MongoDB最佳实践及问题案例分析

阿里云数据库团队
张友东（林青）

About Me

- Work at Alibaba Cloud
- Focus on Distributed storage & NoSQL
- Project
  - TFS (Taobao File System)
  - ApsaraDB for Redis
  - ApsaraDB for MongoDB (current)

主要内容

- 21分钟开始使用MongoDB
  - 特性、版本、部署、配置、工具、连接
- 那些年，我们曾经踩过的坑
  - 问题案例、优化建议

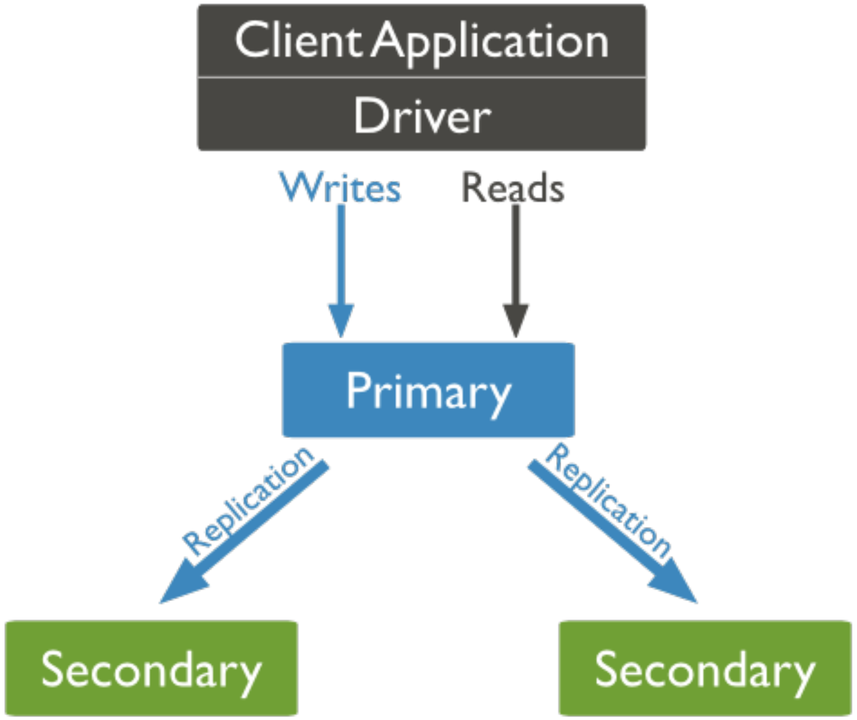
MongoDB特性

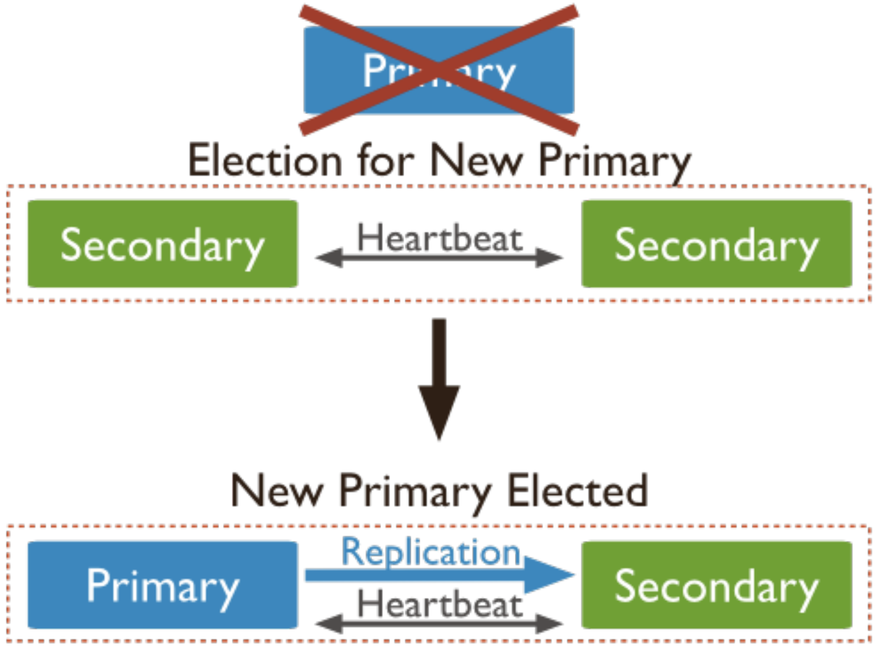
- 文档模型，简单灵活、适合快速开发、迭代场景
- 复制集，保证数据高可靠、服务高可用
- 分片集群，存储容量、服务能力水平扩展
- 功能强大，位置索引、文本索引、TTL索引，GridFS、Aggregation pipeline、MapReduce
- 高性能，wiredtiger、mmapv1、inMemory

