# 金蝶多维数据库运维指南

**V1.5**

|  |  |  |  |
| --- | --- | --- | --- |
| 版本 | 日期 | 说明 | 作者 |
| V1.0 | 2020-12-15 | 高可用版本 | 姚文辉 |
| V1.1 | 2020-06-17 | 更新补充多维库备份策略、高可用健康监控 | 姚文辉 |
| V1.2 | 2020-06-23 | 更新补充多维库目录结构、配置项详细说明 | 姚文辉 |
| V1.3 | 2022-02-21 | 更新https配置 | 姚文辉 |
| V1.4 | 2023-04-10 | 变更部署配置描述 | 姚文辉 |
| V1.5 | 2023-04-13 | 更新主备部署图 | 姚文辉 |

目录

[金蝶多维数据库运维指南 1](#_Toc132286032)

[一、项目/组件简介 3](#_Toc132286033)

[多维数据库目录 4](#_Toc132286034)

[历史版本目录 9](#_Toc132286035)

[二、安装 11](#_Toc132286036)

[环境要求 11](#_Toc132286037)

[安装过程 12](#_Toc132286038)

[三、配置 24](#_Toc132286039)

[多维库配置项详细说明 24](#_Toc132286040)

[高可用配置 36](#_Toc132286041)

[动态计算配置 54](#_Toc132286042)

[开机自启动 55](#_Toc132286043)

[用户名密码 56](#_Toc132286044)

[文件打开数和map\_count 59](#_Toc132286045)

[linux内存分配策略 60](#_Toc132286046)

[数据源配置 61](#_Toc132286047)

[报表插件spreadJs表格控件授权 64](#_Toc132286048)

[四、运维 65](#_Toc132286049)

[多维数据库服务关闭 65](#_Toc132286050)

[多维数据库备份与还原 66](#_Toc132286051)

[多维数据库服务更新 73](#_Toc132286052)

[多维数据库服务迁移 82](#_Toc132286053)

[五、常见问题处理 83](#_Toc132286054)

[常见问题 83](#_Toc132286055)

[注意事项 86](#_Toc132286056)

## 一、项目/组件简介

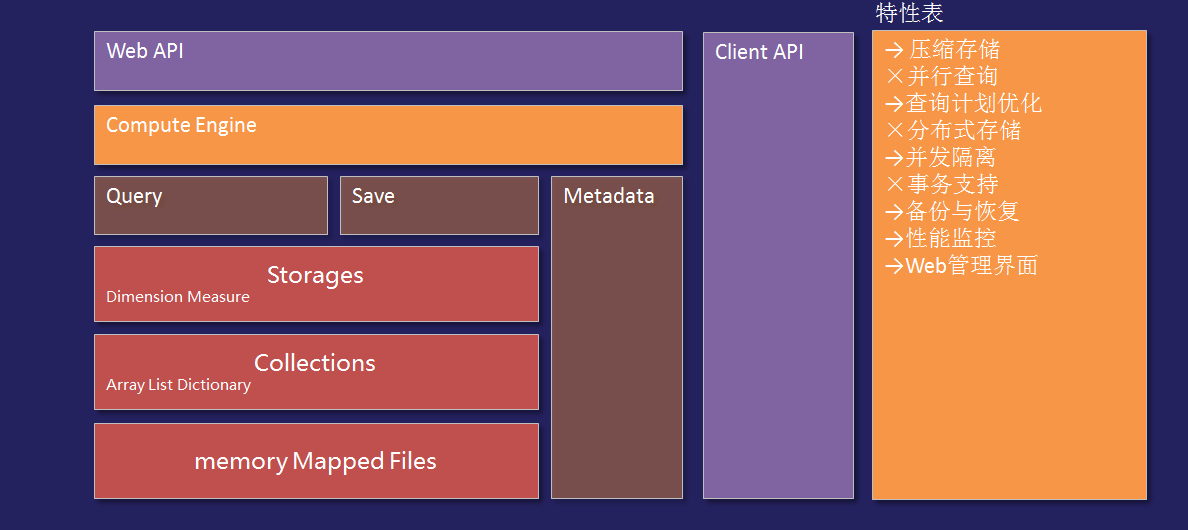
多维数据库服务简介：

Kingdee 多维数据库，代号Shrek，是金蝶自主研发的多维数据库产品；

提供多维数据库的存储、查询和计算能力；

类似 Hyperion 的EssBase产品。

架构图：



现项目为金蝶云苍穹合并报表和预算应用提供数据服务，业务端通过ClientAPI访问多维数据库Web服务接口进行查询、修改、保存、计算等。

目前，合并报表应用数据存储分为两部分，少部分基础数据存储在mysql中，主要数据则存储在多维数据库中。

合并报表应用与金蝶云苍穹其他应用部署不同的是：需单独部署多维数据库服务，多维数据库服务对其他金蝶云苍穹应用（总账、资产、费用等）数据不会造成任何影响。

### 多维数据库目录

安装包bos-olap-server.zip目录结构如下

.

├── 6.0.0 版本号主体程序包文件夹(格式x.x.x)，包含lib包与web应用

├── bin 脚本文件夹，启动脚本等其他工具脚本

├── conf 配置文件夹，web.xml与shiro.ini等配置

└── docs docs文档文件夹，相关介绍说明、部署和说明文档

启动服务后目录结构

.

├── 6.0.0 版本号主体程序包文件夹(格式x.x.x)，包含lib包与web应用

├── backup 自动备份服务备份文件夹（如果没有，启动服务自动创建）

├── database 数据库文件夹（如果没有，启动服务自动创建）

├── bin 脚本文件夹，启动脚本等其他工具脚本

├── conf 配置文件夹，web.xml与shiro.ini等配置

├── olaplogs 所有日志文件夹（如果没有，启动服务自动创建）

└── docs docs文档文件夹，相关介绍说明、部署和说明文档

#### 主体程序包目录

形如6.0.0（格式x.x.x）的文件夹，是多维数据库主体程序包文件夹，目录结构：

. 6.0.0（格式x.x.x）

├── lib 所有jar包

└── webapp web前端页面

##### lib目录

lib目录下所有bos-olap-\*.jar的包是我们自研发的jar包，其他jar包均为第三方包；

. 6.0.0

├──lib

├── bos-olap-core-6.0.0.jar

├── bos-olap-webserver-6.0.0.jar

├── bos-olap-tools-6.0.0.jar

└── bos-olap-computingEngine-6.0.0..jar

手动升级时，如果需要单独替换包且未引用新的第三方包时，同时也不需要更新前端页面时，可以单独替换bos-olap-\*.jar这几个自研发的jar包；

##### webapp目录

. 6.0.0

└── webapp

webapp目录下是web前端页面，一般是整体和版本号文件夹一起更新；

#### 自动备份目录

[开启自动备份](#_开启自动备份) 后，服务启动时会检测服务根目录下有没有backup目录，如果没有该目录，会自动创建，并定时产生对应的cube全备文件；目录结构：

.

├── backup

├── CUBE0000000000000

├── CUBE0000000000000-202106180000.backup

├── CUBE0000000000000-202106180001.backup

├── CUBE0000000000001

├── CUBE0000000000001-202106180000.backup

├── CUBE0000000000001-202106180001.backup

├── CUBE0000000000002

├── CUBE0000000000002-202106180000.backup

├── CUBE0000000000002-202106180001.backup

└──......

#### 数据库目录

##### 业务数据

database目录是多维数据库真正存储业务数据的目录，服务启动时会检测服务根目录下有没有database目录，如果没有该目录，会自动创建，有多个cube目录结构如下：

.

├── database

├── CUBE0000000000000

├── CUBE0000000000001

├── CUBE0000000000003

├──......

└──index.json

* 类似于CUBE开头的CUBE0000000000000，这个是对应业务的体系编码，而在多维库中是一个单独文件夹，用于存储业务报表数据；
* index.json中描述了整个database下的所有CUBE，打开index.json可以看到会为每个cube分配一个id，同是记录了这个cube的name；

##### 事务日志

多维数据库的事务日志格式是以1000个事务id为一个事务日志文件,10个事务日志文件为一个压缩包；如下：

.

├── database

└── CUBE00000000000000001

├── commands\_11000\_20999.logz

├── commands\_1\_10999.logz

├── commands\_21000\_30999.logz

├── commands\_31000\_40999.logz

├── commands\_41000\_50999.logz

├── commands\_51000\_51999.log

├── commands\_52000\_52999.log

└── commands\_53000\_52999.log

上述目录结构是CUBE00000000000000001对应的所有事务日志，历史压缩包以.logz结尾，里面包含10个事务日志.log文件，每个.log包含1000个事务；

可以看到上述commands\_53000\_52999.log为最新的事务日志文件，因为53000>52999，说明该文件正等待被写入；当事务大于53999会生成一个新文件commands\_54000\_53999，同时把commands\_53000\_52999.log变更为commands\_53000\_53999.log，即当写满1000个事务时，滚动写入到新文件中去；

每个事务对于多维数据库来说是一个命令，归类为以下：

* 元数据命令
* 保存命令
* 计算命令
* ……

事务日志一般用于单个cube的紧急数据恢复；

#### 脚本目录

脚本目录

.

├── bin

├── autoProxy.bat window下健康报告启动脚本

├── autoProxy.sh linux下健康报考启动脚本

├── olapTools.bat window下数据库脚本工具

├── olapTools.sh 数据库脚本工具

├── resetAdminPW.bat window下重置管理员密码脚本

├── resetAdminPW.sh linux下重置管理员密码脚本

├── server.log 启动脚本控制台输出信息日志

├── start.bat window下服务启动脚本

├── start.sh linux下服务启动脚本

├── tools 服务运维其他工具

├── upgrade.sh web界面升级脚本

└── version 当前服务的版本号文件

#### 配置文件目录

配置文件夹，里面包含服务端口、用户名密码、主从配置等配置文件

.

├── conf

├── web.xml web服务的相关配置 ，不可缺省

~~├── tools-web.xml~~  tools应用的配置文件，自6.2.16后废弃，6.2.16可删除

├── shiro.ini 用户名密码配置文件，不可缺省

├── ruleList.json 高可用从机的全备、增备出厂策略文件，可缺省

├── replication.xml 高可用主从节点配置文件，可缺省

└── autoProxyConf.xml 高可用主从健康监控服务配置文件，可缺省

#### 日志文件目录

多维数据库产生的日志（不包含事务日志）都在olaplogs里

.

├── olaplogs

├── autoProxy 健康监控报告日志

├── backup 自动备份服务日志

├── console 控制台日志

├── profile 性能分析日志

├── replication 高可用主从日志

└── server 服务命令日志

按功能区分对应的日志信息

* 健康监控报告日志
* 自动备份服务日志
* 控制台日志
* 性能分析日志
* 高可用主从日志
* 服务命令日志

每个功能日志都是按日期滚动的，当天最新的日志以olap\_key.log文件，当天日志如果超过512MB，则也会被压缩；历史日志则会被压缩成olap\_key\_ yyyy-MM-dd.log\_i.zip；

格式如下：

.

├── olaplogs

├── autoProxy 健康监控报告日志

│   └── olap\_autoProxy.log

├── backup 自动备份服务日志

│   ├── olap\_backup\_2021-05-02.0.log.zip

│   └── olap\_backup.log

├── console 控制台日志

│   ├── olap\_console\_2021-06-17.0.log.zip

│   └── olap\_console.log

├── profile 性能分析日志

│   ├── CUBECMPort00966554815947616256

│   │   └── profile\_1\_0.log

│   ├── CUBECMPort00966554815947616256.profile

│   ├── CUBEZYCMPport1112272309445483520

│   │   └── profile\_1\_0.log

│   └── CUBEZYCMPport1112272309445483520.profile

├── replication 高可用主从日志

│   ├── replication\_2021-06-17.0.log.zip

│   └── replication.log

└── server 服务命令日志

├── olap\_server\_2021-06-07.0.log.zip

└── olap\_server.log

### 历史版本目录

#### 5.0.0之前多维数据库目录

5.0.0是2020-06-04发布的，多维数据库服务目录发生重大变更；

5.0.0之前的目录结构如下：

.

├── backup 自动备份服务备份文件夹（如果没有，启动服务自动创建）

├── database 数据库文件夹（如果没有，启动服务自动创建）

├── lib lib包

├── olaplogs 所有日志文件夹（如果没有，启动服务自动创建）

├── server.log 启动脚本控制台输出信息日志

├── start.sh linux下服务启动脚本

└── webapp web应用

最新版本的目录结构如下：

.

