文档数据库（MongoDB）

产品文档

目录

产品简介
产品概述

简介

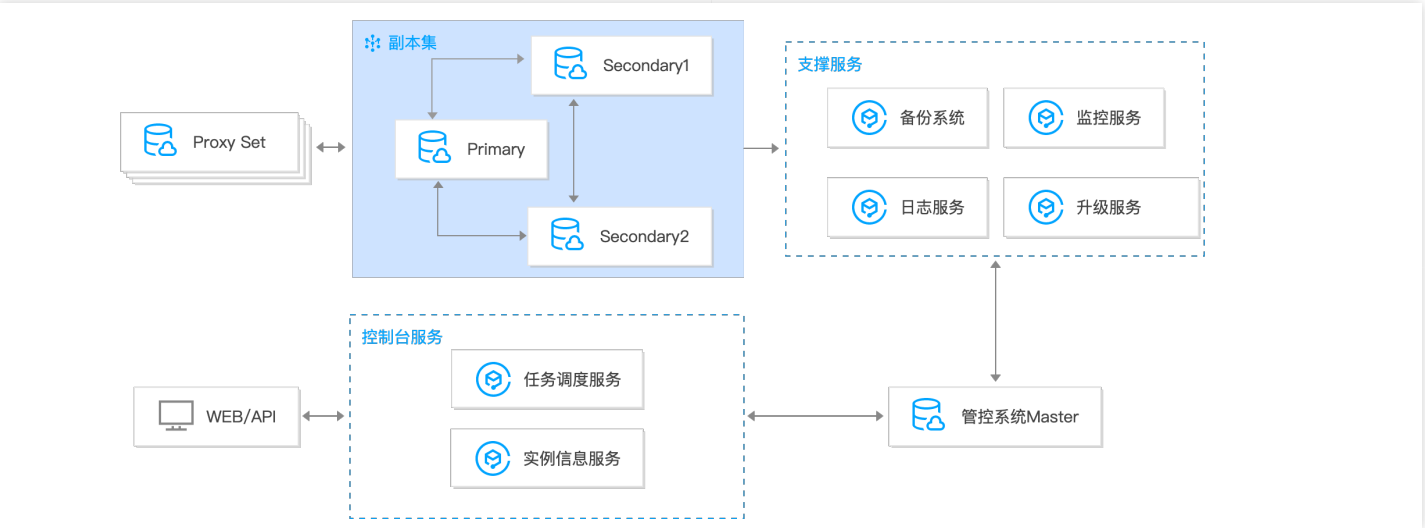
云数据库 MongoDB是我云基于开源非关系型数据库 MongoDB打造专业的高性能、分布式数据存储服务，完全兼容 MongoDB 协议，适用于面向非关系型数据库的场景。

产品特点

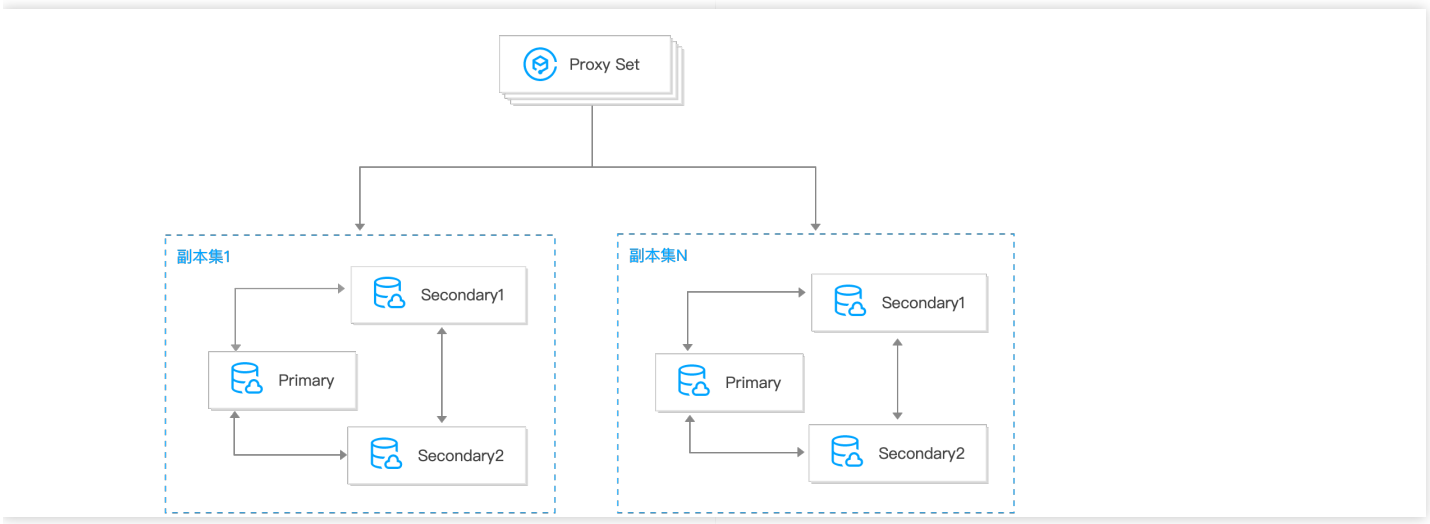
- 提供云存储服务，云存储服务是TCloudFinanceZone平台面向互联网应用的数据存储服务。
- 完全兼容 MongoDB 协议，既适用于传统表结构的场景，也适用于缓存、非关系型数据以及利用 MapReduce 进行大规模数据集的并行运算的场景。
- 提供高性能、可靠、易用、便捷的 MongoDB 集群服务，每一个实例都是至少一主一从的副本集或者是包含多个副本集的分片集群。
- 拥有整合备份、扩容等功能，尽可能的保证用户数据安全以及动态伸缩能力。

产品架构

云数据库 MongoDB 副本集的系统架构图如下：



分片集群由分片、proxy set、config servers 等组件组成，每个分片包含了分片数据的一个子集，云数据库 MongoDB 的每个分片都作为一个副本集部署：


Proxy Set
副本集1
Secondary1
Primary
Secondary2
副本集N
Secondary1
Primary
Secondary2

产品优势

云数据库 MongoDB 优势

- 便捷：用户可以快速地在云平台中申请集群实例资源，通过 URI 直接访问 MongoDB 实例，无需自行安装实例。
- 易用：完全兼容 MongoDB 协议，用户可通过基于 MongoDB 协议的客户端访问实例，可无缝的将原有 MongoDB 应用迁移到云平台。
- 安全：提供至少三份在线的数据存储，确保线上数据安全。同时通过备份机制保存多天的备份数据，以便于在灾难情况下进行数据恢复。

针对传统自建 MongoDB 在使用过程中常出现的性能瓶颈、运维困难、数据可靠性和可用性难题，云数据库 MongoDB 都做了专项优化：

1. 解决运维困难：多达20余项指标自动化监控告警；提供批量数据导入导出，参数模板化修改，帮业务轻松迅速完成部署。
2. 服务高可用：双机甚至更多热备，自动容灾，故障切换和故障转移对用户透明；支持像原生 MongoDB 一样的优先读从库功能，保证高并发读取能力。
3. 数据高可靠：提供数据备份；支持内网防火墙，外网 DDoS 防护。

应用场景

云数据库 MongoDB 是一种通用型数据库，其稳定性、性能、扩展能力基本上可以覆盖绝大部分 no schema 场景，如下是几个典型的应用场景。

游戏行业

游戏需求变化很快，因此 MongoDB 特别适用于游戏后端数据库，使用 MongoDB 存储游戏用户信息、装备、积分等时，会直接以内嵌文档的形式存储，方便查询、更新，no schema 模式可以免去变更表结构的痛苦，大幅度缩短版本迭代周期。
MongoDB 也可当作缓存服务器使用，合理规划热数据，其性能与其他常用缓存服务器相当，同时还为您提供更丰富的查询方式。

移动行业

云数据库 MongoDB 支持二维空间索引，可以方便地查询地理位置关系和检索用户地理位置数据；可实现基于地理位置系统的地图应用和实现附近的人、地点等功能；也可使用 MongoDB 存储用户信息，以及用户发表的朋友圈等信息。

物联网行业

物联网领域的终端设备，例如医疗仪器、运输业车辆 GPS 等，可以轻易且持续的产生 TB 级的数据，使用 MongoDB 可存储所有接入的智能设备的信息，以及设备汇报的日志信息，并对这些信息进行多维度的分析。业务可构建分布式的云数据库 MongoDB 分片集群，达到无上限的容量存储，同时也可在线扩容，轻松处理物联网海量数据。

物流行业

物流订单状态在运送过程中会不断更新，TCloudFinanceZone数据库 MongoDB 存储以 MongoDB 内嵌 JSON 的形式来存储订单信息，一次查询就能将订单所有的变更读取出来。

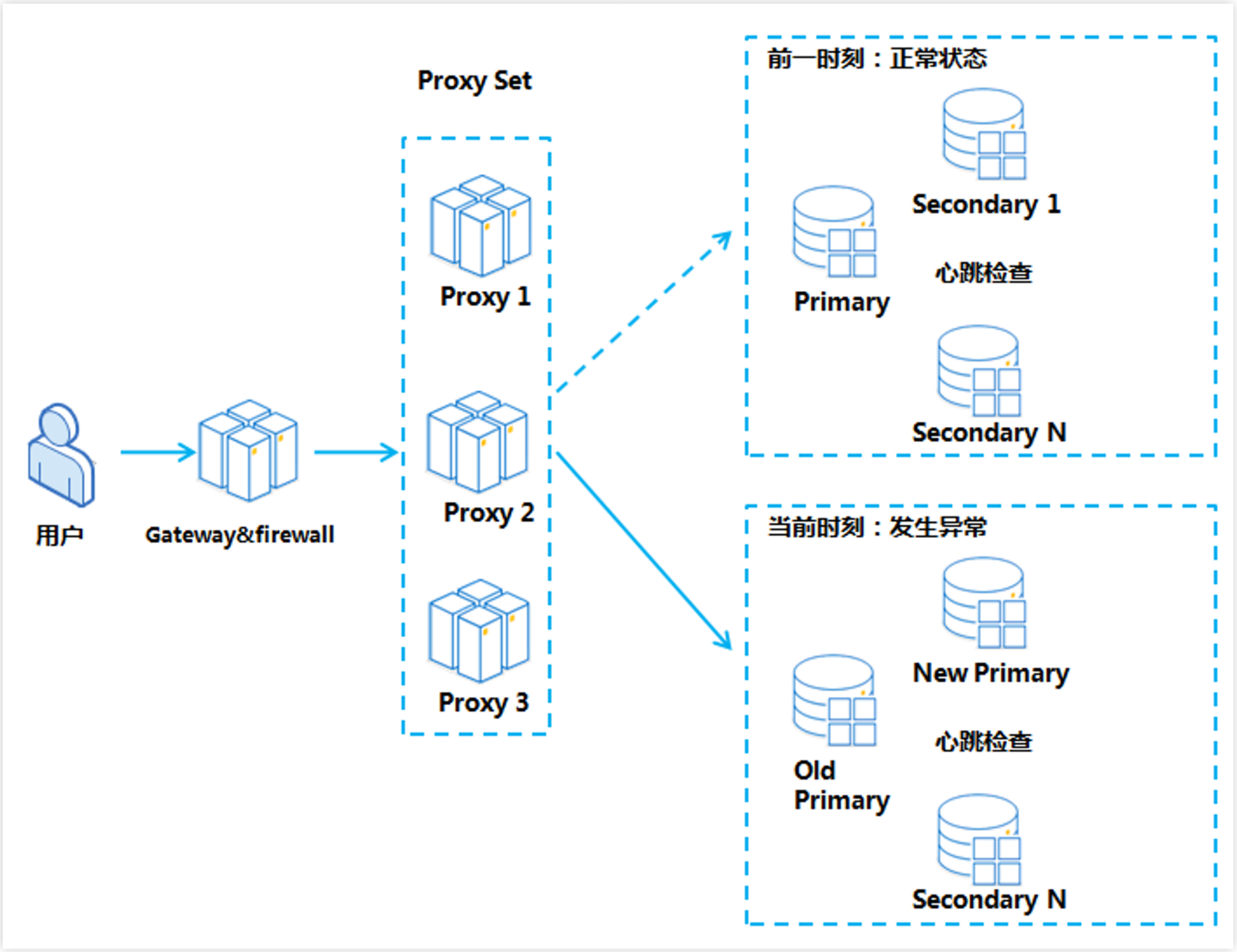
视频直播行业

视频直播行业会产生大量的礼物信息，用户聊天信息等，数据量较大，使用TCloudFinanceZone数据库 MongoDB 可存储用户信息、礼物信息以及日志等信息，同时可通过丰富的聚合查询来进行业务分析。

容灾架构

云数据库 MongoDB 采用主从热备架构，当主节点故障时，服务会自动切换到备节点，主从切换时可能会有10秒左右闪断，请业务做好自动重试。自动容灾原理如下：

1. 当发生意外致使主节点不可达时，集群内部会自动选举出新的主节点。
2. 如果故障的是主节点，重新拉起时，它会变身成一个从节点；如果拉起失败会补充新节点进入集群，以达到用户所选择的集群规模。
3. 同样当任一从节点不可达时，也会尝试拉起节点或者补充新节点。



可用区

简介

可用区指在同一地域内电力和网络互相独立的物理数据中心。目标是能够保证可用区间故障相互隔离（大型灾害或者大型电力故障除外），不出现故障扩散，使得用户的业务持续在线服务。使得在同一地域内可用区与可用区之间内网互通，同一可用区内产品网络延时更小。

功能

支持租户对子账号配置可用区白名单，通过白名单可以控制子账号创建实例的可用区。

购买指南
计费概述

计费模式

云数据库 MongoDB 支持按量计费。

按量计费

- 后付费模式，在业务发展有较大波动性，且无法进行准确预测，或资源使用有临时性和突发性的情况下建议选择按量计费。
- 用户在购买云数据库 MongoDB 时会冻结云账户中一个小时的硬件费用，并在每个整点（国内时间）进行一次结算，计费时间粒度精确到秒。
- 在按量计费模式下，用户只需为云数据库 MongoDB 的实际使用量付费，不需要提前支付费用。
- 计费方式采用线性计费。

计算

计算资源包括 CPU 资源和内存资源，计算资源的选择决定了实例的最大 QPS 和访问连接数等参数，价格请参见 产品价格。

购买指引

购买方式

1. 登录 MongoDB 购买页，根据实际需求选择各项配置信息，确认无误后，单击【立即购买】。

- 计费模式：按量计费。
- 地域和可用区：选择地域和可用区。
- 版本：3.6版本。
- 实例类型与节点数：实例类型包括副本集和分片集群。

选择副本集时，系统默认配置1主2从的架构。

选择分片集群时，可自由选择分片的数量和每片中的节点数，系统默认每片是1主2从的架构，为提高数据可用性，用户可增加每片的节点数，为提高集群的可存储量，用户可增加分片的数量。

- 规格：可选的计算规格参见 计费概述。
- 容量：用户可根据不同的计算规格来选择相应的存储规格。其中，系统默认设定 oplog 的存储空间为所选存储容量的10%，oplog 的大小可在控制台自由调整。
- 网络类型：支持私有网络（推荐）。选择某个私有网络时，仅当前子网下的设备才能访问所创建的数据库实例；
- 购买数量：选择需要购买的数量。

2. 购买成功后，返回实例列表，待实例状态变为“运行中”，即可正常使用。

查看账单

操作步骤

1. 登录控制台，单击右上角【费用】，进入【计费管理】页面。
2. 在左侧导航栏中，选择【账单管理】>【账单明细】>【资源 ID 账单】。

快速入门
初始化实例

操作场景

本文为您介绍通过云数据库 MongoDB 控制台初始化实例的操作。

操作步骤

1. 登录 MongoDB 控制台，在实例列表选择状态为“待初始化”的实例，单击【初始化密码】。

2. 在弹出的对话框，设置用户密码，单击【确认】。
3. 返回实例列表，待实例状态变为“运行中”即可正常使用。

访问连接

实例初始化后，可以通过 MongoDB shell 或者各语言驱动访问数据库，并进行各种管理操作。
使用 云服务器 CVM 连接自动分配给云数据库的内网地址，这种连接方式使用内网高速网络，延迟低。云服务器和数据库须是同一账号，且同一个 VPC 内（保障同一个地域），或同在基础网络内。暂不支持外网访问方式。

前提条件

连接云数据库 MongoDB 最低驱动版本需要3.2，建议使用最新版的客户端驱动以保证最好的兼容性，包括 Shell 套件、Java jar 包、PHP 扩展、Node.js 模块等，具体请参见 MongoDB 官网驱动介绍。

连接方式

shell 方式

mongo shell 是 MongoDB 自带的一种交互式 JavaScript shell，可在 shell 中使用命令行与 MongoDB 实例交互。您可以使用 mongo shell 查询、更新数据，及执行管理操作。
mongo shell 是 MongoDB 发行版的一部分，您需要先下载和安装 MongoDB，再使用 mongo shell 连接您的云数据库 MongoDB。MongoDB 发行版下载请参见 下载地址，具体连接步骤如下：

```
cd <mongodb installation dir>
./bin/mongo -umongouser -plxh2081* 172.x.x.56:27017/admin
```

- 上例中:
- -u 参数指定 [用户名]
- -p 参数指定密码
- 172.x.x.56和27017分别指定 MongoDB 实例的 IP 和端口。

URI 方式

MongoDB 既可以用传统的传参方式进行连接，同时大部分的驱动程序也支持 URI 形式连接。MongoDB 官方推荐使用 URI 的方式连接 MongoDB。

> MongoDB 副本集实例（4.0版）连接方式与其他版本规格有所不同，4.0版提供了3个 IP 进行访问，分别对应副本集的3个节点。现网业务连接时，建议在连接串中配置3个 IP，连接更加安全高效。请参见 副本集实例（4.0版）连接说明。

典型的 URI 如下：

- 例1

mongodb://username:password@IP:27017/admin

- 例2

  mongodb://username:password@IP:27017/somedb?authSource=admin

- 例3

  mongodb://username:password@IP:27017/somedb?authSource=admin&readPreference=secondaryPreferred

URI 组成的各部分解释如下：

组成部分	含义	是否必须
mongodb://	一个特定的字符串，表示 MognoDB 协议	必须
username	用于登录 MongoDB 的用户名	必须，参见本页 默认用户
password	用于登录 MongoDB 的用户密码	必须
IP:27017	MongoDB 的 IP 和端口	必须
/admin	要认证的数据库，云数据库 MongoDB 固定为 admin	必须，参见本页 认证数据库
authMechanism=MONGODB-CR	认证机制	参见本页 认证机制
authSource=admin	身份认证所用库，云数据库 MongoDB 固定为 admin	必须，参见本页 认证数据库
readPreference=secondaryPreferred	可以设置优先读从库	非必须，参见本页 读操作的主从优先级