版本选择

版本	关键特性	建议
MongoDB 2.x	index、writeConcern、readPreference	强烈建议升级
MongoDB 3.0	Pluggable Storage Engine、Wiredtiger，improved mmapv1	建议升级
MongoDB 3.2	Raft 协议、文档校验 部分索引、inMemory、$lookup	强烈建议使用
MongoDB 3.4	并行复制、sharding迁移改进、collation、$facet、$graghLookup	鼓励尝试

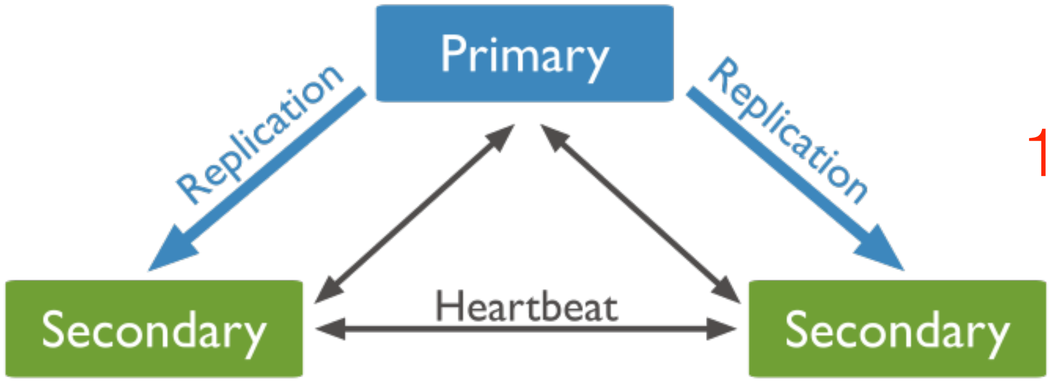
部署形态-复制集



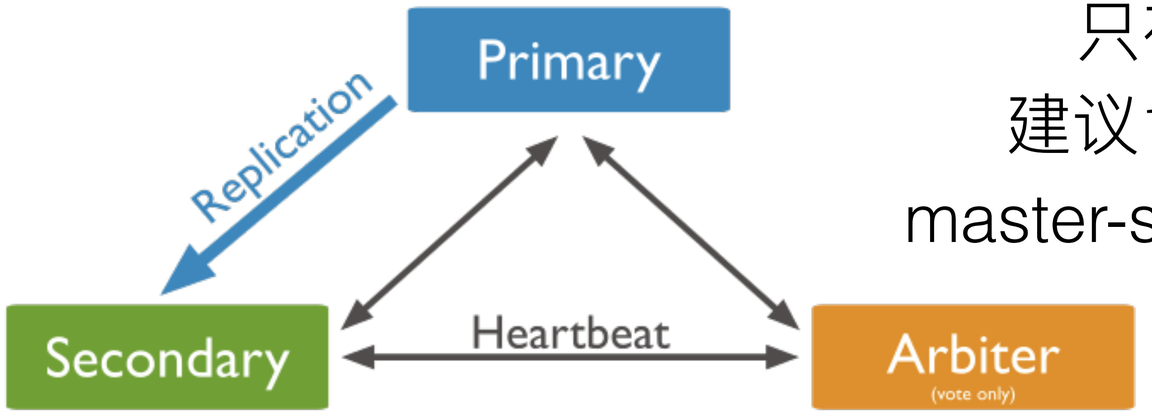


数据高可靠 + 服务高可用

复制集推荐配置



1主2备，强烈推荐

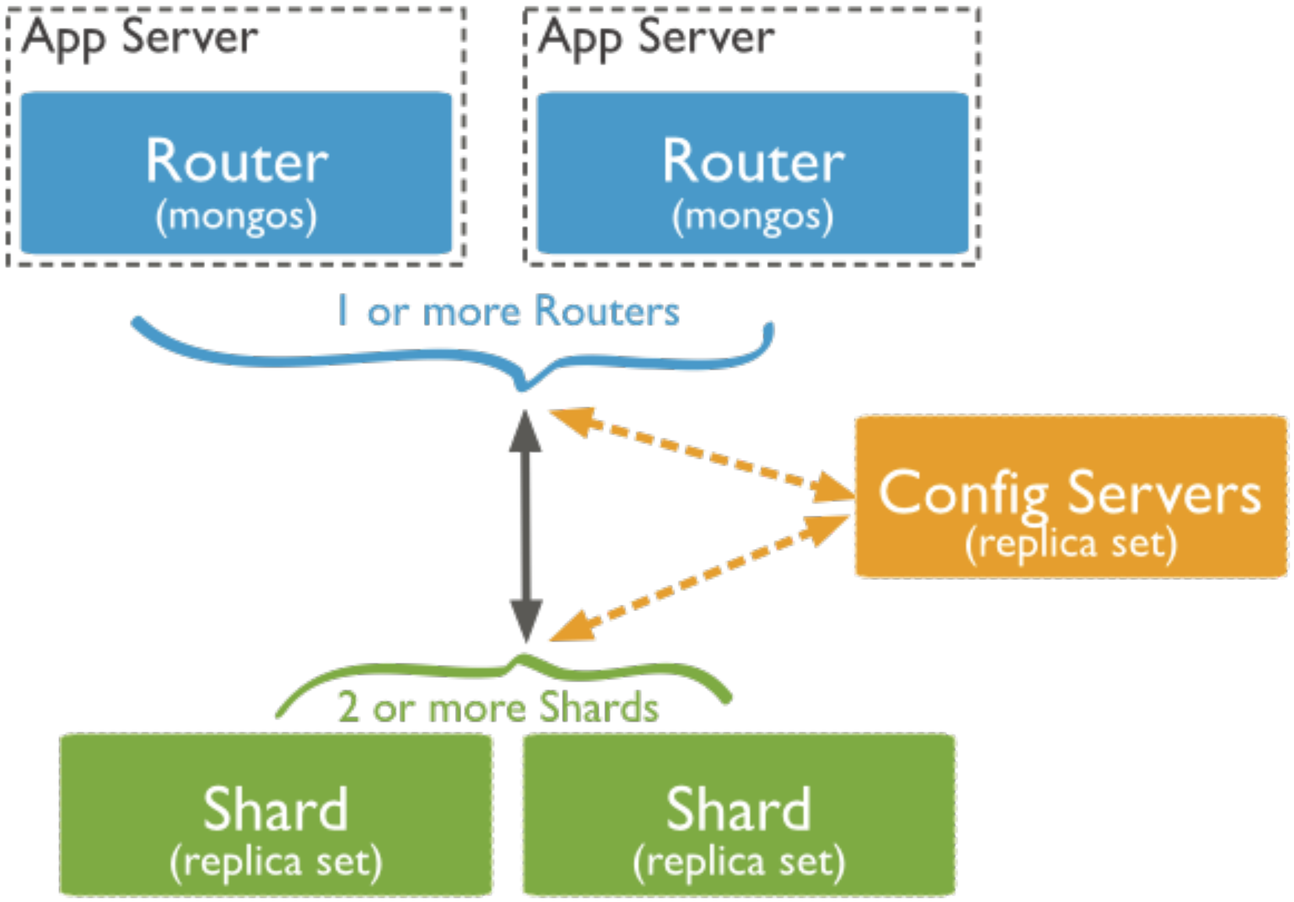


只存2份数据

建议1主1备1仲裁

master-slave基本已废弃

部署形态-分片集群


App Server
Router
(mongos)
App Server
Router
(mongos)
1 or more Routers
Config Servers
(replica set)
2 or more Shards
Shard
(replica set)
Shard
(replica set)

shard key 选择

- 分片方式
  - 范围分片，能很好的支持范围查询
  - hash 分片，读写更好的均分到各个 shard
- shard key选择应结合实际业务需求，需要避免的问题
  - shard key 取值范围太小(low cardinality)
  - shard key 某个值的文档特别多，这样导致单个 chunk 特别大（ jumbo chunk），会影响chunk 迁移及负载均衡。
  - 根据非 shard key 进行查询、更新操作都会变成 scatter-gather 查询，影响效率。

https://yq.aliyun.com/articles/60096

