达梦数据

全栈数据产品与解决方案提供商

DDM
安装手册

达梦新云文档数据库

目录

第一章 产品介绍

本手册将给需要安装达梦新云文档数据库单机和集群模式的用户，给予安装指导和基本组件说明。

在安装 DDM 之前，请用户仔细阅读本手册，本手册包含了重要的安装指导信息。在安装开始之前，首先应该检查所得到的 DDM 产品是否完整，并准备好 DDM 所需的硬件环境、软件环境。

本章主要介绍在安装 DDM 产品前需要进行的准备工作。

1.1. 硬件环境需求

用户进行选型时，需要结合客户产品的业务场景、性能要求、工作负载、数据量大小、持久化要求等因素。应根据 DDM 及应用系统的需求来选择合适的硬件配置，如 CPU 的指标、内存及磁盘容量等。档次一般应尽可能高一些。下面给出安装单机模式 DDM 所需的硬件基本配置：

表 1.1 硬件环境需求

运算符	最低配置	最高配置
CPU	2.0 GHz 4 核心 64 位处理器	3.5 GHz 32 核心 64 位处理器 或更高
内存	8GB DDR4 频率 3000MHz	256GB DDR4 频率 3000MHz 或更高
硬盘	50GB 机械硬盘	不低于 1TB 建议使用 SSD

1.2. 软件环境需求

运行 DDM 所要求的软件环境主要有：

表 1.2 软件环境需求

名称	要求
操作系统	Linux(内核 2.6.18 及以上)
网络协议	TCP/IP 协议
系统盘	至少 1G 以上的剩余空间

此外，如要进行数据库应用开发，正常安装配置应用开发环境即可。DDM 安装包已经提供主流应用开发语言连接的数据驱动。

1.3. 计算机管理员准备工作

在安装 DDM 之前，计算机管理员应当首先完成安装前的准备工作。

主要准备工作有：

1. 正确地安装操作系统、合理地分配磁盘空间、检查机器配置是否满足要求；
2. 关闭正在运行的杀毒、安全防护等软件；
3. 保证网络环境能正常工作；
4. 关闭防火墙或添加白名单、关闭 SELinux；

1.4. 第三方代码及协议信息

DDM 数据库产品中使用了一些第三方代码，所有 DDM 服务器与接口使用的第三方代码名称及版本声明如下，对应的第三方代码的 LICENSE 放在 DDM 数据库安装目录的 thirdparty 子目录中。

TIKV

DDM V1.0 的 ddm_store 使用了 TIKV，该软件的许可协议基于 Apache License Version 2.0，其许可协议文件见上述文件夹的“APACHE_LICENSE-2.0”。

PD

DDM V1.0 的 ddm_meta 使用了 pd，该软件的许可协议基于 Apache License Version 2.0，其许可协议文件见上述文件夹的“APACHE_LICENSE-2.0”。

BR

DDM V1.0 的 ddm_br 使用了 BR，该软件的许可协议基于 Apache License Version 2.0，其许可协议文件见上述文件夹的“APACHE_LICENSE-2.0”。

OPENSSL

DDM 服务器的加密使用了 OPENSSL，该软件的许可协议基于 Apache License Version 2.0，其许可协议文件见上述文件夹的“APACHE_LICENSE-2.0”。

TOML

DDM 服务器的 ddm_engine 使用了 TOML，该软件的许可协议基于 The MIT License（MIT）。其许可协议文件见上述文件夹的“MIT_LICENSE”。

GEO

DDM 服务器的 ddm_engine 使用了 GEO，该软件的许可协议基于 Apache License Version 2.0，其许可协议文件见上述文件夹的“APACHE_LICENSE-2.0”。

SNAPPY

DDM 服务器的 ddm_engine 使用了 SNAPPY，该软件的许可协议基于 BSD-3-Clause license。，其许可协议文件见上述文件夹的“BSD-3-Clause license”。

Compress

DM 服务器的 ddm_engine 使用了 Compress，该软件的许可协议基于 Apache License Version 2.0 以及 MIT License。其许可协议文件见上述文件夹的“APACHE_LICENSE-2.0”，“MIT_LICENSE”。

go-pcre

DDM 服务器的 ddm_engine 使用了 go-pcre，该软件的许可协议基于 The MIT License（MIT）。其许可协议文件见上述文件夹的“MIT_LICENSE”。

Pbkdf2

DM 服务器的 ddm_engine 使用了 Pbkdf2，该软件的许可协议基于 Apache License Version 2.0，其许可协议文件见上述文件夹的“APACHE_LICENSE-2.0”。

Regexp2

DDM 服务器的 ddm_engine 使用了 Regexp2，该软件的许可协议基于 The MIT License（MIT）。其许可协议文件见上述文件夹的“MIT_LICENSE”。

scram

DDM 服务器的 ddm_engine 使用了 scram，该软件的许可协议基于 Apache License Version 2.0，其许可协议文件见上述文件夹的“APACHE_LICENSE-2.0”。

Crypto

DDM 服务器的 ddm_engine 使用了 scram，该软件的许可协议基于“Copyright（c） 2009 The Go Authors. All rights reserved”，其许可协议文件见上述文件夹的“Go_Authors_LICENSE”。

yaml

DDM 服务器的 ddm_engine 使用了 yaml，该软件的许可协议基于 Apache License Version 2.0 以及 MIT License。其许可协议文件见上述文件夹的“APACHE_LICENSE-2.0”，“MIT_LICENSE”。

第二章 组件介绍

为了快速满足用户需求，DDM 在最短时间内推出了 V1.0 版本，DDMV1.0 版本的计算引擎 ddm_engine 完全自主研发，而存储引擎 ddm_store 和元数据引擎 ddm_meta 是在第三方开源组件的基础上改造而成。为了实现完全自主研发的目标，DDM V2.0 将采用完全自主研发的计算引擎、存储引擎和元数据引擎，预计该版本将于 2025 年年中发布，并实现性能的大幅度提升，目前相关研发工作正在进行中。

2.1. ddm_meta

ddm_meta 集群是整个 DDM 分布式集群的元信息管理模块，负责存储每个 ddm_store 节点实时的数据分布情况和集群的整体拓扑结构。并且根据数据分布状态，进行分发执行节点 ddm_engine 的命令到对应 ddm_store 存储节点执行。

2.1.1. 启动介绍

请按照顺序依次启动指定的组件：ddm_meta、ddm_store、ddm_engine。ddm_meta 集群服务的启动步骤如下：

1. 启动 ddm_meta 服务，并初始化集群：使用命令行启动服务

前台启动方式运行

```
./bin/ddm_meta --name=meta_name --client-urls=http://meta_ip:meta_port
--peer-urls=http://peer_meta_ip:peer_meta_port
--initial-cluster="meta_name=http://peer_meta_ip:peer_meta_port"
```

后台服务启动方式运行

```
setsid ./bin/ddm_meta --name=meta_name
--client-urls=http://meta_ip:meta_port
--peer-urls=http://peer_meta_ip:peer_meta_port
--initial-cluster="meta_name=http://peer_meta_ip:peer_meta_port" &
```

2. 启动 ddm_meta 服务，并加入已初始化的集群：使用命令行启动服务

前台启动方式运行

```
./bin/ddm_meta --name=meta_name --client-urls=http://meta_ip:meta_port
--peer-urls=http://peer_meta_ip:peer_meta_port
--join=http://cluster_meta_ip:cluster_meta_port
```

后台服务启动方式运行

```
setsid ./bin/ddm_meta --name=meta_name
--client-urls=http://meta_ip:meta_port
--peer-urls=http://peer_meta_ip:peer_meta_port
--join=http://cluster_meta_ip:cluster_meta_port &
```

其中 meta_name 表示当前启动的 ddm_meta 服务的节点名称；

meta_ip:meta_port 表示运行当前 ddm_meta 服务的机器 IP 地址和端口号，端口号建议使用 2379；

peer_meta_ip:peer_meta_port，目前 peer_meta_ip 地址建议使用 meta_ip 地址，peer_meta_port 建议使用不同于 meta_port 的端口号；

cluster_meta_ip:cluster_meta_port 表示初始化集群的 ddm_meta 服务的

meta_ip 和 meta_port。

下面就启动 ddm_meta 集群服务举个例子，如下所示，分别启动三个 ddm_meta 服务节点（“meta1”、“meta2”、“meta3”），其中启动“meta1”节点时初始化集群，启动“meta2”、“meta2”节点时加入已初始化的集群：

首先，启动 ddm_meta 节点“meta1”，并初始化集群；

```
./bin/ddm_meta --name=meta1 --client-urls=http://192.168.1.10:2379
--peer-urls=http://192.168.1.10:2380
--initial-cluster="meta1=http://192.168.1.10:2380"
```

其次，启动 ddm_meta 节点“meta2”、“meta3”，并加入已初始化的集群。

```
./bin/ddm_meta --name=meta2 --client-urls=http://192.168.1.20:2379
--peer-urls=http://192.168.1.20:2380 --join=http://192.168.1.10:2379
./bin/ddm_meta --name=meta3 --client-urls=http://192.168.1.30:2379
--peer-urls=http://192.168.1.30:2380 --join=http://192.168.1.10:2379
```

2.1.2. 配置文件说明

当前分布式 ddm_meta 组件不单独使用配置文件，相关配置参数已在组件内自动设置。

2.2. ddm_store

ddm_store 集群是数据存储的核心，ddm_store 存储数据的基本单位是 Region，每个 Region 负责存储一个 Key Range 的数据，一般每个 ddm_store 节点会负责多个 Region。不同 ddm_store 节点上的 region 与其他节点的数据互为备份，由 raft 算法保证对应 region 数据的一致性。

2.2.1. 启动介绍

ddm_store 服务的启动支持以下两种命令行方式。

方式一、使用命令行控制台的方式运行

```
./bin/ddm_store --addr store_ip:store_port --pd-endpoints meta_ip:meta_port
```

方式二、使用后台服务的方式运行

```
setsid ./bin/ddm_store --addr store_ip:store_port --pd-endpoints
meta_ip:meta_port &
```

其中 store_ip:store_port 表示运行 ddm_store 数据库存储服务的机器 IP 地址和端口号，端口号建议使用 10180；

meta_ip:meta_port 表示 ddm_meta 集群服务的 IP 地址和端口号，对应 ddm_meta 服务启动命令中参数-client-urls 的 IP 地址和端口号，多个 ddm_meta 实例请使用逗号`,`隔开,例如：

--pd-endpoints 192.168.1.10:2379,192.168.1.20:2379,192.168.1.30:2379

下面就启动 ddm_store 服务举个例子，如下所示，分别启动三个 ddm_store 服务，并连接同一个 ddm_meta 集群服务：

```
./bin/ddm_store --addr 192.168.1.10:10180  --pd-endpoints
192.168.1.10:2379,192.168.1.20:2379,192.168.1.30:2379
./bin/ddm_store  --addr 192.168.1.20:10180  --pd-endpoints
192.168.1.10:2379,192.168.1.20:2379,192.168.1.30:2379
./bin/ddm_store  --addr 192.168.1.30:10180  --pd-endpoints
192.168.1.10:2379,192.168.1.20:2379,192.168.1.30:2379
```

2.2.2. 配置文件说明

当前分布式 ddm_store 组件不单独使用配置文件，相关配置参数已在组件内自动设置。

2.3. ddm_engine

ddm_engine 负责接收客户端的命令请求，对接收到的命令完成解析、加工、计算、优化等一系列操作后，将命令转换成可向下执行的最优结构，并与存储节点和元数据节点进行通信，获取所需的数据，最后将执行结果返回客户端工具。

2.3.1. 启动介绍

ddm_engine 服务的启动步骤如下：

1. 修改配置文件 ddm_engine_config.toml：配置文件位于/conf/目录中，请根据实际部署架构及需求设置配置文件中的各个配置项参数，配置文件详情请参考 2.3.2 中的说明；
2. 启动 ddm_engine 服务：使用命令行启动服务，支持以下两种命令运行方式。

方式一、使用命令行控制台的方式运行

```
/bin/ddm_engine
```

方式二、使用后台服务的方式运行

```
setsid /bin/ddm_engine  &
```

2.3.2. 配置文件说明

当前版本提供的 ddm_engine 配置文件参数的详细说明如下：

表 2.1 配置文件说明

参数	说明
host	ddm_engine 服务运行所在的服务器 IP 地址。
port	ddm_engine 服务运行的端口号，默认值为 27017。
printable	是否打印 ddm_engine 的调试信息，包含两个选项：true、false，分别代表打印和不打印，默认值为 false。
[ddm_meta]	
ddm_meta	ddm_meta 元数据管理服务运行的服务器 IP 地址和端口号，对应 ddm_meta 服务启动命令中参数-client-urls 的 IP 地址和端口号，例如：ddm_meta="192.168.1.10:2379"； 若当前部署架构是分布式多 ddm_meta 实例的情况，请使用逗号`,`隔开；例如：ddm_meta= "192.168.1.10:2379,192.168.1.20:2379,192.168.1.30:2379"。
[log]	
level	日志文件记录等级，支持五个等级：debug、info、warn、error、fatal。
format	日志文件记录格式，支持两种格式：json、text。
[security]	
authorization	启用或禁用基于角色的访问控制，用于管理用户对数据库资源和操作的访问，包含两个选项：enabled、disabled，分别代表启用和禁用，默认值为 disabled，具体描述如下：

	(1) enabled：用户只能访问被授予权限的数据库资源和权限； (2) disabled：用户可以访问任何数据库并执行任何操作。

第三章 许可证(License)的安装

用户安装 DDM 时，可导入相应的许可证(License)。License 的载体是一个加密文件 ddm_license.key，内容是达梦公司对用户使用 DDM 软件的授权。DDM 软件安装后，如果需要得到更多授权的用户，可联系达梦公司获取相应的 License，并按照下面的操作方法进行安装。

- **初次部署 DDM，license 安装**

操作方法如下:

1) 首先，找到 DDM 所在的目录，使用 root 用户或安装用户登录到 Linux 系统，启动终端，执行以下命令即可进入 DDM 程序安装的目录:

```
#注：假设安装目录为/opt/dmncdb/ddmXXX
cd /opt/dmncdb/ddmXXX/bin
```

2) 再将 ddm_license.key 文件拷贝到该目录下，继续安装部署即可。

- **已部署 DDM，license 替换**

操作方法如下:

1) 首先，找到 DDM 所在的目录，使用 root 用户或安装用户登录到 Linux 系统，启动终端，执行以下命令即可进入 DDM 程序安装的目录:

```
#注：假设安装目录为/opt/dmncdb/ddmXXX
cd /opt/dmncdb/ddmXXX/bin
```