此处仅列举了一部分 MongoDB 连接 URI 的参数，更多内容请参见 MongoDB 官网文档。

默认用户

云数据库 MongoDB 默认用户因版本而异，对于最新的实例，我们内建了 rwuser 和 mongouser 两个默认用户。旧实例只有 rwuser，旧实例我们会进行升级，升级之前会联系您。您可以通过 MongoDB 控制台 的数据库管理页查看用户账号，以及管理权限以满足业务需求。

- rwuser（MONGODB-CR 认证）URI 示例
  rwuser 是唯一使用 MONGODB-CR 认证的用户：

  mongodb://rwuser:password@10.66.100.186:27017/admin?authMechanism=MONGODB-CR
  或者
  mongodb://rwuser:password@10.66.100.186:27017/somedb?authMechanism=MONGODB-CR&authSource=admin

- mongouser（SCRAM-SHA-1 认证）URI 示例

- mongouser 以及在 MongoDB 控制台 创建的用户均使用 SCRAM-SHA-1 认证：

  mongodb://mongouser:password@10.66.100.186:27017/admin
  或者
  mongodb://mongouser:password@10.66.100.186:27017/somedb?authSource=admin

认证数据库

云数据库 MongoDB 统一使用 admin 库作为登录鉴权的认证数据库，所以在 URI 中端口后面必须加上“/admin”以指定认证库，通过认证后再切换到具体业务数据库进行读写操作，URI 示例：

mongodb://username:password@IP:27017/admin

当然，也可通过直接指定读写目标数据库和额外的认证库参数（authSource=admin）来直达目标数据库，URI 示例：

mongodb://username:password@IP:27017/somedb?authSource=admin

综上，您必须选择一种方式将 admin 作为认证库代入 URI 中。

认证机制

MongoDB 支持多种认证机制，目前官方推荐 SCRAM-SHA-1。
云数据库 MongoDB 支持 MONGODB-CR 和 SCRAM-SHA-1 两种认证方式。
云数据库 MongoDB 内建了两个默认用户 rwuser 和 mongouser，同时还可在 MongoDB 控制台 创建其他用户，这些用户被分成了两类，分别采用不同的认证机制，分类如下：

用户名	认证机制	URI 处理
rwuser	MONGODB-CR	必须加上参数 “authMechanism=MONGODB-CR”

用户名	认证机制	URI 处理
mongouser 以及在控制台创建的用户	SCRAM-SHA-1（推荐）	不用加任何参数

读操作的主从优先级

云数据库 MongoDB 提供了一个负载均衡 IP 用于访问整个副本集，如需指定访问从库读，请在 URI 里设置 readPreference 参数，具体取值含义如下：

取值	含义	是否默认
primary	只读主节点	是
primaryPreferred	主节点优先，如主节点不可用，则读从节点	否
secondary	只读从节点，如从节点不可用会报错	否
secondaryPreferred	从节点优先，如从节点不可用，则读主节点	否

设置优先读取从节点可以按示例拼接 URI：

```
mongodb://username:password@IP:27017/admin?readPreference=secondaryPreferred
```

连接示例

shell 方式

- Shell 连接示例

URI 方式

- PHP 连接示例
- Node.js 连接示例
- mongoose 连接示例
- Java 连接示例
- Python 连接示例
- 重连机制

操作指南
使用限制

引擎

目前支持WiredTiger 引擎。

副本集和分片集群

创建云数据库 MongoDB 实例时有以下两种选择：

节点配置	说明
副本集	1主2从，包含1个 Primary 节点、 2个 Secondary 节点
分片集群	单个实例至少2个片，每个片至少3个节点，可扩展（当前分片节点的规格支持扩展，分片数和节点数暂不支持）

连接数限制

内存规格	连接数上限
2GB	1500
4GB	2000
6GB	2500
8GB	3500
16GB	6000
24GB	8000
32GB	9500
64GB	16000
128GB	18000
240GB	18000

内存规格	连接数上限
512GB	20000

> 对于副本集来说，连接数上限是针对整个实例的，不是单个节点；对于分片集群来说，连接数上限是针对单个shard。

连接用户名

云数据库 MongoDB 内建了 默认用户 mongouser，mongouser 采用 SCRAM-SHA-1 认证方式，角色为 readWriteAnyDatabase+dbAdmin，也就是说您可以用此用户读写任意数据库，但是不具备高危操作的权限。

3.2版本的实例支持另外一个内建用户 rwuser，采用 MONGODB-CR 认证方式，该认证方式已被官方废弃，建议您优先使用 mongouser 连接数据库。

您也可以使用 MongoDB 控制台 进行账号和权限管理以满足您的业务需要。

避免写满磁盘

实例磁盘被写满100%后，会被禁止执行写操作，请根据业务情况及时调整实例配置，调整实例配置请参见 调整实例规格。
为避免磁盘写满影响数据写入，请及时关注告警信息，及时实施扩容实例。

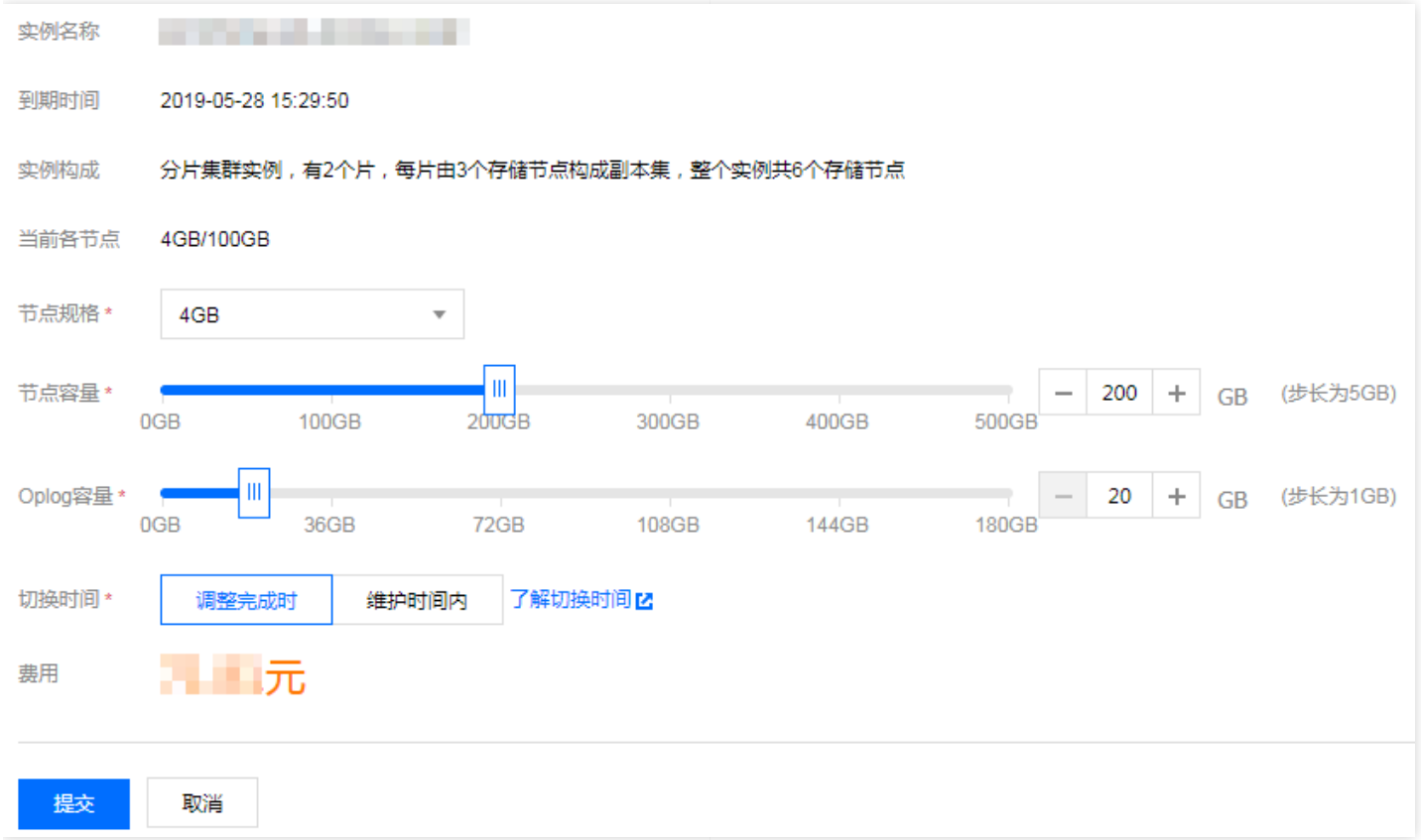
创建实例
创建分片集群

创建分片实例

1. 登录 MongoDB 控制台，在左侧选择【分片实例】页。
2. 单击【新建】，进入 购买页。
3. 实例类型选择【分片集群】，按需选择分片的片数，片内节点数，以及节点规格。每个分片都是多节点的副本集，片内多节点自动容灾，保证服务高可用。
4. 确认无误后，单击【立即购买】。

分片集群扩容

目前云数据库 MongoDB 分片集群的扩容方式只支持将所有节点进行统一扩容，暂不支持通过添加节点的方式进行扩容。
在实例列表页单击【配置调整】，选择需要扩到的容量规格，单击【提交】。



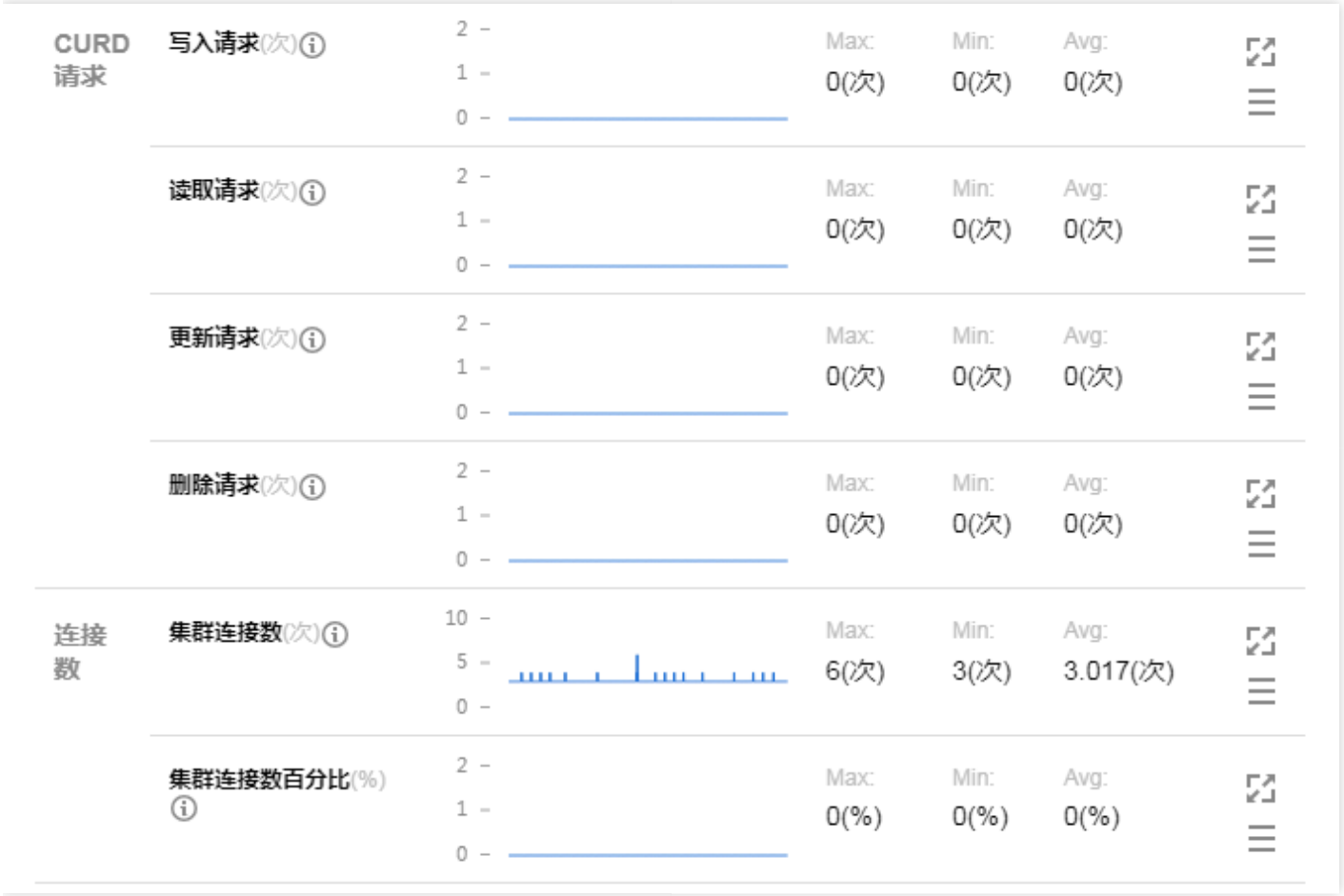
备份和回档

分片集群实例的备份回档和副本集实例的备份回档操作相同，目前只支持实例级别的备份和回档。
在实例列表单击实例名，进入管理页，选择【备份与回档】页。

在回档操作过程中，需要输入需要回档到的日期，目前支持6日内的任意时间回档，但前提是只能选择两次备份（成功且非 oplog 写满状态）之间的时间点进行回档，如果没有满足的备份请执行一次手动备份。

集群实例监控

云数据库 MongoDB 分片集群实例提供三个维度的监控指标，分别是实例维度，片维度以及节点维度来进行整个集群的数据监控。提供操作请求，容量使用，负载等多项指标的监控数据。
在实例列表单击实例名，进入管理页，选择【系统监控】页进行查询。


CURD请求
写入请求(次)
2
1
0
Max:
0(次)
Min:
0(次)
Avg:
0(次)
读取请求(次)
2
1
0
Max:
0(次)
Min:
0(次)
Avg:
0(次)
更新请求(次)
2
1
0
Max:
0(次)
Min:
0(次)
Avg:
0(次)
删除请求(次)
2
1
0
Max:
0(次)
Min:
0(次)
Avg:
0(次)
连接数
集群连接数(次)
10
5
0
Max:
6(次)
Min:
3(次)
Avg:
3.017(次)
集群连接数百分比(%)
2
1
0
Max:
0(%)
Min:
0(%)
Avg:
0(%)

初始化实例

操作场景

本文为您介绍通过云数据库 MongoDB 控制台初始化实例的操作。

操作步骤

1. 登录 MongoDB 控制台，在实例列表选择状态为“待初始化”的实例，单击【初始化密码】。

2. 在弹出的对话框，设置用户密码，单击【确认】。
3. 返回实例列表，待实例状态变为“运行中”即可正常使用。

访问连接

实例初始化后，可以通过 MongoDB shell 或者各语言驱动访问数据库，并进行各种管理操作。
使用 云服务器 CVM 连接自动分配给云数据库的内网地址，这种连接方式使用内网高速网络，延迟低。云服务器和数据库须是同一账号，且同一个 VPC 内（保障同一个地域），或同在基础网络内。暂不支持外网访问方式。

前提条件

连接云数据库 MongoDB 最低驱动版本需要3.2，建议使用最新版的客户端驱动以保证最好的兼容性，包括 Shell 套件、Java jar 包、PHP 扩展、Node.js 模块等，具体请参见 MongoDB 官网驱动介绍。

连接方式

shell 方式

mongo shell 是 MongoDB 自带的一种交互式 JavaScript shell，可在 shell 中使用命令行与 MongoDB 实例交互。您可以使用 mongo shell 查询、更新数据，及执行管理操作。
mongo shell 是 MongoDB 发行版的一部分，您需要先下载和安装 MongoDB，再使用 mongo shell 连接您的云数据库 MongoDB。MongoDB 发行版下载请参见 下载地址，具体连接步骤如下：

```
cd <mongodb installation dir>
./bin/mongo -umongouser -plxh2081* 172.x.x.56:27017/admin
```

> ?上例中，-u 参数指定 用户名，-p 参数指定密码，172.x.x.56和27017分别指定 MongoDB 实例的 IP 和端口。

URI 方式

MongoDB 既可以用传统的传参方式进行连接，同时大部分的驱动程序也支持 URI 形式连接。MongoDB 官方推荐使用 URI 的方式连接 MongoDB。

> ?MongoDB 副本集实例（4.0版）连接方式与其他版本规格有所不同，4.0版提供了3个 IP 进行访问，分别对应副本集的3个节点。现网业务连接时，建议在连接串中配置3个 IP，连接更加安全高效。请参见 副本集实例（4.0版）连接说明。

典型的 URI 如下：

- 例1

```
mongodb://username:password@IP:27017/admin
```

- 例2

  mongodb://username:password@IP:27017/somedb?authSource=admin

- 例3

  mongodb://username:password@IP:27017/somedb?authSource=admin&readPreference=secondaryPreferred

URI 组成的各部分解释如下：

组成部分	含义	是否必须
mongodb://	一个特定的字符串，表示 MognoDB 协议	必须
username	用于登录 MongoDB 的用户名	必须，参见本页 默认用户
password	用于登录 MongoDB 的用户密码	必须
IP:27017	MongoDB 的 IP 和端口	必须
/admin	要认证的数据库，云数据库 MongoDB 固定为 admin	必须，参见本页 认证数据库
authMechanism=MONGODB-CR	认证机制	参见本页 认证机制
authSource=admin	身份认证所用库，云数据库 MongoDB 固定为 admin	必须，参见本页 认证数据库
readPreference=secondaryPreferred	可以设置优先读从库	非必须，参见本页 读操作的主从优先级

此处仅列举了一部分 MongoDB 连接 URI 的参数，更多内容请参见 MongoDB 官网文档。

默认用户

云数据库 MongoDB 默认用户因版本而异，对于最新的实例，我们内建了 rwuser 和 mongouser 两个默认用户。旧实例只有 rwuser，旧实例我们会进行升级，升级之前会联系您。您可以通过 MongoDB 控制台 的数据库管理页查看用户账号，以及管理权限以满足业务需求。

- rwuser（MONGODB-CR 认证）URI 示例
  rwuser 是唯一使用 MONGODB-CR 认证的用户：

mongodb://rwuser:password@10.66.100.186:27017/admin?authMechanism=MONGODB-CR
或者
mongodb://rwuser:password@10.66.100.186:27017/somedb?authMechanism=MONGODB-CR&authSource=admin

