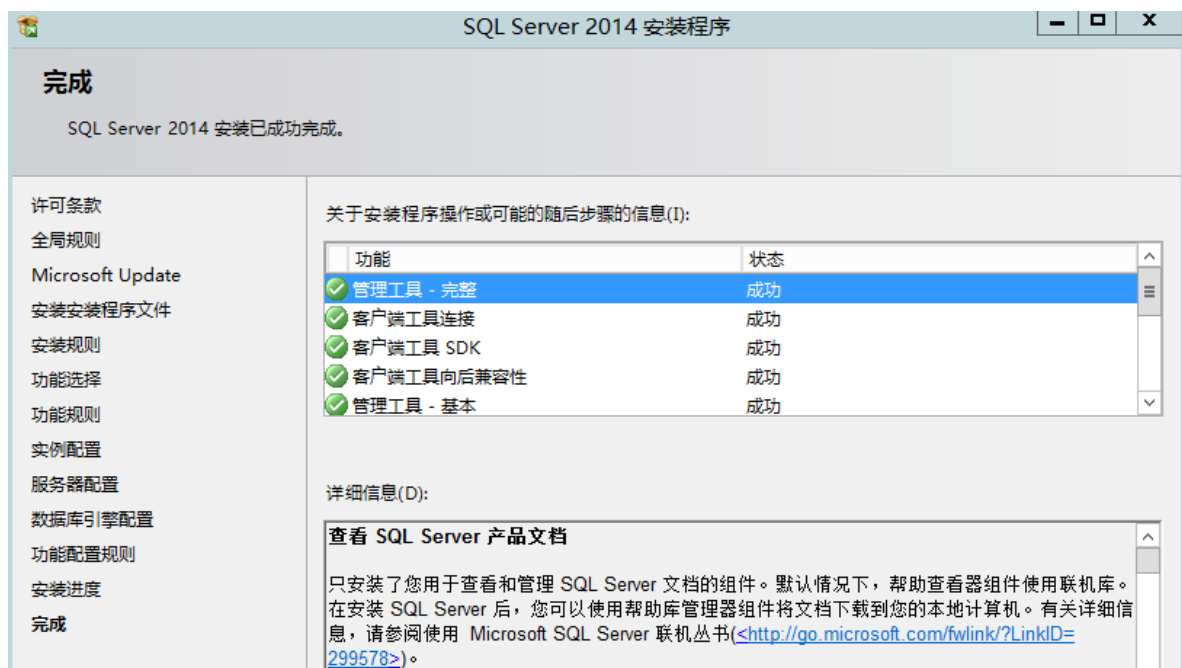
- 提取安装包，开始安装。注意以下窗口在安装过程中不允许关闭掉，否则会安装失败！



- 一直下一步，直到以下步骤时进行 `sa` 账号密码设置



- 选择下一步，等待安装完毕即可



## 1.2 Oracle 部署

如果客户服务器系统是 linux，那么一般采用 `Oracle` 或 `MySQL` 作为数据库，这边要向客户表述 `Oracle` 与 `MySQL` 的区别，其主要区别是 `Oracle` 属于商业产品，盗版可能会被告，但是其性能卓越，抗压性较强。`MySQL` 属于开源产品，抗压性较弱。

一般企业用户所使用的 `linux` 发行版本都是 `CentOS` 或者 `Red Hat`，接下来会以 `CentOS` 系统为例进行 Oracle 部署的介绍

### 1.2.0 快速安装

如果选择脚本安装的方式，则只要看 1.2.0 即可，其他的为具体步骤。

脚本地址：

- `install.sh`: <https://github.com/Y-Aron/cloud-note/blob/master/weaver/shell/oracle/install.sh>
- `dbca.rsp`: <https://github.com/Y-Aron/cloud-note/blob/master/weaver/shell/oracle/dbca.rsp>
- `new_sid.sh`: [https://github.com/Y-Aron/cloud-note/blob/master/weaver/shell/oracle/new\\_sid.sh](https://github.com/Y-Aron/cloud-note/blob/master/weaver/shell/oracle/new_sid.sh)

shell 脚本安装步骤如下:

- 上传 `install.sh`, `dbca.rsp`, `new_sid.sh` 到与 `oracle` 安装包同级目录下
- 切换至 `root` 用户, 执行 `install.sh` 脚本

```
1 su root
2 ./install.sh
```

- `dbca.rsp`: 静默安装 `oracle sid` 的相关配置文件, 可根据实际情况自行配置
- 当出现 `Successfully Set Software` 提醒时, 执行 `new_sid.sh`

```
1 ./new_sid.sh
```

### 1.2.1 安装前准备

- 使用 `xshell` 软件连接服务器: 软件安装自行百度
- 配置镜像源: 复制并执行以下命令

```
1 cd /etc/yum.repos.d/
2 # 下载并安装阿里云镜像
3 yum install wget
4 sudo wget http://mirrors.aliyun.com/repo/Centos-7.repo
5 sudo mkdir repo_bak
6 mv /etc/yum.repos.d/repo_bak/Centos-7.repo /etc/yum.repos.d/
7 # 清除并重新生成yum缓存
8 yum clean all
9 yum makecache
```

- 下载 `Oracle11gR2`: [https://pan.baidu.com/s/1gA33\\_cOlyzq6Bu3C4R8Vfw](https://pan.baidu.com/s/1gA33_cOlyzq6Bu3C4R8Vfw)

### 1.2.2 安装 Oracle

- 安装所需的软件包

```
1 yum -y install binutils compat-libstdc++-33 compat-libstdc++-33.i686
elfutils-libelf elfutils-libelf-devel gcc gcc-c++ glibc glibc.i686 glibc-
common glibc-devel glibc-devel.i686 glibc-headers ksh libaio libaio.i686
libaio-devel libaio-devel.i686 libgcc libgcc.i686 libstdc++ libstdc++.i686
libstdc++-devel make sysstat unixODBC unixODBC
```

- 创建用户组和用户

```

1 # 创建 oinstall 用户组
2 groupadd oinstall
3 # 创建 dba 用户组
4 groupadd dba
5 # 创建 oracle 用户
6 useradd -g oinstall -G dba oracle
7 # 设置密码
8 passwd oracle

```

- 修改 kernel 内核参数

```

1 vi /etc/sysctl.conf
2 # -----
3 # 配置文件内加入 修改以下参数。如果没有可以自己添加，如果默认值比参考值大，则不需要修改。
4 # 这个内核参数用于设置系统范围内共享内存段的最大数量。该参数的默认值是 4096 。通常不需要更改。
5 kernel.shmni = 4096
6 # 该参数定义了共享内存段的最大尺寸（以字节为单位）。缺省为32M，对于oracle来说，该缺省值太低了，# 通常将其设置为2G。
7 kernel.shmmax = 2147483648
8 # 该参数表示系统一次可以使用的共享内存总量（以页为单位）。缺省值就是2097152，通常不需要修改。
9 kernel.shmall = 2097152
10 kernel.sem = 250 32000 100 128
11 fs.aio-max-nr = 1048576
12 # 设置最大打开文件数
13 fs.file-max = 65536
14 # 可使用的IPv4端口范围
15 net.ipv4.ip_local_port_range = 9000 65500
16 net.core.rmem_default = 262144
17 net.core.rmem_max = 4194304
18 net.core.wmem_default = 262144
19 net.core.wmem_max = 1048586
20 # -----
21
22 # 保存退出后设置立刻生效
23 sysctl -p

```

- 修改系统资源限制

```

1 vi /etc/security/limits.conf
2
3 # 配置文件下方加入
4 oracle soft nproc 2047
5 oracle hard nproc 16384
6 oracle soft nofile 1024
7 oracle hard nofile 65536
8 oracle soft stack 10240
9
10 # 修改用户验证选项 关联设置
11 vi /etc/pam.d/login
12 # 在最后一规则之前加入以下session配置
13 session required /lib64/security/pam_limits.so
14 session required pam_limits.so

```

- 创建安装目录和设置目录权限

```
1 # oracle: 安装目录
2 mkdir -p /usr/local/oracle/product/11.2.0/db_1
3 # oradata: 数据存储目录
4 mkdir /usr/local/oracle/oradata
5 # flash_recovery_area: 恢复目录
6 mkdir /usr/local/oracle/flash_recovery_area
7 # oraInventory: 清单目录
8 mkdir /usr/local/oraInventory
9 chown -R oracle:oinstall /usr/local/oracle
10 chown -R oracle:oinstall /usr/local/oraInventory
11 chmod -R 775 /usr/local/oracle
12 chmod -R 775 /usr/local/oraInventory
```

- 上传文件到 /home 目录下, 并解压

```
1 yum install unzip
2 unzip linux.x64_11gR2_database_1of2.zip
3 unzip linux.x64_11gR2_database_2of2.zip
```