2) 关闭 ddm_engine 服务

3) 再将 ddm_license.key 文件拷贝到该目录下，替换原有 ddm_license.key

4) 重启 ddm_engine 服务。

5) 若部署集群多 engine 模式，则需要对每个 engine 节点的执行以上操作，即替换每个 engine 节点的 license 再重启 ddm_engine 服务。

说明：许可证名称为唯一标识请勿修改，可将该目录下原有的 ddm_license.key 文件事先做好备份。

第四章 安装与卸载

根据版本的用途，安装 DDM 程序后，默认装有一个许可证(License)。如果用户想拥有更多授权的许可证，请向达梦公司申请或购买，许可证(License)的安装方法具体参见许可证(License)的安装。

4.1. 安装前准备工作

用户在安装 DDM 之前需要检查或修改操作系统的配置，以保证 DDM 正确安装和运行。

以下安装程序说明将以 CentOS7 系统为例，由于不同操作系统系统命令不尽相同，具体步骤及操作请以本机系统为准，具体细节可向系统管理员咨询。

4.1.1. 检查 Linux(Unix)系统信息

用户在安装 DDM 前，需要检查当前操作系统的相关信息，确认 DDM 安装程序与当前操作系统匹配，以保证 DDM 能够正确安装和运行。用户可以使用以下命令检查操作系统基本信息。如下图所示：

```
#获取系统位数
getconf LONG_BIT
#查询操作系统 release 信息
lsb_release -a
#查询系统信息
cat /etc/issue
#查询系统名称
uname -a
```

```
[root@test22 ~]# getconf LONG_BIT
64
[root@test22 ~]# lsb_release -a
LSB Version:    :core-4.1-amd64:core-4.1-noarch:cxx-4.1-amd64:cxx-4.1-noarch:desktop-4.1-amd64:desktop-4.1-noarch:languages-4.
.1-noarch
Distributor ID: CentOS
Description:    CentOS Linux release 7.9.2009 (Core)
Release:        7.9.2009
Codename:       Core
[root@test22 ~]# cat /etc/issue
\S
Kernel \r on an \m

[root@test22 ~]# uname -a
Linux test22 5.19.8-1.el7.elrepo.x86_64 #1 SMP PREEMPT_DYNAMIC Tue Sep 6 15:12:14 EDT 2022 x86_64 x86_64 x86_64 GNU/Linux
[root@test22 ~]#
```

图 4.1 检查 Linux(Unix)系统信息

4.1.2. Linux(Unix)下检查操作系统限制

在 Linux(Unix)系统中，因为 ulimit 命令的存在，会对程序使用操作系统资源进行限制。为了使 DDM 能够正常运行，建议用户检查当前安装用户的 ulimit 参数。

运行 ulimit -a 进行查询。如下图所示：

```
[root@test22 ~]# ulimit -aa
core file size          (blocks, -c) unlimited
data seg size           (kbytes, -d) unlimited
scheduling priority             (-e) 0
file size               (blocks, -f) unlimited
pending signals                 (-i) 1542238
max locked memory       (kbytes, -l) 8192
max memory size         (kbytes, -m) unlimited
open files                      (-n) 65535
pipe size            (512 bytes, -p) 8
POSIX message queues     (bytes, -q) 819200
real-time priority              (-r) 0
stack size              (kbytes, -s) 8192
cpu time               (seconds, -t) unlimited
max user processes              (-u) 1542238
virtual memory          (kbytes, -v) unlimited
file locks                      (-x) unlimited
[root@test22 ~]# a
```

图 4.2 查看 ulimit 参数

参数使用限制：

1.data seg size

data seg size (kbytes, -d)

建议用户设置为 1048576(即 1GB)以上或 unlimited(无限制)，此参数过小将导致数据库启动失败。

2. file size

file size(blocks, -f)

建议用户设置为 unlimited(无限制)，此参数过小将导致数据库安装或初始化失败。

3. open files

open files(-n)

建议用户设置为 65536 以上或 unlimited(无限制)。

4.virtual memory

virtual memory (kbytes, -v)

建议用户设置为 1048576(即 1GB)以上或 unlimited(无限制)，此参数过小将导致数据库启动失败。

如果用户需要为当前安装用户更改 ulimit 的资源限制，请修改文件/etc/security/limits.conf。

4.1.3. 检查系统内存与存储空间

1.检查内存

为了保证 DDM 的正确安装和运行，需要用户保证至少有 50M 内存空闲。用户可以使用以下命令检查操作内存:

```
#获取内存总大小
grep MemTotal /proc/meminfo
#获取交换分区大小
grep SwapTotal /proc/meminfo
#获取内存使用详情
free
```

```
[root@test22 ~]# grep MemTotal /proc/meminfo
MemTotal:       394852732 kB
[root@test22 ~]# grep SwapTotal /proc/meminfo
SwapTotal:             0 kB
[root@test22 ~]# free
              total        used        free      shared  buff/cache   available
Mem:      394852732    98399264   193739700      240572   102713768   293580580
Swap:             0           0           0
[root@test22 ~]#
```

图 4.3 检查内存

2.检查存储空间

DDM 实例如果不开启持久化，对硬盘无要求。如果需要将数据持久化进磁盘，用户在 DDM 安装前，要保证 1G 可用空间，也应该为数据库数据预留足够的存储空间，规划好数据路径。用户可使用以下命令检查存储空间:

```
#查询目录/mount_point/dir_name 可用空间
df -h /mount_point/dir_name
```

4.1.4. 安装用户准备

如使用非 root 用户安装，创建部署用户 dmncdb 并配置 sudo 免密

```
useradd dmncdb
passwd dmncdb
sed -i "/^#.*NOPASSWD: ALL$/s/^#//" /etc/sudoers
usermod -a -G wheel dmncdb
```

给安装包赋用户组权限（部署的用户）

```
chown -R dmncdb:wheel ./安装包
```

4.1.5. 检查端口号是否被占用

检查分配节点的端口号是否被占用

```
netstat -anp | grep  端口号
```

4.1.6. 防火墙检查

检查防火墙是否关闭

```
systemctl stop firewalld
systemctl status firewalld
```

```
[root@1a16d1b76cd38473ab21107a376fde46 install_info]# systemctl status firewalld
● firewalld.service - firewalld - dynamic firewall daemon
   Loaded: loaded (/usr/lib/systemd/system/firewalld.service; disabled; vendor preset: enabled)
   Active: inactive (dead)
     Docs: man:firewalld(1)
[root@1a16d1b76cd38473ab21107a376fde46 install_info]#
```

4.1.7. 检查部署设备间的网络状态

首先查看设备 IP

```
ifconfig
```

测试多台设备网络的连通性

```
ping  目标 IP 地址或域名
```

4.2. 解压

4.2.1. rpm 包解压

ddm 的安装包分为单机安装 rpm 包和集群安装 rpm 包。单机安装包以 ddm_sa 为前缀，集群安装包以 ddm_cluster 为前缀。用户可以指定目录或者默认/opt 方式解压、安装部署。本节详细介绍 rpm 包解压方式。

- 默认目录安装部署，安装目录在/opt 目录下

```
rpm -ivh ddm_XXX.rpm
```

- 指定目录进行安装部署，提供两种方式

方式一：

```
rpm -ivh --prefix=/path ddm_XXX.rpm
```

方式二：

```
rpm -ivh --relocate /opt/=/path ddm_XXX.rpm
```

注意：已部署安装包后，若需要重新指定目录进行部署，需要先卸载安装包

```
rpm -e ddm_XXX
```

4.2.2. deb 包解压

ddm 的安装包分为单机安装 deb 包和集群安装 deb 包。用户可以指定目录或者默认/opt 方式解压、安装部署。本节详细介绍 deb 包解压方式。

- 默认目录安装部署，安装目录在/opt 目录下

```
dpkg -i ddm_XXX.deb
```

- 指定目录进行安装部署

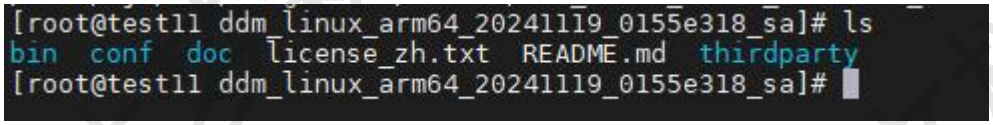
```
dpkg -x ddm_XXX.deb path
```

注意：已部署安装包后，若需要重新指定目录进行部署，需要先卸载安装包

```
dpkg -r ddm
```

4.3. 安装目录介绍

ddm 单机和集群安装包解压后，单机的安装目录为./dmncdb/ddm_sa-Vxxx/，集群的安装目录为./dmncdb/ddm_cluster-Vxxx/。里面的目录架构为 bin、thirdparty、doc、conf 文件夹、license_zh.txt 文件以及 README.md 文件。



（1）默认目录安装路径：

```
#单机命令
cd /opt/dmncdb/ddm_sa-Vxxx
#集群命令
cd /opt/dmncdb/ddm_cluster-Vxxx
```

（2）指定目录安装路径：

```
#单机命令
cd /path/dmncdb/ddm_sa-Vxxx
#集群命令
cd /path/dmncdb/ddm_cluster-Vxxx
```

- bin 文件夹：该文件夹中是所有组件服务的可执行文件，和安装部署脚本。其中可执行文件有 ddm_engine, ddm_meta, ddm_store、ddm_br。其中 ddm_br 是备份还原工具。

（1）集群：ddm_cluster_install.sh、ddm_cluster_uninstall.sh 是提供集群环境的一键部署、卸载脚本。

```
[root@jenkins bin]# ls
ddm_br  ddm_cluster_install.sh  ddm_cluster_uninstall.sh  ddm_engine  ddm_meta  ddm_store  scripts
[root@jenkins bin]#
```

（2）单机：ddm_install.sh、ddm_uninstall.sh 是提供单机环境的一键部署、卸载脚本。

```
[root@jenkins bin]# ls
ddm_br  ddm_engine  ddm_install.sh  ddm_meta  ddm_store  ddm_uninstall.sh  scripts
[root@jenkins bin]#
```

- conf 文件夹：该文件夹为配置文件。

（1）集群：包含 2 个文件，一个是 ddm_engine 的配置文件，hosts 是一键部署脚本用到的集群规划配置文件，包含部署节点 IP 和 port，数据文件地址配置。

```
[root@jenkins conf]# ls
ddm_engine_config.toml  hosts
[root@jenkins conf]#
```

（2）单机：只有 1 个 ddm_engine 的配置文件

```
[root@jenkins conf]# ls
ddm_engine_config.toml
[root@jenkins conf]#
```

- doc 文件夹：doc 包含两个 pdf 文件，分别为达梦新云文档数据库安装手册 v1.pdf 和达梦新云文档数据库用户手册 v1.pdf，帮助用户快速了解 DDM。

```
[root@1-14 doc]# ls
达梦新云文档数据库安装手册v1.pdf  达梦新云文档数据库用户手册v1.pdf
[root@1-14 doc]#
```

- thirdparty 文件夹：license 文件夹包含第三方客户端工具以及所有 DDM 服务器与接口使用的第三方代码或插件的 license。

```
[root@1-14 thirdparty]# tree
.
├── license
│   ├── APACHE_LICENSE-2.0
│   ├── br LICENSE.txt
│   ├── BSD-3-Clause_LICENSE
│   ├── crypto_Go_Authors_LICENSE
│   ├── MIT&APACHE_LICENSE
│   ├── MIT&APACHE_LICENSE-2.0
│   ├── MIT_LICENSE
│   ├── MIT_LICENSE-2014
│   ├── pd LICENSE.txt
│   └── tikv LICENSE.txt
└── tools
    └── mongosh
```

- hosts 模板信息如下：

```
# 根据 DDM 集群的部署规划信息，配置各服务节点 IP、PORT 以及对应服务器系统中用户、密码和 ssh 端口号。
```

```
    # DDM 集群模式分为两种：主备和分布式
    # DDM 主备集群模式一般为：双 store 部署 3-2-2 集群(3 个 meta 节点、2 个 store 节点、2 个 engine 节点)
    # DDM 分布式集群模式一般为：部署 3-3-1 集群（3 个 meta 节点、3 个 store 节点、1 个 engine 节点）

    # DDM_Meta 部署节点信息
    #
    # 节点个数不限（建议部署奇数个节点），每添加一个节点的部署，都需要增加一条 "Meta" 记录，其中：
    # (1) META_IP 表示运行 DDM_Meta 服务的机器 ip 地址；
    # (2) META_PORT 表示运行 DDM_Meta 服务的端口号，端口号建议使用 2379；
    # (3) PEER_META_PORT 请使用不同于 META_PORT 的端口号；
    # (4) USER 表示运行 DDM_Meta 服务的机器系统用户，非 root 用户需要部署前配置好该用户的 sudo 免密权限；
    # (5) PWD 表示运行 DDM_Meta 服务的机器系统用户的密码；
    # (6) SSH_PORT 表示运行设备系统的 ssh 端口。
    #
    # 例如，部署包括三个 DDM_Meta 节点的集群：
    # Meta: "ip=192.168.1.10", "port=2379", "peer_port=2380", "user=root", "password=123456", "sshPort=22"
    # Meta: "ip=192.168.1.20", "port=2379", "peer_port=2380", "user=root", "password=123456", "sshPort=22"
    # Meta: "ip=192.168.1.30", "port=2379", "peer_port=2380", "user=root", "password=123456", "sshPort=22"
    #
    [DDM_Meta]
    Meta: "ip=META_IP", "port=META_PORT", "peer_port=PEER_META_PORT", "user=USER", "password=PWD", "sshPort=SSH_PORT"

    # DDM_Store 部署节点信息
    #
    # 节点个数不限（分布式集群建议部署奇数个节点），每添加一个节点的部署，都需要增加一条 "Store" 记录，其中：
    # (1) STORE_IP 表示运行 DDM_Store 服务的机器 ip 地址；
    # (2) STORE_PORT 表示运行 DDM_Store 服务的端口号，端口号建议使用 10180；
    # (3) USER 表示运行 DDM_Store 服务的机器系统用户，非 root 用户需要部署前配
```

```
置好该用户的 sudo 免密权限;
    # (4) PWD 表示运行 DDM_Store 服务的机器系统用户的密码;
    # (5) SSH_PORT 表示运行设备系统的 ssh 端口。
    #
    # 例如，部署包括三个 DDM_Store 节点的集群:
    # Store: "ip=192.168.1.20", "port=10180", "user=root", "password=123456",
"sshPort=22"
    # Store: "ip=192.168.1.30", "port=10180", "user=root", "password=123456",
"sshPort=22"
    # Store: "ip=192.168.1.40", "port=10180", "user=root", "password=123456",
"sshPort=22"
    #
    [DDM_Store]
    Store: "ip=STORE_IP", "port=STORE_PORT", "user=USER", "password=PWD",
"sshPort=SSH_PORT"

    # DDM_Engine 部署节点信息
    #
    # 节点个数不限，每添加一个节点的部署，都需要增加一条 "Engine" 记录，其中:
    # (1) ENGINE_IP 表示运行 DDM_Engine 服务的机器 ip 地址;
    # (2) ENGINE_PORT 表示运行 DDM_Engine 服务的端口号，端口号建议使用 27017;
    # (3) USER 表示运行 DDM_Engine 服务的机器系统用户，非 root 用户需要部署前配
置好该用户的 sudo 免密权限;
    # (4) PWD 表示运行 DDM_Engine 服务的机器系统用户的密码;
    # (5) SSH_PORT 表示运行设备系统的 ssh 端口。
    #
    # 例如：部署三个 DDM_Engine 节点:
    # Engine: "ip=192.168.1.30", "port=27017", "user=root", "password=123456",
"sshPort=22"
    # Engine: "ip=192.168.1.40", "port=27017", "user=root", "password=123456",
"sshPort=22"
    # Engine: "ip=192.168.1.50", "port=27017", "user=root", "password=123456",
"sshPort=22"
    #
    [DDM_Engine]
    Engine: "ip=ENGINE_IP", "port=ENGINE_PORT", "user=USER", "password=PWD",
"sshPort=SSH_PORT"
```

```
# 数据文件目录
#
# 请确保数据文件存放的路径有足够大的磁盘空间
data_dir="/data/ddm_cluster"
```