├── 6.0.0 版本号主体程序包文件夹(格式x.x.x)，包含lib包与web应用

├── backup 自动备份服务备份文件夹（如果没有，启动服务自动创建）

├── database 数据库文件夹（如果没有，启动服务自动创建）

├── bin 脚本文件夹，启动脚本等其他工具脚本

├── conf 配置文件夹，web.xml与shiro.ini等配置

├── olaplogs 所有日志文件夹（如果没有，启动服务自动创建）

└── docs docs文档文件夹，相关介绍说明、部署和说明文档

最新版本的目录结构详情请参看 [多维数据库目录](#_认识多维数据库目录)

可以看到一个明显的特征，在5.0.0以前：

* 没有版本号主体程序包文件夹(格式x.x.x)
* lib目录在根目录下
* 启动脚本在根目录下start.sh

#### 2.x版本多维数据库目录

具体版本大概为2.x版本，目录结构如下：

.

├── database 数据库文件夹（如果没有，启动服务自动创建）

├── lib lib包

├── olaplogs 所有日志文件夹（如果没有，启动服务自动创建）

├── serverStart.bat windows下服务启动脚本

├── serverStart.sh linux下服务启动脚本

└── webapp web应用

5.0.0之前的目录结构如下：

.

├── backup 自动备份服务备份文件夹（如果没有，启动服务自动创建）

├── database 数据库文件夹（如果没有，启动服务自动创建）

├── lib lib包

├── olaplogs 所有日志文件夹（如果没有，启动服务自动创建）

├── server.log 启动脚本控制台输出信息日志

├── start.sh linux下服务启动脚本

└── webapp web应用

最新版本的目录结构如下：

.

├── 6.0.0 版本号主体程序包文件夹(格式x.x.x)，包含lib包与web应用

├── backup 自动备份服务备份文件夹（如果没有，启动服务自动创建）

├── database 数据库文件夹（如果没有，启动服务自动创建）

├── bin 脚本文件夹，启动脚本等其他工具脚本

├── conf 配置文件夹，web.xml与shiro.ini等配置

├── olaplogs 所有日志文件夹（如果没有，启动服务自动创建）

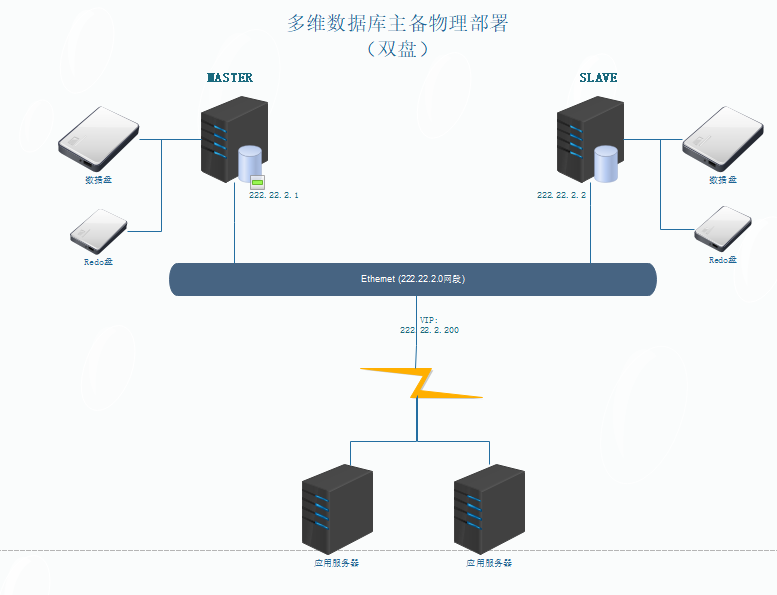
└── docs docs文档文件夹，相关介绍说明、部署和说明文档

可以看到一个明显的特征，在2.x目录：

* 没有版本号主体程序包文件夹(格式x.x.x)
* lib目录在根目录下
* 启动脚本为serverStart.sh，且在根目录下

## 二、安装

### 环境要求



|  |  |  |
| --- | --- | --- |
|  | 生产环境 | 测试环境 |
| 服务器台数 | 两台机器Linux服务器：  一台Master主机、一台Slave主机 | 如果不配置高可用，可以只需一台机器，服务状态为None；配置高可用则需要两台机器 |
| 服务器版本要求 | 系统centos 7.4以上，其他硬件配置请按照《金蝶多维数据库硬件配置指南》推荐部署 | 系统centos 7.4以上, cpu 16C，32GB内存 |
| 部署策略 | 推荐单独部署，不与其他应用（mysql、oracle、redis）共用服务器 | 可根据具体资源情况进行内存配置调整，为保证性能cpu最少16C |

多维数据库服务推荐部署在centos 7.4版本以上、JDK1.8的服务器；

window下不推荐正式环境使用，仅供功能测试（解压安装包运行bin/start.bat即可）

更详细的硬件配置详情见docs目录下《金蝶多维数据库硬件配置指南》 。

### 安装过程

多维数据库服务在linux环境下部署说明：

下面安装步骤是默认是None状态，不配置高可用；

如需要配置高可用，参看 [高可用配置](#_高可用配置_1)

#### 上传安装包

将多维数据库服务安装包bos-olap-server.zip上传到服务器,可以使用rz命令、winscp等其他软件工具，我们使用rz命令。

#未安装rz命令请执行

sudo yum install lrzsz

#在弹出的框中选择文件，上传即可

rz

#### 解压安装包

将服务安装包bos-olap-server.zip解压至指定文件夹olapserver。

unzip bos-olap-server.zip -d olapserver

bos-olap-server.zip安装包文件清单如下

.

├── 6.0.0 版本号文件夹(格式x.x.x)，包含该版本的lib包与web应用

├── bin 脚本文件夹，启动脚本等其他工具脚本

├── conf 配置文件夹，web.xml与shiro.ini等配置

└── docs doc文档文件夹，部署和说明文档

#### 开启自动备份

多维库备份为定时任务式的全量备份，未修改数据将不会进行数据备份。

查看web.xml 中已开启自动备份服务。

vim olapserver/conf/web.xml

如下：

<!-- 备份时间间隔，单位为分，-1表示关闭自动备份服务 -->

**<context-param>**

**<param-name>**interval**</param-name>**

**<param-value>**-1**</param-value>**

**</context-param>**

<!-- 服务的提供结构，为自动备份设置 -->

<!-- 使用scp命令将本地的备份文件copy到远程服务器上-->

**<context-param>**

**<param-name>**command**</param-name>**

**<param-value>**

scp -rvi /kingdee/mdd/.ssh/id\_rsa /kingdee/mdd/olap/backup/%s

user@127.0.0.1:/kingdee/mdd/olap\_bak/%s

**</param-value>**

**</context-param>**

##### interval参数

单位为分钟，即隔多少分钟检查数据是否有变更，进而决策是否需要进行一次备份，当值为-1时，备份服务关闭。

##### command参数

用于将本地备份文件远程拷贝到其它文件服务中。

scp -rvi /kingdee/mdd/.ssh/id\_rsa /kingdee/mdd/olap/backup/%s

user@127.0.0.1:/kingdee/mdd/olap\_bak/%s

其中%s为多维数据库服务自动生成的备份文件名，必须要有。

Command参数可以修改为空，或修改为本地拷贝（cp命令），则不进行远程拷贝。

将command参数置空

<!-- 使用scp命令将本地的备份文件copy到远程服务器上-->

**<context-param>**

**<param-name>**command**</param-name>**

**<param-value></param-value>**

**</context-param>**

注释command配置

<!-- 使用scp命令将本地的备份文件copy到远程服务器上-->

<!--<context-param>

<param-name>command</param-name>

<param-value>

scp -rvi /kingdee/mdd/.ssh/id\_rsa /kingdee/mdd/olap/backup/%s

[kduser@127.0.0.1:/kingdee/mdd/olap\_bak/%s](mailto:kduser@127.0.0.1:/home/kduser/olap_bak/%25s)

</param-value>

</context-param>-->

* 正式环境必须开启自动备份服务，interval必须为正整数；
* 开启服务后第一次备份是服务启动的1分钟后开始本地备份，后备份间隔为interval所设置的数值，在服务根目录下自动生成backup文件夹。
* 如果出现command命令配置正确，但是有些cube备份并没有同步过去，这是由于在command配置生效时，本地已经存在backup备份了，这些cube备份文件夹在是该cube第一次备份时创建的，同时会远程创建目录；配置的cp和scp仅仅是拷贝文件，并不会创建目录，所以在第一次启用command时需要把已存在的backup下的目录手动同步到olap\_bak里；

#### 监控自动备份异常

在olapserver/olaplogs/backup 文件夹下日志文件olap\_backup.log下是否出现“ERROR BackupLog”字样，建议最好自动化监控。

#### 删除过期的备份文件

 <!-- 每个cube保留的最少备份文件个数 -->

**<context-param>**

**<param-name>**minRetainFiles**</param-name>**

**<param-value>30</param-value>**

**</context-param>**

<!-- 保留近多少天的文件 -->

**<context-param>**

**<param-name>**retainDays**</param-name>**

**<param-value>30</param-value>**

**</context-param>**

上述配置意味着是开启自动备份后，对于每个cube来说，少于30个备份文件，将不会进行任何删除操作；超出30个备份文件，会按时间顺序删除30天前的备份文件；

这里备份文件个数和保留天数做了限制，超过90 无效都会取90，即最多每个cube保留90个cube备份文件；每个cube超过90个文件，按时间顺序删除90天前的备份文件；

#### 服务端口

查看服务端口

vim olapserver/conf/web.xml

<!— 服务端口 -->

**<context-param>**

**<param-name>**port**</param-name>**

**<param-value>8080</param-value>**

**</context-param>**

多维数据库服务端口在web.xml配置文件，默认端口号为8080

可直接用vim修改，修改后重启生效。

修改端口后，打不开应用，有时原因为服务器防火墙已打开，需添加端口例外（或关闭防火墙），这里我们可以配置打开相关端口，如下：

#CentOS 7 使用iptables 开放端口

#CentOS 7.0默认使用的是firewall作为防火墙，这里改为iptables防火墙。

#关闭firewall：

systemctl stop firewalld.service

systemctl disable firewalld.service

systemctl mask firewalld.service

#安装iptables防火墙

yum install iptables-services -y

#启动设置防火墙

systemctl enable iptables

systemctl start iptables

#查看防火墙状态

systemctl status firewalld

systemctl status iptables

#编辑防火墙，增加端口

vi /etc/sysconfig/iptables

-A INPUT -m state --state NEW -m tcp -p tcp --dport 22 -j ACCEPT

-A INPUT -m state --state NEW -m tcp -p tcp --dport 80 -j ACCEPT

-A INPUT -m state --state NEW -m tcp -p tcp --dport 8080 -j ACCEPT

-A INPUT -m state --state NEW -m tcp -p tcp --dport 3306 -j ACCEPT

#保存退出

:wq!

#重启配置

#重启防火墙使配置生效

systemctl restart iptables.service

#设置防火墙开机启动

systemctl enable iptables.service

#### 配置linux服务参数

由于多维库服务用到文件映射，需要配置服务的文件打开数和map\_count，不然容易造成too many open file的异常，请参看 [文件打开数和map\_count](#_文件打开数和map_count)

#### 启动服务

执行启动

sh bin/start.sh start

运行后，会生成backup、database、olaplogs文件夹，文件结构如下：

Olapserver运行后文件结构图

.

├── backup 本地备份文件夹（未开启自动备份则不会生成）

├── database 存放数据库文件

├── 6.0.0 版本号文件夹，包含该版本的lib包与web应用

├── bin 脚本文件夹，启动脚本等其他工具脚本

├── conf 配置文件夹，web.xml与shiro.ini等配置

├── docs doc文档文件夹，部署和说明文档

└── olaplogs olapserver日志目录

##### ARM机器

多维数据库服务从8.7.10后适配支持ARM机器，如果机器是华为鲲鹏、飞腾、申威、

龙芯，请修改bin/start.sh文件，在启动命令后面加上一段内容：

-DEnableMddOtherCpu = true

# 修改启动脚本

vim bin/start.sh



#### 初始化密码

初次安装，需要初始化密码<http://localhost:8080/bos-olap-webserver/#/>

这里默认管理员用户名为admin，密码为你设置的密码



初始化密码只有在用户首次安装后出现，设置密码后则为登录页面；

#### 验证服务是否启动成功

直接通过ps查看进程启动情况

# 查看olap相关进程

ps -ef|grep OlapWebServerMain

正常查询情况



初始化密码后，退出，再次登录

打开<http://localhost:8080/bos-olap-webserver/#/>

可以看到如下界面。



#### 配置https

由于使用http认证存在抓包或中间人攻击等隐患，出于安全性考虑，服务建议配置https

Jetty通过OptionalSslConnectionFactory配置同时兼容https和http

需要准备两个文件

* .p12证书文件
* .p12存储密码文件（默认存储密码和管理密码一致）
* 生成证书文件keystore.p12，替换下面的<yourpassword> <youripaddress>

<!—生成jetty使用的OBF格式的密码-->

keytool -genkey -alias KingdeeMDD -keypass yourpassword -storepass yourpassword -storetype PKCS12 -keyalg RSA -keysize 2048 -keystore keystore.p12 -validity 3650 -dname "C=CN,ST=BJ,L=BJ,O=kingdeemdd,OU=kingdeemdd,CN=kingdeemdd" -ext san=dns:localhost,dns:kingdeemdd,ip:127.0.0.1,ip:::1,ip:youripaddress

* 生成OBF格式编码文本keystore，请使用jetty包生成

<!—生成jetty使用的OBF格式的密码-->

java -cp jetty-all-9.4.43.v20210629-uber org.eclipse.jetty.util.security.Password yoursslpassword

* 将生成的keystore.p12和keystore文件放到conf目录下

.