- 准备oracle安装应答模板文件 db\_install.rsp

```
1 cp /home/database/response/* /usr/local/oracle/
2 # 编辑应答文件的相关配置
3 vi /usr/local/oracle/db_install.rsp
4 # -----
5 # 安装类型, 只装数据库软件
6 oracle.install.option=INSTALL_DB_SWONLY
7 # 安装组
8 UNIX_GROUP_NAME=oinstall
9 # INVENTORY目录 (**不填就是默认值, 本例此处需修改, 因个人创建安装目录而定)
10 INVENTORY_LOCATION=/usr/local/oraInventory
11 # 选择语言
12 SELECTED_LANGUAGES=en,zh_CN
13 # oracle_home: 路径根据目录情况注意修改 本例安装路径/usr/local/oracle
14 ORACLE_HOME=/usr/local/oracle/product/11.2.0/db_1
15 # oracle_base: 注意修改
16 ORACLE_BASE=/usr/local/oracle
17 # oracle版本
18 oracle.install.db.InstallEdition=EE
19 # 自定义安装, 否, 使用默认组件
20 oracle.install.db.isCustomInstall=false
21 # dba用户组
22 oracle.install.db.DBA_GROUP=dba
23 # oper用户组
24 oracle.install.db.OPER_GROUP=dba
25 # 注意此参数 设定一定要为true
26 DECLINE_SECURITY_UPDATES=true
27 # -----
```

- 切换用户并开始静默安装

```

1  # 切换到oracle / 进入解压目录
2  su oracle
3  cd /home/database/
4  # 防止报错
5  unset DISPLAY
6  ./runInstaller -silent -ignorePrereq -responseFile
   /usr/local/oracle/db_install.rsp
7
8  # /home/database是安装包解压后的路径，此处根据安装包解压所在位置做修改，因人而异。
9      # runInstaller 是主要安装脚本
10     # -silent 静默模式
11     # -force 强制安装
12     # -ignorePrereq忽略warning直接安装
13     #-responseFile读取安装应答文件

```

- 出现以下提醒则表示安装成功

```

[root@localhost oracle]# su oracle
[oracle@localhost ~]$ clear
[oracle@localhost ~]$ cd database/
[oracle@localhost database]$ ls
doc  install  response  rpm  runInstaller  sshsetup  stage  welcome.html
[oracle@localhost database]$ unset DISPLAY
[oracle@localhost database]$ ./runInstaller -silent -ignorePrereq -responseFile /usr/local/oracle/db_install.rsp
正在启动 Oracle Universal Installer...

检查临时空间：必须大于 120 MB。  实际为 47407 MB    通过
检查交换空间：必须大于 150 MB。  实际为 3967 MB    通过
准备从以下地址启动 Oracle Universal Installer /tmp/OraInstall2020-02-02_12-33-00AM. 请稍候...[oracle@localhost database]$ 可以在以下位置找到本次安装会话的日志：
/usr/local/oraInventory/logs/installActions2020-02-02_12-33-00AM.log
以下配置脚本需要以 "root" 用户的身份执行。
#!/bin/sh
#要运行的 Root 脚本

/usr/local/oraInventory/orainstRoot.sh
/usr/local/oracle/product/11.2.0/db_1/root.sh
要执行配置脚本，请执行以下操作：
    1. 打开一个终端窗口
    2. 以 "root" 身份登录
    3. 运行脚本
    4. 返回此窗口并按 "Enter" 键继续

Successfully Setup Software.

```

- 安装成功后还需要执行以下命令

```

1  su root
2  密码：
3  cd /usr/local/oraInventory/
4  ./orainstRoot.sh
5  cd /usr/local/oracle/product/11.2.0/db_1/
6  ./root.sh

```

- 设置Oracle用户环境变量

```

1  # 切换至oracle账号
2  su oracle
3  # 编辑用户环境变量
4  echo 'export ORACLE_BASE=/usr/local/oracle
5  export ORACLE_HOME=$ORACLE_BASE/product/11.2.0/db_1
6  export LANG="zh_CN.UTF-8"
7  export NLS_LANG="SIMPLIFIED CHINESE_CHINA.AL32UTF8"
8  export NLS_DATE_FORMAT="yyyy-mm-dd hh24:mi:ss"
9  export PATH=$PATH:$ORACLE_HOME/bin ' >> ~/.bashrc
10 # 立即生效
11 source ~/.bashrc

```

- 配置监听以及监听的相关命令

```

1  # 静默方式配置监听
2  netca /silent/responseFile /usr/local/oracle/netca.rsp
3  # 启动监听
4  lsnrctl start
5  # 暂停监听
6  lsnrctl stop
7  # 查看监听
8  lsnrctl status
9  # 检查是否监听正常启动
10 netstat -tnulp | grep 1521
11 (No info could be read for "-p": geteuid()=1001 but you should be root.)
12 tcp6      0      0 :::1521          :::*              LISTEN

```

### 1.2.3 静默方式建立新库

- 新建 dbca.rsp

```

1  su oracle
2  vi /home/oracle/dbca.rsp
3  # -----
4  [GENERAL]
5  RESPONSEFILE_VERSION = "11.2.0"
6  # 必须指定为创建数据库
7  OPERATION_TYPE = "createDatabase"
8  [CREATEDATABASE]
9  # 数据库的Global database name
10 GDBNAME = "orcl"
11 # 数据库的实例名
12 SID = "orcl"
13 TEMPLATENAME = "General_Purpose.dbc"
14 STORAGETYPE=FS
15 # 指定数据文件存储的目录
16 DATAFILEDESTINATION =/usr/local/oracle/oradata
17 # 指定数据文件恢复的目录
18 RECOVERYAREADESTINATION=/usr/local/oracle/flash_recovery_area
19 # 指定字符集
20 CHARACTERSET = "AL32UTF8"
21 # 指定国家字符集
22 NATIONALCHARACTERSET= "AL16UTF16"
23 LISTENERS=LISTENER
24 TOTALMEMORY = "700"
25 # 指定sys用户密码[必填]

```

```

26 SYSPASSWORD = "123456"
27 # 指定system用户密码[必填]
28 SYSTEMPASSWORD = "123456"
29 # 指定使用自动内存管理
30 AUTOMATICMEMORYMANAGEMENT = "TRUE"
31 # -----

```

- 以静默方式开始创建数据库

```
1 dbca -silent -responseFile /home/oracle/dbca.rsp
```

- 修改 Oracle 用户的环境变量

```

1 vi ~/.bashrc
2 # -----
3 # 与上述安装的数据库SID一致
4 export ORACLE_SID=orcl
5 export ORACLE_OWNER=$ORACLE_SID
6 # -----
7
8 # 立即生效
9 source ~/.bashrc
10
11 # 修改监听文件
12 vi /usr/local/oracle/product/11.2.0/db_1/network/admin/listener.ora
13
14 # -----
15 # 添加SID相关配置，注意 SID_NAME区分大小写！
16 SID_LIST_LISTENER =
17     (SID_LIST =
18         (SID_DESC =
19             (ORACLE_HOME = /usr/local/oracle/product/11.2.0/db_1)
20             (SID_NAME = orcl)
21         )
22     )
23
24 LISTENER =
25     (DESCRIPTION_LIST =
26         (DESCRIPTION =
27             (ADDRESS = (PROTOCOL = IPC)(KEY = EXTPROC1521))
28             (ADDRESS = (PROTOCOL = TCP)(HOST = localhost)(PORT = 1521))
29         )
30     )
31
32 ADR_BASE_LISTENER = /usr/local/oracle
33 # -----
34
35 # 重启监听
36 lsnrctl stop
37 lsnrctl start

```

- 出现以下信息则表明服务启动成功



```

Copyright (c) 1991, 2009, Oracle. All rights reserved.

启动/usr/local/oracle/product/11.2.0/db_1/bin/tnslsnr: 请稍候...

TNSLSNR for Linux: Version 11.2.0.1.0 - Production
系统参数文件为/usr/local/oracle/product/11.2.0/db_1/network/admin/listener.ora
写入/usr/local/oracle/diag/tnslsnr/localhost/listener/alert/log.xml的日志信息
监听: (DESCRIPTION=(ADDRESS=(PROTOCOL=ipc) (KEY=EXTPROC1521)))
监听: (DESCRIPTION=(ADDRESS=(PROTOCOL=tcp) (HOST=localhost) (PORT=1521)))

正在连接到 (DESCRIPTION=(ADDRESS=(PROTOCOL=IPC) (KEY=EXTPROC1521)))
LISTENER 的 STATUS
-----
别名          LISTENER
版本          TNSLSNR for Linux: Version 11.2.0.1.0 - Production
启动日期      02-2月 -2020 13:11:01
正常运行时间  0 天 0 小时 0 分 0 秒
跟踪级别      off
安全性        ON: Local OS Authentication
SNMP          OFF
监听程序参数文件 /usr/local/oracle/product/11.2.0/db_1/network/admin/listener.ora
监听程序日志文件 /usr/local/oracle/diag/tnslsnr/localhost/listener/alert/log.xml
监听端点概要...
  (DESCRIPTION=(ADDRESS=(PROTOCOL=ipc) (KEY=EXTPROC1521)))
  (DESCRIPTION=(ADDRESS=(PROTOCOL=tcp) (HOST=localhost) (PORT=1521)))
服务摘要..
服务 "orcl" 包含 1 个实例。
  实例 "orcl", 状态 UNKNOWN, 包含此服务的 1 个处理程序...
命令执行成功
[oracle@localhost ~]$