复制集 vs 分片集群

- 分片集群 = M * 复制集 + N * Mongos + Config Server
- 尽量使用复制集，运维管理成本更低
- 当存储容量或者写入能力不足时使用分片集群扩展
- 建议使用MongoDB 3.2及以上版本，config server也是一个复制集，方便统一部署管理
- 部署分片集群时根据容量或请求量预估部署的节点数，先定一个小目标，逐步加节点扩展

https://yq.aliyun.com/articles/60096

重要配置项

重要配置	含义	推荐配置
systemLog.verbosity	日志级别	0
net.maxIncomingConnections	最大连接数	5000以下
security.authorization	是否开启鉴权	true.
storage.directoryPerDB	每个DB一个目录存储	true.
storage.engine	存储引擎	wiredtiger
storage.journal.enabled	是否开启journal	true.
operationProfiling.mode	profiling行为	slowOp
replication.oplogSizeMB	oplog最大占用空间	5% 磁盘空间
sharding.archiveMovedChunks	是否备份迁移的chunk	false.

https://docs.mongodb.com/v3.2/reference/configuration-options/

生态工具

- 官方命令行工具
  - mongo、mongostat、mongotop、mongodump、mongorestore、mongosniff…
- 客户端Driver支持
  - C/C++、java、C#、go、python、node.js、php、perl、ruby
- 可视化管理工具
  - Cloud manager、Compass、MongoClient、adminMongo、robomongo、MongoChef、Aliyun DMS
- 性能测试工具
  - ycsb、sysbench、iibench