├──conf

├── keystore.p12

└── keystore

* 在web.xml里配置模式

<!-- 是否启用https 值可以为 0 1 2 -->

<!-- 0表示不启用https 仅支持http； 1表示启用https 不支持http； 2表示同时支持https和http；-->

<context-param>

<param-name>enableHttps</param-name>

<param-value>0</param-value>

</context-param>

<!-- 是否启用https -->

<!-- https证书文件名称 keystore.p12-->

<context-param>

<param-name>keyStorePath</param-name>

<param-value>keystore.p12</param-value>

</context-param>

<!-- https证书文件名称 -->

<!-- KeyStorePassword密码文件 -->

<context-param>

<param-name>keyStorePassword</param-name>

<param-value>keystore</param-value>

</context-param>

<!-- KeyStorePassword密码文件-->

打开<https://localhost:8080/bos-olap-webserver/#/>

或者<https://localhost:8080/mdd/#/>

可以看到如下界面：



## 三、配置

### 多维库配置项详细说明

所有多维数据库有关配置项（服务器配置不在内）可在这统一查找

对目录结构不了解的，请参看 [多维数据库目录](#_认识多维数据库目录) ，围绕配置文件夹，展开说明

.

├── conf

├── web.xml web服务的相关配置，不可缺省

~~├── tools-web.xml~~  tools应用的配置文件，自6.2.16后废弃，6.2.16可删除

├── shiro.ini 用户名密码配置文件，不可缺省

├── ruleList.json 高可用从机的定时全备、增备出厂策略文件，可缺省

├── replication.xml 高可用主从节点配置文件，可缺省

└── autoProxyConf.xml 高可用主从健康监控服务配置文件，可缺省

#### web.xml配置

web服务的相关配置，不可缺省，web.xml配置项对照表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 键 | 值类型 | 默认值 | 含义 | 建议配置 |
| port | 整型 | 8080 | 服务端口 | 按需修改 |
| dynamicCalc | 布尔 | true | 是否开启动态计算 | true |
| TransactionLogEnabled | 布尔 | true | 开启事务日志 | true，这个一般不建议修改 |
| interval | 整型 | 720 | 是否开启自动备份，单位为分钟 | 正式环境建议：10以上  测试环境建议：-1  详情参看 [开启自动备份](#_开启自动备份) |
| command | 字符串 | 无（被注释） | 远程拷贝备份文件scp命令 | 详情参看 [开启自动备份](#_开启自动备份) 中command参数 |
| minRetainFiles  retainDays | 整型  整型 | 30  30 | 备份文件是否过期的配置,两个参数搭配使用 | 详情参看  [删除过期的备份文件](#_删除过期的备份文件) |
| defaultWorkerCount | 整型 | 4 | 计算引擎开启线程数 | 建议与cpu核心数一致 |
| computingTimeOut | 整型 | -1 | 计算超时时间，单位为秒，-1表示不限制 | 建议配置20，从6.2.0以后，如果你配置为-1，系统会默认设置成20s |

#### tools-web.xml配置

tools应用的配置文件，自6.2.16后废弃，6.2.16可删除

#### shiro.ini配置

用户名密码配置文件，不可缺省，自5.1.3后，shiro.ini文件加密处理，一般无需修改

如需修改密码，请参看 [忘记密码](#_忘记密码)

#### ruleList.json配置

{"list":[{"jobName":"BACKUP\_JOB\_1605973805546","backupType":"Increment","rule":"0 0 0 ? \* MON,TUE,WED,THU,FRI,SAT"},{"jobName":"BACKUP\_JOB\_1605973786900","backupType":"Total","rule":"0 0 0 ? \* SUN"}]}

高可用从机的定时全备、增备出厂策略文件，可缺省；

请参看 [定时全备、增备（仅限高可用主从）](#_定时全备、增备（仅限高可用主从）)

#### replication.xml配置

高可用主从服务节点配置文件，可缺省，不存在则没有高可用配置；服务节点为None；

<?xml version="1.0" encoding="UTF-8"?>

<config>

<node>

<!-- master设置为1，slave设置为2 -->

<serverId>1</serverId>

<redoPath></redoPath>

<redoKeepSegments>1024</redoKeepSegments>

<vip></vip>

<enableStat>true</enableStat>

<backupPath></backupPath>

</node>

<!-- master服务配置 -->

<master>

<bindPort>9090</bindPort>

<!-- 同步slave节点ID -->

<!-- <synSlavesID></synSlavesID> -->

<masterReplication>false</masterReplication>

<enableWriteDisk>true</enableWriteDisk>

</master>

<!-- slave服务配置 -->

<slave>

<slaveReplication>false</slaveReplication>

<host></host>

<port>9090</port>

</slave>

</config>

##### node配置项

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **配置项** | **键** | **值类型** | **默认值** | **含义** |
| node | serverId | 整型 | 1 | 高可用服务节点标识id |
| redoPath | 整型 | 无 | redo文件存储路径 |
| redoKeepSegments | 整型 | 1024 | redo文件保留的文件个数，超出将会删除旧的文件 |
| vip | 字符串 | 无 | 虚拟ip，一般为keepalived定义的虚拟ip。当Master设置为其本机ip时，则不会进行自动主从切换 |
| enableStat | 布尔 | true | 监控主从同步情况 |
| backupPath | 字符串 | 无 | 定时全备、增备备份存储路径 |

##### master配置项

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **配置项** | | **键** | **值类型** | **默认值** | **含义** |
| master | bindPort | | 整型 | 9090 | 主从redo通信端口（与服务端不同） |
| synSlavesID | | 整型 | 默认注释 | 同步复制Redo的服务节点标识id。当采用同步复制模式时，设置为slave结点id；当采用异步复制时，默认注释即可。通常，建议采用异步模式。 |
| masterReplication | | 布尔 | false | 是否设置当前结点为master结点 |
| enableWriteDisk | | 布尔 | true | redo是否写入磁盘，必须设置为true |

##### slave配置项

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **配置项** | **键** | **值类型** | **默认值** | **含义** |
| slave | slaveReplication | 布尔 | false | 是否设置当前结点为slave结点 |
| host | 字符串 | 无 | 主从redo通信ip地址，设置主从**对端**IP  （写主机真实ip，不是vip） |
| port | 整型 | 9090 | 与主服务节点bindPort保持一致 |

replication.xml配置一共三个部分的配置项：

* node
* master
* slave

服务器节点判断：

|  |  |  |
| --- | --- | --- |
| **masterReplication** | **slaveReplication** | **服务节点最终状态** |
| true | false | Master |
| false | true | Slave |
| false | false | None（不启用主从模式） |
| true | true | 启动异常 |

从上表中可以看出，master配置项中masterReplication和slave配置项slaveReplication控制着配置项的启用部分，对于整个replication.xml配置文件来说，真正启用的永远只有两个部分的配置项或者3个部分的配置项都不启用：

服务器为Master状态时，replication.xml文件中只有下列两个部分的配置项才被启用

* node
* master

服务器为Slave状态时，replication.xml文件中只有下列两个部分的配置项才被启用

* node
* slave

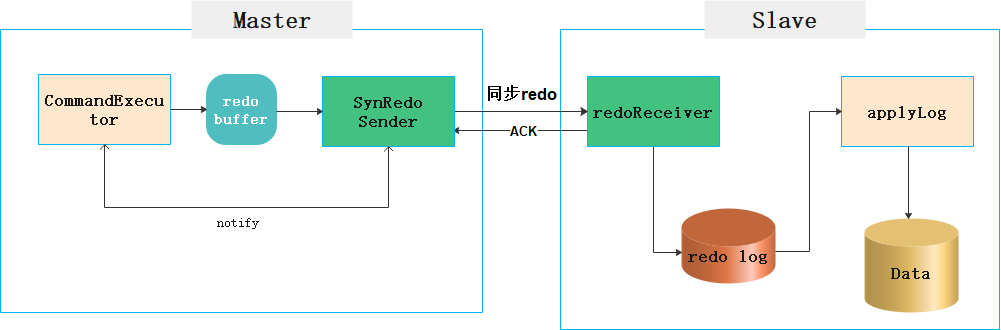
服务器为None状态时，replication.xml文件不存在，或者都不启用

* masterReplication false
* slaveReplication false

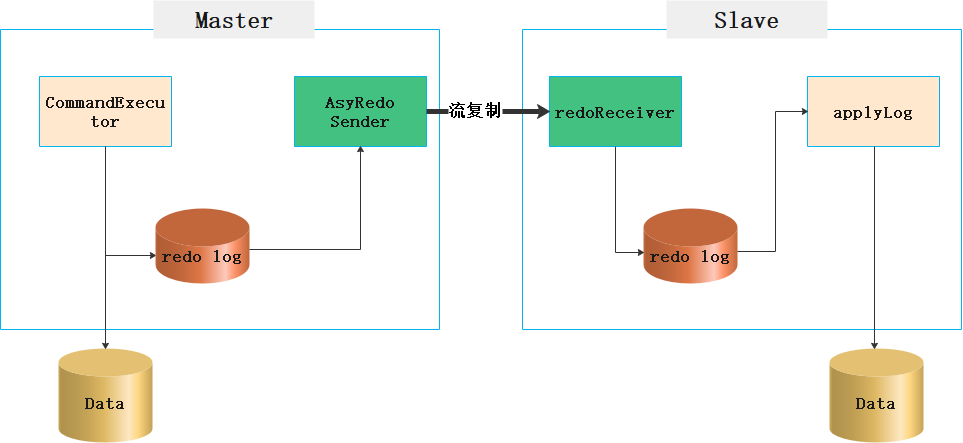
##### redo传输模式

同步模式和异步模式区别在于：

同步模式下主机器需要等到从机器从网络中接收redo数据完成后，才进行下一步操作；这样能保证主从数据是实时一致的；



异步模式下主机器将redo数据写入本地磁盘后，即可进行下一步操作；redo数据的传输是异步进行的，这样并不保证主从数据实时一致，但此配置可以避免master受从端机器的影响；



###### 同步模式

Master

10.10.1.1

Slave

10.10.1.2

serverid:1

redoPath:/redo

redoKeepSegments:1024

vip:10.10.1.100

enableState:true

backupPath:/backup

serverid:2

redoPath:/redo

redoKeepSegments:1024

vip:10.10.1.100

enableState:true

backupPath:/backup

bindPort:9090

synSlavesID:2

masterReplication:true

enableWriteDisk:true

bindPort:9090

synSlavesID:1

masterReplication:false

enableWriteDisk:true

slaveReplication:false

host:10.10.1.2

port:9090

slaveReplication:true

host:10.10.1.1

port:9090

tcp连接

高可用主从-同步模式

上图配置了标准的高可用主从-同步模式（一主一从），Master机器ip为10.10.1.1，Slave机器ip为10.10.1.2，keepalived配置虚拟ip为10.10.1.100

可以看到高可用主从-同步模式关键配置点在于

* serverid
* masterReplication
* synSlavesID
* slaveReplication
* host
* serverId控制了服务节点标识；
* 主Master机器serverID为1， synSlavesID配置为2；代表当前机器为Master机器时，产生的redo数据会同步传输给serverID为2的Slave机器；从Slave机器serverID为2， synSlavesID配置为1；代表当前机器为Master机器时，产生的redo数据会同步传输给serverID为1的Slave机器；而serverID为1或2的机器都存在，所以redo的传输模式为同步传输；
* masterReplication和slaveReplication控制着当前master配置项或slave配置项是否启用；

Master机器中masterReplication为true，slaveReplication为false，则master机器启用了node配置项和master配置项，对应图中绿色部分；master机器slave配置项暂时失效状态，对应master机器图中灰色部分；

Slave机器中masterReplication为false，slaveReplication为true，则slave机器启用了node配置项和slave配置项的配置，对应图中绿色部分；Slave机器master配置项暂时失效状态，对应slave机器图中灰色部分；

* 主从redo是通过tcp连接进行通信传输的，所以需要从机器连接到主机器；host是和port一起配置使用，是在slave配置项上，代表的含义是当启用slave配置项（slaveReplication为true）时，需要连接的主机器的ip（非虚拟ip）和对应的监听端口；