```

## 1.2.4 开启远程连接

```

1  # 关闭centos7防火墙
2  systemctl stop firewalld.service
3  # 关闭开机启动
4  systemctl disable firewalld.service
5  # 安装iptables防火墙
6  yum install iptables-services
7  # 新增1521接口
8  iptables -A INPUT -p tcp --dport 1521 -j ACCEPT
9  service iptables save
10 # 设置开机启动
11 systemctl enable iptables
12 # 重启iptables服务
13 systemctl restart iptables.service

```

## 1.2.5 开机自动启动

```

1  vi /usr/local/oracle/product/11.2.0/db_1/bin/dbstart
2  # -----
3  # 修改ORACLE_HOME_LISTNER参数
4  ORACLE_HOME_LISTNER=$ORACLE_HOME
5  # -----
6
7  vi /usr/local/oracle/product/11.2.0/db_1/bin/dbshut
8  # -----
9  # 修改ORACLE_HOME_LISTNER参数
10 ORACLE_HOME_LISTNER=$ORACLE_HOME
11 # -----
12
13 vi /etc/oratab

```

```

14 # -----
15 # 修改为 :Y
16 orcl:/usr/local/oracle/product/11.2.0/db_1:Y
17 # -----

```

切换到 root 用户, 执行 `vi /etc/init.d/oracle`

```

1  #!/bin/sh
2  # chkconfig: 345 61 61
3  # description: Oracle 11g R2 AutoRun Servicmces
4  # /etc/init.d/oracle
5  # Run-level startup script for the Oracle Instance, Listener, and
6  # Web Interface
7  export ORACLE_BASE=/usr/local/oracle
8  export ORACLE_HOME=$ORACLE_BASE/product/11.2.0/db_1
9  export ORACLE_SID=orcl
10 export ORACLE_UNQNAME=$ORACLE_SID
11 export PATH=$ORACLE_HOME/bin:/user/sbin:$PATH
12 export LD_LIBRARY_PATH=$ORACLE_HOME/lib:$LD_LIBRARY_PATH
13 ORA_OWNR="oracle"
14 # if the executables do not exist -- display error
15 if [ ! -f $ORACLE_HOME/bin/dbstart -o ! -d $ORACLE_HOME ]
16 then
17 echo "Oracle startup: cannot start"
18 exit 1
19 fi
20 # depending on parameter -- startup, shutdown, restart
21 # of the instance and listener or usage display
22 case "$1" in
23 start)
24 # Oracle listener and instance startup
25 su $ORA_OWNR -lc $ORACLE_HOME/bin/dbstart
26 echo "Oracle Start Succesful!OK."
27 ;;
28 stop)
29 # Oracle listener and instance shutdown
30 su $ORA_OWNR -lc $ORACLE_HOME/bin/dbshut
31 echo "Oracle Stop Succesful!OK."
32 ;;
33 reload|restart)
34 $0 stop
35 $0 start
36 ;;
37 *)
38 echo $"Usage: `basename $0` {start|stop|reload|reload}"
39 exit 1
40 esac
41 exit 0

```

保存退出后执行以下命令

```

1 cd /etc/init.d
2 chmod +x oracle
3 chkconfig --add oracle
4 chkconfig oracle on

```

## 1.2.6 创建表空间与用户

- 进入 sqlplus

```
1 | sqlplus / as sysdba
```

- 创建表空间

```
1 | create tablespace e9 logging datafile '/usr/local/oracle/oradata/orcl/e9.dbf'  
   size 1024m autoextend on next 100m maxsize 10240m extent management local;
```

- 创建用户

```
1 | create user weaver identified by 123456 default tablespace e9;
```

- 5.4 给用户赋权限

```
1 | grant dba to weaver;
```

## 1.2.7 常见问题

- ORA-01034: ORACLE not available ORA-27101

```
1 | # Step1: 启动oracle监听  
2 | lsnrctl start  
3 | # Step2: 设置实例名, 实例名一般是"orcl"  
4 | set ORACLE_SID=orcl  
5 | # Step3: 启动oracle服务  
6 | # 进入 sql 控制台, 执行 startup命令, 如果被告知已经启动, 可以先输入shutdown immediate,  
   然后在输入startup  
7 | sqlplus / as sysdba  
8 | startup;
```

## 1.3 下载与安装 E9

安装包下载文档: [https://www.e-cology.com.cn/spa/document/index.jsp?id=3861088&newsDataType=fullSearch&router=1#/main/document/detail?\\_key=niim1h](https://www.e-cology.com.cn/spa/document/index.jsp?id=3861088&newsDataType=fullSearch&router=1#/main/document/detail?_key=niim1h)

将 ecology, jdk, resin 下载并放在同一目录下

### 1.3.1 E9 程序包介绍

- Ecology: E9 的主程序包
- Jdk: Java 运行程序
- Resin: Resin 中间件

### 1.3.2 Resin 基本配置

resin/conf/resin.xml

```

1 <!-- 修改jdk路径，可以是相对路径也可以是绝对路径 -->
2 <javac compiler="../jdk/bin/javac" args="-encoding UTF-8"/>
3
4 <!-- 修改ecology路径，可以是相对路径也可以是绝对路径 -->
5 <web-app id="/" root-directory="../ecology">

```

resin/conf/resin.properties

```

1 # 修改服务端口
2 app.http      : 80
3 app.https     : 8443

```

### 1.3.3 Resin-Win 配置

在 window 环境下，resin/resinstart.bat 为启动文件，可右键编辑内容，其中 java\_home 为 jdk 路径，这里使用相对路径即可，当然也可以使用绝对路径。单击 resinstart.bat 启动服务

```

1 set java_home=../jdk
2 resin.exe

```

### 1.3.4 Resin-Linux 配置

- 编译安装 resin

```

1 # prefix: resin路径
2 # with-java-home: jdk路径
3 cd /usr/local/weaver/resin
4 chmod +x ./configure
5 ./configure --prefix=/usr/local/weaver/resin --with-java-
  home=/usr/local/weaver/jdk --enable-64bit
6 make
7 make install

```

- 修改相关配置文件

```

1 # 修改 resin/bin/resin.sh 中的jdk路径
2 sed -i "s/jdk1.8.0_151/jdk/g" /usr/local/weaver/resin/bin/resin.sh
3 sed -i "s/weaver/local\\weaver/g" /usr/local/weaver/resin/bin/resin.sh
4 # 修改 resin.sh 路径
5 sed -i "s/weaver/local\\weaver/g"
  /usr/local/weaver/resin/bin/startresin.sh
6 sed -i "s/weaver/local\\weaver/g" /usr/local/weaver/resin/bin/stopresin.sh
7 # 修改 resin/conf/resin.xml 中的jdk和ecology路径
8 # jdk 必须是绝对路径，ecology可以是绝对路径也可以是相对 路径
9 sed -i
  "s/#INSTALLDIR#JDK\\bin\\javac\\/usr\\local\\weaver\\jdk\\bin\\java
  c/g" /usr/local/weaver/resin/conf/resin.xml
10 sed -i "s/#INSTALLDIR#ecology/..\\ecology/g"
  /usr/local/weaver/resin/conf/resin.xml

```

- 启动 resin，打开浏览器输入 http://127.0.0.1 即可使用 e9

## 1.4 Redis 部署以及配置

E9 默认使用 `ehcache2.x` 进行数据缓存, 支持使用 `redis` 进行数据缓存。

Redis 是完全开源免费的, 遵守BSD协议, 是一个高性能的key-value数据库。

### 1.4.1 window 安装 Redis

window安装 redis 比较简单, 下载安装包即可, 地址: <https://github.com/microsoftarchive/redis/releases>

下载安装包 `Redis-x64-3.2.100.msi`, 点击下一步即可~

## 3.2.100

 enricogior released this on 1 Jul 2016 · 1210 commits to 3.0 since this release





This is the first release of Redis on Windows 3.2.

This release is based on antirez/redis/3.2.1 plus some Windows specific fixes. It has passed all the standard tests but it hasn't been tested in a production environment.