- mongouser（SCRAM-SHA-1 认证）URI 示例
  mongouser 以及在 MongoDB 控制台 创建的用户均使用 SCRAM-SHA-1 认证：

  mongodb://mongouser:password@10.66.100.186:27017/admin
  或者
  mongodb://mongouser:password@10.66.100.186:27017/somedb?authSource=admin

认证数据库

云数据库 MongoDB 统一使用 admin 库作为登录鉴权的认证数据库，所以在 URI 中端口后面必须加上"/admin"以指定认证库，通过认证后再切换到具体业务数据库进行读写操作，URI 示例：

mongodb://username:password@IP:27017/admin

当然，也可通过直接指定读写目标数据库和额外的认证库参数（authSource=admin）来直达目标数据库，URI 示例：

mongodb://username:password@IP:27017/somedb?authSource=admin

综上，您必须选择一种方式将 admin 作为认证库代入 URI 中。

认证机制

MongoDB 支持多种认证机制，目前官方推荐 SCRAM-SHA-1。
云数据库 MongoDB 支持 MONGODB-CR 和 SCRAM-SHA-1 两种认证方式。
云数据库 MongoDB 内建了两个默认用户 rwuser 和 mongouser，同时还可在 MongoDB 控制台 创建其他用户，这些用户被分成了两类，分别采用不同的认证机制，分类如下：

用户名	认证机制	URI 处理
rwuser	MONGODB-CR	必须加上参数 "authMechanism=MONGODB-CR"
mongouser 以及在控制台创建的用户	SCRAM-SHA-1（推荐）	不用加任何参数

读操作的主从优先级

云数据库 MongoDB 提供了一个负载均衡 IP 用于访问整个副本集，如需指定访问从库读，请在 URI 里设置

readPreference 参数，具体取值含义如下：

取值	含义	是否默认
primary	只读主节点	是
primaryPreferred	主节点优先，如主节点不可用，则读从节点	否
secondary	只读从节点，如从节点不可用会报错	否
secondaryPreferred	从节点优先，如从节点不可用，则读主节点	否

设置优先读取从节点可以按示例拼接 URI：

```
mongodb://username:password@IP:27017/admin?readPreference=secondaryPreferred
```

连接示例

shell 方式

- Shell 连接示例

URI 方式

- PHP 连接示例
- Node.js 连接示例
- mongoose 连接示例
- Java 连接示例
- Python 连接示例
- 重连机制

实例生命周期
连接实例
Shell连接示例

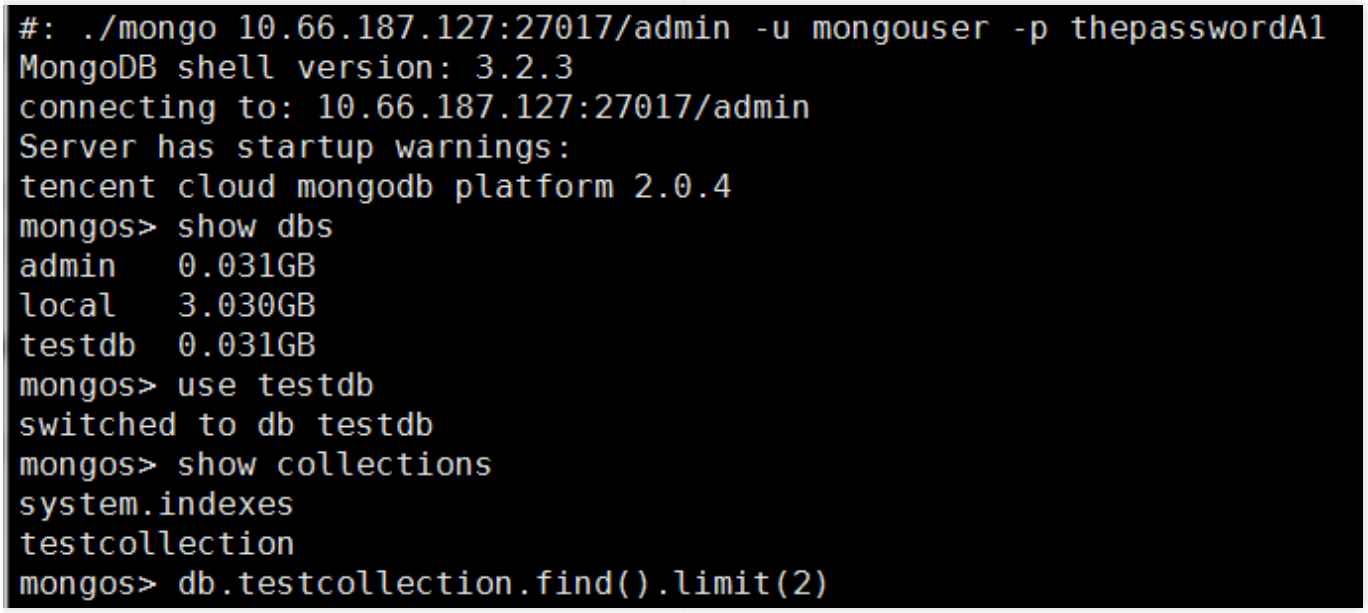
在云服务器 CVM 中可用 MongoDB 提供的 shell 客户端（查看安装文档）连接云数据库 MongoDB 进行数据管理，请注意使用最新版本的 MongoDB 客户端套件。

快速开始

典型的连接命令如下：

```
mongo 10.66.187.127:27017/admin -u mongouser -p thepasswordA1
```

如图：



多种认证方式的连接说明

在 连接实例 有说明，云数据库 MongoDB 默认提供 rwuser 和 mongouser 两个用户名，分别支持 MONGODB-CR 和 SCRAM-SHA-1 两种认证方式。
对于这两种认证方式，shell 的参数是不一样的，具体请看下文。

SCRAM-SHA-1 认证（mongouser）

默认用户 mongouser 以及在控制台创建的所有新用户都使用 SCRAM-SHA-1 认证，其 shell 连接参数与 快速开始 章节完全一样，无需添加额外参数，示例如下：

mongo 10.66.187.127:27017/admin -u mongouser -p thepasswordA1

如果您希望连接 MongoDB 服务后直接进入到某一个 db，例如 singer，请按示例操作：

mongo 10.66.187.127:27017/singer -u mongouser -p thepasswordA1 --authenticationDatabase admin

如图：

```
#: ./mongo 10.66.187.127:27017/singer -u mongouser -p thepasswordA1 --authenticationDatabase admin
MongoDB shell version: 3.2.3
connecting to: 10.66.187.127:27017/singer
Server has startup warnings:
tencent cloud mongodb platform 2.0.4
mongos> db
singer
mongos>
```

MONGODB-CR 认证（rwuser）

请注意，只有默认用户 rwuser 使用 MONGODB-CR 认证，其 shell 连接参数需要指明认证方式为 MONGODB-CR，示例如下：

mongo 10.66.187.127:27017/admin -u rwuser -p thepasswordA1 --authenticationMechanism=MONGODB-CR

如图：

```
#: ./mongo 10.66.187.127:27017/admin -u rwuser -p thepasswordA1 --authenticationMechanism=MONGODB-CR
MongoDB shell version: 3.2.3
connecting to: 10.66.187.127:27017/admin
Server has startup warnings:
tencent cloud mongodb platform 2.0.4
mongos> show dbs
admin   0.031GB
local   3.030GB
testdb  0.031GB
mongos>
```

如果您希望连接 MongoDB 服务后直接进入到某一个 db，例如 singer，请按示例操作：

mongo 10.66.187.127:27017/singer -u rwuser -p thepasswordA1 --authenticationMechanism=MONGODB-CR --authenticationDatabase admin

如图：

```
#: ./mongo 10.66.187.127:27017/singer -u rwuser -p thepasswordA1 --authenticationMechanism=MONGODB-CR --
authenticationDatabase admin
MongoDB shell version: 3.2.3
connecting to: 10.66.187.127:27017/singer
Server has startup warnings:
tencent cloud mongodb platform 2.0.4
mongos> db
singer
mongos>
```

使用 shell 进行数据导入和导出

上文所述的两种认证方式都可以在 shell 里进行数据导入和导出，请参见 导出导入。

PHP连接示例

相关说明

云数据库 MongoDB 默认提供 rwuser 和 mongouser 两个用户名，分别支持 MONGODB-CR 和 SCRAM-SHA-1 两种认证方式，对于这两种认证方式，连接 URI 需要做不同的处理，具体参见 连接实例。

在 PHP 里，有 两套驱动 可用于连接操作 MongoDB 数据库，它们分别是：

- mongodb（PHP 官网文档）- MongoDB 官方推荐 mongodb 驱动，但需要 PHP 5.4 以上版本。
- mongo（PHP 官网文档）- mongo 比较旧，但也可以用，如果要用请选择 1.6 版本。
  下面分别用上述两个驱动演示连接云数据库 MongoDB 并进行读写。

使用 mongodb 驱动

mongodb 安装方法参考 官方安装步骤。
mongodb 驱动可以用 MONGODB-CR 和 SCRAM-SHA-1 两种认证方式，具体参见 连接实例。

示例代码:

```
<?php
// 拼接连接 URI
$uri = 'mongodb://mongouser:thepasswordA1@10.66.187.127:27017/admin';
$manager = new MongoDB\Driver\Manager($uri);

// 准备写入数据
$document1 = [
    'username' => 'lily',
    'age'      => 34,
    'email'    => 'lily@163.com'
];

// 驱动预处理数据，这里可以看到 MongoDB 的 _id 是驱动生成的
$bulk = new MongoDB\Driver\BulkWrite;
$_id1 = $bulk->insert($document1);

$result = $manager->executeBulkWrite('tsdb.table1', $bulk);

// 或者根据实际需要使用下面的代码确保数据写入到大多数节点
// $writeConcern = new MongoDB\Driver\WriteConcern(MongoDB\Driver\WriteConcern::MAJORITY, 1
000);
// $result = $manager->executeBulkWrite('testdb.testcollection', $bulk, $writeConcern);

// 查询
$filter = ['_id' => $_id1];
$query = new MongoDB\Driver\Query($filter);
```

```
$rows = $manager->executeQuery('tsdb.table1', $query); // 也可选择优先从从库读
foreach($rows as $r){
  print_r($r);
}
```

输出：

```
stdClass Object
(
    [_id] => MongoDB\BSON\ObjectID Object
        (
            [oid] => 582c001618c90a16363abc31
        )

    [username] => lily
    [age] => 34
    [email] => lily@163.com
)
```

使用 mongo 驱动

mongo 驱动只支持 MONGODB-CR 认证，对应的只能用 rwuser 进行连接，具体参见 连接实例。

示例代码:

```
<?php
// 推荐使用 URI 的方式连接，两种 URI 任选其一
$uri = "mongodb://rwuser:thepasswordA1@10.66.187.127:27017/admin?authMechanism=MONGOD
B-CR";
$uri = "mongodb://rwuser:thepasswordA1@10.66.187.127:27017/?authMechanism=MONGODB-CR&a
uthSource=admin";
$connection = new MongoClient($uri);

/*
// 或者这样也可以
$connection = new MongoClient("mongodb://10.66.116.103:27017/admin",
    array(
        "username" => "rwuser",
        "password" => "password",
        "authMechanism" => "MONGODB-CR"
    )
);
*/
$db = $connection->tsdb;
$collection = $db->table1;
```

```
$q = array(
    'id' => 1,
    'test1' => 'xxx',
    'ss' => 'xxxxxxxx',
);
$collection->save($q);
$one = $collection->findOne();
var_dump($one);
```

推荐使用 PHPLIB 库（基于 mongodb 驱动封装）

使用 mongodb 驱动推荐搭配 PHPLIB 使用，查看相关文档。

PHPLIB 的安装方法参考 官方安装步骤，请注意 PHPLIB 依赖与 mongodb 驱动。

示例代码:

```
<?php
require_once __DIR__ . "/vendor/autoload.php";

// 初始化
$mongoClient = new MongoDB\Client('mongodb://mongouser:thepasswordA1@10.66.187.127:2701
7/admin');

// 使用 demo 库下的 users 集合
$collection = $mongoClient->demo->users;

// 写入一条数据
$insertOneResult = $collection->insertOne(['name' => 'gomez']);

printf("Inserted %d document(s)\n", $insertOneResult->getInsertedCount());
var_dump($insertOneResult->getInsertedId());

// 查询数据
$document = $collection->findOne(['name' => 'gomez']);

var_dump($document);
```

输出：

```
Inserted 1 document(s)
object(MongoDB\BSON\ObjectID)#11 (1) {
  ["oid"]=>
  string(24) "57e3bf20bf605714a53e69c1"
}
object(MongoDB\Model\BSONDocument)#16 (1) {
  ["storage":"ArrayObject":private]=>
```

```
  array(2) {
    ["_id"]=>
    object(MongoDB\BSON\ObjectID)#14 (1) {
      ["oid"]=>
      string(24) "57e3bf20bf605714a53e69c1"
    }
    ["name"]=>
    string(5) "gomez"
  }
}
```

Node.js连接示例

相关说明

云数据库 MongoDB 默认提供 rwuser 和 mongouser 两个用户名，分别支持 MONGODB-CR 和 SCRAM-SHA-1 两种认证方式，对于这两种认证方式，连接 URI 需要做不同的处理，具体参见 连接实例。

Node.js MongoDB 驱动文档

快速开始

Node.js 原生示例代码

Shell 安装驱动包：

```
npm install mongodb --save
（如遇安装不成功可以尝试更换源，npm config set registry http://registry.cnpmjs.org）
npm init
```

程序代码：

```
'use strict';

var mongoClient = require('mongodb').MongoClient,
    assert = require('assert');

// 拼接 URI
var url = 'mongodb://mongouser:thepasswordA1@10.66.161.177:27017/admin';

mongoClient.connect(url, function(err, db) {
    assert.equal(null, err);
    var db = db.db('testdb'); // 选择一个 db
    var col = db.collection('demoCol'); // 选择一个集合(表)
   // 插入数据
    col.insertOne(
        {
            a: 1,
            something: "yy"
        },
        //可选参数
        //{
        //    w: 'majority' // 开启 “大多数”模式，保证数据写入 Secondary 节点
```

```
        //},
        function(err, r) {
            console.info("err:", err);
            assert.equal(null, err);
            // 断言写入成功
            assert.equal(1, r.insertedCount);
            // 查询数据
            col.find().toArray(function(err, docs) {
                assert.equal(null, err);
                console.info("docs:", docs);
                db.close();
            });
        }
    );
});
```

输出：

```
[root@VM_2_167_centos node]# node index.js
docs: [ { _id: 567a1bf26773935b3ff0b42a, a: 1, something: 'yy' } ]
```

Node.js mongoose 连接示例

```
var dbUri = "mongodb://" + user + ":" + password + "@" + host + ":" + port + "/" + dbName;
var opts = {
    auth: {
        authMechanism: 'MONGODB-CR', // 如果使用 SCRAM-SHA-1 认证则不需要此参数
        authSource: 'admin'
    }
};
var connection = mongoose.createConnection(dbUri, opts);
```

Java连接示例

相关说明

云数据库 MongoDB 默认提供 rwuser 和 mongouser 两个用户名，分别支持 MONGODB-CR 和 SCRAM-SHA-1 两种认证方式，对于这两种认证方式，连接 URI 需要做不同的处理，具体参见 连接实例。

Java MongoDB 驱动文档

Java Jar 包下载，请选择3.2以上版本

快速开始

原生 Java 示例代码

```
package mongodbdemo;

import org.bson.*;
import com.mongodb.*;
import com.mongodb.client.*;

public class MongodbDemo {

    public static void main(String[] args) {
        String mongoUri = "mongodb://mongouser:thepasswordA1@10.66.187.127:27017/admin";
        MongoClientURI connStr = new MongoClientURI(mongoUri);
        MongoClient mongoClient = new MongoClient(connStr);
        try {
            // 使用名为 someonedb 的数据库
            MongoDatabase database = mongoClient.getDatabase("someonedb");
            // 取得集合/表 someonetable 句柄
            MongoCollection<Document> collection = database.getCollection("someonetable");

            // 准备写入数据
            Document doc = new Document();
            doc.append("key", "value");
            doc.append("username", "jack");
            doc.append("age", 31);

            // 写入数据
            collection.insertOne(doc);
            System.out.println("insert document: " + doc);
```

```
            // 读取数据
            BsonDocument filter = new BsonDocument();
            filter.append("username", new BsonString("jack"));
            MongoCursor<Document> cursor = collection.find(filter).iterator();
            while (cursor.hasNext()) {
                System.out.println("find document: " + cursor.next());
            }
        } finally {
            //关闭连接
            mongoClient.close();
        }
    }
}
```

输出：

```
INFO: Opened connection [connectionId{localValue:2, serverValue:67621}] to 10.66.122.28:27017
insert document: Document{{key=value, username=jack, age=31, _id=56a6ebb565b33b771f9826dd}}
find document: Document{{_id=56a3189565b33b2e7ca150ba, key=value, username=jack, age=31}}
Jan 26, 2016 11:44:53 AM com.mongodb.diagnostics.logging.JULLogger log
INFO: Closed connection [connectionId{localValue:2, serverValue:67621}] to 10.66.122.28:27017 becaus
e the pool has been closed.
```

Spring Data MongoDB 配置示例

本示例主要为了体现出 认证库 admin 的配置方法，具体还请参考您使用的 Spring 和 Spring Data MongoDB 的版本而定。

```
<bean id="mongoTemplate" class="org.springframework.data.mongodb.core.MongoTemplate">
    <constructor-arg name="mongoDbFactory" ref="mongoDbFactory" />
</bean>
<bean id="mongoDbFactory" class="org.springframework.data.mongodb.core.SimpleMongoDbFactor
y">
    <constructor-arg name="mongo" ref="mongo" />
    <constructor-arg name="databaseName" value="您的目标库" />
    <constructor-arg name="credentials" ref="userCredentials" />
    <constructor-arg name="authenticationDatabaseName" value="admin" />
</bean>
<bean id="userCredentials" class="org.springframework.data.authentication.UserCredentials">
    <constructor-arg name="username" value="用户名" />
    <constructor-arg name="password" value="密码" />
</bean>
```

Python连接示例

相关说明

云数据库 MongoDB 默认提供 rwuser 和 mongouser 两个用户名，分别支持 MONGODB-CR 和 SCRAM-SHA-1 两种认证方式，对于这两种认证方式，连接 URI 需要做不同的处理，具体参见 连接实例。