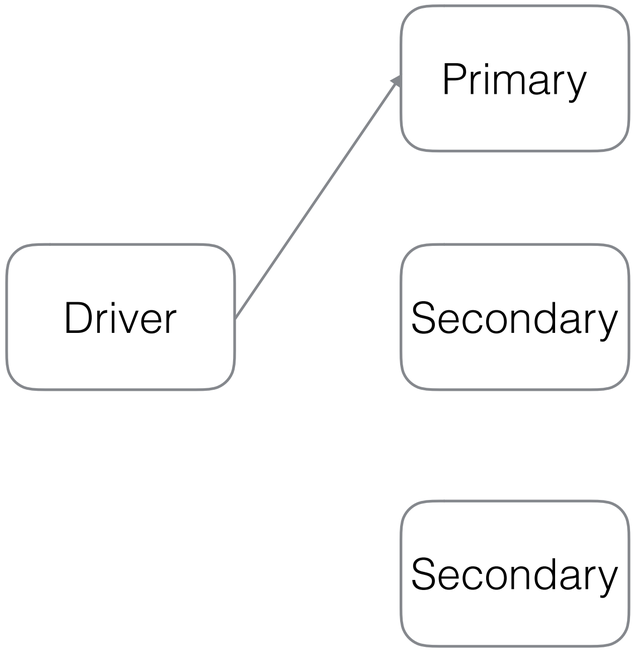
https://docs.mongodb.com/ecosystem/drivers/