README 文件简要介绍了组件功能，快速部署方式，帮助用户快速部署 DDM。

4.4. 安装

以下为手动安装部署脚本传人参数的具体解释

参数	解释
-mi	指定 ddm_meta 服务的机器 IP 地址
-mp	指定 ddm_meta 服务的机器端口号
-pmp	指定 ddm_meta 元数据管理服务的 PEER_META_PORT
-ic	设置集群是否初始化,值为 true 或 false，默认 fasle
-jc	设置是否加入集群,值为 true 或 false，默认 fasle
-jcn	指定 ddm_meta 加入集群的节点名称
-jci	指定 ddm_meta 加入集群的 IP
-jcp	指定 ddm_meta 加入集群的 PORT
-si	指定 ddm_store 数据库存储服务的机器 IP 地址
-sp	指定 ddm_store 数据库存储服务的机器端口号
-pd	设置 ddm_store 连接 ddm_meta 集群地址池
-ei	指定 ddm_engine 数据库执行引擎服务 IP 地址
-ep	指定 ddm_engine 数据库执行引擎服务端口号
-data	指定数据文件存放路径

4.4.1. 单机安装

rpm 解压完成，用户可以按组件介绍章节的启动顺序，一次启动服务，来完成 ddm 服务的搭建；也可以使用安装包提供的对应的注册系统服务的脚本，并设置其开机自启功能来进行一键搭建。

- 节点规划

单机版本可在单节点上部署 3 个组件，也支持在不同节点上分别部署组件，本文以单节点为例，以下为默认端口号配置，可根据实际自定义配置，保证端口号无冲突：

节点	地址
ddm_meta	10.14.1.160:2379
ddm_store	10.14.1.160:10180
ddm_engine	10.14.1.160:27017

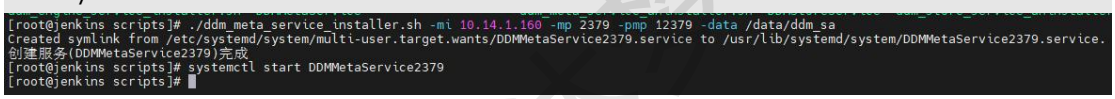
4.4.1.1. 手动安装

单机版本根据解压指引解压安装包后，进入 dmncdb/ddm_XXX/bin/scripts 目录下，依次按照 ddm_meta、ddm_store、ddm_engine 顺序启动服务，即完成了单机版本 ddm 对服务的搭建。以下脚本传入的参数解释参见安装。

注意：以下命令如果在非 root 用户下执行，需要加 sudo 或者切换到 root 用户下执行

1） 根据节点规划给脚本传人参数进行注册 mete 组件服务，注册后手动启动组件服务。

```
./ddm_meta_service_installer.sh -mi 10.14.1.160 -mp 2379 -pmp 12379 -data /data/ddm_sa
systemctl start DDMMetaService2379
```



2）根据节点规划给脚本传人参数进行注册 store 组件服务，注册后手动启动组件服务。

```
./ddm_store_service_installer.sh -mi 10.14.1.160 -mp 2379 -si 10.14.1.160 -sp 10180 -data /data/ddm_sa
systemctl start DDMStoreService10180
```

3）根据节点规划给脚本传人参数进行注册 engine 组件服务，注册后手动启动组件服务。

```
./ddm_engine_service_installer.sh -mi 10.14.1.160 -mp 2379 -ei 10.14.1.160 -ep 27017 -data /data/ddm_sa
systemctl start DDMEngineService27017
```

注意：以上 1）2）3）步骤执行完成后，即单机版环境搭建完成，如想进一步进行服务状态的管控，可参考步骤 4）,其中步骤 4）为非必选项。

4）服务管控脚本一键管控所有组件服务（切换 root 用户）

①进入 dmncdb/ddm_XXX/conf 目录下创建 install_info 文件夹

```
mkdir -p install_info
```

②在 install_info 文件夹下创建 hosts 文件

内容:

```
DDMMeta=10.14.1.160:2379
DDMStore=10.14.1.160:10180
DDMEngine=10.14.1.160:27017
DATA_DIR=/data/ddm_sa
```

③复制 dmncdb/ddm_XXX/bin/scripts 目录下的 ddm_service 至 /usr/local/bin 目录下，（为了方便查看服务的 IP 和端口号，方便管理，建议将 ddm_service 改名为 ddm_service_10.14.1.160_27017（engine 的 ip 和端口））。

④将 ddm_service 中的 DDM_HOME 的值修改为安装包目录 bin 的上一层目录，INSTALL_INFO_FILE 的值修改为 hosts 的对应路径。

⑤可以用 ddm_service_<ip_port> start/stop/restart/status/remove 进行服务管控。

```
[root@jenkins bin]# ddm_service_10.14.1.160_27017 start
[10.14.1.160          DDMMetaService2379           ]          [Starting]
[10.14.1.160          DDMStoreService10180         ]          [Starting]
[10.14.1.160          DDMEngineService27017        ]          [Starting]
[root@jenkins bin]# ddm_service_10.14.1.160_27017 stop
[10.14.1.160          DDMEngineService27017        ]          [Stopping]
[10.14.1.160          DDMStoreService10180         ]          [Stopping]
[10.14.1.160          DDMMetaService2379           ]          [Stopping]
[root@jenkins bin]# ddm_service_10.14.1.160_27017 restart
[10.14.1.160          DDMEngineService27017        ]          [Stopped]
[10.14.1.160          DDMStoreService10180         ]          [Stopped]
[10.14.1.160          DDMMetaService2379           ]          [Stopped]

[10.14.1.160          DDMMetaService2379           ]          [Starting]
[10.14.1.160          DDMStoreService10180         ]          [Starting]
[10.14.1.160          DDMEngineService27017        ]          [Starting]
[root@jenkins bin]# ddm_service_10.14.1.160_27017 status
[10.14.1.160          DDMMetaService2379           ]          [Running]
[10.14.1.160          DDMStoreService10180         ]          [Running]
[10.14.1.160          DDMEngineService27017        ]          [Running]
[root@jenkins bin]#
```

4.4.1.2. 一键安装

进入安装目录/dmncdb/ddm_sa-Vxxx/bin 下，执行一键部署脚本

```
./ddm_install.sh
```

1) 出现 Press [Enter] key to continue [Enter]，按回车键继续

```
+-------------------------------------------------------------------------------------------------------------------------+
| [声明]:                                                                                                                 |
|     1、达梦新云文档数据库，简称DDM，是一款文档型数据库，目前由以下三个组件组成：                                                                    |
|     > (1) DDM_Engine：数据库执行引擎服务，负责接收和处理客户端请求；                                                                      |
|     > (2) DDM_Meta：数据库元数据管理服务，存储和管理数据库的元数据；                                                                      |
|     > (3) DDM_Store：数据库存储服务，提供了数据的持久化和读写服务。                                                                        |
|     2、该安装包是实现在一台设备上部署DDM，请根据场景和需求事先规划好：DDM_Meta、DDM_Store、DDM_Engine各节点IP、port。                                |
|     3、规划好相关信息根据提示进行输入，非本机需根据提示输入对应设备密码，根据提示输入信息时请确保输入的信息准确性和可用性，                                                  |
|        信息错误可能会导致安装部署终止。                                                                                                |
|     4、该安装包提供两种部署模式：各服务均部署在一台设备（AIO）、各服务部署在不同设备（ADD）。部署时请根据提示选择。                                                    |
|     5、请耐心等待，直至各服务出现“running”状态即可。                                                                                      |
|     6、安装过程中如需帮助请联系DMNCDB相关技术支持人员。感谢对DMNCDB团队的支持，谢谢！                                                                  |
+-------------------------------------------------------------------------------------------------------------------------+
Press [Enter] key to continue [Enter]
```

2) 根据提示输入 aio(在一台设备部署)或者 add（不同设备部署），本文默认选择 aio 模式,按回车键继续

```
+-------------------------------------------------------------------------------------------------------------------------+
|  请选择部署模式：各服务均部署在一台设备（AIO）,请输入AIO、各服务部署在不同设备（ADD）,请输入ADD。请正确输入！                                                    |
+-------------------------------------------------------------------------------------------------------------------------+
Please Input [AIO(aio)/ADD(add)]:
```

3) 输入部署 DDM 的 IP 后，按回车键继续

```
Please Input [AIO(aio)/ADD(add)]: aio

 > 根据提示输入规划的DDM相关基本信息 .

请输入部署DDM的设备IP:10.14.1.160
```

4) 使用默认端口号部署，输入 y，按回车键继续

```
是否全部使用默认端口进行部署?(Y/y:是 N/n:否): y
 默认数据文件存放的路径为：/opt/dmncdb/ddm_sa-V1.0-20240726_82f3217b，自定义数据文件存放的路径请确保有足够大的磁盘空间
```

5) 使用默认数据文件路径部署，输入 y，按回车键继续

```
默认数据文件存放的路径为：/opt/dmncdb/ddm_sa-V1.0-20240726_82f3217b，自定义数据文件存放的路径请确保有足够大的磁盘空间
是否使用默认数据文件存放的路径?(Y/y:是 N/n:否): y
```

6) 等待服务状态均为 Running 时，部署完成

```
 > 部署完成，查看部署的服务状态 .
+ /bin/bash /usr/local/bin/ddm_service_10.14.1.160_27017 status
[10.14.1.160          DDMMetaService2379          ]          [Running]
[10.14.1.160          DDMStoreService10180        ]          [Running]
[10.14.1.160          DDMEngineService27017       ]          [Running]


 >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>> Install Complete <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
[root@jenkins bin]#
```

- 服务状态查看

  使用以下命令查看单机数据库服务的状态:

  ddm_service_<*ip_port*> status

```
[root@jenkins bin]# ddm_service_10.14.1.160_27017 status
[10.14.1.160          DDMMetaService2379          ]          [Running]
[10.14.1.160          DDMStoreService10180        ]          [Running]
[10.14.1.160          DDMEngineService27017       ]          [Running]
[root@jenkins bin]#
```

以上所有服务都可以用 systemctl 对每个节点服务进行 start, restart, stop 操作。

systemctl stop DDMEngineServiceXXX

```
[root@jenkins bin]# systemctl stop DDMEngineService27017
[root@jenkins bin]# ddm_service_10.14.1.160_27017 status
[10.14.1.160          DDMMetaService2379          ]          [Running]
[10.14.1.160          DDMStoreService10180        ]          [Running]
[10.14.1.160          DDMEngineService27017       ]          [Failed]
[root@jenkins bin]#
```

如果是对单机数据库服务统一操作，则可以用 ddm_service_<*ip_port*> start/stop/restart 进行统一操作。

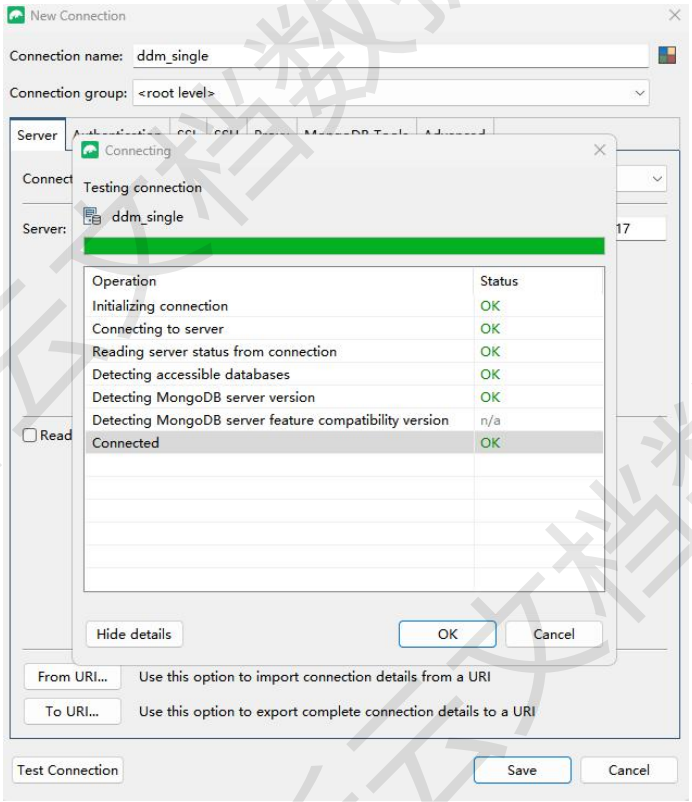
```
[root@jenkins bin]# ddm_service_10.14.1.160_27017 restart
[10.14.1.160          DDMEngineService27017       ]          [Stopped]
[10.14.1.160          DDMStoreService10180        ]          [Stopping]
[10.14.1.160          DDMMetaService2379          ]          [Stopping]

[10.14.1.160          DDMMetaService2379          ]          [Starting]
[10.14.1.160          DDMStoreService10180        ]          [Starting]
[10.14.1.160          DDMEngineService27017       ]          [Starting]
[root@jenkins bin]# ddm_service_10.14.1.160_27017 status
[10.14.1.160          DDMMetaService2379          ]          [Running]
[10.14.1.160          DDMStoreService10180        ]          [Running]
[10.14.1.160          DDMEngineService27017       ]          [Running]
```