这里Master机器（10.10.1.1）中slave配置项虽然没启用（slaveReplication为false），但是host也进行了配置，值为Slave机器（10.10.1.2）的ip，端口为9090；是因为当发生主从自动切换时，Master机器（10.10.1.1）状态在内存中会被变更为Slave，此时，Master机器（10.10.1.1）中slave配置项会被启用，需要连接到主机器（原Slave机器10.10.1.2会变为Master机器，侦听端口9090）；

###### 异步模式

高可用主从-异步模式

Master

10.10.1.1

Slave

10.10.1.2

serverid:1

redoPath:/redo

redoKeepSegments:1024

vip:10.10.1.100

enableState:true

backupPath:/backup

serverid:2

redoPath:/redo

redoKeepSegments:1024

vip:10.10.1.100

enableState:true

backupPath:/backup

bindPort:9090

<!-- synSlavesID-->

masterReplication:true

enableWriteDisk:true

bindPort:9090

<!-- synSlavesID-->

masterReplication:false

enableWriteDisk:true

slaveReplication:false

host:10.10.1.2

port:9090

slaveReplication:true

host:10.10.1.1

port:9090

tcp连接

上图配置了标准的高可用主从-异步模式（一主一从），Master机器ip为10.10.1.1，Slave机器ip为10.10.1.2，keepalived配置虚拟ip为10.10.1.100

可以看到高可用主从-异步模式关键配置点在于

* serverid
* masterReplication
* synSlavesID
* slaveReplication
* host

这里所有配置与上面的高可用主从-同步模式一致，除了synSlavesID配置

* synSlavesID两边都是注释的，即不管哪台机器为Master机器，都没有对应的同步服务器节点机器；这里也可以将注释打开；主Master机器serverID为1， synSlavesID配置为3；代表当前机器为Master机器时，产生的redo数据会同步传输给serverID为3的Slave机器；从Slave机器serverID为2， synSlavesID配置为4；代表当前机器为Master机器时，产生的redo数据会同步传输给serverID为4的Slave机器；而serverID为3或4的机器不存在，所以redo的传输模式为异步传输；

##### 主从自动切换

配置文件replication.xml只有一份；当部署了高可用时，由于有keepalived的存在，程序会自动检测vip是否在本机，来判断是否需要主从自动切换，所以当发生自动切换时，服务状态在内存中发生改变，会从当前服务节点读取replication.xml对应的配置；

当发生自动主从切换时：

原先为Master状态，内存中会切换为Slave状态，并且读取对应的replication.xml中slave配置项；

原先为Slave状态，内存中会切换为Master状态，并且读取对应的replication.xml中master配置项；

Master

10.10.1.1

Slave

10.10.1.2

serverid:1

redoPath:/redo

redoKeepSegments:1024

vip:10.10.1.100

enableState:true

backupPath:/backup

serverid:2

redoPath:/redo

redoKeepSegments:1024

vip:10.10.1.100

enableState:true

backupPath:/backup

bindPort:9090

<!-- synSlavesID-->

masterReplication:true

enableWriteDisk:true

bindPort:9090

<!-- synSlavesID-->

masterReplication:false

enableWriteDisk:true

slaveReplication:false

host:10.10.1.2

port:9090

slaveReplication:true

host:10.10.1.1

port:9090

tcp连接

高可用-主从自动切换前

高可用-主从自动切换后

Slave

10.10.1.1

Master

10.10.1.2

serverid:1

redoPath:/redo

redoKeepSegments:1024

vip:10.10.1.100

enableState:true

backupPath:/backup

serverid:2

redoPath:/redo

redoKeepSegments:1024

vip:10.10.1.100

enableState:true

backupPath:/backup

bindPort:9090

<!-- synSlavesID-->

masterReplication:true

enableWriteDisk:true

bindPort:9090

<!-- synSlavesID-->

~~masterReplication:false~~

enableWriteDisk:true

~~slaveReplication:false~~

host:10.10.1.2

port:9090

slaveReplication:true

host:10.10.1.1

port:9090

tcp连接

高可用主从自动切换并不会修改磁盘中replication.xml中的配置，而是将内存中的配置状态改变，再去读取对应的配置项信息，图中的黑色配置部分为启用，但是masterReplication 与slaveReplication 无效，灰色配置部分无效；

即切换后的主机器（10.10.1.2）读取master配置项的masterReplication:false（内存中状态已经变更，此值无效）bindPort:9090，去开启对应的侦听端口10.10.1.2:9090；

即切换后的从机器（10.10.1.1）读取slave配置项的slaveReplication:false（内存中状态已经变更，此值无效）host:10.10.1.2；port:9090；去连接主机器10.10.1.2:9090；

这里需要注意，因为切换后的服务状态（Master->Slave;Slave->Master）是在内存中变更，而磁盘中replication.xml中的配置项并没有发生变化，如果对发生主从自动切换后的多维库服务重启或linux服务器重启，对应的服务节点的状态会发生变化；

#### autoProxyConf.xml配置

高可用主从健康监控服务配置文件，可缺省；不存在则启动不了健康监控服务；

<?xml version="1.0" encoding="UTF-8"?>

<healthMonitor>

<!-- 进行任务（健康报告同步任务、数据对比任务等）的OLAP服务1，

格式为 IP:Port 例如：127.0.0.1:8080 -->

<config key="serverIP1" value="10.10.1.1:8080"/>

<!-- 进行任务（健康报告同步任务、数据对比任务等）的OLAP服务2，

格式为 IP:Port 例如：127.0.0.1:8081 -->

<config key="serverIP2" value="10.10.1.2:8080"/>

<!-- 数据对比任务：定时对两个Shrek服务进行对比，包括cube数量、顺序；

每个cube的元数据、数据。 -->

<!-- 是否开启数据对比任务，“true”或者“false”。 -->

<config key="executeComparisonTask" value="true" />

<!-- 初次数据对比任务的时间：时、分、秒 -->

<config value="02" key="comparisonHour"/>

<config value="00" key="comparisonMinute"/>

<config value="00" key="comparisonSecond"/>

<!-- 数据对比任务的执行间隔，单位为秒 -->

<config value="86400" key="comparisonTaskInterval"/>

<!-- 每个数据对比任务进行数据对比的次数 -->

<config value="3" key="comparisonNumber"/>

<!-- 每个数据对比任务进行数据对比的间隔 -->

<config value="1800" key="comparisonInterval"/>

</healthMonitor>

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 键 | 值类型 | 默认值 | 含义 | 建议配置 |
| serverIP1 | 字符串 | 127.0.0.1:8080 | Ip+服务端口 | 按需修改 |
| serverIP2 | 字符串 | 127.0.0.1:8081 | Ip+服务端口 | 按需修改 |
| executeComparisonTask | 布尔 | true | 是否开启数据对比任务 | true，这个一般不建议修改 |
| comparisonHour  comparisonMinute  comparisonSecond | 整型 | 02:00:00 | 初次数据对比任务的时间：时、分、秒 | 按需修改，还是建议凌晨2点左右进行数据对比，这里不设凌晨00:00:00是为了防止与定时全备时间冲突 |
| comparisonTaskInterval | 整型 | 86400 | 数据对比任务的执行间隔，单位为秒 | 这里86400为24小时，代表每天，建议不修改 |
| comparisonNumber | 整型 | 3 | 每个数据对比任务进行数据对比的次数 | 这个一般不建议修改；3次中只要有1次正确则通过 |
| comparisonInterval | 整型 | 1800 | 每个数据对比任务进行数据对比的间隔，单位为秒 | 这个一般不建议修改 |

上述默认配置：

启用一个定时任务，在每天的02:00:00开启，对10.10.1.1:8080（主服务节点）与10.10.1.2:8081（从服务节点）的所有cube进行数据扫描比对，总共进行comparisonNumber （3）次数据扫描比对，每次数据扫描比对时间间隔为comparisonInterval（1800）30分钟，如果有任意cube数据不一致，则本次数据扫描比对结束；等待30分钟后，进行下一次数据扫描比对，直到comparisonNumber （3）次数据扫描比对结束，然后等待下一个凌晨00:00:00；

* 如果comparisonNumber （3）次数据扫描比对只要有任意一次数据一致，

则判定主从机器数据一致；

* 如果comparisonNumber （3）次数据扫描比对结果都不一致，

那么判定为主从机器数据不一致；

### 高可用配置

高可用服务状态有

* Master
* Slave
* None

如果不进行配置（或者conf目录下无replication.xml文件时），则默认是None状态

详情可参看 [replication.xml配置](#_replication.xml配置)

#### 环境准备

配置Master或Slave需要执行下列步骤，如果不配置高可用，则可以跳过这个步骤。

本文档ip地址基准：

master 对应的地址 10.10.1.1 slave 对应的地址 10.10.1.2

vip（虚拟ip）对应配置 10.10.1.100

##### 检查环境

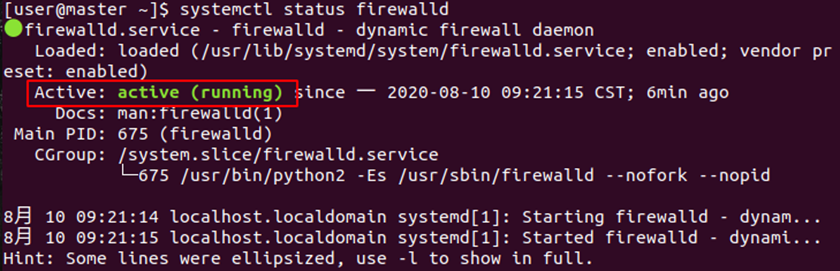
1. 检查服务器之间是否能联通（ping）
2. 关闭selinux，使keepalive有权限执行检测脚本

sudo setenforce 0

1. 检查防火墙是否关闭

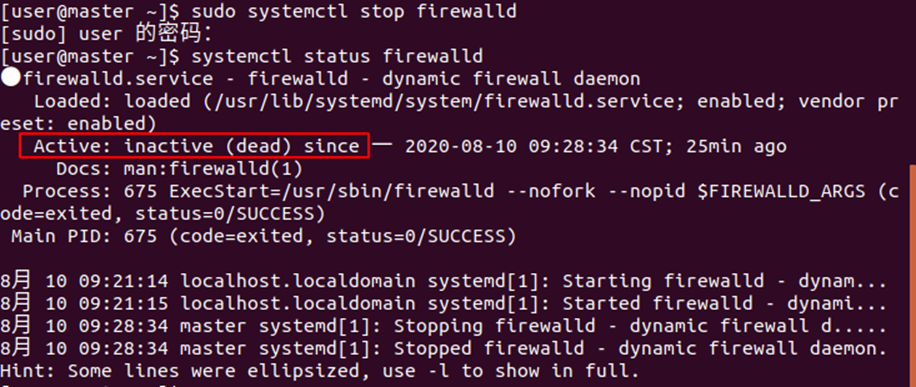
# 查看防火墙状态

systemctl status firewalld



# 停止防火墙

systemctl stop firewalld



##### 配置ntp服务器

Master机器安装ntp

# 安装ntp服务

yum install -y ntp

1. 将master改为本地同步

改为本地同步前，请先同步网络时间将时区设置为CST；

如果可以连接外网，本操作可忽略

127.0.0.1 表示本地服务时间服务器

# 由于master主机不能连外网，这里配置为本地更新时间

vim /etc/ntp.conf

# 新增本地时间同步配置

server 127.0.0.1

fudge 127.0.0.1 stratum 8

# 注释下面

#server 0.centos.pool.ntp.org iburst

#server 1.centos.pool.ntp.org iburst

#server 2.centos.pool.ntp.org iburst

#server 3.centos.pool.ntp.org iburst

1. 开机自启动，并启动

systemctl enable ntpd

systemctl start ntpd

1. ntpstat 查看是否已经同步

ntpstat

执行后，正常显示

synchronised to local net at stratum 9

time correct to within 12 ms

polling server every 64 s

1. ntpq -p 查看上层同步状态

ntpq –p

执行后，正常显示

remote refid st t when poll reach delay offset jitter

=============================================================================

\*LOCAL(0) .LOCL. 8 l 51 64 377 0.000 0.000 0.000

# remote - 本机和上层ntp的ip或主机名，“+”表示优先，“\*”表示次优先

# refid - 参考上一层ntp主机地址

# st - stratum阶层

# when - 多少秒前曾经同步过时间

# poll - 下次更新在多少秒后

# reach - 已经向上层ntp服务器要求更新的次数

# delay - 网络延迟

# offset - 时间补偿

# jitter - 系统时间与bios时间差

1. slave机器安装ntp

# 安装ntp服务

yum install -y ntp

1. 修改slave服务器时间校准服务器地址为master(10.10.1.1)