连接MongoDB

- 正确连接复制集
- 合理控制连接池大小
- WriteConcern定制写策略
- ReadPreference定制读策略

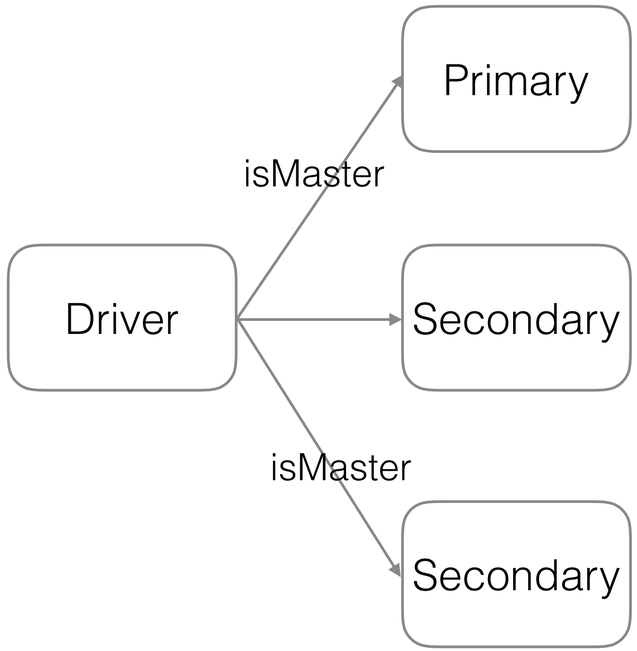
正确连接复制集

直连 Primary



Primary 故障时，不可读写

Driver指定多个节点连接复制集，强烈建议

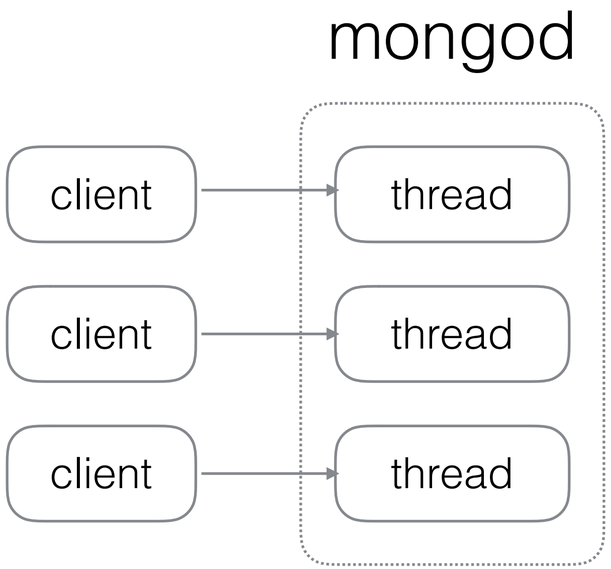


跟所有节点保持心跳，Primary 故障时，
Client 联系新的 Primary 读写

https://yq.aliyun.com/articles/8461

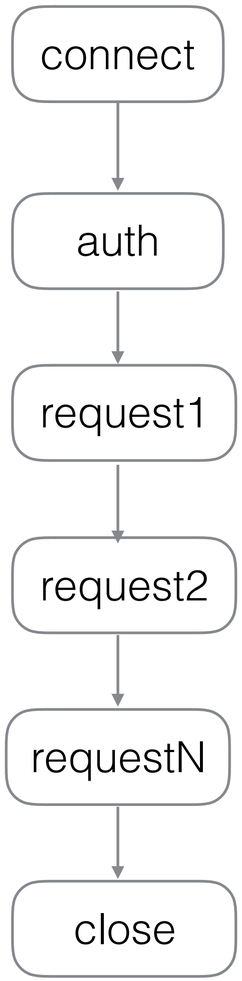
合理配置连接池

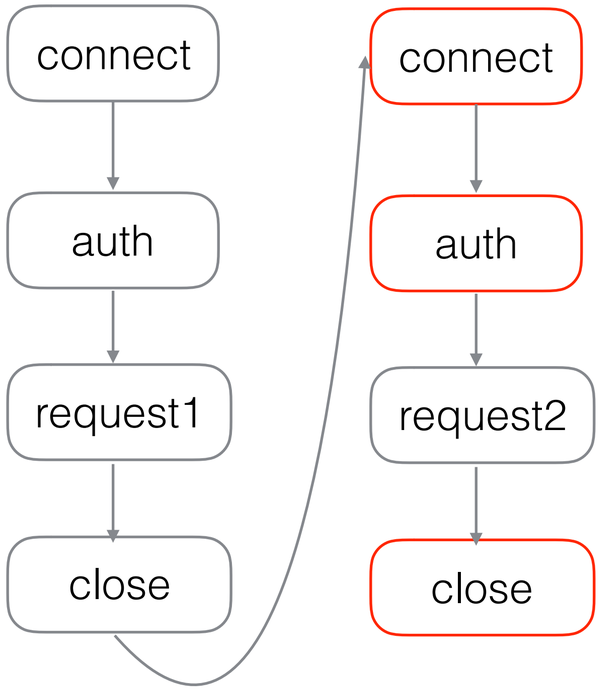
- thread per connection 网络服务模型
  - 每个线程需要1MB 的栈空间
  - 大量连接时，线程切换开销大
- 限制连接数资源
  - mongod 配置 net.maxIncomingConnections 参数
  - Driver通过 Connection String URI 的 maxPoolSize 参数来配置连接池大小



https://yq.aliyun.com/articles/59283

避免使用短连接





- 短连接增加额外的 connect、auth、close、线程创建及销毁开销
- MongoDB 鉴权性能问题，优化后性能提示10+倍

https://yq.aliyun.com/articles/35148

定制写策略

WriteConcern	含 义
{w: 0}	写入不需要server端确认 适合批量写入，不关心正确性场景
{w: 1}	写入到Primary内存向客户端确认，3.x driver的默认行为 后台默认每100ms会保证日志刷盘，每60s数据刷盘
{w: 1, j: 1}	写入到Primary，并且日志刷盘后向客户端确认
{w: “majority”}	写入到复制集大多数节点后向客户端确认 适合可靠性要求非常高的场景，性能会下降

https://yq.aliyun.com/articles/54367

定制读策略

readPreferecne	含义
primary	driver默认行为，所有请求都从primary读取
primaryPreferred	优先从primary读取 无primary时，从secondary读取
secondary	所有请求都从secondary读取
secondaryPreferred	优先从secondary读取 无secondary成员时，从primary读取
nearest	根据网络距离就近读取

https://yq.aliyun.com/articles/60553


云数据库MongoDB版
返回实例列表
https://www.aliyun.com/product/mongodb
包年包月
按量付费
基本配置
地域:
华东 1
华北 2
华东 2
华南 1
华北 1
可用区:
华东 1 可用区 B
数据库版本:
MongoDB 3.2
网络
网络类型:
经典网络
专有网络
规格配置
规格:
1核2G
2核4G
4核8G
8核16G
8核32G
16核64G
最大连接数: 1000 IOPS: 3200
存储空间:
500GB
1000GB
2000GB
500
GB

干掉长(慢)请求

- 客户端发起耗时请求，如遍历集合、建索引、mapreduce、aggregation 等，主动断开连接后，后端的请求仍然在执行
- currentOp + killOp 中止后端正在运行的请求

killOp原理：长时间执行的请求会设置检查点，发现有 kill 操作就会主动退出

killOp 后，请求执行到checkForInterrupt()就会退出

```
while (!createIndexFinished) {

    createIndexForOneElement();
    checkForInterrupt();
}
```

https://yq.aliyun.com/articles/6647

尽量后台建索引

- MongoDB 默认前台建索引，效率更高、索引体积更小
- Primary
    - 前台：DB 写锁，阻塞 DB 上的所有读写
    - 后台：DB 写意向锁，对读写无影响
- Secondary
    - 前台：整个建索引过程中，阻塞所有请求，包括鉴权
    - 后台：读请求不阻塞
- 建议
    - 尽量在创建集合时，规划好索引，在集合为空的时候就创建索引
    - 针对已有大量数据的集合，尽量后台建索引

https://yq.aliyun.com/articles/55821

控制集合数量

- wiredtiger引擎特性
  - 每个集合对应一个物理文件，每个索引对应一个物理文件
  - listDatabases 时，需要遍历所有的集合及索引，逐个获取物理文件大小信息
- 问题及优化
  - 物理文件太多，数据库管理开销增加，影响性能，建议启用 storage.directoryPerDB选项，尽量让物理文件分散到多个目录。
  - listDatabases 开销太大，导致监控系统无法正常工作，还可能影响到主备同步（全量同步时，会先 listDatabases 拉取 DB 列表，设置的超时时间为30s）
  - 如果一定需要大量的集合，可考虑使用 mmapv1或 rocksdb 引擎

https://yq.aliyun.com/articles/58531

避免wiredtiger hang

- 内存写入与磁盘速度差异，导致内存的数据积累太多，导致应用线程参与page evict，引发请求hang
- 升级到3.2.10+
- wiredtiger 数据与日志分开存储
- SATA 盘升级到 SSD
- eviction 参数调优

https://yq.aliyun.com/articles/66554

位置查询内存优化

- $near位置查询，cursor 会缓存结果数据，可能占用大量内存，默认10分钟后 cursor 超时释放
- 优化方法
  - 设置更小的 cursor timeout
  - 如无需遍历，find 时设置 singleBatch 选项