Python 驱动下载

快速开始

Python 示例代码 1

```
#!/usr/bin/python
import pymongo
import random

mongodbUri = 'mongodb://mongouser:thepasswordA1@10.66.187.127:27017/admin'

client = pymongo.MongoClient(mongodbUri)
db = client.somedb
db.user.drop()
element_num=10
for id in range(element_num):
    name = random.choice(['R9','cat','owen','lee','J'])
    sex = random.choice(['male','female'])
    db.user.insert_one({'id':id, 'name':name, 'sex':sex})

content = db.user.find()
for i in content:
    print i
```

Python 示例代码 2

```
#!/usr/bin/python
import pymongo
mongodbUri = 'mongodb://mongouser:thepasswordA1@10.66.187.127:27017/admin'
client = pymongo.MongoClient(mongodbUri)
db = client.someonedb

inserted_id = db.somecoll.insert_one({"somekey":"yiqihapi"}).inserted_id
```

```
print inserted_id

for doc in db.somecoll.find(dict(_id=inserted_id)):
    print doc

for doc in db.somecoll.find({"somekey":"yiqihapi"}):
    print doc
```

输出：

```
5734431e101e2f6d699b37ef
{u'somekey': u'yiqihapi', u'_id': ObjectId('5734431e101e2f6d699b37ef')}
{u'somekey': u'yiqihapi', u'_id': ObjectId('5734431e101e2f6d699b37ef')}
```

PHP重连示例

说明

云数据库 MongoDB 提供的不是简单的 mongod 访问，给到用户访问的是一个负载均衡 IP，此 IP 后面是连接到一系列类似 mongos 一样存在的路由接入层。
客户端驱动会透过负载均衡 IP 与接入机建立一个长连接，当此连接处于长期间活跃状态时，我们不会对其做任何干预，但是当长连接闲置时间超过1天时（此时间会随着版本优化而调整），路由接入层会踢掉该连接。
一般来说，客户端驱动会实现一个自动重连的过程，但是也有部分语言的驱动并没有实现。对于没有实现自动重连的语言驱动，当用户使用一个已经被踢掉的连接来尝试与云数据库 MongoDB 通信时可能会得到 “Remote server has closed the connection” 之类的错误信息，所以需手动进行重连，本文给出一个 PHP 重连的 demo。

基于 php mongo 驱动的重连实现

```
<?php

function getConnection() {
    $connection = false;
    $uri = 'mongodb://rwuser:1234567a@10.66.148.142:27017/admin?authMechanism=MONGODB-C
R';
    $maxRetries = 5;
    for( $counts = 1; $counts <= $maxRetries; $counts++ ) {
        try {
            $connection = new MongoClient($uri);
        } catch( Exception $e ) {
     // 或者 根据需要使用下面的catch代码行，注意那一个“\”，某些框架使用命名空间时需要用到。
     // } catch( \Exception $e ) {
            continue;
        }
        break;
    }
    return $connection;
}

$connection = getConnection();

if($connection) {
        $db = $connection->testdb;
        $collection = $db->testcollection;

        $one = $collection->findOne();
```

```
    var_dump($one);
}
```

维护实例
重启实例

操作场景

由云数据库 MongoDB 的架构所决定，重启实例分为重启 Mongos 和重启 Mongod 两部分。由于重启实例会造成连接闪断，特别是重启 Mongod 时，如果有数据写入可能会造成 rollback，进而丢失数据。
重启期间，云数据库 MongoDB 实例将无法提供正常服务，请提前做好准备，以免对业务造成影响。目前，重启 Mongod 处于白名单控制，需 提交工单 联系我们。重启属高危操作，请谨慎处理。

操作步骤

1. 登录 MongoDB 控制台，进入副本集或者分片实例列表。

2. 勾选需要重启的实例，在上方单击【重启】，或在实例的操作列选择【更多】>【重启】。
3. 在弹出的对话框，勾选需要重启的组件，单击【确认】，等待任务完成即可。

为实例指定项目

云数据库 MongoDB 支持将实例分配至不同的项目进行管理。

需要注意的几点特性是：

- 数据库实例在项目间进行分配和移动，不会影响实例对外提供的服务。
- 用户须在新购实例时为实例指定所属的项目，缺省为【默认项目】。
- 已指定项目的实例可通过 控制台 的【分配至项目】功能重新指定到其他项目。

设置实例维护时间

操作场景

维护时间对于云数据库 MongoDB 而言是非常重要的概念，为保证您的云数据库 MongoDB 实例的稳定性，后台系统会不定期在维护时间内对实例进行维护操作。建议您对业务实例设置自己可接受的维护时间，一般设置在业务低峰期，将对业务的影响降到最低。

操作步骤

1. 登录 MongoDB 控制台，在实例列表，找到需要修改的实例，单击实例名或操作列的【更多】>【管理】。
2. 在实例详情页的"维护时间"处，单击【修改】。

3. 在弹出的对话框，选择维护时间后单击【确认】即可完成实例维护时间设置。

变更实例
调整实例规格

云数据库 MongoDB 支持快捷调整实例的规格，提供灵活的扩缩容操作。用户可根据其业务所处的实际情况（业务初期、业务快速发展期、业务高峰期、业务低谷期等）灵活的调整其 MongoDB 实例的规格，从而更好满足用户的资源充分利用和成本实时优化等需求。

配置规则

1. 云数据库 MongoDB 实例及其所关联的实例处于正常状态下（运行中）并且当前没有任务执行时才能够发起调整配置操作。
2. 调整配置过程中，不允许取消本次调整配置操作。
3. 调整配置前后实例的名称、访问 IP、访问端口均不发生变化。
4. 调整配置过程中，可能会涉及到数据的搬迁，期间云数据库 MongoDB 实例可正常访问，业务不受影响。
5. 调整配置完毕时可能会涉及实例切换，建议程序有自动重连功能，并且强烈建议选择实例可维护时间内做切换。有关维护时间请参见 设置实例维护时间。

调整配置

1. 登录 MongoDB 控制台，在实例列表，选择对应地域。
2. 在列表选择需要调整的实例，在“操作”列，单击【配置调整】。
3. 在弹出页，选择调整后的配置，单击【提交】。

销毁实例

操作场景

根据业务需求，您可以在控制台自助退还按量计费和包年包月实例。

- 包年包月实例退还后，实例被移入云数据库回收站保留7天，期间实例无法访问。如您想恢复该实例，可在回收站进行续费恢复。
- 按量计费实例退还后，会直接销毁，无保留期，请谨慎操作。
  自助退还后，实例的状态一旦变为"已隔离"时，就不再产生与该实例相关的费用。
  > - 实例销毁后数据将无法找回，请提前备份实例数据。
  > - 实例销毁后 IP 资源同时释放，如果该实例有相关的只读或灾备实例：
  > - 只读实例将同时被销毁。
  > - 灾备实例将会断开同步连接，自动升级为主实例。
  > - 实例销毁后，退款处理：
  > - 5天无理由自助退还的金额将退还至TCloudFinanceZone账户。
  > - 普通自助退还的金额将按购买支付使用的现金和赠送金支付比例退还至您的TCloudFinanceZone账户。

操作步骤

包年包月实例

1. 登录 MongoDB 控制台，在实例列表的"操作"列中，选择【更多】>【退货退费】。
   > 包年包月实例，当退货按钮不可用时，表示该账号已使用完包年包月自助退还的限额，表明包年包月实例无法手动销毁，到期后会自动销毁。

2. 在弹出的对话框中，确认无误后，单击【确认】。

3. 在退款信息页，确认无误后，单击【确认退款】。

按量计费实例

1. 登录 MongoDB 控制台，在实例列表的“操作”列中，选择【更多】>【销毁】。
2. 在弹出的对话框中，确认无误后，单击【确认】。

   > 单击【确认】后实例会直接销毁，无保留期，请谨慎操作。

备份与恢复
恢复数据

操作场景

本文为您介绍通过控制台恢复云数据库 MongoDB 数据的操作。
实例的 oplog 空间为固定集合（Capped Collection），当集合空间用完后，再插入的元素就会覆盖最初始的头部的元素。oplog 空间被覆盖可能导致备份和恢复失败，以及无法保证数据恢复的时间点，请根据业务详情合理设置 oplog 空间大小。

> 请关注：实例管理页【系统监控】里的【oplog时间差】监控指标，在业务有频繁写入、更新和删除操作时，该指标越小，oplog 被覆盖的风险越大。

前提条件

已备份实例数据，请参见 备份数据。

操作步骤

1. 登录 云数据库 MongoBD 控制台 ，在实例列表中，单击实例名进入实例管理页面。
2. 选择【备份与回档】>【备份列表】页，在需要回档的备份文件列，单击【回档实例】。

3. 在回档页，可选择回档的时间点，以及回档类型。
   选择【整实例回档】时，系统为您免费创建了一个临时实例，用于存放回档数据，临时实例的用户名和密码与原实例一致。原实例保持不变，不会对业务造成任何影响。
   整实例回档在回档完成后的48小时内，用户需要访问临时实例确认回档数据。对于临时实例您可选择以下操作：

- “转正”：将临时实例转为独立于原实例的正式实例，供业务使用。
- “替换”：用临时实例替换掉原实例（原实例内网 IP 绑定到临时实例）。

备份数据

云数据库 MongoDB 默认每日进行自动备份，用户也可以进行手动备份。本文为您介绍通过控制台备份云数据库 MongoDB 的操作

> 注意：
>
> 实例备份过程中不影响业务使用。

备份类型

自动备份：平台内置任务，每天会自动发起备份，并根据节点资源负载情况调整备份时间，尽可能降低对DB的影响。
手工备份：租户，根据自身需要，手工发起备份数据。

自动备份

1. 登录 云数据库 MongoDB 控制台，在实例列表页，单击实例名进入管理页面，选择【备份与回档】>【自动备份设置】页。
2. 选择备份参数，单击【保存】。参数说明如下：

参数	说明
数据备份保留	默认保留7天。目前免费，后续备份空间收费将另行通知。
备份方式	支持物理备份和逻辑备份。云平台支持逻辑备份。
备份时间间隔	支持每12小时备份一次和每24小时备份一次。默认为24小时，即每天备份一次。
备份开始时间	支持不同时间段。默认开始时间为01:00-02:00，即系统会在每天01:00-02:00时间段内开始备份任务。具体的开始时间会随着备份任务具体调度而变化。 当备份时间间隔为每12小时一次，则备份开始时间为首次备份开始时间。
备份异常是否通知	指备份任务执行异常时是否通知用户。MongoDB 的云监控支持事件请参见事件列表。

3. 事件列表

事件中文名称	事件英文名称	事件类型	从属维度	有无恢复概念	事件描述	处理方法和建议
备份 oplog 不足	oplogInsufficient	异常事件	云数据库 MongoDB	无	云数据库 MongoDB	建议在 MongoDB 控

事件中文名称	事件英文名称	事件类型	从属维度	有无恢复概念	事件描述	处理方法和建议
			实例维度		在备份时，无法读取到上次备份到本次备份的完整oplog，这将影响您的数据库回档到 7 天内的任意时间点	制台 调整云数据库 MongoDB oplog 的大小或备份频率；如您不需要该事件通知，可以在 MongoDB 控制台 备份界面进行设置以关闭该事件通知
连接数超限	connectionOverlimit	异常事件	云数据库 MongoDB 实例维度	有	实例连接数使用超过限制	评估当前实例所配置连接数是否满足业务需求，如果需要更大的连接配置建议，升级MongoDB数据库实例配置
主从切换	primarywitch	异常事件	云数据库 MongoDB 实例维度	有	实例 Primary 和 Secondary 切换	当物理机故障时可能会触发该事件，请确认实例状态是否正常
磁盘空间已耗尽	instanceOutOfDisk	异常事件	云数据库 MongoDB 实例维度	有	磁盘空间写满，造成实例只读	清理磁盘空间
实例 Rollback	instanceRollback	异常事件	云数据库 MongoDB 实例维度	有	实例数据 rollback	当主节点有部分数据还没有及时同步到从节点时主节点故障并发生主从切换可能会触发该事件，请确认实例状态是否正常

事件中文名称	事件英文名称	事件类型	从属维度	有无恢复概念	事件描述	处理方法和建议
节点 CPU 异常	NodeCPUAbnormal	异常事件	云数据库 MongoDB 实例维度	有	集群中有任一节点CPU使用率达到80%，即触发告警	单次告警仅代表实例有单个节点负载较高，可结合连接数和慢查询等其他实例运行状态升级综合评估，必要时升级MongoDB数据库实例配置

手动备份

1. 在管理页面的右上角单击【手动备份】。

2. 为增加区分度，在弹出的对话框添加备注信息，单击【确认】即可。

网络与安全
访问管理
访问管理概述

存在问题

如果您在TCloudFinanceZone中使用到了云服务器、私有网络、云数据库等多项服务，这些服务由不同的人管理，但都共享您的云账号密钥，将存在如下问题：

- 您的密钥由多人共享，泄密风险高。
- 您无法限制其它人的访问权限，易产生误操作造成安全风险。

解决方案

您可以通过 子账号 实现不同的人管理不同的服务来规避以上的问题。默认情况下，子账号没有使用云服务的权利或者相关资源的权限。因此，我们就需要创建策略来允许子账号使用他们所需要的资源或权限。

访问管理（Cloud Access Management，CAM）是TCloudFinanceZone提供的一套 Web 服务，主要用于帮助用户安全管理TCloudFinanceZone账户下资源的访问权限。通过 CAM，您可以创建、管理和销毁用户（组），并通过身份管理和策略管理控制指定用户可以使用的TCloudFinanceZone资源。

当您使用 CAM 的时候，可以将策略与一个用户或一组用户关联起来，策略能够授权或者拒绝用户使用指定资源完成指定任务。有关 CAM 策略的更多基本信息，请参见 策略语法。

若您不需要对子账户进行云数据库相关资源的访问管理，您可以跳过此章节。跳过这些部分不会影响您对文档中其余部分的理解和使用。

快速入门

CAM 策略必须授权使用一个或多个 MongoDB 操作，或者必须拒绝使用一个或多个 MongoDB 操作，同时还必须指定可以用于操作的资源（可以是全部资源，某些操作也可以是部分资源），策略还可以包含操作资源所设置的条件。

> - 建议用户使用 CAM 策略来管理 MongoDB 资源和授权 MongoDB 操作，对于存量分项目权限的用户体验不变，但不建议再继续使用分项目权限来管理资源与授权操作。
> - MongoDB 暂时不支持相关生效条件设置。

相关内容	链接

相关内容	链接
了解策略基本结构	策略语法
在策略中定义操作	MongoDB 的操作
在策略中定义资源	MongoDB 的资源路径
资源级权限	MongoDB 支持的资源级权限

授权策略语法

CAM 策略语法

```
{
      "version":"2.0",
      "statement":
      [
        {
          "effect":"effect",
          "action":["action"],
          "resource":["resource"],
           "condition": {"key":{"value"}}
        }
     ]
}
```

- 版本 version：必填项，目前仅允许值为"2.0"。
- 语句 statement：用来描述一条或多条权限的详细信息。该元素包括 effect、action、resource、condition 等多个其他元素的权限或权限集合。一条策略有且仅有一个 statement 元素。
- 影响 effect：必填项，描述声明产生的结果是"允许"还是"显式拒绝"。包括 allow（允许）和 deny（显式拒绝）两种情况。
- 操作 action：必填项，用来描述允许或拒绝的操作。操作可以是 API 或者功能集（一组特定的 API ，以 permid 前缀描述）。
- 资源 resource：必填项，描述授权的具体数据。资源是用六段式描述，每款产品的资源定义详情会有所区别。
- 生效条件 condition：必填项，描述策略生效的约束条件。条件包括操作符、操作键和操作值组成。条件值可包括时间、IP 地址等信息，有些服务允许您在条件中指定其他值。

MongoDB 的操作

在 CAM 策略语句中，您可以从支持 CAM 的任何服务中指定任意的 API 操作。对于 MongoDB，请使用以 mongodb: 为前缀的 API。例如 mongodb:BackupDBInstance 或 mongodb:CreateAccountUser。
如果您要在单个语句中指定多个操作的时候，请使用逗号将它们隔开，如下所示：

```
"action":["mongodb:action1","mongodb:action2"]
```

您也可以使用通配符指定多项操作。例如，您可以指定名字以单词" Describe "开头的所有操作，如下所示：

```
"action":["mongodb:Describe*"]
```

如果您要指定 MongoDB 中所有操作，请使用 * 通配符，如下所示：

```
"action"：["mongodb:*"]
```

MongoDB 的资源路径

每个 CAM 策略语句都有适用于自己的资源。
资源路径的一般形式如下：

```
qcs:project_id:service_type:region:account:resource
```

- project_id：描述项目信息，仅为了兼容 CAM 早期逻辑，无需填写。
- service_type：产品简称，如 mongodb。
- region：地域信息，如 bj。
- account：资源拥有者的主帐号信息，如 uin/12345678。
- resource：各产品的具体资源详情，如 instance/instance_id 或者 instance/*。
  例如，您可以使用特定实例（cmgo-aw6g1g0z）在语句中指定它，如下所示：

```
"resource":[ "qcs::mongodb:bj:uin/12345678:instance/cmgo-aw6g1g0z"]
```

您还可以使用 * 通配符指定属于特定账户的所有实例，如下所示：

```
"resource":[ "qcs::mongodb:bj:uin/12345678:instance/*"]
```

您要指定所有资源，或者如果特定 API 操作不支持资源级权限，请在 resource 元素中使用 * 通配符，如下所示：

```
"resource": ["*"]
```

如果您想要在一条指令中同时指定多个资源，请使用逗号将它们隔开，如下所示为指定两个资源的例子：

```
"resource":["resource1","resource2"]
```

下表描述了 MongoDB 能够使用的资源和对应的资源描述方法。其中，$ 为前缀的单词均为代称，project 指项目 ID，region 指地域，account 指账户 ID。

资源	授权策略中的资源描述方法
实例	qcs::mongodb:$region:$account:instance/$instanceId
VPC	qcs::vpc:$region:$account:vpc/$vpcId
安全组	qcs::cvm:$region:$account:sg/$sgId

authresourtype

资源级权限指的是能够指定用户对哪些资源具有执行操作的能力。MongoDB 部分支持资源级权限，即表示针对支持资源级权限的 MongoDB 操作，您可以控制何时允许用户执行操作或是允许用户使用特定资源。访问管理 CAM 中可授权的资源类型如下：

资源类型	授权策略中的资源描述方法
MongoDB 实例相关	qcs::mongodb:$region::instance/* qcs::mongodb:$region:$account:instanceId/$instanceId

下表将介绍当前支持资源级权限的 MongoDB API 操作，以及每个操作支持的资源。指定资源路径的时候，您可以在路径中使用 * 通配符。

> 表中未列出的云数据库 API 操作，即表示该云数据库 API 操作不支持资源级权限。针对不支持资源级权限的云数据库 API 操作，您仍可以向用户授予使用该操作的权限，但策略语句的资源元素必须指定为 *。