Therefore, before considering using this release in production, make sure to test it thoroughly in your own test environment.

See the [release notes](#) for details.

#### ▼ Assets 4

 <a href="#">Redis-x64-3.2.100.msi</a>	5.8 MB
 <a href="#">Redis-x64-3.2.100.zip</a>	4.98 MB
 <a href="#">Source code (zip)</a>	
 <a href="#">Source code (tar.gz)</a>	

### 1.4.2 Linux 安装 Redis

- 安装编译环境

```
1 # ubuntu
2 sudo apt-get install make gcc
3 # centos
4 yum install make gcc
```

- 安装 redis

```
1 cd ~
2 # 下载 redis 源码
3 wget http://download.redis.io/releases/redis-5.0.6.tar.gz
4 # 解压
5 tar xzf redis-5.0.6.tar.gz
6 # 切换到解压目录
7 cd redis-5.0.6
8 # 切换到 root 权限
9 su root
10 # 编译安装
11 make PREFIX=/usr/local/redis install
```

- 配置和启动

redis.conf 是 redis 的配置文件，redis.conf 在 redis 源码目录下

拷贝配置文件到安装目录下

```
1 # 进入源码目录
2 cd ~/redis-5.0.6
3 # 拷贝配置文件至安装目录
4 cp ~/redis-5.0.6/redis.conf /usr/local/redis/redis.conf
```

- 前台启动

```
1 # 进入安装目录
2 cd /usr/local/redis/bin
3 ./redis-server
```

- 后端模式启动

修改 redis.conf 配置文件，daemonize 参数设置为 yes，表示以后端模式启动

```
##### GENERAL #####

# By default Redis does not run as a daemon. Use 'yes' if you need it.
# Note that Redis will write a pid file in /var/run/redis.pid when daemonized.
daemonize yes

# If you run Redis from upstart or systemd, Redis can interact with your
# supervision tree. Options:
# supervised no      - no supervision interaction
# supervised upstart - signal upstart by putting Redis into SIGSTOP mode
# supervised systemd - signal systemd by writing READY=1 to $NOTIFY_SOCKET
# supervised auto    - detect upstart or systemd method based on
#                       UPSTART_JOB or NOTIFY_SOCKET environment variables
# Note: these supervision methods only signal "process is ready."
#       They do not enable continuous liveness pings back to your supervisor.
supervised no
```

修改完毕后保存退出，重新启动 redis

```
1 # 进入安装目录
2 cd /usr/local/redis
3 # 已后端模式启动 redis
4 ./bin/redis-server ./redis.conf
```

### 1.4.3 E9 使用 Redis

ecology/WEB-INF/prop/weaver\_new\_session.properties，开启redis并修改相关配置，注意redis的密码一定要有，否则会报错！

```
1 # 1表示启用新的session方式，其他值表示不启用
2 status=1
3 # 用于调试的USERID
4 debugUsers=
5 # session id生成模式，1表示自定义生成模式（UUID模式），其他值表示中间件自定义生成模式
6 useCustomSessionId=1
7 # 同步频率设置（单位，秒）
8 # 主表同步频率
9 SessionTablesSync=2
10 # 明细表同步频率
11 SessionItemTablesSync=5
12 # 超时扫描频率
```

```
13 SessionOverTime=300
14 # 垃圾数据清理扫描频率
15 SessionLeak=3600
16
17 #启动模式，默认是数据库模式
18 # className=weaver.session.util.DBUtil
19 className=weaver.session.util.RedisSessionUtil
20 # redis ip
21 redisIp=127.0.0.1
22 # redis port
23 redisPort=6379
24 # redis password，注意密码必须设置，否则会报错！
25 redisPassword=123456
26 enableImmediatelySync=true
27 etoken=
```

使用 redis 可视化工具可以看到如下结果，则表明启动了redis



## 2. 集群部署

e9 集群部署与 e8 基本一致，只增加了一个 redis 配置。

集群方案有很多，比如：ec+nginx+weblogic，ec+nginx+resin，ec+F5+resin，本文只介绍 ec+nginx+resin

### 2.0 集群准备

集群部署需要准备：

- 一台文件服务器（NFS）服务器地址：192.168.44.143
- 一台主服务器（Ecology+MySQL+Nginx）服务器地址：192.168.44.139
- 一台从服务器（Ecology）服务器地址：192.168.44.144

### 2.1 文件服务器配置

#### 2.1.1 NFS 服务器部署

- 安装 nfs

```

1 yum install -y nfs-utils
2 # 设置开启启动
3 sudo systemctl enable rpcbind
4 sudo systemctl enable nfs

```

- 配置对外共享文件: `/etc/exports`

```

1 vi /etc/exports
2 # 在 /etc/exports 中添加以下内容
3 # 意思是将本地文件夹 /data 共享给192.168.44.139和144服务器
4 # 也可以用*号代替, 比如: /data *(rw, sync, no_root_squash)
5 # 意思是将本地文件夹 /data 共享到所有和这个文件服务器相通的机器
6 /data 192.168.44.139(rw, sync, no_root_squash)
7 /data 192.168.44.144(rw, sync, no_root_squash)

```

- 重启 nfs 服务

```

1 exportfs -rv
2 systemctl restart nfs
3 # 针对 rhel5
4 systemctl restart portmap
5 # 针对 rhel6
6 systemctl restart rpcbin

```

## 2.1.2 文件同步配置

切换至主 (从) 服务器操作

- 在需要共享的节点挂载共享文件到对应目录

```

1 # 主服务器: 192.168.44.139
2 yum install -y nfs-utils
3 # 创建挂载目录
4 mkdir /data
5 # 设置文件挂载
6 mount -t nfs 192.168.44.143:/data /data

```

- 将 ecology 以下目录拷贝到 `/data/ec9` 中, 一般情况下主服务器做拷贝即可。

```

1 # 备份文件, 注意这里使用mv而不是cp
2 cd /usr/local/weaver/ecology
3 mkdir -p nfs_bak/WEB-INF
4 # 移动以下目录至 nfs_bak 下
5 \mv album formmode mobilemode mobile email filesystem images
  images_face images_frame LoginTemplateFile messenger m_img others page
  upgrade wui nfs_bak
6 # 进入WEB-INF目录, 移动以下文件夹至 nfs_bak/WEB-INF 下
7 cd WEB-INF
8 \mv securityRule securityXML service lib/keys weaver_security_rules.xml
  weaver_security_config.xml hrmssettings.xml ../nfs_bak/WEB-INF
9
10 # 将相关文件拷贝到 /data 目录下
11 # 在/data中创建 ec/WEB-INF 目录
12 mkdir -p /data/ec/WEB-INF
13 \cp -rf /usr/local/weaver/ecology/nfs_bak /data/ec

```



```
14 # 以下两个文件是不存在的，需要手动创建
15 mkdir -p /data/ec/LoginTemplateFile /data/ec/others
```

- 创建软链接

```
1 ln -sf /data/ec/album /usr/local/weaver/ecology
2 ln -sf /data/ec/formmode /usr/local/weaver/ecology
3 ln -sf /data/ec/mobilemode /usr/local/weaver/ecology
4 ln -sf /data/ec/email /usr/local/weaver/ecology
5 ln -sf /data/ec/filesystem /usr/local/weaver/ecology
6 ln -sf /data/ec/images /usr/local/weaver/ecology
7 ln -sf /data/ec/images_face /usr/local/weaver/ecology
8 ln -sf /data/ec/images_frame /usr/local/weaver/ecology
9 ln -sf /data/ec/LoginTemplateFile /usr/local/weaver/ecology
10 ln -sf /data/ec/messenger /usr/local/weaver/ecology
11 ln -sf /data/ec/m_img /usr/local/weaver/ecology
12 ln -sf /data/ec/others /usr/local/weaver/ecology
13 ln -sf /data/ec/page /usr/local/weaver/ecology
14 ln -sf /data/ec/upgrade /usr/local/weaver/ecology
15 ln -sf /data/ec/wui /usr/local/weaver/ecology
16 ln -sf /data/ec/WEB-INF/securityRule /usr/local/weaver/ecology/WEB-INF
17 ln -sf /data/ec/WEB-INF/securityXML /usr/local/weaver/ecology/WEB-INF
18 ln -sf /data/ec/WEB-INF/service /usr/local/weaver/ecology/WEB-INF
19 ln -sf /data/ec/WEB-INF/lib/keys /usr/local/weaver/ecology/WEB-INF/lib/keys
20 ln -sf /data/ec/WEB-INF/weaver_security_rules.xml
   /usr/local/weaver/ecology/WEB-INF/weaver_security_rules.xml
21 ln -sf /data/ec/WEB-INF/weaver_security_config.xml
   /usr/local/weaver/ecology/WEB-INF/weaver_security_config.xml
22 ln -sf /data/ec/WEB-INF/hrmsettings.xml /usr/local/weaver/ecology/WEB-
   INF/hrmsettings.xml
```

