# 云数据库 MongoDB

# 最佳实践

# 产品文档

腾讯云

【联系我们】

我们致力于为您提供个性化的售前购买咨询服务，及相应的技术售后服务，任何问题请联系 4009100100。

# 文档目录

# 最佳实践
# 分片集群使用指引

最近更新时间：2021-10-13 11:14:43

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，本文重点介绍使用腾讯云 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。
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。

- **避免单调递增或递减**
  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。

**修改分片健值**

MongoDB 4.2 之前的版本，文档的分片键字段值不可变。

从 MongoDB 4.2 版本开始，除非分片键字段是不可变的 _id 字段，否则您可以更新文档的分片键值。若要更新，请使用以下方式来更新文档的分片键值：

| 命令 | 方法 |
| --- | --- |
| update with multi: false | • db.collection.replaceOne()<br>• db.collection.updateOne()<br>• db.collection.update() with multi: false |
| findAndModify | • db.collection.findOneAndReplace()<br>• db.collection.findOneAndUpdate()<br>• db.collection.findAndModify() |
| – | • db.collection.bulkWrite()<br>• Bulk.find.updateOne()<br>如果分片键修改导致将文档移动到另一个分片，则在批量操作中不能指定多个分片键修改；即批量大小为1。<br>如果分片键修改不会导致将文档移动到另一个分片，则可以在批量操作中指定多个分片键修改。 |

修改分片健时需要注意：

- 必须在事务中或以可重试写入方式在 mongos 上运行，不要直接在分片上执行操作。
- 您必须在查询过滤器的完整分片键上包含相等条件。例如，如果一个分片集合内使用 {country：1，userid：1} 作为分片键，要想更新文档的分片键，则必须在查询过滤器中包含 country：，userid：，也可以根据需要在查询中包括其他字段。

## 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 联系我们处理。

- **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 balance 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

最近更新时间：2021-07-20 16:25:35

**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]#
[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

最近更新时间：2022-05-27 16:40:41

通过云服务器 CVM 连接云数据库 MongoDB 可以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 连接实例。

> ⚠ 注意：
> 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。

## 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
```

如下图所示，则执行成功：

```
#: ./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
```

如下图所示，则执行成功：

```
#: ./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
#:
```

### 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 --authenticationDatabase=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 --authenticationDatabase=admin --db=testdb --collection=testcollection2 --file=/data/export_testdb_testcollection.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=admin --authenticationMechanism=MONGODB-CR --db=testdb -o /data/dump_testdb
```