MongoDB 实例相关

API 操作	资源路径
BackupDBInstance	qcs::mongodb:$region:$account:instanceId/*``qcs::mongodb:$region:$account:instanceId/$instanceId
CreateAccountUser	qcs::mongodb:$region:$account:instanceId/*``qcs::mongodb:$region:$account:instanceId/$instanceId
CreateDBInstance	qcs::mongodb:$region:$account:instanceId/*``qcs::mongodb:$region:$account:instanceId/$instanceId
CreateDBInstanceHour	qcs::mongodb:$region:$account:instanceId/*``qcs::mongodb:$region:$account:instanceId/$instanceId
DeleteAccountUser	qcs::mongodb:$region:$account:instanceId/*``qcs::mongodb:$region:$account:instanceId/$instanceId
DescribeAccountUsers	qcs::mongodb:$region:$account:instanceId/*``qcs::mongodb:$region:$account:instanceId/$instanceId
DescribeBackupAccess	qcs::mongodb:$region:$account:instanceId/*``qcs::mongodb:$region:$account:instanceId/$instanceId
DescribeBackupRules	qcs::mongodb:$region:$account:instanceId/*``qcs::mongodb:$region:$account:instanceId/$instanceId

API 操作	资源路径
DescribeClientConnections	qcs::mongodb:$region:$account:instanceId/*``qcs::mongodb:$region:$account:instanceId/$instanceId
DescribeDBBackups	qcs::mongodb:$region:$account:instanceId/*``qcs::mongodb:$region:$account:instanceId/$instanceId
DescribeDBInstances	qcs::mongodb:$region:$account:instanceId/*``qcs::mongodb:$region:$account:instanceId/$instanceId
DescribeInstanceDB	qcs::mongodb:$region:$account:instanceId/*``qcs::mongodb:$region:$account:instanceId/$instanceId
DescribeInstanceTaskInfo	qcs::mongodb:$region:$account:instanceId/*``qcs::mongodb:$region:$account:instanceId/$instanceId
DescribeSlowLog	qcs::mongodb:$region:$account:instanceId/*``qcs::mongodb:$region:$account:instanceId/$instanceId
DescribeSlowLogPattern	qcs::mongodb:$region:$account:instanceId/*``qcs::mongodb:$region:$account:instanceId/$instanceId
DescribeSpecInfo	qcs::mongodb:$region:$account:instanceId/*``qcs::mongodb:$region:$account:instanceId/$instanceId
ExchangeInstance	qcs::mongodb:$region:$account:instanceId/*``qcs::mongodb:$region:$account:instanceId/$instanceId
IsolateDBInstance	qcs::mongodb:$region:$account:instanceId/*``qcs::mongodb:$region:$account:instanceId/$instanceId
ModifyDBInstanceSpec	qcs::mongodb:$region:$account:instanceId/*``qcs::mongodb:$region:$account:instanceId/$instanceId
OfflineIsolatedDBInstance	qcs::mongodb:$region:$account:instanceId/*``qcs::mongodb:$region:$account:instanceId/$instanceId
RemoveCloneInstance	qcs::mongodb:$region:$account:instanceId/*``qcs::mongodb:$region:$account:instanceId/$instanceId
RenameInstance	qcs::mongodb:$region:$account:instanceId/*``qcs::mongodb:$region:$account:instanceId/$instanceId
RenewInstance	qcs::mongodb:$region:$account:instanceId/*``qcs::mongodb:$region:$account:instanceId/$instanceId
ResizeOplog	qcs::mongodb:$region:$account:instanceId/*``qcs::mongodb:$region:$account:instanceId/$instanceId

API 操作	资源路径
RestartInstance	qcs::mongodb:$region:$account:instanceId/*``qcs::mongodb:$region:$account:instanceId/$instanceId
RestoreDBInstance	qcs::mongodb:$region:$account:instanceId/*``qcs::mongodb:$region:$account:instanceId/$instanceId
SetAccountUserPrivilege	qcs::mongodb:$region:$account:instanceId/*``qcs::mongodb:$region:$account:instanceId/$instanceId
SetAutoRenew	qcs::mongodb:$region:$account:instanceId/*``qcs::mongodb:$region:$account:instanceId/$instanceId
SetBackupRules	qcs::mongodb:$region:$account:instanceId/*``qcs::mongodb:$region:$account:instanceId/$instanceId
SetInstanceFormal	qcs::mongodb:$region:$account:instanceId/*``qcs::mongodb:$region:$account:instanceId/$instanceId
SetInstanceMaintenance	qcs::mongodb:$region:$account:instanceId/*``qcs::mongodb:$region:$account:instanceId/$instanceId
SetPassword	qcs::mongodb:$region:$account:instanceId/*``qcs::mongodb:$region:$account:instanceId/$instanceId
SetReadOnlyToNormal	qcs::mongodb:$region:$account:instanceId/*``qcs::mongodb:$region:$account:instanceId/$instanceId
TerminateDBInstance	qcs::mongodb:$region:$account:instanceId/*``qcs::mongodb:$region:$account:instanceId/$instanceId
TerminateDBInstanceHour	qcs::mongodb:$region:$account:instanceId/*``qcs::mongodb:$region:$account:instanceId/$instanceId
UpgradeDBInstance	qcs::mongodb:$region:$account:instanceId/*``qcs::mongodb:$region:$account:instanceId/$instanceId
UpgradeDBInstanceHour	qcs::mongodb:$region:$account:instanceId/*``qcs::mongodb:$region:$account:instanceId/$instanceId

重置密码

操作场景

本文指导您在需要对实例重置密码时，通过云数据库 MongoDB 控制台进行重置操作。

操作步骤

1. 登录 MongoDB 控制台，在实例列表，单击实例名或"操作"列的【更多】>【管理】，进入管理页面。
2. 在实例管理页，选择【数据库管理】>【账号管理】页，在"操作"列单击【修改密码】。
3. 在弹出的对话框，输入新密码等信息，单击【确认】即可。 > - 数据库密码建议定期更换，最长间隔不超过3个月。

- 数据库密码规格需要8-16个字符，可输入 [A-Za-z0-9!@#%^*()] 范围内的字符，不能为单一的字母或者数字。

监控与告警
监控介绍

监控粒度

云数据库 MongoDB 暂不支持监控粒度的自定义选择，监控自适应策略如下：

时间跨度类型	时间跨度	监控粒度	保留时长
近24小时	0分钟 - 1分钟	1分钟	31天
近24小时	0分钟 - 5分钟	1分钟	31天
近24小时	0小时 -1小时	1分钟	31天
近24小时	0天 -1天	1分钟	31天
近7天	0天 - 2天	1分钟	31天
近7天	2天 - 7天	1分钟	31天
近7天	7天 - 30天	1分钟	31天

支持监控的实例类型

云数据库 MongoDB 支持主实例、只读实例和灾备实例的监控，并为每个实例提供独立的监控视图供查询。

监控指标

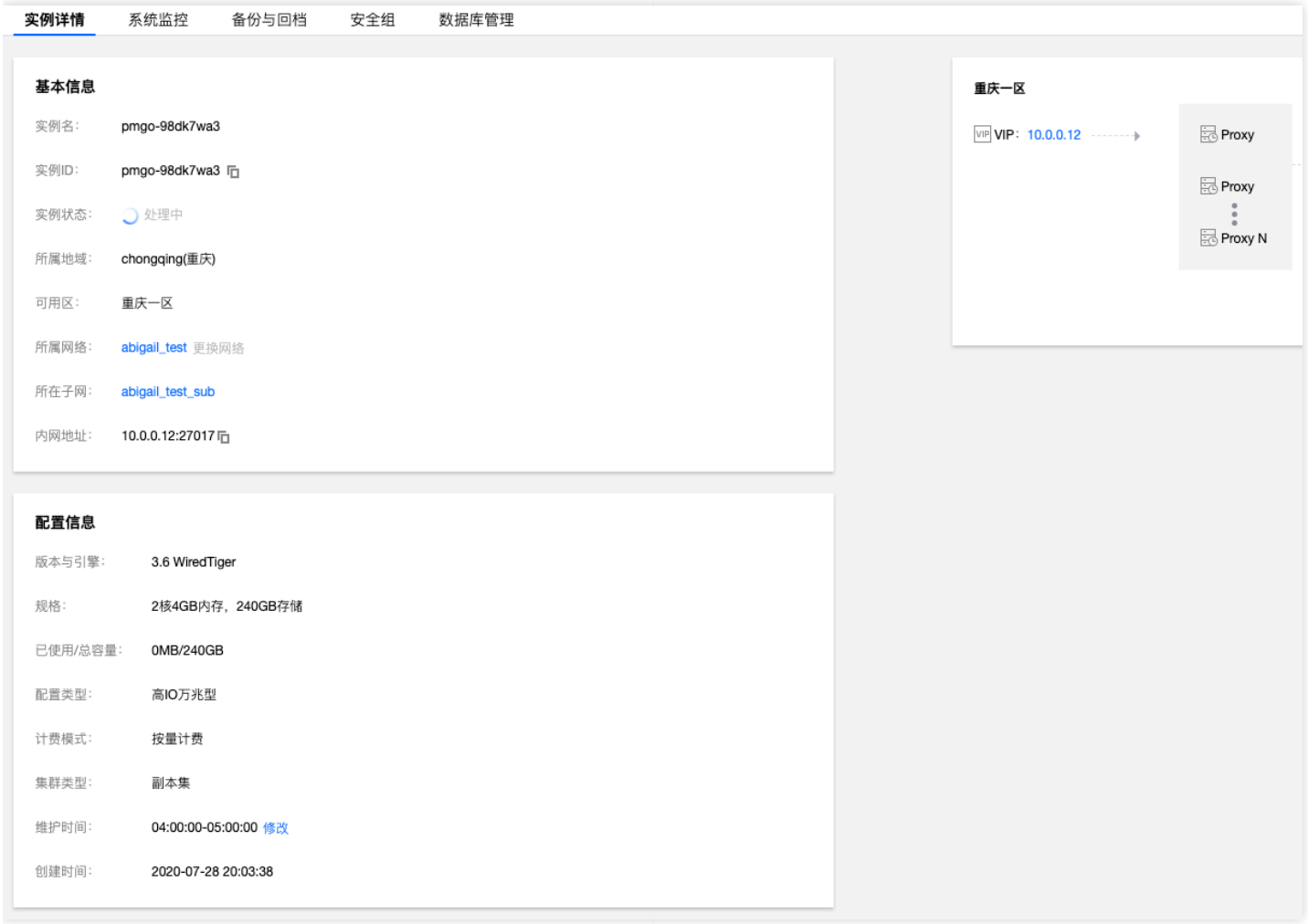
TCloudFinanceZone监控为云数据库 MongoDB 实例提供以下监控指标：

指标中文名	指标英文名	单位	维度	指标说明
写入请求	inserts	次	集群	当前集群的写入请求数
读取请求	reads	次	集群	当前集群的读取请求数
更新请求	updates	次	集群	当前集群的更新请求数
删除请求	deletes	次	集群	当前集群的删除请求数

指标中文名	指标英文名	单位	维度	指标说明
count 请求	counts	次	集群	当前集群的总请求数
Aggregates 请求	aggregates	次	集群	当前集群的聚合请求数
集群连接数	conn	次	集群	集群总连接数，指当前集群 proxy 收到的连接数
集群连接数百分比	connper	%	集群	当前集群的连接数与集群总连接配置的比例
容量使用率	diskusage	%	集群	集群当前实际占用存储空间与总容量配置的比例
QPS	qps	次/秒	集群	每秒操作数，包含 CRUD 操作
10毫秒 - 50毫秒	10ms	次	集群	执行时间在10毫秒和50毫秒之间的请求数
50毫秒 - 100毫秒	50ms	次	集群	执行时间在50毫秒和100毫秒之间的请求数
100毫秒	100ms	次	集群	执行时间超过100毫秒的请求数
超时	timeouts	次	集群	执行时间超时的请求数
主从延迟	slavedelay	秒	集群	主从延迟时间
oplog 时间差	oplogreservedtime	小时	集群	oplog 记录中最后一次操作和首次操作时间差
CPU 使用率	cpuusage	%	节点	当前节点的 CPU 使用率
内存使用率	memusage	%	节点	当前节点的内存使用率
qr	qr	个	节点	等待读操作的客户端队列长度
qw	qw	个	节点	等待写操作的客户端队列长度
连接数	conn	个	节点	当前 mongod 节点的连接数
netin	netin	MB/s	节点	入站流量
netout	netout	MB/s	节点	出站流量

查看及监控实例数据

登录 MongoDB 控制台，在实例列表，单击实例名进入管理页面，进入实例详情页和系统监控页可查看详细信息。




实例详情
系统监控
备份与回档
安全组
数据库管理
集群总览
cmgo-
主节点
从节点-0
从节点-1
实时
近24小时
近7天
选择日期
数据对比
时间粒度：
1分钟
注释：Max、Min和Avg数值统计为当前折线图内所有点的最大值、最小值和平均值
CURD请求
写入请求(次)
读取请求(次)
更新请求(次)
删除请求(次)
2
1
0

最佳实践
分片集群使用最佳实践

分片集群为 MongoDB 的分布式版本，相较副本集，分片集群数据被均衡的分布在不同分片中， 不仅大幅提升了整个集群的数据容量上限，也将读写的压力分散到不同分片，以解决副本集性能瓶颈的难题，但分片集群的架构更加复杂，本文重点介绍使用TCloudFinanceZone MongoDB 分片集群时的注意事项。

分片集群组件

一个 MongoDB 分片集群由如下三个组件构成，缺一不可：

- shard：每个分片是整体数据的一部分子集，每个分片都部署为副本集。
- mongos：充当查询路由器，提供客户端应用程序和分片集群之间的接口。
- config servers：配置服务器存储集群的元数据和配置，包括权限认证相关。

分片集群 sharding 方式及性能影响

MongoDB 分片集群提供三种 Sharding（数据分布）方式，分别为基于范围、基于 Hash、基于 zone/tag。不同的 Sharding 方式使用不同的业务，也会对性能产生不同的影响。

- 基于范围
  优势：分片键范围查询性能较好，读性能较好。
  劣势：数据分布可能不均匀，存在热点。
- 基于 Hash
  优势：数据分布均匀，写性能较好，适用于日志、物联网等高并发场景。
  劣势：范围查询效率较低。
- 基于 zone/tag
  若数据具备一些天然的区分，如基于地域、时间等标签，数据可以基于标签来做区分。
  优势：数据分布较为合理。

分片键的选择

分片键是文档中的某一个字段，用来进行路由查询，分片键是不可变的，且必须有索引。
选择合适的片键对 sharding 效率影响很大，主要基于如下四个因素：

- 取值基数
  取值基数建议尽可能大，如果用小基数的片键，因为备选值有限，那么块的总数量就有限，随着数据增多，块

的大小会越来越大，导致水平扩展时移动块会非常困难。
例如：选择年龄做一个基数，范围最多只有100个，随着数据量增多，同一个值分布过多时，导致 chunck 的增长超出 chuncksize 的范围，引起 jumbo chunk，从而无法迁移，导致数据分布不均匀，性能瓶颈。

- 取值分布
  取值分布建议尽量均匀，分布不均匀的片键会造成某些块的数据量非常大，同样有上面数据分布不均匀，性能瓶颈的问题。
- 查询带分片
  查询时建议带上分片，使用分片键进行条件查询时，mongos 可以直接定位到具体分片，否则 mongos 需要将查询分发到所有分片，再等待响应返回。
- 避免单调底层或递减
  单调递增的 sharding key，数据文件挪动小，但写入会集中，导致最后一篇的数据量持续增大，不断发生迁移，递减同理。
  综上，在选择片键时要考虑以上4个条件，尽可能满足更多的条件，才能降低 MoveChuncks 对性能的影响，从而获得最优的性能体验。

分片集群 balance 介绍及相关参数

在一个分片集群内部，MongoDB 会把数据分为 chunks，后台进程 balancer 负责 chunk 的迁移，从而均衡各个 shard server 的负载，每个 chunk 包含一部分数据，chunk 的产生和迁移会导致 balance 的产生。

> 说明：
>
> 系统初始仅1个 chunk，chunk size 默认值64MB。

chunck 迁移时会造成集群的读写性能下降，因此需要通过适当配置 balance 活动窗口来避免 balance 对业务高峰期的影响，也可以通过命令来关闭 balance。

下面介绍管理 balance 的相关命令，若某些指令无权限执行，请提交工单处理。

- 查看 mongo 集群是否开启了 balance

```
mongos> sh.getBalancerState()
true
```

也可通过执行 sh.status() 查看 balance 状态。

- 查看是否正在有数据的迁移

```
mongos> sh.isBalancerRunning()
false
```

- 设置 balance 窗口
- 修改 balance 窗口的时间：

```
db.settings.update(
  { _id: "balancer" },
  { $set: { activeWindow : { start : "<start-time>", stop : "<stop-time>" } } },
  { upsert: true }
)
```

- 删除 balance 窗口：

```
use config
db.settings.update({ _id : "balancer" }, { $unset : { activeWindow : true } })
```

- 关闭 balance
- 默认 balance 的运行可以在任何时间，迁移只需要迁移的 chunk，如需关闭 balance，可执行下列命令：

```
sh.stopBalancer()
sh.getBalancerState()
```

- 停止 balace 后，查看是否有迁移进程正在执行，可执行下列命令：

```
use config
while( sh.isBalancerRunning() ) {
        print("waiting...");
        sleep(1000);
}
```

- 打开 balance
- 如您需要准备重新打开 balance，可执行下列命令：

```
sh.setBalancerState(true)
```

- 当驱动版本不支持 sh.startBalancer() 时，可执行下列命令来重新打开 balance：

```
use config
db.settings.update( { _id: "balancer" }, { $set : { stopped: false } } , { upsert: true } )
```

- 集合的 balance
- 关闭某个集合的 balance：

```
sh.disableBalancing("students.grades")
```

- 打开某个集合的 balance：

```
sh.enableBalancing("students.grades")
```

- 查看某个集合是否开启了 balance：

```
db.getSiblingDB("config").collections.findOne({_id : "students.grades"}).noBalance
```