# slave的时间从master服务器上同步

vim /etc/ntp.conf

# 新增从master(10.10.1.1)服务器上同步时间

server 10.10.1.1 iburst

# 注释下面

#server 0.centos.pool.ntp.org iburst

#server 1.centos.pool.ntp.org iburst

#server 2.centos.pool.ntp.org iburst

#server 3.centos.pool.ntp.org iburst

1. 开机自启动，并启动

systemctl enable ntpd

systemctl start ntpd

1. ntpstat 查看是否已经从master（10.10.1.1）同步，

ntpstat

执行后，正常显示

synchronised to NTP server(10.10.1.1) at stratum 9

time correct to within 12 ms

polling server every 64 s

如未同步可以重启下ntp服务看下

systemctl restart ntpd

1. ntpq -p 查看上层同步状态

ntpq –p

执行后，正常显示

remote refid st t when poll reach delay offset jitter

==============================================================================

\*10.10.1.1 LOCAL(0) 8 l 51 64 377 0.000 0.000 0.000

注意：服务器时区不同 ，可能造成时间无法同步，请先手动调整服务器对应的时区为CST（同一时区即可）例如：设置为上海

timedatectl set-timezone Asia/Shanghai

##### keepalived配置

1. master安装keepalived

sudo yum install -y keepalived

1. master编写keepalived检查脚本，并保存

# master端配置 keepalived\_check.sh

vim /etc/keepalived/ keepalived\_check.sh

#!/bin/bash

status=$(sh /kingdee/mdd/autotest/olap/bin/start.sh status)

dateTime=$(date +%Y-%m-%d)

if [ $status -eq 1 ];then

echo "check on $dateTime $(date +%H):$(date +%M):$(date +%S) ,olap server status is started." > /etc/keepalived/check.log

exit 0

else

echo "check on $dateTime $(date +%H):$(date +%M):$(date +%S) ,olap server had been stopped." > /etc/keepalived/check.log

exit 1

fi

1. master修改keepalived配置文件

将“/etc/keepalived/keepalived.conf”文件内容修改为对应内容

# master端配置 keepalived.conf

vim /etc/keepalived/keepalived.conf

! Configuration File for keepalived

global\_defs { #全局配置

router\_id Node1 #机器标识

script\_user root

}

vrrp\_script chk\_shrek {

script "/etc/keepalived/keepalived\_check.sh" #检查脚本

interval 2 #脚本执行间隔健康检查周期

weight 20 #优先级变化幅度，脚本结果导致的优先级变更：

10表示优先级+10；-10则表示优先级-10

fall 3 #判定服务异常的检查次数

rise 2 #判定服务正常的检查次数

}

vrrp\_instance VI\_1 {

state BACKUP #指定实例初始状态，实际的MASTER和BACKUP是选举决定的。

interface ens160 #指定实例绑定的网卡,固有IP的网卡，用来发VRRP包

virtual\_router\_id 50 #设置VRID标记，多个集群不能重复(0..255)

priority 100 #设置优先级，优先级高的会被竞选为Master

advert\_int 1 #检查的时间间隔，默认1s

nopreempt #工作模式，nopreempt表示工作在非抢占模式，默认是抢占模

式 preempt,说明：这个配置只能在BACKUP主机上面设置

authentication {

auth\_type PASS

auth\_pass 1111

}

virtual\_ipaddress {

10.10.1.100 # vip

}

track\_script { # 追踪脚本，通常用于去执行上面的vrrp\_script定义的脚本内容

chk\_shrek

}

}

1. master启动keepalived

systemctl start keepalived

1. slave安装keepalived

yum install -y keepalived

1. slave编写keepalived检查脚本，并保存

# slave端配置 keepalived\_check.sh

vim /etc/keepalived/ keepalived\_check.sh

#!/bin/bash

status=$(sh /kingdee/mdd/autotest/olap/bin/start.sh status)

dateTime=$(date +%Y-%m-%d)

if [ $status -eq 1 ];then

echo "check on $dateTime $(date +%H):$(date +%M):$(date +%S) ,olap server status is started." > /etc/keepalived/check.log

exit 0

else

echo "check on $dateTime $(date +%H):$(date +%M):$(date +%S) ,olap server had been stopped." > /etc/keepalived/check.log

exit 1

fi

1. slave修改keepalived配置文件

将“/etc/keepalived/keepalived.conf”文件内容修改为对应内容

# slave端配置 keepalived.conf

vim /etc/keepalived/keepalived.conf

! Configuration File for keepalived

global\_defs { #全局配置

router\_id Node2 #机器标识

script\_user root

}

vrrp\_script chk\_shrek {

script "/etc/keepalived/keepalived\_check.sh" #检查脚本

interval 2 #脚本执行间隔健康检查周期

weight 20 #优先级变化幅度，脚本结果导致的优先级变更：

10表示优先级+10；-10则表示优先级-10

fall 3 #判定服务异常的检查次数

rise 2 #判定服务正常的检查次数

}

vrrp\_instance VI\_1 {

state BACKUP #指定实例初始状态，实际的MASTER和BACKUP是选举决定的。

interface ens192 #指定实例绑定的网卡,固有IP的网卡，用来发VRRP包

virtual\_router\_id 50 #设置VRID标记，多个集群不能重复(0..255)

priority 90 #设置优先级，优先级高的会被竞选为Master

advert\_int 1 #检查的时间间隔，默认1s

#nopreempt #工作模式，nopreempt表示工作在非抢占模式，默认是抢占模

式 preempt,说明：这个配置只能在BACKUP主机上面设置

authentication {

auth\_type PASS

auth\_pass 1111

}

virtual\_ipaddress {

10.10.1.100 # vip

}

track\_script { # 追踪脚本，通常用于去执行上面的vrrp\_script定义的脚本内容

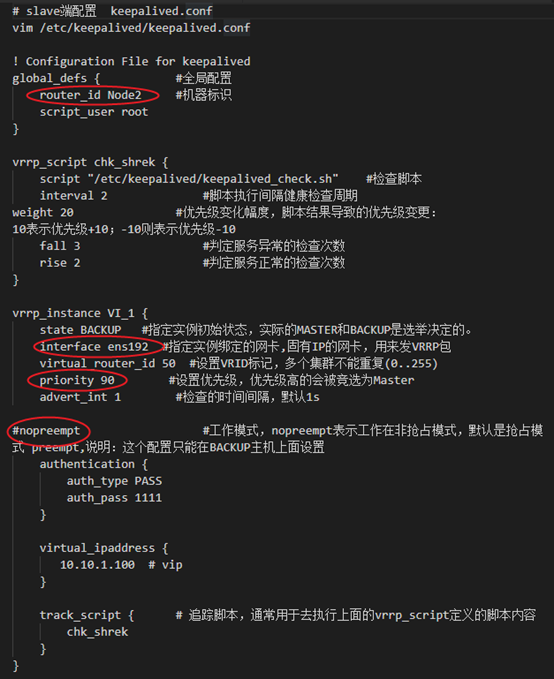
chk\_shrek

}

}

Slave端的配置与master变更项为

* router\_id
* interface
* nopreempt
* priority

其他项与master配置一致；

1. slave启动keepalived

systemctl start keepalived

##### 配置linux服务参数

由于多维库服务用到文件映射，需要配置服务的文件打开数和map\_count，不然容易造成too many open file的异常，请参看 [文件打开数和map\_count](#_文件打开数和map_count)

#### Master服务器

配置master服务前，请确保 [环境准备](#_环境准备) 步骤中已完成，包括（ntp和keepalive服务）

1. 修改master服务配置

# 修改conf/ replication.xml下的配置

vim conf/ replication.xml

<?xml version="1.0" encoding="UTF-8"?>

<config>

<node>

<!-- master设置为1，slave设置为2 -->

<serverId>1</serverId>

<redoPath>/kingdee/mdd/redo</redoPath>

<redoKeepSegments>1024</redoKeepSegments>

<vip>10.10.1.100</vip>

<enableStat>true</enableStat>

<backupPath>/kingdee/mdd/backupDir</backupPath>

</node>

<!-- master服务配置 -->

<master>

<bindPort>9090</bindPort>

<!-- 同步slave节点ID -->

<!-- <synSlavesID></synSlavesID> -->

<masterReplication>true</masterReplication>

<enableWriteDisk>true</enableWriteDisk>

</master>

<!-- slave服务配置 -->

<slave>

<slaveReplication>false</slaveReplication>

<host>10.10.1.2</host>

<port>9090</port>

</slave>

</config>

1. 开启自动备份服务

开启自动备份服务，请参考[**开启自动备份**](#_开启自动备份)

vim conf/web.xml

将interval节点改为360，代表开启自动备份服务，每360分钟检查一次

<!-- 备份时间间隔，单位为分，-1表示关闭自动备份服务 -->

**<context-param>**

**<param-name>**interval**</param-name>**

**<param-value>360</param-value>**

**</context-param>**

<!-- 服务的提供结构，为自动备份设置 -->

1. 删除默认的全备、增量规则

主服务器 不开启全量、增量备份

rm –f conf/ruleList.json

1. 启动master服务

sh bin/start.sh start

1. 初始化密码

如何初始化密码，参看 [初始化密码](#_初始化密码)

1. 验证主服务器状态

配置成功后登录主页，会有Master状态标识；

主服务 <http://10.10.1.1:8080/bos-olap-webserver/#/admin>

Vip服务 <http://10.10.1.100:8080/bos-olap-webserver/#/admin>



#### Slave服务器

配置slave服务时，请确保 [环境准备](#_环境准备) 步骤中已完成，包括（ntp和keepalive服务）

1. 修改slave服务配置

# 修改conf/ replication.xml下的配置

vim conf/ replication.xml

<?xml version="1.0" encoding="UTF-8"?>

<config>

<node>

<!-- master设置为1，slave设置为2 -->

<serverId>2</serverId>

<redoPath>/kingdee/mdd/redo</redoPath>

<redoKeepSegments>1024</redoKeepSegments>

<vip>10.10.1.100</vip>

<enableStat>true</enableStat>

<backupPath>/kingdee/mdd/backupDir</backupPath>

</node>

<!-- master服务配置 -->

<master>

<bindPort>9090</bindPort>

<!-- 同步slave节点ID ，默认注释，为异步模式-->

<!-- <synSlavesID></synSlavesID> -->

<masterReplication>false</masterReplication>

<enableWriteDisk>true</enableWriteDisk>

</master>

<!-- slave服务配置 -->

<slave>

<slaveReplication>true</slaveReplication>

<host>10.10.1.1</host>

<port>9090</port>

</slave>

</config>

1. 开启自动备份

vim conf/web.xml

将interval配置改为360，代表开启自动备份服务，每360分钟检查一次

<!-- 备份时间间隔，单位为分，-1表示关闭自动备份服务 -->

**<context-param>**

**<param-name>**interval**</param-name>**

**<param-value>360</param-value>**

**</context-param>**

<!-- 服务的提供结构，为自动备份设置 -->

1. 开启全备或者增备

默认出厂会有一个全备和增备配置文件在conf/ruleList.json，请检查服务器上这个文件存不存在；文件内容：

{"list":[{"jobName":"BACKUP\_JOB\_1605973805546","backupType":"Increment","rule":"0 0 0 ? \* MON,TUE,WED,THU,FRI,SAT"},{"jobName":"BACKUP\_JOB\_1605973786900","backupType":"Total","rule":"0 0 0 ? \* SUN"}]}

配置成功后，在界面上可以看到是个crontab表达式定义的定时器，可新增或删除定时任务；



可以看到，出厂是周一到周六凌晨进行增量备份；周日进行全量备份；

1. 启动slave服务

sh bin/start.sh start

1. 配置slave服务密码与master密码一致

如何初始化密码，参看 [初始化密码](#_初始化密码)

1. 验证从服务器状态

配置成功后登录主页，后会有Slave状态标识；

从服务 <http://10.10.1.2:8080/bos-olap-webserver/#/admin>



#### 启动健康监控服务（仅限高可用）

##### 配置健康监控服务器ip地址

配置文件在conf/autoProxyConf.xml

vi conf/autoProxyConf.xml

配置样例

<?xml version="1.0" encoding="UTF-8"?>

<healthMonitor>

<!-- 进行任务（健康报告同步任务、数据对比任务等）的OLAP服务1，格式为 IP:Port

例如：127.0.0.1:8080 -->

**<config key="serverIP1" value="10.10.1.1:8080"/>**

<!-- 进行任务（健康报告同步任务、数据对比任务等）的OLAP服务2，格式为 IP:Port

例如：127.0.0.1:8081 -->

**<config key="serverIP2" value="10.10.1.2:8080"/>**

<!-- 数据对比任务：定时对两个Shrek服务进行对比，包括cube数量、顺序；

每个cube的元数据、数据。 -->

<!-- 是否开启数据对比任务，“true”或者“false”。 -->

<config value="true" key="executeComparisonTask"/>

<!-- 初次数据对比任务的时间：时、分、秒 -->

<config value="00" key="comparisonHour"/>

<config value="00" key="comparisonMinute"/>

<config value="00" key="comparisonSecond"/>

<!-- 数据对比任务的执行间隔，单位为秒 -->

<config value="86400" key="comparisonTaskInterval"/>

<!-- 每个数据对比任务进行数据对比的次数 -->

<config value="3" key="comparisonNumber"/>

<!-- 每个数据对比任务进行数据对比的间隔 -->

<config value="1800" key="comparisonInterval"/>

</healthMonitor>

一般变更上面ip和端口就可以，这里的ip为真实ip地址（非虚拟ip）； 这里的端口是web.xml里配置的port端口：

* 修改serverIP1值为主机器ip:port（10.10.1.1:8080）；
* 修改serverIP2值为从机器ip:port（10.10.1.2:8080）；