4.4.1.3. 连接示例

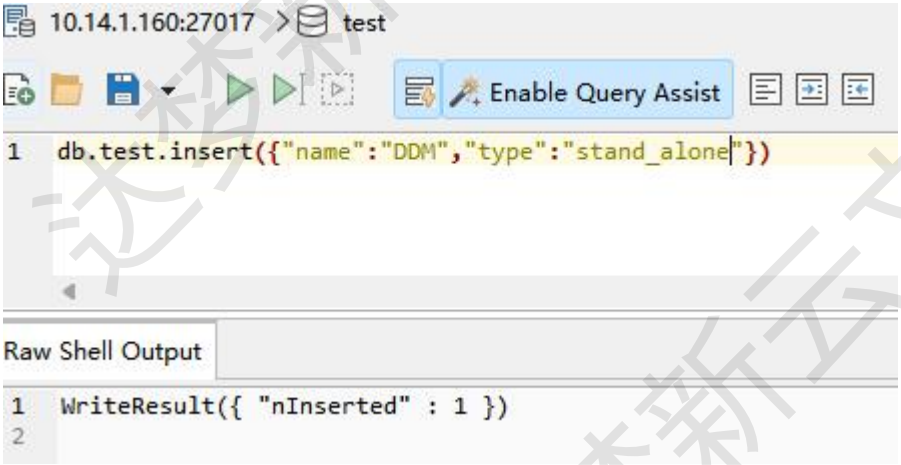
- Studio 3T

  使用 3T 工具，连接 ddm_engine，并做数据插入查询操作。

1） 测试连接



2） 连接后添加文档



3） 查询文档

4） 至此 ddm 单机搭建并验证完成

- Mongosh

使用 mongosh 工具，连接 ddm_engine，并做数据插入查询操作。

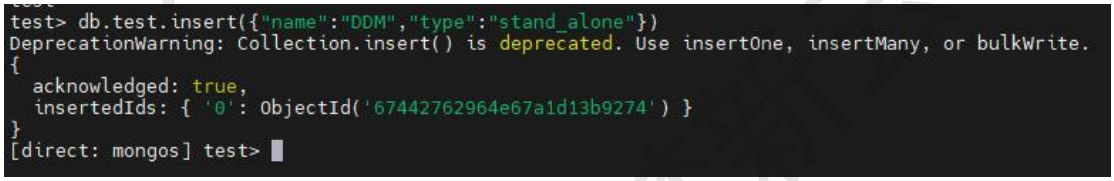
1） 连接测试

进入安装目录的 dmncdb/ddm_XXX/thirdparty/tools 目录下执行以下命令

```
./mongosh --host 10.14.1.160 --port 27017
```

2） 连接后添加文档

```
db.test.insert({"name":"DDM","type":"stand_alone"})
```



3） 查询文档

```
db.test.find({"name":"DDM"}).toArray()
```

```
[direct: mongos] test> db.test.find({"name":"DDM"}).toArray()
[
  {
    _id: ObjectId('67442762964e67a1d13b9274'),
    name: 'DDM',
    type: 'stand_alone'
  }
]
[direct: mongos] test>
```

4） 至此 ddm 单机搭建并验证完成

4.4.2. 集群安装

4.4.2.1. 分布式集群安装

- 集群节点规划

ddm_store 节点作为分布式集群数据存储核心节点，建议尽量单独部署在不同的节点上，ddm_engine 和 ddm_meta 可以共享主机。

本次将以在 x86_centos 服务器上部署 3-3-1 的分布式集群为例，演示部署。由于只有 3 台服务器，示例规划存在 ddm_engine，ddm_meta，ddm_store 规划在同一服务器上，示例节点规划如下：

节点	地址
ddm_meta1	10.14.135.35:2379
ddm_meta2	10.14.135.36.:2379
ddm_meta3	10.14.135.37:2379
ddm_store1	10.14.135.35:10180
ddm_store2	10.14.135.36:10180
ddm_store3	10.14.135.37:10180
ddm_engine1	10.14.135.36:27018

4.4.2.1.1. 手动安装

分布式集群版本将安装包分别上传到三台部署的机器上，根据解压指引解压到相同的路径下，进入 dmncdb/ddm_XXX/bin/scripts 目录下，依次按照 ddm_meta、ddm_store、ddm_engine 顺序启动服务，即完成了分布式集群版本 ddm 对服务的搭建。以下脚本传入的参数解释参见安装。

注意:如使用非 root 用户部署，三台部署机器需要分别创建相同的用户并给安装包赋用户组权限，以下命令如果在非 root 用户下执行，需要加 sudo 或者切换到 root 用户下执行。

1）根据节点规划给脚本传人参数进行注册 mete 组件服务，注册后手动启动组件服务。

①在 10.14.135.35 机器上部署 mete1 节点

```
./ddm_meta_service_installer.sh -mi 10.14.135.35 -mp 2379 -pmp 2380 -data /data/ddm_cluster -ic true
systemctl start DDMMetaService2379
```

```
[root@1a16d1b76cd38473ab21107a376fde46 scripts]# ./ddm_meta_service_installer.sh -mi 10.14.135.35 -mp 2379 -pmp 2380 -data /data/ddm_cluster -ic true
Created symlink /etc/systemd/system/multi-user.target.wants/DDMMetaService2379.service → /usr/lib/systemd/system/DDMMetaService2379.service.
创建服务(DDMMetaService2379)完成
[root@1a16d1b76cd38473ab21107a376fde46 scripts]# systemctl start DDMMetaService2379
[root@1a16d1b76cd38473ab21107a376fde46 scripts]#
```

②在 10.14.135.36 机器上部署 mete2 节点

```
./ddm_meta_service_installer.sh -mi 10.14.135.36 -mp 2379 -pmp 2380 -data /data/ddm_cluster -jc true -jcn Meta2 -jci 10.14.135.35 -jcp 2379
systemctl start DDMMetaService2379
```

```
[root@c0b087a753da9429454010dd89e4a64e scripts]# ./ddm_meta_service_installer.sh -mi 10.14.135.36 -mp 2379 -pmp 2380 -data /data/ddm_cluster
Created symlink /etc/systemd/system/multi-user.target.wants/DDMMetaService2379.service → /usr/lib/systemd/system/DDMMetaService2379.service.
创建服务(DDMMetaService2379)完成
[root@c0b087a753da9429454010dd89e4a64e scripts]# systemctl start DDMMetaService2379
[root@c0b087a753da9429454010dd89e4a64e scripts]#
```

③在 10.14.135.37 机器上部署 mete3 节点

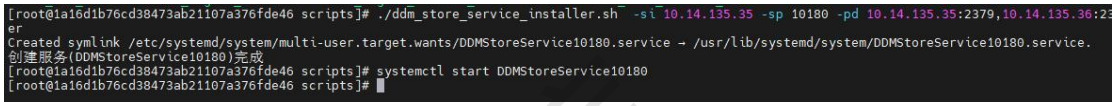
```
./ddm_meta_service_installer.sh -mi 10.14.135.37 -mp 2379 -pmp 2380 -data /data/ddm_cluster -jc true -jcn Meta3 -jci 10.14.135.35 -jcp 2379
systemctl start DDMMetaService2379
```

```
[root@d6558399e0e5ef03ed4a9c9150b6dc29 scripts]# ./ddm_meta_service_installer.sh -mi 10.14.135.37 -mp 2379 -pmp 2380 -data /data/ddm_cluster -
Created symlink /etc/systemd/system/multi-user.target.wants/DDMMetaService2379.service → /usr/lib/systemd/system/DDMMetaService2379.service.
创建服务(DDMMetaService2379)完成
[root@d6558399e0e5ef03ed4a9c9150b6dc29 scripts]# systemctl start DDMMetaService2379
[root@d6558399e0e5ef03ed4a9c9150b6dc29 scripts]#
```

2）根据节点规划给脚本传人参数进行注册 store 组件服务，注册后手动启动组件服务。

①在 10.14.135.35 机器上部署 store1 节点

```
./ddm_store_service_installer.sh  -si 10.14.135.35 -sp 10180 -pd 10.14.135.35:2379,10.14.135.36:2379,10.14.135.37:2379 -data /data/ddm_cluster
systemctl start DDMStoreService10180
```



②在 10.14.135.36 机器上部署 store2 节点

./ddm_store_service_installer.sh -si 10.14.135.36 -sp 10180 -pd 10.14.135.35:2379,10.14.135.36:2379,10.14.135.37:2379 -data /data/ddm_cluster

systemctl start DDMStoreService10180

③在 10.14.135.37 机器上部署 store3 节点

./ddm_store_service_installer.sh -si 10.14.135.37 -sp 10180 -pd 10.14.135.35:2379,10.14.135.36:2379,10.14.135.37:2379 -data /data/ddm_cluster

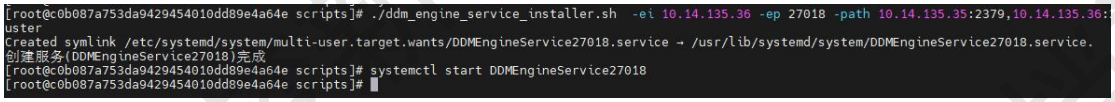
systemctl start DDMStoreService10180

3）根据节点规划给脚本传人参数进行注册 engine 组件服务，注册后手动启动组件服务。

①在 10.14.135.36 机器上部署 engine1 节点

./ddm_engine_service_installer.sh -ei 10.14.135.36 -ep 27018 -path 10.14.135.35:2379,10.14.135.36:2379,10.14.135.37:2379 -data /data/ddm_cluster

systemctl start DDMEngineService27018



注意：以上 1）2）3）步骤执行完成后，即分布式集群搭建完成，如想进一步进行服务状态的管控，可参考步骤 4）,其中步骤 4）为非必选项。

4）服务管控脚本一键管控所有节点服务（切换 root 用户）

切换到部署 engine 的第一个节点的机器上（10.14.135.36）

①进入 dmncdb/ddm_cluster-XXX/conf 目录下创建 install_info 文件夹

mkdir -p install_info

② 将 dmncdb/ddm_cluster_XXX/conf 目录下的 hosts 文件复制至 install_info 文件夹下

③在 hosts 中填写节点相关信息（将步骤 1）、2）、3）部署的相关 IP 和端口号对应填写）

```
#
# 节点个数不限（建议部署奇数个节点），每添加一个节点的部署，都需要增加一条 "Meta" 记录，其中：
# (1) META_IP 表示运行 DDM_Meta 服务的机器 ip 地址；
# (2) META_PORT 表示运行 DDM_Meta 服务的端口号，端口号建议使用 2379；
# (3) PEER_META_PORT 请使用不同于 META_PORT 的端口号；
# (4) USER 表示运行 DDM_Meta 服务的机器系统用户，非 root 用户需要部署前配置好该用户的 sudo 免密权限；
# (5) PWD 表示运行 DDM_Meta 服务的机器系统用户的密码；
# (6) SSH_PORT 表示运行设备系统的 ssh 端口。
#
# 例如，部署包括三个 DDM_Meta 节点的集群：
# Meta: "ip=192.168.1.10", "port=2379", "peer_port=2380", "user=root", "password=123456", "sshPort=22"
# Meta: "ip=192.168.1.20", "port=2379", "peer_port=2380", "user=root", "password=123456", "sshPort=22"
# Meta: "ip=192.168.1.30", "port=2379", "peer_port=2380", "user=root", "password=123456", "sshPort=22"
#
[DDM_Meta]

Meta: "ip=10.14.135.35", "port=2379", "peer_port=2380", "user=root", "password=123456", "sshPort=22"
Meta: "ip=10.14.135.36", "port=2379", "peer_port=2380", "user=root", "password=123456", "sshPort=22"
Meta: "ip=10.14.135.37", "port=2379", "peer_port=2380", "user=root", "password=123456", "sshPort=22"

# DDM_Store 部署节点信息
# 节点个数不限（分布式集群建议部署奇数个节点），每添加一个节点的部署，都需要增加一条 "Store" 记录，其中：
# (1) STORE_IP 表示运行 DDM_Store 服务的机器 ip 地址；
# (2) STORE_PORT 表示运行 DDM_Store 服务的端口号，端口号建议使用 10180；
# (3) USER 表示运行 DDM_Store 服务的机器系统用户，非 root 用户需要部署前配置好该用户的 sudo 免密权限；
# (4) PWD 表示运行 DDM_Store 服务的机器系统用户的密码；
# (5) SSH_PORT 表示运行设备系统的 ssh 端口。
# 例如，部署包括三个 DDM_Store 节点的集群：
# Store: "ip=192.168.1.20", "port=10180", "user=root", "password=123456", "sshPort=22"
# Store: "ip=192.168.1.30", "port=10180", "user=root", "password=123456", "sshPort=22"
# Store: "ip=192.168.1.40", "port=10180", "user=root", "password=123456", "sshPort=22"
[DDM_Store]

Store: "ip=10.14.135.35", "port=10180", "user=root", "password=123456", "sshPort=22"
Store: "ip=10.14.135.36", "port=10180", "user=root", "password=123456", "sshPort=22"
Store: "ip=10.14.135.37", "port=10180", "user=root", "password=123456", "sshPort=22"

# DDM_Engine 部署节点信息
# 节点个数不限，每添加一个节点的部署，都需要增加一条 "Engine" 记录，其中：
# (1) ENGINE_IP 表示运行 DDM_Engine 服务的机器 ip 地址；
# (2) ENGINE_PORT 表示运行 DDM_Engine 服务的端口号，端口号建议使用 27017；
# (3) USER 表示运行 DDM_Engine 服务的机器系统用户，非 root 用户需要部署前配置好该用户的 sudo 免密权限；
# (4) PWD 表示运行 DDM_Engine 服务的机器系统用户的密码；
# (5) SSH_PORT 表示运行设备系统的 ssh 端口。
# 例如：部署三个 DDM_Engine 节点：
# Engine: "ip=192.168.1.30", "port=27017", "user=root", "password=123456", "sshPort=22"
# Engine: "ip=192.168.1.40", "port=27017", "user=root", "password=123456", "sshPort=22"
# Engine: "ip=192.168.1.50", "port=27017", "user=root", "password=123456", "sshPort=22"

[DDM_Engine]

Engine: "ip=10.14.135.36", "port=27018", "user=root", "password=123456", "sshPort=22"

# 数据文件目录
# 请确保数据文件存放的路径有足够大的磁盘空间
data_dir="/data/ddm_cluster"
```