### 2.1.3 缓存同步设置

- 配置 `/etc/hosts`：清空原有127默认配置，将集群各节点 `ip` 地址加入到 `hosts` 中

```
192.168.44.144 localhost localhost.localdomain localhost4 localhost4.localdomain4
::1 localhost localhost.localdomain localhost6 localhost6.localdomain6
```

- 修改 `ecology/WEB-INF/weaver.properties`，加入以下内容。注意，所有节点都需要配置！

```
1 # 主节点ip(集群中任意一个节点，但有且只能有一个)
2 MainControlIP = 192.168.44.139
3 # 本机ip:本机服务端口port
4 ip = 192.168.44.144:80
5 broadcast=231.12.21.132
6 syncType=http
7 # initial_hosts为参数为所有的应用服务器的节点的访问地址
8 # 格式: ip1:port,ip2:port （中间以逗号分隔）
9 initial_hosts= 192.168.44.139:80,192.168.44.144:80
```

## 2.2 部署 Nginx

- 在主服务器 (192.168.44.139) 下载安装 `nginx`

```

1 # 下载nginx安装包
2 yum install -y wget
3 wget https://nginx.org/download/nginx-1.17.8.tar.gz
4 # 下载编译依赖包
5 yum install -y gcc-c++ pcre pcre-devel zlib zlib-devel openssl openssl-
  devel
6 tar -zxvf nginx-1.17.8.tar.gz
7 cd nginx-1.17.8
8 # 编译安装
9 ./configure --prefix=/usr/local/nginx
10 make && make install
11 # 查看安装目录
12 whereis nginx
13 # nginx: /usr/local/nginx

```

- 编辑 `/usr/local/nginx/conf/nginx.conf`

```

1 worker_processes 1;
2
3 events {
4     worker_connections 1024;
5 }
6
7 http {
8     upstream ecologycluster {
9         ip_hash;
10        # e-cology服务器地址1
11        server 192.168.44.139:80;
12        # e-cology服务器地址2
13        server 192.168.44.144:80;
14        # ...
15    }
16    include mime.types;
17    default_type application/octet-stream;
18
19    sendfile on;
20
21    keepalive_timeout 65;
22    server {
23        listen 8889;
24        server_name e9;
25        location / {
26            root html;
27            index index.html index.htm index.jsp;
28            proxy_pass http://ecologycluster;
29            proxy_next_upstream http_502 http_504 http_500 http_404
http_403 error timeout invalid_header;
30            proxy_redirect off;
31            proxy_set_header X-Forwarded-For
$proxy_add_x_forwarded_for;
32            proxy_set_header X-Real-IP $remote_addr;
33            proxy_set_header Host $http_host;
34            proxy_read_timeout 3600;
35            proxy_send_timeout 3600;
36            access_log off;
37            send_timeout 300;

```

```
38 |     }
39 | }
40 | }
```

- 启动 `nginx`

```
1 | cd /usr/local/nginx/sbin
2 | ./nginx -c /usr/local/nginx/conf/nginx.conf
```

- 打开浏览器输入 `http://192.168.44.139:8889`，注意这里访问的是 `nginx` 地址！

## 2.3 故障测试

- 将两台服务器都停掉，再访问 `http://192.168.44.139:8889`
- 开启从服务器，将主服务器（192.168.44.139）停掉，再访问 `http://192.168.44.139:8889`
- 开启主服务器，将从服务器（192.168.44.144）停掉，再访问 `http://192.168.44.139:8889`

# 3. Docker 部署

Docker 部署适用于客户服务器系统为 `linux`，`window` 系统不推荐使用。

Docker 部署不是成熟方案，并且需要一定技术支撑，该方案量力而为。

下面使用 `CentOS` 系统进行 `Docker-E9` 部署

## 3.1 Docker 安装

```
1 | # step 1: 安装必要的一些系统工具
2 | sudo yum install -y yum-utils device-mapper-persistent-data lvm2
3 | # step 2: 添加软件源信息
4 | sudo yum-config-manager --add-repo http://mirrors.aliyun.com/docker-
   | ce/linux/centos/docker-ce.repo
5 | # Step 3: 更新并安装Docker-CE
6 | sudo yum makecache fast
7 | sudo yum -y install docker-ce
8 | # Step 4: 开启Docker服务
9 | sudo service docker start
```

## 3.2 配置加速器

```
1 | # 配置阿里云加速器
2 | echo "{
3 |     \"registry-mirrors\": [\"https://xh36jugz.mirror.aliyuncs.com\"]
4 | }">/etc/docker/daemon.json
5 | # 重启docker
6 | sudo systemctl daemon-reload
7 | sudo systemctl restart docker
```

## 3.3 制作 E9 镜像

创建目录：`mkdir /usr/local/weaver`

### 3.3.1 使用 `ubuntu` 依赖

- 上传 `EcoLogy9.00.xxxx.xx.zip`, `jdk-8u231-linux-x64.tar.gz`, `Resin-4.0.58.zip` 至 `/usr/local/weaver` 目录
- 解压文件

```
1 # 安装解压工具, 如果已经安装可省略
2 yum install unzip
3 # 解压文件夹, ecology按照具体的版本号
4 unzip EcoLogy9.00.xxxx.xx.zip
5 unzip Resin-4.0.58.zip
6 unzip jdk-8u231-linux-x64.tar.gz
7 # 对解压出来的文件夹重命名
8 mv Resin resin
9 mv jdk1.8.0_231 jdk
```

- 修改 resin 相关配置: 查看 [1.3.4 Resin-Linux 配置](#)
- 创建 `ubuntu` 依赖的 `Dockerfile`

```
1 echo 'FROM ubuntu
2 COPY ecology/ /usr/local/weaver/ecology/
3 COPY resin/ /usr/local/weaver/resin/
4 COPY jdk/ /usr/local/weaver/jdk/
5 EXPOSE 8080
6 RUN chmod +x /usr/local/weaver/resin/bin/resin.sh
7 CMD ["/bin/sh", "-c", "/usr/local/weaver/resin/bin/resin.sh
  console"]'>/usr/local/weaver/Dockerfile
```

### 3.3.2 使用 `resin:e9` 依赖

- 使用该依赖则只需上传 `EcoLogy9.00.xxxx.xx.zip` 至 `/usr/local/weaver` 目录即可
- 解压文件

```
1 # 安装解压工具, 如果已经安装可省略
2 yum install unzip
3 # 解压文件夹, ecology按照具体的版本号
4 unzip EcoLogy9.00.xxxx.xx.zip
```

- 创建 `resin:e9` 依赖的 `Dockerfile`

```
1 echo 'FROM registry.cn-hangzhou.aliyuncs.com/weaver/resin:e9
2 COPY ecology/ /usr/local/weaver/ecology/'>Dockerfile
```

### 3.3.3 构建镜像

- 路径切换到 `/usr/local/weaver` 下, 执行镜像构建命令, 注意末尾的 `.` 号是必须的!

```
1 cd /usr/local/weaver
2 # ec系统瘦身
3 rm -rf $(find /usr/local/weaver/ecology -name="*.map.js")
4 # 修改初始验证码
5 sed -i 's/wEAver2018/1/g' /usr/local/weaver/ecology/WEB-INF/code.key
6 # 开始构建镜像
7 docker build -t e9:191202 .
```

- 出现以下信息表示镜像构建成功

```
[root@localhost weaver]# ls
Dockerfile  Ecology9.00.1912.02.zip  jdk-8u231-linux-x64.tar.gz  Resin-4.0.58.zip
ecology     jdk                      resin
[root@localhost weaver]# docker build -t e9:191202 .
Sending build context to Docker daemon 7.553GB
Step 1/7 : FROM ubuntu
--> ccc6e87d482b
Step 2/7 : COPY ecology/ /usr/local/weaver/ecology/
--> Using cache
--> 82364e9670c9
Step 3/7 : COPY resin/ /usr/local/weaver/resin/
--> Using cache
--> 49b99397c7d8
Step 4/7 : COPY jdk/ /usr/local/weaver/jdk/
--> Using cache
--> 1d1db3755ab4
Step 5/7 : EXPOSE 8080
--> Using cache
--> 173ec40f4eb4
Step 6/7 : RUN chmod +x /usr/local/weaver/resin/bin/resin.sh
--> Using cache
--> d2bcf68c9d2b
Step 7/7 : CMD [/bin/sh, -c, /usr/local/weaver/resin/bin/resin.sh console]
--> Using cache
--> a3c0ef84c9a6
Successfully built a3c0ef84c9a6
Successfully tagged e9:191202
[root@localhost weaver]# docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
e9	191202	a3c0ef84c9a6	14 minutes ago	5.05GB
ubuntu	latest	ccc6e87d482b	2 weeks ago	64.2MB