##### 启动健康监控

启动健康监控

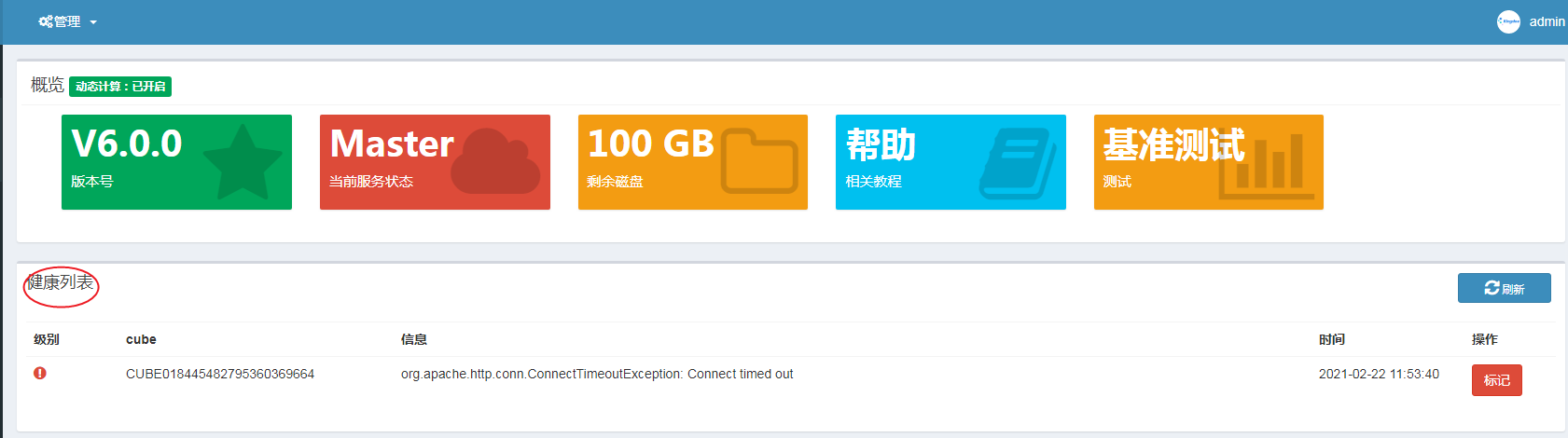
sh bin/autoProxy.sh start

查看监控服务启动情况

ps –ef|grep OlapAutoProxyMain



Web界面页面上查看健康列表



### 动态计算配置

动态计算在新版本是默认开启的，由于web界面更新并不会更新conf目录，所以有些旧版本一直未开启动态计算；如果需要启用动态计算功能：

1. 修改conf/web.xml里的dynamicCalc为true

<!-- 启用动态计算功能 -->

**<context-param>**

**<param-name>**dynamicCalc**</param-name>**

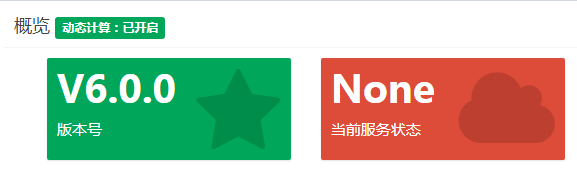
**<param-value>**true**</param-value>**

**</context-param>**

1. 重启多维库服务

sh bin/start.sh restart

可以在web界面主页上看到，动态计算：已开启



### 开机自启动

如果当前状态为Master 或 Slave，则开机自启动则不需要配置；这是因为如果配置了主从，关机重启，启动顺序可能会造成主从状态置换，造成数据的错误同步，所以配置了主从，可以不配置开机自启动。