④ 复制 dmncdb/ddm_cluster_XXX/bin/scripts 目录下 ddm_cluster_service 至/usr/local/bin 目录下，（为了方便查看服务的 IP 和端口号，方便管理，建议将 ddm_cluster_service 改名为 ddm_cluster_service_10.14.135.36_27018（engine 的 ip 和端口））

⑤修改 ddm_cluster_service 中，DDMDC_HOME 的值修改为安装包目录 bin 的上一层目录，DDMDC_INI_PATH 的值修改为 install_info 文件夹下 hosts 的实际路径。

⑥ 可以用 ddm_cluster_service_<ip_port> start/stop/restart/status/remove 进行服务管控。

```
[root@c0b087a753da9429454010dd89e4a64e bin]# ddm_cluster_service_10.14.135.36_27018 status
[10.14.135.35         DDMMetaService2379          ]          [Running]
[10.14.135.36         DDMMetaService2379          ]          [Running]
[10.14.135.37         DDMMetaService2379          ]          [Running]
[10.14.135.35         DDMStoreService10180        ]          [Running]
[10.14.135.36         DDMStoreService10180        ]          [Running]
[10.14.135.37         DDMStoreService10180        ]          [Running]
[10.14.135.36         DDMEngineService27018       ]          [Running]
[root@c0b087a753da9429454010dd89e4a64e bin]# ddm_cluster_service_10.14.135.36_27018 start
[10.14.135.35         DDMMetaService2379          ]          [Running]
[10.14.135.36         DDMMetaService2379          ]          [Running]
[10.14.135.37         DDMMetaService2379          ]          [Running]
[10.14.135.35         DDMStoreService10180        ]          [Running]
[10.14.135.36         DDMStoreService10180        ]          [Running]
[10.14.135.37         DDMStoreService10180        ]          [Running]
[10.14.135.36         DDMEngineService27018       ]          [Running]
```

4.4.2.1.2. 一键安装

在执行一键安装脚本前，先将规划好 3-3-1 分布式集群的地址填写进 conf 文件夹下的 hosts 文件中。hosts 文件 ddm_meta 部署节点信息配置如下：

```
[ddm_meta]
Meta: "ip=10.14.135.35", "port=2379", "peer_port=2380", "user=root", "password=***","sshPort=22"
Meta: "ip=10.14.135.36", "port=2379", "peer_port=2380", "user=root", "password=***","sshPort=22"
Meta: "ip=10.14.135.37", "port=2379", "peer_port=2380", "user=root", "password=***","sshPort=22"
```

ddm_store 部署节点信息配置如下：

```
[ddm_store]
Store: "ip=10.14.135.35", "port=10180", "user=root", "password=***","sshPort=22"
Store: "ip=10.14.135.36", "port=10180", "user=root", "password=***","sshPort=22"
Store: "ip=10.14.135.37", "port=10180", "user=root", "password=***","sshPort=22"
```

ddm_engine 部署节点信息配置如下：

```
[ddm_engine]
Engine: "ip=10.14.135.36", "port=27018", "user=root", "password=***","sshPort=22"
```

数据文件目录配置后，将当前服务器所有节点的数据文件将存储在这个路径下。Hosts 文件中数据文件目录配置如下：

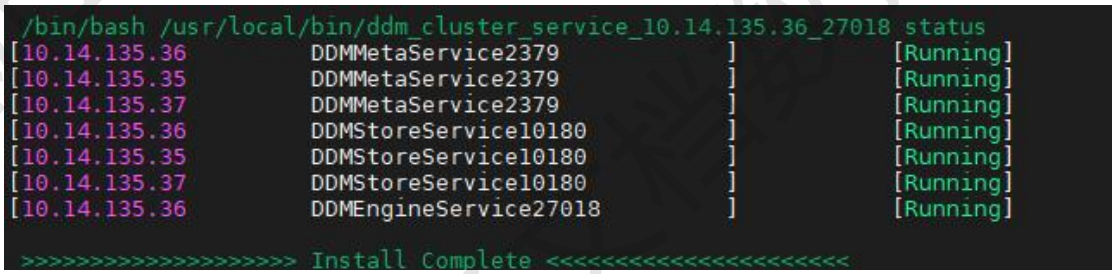
```
data_dir="/home/kaka/ddm_cluster"
```

在 bin 目录下执行 ddm_cluster_install.sh,脚本将依照 hosts 文件中规划配置，依次部署启动元数据节点服务、存储数据节点服务和执行节点服务。并为各组件服务自动注册系统服务，设置该服务为开启自启动模式。

出现 Press [Enter] key to continue [Enter]时，点击回车键，进行下一步，会自

动开始部署。

一键部署脚本成功执行后会出现以下界面，各服务状态均为 Running,打印 Install Complete。



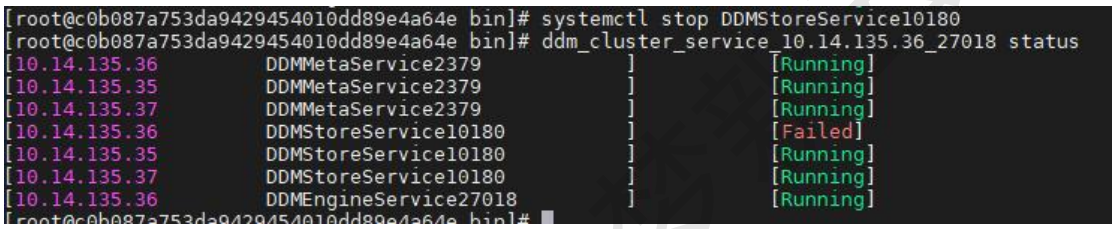
- 服务状态查看

  使用以下脚本查看集群所有节点的状态:

  ddm_cluster_service_<*ip_port*> status

以上所有服务都可以用 systemctl 对每个节点服务进行 start, restart, stop 操作。

systemctl stop DDMStoreServiceXXX



如果是对 ddm_cluster_service 统一操作，则可以用 ddm_cluster_service_<*ip_port*> start/stop/restart 进行统一操作。

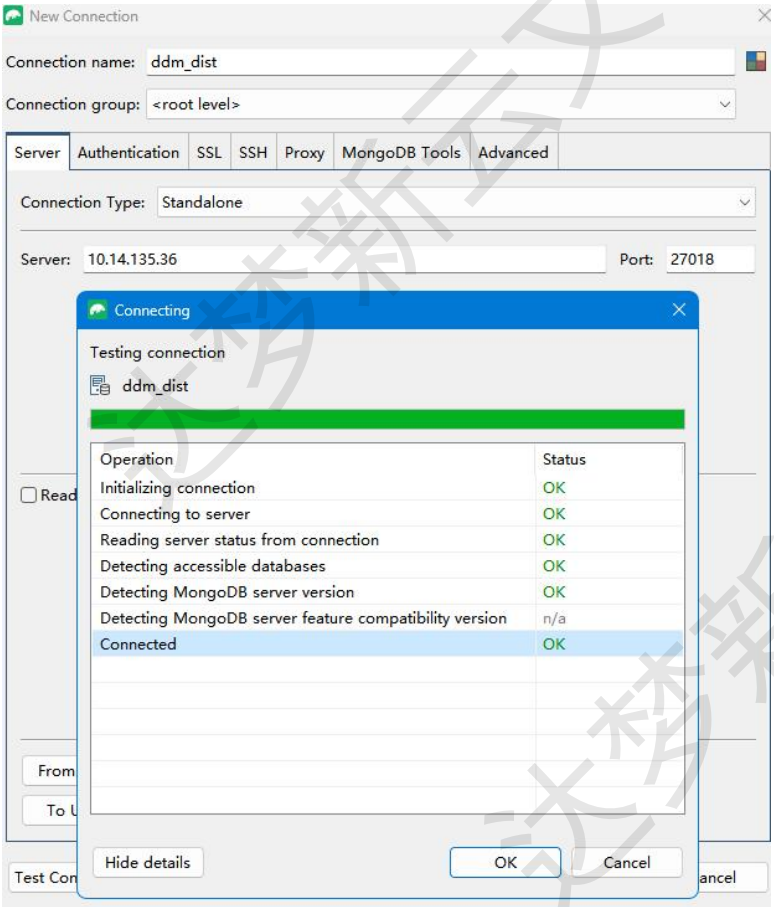
```
[root@c0b087a753da9429454010dd89e4a64e bin]# ddm_cluster_service_10.14.135.36_27018 restart
[10.14.135.36         DDMEngineService27018         ]         [Stopping]
[10.14.135.37         DDMStoreService10180          ]         [Stopping]
[10.14.135.35         DDMStoreService10180          ]         [Stopping]
[10.14.135.36         DDMStoreService10180          ]         [Stopped]
[10.14.135.37         DDMMetaService2379            ]         [Stopping]
[10.14.135.35         DDMMetaService2379            ]         [Stopping]
[10.14.135.36         DDMMetaService2379            ]         [Stopping]

[10.14.135.36         DDMMetaService2379            ]         [Starting]
[10.14.135.35         DDMMetaService2379            ]         [Starting]
[10.14.135.37         DDMMetaService2379            ]         [Starting]
[10.14.135.36         DDMStoreService10180          ]         [Starting]
[10.14.135.35         DDMStoreService10180          ]         [Starting]
[10.14.135.37         DDMStoreService10180          ]         [Starting]
[10.14.135.36         DDMEngineService27018         ]         [Starting]
[root@c0b087a753da9429454010dd89e4a64e bin]# ddm_cluster_service_10.14.135.36_27018 status
[10.14.135.36         DDMMetaService2379            ]         [Running]
[10.14.135.35         DDMMetaService2379            ]         [Running]
[10.14.135.37         DDMMetaService2379            ]         [Running]
[10.14.135.36         DDMStoreService10180          ]         [Running]
[10.14.135.35         DDMStoreService10180          ]         [Running]
[10.14.135.37         DDMStoreService10180          ]         [Running]
[10.14.135.36         DDMEngineService27018         ]         [Running]
[root@c0b087a753da9429454010dd89e4a64e bin]#
```

4.4.2.1.3. 连接示例

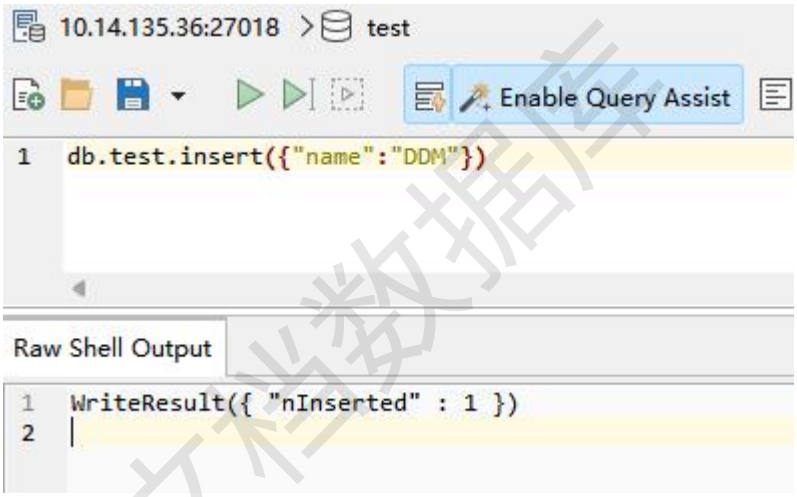
- Studio 3T

  使用 3T 工具，连接 ddm_engine，并做数据插入查询操作。

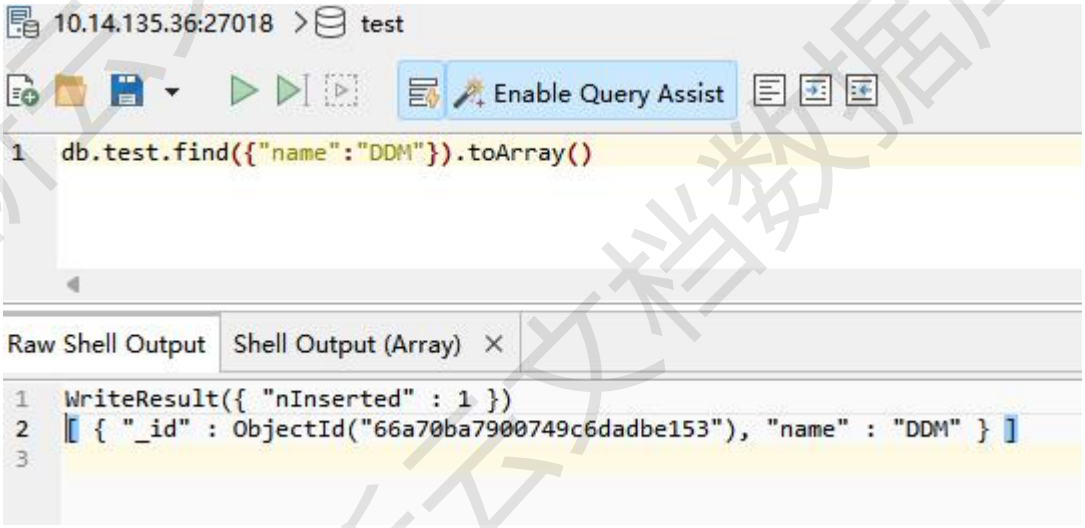
1) 测试连接 engine



2） 连接后添加文档



3） 查询文档



4） 至此 ddm 分布式集群搭建并验证完成。

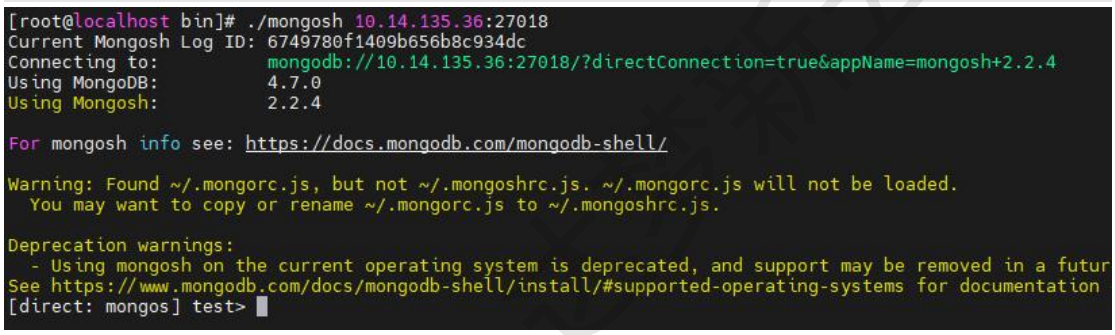
- **mongosh 连接**

  使用 mongosh 工具，连接 ddm_engine，并做数据插入查询操作。

1） 连接测试

进入安装目录的 dmncdb/ddm_XXX/thirdparty/tools 目录下执行以下命令

```
./mongosh --host 10.14.135.36 --port 27018
```