```
[root@localhost weaver]#
```

### 3.3.4 镜像瘦身

`e-cology9` 构建镜像时，会存在一个让客户吐槽的问题，那就是构建出的镜像过大，从图中可以看出，`e9:191202` 镜像包已经达到 `5.05GB` 大小，下面介绍几个镜像瘦身的方案，大约能瘦身 `1-2G` 左右

通过软件分析 `e-cology9` 程序，发现文件扩展名为 `.map`，`.jar`，`.js` 的文件占比最大。其中 `.jar` 为后端文件，`.js` 为前端文件，这两个不要轻易的进行删减！那么我们可以动的只剩下 `.map` 文件了。

树查看	文件查看	关于	父级百分比	大小	分配	项	扩展名	文件类型	百分比	大小	分配	文件
文件类	D:\Weaver\ecology		100.0 %	3.6 GB	3.7 GB	71	.map	MAP 文件	28.7 %	1.0 GB	1.0 GB	1,145
spa			30.7 %	1.1 GB	1.1 GB		.jar	Executable Jar File	17.5 %	651.6 MB	653.0 MB	760
WEB-INF			19.1 %	710.5 MB	715.6 MB		.js	JavaScript 源文件	15.9 %	589.5 MB	607.3 MB	12,492
cloudstore			5.6 %	207.3 MB	212.8 MB		.zip	压缩(zipped)文件	12.0 %	446.5 MB	448.9 MB	1,279
integration			5.5 %	204.6 MB	206.5 MB		.exe	应用程序	4.0 %	147.7 MB	147.7 MB	50
resource			4.5 %	166.1 MB	166.1 MB		.png	PNG 文件	3.1 %	114.9 MB	130.3 MB	12,677
page			3.9 %	144.1 MB	145.6 MB		.mp4	MP4 - MPEG-4 电	2.7 %	99.3 MB	99.3 MB	2
ecode			3.7 %	136.8 MB	137.8 MB		.jsp	JSP 文件	2.7 %	99.0 MB	115.3 MB	8,929
formmode			3.2 %	119.2 MB	127.1 MB		.rar	RAR 文件	1.8 %	66.0 MB	66.0 MB	15
filesystem			2.3 %	84.3 MB	86.0 MB		.css	层叠样式表文档	1.4 %	52.4 MB	58.1 MB	3,246
messenger			2.2 %	82.4 MB	82.8 MB		.xml	XML 源文件	1.3 %	47.9 MB	49.1 MB	776
weaverplugin			1.9 %	70.6 MB	70.6 MB		.sql	SQL 源文件	1.2 %	43.8 MB	55.4 MB	10,517
wui			1.6 %	60.0 MB	65.2 MB		.log	LOG 文件	1.0 %	37.0 MB	37.3 MB	146
js			1.5 %	56.9 MB	65.2 MB		.jpg	JPG 文件	0.9 %	33.0 MB	34.0 MB	684
mobilemode			1.5 %	56.6 MB	65.4 MB		.doc	DOC 文档	0.7 %	25.6 MB	25.6 MB	17
							.msi	Windows Installer	0.6 %	22.3 MB	22.3 MB	3

关于 `.map.js` 文件介绍：

`source map` 文件是 `js` 文件压缩后，文件的变量名替换对应、变量所在位置等元信息数据文件，一般这种文件和 `min.js` 主文件放在同一个目录下。比如压缩后原变量是 `map`，压缩后通过变量替换规则可能会被替换成 `a`，这时 `source map` 文件会记录下这个 `mapping` 的信息，这样的好处就是说，在调试的时候，如果有一些 `js` 报错，那么浏览器会通过解析这个 `map` 文件来重新 `merge` 压缩后的 `js`，使开发者可以用未压缩前的代码来调试，这样会给我们带来很大的方便！

换句话说以 `.map` 后缀的文件，唯一的作用是方便在开发过程中进行调试，但是在生产环境中，可以被删除。

OK, 删除掉 `.map` 文件后, 整个程序的体积瞬间少了 1.0GB 左右, 接下来还有一个能删除的目录, 那就是 `ecology/data`, 这个目录下存放的是数据库初始化的 `sql` 文件, 从图中可以看出这些 `sql` 文件差不多占 200M 左右, 个人建议第一次构建镜像时不要删除这些文件, 等待数据库初始化完毕, 二次构建镜像时把这些文件删除, 这样 `e-ecology9` 镜像体积又能瘦下来了~

```
[root@localhost data]# ll -ll
总用量 2524
drwxr-xr-x. 2 root root 229376 10月 23 15:26 DM
drwxr-xr-x. 2 root root 331776 10月 23 15:26 Mysql
drwxr-xr-x. 2 root root 532480 10月 23 15:26 Oracle
drwxr-xr-x. 2 root root 528384 10月 23 15:26 SQLServer
drwxr-xr-x. 2 root root 225280 10月 23 15:26 ST
```

### 3.4 安装 docker-compose

百度云盘下载: <https://pan.baidu.com/s/1bEGXrwbQo32Gu8UbXgyFQQ>

下载 `docker-compose` 上传至 `/usr/local/bin/` 下, 然后执行以下命令

```
1 | chmod +x /usr/local/bin/docker-compose
```

### 3.5 启动 Ecology

#### 3.5.1 创建 `docker-compose.yml`

```
1 version: '3.1'
2 services:
3   ecology:
4     restart: always
5     image: e9-mysql:190902
6     container_name: e9
7     ports:
8       - 8080:80
9     volumes:
10      - ./ecology/album:/usr/local/weaver/ecology/album
11      - ./ecology/formmode:/usr/local/weaver/ecology/formmode
12      - ./ecology/messenger:/usr/local/weaver/ecology/messenger
13      - ./ecology/mobilemode:/usr/local/weaver/ecology/mobilemode
14      - ./ecology/mobile:/usr/local/weaver/ecology/mobile
15      - ./ecology/email:/usr/local/weaver/ecology/email
16      - ./ecology/filesystem:/usr/local/weaver/ecology/filesystem
17      - ./ecology/images_face:/usr/local/weaver/ecology/images_face
18      - ./ecology/images_frame:/usr/local/weaver/ecology/images_frame
19      - ./ecology/license:/usr/local/weaver/ecology/license
20      - ./ecology/log:/usr/local/weaver/ecology/log
21      -
22      - ./ecology/LoginTemplateFile:/usr/local/weaver/ecology/LoginTemplateFile
23      - ./ecology/m_img:/usr/local/weaver/ecology/m_img
24      - ./ecology/wui:/usr/local/weaver/ecology/wui
25      - ./ecology/others:/usr/local/weaver/ecology/others
26      - ./ecology/page:/usr/local/weaver/ecology/page
27      - ./ecology/upgrade:/usr/local/weaver/ecology/upgrade
28      - ./ecology/WEB-INF/securityRule:/usr/local/weaver/ecology/WEB-INF/securityRule
29      - ./ecology/WEB-INF/securityXML:/usr/local/weaver/ecology/WEB-INF/securityXML
```

```

29 - ./ecology/WEB-INF/service:/usr/local/weaver/ecology/WEB-INF/service
30 - ./ecology/WEB-INF/prop:/usr/local/weaver/ecology/WEB-INF/prop
31 - ./resin/log:/usr/local/weaver/resin/log
32 - ./resin/logs:/usr/local/weaver/resin/logs
33 environment:
34     TZ: Asia/Shanghai

```

### 3.5.2 数据持久化说明

在 `docker` 中，无论是原生的 `-v` 参数也好，还是 `docker-compose` 的 `volumes` 参数也罢。在启动容器时，会将宿主机上的相关目录复制到容器内部。以 `ecology/album` 目录为例，当宿主主机上 `./ecology/album` 目录不存在，或者该目录下不存在任何文件，那么启动容器后，使用 `exec` 命令登陆容器后会发现容器内的 `/usr/local/ecology/album` 目录下的所有文件都被清空！

```

[root@localhost e9]# docker run -itd -v /usr/local/docker/e9/ecology/album:/usr/local/weaver/ecology/album --name=e9 e9-mysql:190
f9dbadece031102922b919418a0ad3e6f0e724df7cb0e52b7994fe096d79f625
[root@localhost e9]# cd /usr/local/docker/e9/ecology/
[root@localhost ecology]# ls
album
[root@localhost ecology]# cd album/
[root@localhost album]# docker exec -it f9dbadece031102922b919418a0ad3e6f0e724df7cb0e52b7994fe096d79f625 bash
root@f9dbadece031:/# cd /usr/local/weaver/ecology/album/
root@f9dbadece031:/usr/local/weaver/ecology/album# ls
root@f9dbadece031:/usr/local/weaver/ecology/album#

```

登陆到容器中

album目录下的文件被清空

**总结：在启动容器时，需要手动创建所要持久化的目录，并且将相关目录下的文件 copy 到对应目录中**

如果使用已经构建好的镜像，不确定镜像内的文件是什么。那么可以采用以下方案：

1. 启动容器时先不挂载数据卷：`docker run -itd --name=e9 e9-mysql:190902`
2. 使用 `cp` 命令将容器内的文件 `copy` 到宿主机中：`sudo docker cp containerID:container_path host_path`