如果当前状态为None，则需要设置开机自启动；

服务主机宕机或突然断电后，多维数据库需跟随主机一起启动

以centos7为例，加入crontab 列表

# 注意当前用户

crontab -e

@reboot olapserver/bin/start.sh start

这里需要注意：

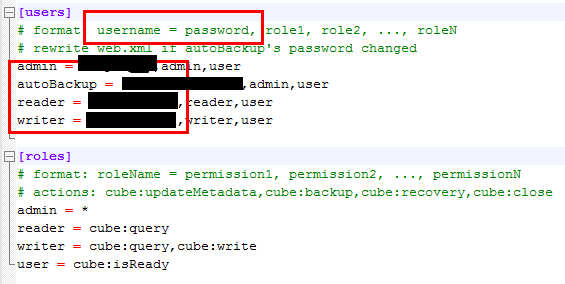
1、crontab每个用户是独立的，请与多维数据库服务文件夹权限的用户保持一致

2、多维数据库服务启动后外部可能访问不了，请确认防火墙已添加端口例外

### 用户名密码

用户名密码在olapserver/conf/shiro.ini

找到[users]属性，格式为 用户名=密码，角色1，角色2。。。。。。



#### 忘记密码

1.shiro.ini文件里密码已加密，第一次部署服务启动时，需要设置密码

<http://localhost:8080/bos-olap-webserver/#/>

这里默认管理员用户名为admin，密码为你设置的密码



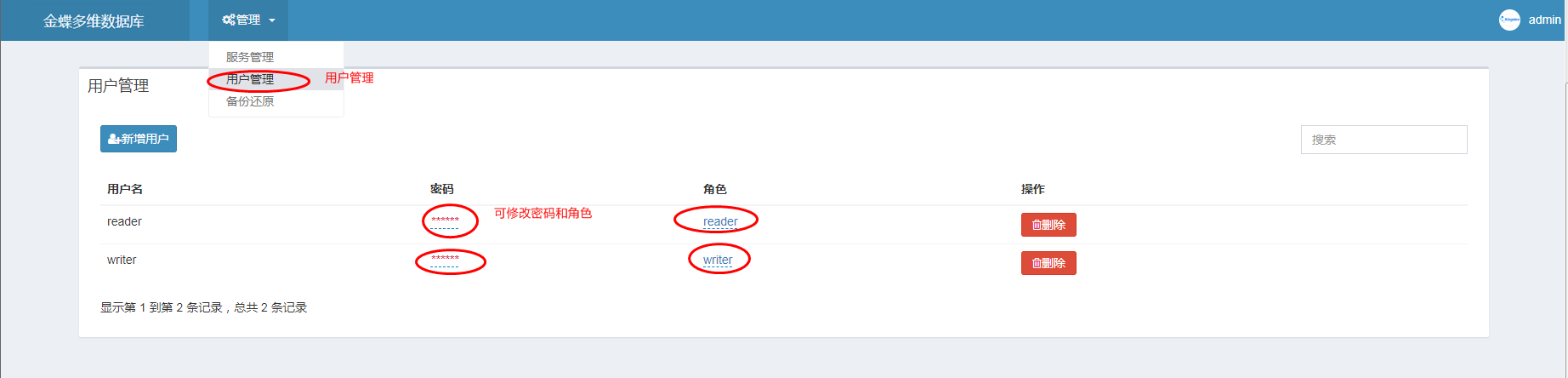
2.管理员忘记密码，可执行bin目录下的resetAdminPW.sh脚本重置

# 先停服务

sh bin/start.sh stop

# 重置管理员密码

sh bin/resetAdminPW.sh

3.其他普通用户忘记密码，可用管理员登陆后台修改对应的用户的密码和角色

### 文件打开数和map\_count

1、查看每个进程可以打开的文件数目，默认为1024。

#查看每个进程可以打开的文件数目，默认为1024

ulimit –n

2、将每个进程可以打开的文件数目加大到1048576

#修改每个进程可以打开的文件数目1048576

ulimit -n 1048576

#查看修改后的数目

ulimit –n

上述修改仅适用于登录shell会话期间，需要永久保存则需要修改limits.conf

3、切换到root权限，修改limits.conf

#切换为root用户

su -

#修改配置文件

vim /etc/security/limits.conf

#将软硬限制修改为如下

\* soft nproc 1048576

\* hard nproc 1048576

\* soft nofile 1048576

\* hard nofile 1048576

#说明：

# \* 代表针对所有用户，noproc 是代表最大进程数，nofile 是代表最大文件打开数

# 软限制(soft)：指Linux在当前系统能够承受的范围内进一步限制用户同时打开的文件数

# 硬限制(hard)：指根据系统硬件资源状况（系统内存）计算出来的系统最多可同时打开的文件数量

# 也可用 \* - noproc 1048576 \* - nofile 1048576代替，-的意思为soft 和hard 全部限制 。

3、修改map\_count

# 切换到root用户

sysctl -w vm.max\_map\_count=1048576

# 查看修改后的结果显示：vm.max\_map\_count = 1048576

sysctl -a|grep vm.max\_map\_count

# 上述方法修改之后，如果重启虚拟机将失效，所以需在/etc/sysctl.conf追加

vm.max\_map\_count=1048576

### linux内存分配策略

注意：以下操作以root用户进行，如何切换root用户在上文中已提及，此处不赘述

1、查看当前linux机器的内存分配策略，如果内存分配策略不为0，那么需要进行第二步和第三步的操作

# 查看当前linux机器的内存分配策略

sysctl -a|grep vm.overcommit\_memory

# 查看结果如下所示，如果不等于0，那么需要进行第二步和第三步的操作

vm.overcommit\_memory = 0

2、修改系统配置文件: /etc/sysctl.conf

# 修改系统配置文件/etc/sysctl.conf，检查是否包含这行内容：vm.overcommit\_memory=0

vi /etc/sysctl.conf

# 如果没有上述的那行内容，那么需要在文件末尾加上一行：vm.overcommit\_memory=0（注意没有空格）

3、使修改的配置文件生效

# 输入下列命令

sysctl -p

### 数据源配置

**最新版本的数据源可在mc里配置，有问题可联系相应的运维修改；**

旧版本的配置：

多维数据库服务启动成功后，需进行数据源配置，以合并报表为例。

🡪打开并登陆金蝶云苍穹

🡪法定合并

🡪合并体系

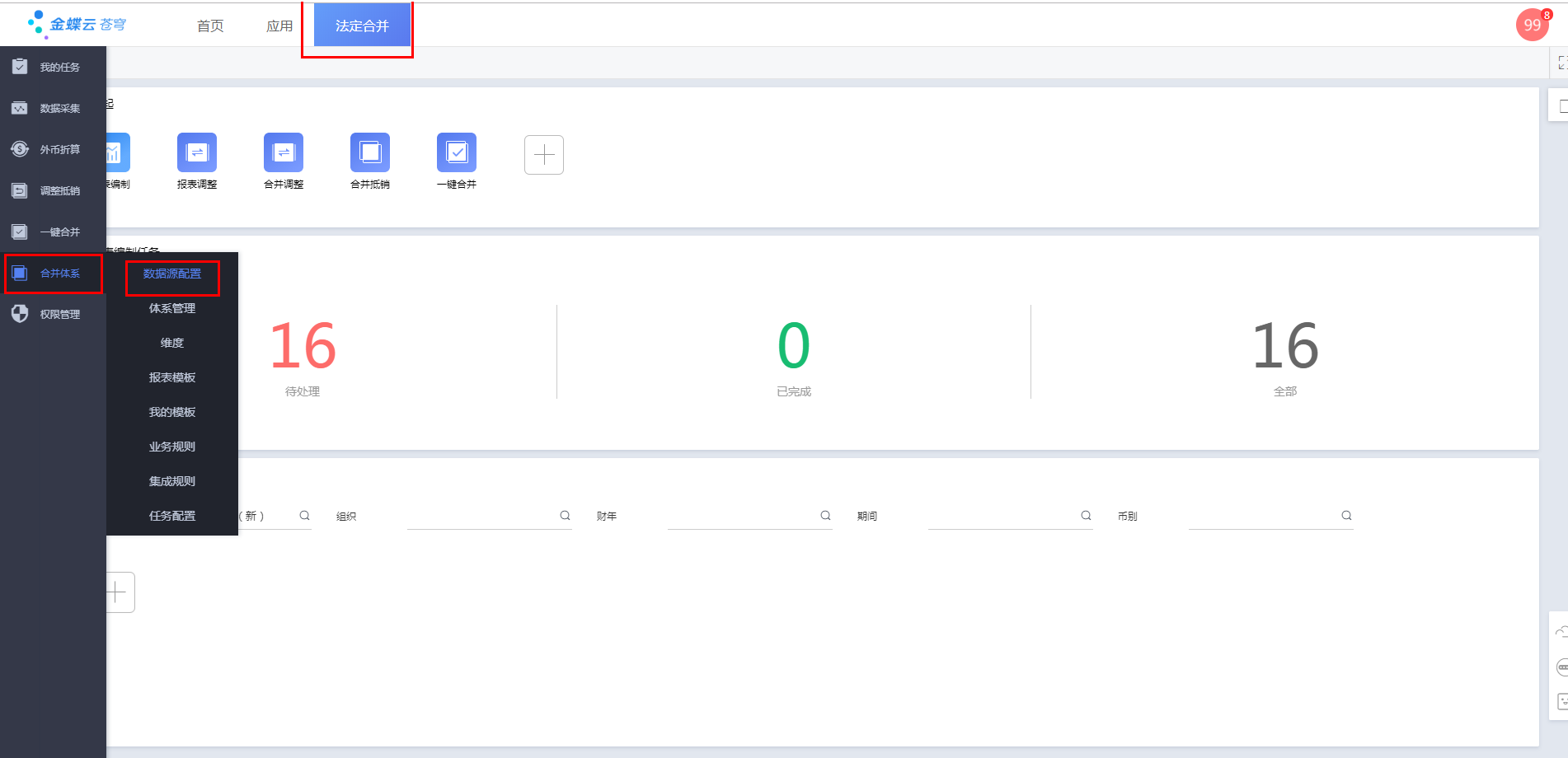
🡪数据源配置

🡪新增

🡪验证并保存

🡪通知业务或测试人员查看是否正常。

打开数据源配置



新增数据源配置，用户名密码在conf/shiro.ini对应。



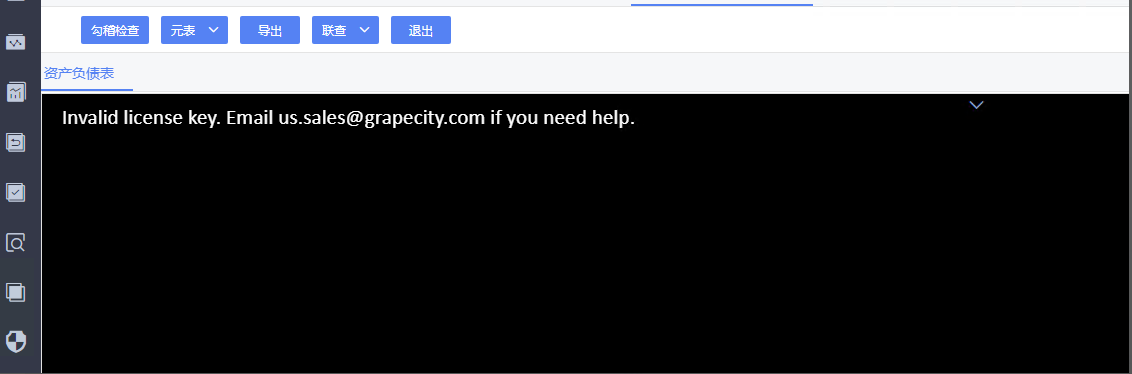


通知业务或测试人员进行验证合并报表应用数据是否能够存储正常。

### 报表插件spreadJs表格控件授权

私有云请联系总部苍穹云产品支持部获得

spreadjs表格的部署许可授权，否则，无法正常使用报表。



## 四、运维

### 多维数据库服务关闭

注意：为防止olap数据库文件损坏等情况，停止多维数据库服务请使用安全关闭多维数据库服务方式，禁止使用kill 或 kill -9 pid命令杀进程方式，脚本里使用的kill -15 pid 信号量处理方式，进程捕捉到后会有相应的后台代码进行处理，优先使用脚本安全关闭。

#### None服务关闭

1、通过脚本安全关闭

sh olapserver/bin/start.sh stop

# olap server had been stopped.

会等待服务安全关闭，脚本提示已经停止

2、选择管理-服务管理，找到关闭服务页签



点击图中关闭按钮，服务停止

#### 高可用服务关闭

高可用关闭顺序是：

1. 停slave多维库服务
2. 停slave的keepalived
3. 停master 多维库服务
4. 停master的keepalived

关闭方式可以选择上述的通过脚本关闭，和管理-服务管理界面关闭

### 多维数据库备份与还原

多维库正确部署的备份策略如下：

|  |  |  |  |
| --- | --- | --- | --- |
| 是否高可用服务 | 状态 | 自动备份服务 | 定时全备、增备 |
| 非高可用 | None | 需开启 | 不用配置 |
| 高可用主从 | Master | 需开启 | 不用配置 |
| Slave | 需关闭 | 需要开启  请参看 [定时全备、增备（仅限高可用主从）](#_定时全备、增备（仅限高可用主从）) |

这里出现两个备份机制：

* 自动备份服务
* 定时全备、增备

这两个备份机制是针对不同场景的，自动备份服务是针对单个cube的数据备份，还原时是单个cube数据还原的； 而定时全备、增备是针对整个多维库服务节点的数据备份，还原时是整个多维库服务节点的数据还原；由于定时全备对服务性能影响较大，目前仅在高可用主从中的从服务器节点上启用；

#### 备份整个多维库

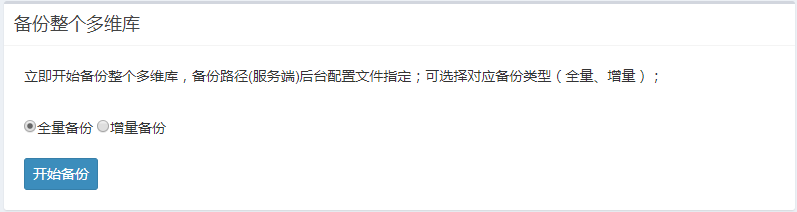
##### 自动备份服务

测试环境可以不用开启自动备份服务，正式环境建议开启；参看 [开启自动备份](#_开启自动备份)

默认出厂是每隔720分钟检查整个多维库（所有cube）一次，如果对应的cube数据有修改，则产生对应cube的全备文件在根目录下backup/CUBE…/CUBE….backup下，如果对应的cube数据未修改，则不产生对应的备份文件；

* 720分钟是12小时定时检查一次，正式环境可按需改成10分钟-360分钟检查备份一次；
* 自动备份服务不会锁全库，只会锁单个cube，一般影响较小；

##### 手动全备、增备（仅限高可用主从）



全量备份会锁整个库，可能耗时比较长，请谨慎操作

##### 定时全备、增备（仅限高可用主从）

目前只有Slave状态才开启定时全备、增备策略

默认出厂Slave会有全备和增备备份策略，配置参看 [Slave服务器](#_Slave服务器)

在Master或者Slave服务节点，【管理】-【备份还原】-【定时备份策略配置】如果看到上述界面，表示周一到周六凌晨进行增量备份；周日进行全量备份；

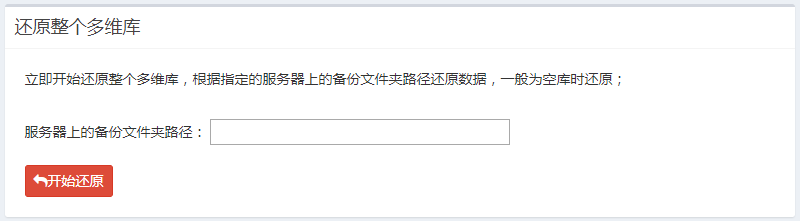
定时备份启用，可以观察对应的备份文件列表，和下次备份时间；





#### 还原整个多维库（高可用全量、增量）

目前只有None状态且空库的时候才允许还原整个库 服务器路径示例



#### 备份单个cube

一个cube对应关系库中一个体系，多维数据库是通过体系编码来区分和标识体系的，

体系编码格式如下：

CUBE018445482795360369664

CUBE06445482795326815232

CUBE02445482795293260800

……

1. 在关系库中找到对应的体系编码



1. 在多维库找到对应的cube，可以搜索或直接地址定位：

输入 test 或者 CUBEtest798443781484447744，搜索结果即可

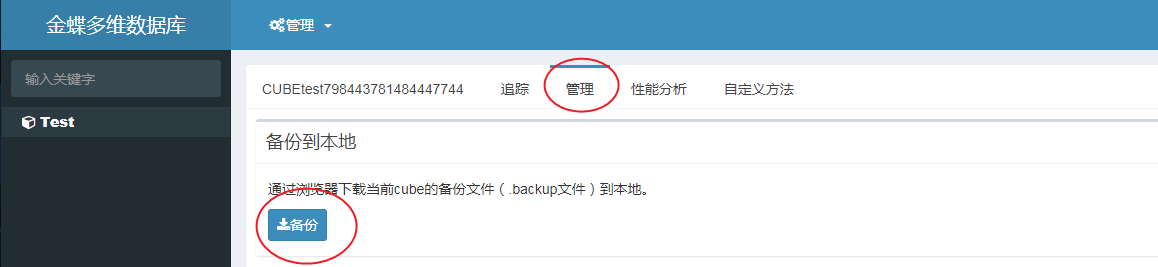


或者直接链接地址定位

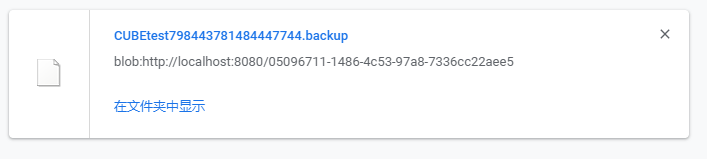
<http://localhost:8080/bos-olap-webserver/#/admin/cube/CUBEtest798443781484447744>



1. 选择【管理】选项，点击备份按钮，进行单个cube备份



备份完成



#### 还原单个cube

1. 在关系库中找到对应的体系编码

你要还原体系编码CUBEtest798443781484447744



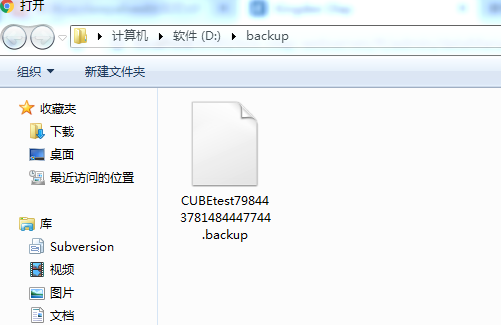
1. 进入备份还原页面，单个cube还原



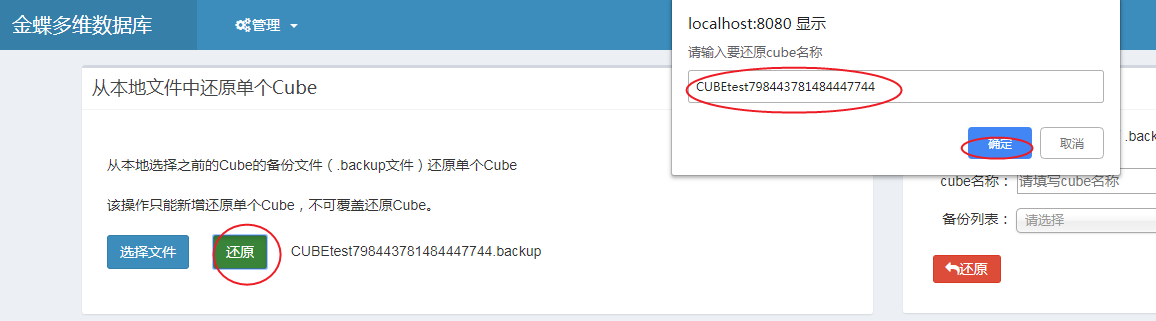
从本地文件中还原单个Cube，点击选择文件



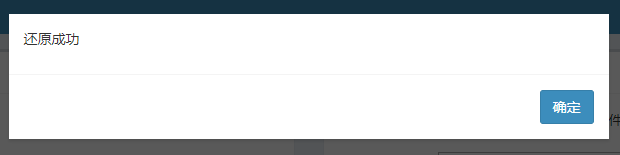
在弹出的选择框中，选择对应的备份文件



点击还原，填入要还原的cube名称 CUBEtest798443781484447744



还原成功



### 多维数据库服务更新

#### web界面更新

若已配置高可用，当前服务状态显示为Master或Slave，则禁止通过web界面进行更新





若当前服务未配置高可用，可以看到当前服务状态显示为None，可以通过web界面进行更新，更新并不会修改配置文件夹conf下内容，如果需要修改配置，则需要手动修改



更新包bos-olap-setup.zip从官网下载结构：

.