2） 连接后添加文档

```
db.test.insert({"name":"DDM"})
```

```
[root@localhost bin]# ./mongosh 10.14.135.36:27018
Current Mongosh Log ID: 6749780f1409b656b8c934dc
Connecting to:          mongodb://10.14.135.36:27018/?directConnection=true&appName=mongosh+2.2
Using MongoDB:          4.7.0
Using Mongosh:          2.2.4

For mongosh info see: https://docs.mongodb.com/mongodb-shell/

Warning: Found ~/.mongorc.js, but not ~/.mongoshrc.js. ~/.mongorc.js will not be loaded.
  You may want to copy or rename ~/.mongorc.js to ~/.mongoshrc.js.

Deprecation warnings:
  - Using mongosh on the current operating system is deprecated, and support may be removed in
See https://www.mongodb.com/docs/mongodb-shell/install/#supported-operating-systems for documen
[direct: mongos] test> db.test.insert({"name":"DDM"})
DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or bulkWrite.
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('674978d01409b656b8c934dd') }
}
[direct: mongos] test>
```

3） 查询文档

```
db.test.find({"name":"DDM"}).toArray()
```

```
[root@localhost bin]# ./mongosh 10.14.135.36:27018
Current Mongosh Log ID: 6749780f1409b656b8c934dc
Connecting to:          mongodb://10.14.135.36:27018/?directConnection=true&appName=mongosh+2.2.4
Using MongoDB:          4.7.0
Using Mongosh:          2.2.4

For mongosh info see: https://docs.mongodb.com/mongodb-shell/

Warning: Found ~/.mongorc.js, but not ~/.mongoshrc.js. ~/.mongorc.js will not be loaded.
  You may want to copy or rename ~/.mongorc.js to ~/.mongoshrc.js.

Deprecation warnings:
  - Using mongosh on the current operating system is deprecated, and support may be removed in a fut
See https://www.mongodb.com/docs/mongodb-shell/install/#supported-operating-systems for documentatio
[direct: mongos] test> db.test.insert({"name":"DDM"})
DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or bulkWrite.
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('674978d01409b656b8c934dd') }
}
[direct: mongos] test> db.test.find({"name":"DDM"}).toArray()
[ { _id: ObjectId('674978d01409b656b8c934dd'), name: 'DDM' } ]
[direct: mongos] test>
```

4） 至此 ddm 单机搭建并验证完成

4.4.2.2. 主备集群安装

ddm 的主备集群，即双 ddm_store 的集群，两个 ddm_store 相互备份，任意一个节点故障都不影响集群持续对外服务。本节将以 3 个 ddm_meta、2 个 ddm_store、2 个 ddm_engine 为做为搭建示例。

- 节点规划

  主备双 store 部署 3-2-2 集群，节点规划如下：

节点	地址
ddm_meta1	10.14.135.35:2712
ddm_meta2	10.14.135.36:2712
ddm_meta3	10.14.135.37:2712
ddm_store1	10.14.135.35:10112
ddm_store2	10.14.135.36:10112
ddm_engine1	10.14.135.35:27112
ddm_engine2	10.14.135.36:27112

4.4.2.2.1. 手动安装

主备集群版本将安装包分别上传到三台部署的机器上，根据解压指引解压到相同的路径下，进入 dmncdb/ddm_XXX/bin/scripts 目录下，依次按照 ddm_meta、ddm_store、ddm_engine 顺序启动服务，即完成了主备集群版本 ddm 对服务的搭建。以下脚本传入的参数解释参见安装。

注意:如使用非 root 用户部署，三台部署机器需要分别创建相同的用户并给安装包赋用户组权限，以下命令如果在非 root 用户下执行，需要加 sudo 或者切换到 root 用户下执行。

1）根据节点规划给脚本传人参数进行注册 mete 组件服务，注册后手动启动组件服务。

①在 10.14.135.35 机器上部署 metel 节点

```
./ddm_meta_service_installer.sh -mi 10.14.135.35 -mp 2712 -pmp 2380 -data /data/ddm_cluster -ic true
systemctl start DDMMetaService2712
```

```
[root@1a16d1b76cd38473ab21107a376fde46 scripts]# ./ddm_meta_service_installer.sh -mi 10.14.135.35 -mp 2712 -pmp 2380 -data /data/ddm_cluster -ic true
Created symlink /etc/systemd/system/multi-user.target.wants/DDMMetaService2712.service → /usr/lib/systemd/system/DDMMetaService2712.service.
创建服务(DDMMetaService2712)完成
[root@1a16d1b76cd38473ab21107a376fde46 scripts]# systemctl start DDMMetaService2712
[root@1a16d1b76cd38473ab21107a376fde46 scripts]#
```

②在 10.14.135.36 机器上部署 mete2 节点

```
./ddm_meta_service_installer.sh -mi 10.14.135.36 -mp 2712 -pmp 2380 -data /data/ddm_cluster -jc true -jcn Meta2 -jci 10.14.135.35 -jcp 2712
systemctl start DDMMetaService2712
```

```
[root@c0b087a753da9429454010dd89e4a64e scripts]# ./ddm_meta_service_installer.sh -mi 10.14.135.36 -mp 2712 -pmp 2380 -data /data/ddm_cluster -jc true -jcn Meta2 -jci 10.14.135.35 -jcp 2712
Created symlink /etc/systemd/system/multi-user.target.wants/DDMMetaService2712.service → /usr/lib/systemd/system/DDMMetaService2712.service.
创建服务(DDMMetaService2712)完成
[root@c0b087a753da9429454010dd89e4a64e scripts]# systemctl start DDMMetaService2712
[root@c0b087a753da9429454010dd89e4a64e scripts]#
```

③在 10.14.135.37 机器上部署 mete3 节点

./ddm_meta_service_installer.sh -mi 10.14.135.37 -mp 2712 -pmp 2380 -data /data/ddm_cluster -jc true -jcn Meta3 -jci 10.14.135.35 -jcp 2712

systemctl start DDMMetaService2712

```
[root@d6558399e0e5ef03ed4a9c9150b6dc29 scripts]# ./ddm_meta_service_installer.sh -mi 10.14.135.37 -mp 2712 -pmp 2380 -data /data/ddm_cluster -jc true -jcn Meta3 -jci 10.14.135.35 -jcp 2712
Created symlink /etc/systemd/system/multi-user.target.wants/DDMMetaService2712.service → /usr/lib/systemd/system/DDMMetaService2712.service.
创建服务(DDMMetaService2712)完成
[root@d6558399e0e5ef03ed4a9c9150b6dc29 scripts]# systemctl start DDMMetaService2712
[root@d6558399e0e5ef03ed4a9c9150b6dc29 scripts]#
```

2）根据节点规划给脚本传人参数进行注册 store 组件服务，注册后手动启动组件服务。

①在 10.14.135.35 机器上部署 store1 节点

./ddm_store_service_installer.sh -si 10.14.135.35 -sp 10112 -pd 10.14.135.35:2712,10.14.135.36:2712,10.14.135.37:2712 -data /data/ddm_cluster

systemctl start DDMStoreService10112

```
[root@1a16d1b76cd38473ab21107a376fde46 scripts]# ./ddm_store_service_installer.sh  -si 10.14.135.35 -sp 10112 -pd 10.14.135.35:2712,10.14.135.36:2712,1
_cluster
创建服务(DDMStoreService10112)完成
[root@1a16d1b76cd38473ab21107a376fde46 scripts]# systemctl start DDMStoreService10112
[root@1a16d1b76cd38473ab21107a376fde46 scripts]#
```

②在 10.14.135.36 机器上部署 store2 节点

./ddm_store_service_installer.sh -si 10.14.135.36 -sp 10112 -pd 10.14.135.35:2712,10.14.135.36:2712,10.14.135.37:2712 -data /data/ddm_cluster

systemctl start DDMStoreService10112

```
[root@c0b087a753da9429454010dd89e4a64e scripts]# ./ddm_store_service_installer.sh  -si 10.14.135.36 -sp 10112 -pd 10.14.135.35:2712,10.14.135.36:2712,10
Created symlink /etc/systemd/system/multi-user.target.wants/DDMStoreService10112.service → /usr/lib/systemd/system/DDMStoreService10112.service.
创建服务(DDMStoreService10112)完成
[root@c0b087a753da9429454010dd89e4a64e scripts]# systemctl start DDMStoreService10112
[root@c0b087a753da9429454010dd89e4a64e scripts]#
```

3）根据节点规划给脚本传人参数进行注册 engine 组件服务，注册后手动启动组件服务。

①在 10.14.135.35 机器上部署 engine1 节点

./ddm_engine_service_installer.sh -ei 10.14.135.35 -ep 27112 -path 10.14.135.35:2712,10.14.135.36:2712,10.14.135.37:2712 -data /data/ddm_cluster

systemctl start DDMEngineService27112

```
[root@1a16d1b76cd38473ab21107a376fde46 scripts]# ./ddm_engine_service_installer.sh  -ei 10.14.135.35 -ep 27112 -path 10.14.135.35:2712,10.14.
ddm_cluster
创建服务(DDMEngineService27112)完成
[root@1a16d1b76cd38473ab21107a376fde46 scripts]# systemctl start DDMEngineService27112
[root@1a16d1b76cd38473ab21107a376fde46 scripts]#
```

①在 10.14.135.36 机器上部署 engine2 节点

./ddm_engine_service_installer.sh -ei 10.14.135.36 -ep 27112 -path 10.14.135.35:2712,10.14.135.36:2712,10.14.135.37:2712 -data /data/ddm_cluster

systemctl start DDMEngineService27112

```
[root@c0b087a753da9429454010dd89e4a64e scripts]# ./ddm_engine_service_installer.sh  -ei 10.14.135.36 -ep 27112 -path 10.14.135.35:2712,10.14.135.36:2712,
Created symlink /etc/systemd/system/multi-user.target.wants/DDMEngineService27112.service → /usr/lib/systemd/system/DDMEngineService27112.service.
创建服务(DDMEngineService27112)完成
[root@c0b087a753da9429454010dd89e4a64e scripts]# systemctl start DDMEngineService27112
[root@c0b087a753da9429454010dd89e4a64e scripts]#
```

注意：以上 1）2）3）步骤执行完成后，即主备集群搭建完成，如想进一步进行服务状态的管控，可参考步骤 4）,其中步骤 4）为非必选项。

4）服务管控脚本一键管控所有节点服务（切换 root 用户）

切换到部署 engine 的第一个节点的机器上（10.14.135.35）

①进入 dmncdb/ddm_cluster-XXX/conf 目录下创建 install_info 文件夹

```
mkdir -p install_info
```

② 将 dmncdb/ddm_cluster_XXX/conf 目录下的 hosts 文件复制至 install_info 文件夹下

③在 hosts 中填写节点相关信息（将步骤 1）、2）、3）部署的相关 IP 和端口号以及数据存储路径对应填写）

```
# (2) META_PORT 表示运行 DDM_Meta 服务的端口号，端口号建议使用 2379；
# (3) PEER_META_PORT 请使用不同于 META_PORT 的端口号；
# (4) USER 表示运行 DDM_Meta 服务的机器系统用户，非 root 用户需要部署前配置好该用户的 sudo 免密权限；
# (5) PWD 表示运行 DDM_Meta 服务的机器系统用户的密码；
# (6) SSH_PORT 表示运行设备系统的 ssh 端口。
#
# 例如，部署包括三个 DDM_Meta 节点的集群：
# Meta: "ip=192.168.1.10", "port=2379", "peer_port=2380", "user=root", "password=123456", "sshPort=22"
# Meta: "ip=192.168.1.20", "port=2379", "peer_port=2380", "user=root", "password=123456", "sshPort=22"
# Meta: "ip=192.168.1.30", "port=2379", "peer_port=2380", "user=root", "password=123456", "sshPort=22"
#
[DDM_Meta]

Meta: "ip=10.14.135.35", "port=2712", "peer_port=2380", "user=root", "password=123456", "sshPort=22"
Meta: "ip=10.14.135.36", "port=2712", "peer_port=2380", "user=root", "password=123456", "sshPort=22"
Meta: "ip=10.14.135.37", "port=2712", "peer_port=2380", "user=root", "password=123456", "sshPort=22"


# DDM_Store 部署节点信息
#
# 节点个数不限（分布式集群建议部署奇数个节点），每添加一个节点的部署，都需要增加一条 "Store" 记录，其中：
# (1) STORE_IP 表示运行 DDM_Store 服务的机器 ip 地址；
# (2) STORE_PORT 表示运行 DDM_Store 服务的端口号，端口号建议使用 10180；
# (3) USER 表示运行 DDM_Store 服务的机器系统用户，非 root 用户需要部署前配置好该用户的 sudo 免密权限；
# (4) PWD 表示运行 DDM_Store 服务的机器系统用户的密码；
# (5) SSH_PORT 表示运行设备系统的 ssh 端口。
#
# 例如，部署包括三个 DDM_Store 节点的集群：
# Store: "ip=192.168.1.20", "port=10180", "user=root", "password=123456", "sshPort=22"
# Store: "ip=192.168.1.30", "port=10180", "user=root", "password=123456", "sshPort=22"
# Store: "ip=192.168.1.40", "port=10180", "user=root", "password=123456", "sshPort=22"
#
[DDM_Store]

Store: "ip=10.14.135.35", "port=10112", "user=root", "password=123456", "sshPort=22"
Store: "ip=10.14.135.36", "port=10112", "user=root", "password=123456", "sshPort=22"

# DDM_Engine 部署节点信息
#
# 节点个数不限，每添加一个节点的部署，都需要增加一条 "Engine" 记录，其中：
# (1) ENGINE_IP 表示运行 DDM_Engine 服务的机器 ip 地址；
# (2) ENGINE_PORT 表示运行 DDM_Engine 服务的端口号，端口号建议使用 27017；
# (3) USER 表示运行 DDM_Engine 服务的机器系统用户，非 root 用户需要部署前配置好该用户的 sudo 免密权限；
# (4) PWD 表示运行 DDM_Engine 服务的机器系统用户的密码；
# (5) SSH_PORT 表示运行设备系统的 ssh 端口。
#
# 例如：部署三个 DDM_Engine 节点：
# Engine: "ip=192.168.1.30", "port=27017", "user=root", "password=123456", "sshPort=22"
# Engine: "ip=192.168.1.40", "port=27017", "user=root", "password=123456", "sshPort=22"
# Engine: "ip=192.168.1.50", "port=27017", "user=root", "password=123456", "sshPort=22"
[DDM_Engine]

Engine: "ip=10.14.135.35", "port=27112", "user=root", "password=123456", "sshPort=22"
Engine: "ip=10.14.135.36", "port=27112", "user=root", "password=123456", "sshPort=22"

# 数据文件目录
# 请确保数据文件存放的路径有足够大的磁盘空间
data_dir="/data/ddm_cluster"
```