MongoDB协议实例读写示例

本文以 Python 代码示例来演示 MongoDB 分片集群的数据基本读写操作。首先在控制台创建分片集群实例，创建完成之后，在业务侧补充下述代码：

示例代码：

```
#!/usr/bin/python
import pymongo
import random


mongodbUri = 'mongodb://mongouser:1234567a@10.66.153.111:27017/admin'

client = pymongo.MongoClient(mongodbUri)
db = client.test

if 'num' in db.collection_names():
    db.drop_collection('num')

#create database and shardkey,shardkey is name
db_admin=client.admin
db_admin.command('enableSharding', 'test')
db_admin.command('shardCollection', 'test.num', key = {'name':1})

#insert data
print 'insert docs'
db.num.insert_one({'id':1, 'name':'R9', 'des':'pretty'})
db.num.insert_one({'id':2, 'name':'BOY', 'des':'handsome'})
db.num.insert_one({'id':3, 'name':'cat', 'des':'nice'})
db.num.insert_one({'id':4, 'name':'dog', 'des':'clever'})
print 'list all docs'
for i in db.num.find(): print i

#insert update doc
print 'update R9 and delete BOY'
db.num.update_one({"name":"R9"},{"$set":{"des":"good"}})
db.num.delete_one({"name":"BOY"})
db.num.update_one({"id":3}, {"$set":{"des":"kind"}})

print 'print R9'
for i in db.num.find({"name":"R9"}): print i
print 'list all docs'
for i in db.num.find(): print i
```

运行结果：

```
[root@VM_63_228_centos distribute_test]# python demo.py
insert docs
list all docs
{u'_id': ObjectId('589c62e99d89702a48ebb10c'), u'des': u'pretty', u'id': 1, u'name': u'R9'}
{u'_id': ObjectId('589c62e99d89702a48ebb10e'), u'des': u'nice', u'id': 3, u'name': u'cat'}
{u'_id': ObjectId('589c62e99d89702a48ebb10f'), u'des': u'clever', u'id': 4, u'name': u'dog'}
{u'_id': ObjectId('589c62e99d89702a48ebb10d'), u'des': u'handsome', u'id': 2, u'name': u'BOY'}
update R9 and delete BOY
print R9
{u'_id': ObjectId('589c62e99d89702a48ebb10c'), u'des': u'good', u'id': 1, u'name': u'R9'}
list all docs
{u'_id': ObjectId('589c62e99d89702a48ebb10c'), u'des': u'good', u'id': 1, u'name': u'R9'}
{u'_id': ObjectId('589c62e99d89702a48ebb10e'), u'des': u'kind', u'id': 3, u'name': u'cat'}
{u'_id': ObjectId('589c62e99d89702a48ebb10f'), u'des': u'clever', u'id': 4, u'name': u'dog'}
[root@VM_63_228_centos distribute_test]#
```

导出导入

local 数据库主要存储副本集的配置信息、oplog 等元数据；admin 数据库则主要存储用户、角色等信息。为了防止数据错乱、鉴权失败等现象发生，云数据库 MongoDB 禁止将 local 和 admin 数据库导入实例。

在 CVM 中可用 MongoDB 提供的 shell 客户端连接云数据库 MongoDB 进行数据导入和导出，请注意使用最新版本的 MongoDB 客户端套件，具体操作可参见 连接示例。

导出导入命令

MongoDB 官方提供了两套数据导入导出工具：

- mongodump 和 mongorestore
- mongoexport 和 mongoimport

mongodump 和 mongorestore

进行整库导出导入时，通常使用 mongodump 和 mongorestore，这一对组合操作的数据是 BSON 格式，进行大量 dump 和 restore 时效率较高。

- mongodump 导出命令如下：

  mongodump --host 10.66.187.127:27017 -u mongouser -p thepasswordA1 --authenticationDatabase=admin --db=testdb -o /data/dump_testdb

```
#: ./mongodump --host 10.66.187.127:27017 -u mongouser -p thepasswordA1 --authenticationDatabase=admin
 --db=testdb -o /data/dump_testdb
2016-11-16T12:12:06.114+0800    writing testdb.system.indexes to
2016-11-16T12:12:06.116+0800    done dumping testdb.system.indexes (1 document)
2016-11-16T12:12:06.116+0800    writing testdb.testcollection to
2016-11-16T12:12:06.118+0800    done dumping testdb.testcollection (3 documents)
```

- mongorestore 导入命令如下：

  mongorestore --host 10.66.187.127:27017 -u mongouser -p thepasswordA1 --authenticationDatabase=admin --dir=/data/dump_testdb

```
#: ./mongorestore --host 10.66.187.127:27017 -u mongouser -p thepasswordA1 --authenticationDatabase=ad
min --dir=/data/dump_testdb
2016-11-16T12:13:23.654+0800    building a list of dbs and collections to restore from /data/dump_test
db dir
2016-11-16T12:13:23.678+0800    reading metadata for testdb.testcollection from /data/dump_testdb/test
db/testcollection.metadata.json
2016-11-16T12:13:23.678+0800    restoring testdb.testcollection from /data/dump_testdb/testdb/testcoll
ection.bson
2016-11-16T12:13:23.740+0800    restoring indexes for collection testdb.testcollection from metadata
2016-11-16T12:13:23.740+0800    finished restoring testdb.testcollection (3 documents)
2016-11-16T12:13:23.741+0800    done
#:
```

mongoexport 和 mongoimport

进行单个集合导出导入时，通常使用 mongoexport 和 mongoimport，这一对组合操作的数据是 JSON 格式，可读性较高。

- mongoexport 导出命令如下：

  ```
  mongoexport --host 10.66.187.127:27017 -u mongouser -p thepasswordA1 --authenticationDat
  abase=admin --db=testdb --collection=testcollection -o /data/export_testdb_testcollection.json
  ```

另外您也可以加上 -f 参数指定需要的字段，-q 参数指定一个查询条件来限定要导出的数据。

- mongoimport 导入命令如下：

  ```
  mongoimport --host 10.66.187.127:27017 -u mongouser -p thepasswordA1 --authenticationDat
  abase=admin --db=testdb --collection=testcollection2 --file=/data/export_testdb_testcollectio
  n.json
  ```

多种认证方式的参数说明

在 连接示例 中有说明，云数据库 MongoDB 默认提供了“rwuser”和“mongouser”两个用户名分别支持“MONGODB-CR”和“SCRAM-SHA-1”两种认证方式。

- 对于“mongouser”以及在控制台创建的所有新用户，在使用导出导入命令工具时，根据上文示例操作即可。
- 对于“rwuser”，需要在每个命令里加入参数“--authenticationMechanism=MONGODB-CR”。
  mongodump 示例：

```
mongodump --host 10.66.187.127:27017 -u rwuser -p thepasswordA1 --authenticationDatabase=admi
n --authenticationMechanism=MONGODB-CR --db=testdb -o /data/dump_testdb
```

开发指南
3.6版本命令支持情况

MongoDB 官方命令请参见 官网文档。

TCloudFinanceZone数据库 MongoDB 3.6 版本不支持的命令如下表所示：

命令分类	不支持的命令
Sharding Commands	addShard
	removeShard
Query and Write Operation Commands	getPrevError
Role Management Commands	dropAllRolesFromDatabase
Replication Commands	replSetAbortPrimaryCatchUp
	replSetFreeze
	replSetGetConfig
	replSetGetStatus
	replSetInitiate
	replSetMaintenance
	replSetReconfig
	replSetResizeOplog
	replSetStepDown
	replSetSyncFrom
	resync
Administration Commands	cloneCollection
	cloneCollection
	cloneCollectionAsCapped
	compact
	connPoolSync

	logRotate
	setParameter
	shutdown
Diagnostic Commands	availableQueryOptions
	dbHash
	getCmdLineOpts
	getLog
	shardConnPoolStats
System Events Auditing Commands	logApplicationMessage

开发运维
3.6开发运维

问题描述

MongoDB 3.6版本实例如果反复 drop 一个 Database 然后创建一个相同名字的 Database，读写或者 drop 该 Database 时可能会报错 ‘database does not exist’，具体类似下图所示：

```
mongos> show dbs
admin   0.000GB
config  0.001GB
local   0.209GB
scrm    0.001GB
mongos> use scrm
switched to db scrm
mongos> db.dropDatabase()
{
        "info" : "database does not exist",
        "ok" : 1,
        "$clusterTime" : {
                "clusterTime" : Timestamp(1546858044, 2),
                "signature" : {
                        "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAA
                        "keyId" : NumberLong(0)
                }
        },
        "operationTime" : Timestamp(1546858044, 2)
}
mongos>
```

解决办法

该问题是一个共性问题。出现该问题的原因是 mongos 可能没有更新其 metadata cache。具体请参考官方说明。如下图所示：

> **WARNING:**
> If you drop a database and create a new database with the same name, you must either restart all `mongos` instances, or use the `flushRouterConfig` command on all `mongos` instances before reading or writing to that database. This action ensures that the `mongos` instances refresh their metadata cache, including the location of the primary shard for the new database. Otherwise, the `mongos` may miss data on reads and may write data to a wrong shard.

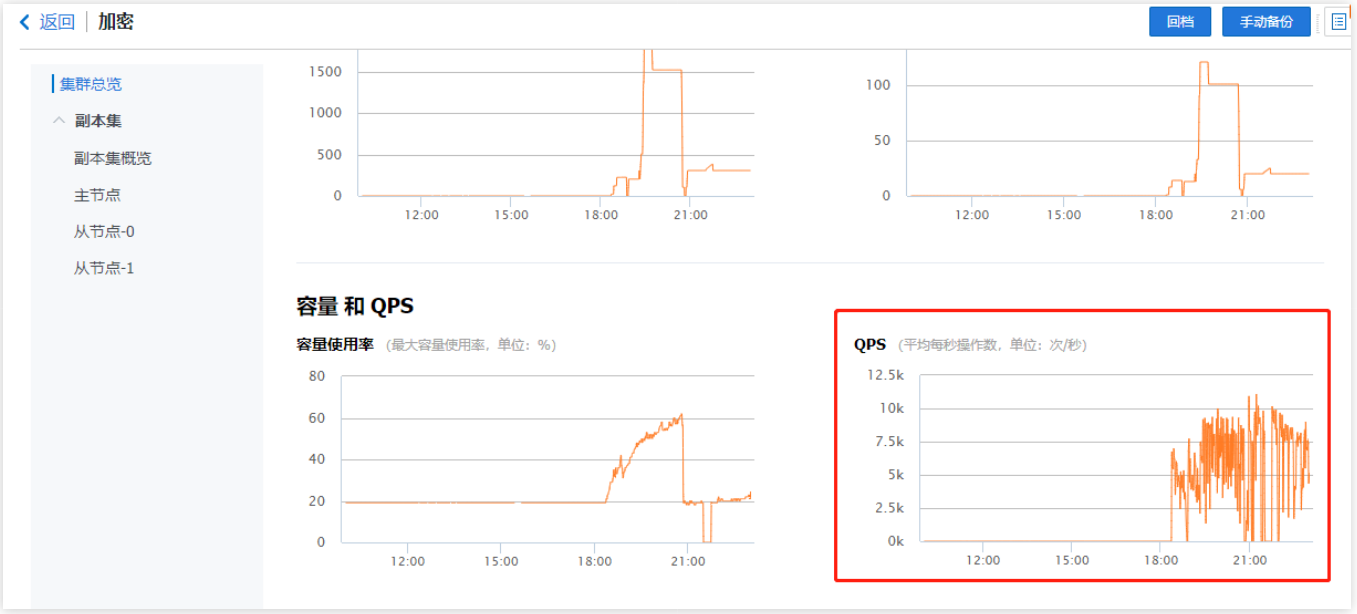
解决办法有两种，请选择其中一种：

1. 重启 mongos，该操作可以在控制台进行。
2. 或者运行 flushRouterConfig 命令，内附详细说明。

CPU使用高问题排查

用户在使用 MongoDB 时发现 CPU 使用率高，可以从以下几个方面来排查问题。

1. 首先需要确定业务是否有很高的操作数据库频率。
   请查看控制台监控指标，具体如下图所示：若业务 QPS 确实高，请评估是否需要升级实例配置。若业务 QPS 并不高，此时需要排查是不是有慢查询。



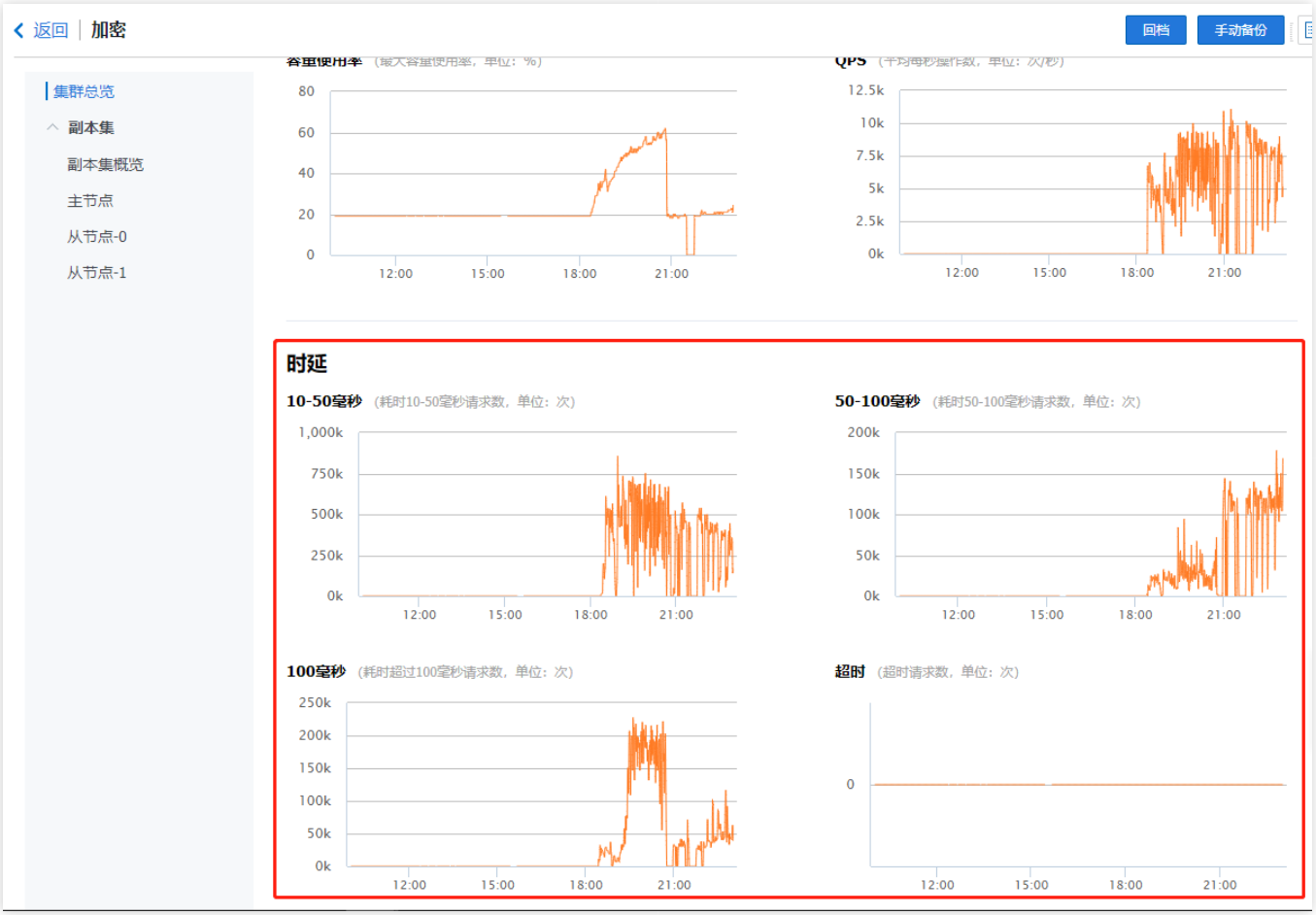
2. 查看当前 mongod 上有没有慢日志。请关注：command、COLLSCAN、IXSCAN、keysExamined、docsExamined 等关键字。

- 慢日志，需要联系平台管理员，协助获取。
- command 指出慢日志中记录的操作。
- COLLSCAN 代表该查询进行了全表扫描，IXSCAN 代表进行了索引扫描。更多的字段描述请参考 MongoDB 官网。
- keysExamined 代表索引扫描条目，docsExamined 代表文档扫描条目。keysExamined 和 docsExamined 越大代表没有建索引或者索引的区分度不高。请确认索引的创建字段。
  更多的日志说明请参考 MongoDB 官网

慢查询类问题排查

您在使用 MongoDB 时发现请求延迟明显变长时，可按照如下步骤进行排查：

1. 请检查实例的请求延迟监控指标有无异常。
   检查实例的监控数据，如下图所示：时延监控指标主要反馈的是从请求到达接入层直至处理完返回客户端的时间。若请求时延较高，则需检查 mongod 上是否有慢日志。



   参考步骤如下：
2. 查看当前 mongod 上是否有慢日志。
   请关注：command、COLLSCAN、IXSCAN、keysExamined、docsExamined 等关键字，更多的日志说明请参考 MongoDB 官网。
   需注意的关键字如下：

- command 指出慢日志中记录的操作。
- COLLSCAN 代表该查询进行了全表扫描，IXSCAN 代表进行了索引扫描。更多的字段描述请参考 MongoDB 官网。
- keysExamined 代表索引扫描条目，docsExamined 代表文档扫描条目。keysExamined 和 docsExamined 越大代表没有建索引或者索引的区分度不高。请确认索引的创建字段。

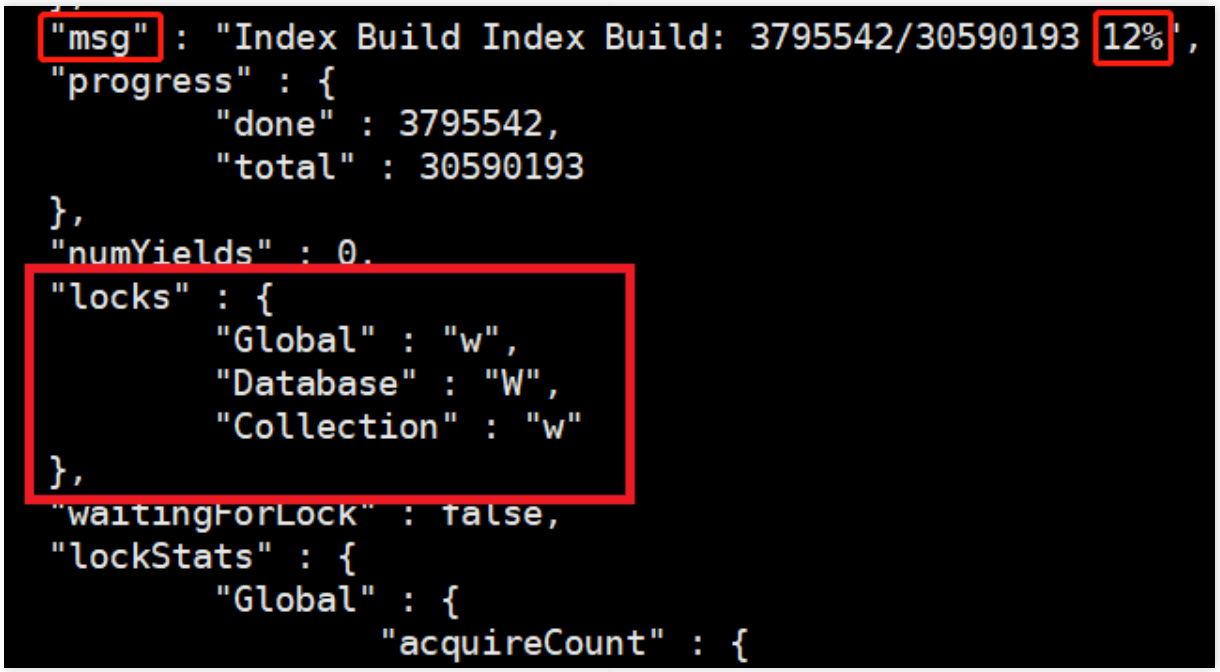
2. 请确认是否在前台建索引导致请求被锁住。
   如果业务查询所用索引并无问题，请确认当前是否在业务繁忙时段进行了前台建索引操作。当为一个集合创建索引时

默认方式为前台方式（background 选项的值为 false）。该操作将阻塞其他的所有操作，直到前台完成索引创建。但如果采用后台方式建索引，则在创建索引期间，MongoDB 依旧可以正常提供读写操作服务，不过，后台建索引方式是有代价的，即后台方式建索引可能会导致索引创建时间变长。具体创建索引的选项请参考 MongoDB 官网。

3. 可通过 currentOp 命令来查看当前建索引的进度，具体的命令如下：

```
db.currentOp(
  {
    $or: [
      { op: "command", "query.createIndexes": { $exists: true } },
      { op: "insert", ns: /\.system\.indexes\b/ }
    ]
  }
  )
```

返回如下图所示，msg 字段代表了当前建索引的进度，locks 字段代表该操作的锁类型，更多锁的说明请参考 MongoDB 官网。



Lock Mode	Description
R	Represents Shared (S) lock.
W	Represents Exclusive (X) lock.
r	Represents Intent Shared (IS) lock.
w	Represents Intent Exclusive (IX) lock.