```

[root@localhost docker]# docker run -itd --name=e9 e9-mysql:190902
55ffca40ce55ecbb19eb743f63de044b0e68202f2a1dd8715083514a918e5569
[root@localhost docker]# cd e9
[root@localhost e9]# ls
docker-compose.yml
[root@localhost e9]# mkdir -p ecology/album
[root@localhost e9]# docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS              NAMES
55ffca40ce55       e9-mysql:190902    "/bin/sh -c '/usr/lo..." 23 seconds ago      Up 22 seconds      8888/tcp           e9
8b691cc04f17       mysql              "docker-entrypoint.s..." 14 hours ago        Up 8 hours         0.0.0.0:3306->3306/tcp, 33060/tcp  mysql_db_1
[root@localhost e9]# docker cp 55ffca40ce55:/usr/local/weaver/ecology/album/ /usr/local/docker/e9/ecology/
[root@localhost e9]# ls
docker-compose.yml  ecology
[root@localhost e9]# cd ecology/
[root@localhost ecology]# ls
album
[root@localhost ecology]# cd album/
[root@localhost album]# ls
AlbumBlankTab.jsp      AlbumList.jsp          AlbumSubcompany.jsp    ImageUploader0.jsp     PhotoSearchMenu.jsp    UImageUploaderN.cab
AlbumBrowser.jsp       AlbumListTab.jsp       AlbumSubcompanyTreeData.jsp  img                    PhotoSearchMenuTree.jsp  UImageUploaderXPN.cab
AlbumBrowserTreeNodes.jsp  AlbumMenu.jsp          AlbumSubcompanyTree.jsp  js                     PhotoSearchOperation.jsp  UploadFolder
AlbumFolderAdd.jsp       AlbumMenuTree.jsp      AlbumToggle.jsp         PhotoOperation.jsp     PhotoSearchResult.jsp
AlbumFolderEdit.jsp      AlbumSizeBatchEdit.jsp flash                   PhotoReloadInfo.jsp   PhotoView.jsp
AlbumFolderOperation.jsp  AlbumSubcompanyAll.jsp Frame.jsp               PhotoReview.jsp       ReviewOperation.jsp
AlbumFolderTree.jsp      AlbumSubcompanyAllTab.jsp ImageUploadDealN.jsp    PhotoSearch.jsp        slider

```

### 3.5.3 启动服务

- 执行 `docker-compose up` 启动服务，出现以下信息表示服务启动成功



```

e9 | [20-02-03 03:36:39.565] {main}
e9 | [20-02-03 03:36:39.702] {main} Table[mnode:2,/usr/local/weaver/resin/resin-data/app-0/distcache/mnode.db] validating indexes due to unclean shutdown.
e9 | [20-02-03 03:36:39.762] {main} Table[data:3,/usr/local/weaver/resin/resin-data/app-0/distcache/data.db] validating indexes due to unclean shutdown.
e9 | [20-02-03 03:36:39.778] {main}
e9 | [20-02-03 03:36:39.778] {main} resin.home = /usr/local/weaver/resin/
e9 | [20-02-03 03:36:39.778] {main} resin.root = /usr/local/weaver/resin/
e9 | [20-02-03 03:36:39.779] {main} resin.conf = /usr/local/weaver/resin/conf/resin.xml
e9 | [20-02-03 03:36:39.779] {main}
e9 | [20-02-03 03:36:39.779] {main} server = 127.0.0.1:6800 (app:app-0)
e9 | [20-02-03 03:36:39.779] {main} stage = production
e9 | Connection refused (Connection refused)
e9 | [20-02-03 03:37:13.014] {main} In-place class redefinition (HotSwap) is available.
e9 | [20-02-03 03:37:19.588] {MonitorCheckTask-1580672239516} check and start monitor begin...
e9 | [20-02-03 03:37:19.616] {MonitorCheckTask-1580672239516} check if require to copy and init monitor files...
e9 | [20-02-03 03:37:19.617] {MonitorCheckTask-1580672239516} start monitor process...
e9 | [20-02-03 03:37:19.803] {MonitorCheckTask-1580672239516} no require to copy and init monitor files...
e9 | [20-02-03 03:37:19.803] {MonitorCheckTask-1580672239516} updateRulePath:/usr/local/weaver/resin/monitor/resin
/app/update/rules
e9 | [20-02-03 03:37:19.803] {MonitorCheckTask-1580672239516} ruleUpdateSourcePath:/usr/local/weaver/ecology/WEB-INF/monitorRuleUpdate
e9 | [20-02-03 03:37:19.817] {MonitorCheckTask-1580672239516} check and start monitor end...
e9 | [20-02-03 03:37:24.028] {resin-47} WebApp[production/webapp/default/ROOT] active
e9 | [20-02-03 03:37:24.029] {main} Host[production/host/default] active
e9 | [20-02-03 03:37:24.045] {main} ServletService[id=app-0,cluster=app] active
e9 | [20-02-03 03:37:24.045] {main}
e9 | [20-02-03 03:37:24.045] {main} http listening to *:80
e9 | [20-02-03 03:37:27.330] {main} https listening to *:8443
e9 | [20-02-03 03:37:27.330] {main}
e9 | [20-02-03 03:37:27.418] {main} Resin[id=app-0] started in 49248ms

```

- 打开浏览器输入：`http://ip:8080`

## 4. 移动平台部署

移动平台可以通过泛微官方地址：

<https://emobile.weaver.com.cn/emp/download>

### 4.1 常规部署

关于 E-mobile7 常规部署参考 [泛微移动平台安装手册](#) 即可！

### 4.2 独立部署

E-Mobile7 包含 `em7`，`redis`，`mysql`，`active-mq`，`tomcat` 五大组成部分。其中 `em7` 依赖于 `tomcat` 启动。其他部分均可独立部署。也就是说 E-Mobile7 可以拆分成四大部分独立部署。

相关配置可查看：<https://yq.weaver.com.cn/eb/ga/view/#/question/5fb7e8e6124948129b1bfada0737f9f6>

### 4.3 Docker部署

#### 4.3.1 镜像制作

- 下载解压安装包和补丁包



根据操作系统类型选择以下安装包、仅第一次安装使用

由于某些Windows环境下Redis与Mysql存在不稳定性，所以强烈建议优先选择Linux版



Linux64 版

版本: V7.0.7 运行环境: Linux64位 Red Hat/CentOS 6及以上  
MD5: 7895a818b278239b6ed349748f231660



Windows64 版

版本: V7.0.7 运行环境: Win64位 2012及以上  
MD5: 4baaf125531db290f3bd065f3134c8a8

以下升级包适用于移动平台任意版本升级



移动平台补丁包

版本: 20200109

MD5: fcc0f636500c57b434d367f788e35332

```
[root@localhost emp]# ls
appsvr  data  emp_install_linux64_20191226.tar.gz  jdk  patch  start.sh  uninstall.sh
cachesvr  dbsvr  install.sh  msgsvr  restart.sh  stop.sh  work
```

```
1 tar -zxvf emp_install_linux64_20191226.tar.gz
2 # 如果已经安装unzip则跳过
3 # yum install unzip
4 unzip -o emp_patch_20200109.zip
```

- 创建 `/usr/local/docker/emp`，将 `appsvr`，`jdk`，`work` copy到该目录下

```
1 mkdir -p /usr/local/docker/emp
2 \cp -r appsvr /usr/local/docker/emp
3 \cp -r jdk /usr/local/docker/emp
4 \cp -r work /usr/local/docker/emp
```

- 使用 `ubuntu` 依赖的 `Dockerfile`

```
1 FROM ubuntu
2 COPY appsvr/tomcat /usr/local/emp/tomcat/
3 COPY work/ /usr/local/emp/work/
4 COPY start.sh /usr/local/emp/
5 COPY jdk/ /usr/local/jdk/
6 ENV
7 PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/local/jdk/bin:/usr/local/emp/tomcat/bin:/usr/local/emp/tomcat/bin
7 ENV TZ=Asia/Shanghai
8 ENV JAVA_HOME=/usr/local/jdk
9 ENV CLASSPATH=/usr/local/jdk/lib/dt.jar:/usr/local/jdk/lib/tools.jar
10 ENV CATALINA_HOME=/usr/local/emp/tomcat
11 ENV CATALINA_HOME_BASE=/usr/local/emp
12 ENV LC_ALL=zh_CN.gbk
13 ENV LANG=zh_CN.gbk
14 EXPOSE 8095
15 EXPOSE 8999
16 EXPOSE 9443
```