④ 复制 dmncdb/ddm_cluster_XXX/bin/scripts 目录下 ddm_cluster_service 至/usr/local/bin 目录下，（为了方便查看服务的 IP 和端口号，方便管理，建议将 ddm_cluster_service 改名为 ddm_cluster_service_10.14.135.35_27112（engine 第一个节点的 ip 和端口））

⑤修改 ddm_cluster_service 中，DDMDC_HOME 的值修改为安装包目录 bin 的上一层目录，DDMDC_INI_PATH 的值修改为 install_info 文件夹下 hosts 的实际路径。

⑥ 可以用 ddm_cluster_service_<ip_port> start/stop/restart/status/remove 进行服务管控。

```
[root@1a16d1b76cd38473ab21107a376fde46 bin]# ddm_cluster_service_10.14.135.35_27112 restart
[10.14.135.36         DDMEngineService27112          ]          [Stopping]
[10.14.135.35         DDMEngineService27112          ]          [Stopping]
[10.14.135.36         DDMStoreService10112           ]          [Stopping]
[10.14.135.35         DDMStoreService10112           ]          [Stopping]
[10.14.135.37         DDMMetaService2712             ]          [Stopping]
[10.14.135.36         DDMMetaService2712             ]          [Stopping]
[10.14.135.35         DDMMetaService2712             ]          [Stopping]

[10.14.135.35         DDMMetaService2712             ]          [Starting]
[10.14.135.36         DDMMetaService2712             ]          [Starting]
[10.14.135.37         DDMMetaService2712             ]          [Starting]
[10.14.135.35         DDMStoreService10112           ]          [Starting]
[10.14.135.36         DDMStoreService10112           ]          [Starting]
[10.14.135.35         DDMEngineService27112          ]          [Starting]
[10.14.135.36         DDMEngineService27112          ]          [Starting]
[root@1a16d1b76cd38473ab21107a376fde46 bin]# ddm_cluster_service_10.14.135.35_27112 status
[10.14.135.35         DDMMetaService2712             ]          [Running]
[10.14.135.36         DDMMetaService2712             ]          [Running]
[10.14.135.37         DDMMetaService2712             ]          [Running]
[10.14.135.35         DDMStoreService10112           ]          [Running]
[10.14.135.36         DDMStoreService10112           ]          [Running]
[10.14.135.35         DDMEngineService27112          ]          [Running]
[10.14.135.36         DDMEngineService27112          ]          [Running]
[root@1a16d1b76cd38473ab21107a376fde46 bin]#
```

4.4.2.2.2. 一键安装

在执行一键安装脚本前，先将规划好 3-2-2 分布式集群的地址填写进 conf 文件夹下的 hosts 文件中。hosts 文件 ddm_meta 部署节点信息配置如下：

```
[ddm_meta]
Meta: "ip=10.14.135.35", "port=2712", "peer_port=2380", "user=root", "password=***","sshPort=22"
Meta: "ip=10.14.135.36", "port=2712", "peer_port=2380", "user=root", "password=***","sshPort=22"
Meta: "ip=10.14.135.37", "port=2712", "peer_port=2380", "user=root", "password=***","sshPort=22"
```

ddm_store 部署节点信息配置如下：

```
[ddm_store]
Store: "ip=10.14.135.35", "port=10112", "user=root", "password=***","sshPort=22"
Store: "ip=10.14.135.36", "port=10112", "user=root", "password=***","sshPort=22"
```

ddm_engine 部署节点信息配置如下：

```
[ddm_engine]
Engine:"ip=10.14.135.35", "port=27112", "user=root", "password=***","sshPort=22"
```

Engine: "ip=10.14.135.36", "port=27112", "user=root", "password=***","sshPort=22"

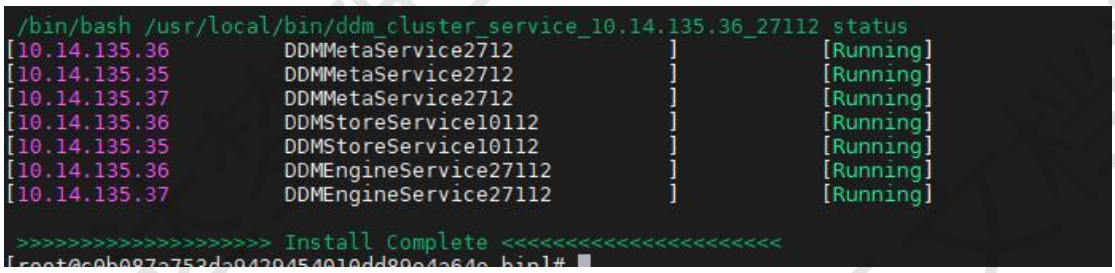
数据文件目录配置后，将当前服务器所有节点的数据文件将存储在这个路径下。Hosts 文件中数据文件目录配置如下：

data_dir="/home/kaka/ddm_cluster"

按照规划配置好 hosts 文件后，在 bin 目录下执行 ddm_cluster_install.sh,脚本将依照 hosts 文件中规划配置，依次部署启动元数据节点服务、存储数据节点服务和执行节点服务。并为各组件服务自动注册系统服务，设置该服务为开启自启动模式。

出现 Press [Enter] key to continue [Enter]时，点击回车键，进行下一步，会自动开始部署。

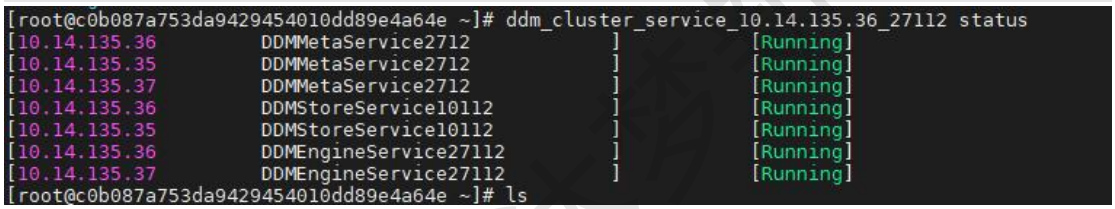
一键部署脚本成功执行后会出现以下界面，各服务状态均为 Running,打印 Install Complete。



- 服务状态查看

使用以下脚本查看集群所有节点的状态：

ddm_cluster_service_<*ip_port*> status



以上所有服务都可以用 systemctl 对每个节点服务进行 start，restart，stop 操作。

systemctl stop DDMStoreServiceXXX

```
[root@c0b087a753da9429454010dd89e4a64e conf]# systemctl stop DDMStoreService10112
[root@c0b087a753da9429454010dd89e4a64e conf]# ddm_cluster_service_10.14.135.36_27112 status
[10.14.135.36         DDMMetaService2712          ]          [Running]
[10.14.135.35         DDMMetaService2712          ]          [Running]
[10.14.135.37         DDMMetaService2712          ]          [Running]
[10.14.135.36         DDMStoreService10112        ]          [Failed]
[10.14.135.35         DDMStoreService10112        ]          [Running]
[10.14.135.36         DDMEngineService27112       ]          [Running]
[10.14.135.37         DDMEngineService27112       ]          [Running]
[root@c0b087a753da9429454010dd89e4a64e conf]#
```

如果是对 ddm_cluster_service 统一操作，则可以用 ddm_cluster_service_*<ip_port>* start/stop/restart 进行统一操作。

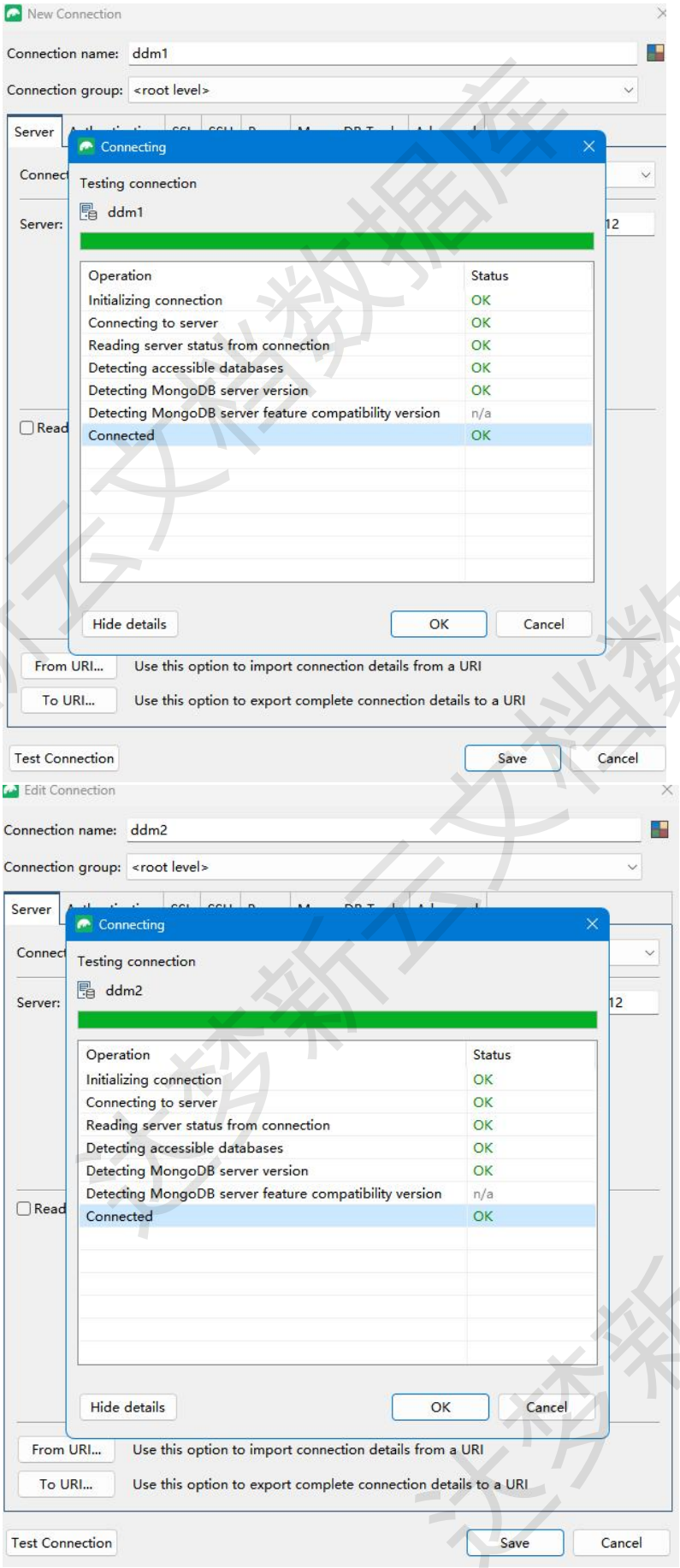
```
[root@c0b087a753da9429454010dd89e4a64e conf]# ddm_cluster_service_10.14.135.36_27112 restart
[10.14.135.37         DDMEngineService27112       ]          [Stopping]
[10.14.135.36         DDMEngineService27112       ]          [Stopping]
[10.14.135.35         DDMStoreService10112        ]          [Stopping]
[10.14.135.36         DDMStoreService10112        ]          [Stopped]
[10.14.135.37         DDMMetaService2712          ]          [Stopping]
[10.14.135.35         DDMMetaService2712          ]          [Stopping]
[10.14.135.36         DDMMetaService2712          ]          [Stopping]

[10.14.135.36         DDMMetaService2712          ]          [Starting]
[10.14.135.35         DDMMetaService2712          ]          [Starting]
[10.14.135.37         DDMMetaService2712          ]          [Starting]
[10.14.135.36         DDMStoreService10112        ]          [Starting]
[10.14.135.35         DDMStoreService10112        ]          [Starting]
[10.14.135.36         DDMEngineService27112       ]          [Starting]
[10.14.135.37         DDMEngineService27112       ]          [Starting]
[root@c0b087a753da9429454010dd89e4a64e conf]# ddm_cluster_service_10.14.135.36_27112 status
[10.14.135.36         DDMMetaService2712          ]          [Running]
[10.14.135.35         DDMMetaService2712          ]          [Running]
[10.14.135.37         DDMMetaService2712          ]          [Running]
[10.14.135.36         DDMStoreService10112        ]          [Running]
[10.14.135.35         DDMStoreService10112        ]          [Running]
[10.14.135.36         DDMEngineService27112       ]          [Running]
[10.14.135.37         DDMEngineService27112       ]          [Running]
[root@c0b087a753da9429454010dd89e4a64e conf]#
```