4. 检查是否 mongos 负载过高。
5. 时延监控指标主要反馈的是从请求到达接入层直至处理完返回客户端的时间，若已确认 mongod 上没有慢操作，但请求时延较高，可能是 mongos 负载过高导致。造成 mongos 负载高的原因有多种，例如业务瞬间建立大量连接可能会导致 mongos 负载过高，或者，多个分片的数据需要汇总也可能会造成 mongos 负载高。此时可以进行 mongos 重启。重启 mongos 可以在控制台进行，您可以自行重启。

   > 说明：
   >
   > 重启 mongos 会导致实例所有的连接在重启的一瞬间中断，业务直接进行重连即可，不存在持续影响业务的可能。

连接类问题排查

您在使用 MongoDB 的过程中经常会遇见两类连接问题，下面介绍可以参考的排查思路。

连接使用率高

若您在使用过程中发现实例的连接使用率高，可参考如下步骤来排查问题。

1. 确认业务是否使用了连接池。

   MongoDB 的服务模型是每个网络连接由一个单独的线程（ one-thread-per-connection ）来处理，当网络连接数太多时，过多的线程会导致上下文切换开销变大，同时内存开销也会上涨。每次请求都建立连接和鉴权会极大的影响性能。因此，要限制实例的连接数且连接使用完毕需要释放，推荐业务使用连接池。同时，MongoDB 各个语言的 Driver 基本都会封装一个对象，应该在使用时构造一个全局的对象，然后在后续的请求中使用该全局对象来发送请求给实例。Driver 的对象有默认连接池大小，也可以在构造对象时指定 maxPoolSize 选项来设置连接池大小。
2. 如果已经设置了连接池，请检查业务是否有突发异常。

- 请确认业务是否有发布变更，代码逻辑缺陷导致建立大量连接。
- 请检查是否有连接泄漏，在相关 CVM 上统计检查连接数是否异常。
- 请确认业务是否有大量真实突发请求。

3. 请检查是否有慢查询导致连接被占用。

- 若确认业务无异常，请检查索引是否有异常，例如之前建立的索引被误删等。
- 若索引无异常，请检查当前是否有大量慢查询。慢查询导致连接一直占用未被释放，因此会建立更多的连接。

   请关注：command、COLLSCAN、IXSCAN、keysExamined、docsExamined 等关键字，更多的日志说明请参考 MongoDB 官网，

   需注意的关键字如下：

4. command 指出慢日志中记录的操作。
5. COLLSCAN 代表该查询进行了全表扫描，IXSCAN 代表进行了索引扫描。更多的字段描述请参考 MongoDB 官网。
6. keysExamined 代表索引扫描条目，docsExamined 代表文档扫描条目。keysExamined 和 docsExamined 越大代表没有建索引或者索引的区分度不高。请确认索引的创建字段。
7. 请确认是否在前台建索引导致请求被锁住

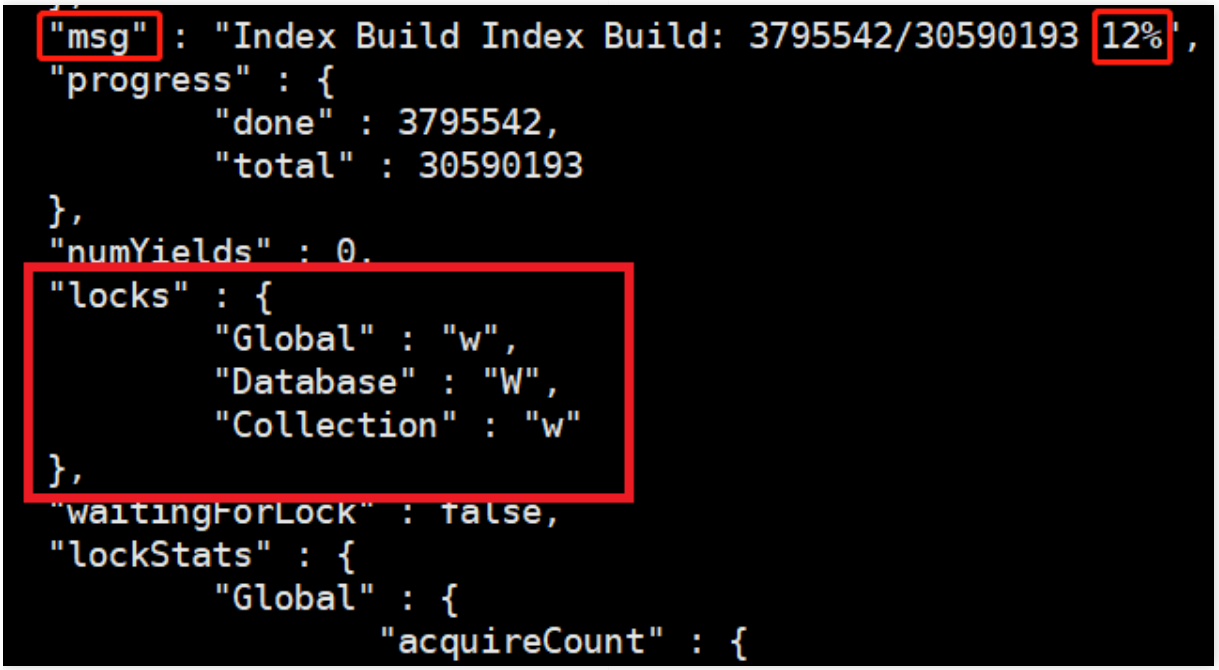
   若业务查询所用索引并无问题，请确认当前是否在业务繁忙时段进行了前台建索引操作。当为一个集合创建索引时默认方式为前台方式（ background 选项的值为 false ）。该操作将阻塞其他的所有操作，直到前台完成索引创建。但如果采用后台方式建索引，则在创建索引期间，MongoDB 依旧可以正常的提供读写操作服务，但后台建索引方式是

有代价的，即后台方式建索引可能会导致索引创建时间变长。具体创建索引的选项请参考 MongoDB 官网。

可通过 currentOp 命令来查看当前建索引的进度，具体的命令如下：

```
db.currentOp(
  {
    $or: [
      { op: "command", "query.createIndexes": { $exists: true } },
      { op: "insert", ns: /\.system\.indexes\b/ }
    ]
  }
)
```

返回如下图所示，msg 字段代表了当前建索引的进度，locks 字段代表该操作的锁类型，更多锁的说明请参考 MongoDB 官网。



Lock Mode	Description
R	Represents Shared (S) lock.
W	Represents Exclusive (X) lock.
r	Represents Intent Shared (IS) lock.
w	Represents Intent Exclusive (IX) lock.

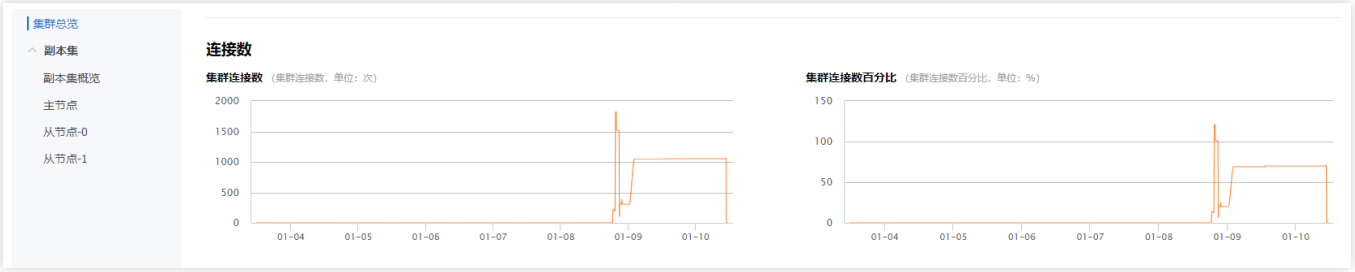
8. 评估实例配置是否满足业务要求

若经过使用连接池和建立区分度高的索引之后从控制台监控上看业务的连接使用率仍然很高，则可能是实例的配置已经不能满足业务实际需要导致。此时，需要您根据自身业务模型，流量峰值、 QPS 和 TPS 等要求来评估实际需要的实例规格配置并根据需要升级实例。

连接拒绝

若在实际使用过程中出现了连接拒绝，请参考如下步骤排查问题。

1. 确认实例连接使用率是否达到100%。
   登录 TCloudFinanceZone MongoDB 控制台，查看控制台的连接数和连接使用率监控指标，如下图所示：



   若实例连接使用率100%，请排查自身业务是否有异常，另一方面紧急情况可以通过在控制台重启 mongos 来快速释放连接。

   > ！重启 mongos 会导致实例所有的连接在重启的一瞬间中断，业务直接进行重连即可，不存在持续影响业务的可能。若重启后业务连接数迅速增加又导致连接使用率100%，则说明业务确实存在大量有效连接，不属于连接泄漏的场景。此时需要业务首先排查产生大量连接的原因，可参考连接使用率高问题的排查方法。

2. 确认用户名与密码是否错误。
   请确保用户名和密码正确，如有错，请前往控制台更改。路径为【数据库管理】->【账号管理】，具体如下图所示：

3. 确认 Mongoshell 版本。
   为保障鉴权成功，请安装 Mongo Shell 3.0及以上版本。安装步骤请参考 官方文档。

4. 确认认证库是否正确。
   > !在官网控制台创建的用户其认证库均为 admin，因此用户登录时需要制定认证库为 admin。用命令行创建的用户，例如在 test 库下建立的用户登录时指定的认证库为 test。

常见问题
功能特性问题

如何获取实例的慢日志？

专有云MongoDB产品，暂不提供在线下载/查看慢日志功能。
如果需要慢日志，请联系平台管理员协助获取。

MongoDB 是否支持外网访问？

暂不支持，用户如有外网需求，需要直接搭建代理，购买 CVM，通过内网的方式进行访问。

MongoDB 是否支持无密码访问？

基于安全原因，MongoDB 不支持无密码访问。

如何设置从库 dump？

在 mongodump 的参数中设置 readPreference=secondaryPreferred。

云数据库 MongoDB 是否支持动态添加 Secondary 节点？

目前暂不支持。

云数据库 MongoDB 与自建 MongoDB 的区别是什么？

详细介绍请参见 产品优势 。

oplog 大小是多少，是否支持调整？

oplog 大小默认为实例容量的10%，用户可在控制台调整其大小，最小为实例容量的10%，最大为实例容量的90%。
oplog暂不支持独立扩容，需要随数据存储容量一起扩。

购买的容量是否包含 oplog？

由于 oplog 存在 MongoDB 数据库内部，所以会占用实例的购买容量，默认是10%。

当前开放了哪些角色权限？

当前只开放 RoleDBAdminAny和RoleReadWriteAny 两种角色的权限，暂时不开放 root 权限，后续会逐步放开一些权限，以及开放更多便捷实用的管理控制台功能来代替某些特殊权限的调用。

磁盘使用率达到100%会发生什么？

此时实例将处于封禁状态，该状态下不可写入数据，只能做读操作，尝试写入数据的连接将会被关闭。
请及时关注自身业务发展和实例使用情况，当容量使用达到一定阈值时请适当扩容。

MongoDB 的监控里内存占用比很高？

MongoDB 采用一种贪婪策略会尽量分配可用的内存用作缓存，以提高性能，具体请参见 官方文档。

MongoDB 目前支持哪些引擎？

目前支持 WiredTiger引擎。

MongoDB 是否支持维护时间窗？

目前暂不支持。

为什么 MongoDB 删除数据后没有回收空间？

除了直接删除 db 或者表，其他情况下删除数据 MongoDB 并不会回收空间。
WiredTiger 引擎的空间回收方式请参见 官方文档。

分片集群问题

MongoDB 是否支持分片（sharding）？

支持，详情参见 创建分片集群。

什么是 MongoDB 分片集群？

云数据库 MongoDB 目前已经支持分片功能。

- 分片集群将数据按照片键分布存储在多台物理机上，平滑的扩展能力，非常适用于 TB 或 PB 级的数据存储场景。
- 分片集群支持实例级别的备份和回档来保证数据高可靠。每个分片内采用多节点自动容灾的机制，保证服务高可用。
- 可以使用TCloudFinanceZone MongoDB 分片功能便捷高效的搭建海量分布式存储系统。

如何创建 MongoDB 分片集群？

登录 MongoDB 购买页 ，在“实例类型”选择【分片集群】，按需选择分片的片数，片内节点数，以及节点规格。
每个分片都是多节点的副本集，片内多节点自动容灾，保证服务高可用。

如何查询 MongoDB 分片集群的信息？

在 控制台 中可以查看分片集群实例的详细信息，如分片的构成，片节点的规格和已使用容量，同时也可以在控制台上进行实例的 续费管理 以及 扩容 等操作。

MongoDB 分片集群扩容方式有哪些？

目前只支持将所有节点进行统一扩容，暂不支持通过添加节点的方式进行扩容。
在 控制台 实例列表页单击【配置调整】，选择需要扩到的容量规格。

MongoDB 如何实现分片集群实例监控？

云数据库 MongoDB 分片集群实例提供三个维度的监控指标，来进行整个集群的数据监控。

- 实例维度
- 片维度
- 节点维度
  提供操作请求，容量使用，负载等多项指标的监控数据，可在实例的【系统监控】页查看。

MongoDB 的分片策略是什么？

- 支持 hash key 的分片机制。
- 支持联合字段的 shard key。
- 分片实例下所有数据集合必须使用分片，建议把不分片的数据放到单独的副本集实例下。

MongoDB 分片认证机制是什么？

MongoDB 完全兼容支持 SCRAM-SHA-1 和 MONGODB-CR 两种机制。

MongoDB 分片集群命令支持情况？

详细请参见 分片集群命令支持情况。

实例相关问题

MongoDB 如何查看实例详情？

在实例列表，单击实例名可以进入详细信息页面查看实例详情。

如何访问 MongoDB 实例？

云数据库 MongoDB 提供多种语言连接方式，如 Shell，PHP，Node.js，Java，Python。
连接示例请参见 完整的连接说明 。

MongoDB 的升级实例规格花费时间与实例已用容量有关吗？

升级实例规格所需的时间取决于实例已用容量，升级期间实例会发生一次切主，切主期间会出现短暂的不可访问，大约十秒左右。

MongoDB 创建实例的流程？

可通过 购买页 按需选择规格大小和时长，单击【立即购买】创建实例。

如何在项目中查找 MongoDB 已分配项目的实例？

查找已分配项目的实例，可参考接口 DescribeMongoDBInstances 查询副本集实例列表。

MongoDB 实例的连接数规格是多少？是否支持升级连接数？

实例的连接数和实例规格相关，租户在创建/购买实例页面可直接看到，后期可以通过升级规格以获取更大的连接数。

MongoDB 如何查看实例的慢查询？

专有云MongoDB产品，暂不支持在线查看慢查询。

MongoDB 查询可创建的实例规格?