$near位置查询示例

```
{
    "find" : "userData",
    "filter" : {
        "latlng" : {
            "$near" : [
                116.34642045073839,
                39.87082232130999
            ],
            "$maxDistance" : 0.90        }
    },
    "ntoreturn" : 1000
}
```

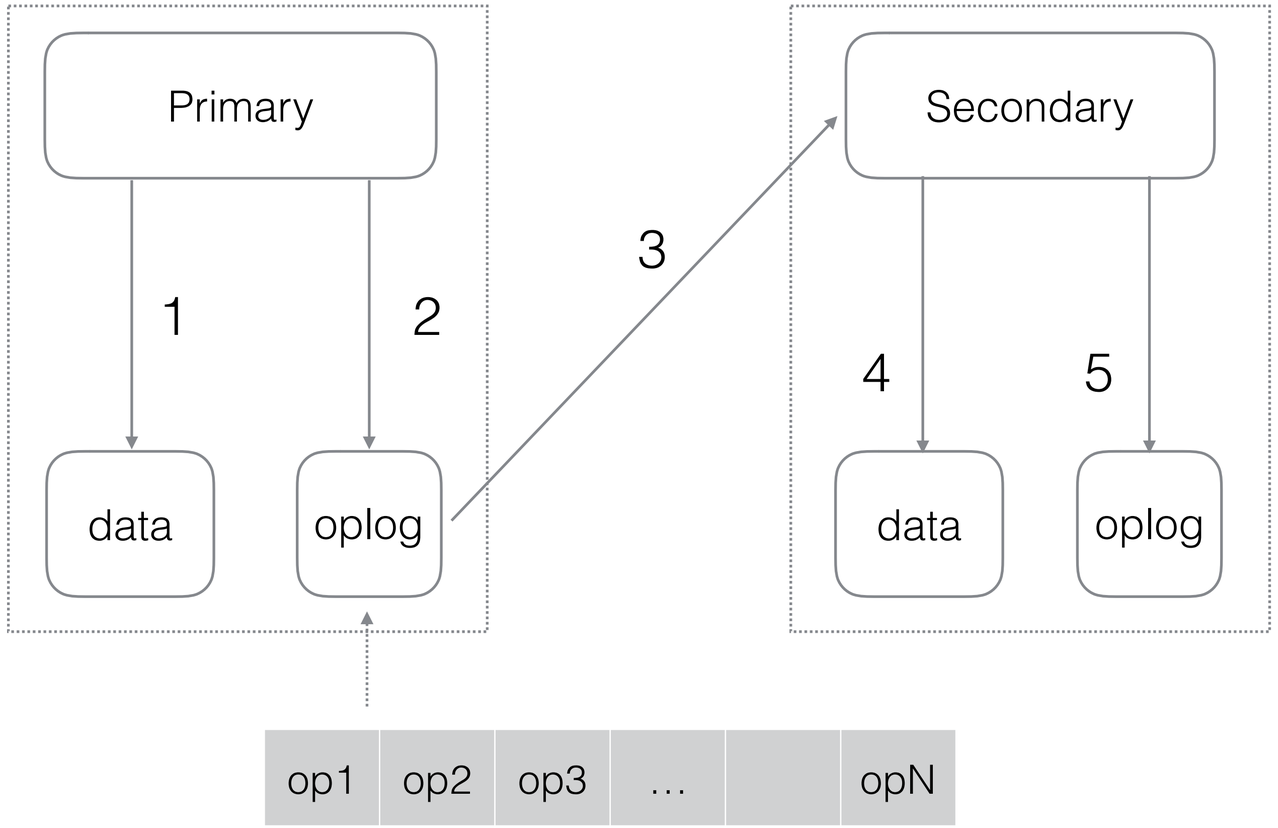
https://yq.aliyun.com/articles/58532

避免OOM

- 存储引擎的cacheSizeGB
- 网络连接管理、主备同步buffer
- 请求临时数据、cursor、sort
- 内存碎片（memory fragmentation）
- tcmalloc freelist cache