```
17 RUN [ "/bin/sh", "-c", "chmod +x /usr/local/emp/start.sh" ]
18 CMD [ "/bin/sh", "-c", "/usr/local/emp/start.sh" ]
```

- 下载 [start.sh](#) 至 `/usr/local/docker/emp`
- 创建 `/usr/local/docker/emp/docker-compose.yml`

```
1  version: '3.1'
2  services:
3    ecology:
4      build:
5        context: .
6        dockerfile: Dockerfile
7      restart: always
8      image: em7
9      container_name: em7
10     ports:
11       - 8999:8999
12     environment:
13       EMP_ACTIVEMQ_ENABLED: "true"
14       EMP_ACTIVEMQ_IP: "192.168.44.155"
15       EMP_ACTIVEMQ_PORT: 61616
16       EMP_ACTIVEMQ_USER: "system"
17       EMP_ACTIVEMQ_PASSWORD: "manager"
18       EMP_JTA_ENABLED: "false"
19       EMP_DB_TYPE: "mysql"
20       EMP_DB_NAME: "emp_app"
21       EMP_DB_IP: "192.168.44.155"
22       EMP_DB_PORT: 3306
23       # oracle 使用 EMP_DB_SID: orcl
24       EMP_DB_USERNAME: "root"
25       EMP_DB_PASSWORD: 123456
26       EMP_REDIS_ENABLED: "true"
27       EMP_REDIS_HOST: "192.168.44.155"
28       EMP_REDIS_PORT: 6381
29       EMP_REDIS_PASSWORD: "WEAVERemobile7*()"
30       EMP_REDIS_DB: 0
31       # # 集群配置
32       # 集群节点ID:全局唯一
33       EMP_CLUSTER_NODE_ID: 1
34       # CDN地址
35       EMP_CDN_URL:
36       # session存储介质，支持none、jdbc和redis，其中none表示session存储在servlet容
器（即原生session）
37       # jdbc表示session存储在数据库，redis表示session存储在redis里面（需配置redis数
据源）
38       EMP_SESSION_STORE_TYPE: "none"
39       # session有效期默认7天
40       EMP_SESSION_TIMEOUT: "7d"
41       # 基本信息缓存方式，支持none、ehcache和redis，其中none表示不缓存数据，ehcache
表示数据缓存在ehcache里面，redis表示数据缓存在redis里面（需配置redis数据源）
42       EMP_BASE_CACHE_TYPE: "ehcache"
43       # 操作日志保留期限（天），小于或者等于0表示不清理
44       EMP_LOG_RETAIN_DAYS: 30
45       # 从OA下载的文件保留期限（天），小于或者等于0表示不清理
46       EMP_DOWNLOAD_RETAIN_DAYS: 10
```

- 构建镜像命令： `docker-compose build`

```
[root@localhost emp]# ls
appsrvr  docker-compose.yml  Dockerfile  jdk  start.sh  work
[root@localhost emp]# pwd
/usr/local/docker/emp
[root@localhost emp]#
```

### 4.3.2 启动 Ecology

- 使用命令： `docker-compose up`

```
em7 | 20:49:30.049 [main] DEBUG org.apache.commons.configuration.ConfigurationUtils - Loading configuration from the context classpath (config/application-inner-custom.properties)
em7 | 20:49:30.049 [main] DEBUG org.apache.commons.configuration.ConfigurationUtils - ConfigurationUtils.locate(): base is null, name is config/application-inner-custom.properties
em7 | 20:49:30.049 [main] DEBUG org.apache.commons.configuration.DefaultFileSystem - Could not locate file config/application-inner-custom.properties at null: no protocol: config/application-inner-custo
m.properties
em7 | 20:49:30.049 [main] DEBUG org.apache.commons.configuration.ConfigurationUtils - Loading configuration from the context classpath (config/application-inner-custom.properties)
em7 | 20:49:30.049 [main] DEBUG org.apache.commons.configuration.ConfigurationUtils - ConfigurationUtils.locate(): base is null, name is config/application-inner-custom.properties
em7 | 20:49:30.049 [main] DEBUG org.apache.commons.configuration.DefaultFileSystem - Could not locate file config/application-inner-custom.properties at null: no protocol: config/application-inner-custo
m.properties
em7 | 20:49:30.050 [main] DEBUG org.apache.commons.configuration.ConfigurationUtils - Loading configuration from the context classpath (config/application-inner-custom.properties)
em7 | 20:49:30.050 [main] DEBUG org.apache.commons.configuration.ConfigurationUtils - ConfigurationUtils.locate(): base is null, name is config/application-inner-custom.properties
em7 | 20:49:30.050 [main] DEBUG org.apache.commons.configuration.DefaultFileSystem - Could not locate file config/application-inner-custom.properties at null: no protocol: config/application-inner-custo
m.properties
em7 | 20:49:30.050 [main] DEBUG org.apache.commons.configuration.ConfigurationUtils - Loading configuration from the context classpath (config/application-inner-custom.properties)
em7 | 20:49:30.050 [main] DEBUG org.apache.commons.configuration.ConfigurationUtils - ConfigurationUtils.locate(): base is null, name is config/application-inner-custom.properties
em7 | 20:49:30.050 [main] DEBUG org.apache.commons.configuration.DefaultFileSystem - Could not locate file config/application-inner-custom.properties at null: no protocol: config/application-inner-custo
m.properties
em7 | 20:49:30.050 [main] DEBUG org.apache.commons.configuration.ConfigurationUtils - Loading configuration from the context classpath (config/application-inner-custom.properties)
em7 | 20:49:30.050 [main] DEBUG org.apache.commons.configuration.ConfigurationUtils - ConfigurationUtils.locate(): base is null, name is config/application-inner-custom.properties
em7 | 20:49:30.050 [main] DEBUG org.apache.commons.configuration.DefaultFileSystem - Could not locate file config/application-inner-custom.properties at null: no protocol: config/application-inner-custo
m.properties
em7 | 20:49:30.051 [main] DEBUG org.apache.commons.configuration.ConfigurationUtils - Loading configuration from the context classpath (config/application-inner-custom.properties)
em7 | 20:49:30.051 [main] INFO org.apache.commons.configuration.PropertiesConfiguration - Reloading configuration. URL is file:///usr/local/emp/tomcat/webapps/ROOT/WEB-INF/classes/config/application-inner
-custom.properties
em7 | 20:49:30.051 [main] DEBUG org.apache.commons.configuration.ConfigurationUtils - ConfigurationUtils.locate(): base is null, name is config/application-inner-custom.properties
em7 | 20:49:30.051 [main] DEBUG org.apache.commons.configuration.DefaultFileSystem - Could not locate file config/application-inner-custom.properties at null: no protocol: config/application-inner-custo
m.properties
em7 | 20:49:30.051 [main] DEBUG org.apache.commons.configuration.ConfigurationUtils - Loading configuration from the context classpath (config/application-inner-custom.properties)
em7 | 20:49:30.052 [main] DEBUG org.apache.commons.configuration.ConfigurationUtils - ConfigurationUtils.locate(): base is null, name is config/application-inner-custom.properties
em7 | 20:49:30.052 [main] DEBUG org.apache.commons.configuration.DefaultFileSystem - Could not locate file config/application-inner-custom.properties at null: no protocol: config/application-inner-custo
m.properties
em7 | 20:49:30.052 [main] DEBUG org.apache.commons.configuration.ConfigurationUtils - Loading configuration from the context classpath (config/application-inner-custom.properties)
em7 | 20:49:30.052 [main] DEBUG org.apache.commons.configuration.PropertiesConfiguration - Base path set to file:///usr/local/emp/tomcat/webapps/ROOT/WEB-INF/classes/config/application-inner-custom.prop
erties
em7 | 11-Feb-2020 20:50:00.144 INFO [main] org.apache.catalina.startup.HostConfig.deployDirectory Deployment of web application directory [/usr/local/emp/tomcat/webapps/ROOT] has finished in [38,146] ms
em7 | 11-Feb-2020 20:50:00.201 INFO [main] org.apache.coyote.AbstractProtocol.start Starting ProtocolHandler ["http-nio-8999"]
em7 | 11-Feb-2020 20:50:00.214 INFO [main] org.apache.coyote.AbstractProtocol.start Starting ProtocolHandler ["https-nio-9443"]
em7 | 11-Feb-2020 20:50:00.222 INFO [main] org.apache.catalina.startup.Catalina.start Server startup in [38,272] milliseconds
```

- 打开浏览器，输入 `http://ip:port` 即可进入 E-Mobile7 后台！