4.4.2.2.3. 连接示例

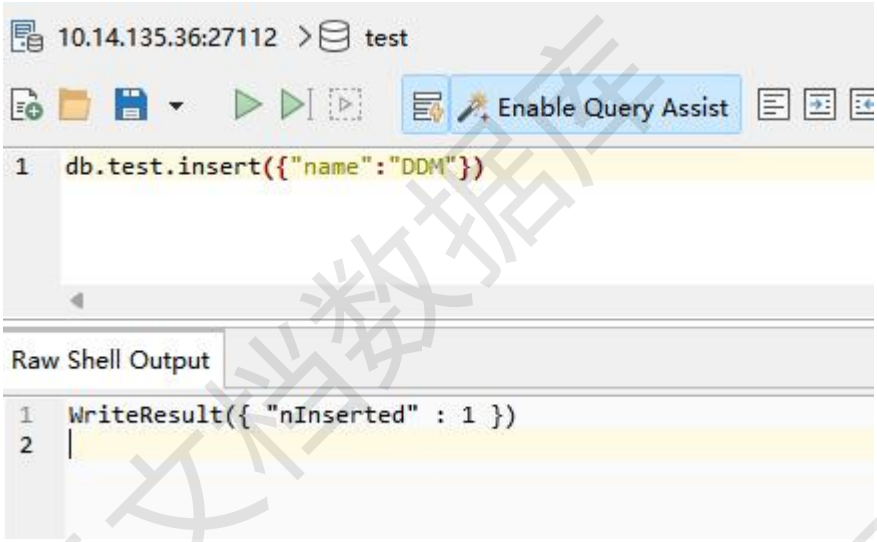
- Studio 3T

  使用 3T 工具，连接 ddm_engine，并做数据插入查询操作。

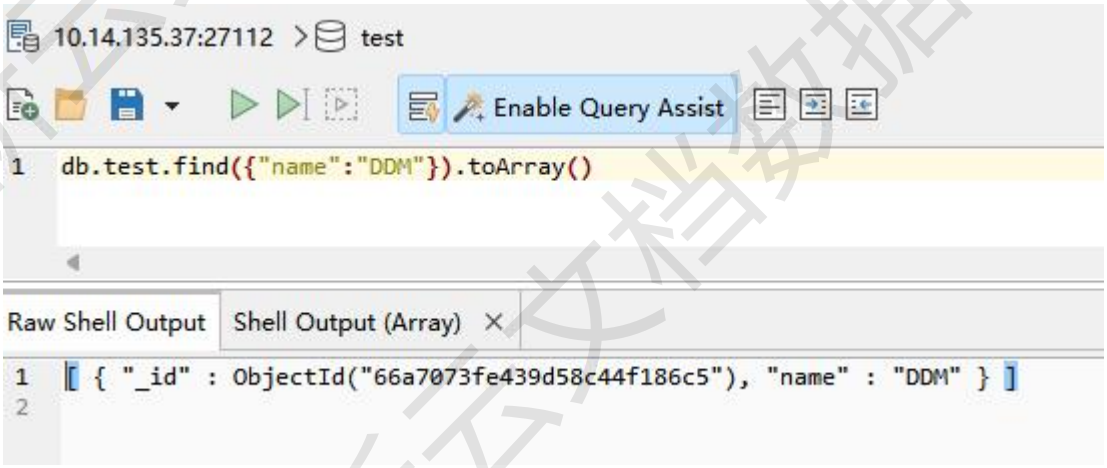
1） 测试连接 engine1 和 engine2


New Connection
Connection name: ddm1
Connection group: <root level>
Server
Connecting
Testing connection
ddm1
Operation
Status
Initializing connection
OK
Connecting to server
OK
Reading server status from connection
OK
Detecting accessible databases
OK
Detecting MongoDB server version
OK
Detecting MongoDB server feature compatibility version
n/a
Connected
OK
Hide details
OK
Cancel
From URI...
Use this option to import connection details from a URI
To URI...
Use this option to export complete connection details to a URI
Test Connection
Save
Cancel
Edit Connection
Connection name: ddm2
Connection group: <root level>
Server
Connecting
Testing connection
ddm2
Operation
Status
Initializing connection
OK
Connecting to server
OK
Reading server status from connection
OK
Detecting accessible databases
OK
Detecting MongoDB server version
OK
Detecting MongoDB server feature compatibility version
n/a
Connected
OK
Hide details
OK
Cancel
From URI...
Use this option to import connection details from a URI
To URI...
Use this option to export complete connection details to a URI
Test Connection
Save
Cancel

2） engine1 连接后添加文档



3） engine2 连接后查询文档



4） 至此 ddm 主备集群搭建并验证完成。

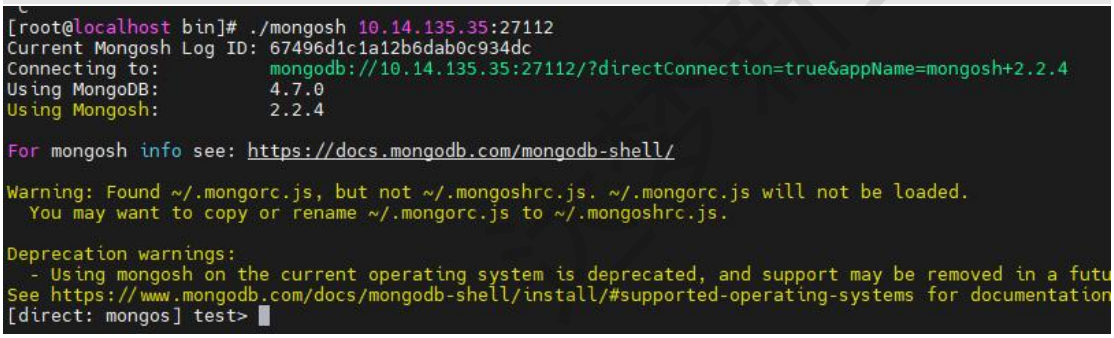
- Mongosh

使用 mongosh 工具，连接 ddm_engine1 和 ddm_engine2，并做数据插入查询操作。

1） 连接测试 engine1 和 engine2

进入安装目录的 dmncdb/ddm_XXX/thirdparty/tools 目录下执行以下命令

```
./mongosh --host 10.14.135.35 --port 27112
```



./mongosh --host 10.14.135.36--port 27112

```
[root@localhost bin]# ./mongosh 10.14.135.36:27112
Current Mongosh Log ID: 67496db178cfe9c0eac934dc
Connecting to:          mongodb://10.14.135.36:27112/?directConnection=true&appName=mongosh+2.2.4
Using MongoDB:          4.7.0
Using Mongosh:          2.2.4

For mongosh info see: https://docs.mongodb.com/mongodb-shell/

Warning: Found ~/.mongorc.js, but not ~/.mongoshrc.js. ~/.mongorc.js will not be loaded.
  You may want to copy or rename ~/.mongorc.js to ~/.mongoshrc.js.

Deprecation warnings:
  - Using mongosh on the current operating system is deprecated, and support may be removed in a fu
See https://www.mongodb.com/docs/mongodb-shell/install/#supported-operating-systems for documentatio
[direct: mongos] test>
```

2） engine1 连接后添加文档

db.test.insert({"name":"DDM"})

```
[root@localhost bin]# ./mongosh 10.14.135.35:27112
Current Mongosh Log ID: 67496deea7b2929e9dc934dc
Connecting to:          mongodb://10.14.135.35:27112/?directConnection=true&appName=mongosh+2.2.4
Using MongoDB:          4.7.0
Using Mongosh:          2.2.4

For mongosh info see: https://docs.mongodb.com/mongodb-shell/

Warning: Found ~/.mongorc.js, but not ~/.mongoshrc.js. ~/.mongorc.js will not be loaded.
  You may want to copy or rename ~/.mongorc.js to ~/.mongoshrc.js.

Deprecation warnings:
  - Using mongosh on the current operating system is deprecated, and support may be removed in a future
See https://www.mongodb.com/docs/mongodb-shell/install/#supported-operating-systems for documentation
[direct: mongos] test> db.test.insert({"name":"DDM"})
DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or bulkWrite.
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('67496e0ca7b2929e9dc934dd') }
}
[direct: mongos] test>
```

3） engine2 连接后查询文档

db.test.find({"name":"DDM"}).toArray()

```
[root@localhost bin]# ./mongosh 10.14.135.36:27112
Current Mongosh Log ID: 67496e470615692c29c934dc
Connecting to:          mongodb://10.14.135.36:27112/?directConnection=true&appName=mongosh+2.2.4
Using MongoDB:          4.7.0
Using Mongosh:          2.2.4

For mongosh info see: https://docs.mongodb.com/mongodb-shell/

Warning: Found ~/.mongorc.js, but not ~/.mongoshrc.js. ~/.mongorc.js will not be loaded.
  You may want to copy or rename ~/.mongorc.js to ~/.mongoshrc.js.

Deprecation warnings:
  - Using mongosh on the current operating system is deprecated, and support may be removed in a
See https://www.mongodb.com/docs/mongodb-shell/install/#supported-operating-systems for documentat
[direct: mongos] test> db.test.find({"name":"DDM"}).toArray()
[ { _id: ObjectId('67496e0ca7b2929e9dc934dd'), name: 'DDM' } ]
[direct: mongos] test>
```

4） 至此 ddm 主备集群搭建并验证完成。

4.4.3. 卸载

4.4.3.1. 单组件卸载

在 bin 目录的 scripts 文件夹中有 ddm_meta、ddm_engine、ddm_store 节点卸载的脚本，以 ddm_store 为例：

```
./ddm_store_service_uninstaller.sh -n DDMStoreServiceXXX
```

```
[root@localhost scripts]# ./ddm_store_service_uninstaller.sh -n DDMStoreService10181
是否删除服务(DDMStoreService10181)?(Y/y:是 N/n:否): y
Removed /etc/systemd/system/multi-user.target.wants/DDMStoreService10181.service.
删除服务文件(/usr/lib/systemd/system/DDMStoreService10181.service)完成
删除服务(DDMStoreService10181)完成
```

4.4.3.2. 单机卸载

- 只卸载服务

ddm_service 可以执行 start|stop|restart|status|remove，remove 可以卸载整个集群服务。

```
ddm_service_<ip_port> remove
```

```
[root@jenkins bin]# ddm_service_10.14.1.160_27017 remove
[10.14.1.160          DDMEngineService27017         ]          [Removing]
[10.14.1.160          DDMStoreService10180          ]          [Removing]
[10.14.1.160          DDMMetaService2379            ]          [Removing]

[root@jenkins bin]#
```

- 卸载服务且删除安装包

在 bin 目录下的 ddm_uninstall.sh 为卸载脚本，-n 指定需要卸载的数据库的 ip 和端口号

```
./ddm_uninstall.sh -n <ip_port>
```

```
[root@jenkins bin]# ./ddm_uninstall.sh  -h
Usage: ddm_uninstall.sh [-n ddm_uninstall_engine_ip_port]
  or ddm_uninstall.sh -h

Example: ./ddm_uninstall.sh -n 192.168.1.10_27017


   -n               卸载 DDM 中 Engine 的 ip 和 port
   -h               帮助
[root@jenkins bin]# ./ddm_uninstall.sh  -n 10.14.1.160_27017
请确认是否卸载 DDM 安装包？ [Y/y:是 N/n:否]: y
[10.14.1.160          DDMEngineService27017         ]          [Removing]
[10.14.1.160          DDMStoreService10180          ]          [Removing]
[10.14.1.160          DDMMetaService2379            ]          [Removing]
```

4.4.3.3. 集群卸载

- 只卸载服务

ddm_cluster_service 可以执行 start|stop|restart|status|remove，remove 可以卸载整个集群服务（在执行一键部署的节点上执行）。

```
ddm_cluster_service_<ip_port> remove
```

```
[root@c0b087a753da9429454010dd89e4a64e bin]# ddm_cluster_service_10.14.135.36_27112 remove
[10.14.135.37         DDMEngineService27112         ]          [Removing]
[10.14.135.36         DDMEngineService27112         ]          [Removing]
[10.14.135.35         DDMStoreService10112          ]          [Removing]
[10.14.135.36         DDMStoreService10112          ]          [Removing]
[10.14.135.37         DDMMetaService2712            ]          [Removing]
[10.14.135.35         DDMMetaService2712            ]          [Removing]
[10.14.135.36         DDMMetaService2712            ]          [Removing]
[root@c0b087a753da9429454010dd89e4a64e bin]#
```

- 卸载服务且删除安装包

在 bin 目录下的 ddm_cluster_uninstall.sh 为卸载脚本，直接执行选择需要卸载的集群，即卸载服务且删除安装包

```
./ddm_cluster_uninstall.sh
```

```
[root@c0b087a753da9429454010dd89e4a64e bin]# ./ddm_cluster_uninstall.sh
+------------------------------------------------------------------------------+

          (1) hosts_10.14.135.36_27112

+------------------------------------------------------------------------------+
请选择要操作的 hosts 文件选项（默认：1）：1

请确认是否移除 DDM 相关服务，并卸载安装包？ [Y/y:是 N/n:否]: y
[10.14.135.37         DDMEngineService27112         ]          [Removing]
[10.14.135.36         DDMEngineService27112         ]          [Removing]
[10.14.135.35         DDMStoreService10112          ]          [Removing]
[10.14.135.36         DDMStoreService10112          ]          [Removing]
[10.14.135.37         DDMMetaService2712            ]          [Removing]
[10.14.135.35         DDMMetaService2712            ]          [Removing]
[10.14.135.36         DDMMetaService2712            ]          [Removing]
spawn ssh dmncdb@10.14.135.35 sudo rm -rf /opt/dmncdb/ddm_linux_arm64_v1.0.2.107_cluster/bin/{scripts,ddm

Authorized users only. All activities may be monitored and reported.
dmncdb@10.14.135.35's password:
spawn ssh dmncdb@10.14.135.37 sudo rm -rf /opt/dmncdb/ddm_linux_arm64_v1.0.2.107_cluster/bin/{scripts,ddm

Authorized users only. All activities may be monitored and reported.
dmncdb@10.14.135.37's password:
Skipping 10.14.135.36, already processed.
Skipping 10.14.135.35, already processed.
Skipping 10.14.135.36, already processed.
Skipping 10.14.135.37, already processed.
```

4.4.3.4. rpm 包卸载

查看安装包名称

```
rpm -qa | grep ddm
```

```
[root@jenkins amd]# rpm -qa | grep ddm
ddm_sa-V1.0-20240726_82f3217b.x86_64
ddm_cluster-V1.0-20240725_a047a134.x86_64
ddm_cluster-V1.0-20240726_daab6195.x86_64
[root@jenkins amd]#
```

卸载指定 ddm rpm 安装包名称

注意：在数据库服务卸载掉的情况下，再执行 rpm 包卸载

```
rpm -e ddm_XXX
```

```
[root@jenkins amd]# rpm -e ddm_sa-V1.0-20240726_82f3217b.x86_64
[root@jenkins amd]#
```

4.4.3.5. deb 包卸载

卸载指定 ddm deb 安装包名称

注意：在数据库服务卸载掉的情况下，再执行 deb 包卸载

```
dpkg -r ddm
```

```
root@Linx-1-210:/home/DDM/kkk/opt/dmncdb/ddm_linux_amd64_20241024_5cd20803_sa/bin# dpkg -r ddm
(正在读取数据库 ... 系统当前共安装有 227003 个文件和目录。)
正在卸载 ddm (1.0) ...
```