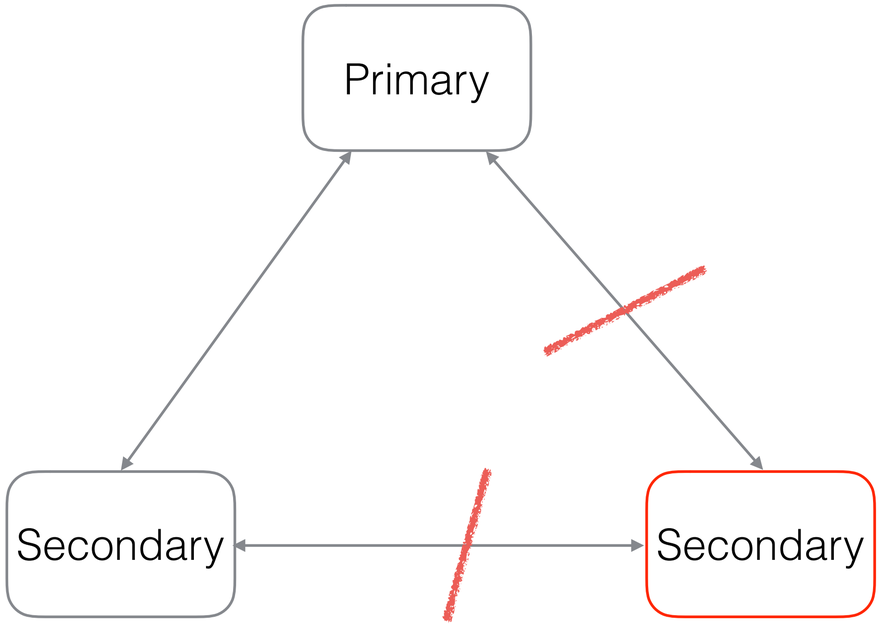
增加可用内存 or 调小cacheSizeGB + 降低并发

关于 oplog



oplog是固定大小集合，按时间戳排序，满了时自动删除最老的数据

oplog 问题 (1)