├── 6.0.0 版本号文件夹(格式x.x.x)，包含该版本的lib包与web应用

├── bin 脚本文件夹，启动脚本等其他工具脚本

├── conf 配置文件夹，web.xml与shiro.ini等配置

└── docs doc文档文件夹，部署和说明文档

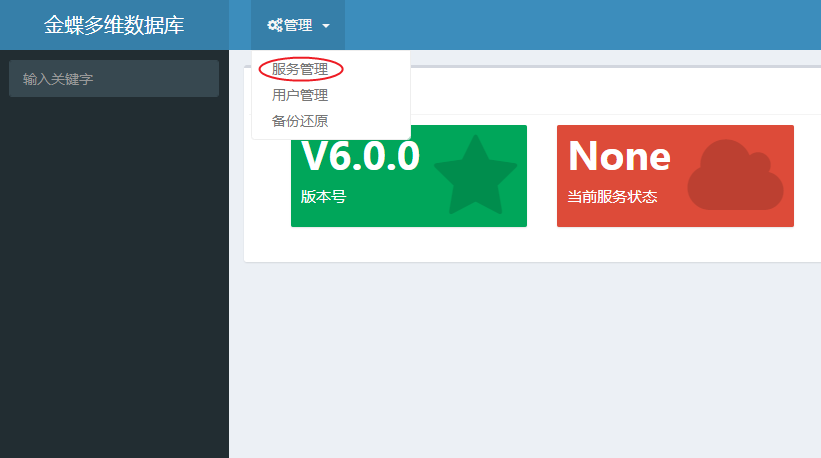
通过界面进行安装包上传与升级切换：

1.登录多维数据库后台管理界面

<http://localhost:8080/bos-olap-webserver>



2.点击管理菜单下的服务管理，进入管理界面

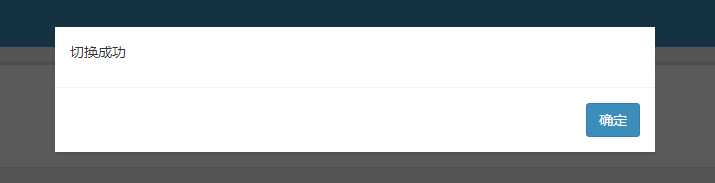


3. 找到升级多维数据库服务，进行升级

* 上传要相应的安装包
* 选择对应的版本
* 切换到指定版本



切换成功



#### 手动更新

web界面更新实际上是更新了下面两个目录：

.

├── 6.0.0 版本号文件夹(格式x.x.x)，包含该版本的lib包与web应用

├── bin 脚本文件夹，启动脚本等其他工具脚本

所以可以手动删除（或备份）旧的版本目录（5.9.9）和bin目录，然后更新目录就可以

# 停止多维库服务

sh bin/start.sh stop

# 删除旧的文件夹5.9.9和bin

rm –f 5.9.9 bin

# 更新

cp -r upload/6.0.0 olapserver/

cp -r upload/bin olapserver/

# 启动多维库服务

sh bin/start.sh start

##### 高可用服务更新

高可用更新顺序是：

1. 停slave多维库服务
2. 停slave的keepalived
3. 停master 多维库服务
4. 停master的keepalived
5. 手动更新master服务
6. 启动master的keepalived
7. 启动master的多维库服务
8. 手动更新slave服务
9. 启动slave的keepalived
10. 启动slave的多维库服务

可以在master 和slave分别写对应的脚本放到olapserver下面

stopHa.sh 样例：

vim stopHa.sh

# 停止多维库服务

sh bin/start.sh stop

# 停止keepalived

systemctl stop keepalived

updateHa.sh 样例：

vim updateHa.sh

# 旧版本号5.9.9可以通过bin/version获取

# cat bin/version

# 删除旧的文件夹5.9.9和bin

rm –f 5.9.9 bin

# 更新

cp -r upload/6.0.0 olapserver/

cp -r upload/bin olapserver/

startHa.sh 样例：

vim startHa.sh

# 启动keepalived

systemctl start keepalived

# 启动多维库服务

sh bin/start.sh start

最终通过脚本手动更新步骤

# 分别上传新版本文件夹到olapserver/upload（master和slave都需要上传）

# 停止slave端高可用

sh stopHa.sh

# 停止master端高可用

sh stopHa.sh

# 手动更新master

sh updateHa.sh

# 启动master高可用

sh startHa.sh

# 手动更新slave

sh updateHa.sh

# 启动slave高可用

sh startHa.sh

启动成功后，首页查看服务状态

主服务 <http://10.10.1.1:8080/bos-olap-webserver/#/admin>

Vip服务 <http://10.10.1.100:8080/bos-olap-webserver/#/admin>



从服务 <http://10.10.1.2:8080/bos-olap-webserver/#/admin>



##### 历史版本服务的更新

这里说的历史版本是指服务版本过旧（小于5.0.0版本），并没有web界面更新功能（5.0.0之后），参看 [历史版本目录](#_历史版本目录) 判定是否是历史版本目录

如果判定为5.0.0或者2.x版本的数据库服务，进行手动升级；

手动升级步骤：

* 记下旧服务的端口，停止多维库服务

旧服务器的端口可通过看进程号获取，或者直接查看start.sh

5.0.0之前的版本

sh start.sh stop

# olap server had been stopped.

2.x版本

找到界面关闭服务



通过点击关闭按钮，关闭服务；

* 备份整个多维库文件夹

压缩或拷贝整个多维库目录

* 合并文件夹

将旧服务的backup database olaplogs三个目录放置到新安装包目录下；

服务安装包bos-olap-setup.zip目录结构

.

├── 6.0.0 版本号主体程序包文件夹(格式x.x.x)，包含lib包与web应用

├── bin 脚本文件夹，启动脚本等其他工具脚本

├── conf 配置文件夹，web.xml与shiro.ini等配置

└── docs docs文档文件夹，相关介绍说明、部署和说明文档

则合并后的目录结构：

.

├── 6.0.0 版本号主体程序包文件夹(格式x.x.x)，包含lib包与web应用

├── backup 自动备份服务备份文件夹（如果没有，启动服务自动创建）

├── database 数据库文件夹（如果没有，启动服务自动创建）

├── bin 脚本文件夹，启动脚本等其他工具脚本

├── conf 配置文件夹，web.xml与shiro.ini等配置

├── olaplogs 所有日志文件夹（如果没有，启动服务自动创建）

└── docs docs文档文件夹，相关介绍说明、部署和说明文档

* 恢复用户名与密码

找到旧服务的webapp/WEB-INF/shiro.ini，将其拷贝并覆盖合并后目录文件conf/shiro.ini

* 恢复服务端口

vim olapserver/conf/web.xml

<!— 服务端口 -->

**<context-param>**

**<param-name>**port**</param-name>**

**<param-value>8080</param-value>**

**</context-param>**

* 启动服务

sh bin/start.sh start

### 多维数据库服务迁移

多维数据库服务从一台服务器迁移到另一台服务器：

**注意：拷贝数据一定要先停服务，不停服务，一边在写数据，一边在拷贝数据，拷贝后的数据文件不全，会造成数据库损坏，所以拷贝或压缩数据一定要保证多维库服务已停止；**

将服务停止后，打包olapserver文件夹迁移到其他服务器解压启动即可。

#停止多维数据库服务

sh olapserver/bin/start.sh stop

# olap server had been stopped.

#打包多维数据库服务文件夹

zip –r olapserver.zip olapserver

在新服务器上

#解压zip包

unzip olapserver.zip

#打包多维数据库服务文件夹

sh olapserver/bin/start.sh start

#olap server had been started.

如果不需要保留数据和日志备份文件，请从官网下载安装包或者找到当时安装包解压启动。

按需修改服务端口

按需开启备份服务，变更主从机器，角色，ip，scp参数

修改olapserver/conf/shiro.ini

按需修改用户名、密码，角色

详细步骤参考安装过程。

受影响的文件有web.xml shiro.ini

差异：服务端口、是否开启自动备份、用户名、密码、主备角色

启动成功后需要重新配置应用的数据源，修改为新版本的数据源ip和端口

详情参看[数据源配置](#_合并报表应用：数据源配置_1)

## 五、常见问题处理

### 常见问题

1. 端口被占用

提示信息：

Exception in thread "main" java.net.BindException: Address already in use: bind.

常发生在启动时，已经有程序占用端口，提示报这信息。

1. 文件打开数过多

提示信息：

Too many open files

服务器会对每个进程打开文件数有相关限制，用ulimit –n查看和修改打开文件数限制

1. 服务停不下来

执行sh bin/start.sh stop时，一直卡在please waiting olap server stopped...

这个问题是由于资源未释放原因造成的，现有可能出现的原因是：有可能数据库正在备份；

排查办法：

1. 找到对应进程的pid

# 查看olap相关进程

ps -ef|grep OlapWebServerMain

1. 使用jstack命令查看堆栈信息

# 查看进程pid对应的堆栈

jstack –l pid

查看对应的堆栈信息

如果存在关键堆栈

at java.util.concurrent.ThreadPoolExecutor.awaitTermination(ThreadPoolExecutor.java:1475)

at kd.bos.olap.schedule.BackupFlow.close(BackupFlow.kt:491)

at kd.bos.olap.server.HttpOlapServer.close(HttpOlapServer.kt:301)

上述堆栈中存在：

ThreadPoolExecutor.awaitTermination(ThreadPoolExecutor.java:1475)

BackupFlow.close(BackupFlow.kt:491)

则判定为数据库正在备份，导致服务未能正常停止；数据库备份造成一直卡在please waiting olap server stopped...只能等待数据库备份完成后，正常关闭服务器；

1. jvm内存不足

提示信息：

java.io.IOException: Map failed

在启动脚本里加入java -Xmx40480m -Xms2048m 启动参数

1. IPVS

启动keepalived，发现启动后又停止了，查看日志cat /var/log/messages (或查看状态，systemctl status keepalived) 可以看到下面字样

IPVS: Can't initialize ipvs: Protocol not available

解决：这个问题是Ip\_vs模块没加载造成的

1. 查看是否ip\_vs是否已加载

lsmod | grep ip\_vs

1. 如果没加载，手动加载

modprobe ip\_vs #动态加载ip\_vs模块

modprobe ip\_vs\_wrr

1. 上述加载仅仅是手动加载，需要开机自动挂载内核模块

echo “modprobe ip\_vs” >> /etc/rc.sysinit

echo “modprobe ip\_vs\_wrr” >> /etc/rc.sysinit

1. 如未解决请核对主从发行版本是否一致，之前遇到问题是centos 7.2存在该问题
2. 启动主从后，从状态显示为None。
3. 查看主从replication\_json.json配置是否正确
4. 查看vip是否在Master机器上，Slave没有vip
5. 查看Slave端olaplogs/replication/replication.log有Disagree字符。
6. Mater异常（如主机异常断电、重启、磁盘不足）后，数据是否可用？

不可用，以下步骤进行主从重建

1. 执行scp脚本，将备份文件从Slave机器拷贝到Master。
2. 修改Mater的replication\_conf.json，设置master的masterReplication及slave的replication均为false，启动多维库后显示状态为None，此时可通过全量备份文件进行数据恢复。
3. 重新配置Replication及KeepAlived，建立主从。
4. 主从两台机器出现双vip
5. 关闭Master及Slave机器的防火墙

shell执行：systemctl stop firewalld 以及systemctl stop iptables,

1. 重启Master的KeepAlived，接着重启Master的KeepAlived
2. VIP访问不了，但查看Master机器ip a的确存在

重启Master的KeepAlived，接着重启Master的KeepAlived

### 注意事项

下面是常见的由于配置与操作不当造成多维数据库文件损坏或进程崩溃：

1. 启动服务时，未正确配置 [文件打开数和map\_count](#_文件打开数和map_count)
2. 停止服务时，未按 [多维数据库服务关闭](#_多维数据库服务关闭) 去关闭，或者直接使用kill命令

这种操作会造成数据库文件写入不完整，且不能修复；

1. 备份数据时，未安全停止服务便拷贝database目录，或者自己编写备份database文件

夹脚本，脚本里并没有停止服务；

这种操作会造成数据库文件写入不完整，且不能修复；

如果需要备份多维数据库，请参考 [多维数据库备份与还原](#_多维数据库备份与还原) ，不建议自己写定时脚本备份database目录；

1. 启用了高可用，但是主Master机器中keepalived并没有配置成非抢占模式（nopreempt）；

这种配置有可能会造成主从来回切换，易造成数据丢失与主从数据损坏；

1. 高可用主从自动切换后，此时原先的主宕机，原先的从切换为主；试图停止现有主（原

先的从）keepalived，重启原先宕机的主，这种操作会造成主从数据损坏；