可以通过 DescribeMongoDBProduct 接口查询可创建的实例规格。

回档备份问题

MongoDB 每日自动备份和手动备份如何操作？

云数据库 MongoDB 支持两种备份方式，一种是每日自动备份，一种是手动备份。备份数据默认保留7天。

- 自动备份
  实例可提供每天一次的自动备份，您可以在TCloudFinanceZone MongoDB 控制台 实例详情页的【回档与备份】中查看。
- 手动备份
  在实例详情页的【回档与备份】中，单击右上角的【手动备份】，在弹框中输入备份的备注，提交后即可完成手动备份。

回档后进行了替换操作，是否还可以再次进行回档？

不可以，替换后原备份文件已经不再适用于替换后实例，无法再次进行回档，用户选择替换操作前务必确认。

MongoDB 回档的时间取决于什么？

回档是基于最近的一次全量备份的镜像 +oplog 进行的，回档的时间取决于回放 oplog 的量。
如果全量备份的时间点距离回档的时间很久，就需要较长的时间进行 oplog 的回放。

MongoDB 回档的转正操作是什么意思？

转正是把回档后的临时实例转化为一个全新的实例运行，该实例与原实例无任何对应关系。默认会为临时实例设置2天的有效期，请及时续费。

MongoDB 如何完成副本集实例的备份回档？

副本集实例目前支持实例级别和库表级别的备份和回档。

- 备份
  在 控制台 实例详情页单击【手动备份】，或在备份与回档页进行自动备份设置。
- 回档
  在回档操作过程中，需要输入需要回档到的日期，目前支持5日内的任意时间回档，但前提是只能选择两次备份（成功且非 oplog 写满状态）之间的时间点进行回档。
  如果没有满足的备份请执行一次 手动备份 。

MongoDB 如何完成分片集群实例的备份回档？

分片集群实例目前支持实例级别的备份和回档。

- 备份
  在 控制台 实例详情页单击【手动备份】，或在备份与回档页进行自动备份设置。
- 回档

实例的 oplog 空间为固定集合（Capped Collection），当集合空间用完后，再插入的元素就会覆盖最初始的头部的元素。为了避免 oplog 空间被覆盖导致备份和恢复失败，请根据业务详情合理设置 oplog 空间大小。当业务写入、删除和更新操作频繁时，为了防止两次备份时间点之间的 oplog 被覆盖，可以设置每天备份多次。两次备份时间点之间的 oplog 空间被覆盖，可能无法保证数据恢复的时间点。

连接相关问题

MongoDB 连接断开怎么操作？

请参见 连接示例 排除认证问题。

MongoDB 出现“Remote server has closed the connection”信息？

首先参见 连接示例 排除认证问题 ，如果能连上但是依然会出现这个问题，可能需要实现一个重连机制，详情参见 重连示例。

WiredTiger 3.2 存在锁表问题，云版本 MongoDB 是否存在类似问题？

需要根据具体问题分析，例如默认建索引肯定会加全局锁，以及用户执行 fsynclock 命令也是会加锁的。
锁是数据库的一个功能，处理并发访问的一系列问题，正常的加锁是必须的，只要不影响业务正常运行就可以。

MongoDB 应该选哪个版本的驱动程序？

推荐使用最新版本，例如 PHP 可以选择 mongo-1.6 及以上。

MongoDB 提供哪些语言连接方式？

云数据库 MongoDB 提供多种语言连接方式，例如 Shell、PHP、Node.js、Java、Python，详情请参见 连接示例。

云数据库 MongoDB 版支持哪些语言的客户端进行连接？

云数据库 MongoDB 版针对客户端连接完全兼容 MongoDB，只要是官方 MongoDB 版支持的客户端，云数据库全部支持。例如 C、C++、C#、Java、Node.js、Python、PHP、Perl 等，详情请参见 MongoDB 官方文档。

在 shell 里怎么连接MongoDB？

详情请参见 Shell 连接示例。

业务程序里连接 MongoDB 的 URI 是什么样的？

详情请参见 连接示例。

用 meteor 等各类框架、类库无法连接MongoDB，如何处理？

一般来说都是连接方式、URI 拼接错误，请先检查核实。

在 PHP 中，如何设置 MongoDB 最大连接数？

MongoDB 驱动（PHP 官网文档）可以通过在连接 URL 中配置 maxPoolSize 参数控制连接数。
MongoDB 驱动（PHP 官网文档）可以通过 Mongo::setPoolSize() 方法设置连接数，详情请参见 MongoPool::setSize。

MongoDB 连接数限制是多少？

内存规格	连接数上限
2GB	1500
4GB	2000
6GB	2500
8GB	3500
16GB	6000
24GB	8000
32GB	9500
64GB	16000
128GB	18000
240GB	18000
512GB	20000

> 对于副本集来说，连接数上限是针对整个实例的，不是单个节点；对于分片集群来说，连接数上限是针对单个 shard，详见 使用限制。

手动重连 MongoDB 怎么操作？

MongoDB 数据库服务提供的不是简单的 mongod 访问，给到用户访问的是一个负载均衡 IP，此 IP 后面是连接到一系列类似 mongos 一样存在的路由接入层。
客户端驱动会透过负载均衡 IP 与接入机建立一个长连接，当此连接处于长期间活跃状态时，TCloudFinanceZone不会对其做任何干预，但是当长连接闲置时间超过1天时（此时间会随着版本优化而调整），路由接入层会踢掉该连接。
一般来说，客户端驱动会实现一个自动重连的过程，但是也有部分语言的驱动并没有实现。对于没有实现自动重连的语言驱动，当用户使用一个已经被踢掉的连接来尝试与TCloudFinanceZone MongoDB 服务通信时可能会得到“Remote server has closed the connection”之类的错误信息，所以需要手动进行重连，这里给出一个 PHP 重连的 demo。

基于 PHP mongo 驱动的重连实现

```php
<?php

function getConnection() {
    $connection = false;
    $uri = 'mongodb://rwuser:1234567a@10.66.148.142:27017/admin?authMechanism=MONGODB-CR';
    $maxRetries = 5;
    for( $counts = 1; $counts <= $maxRetries; $counts++ ) {
        try {
            $connection = new MongoClient($uri);
        } catch( Exception $e ) {
     // 或者 根据需要使用下面的catch代码行，注意那一个“\”，某些框架使用命名空间时需要用到。
     // } catch( \Exception $e ) {
            continue;
        }
        break;
    }
    return $connection;
}

$connection = getConnection();

if($connection) {
        $db = $connection->testdb;
        $collection = $db->testcollection;

        $one = $collection->findOne();

        var_dump($one);
}
```

如何使用 mongoose 连接云数据库 MongoDB？

mongoose 连接TCloudFinanceZone MongoDB 参数如下：

```
var dbUri = " mongodb:// " + user + " : " +password + " @ " +host + ":" +port + " / " + dbName;

var opts = {
     auth :  {
          authMechanism : ' MONDODB-CR'
     }
} ;
var connection = mongodb.createConnection(dbUri, opts);
```

MongoDB 支持外网连接吗？

MongoDB 目前只支持内网连接，连接方式参见 连接示例 。
目前暂不支持开通外网访问，如果您要在本地连接 MongoDB，可以使用与 MongoDB 同一账号同一内网下的服务器做端口转发实现。

数据迁移问题

从 MongoDB 数据库中导出数据，如何设置参数？

mongodump 的参数中设置 --readPreference=secondaryPreferred。

MongoDB 支持哪些数据迁移？

目前支持两类迁移：云数据库 CVM 自建实例迁移、外网实例迁移，详情请参见 MongoDB 数据迁移。

使用 mongodump（整库）或者 mongoexport（单个集合），如何把 MongoDB 的数据导出到本地？

在CVM 中可用 MongoDB 提供的 shell 客户端 连接云数据库 MongoDB 进行数据导出，请注意使用最新版本的 MongoDB 客户端套件。
MongoDB 官方提供了两套数据导出工具，一般来说，进行整库导出时使用 mongodump，操作的数据是 BSON 格式，进行大量 dump 效率较高；进行单个集合导出时使用 mongoexport，操作的数据是 JSON 格式，可读性较高。
1. 使用 mongodump 进行整库导出备份
导出命令如下：

```
mongodump --host 10.66.187.127:27017 -u mongouser -p thepasswordA1 --authenticationDatabase=admin --db=testdb -o /data/dump_testdb
```

```
#: ./mongodump --host 10.66.187.127:27017 -u mongouser -p thepasswordA1 --authenticationDatabase=admin
 --db=testdb -o /data/dump_testdb
2016-11-16T12:12:06.114+0800    writing testdb.system.indexes to
2016-11-16T12:12:06.116+0800    done dumping testdb.system.indexes (1 document)
2016-11-16T12:12:06.116+0800    writing testdb.testcollection to
2016-11-16T12:12:06.118+0800    done dumping testdb.testcollection (3 documents)
```

2. 使用 mongoexport 进行单个集合导出备份
导出命令如下:

```
mongoexport --host 10.66.187.127:27017 -u mongouser -p thepasswordA1 --authenticationDatabase=admin --db=testdb --collection=testcollection  -o /data/export_testdb_testcollection.json
```

> ?您也可以加上 -f 参数指定需要的字段，-q 参数指定一个查询条件来限定要导出的数据。

3. 关于 rwuser 和 mongouser 用户名在写导出命令时的参数说明
在 连接示例 文档中有说明，TCloudFinanceZone MongoDB 默认提供了 rwuser 和 mongouser 两个用户名，分别支持 MONGODB-CR 和 SCRAM-SHA-1 两种认证方式。

- 对于 mongouser 以及在控制台创建的所有新用户，在使用导出命令工具时按照上文示例操作即可。
- 对于 rwuser，需要在每个命令里加入参数 --authenticationMechanism=MONGODB-CR 。

mongodump 示例说明：

```
mongodump --host 10.66.187.127:27017 -u rwuser -p thepasswordA1 --authenticationDatabase=admin --authenticationMechanism=MONGODB-CR --db=testdb -o /data/dump_testdb
```

使用 mongorestore（整库）或者 mongoimport（单个集合），如何把数据从本地导入到 MongoDB？

在 CVM 中可用 MongoDB 提供的 shell 客户端 连接云数据库 MongoDB 进行数据导入，请注意使用最新版本的 MongoDB 客户端套件。
MongoDB 官方提供了两套数据导入工具，一般来说，进行整库导出时使用 mongorestore，操作的数据是 BSON 格式，进行大量 mongorestore 效率较高；进行单个集合导入时使用 mongoimport，操作的数据是 JSON 格式，可读性较高。
1. 使用 mongorestore 进行整库导入备份
导入命令如下：

```
mongorestore --host 10.66.187.127:27017 -u mongouser -p thepasswordA1 --authenticationDatabase=admin --dir=/data/dump_testdb
```

```
#: ./mongorestore --host 10.66.187.127:27017 -u mongouser -p thepasswordA1 --authenticationDatabase=admin --dir=/data/dump_testdb
2016-11-16T12:13:23.654+0800    building a list of dbs and collections to restore from /data/dump_testdb dir
2016-11-16T12:13:23.678+0800    reading metadata for testdb.testcollection from /data/dump_testdb/testdb/testcollection.metadata.json
2016-11-16T12:13:23.678+0800    restoring testdb.testcollection from /data/dump_testdb/testdb/testcollection.bson
2016-11-16T12:13:23.740+0800    restoring indexes for collection testdb.testcollection from metadata
2016-11-16T12:13:23.740+0800    finished restoring testdb.testcollection (3 documents)
2016-11-16T12:13:23.741+0800    done
#:
```

2. 使用mongoimport进行单个集合导入备份
导入命令如下:

```
mongoimport --host 10.66.187.127:27017 -u mongouser -p thepasswordA1 --authenticationDatabase=admin --db=testdb --collection=testcollection2  --file=/data/export_testdb_testcollection.json
```

3.关于 rwuser 和 mongouser 用户名在写导入命令时的参数说明
在 连接示例 文档有说明，TCloudFinanceZone MongoDB 默认提供了 rwuser 和 mongouser 两个用户名，分别支持 MONGODB-CR 和 SCRAM-SHA-1 两种认证方式。

- 对于 mongouser 以及在控制台创建的所有新用户，在使用导入命令工具时按照上文示例操作即可。
- 对于 rwuser，需要在每个命令里加入参数 --authenticationMechanism=MONGODB-CR。
  用 mongorestore 示例：

```
mongorestore --host 10.66.187.127:27017 -u rwuser -p thepasswordA1 --authenticationDatabase=admin --authenticationMechanism=MONGODB-CR --db=testdb -o /data/dump_testdb
```

为什么数据导入到 MongoDB 实例后，占用空间比自建的 MongoDB 小？

可能存在以下几个原因：

- 原始数据库长时间运行积累了大量的增删改操作。
- 写操作时 MongoDB 出于性能考虑在空间分配时分配了大于实际数据的空间。
- 删除数据后原空间没有被再次利用。
  综合下来导致整个数据库空间的空洞率较高，而导入数据时相当于做了一次类似磁盘整理的操作，使导入后的数据保存得相对紧凑，所以看起来数据变小了。

MongoDB 的 mongodump 无法导出数据，如何处理？

mongodump 使用参见 导入导出，mongodump 工具建议使用3.2.10以上版本。

其他常见问题

MongoDB 云数据库怎么删除，到期后不续费会自动删除吗？

云数据库 MongoDB 实例到期后不续费会自动销毁，请您确认并备份数据。您可在控制台实例列表，选择【操作】>【更多】>【退货退费】执行自助销毁操作。

MongoDB 如何申请安全凭证？

第一次使用云 API 之前，您需要在云 CVM 控制台上申请安全凭证。
安全凭证包括 SecretId 和 SecretKey：

- SecretId：用于标识 API 调用者身份。
- SecretKey：用于加密签名字符串和服务器端验证签名字符串的密钥。

> API 密钥是构建云 API 请求的重要凭证，使用云 API 可以操作您名下的所有云资源，为了您的财产和服务安全，请妥善保存和定期更换密钥，当您更换密钥后，请及时删除旧密钥，详情请参见 签名方法。

MongoDB 用户名使用限制是什么？

实例内建了 rwuser 和 mongouser 两个默认用户。
内建用户的角色为 readWriteAnyDatabase+dbAdmin，即您可以用此用户读写任意数据库，但不具备高危操作的权限。
因 MongoDB 版本而异，部分实例只有 rwuser（对于这批实例会进行升级，升级之前会联系您）。
您也可以使用 MongoDB 控制台进行账号和权限管理以满足您的业务需要，详情参见 限制说明。

通用参考
性能数据

本文主要介绍针对云数据库 MongoDB 实例进行标准化的性能测试，测试得出的性能数据仅供用户参考。

测试环境

- 测试时间：2020年8月。
- 客户端配置：云服务器 CVM 规格为8核32GB。经验证，实例规格较小时，一个8核32GB的 CVM 压测即可将副本实例 CPU 打满，且效果比多个 CVM 更好一些，当一个 CVM 压不到100%时，可用4个 CVM 均摊并发线程数来压测。

测试工具

YCSB 下载地址

测试场景

准备数据约10GB，对于每一种规格，用 YCSB 分别测试 0.5read/130618223492100096.5update 和 0.95read/130618223492100096.05update 场景下的 throughput(ops/sec)、RAL(us) 平均读时延、WAL(us) 平均写时延，主要关注100和200并发两种情况下的性能数据。

时延

CVM 到 MongoDB 实例的平均时延为0.35ms。
时延：Minimum = 0.30ms、Maximum = 0.44ms、Average = 0.35ms

相关命令

1. 准备数据（约10GB）

nohup ./ycsb-0.15.0/bin/ycsb load mongodb -s -P workloads/workloada
-p mongodb.url=mongodb://mongouser:password@10.216.0.30:27017,10.216.0.28:27017,10.216.0.5:27017/admin?w=0 -p table=test -threads 300 -p recordcount=10000000>loadlog.txt &

``` bash
```

2. 0.5read/130618223492100096.5update

nohup ./ycsb-0.15.0/bin/ycsb run mongodb -s -P workloads/workloada -p mongodb.url=mongodb://mongouser: password @10.216.0.30:27017,10.216.0.28:27017,10.216.0.5:27017/admin?w=0 -p table=test -p recordcount=10000000 -p readproportion=0.5 -p updateproportion=0.5 -p insertproportion=0 -p operationcount=100000 -threads 100 >runlog.txt &

3. 0.95read/130618223492100096.05update

nohup ./ycsb-0.15.0/bin/ycsb run mongodb -s -P workloads/workloada -p mongodb.url=mongodb://mongouser: password @10.216.0.30:27017,10.216.0.28:27017,10.216.0.5:27017/admin?w=0 -p table=test -p recordcount=10000000 -p readproportion=0.5 -p updateproportion=0.5 -p insertproportion=0 -p operationcount=100000 -threads 100 >runlog.txt &

> **说明：**
>
> - `-p operationcount=100000`根据具体执行时间动态调整，需要保证执行时间在二十分钟以上，否则时间过短没有代表性。
> - `?w=0`中的 w 表示 write concern。
> - w:1（应答式写入）要求确认操作已经传播到指定的单个 mongod 实例或副本集主实例，缺省为1。
> - w:0（非应答式写入）不返回任何响应，所以无法知道写入是否成功，但对于尝试向已关闭的套接字写入或者网络故障会返回异常信息。
> - w:>1（用于副本集环境）该值用于设定写入节点的数目，包括主节点。

测试数据

读&更新比50:50

MongoDB 规格	threads	throughput(ops/sec)	RAL(us)	WAL(us)	CPU 利用率
2核4GB	100	3188	24091	38254	100%
2核4GB	200	5510	34475	38022	100%
4核8GB	100	7058	8355	19887	100%
4核8GB	200	13590	14391	14983	100%
6核16GB	100	8970	22132	51	100%
6核16GB	200	10041	28696	10966	100%
12核32GB	100	29462	6727	35	100%
12核32GB	200	47815	4673	3681	100%
24核64GB	100	107047	1826	33	100%
24核64GB	200	51046	7802	27	100%
24核128GB	100	130811	1486	32	100%
24核128GB	200	49274	8054	27	100%
32核240GB	100	154253	1254	32	100%
32核240GB	200	52148	8243	1108	100%

|48核512GB|100|174284|1103|28|100%|
|48核512GB|200|121713|3237|32|100%|

读&更新比95:5

MongoDB 规格	threads	throughput(ops/sec)	RAL(us)	WAL(us)	CPU 利用率
2核4GB	100	2738	38216	178	100%
2核4GB	200	10093	20178	11561	100%
4核8GB	100	14380	6864	7631	100%
4核8GB	200	26459	7651	5369	100%
6核16GB	100	13707	7650	56	100%
6核16GB	200	45796	4383	3928	100%
12核32GB	100	115529	902	37	100%
12核32GB	200	56751	3658	31	100%
24核64GB	100	160227	668	29	100%
24核64GB	200	112755	1876	32	100%
24核128GB	100	159130	659	26	100%
24核128GB	200	112993	1936	32	100%
32核240GB	100	167518	634	28	74%
32核240GB	200	172424	1244	35	100%
48核512GB	100	173768	608	31	50%
48核512GB	200	211986	1012	33	85%

词汇表

QPS

每秒查询率 QPS 是对一个特定的查询服务器在规定时间内所处理流量多少的衡量标准。

数据库存储

用于持久化保存数据库数据和日志的底层存储资源。

数据库连接数

连接到数据库实例的客户端会话数。

数据库迁移

随着应用业务的变化，数据库从一个环境迁移到另一个环境。例如从本地 IDC 迁移到云上，或者从某个云迁移到另一个云上。

数据库实例

数据库实例是在云中运行的独立数据库环境，也是云数据库的基本构建数据块。一个数据库实例可以包含由数据库用户创建的多个数据库，并且可以使用与独立数据库实例相同的客户端工具和应用程序进行访问。

数据库引擎

数据库引擎是用于存储、处理和保护数据的核心服务。利用数据库引擎可控制访问权限并快速处理事务，从而满足企业内大多数需要处理大量数据的应用程序的要求。 每个数据库实例均支持数据库引擎。