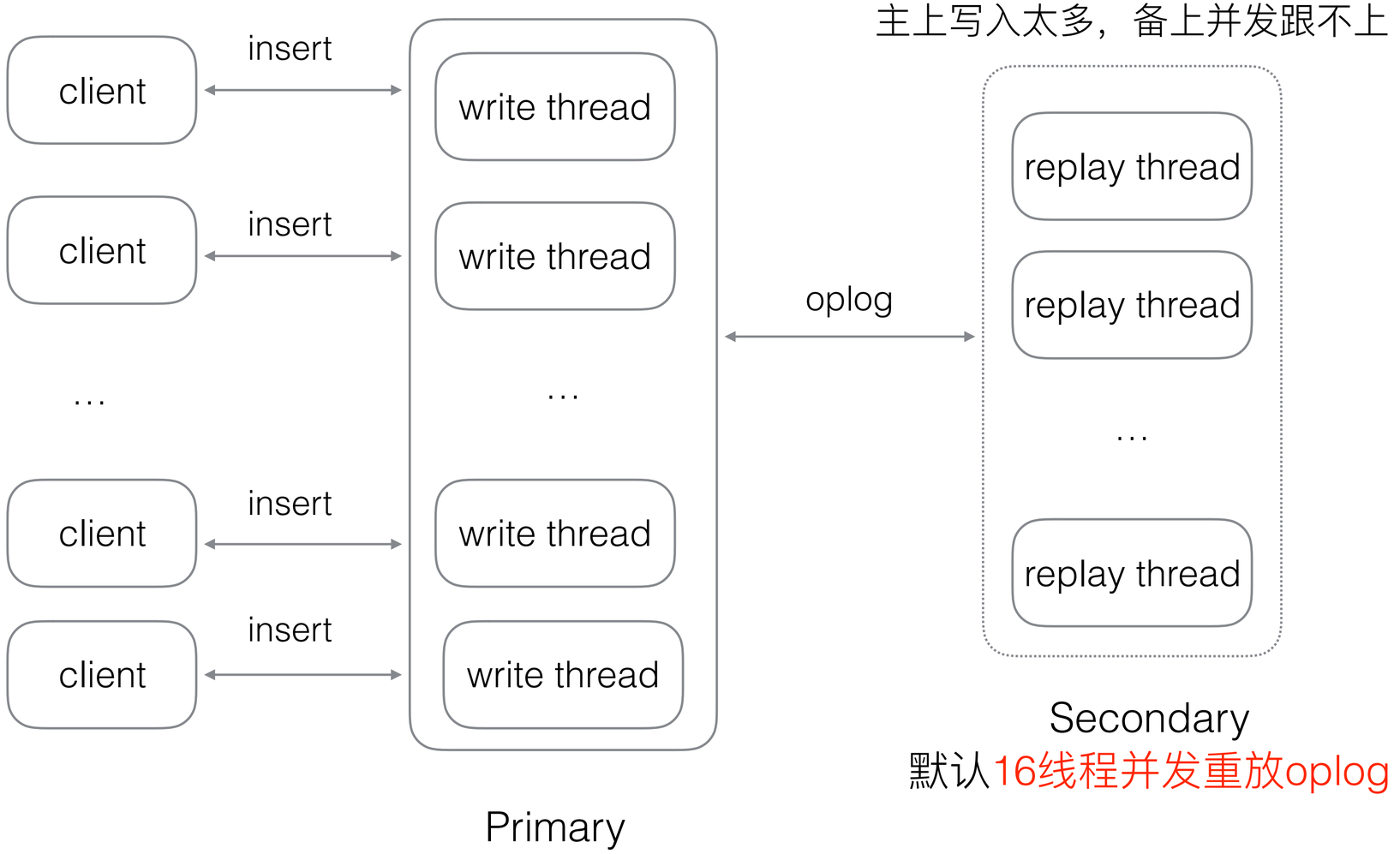


- Secondary 网络断开，或者停机维护，恢复时可能因 oplog 不足无法继续同步
- 停机维护尽量选择在业务低峰期来做

阿里云
云数据库 MongoDB 版

oplog 问题 (2)

主上写入太多，备上并发跟不上



默认16线程并发重放oplog

oplog 问题 (3)

```
mongo:PRIMARY> db.coll.find()
{ "_id" : 1, "x" : [ 1, 2, 3, 4, 5 ] }
mongo:PRIMARY> db.coll.update({_id: 1}, {$push: {x: { $each: [6, 7], $position: 0 }}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
mongo:PRIMARY> db.coll.find()
{ "_id" : 1, "x" : [ 6, 7, 1, 2, 3, 4, 5 ] }
mongo:PRIMARY> use local
switched to db local
mongo:PRIMARY> db.oplog.rs.find().sort({$natural: -1}).limit(1)
{ "ts" : Timestamp(1464082056, 1), "h" : NumberLong("6563273714951530720"),
  "v" : 2, "op" : "u", "ns" : "test.coll", "o2" : { "_id" : 1 },
  "o" : { "$set" : { "x" : [ 6, 7, 1, 2, 3, 4, 5 ] } }
}
```

- 为保证oplog 重放幂等性，往数组开始位置$push操作被转换为$set操作，并且带上整个数组的内容，数组较大时，同步网络流量被严重放大，影响主备同步
- 使用数组时，数组元素尽量不要太多，尽量使用$set 或 在末尾$push

https://yq.aliyun.com/articles/52404

oplog 问题（4）

```
mongo:PRIMARY> db.coll.find()
{ "_id" : ObjectId("57fcd82a66e32bd8f589c3f4"), "x" : 1 }
{ "_id" : ObjectId("57fcd82b66e32bd8f589c3f5"), "x" : 1 }
mongo:PRIMARY> db.coll.update({x: 1}, {$set: {x: 2}}, {multi: true})
WriteResult({ "nMatched" : 2, "nUpserted" : 0, "nModified" : 2 })
mongo:PRIMARY> use local
switched to db local
mongo:PRIMARY> db.oplog.rs.find().sort({$natural: -1}).limit(2)
{ "ts" : Timestamp(1476188286, 2), "t" : NumberLong(23),
"h" : NumberLong("-8689997572803504719"), "v" : 2, "op" : "u", "ns" : "test.coll",
"o2" : { "_id" : ObjectId("57fcd82b66e32bd8f589c3f5") }, "o" : { "$set" : { "x" : 2 } }
{ "ts" : Timestamp(1476188286, 1), "t" : NumberLong(23),
"h" : NumberLong("-3016675555126288909"), "v" : 2, "op" : "u", "ns" : "test.coll",
"o2" : { "_id" : ObjectId("57fcd82a66e32bd8f589c3f4") }, "o" : { "$set" : { "x" : 2 } }
```

- Primary 上一次性 update 或 remove 多个文档时，每个文档会对应一条 oplog，Secondary 上重放每条 oplog 来保持跟 Primary 数据一致，但开销却比 Primary 大，这样的操作很多时，可能导致备同步无法跟上。

oplog 问题（5）

- mongodump —oplog 因数据集太大，备份时间过长，结束时最初的 oplog 已经被删除，导致全量备份失败
- 使用 tailable cursor不断抓取 oplog（增量备份、或者跨机房同步），如果抓取速度跟不上写入速度，可能导致需要的 oplog 已经被删除，抓取的增量数据不全

如何管理 oplog?

- 支持在线动态修改 oplog 大小
  - db.runCommand({collMod: “oplog.rs”, maxSize: 1024000000})
- 支持复制集成员根据同步进度自适应管理 oplog
- 支持设置一个 oplog 保护时间戳，所有超过该时间戳的 oplog都会自动保留，支持全量、增量备份
  - db.runCommand({collMod: “oplog.rs”, oplogDeleteGuard: 1400000000})

https://yq.aliyun.com/articles/54392

更多案例

- 复制集脑裂问题 （3.2 raft解决）
- MongoDB Secondary 延时高问题
- see https://yq.aliyun.com/users/1134812

学习MongoDB

- 官网文档 https://docs.mongodb.com/
- MongoDB university https://university.mongodb.com/
- MongoDB中文社区 www.mongoing.com
- 阿里云栖社区 https://yq.aliyun.com/groups/11
- 欢迎与我交流 zyd_com@126.com