打包与配置

2015年10月22日 15:04

1. 下载代码：git clone https://github.com/tgou/RocketMQ
2. 打包：mvn -Dmaven.test.skip=true clean package install assembly:assembly -U
3. 普通配置 - 该配置都写在properties文件中
   a. broker配置
      i. rocketmqHome=/home/admin/XX 可选）
         1) 该配置首先从java属性rocketmq.home.dir中获取，如果没有获取到从系统环境变量ROCKETMQ_HOME中获取
         2) 虽然该配置是可选，但是前提是已经配置了rocketmq.home.dir或者ROCKETMQ_HOME
      ii. namesrvAddr=127.0.0.1:9876;192.168.0.3:9876（可选）
         1) 该配置首先从java属性rocketmq.namesrv.addr中获取，如果没有获取到从系统环境变量NAMESRV_ADDR中获取
         2) 该配置也可以从启动broker的参数中配置：-n "127.0.0.1:9876"
         3) 虽然该配置是可选，但是前提是已配置上述三种方法之一
      iii. brokerIP1=127.0.0.1（可选）
         1) 该配置默认从系统可用地址中选择一个
         2) 在某些场景可以手动配置ip，例如程序运行在虚拟机中外部无法访问默认地址，或者用来解决docker本地ip外部无法访问的问题
      iv. brokerIP2=127.0.0.1（可选）
         1) broker的ha地址，其它用途同上，不过一般不需要配置
      v. brokerName=broker_aaa_bb（必填）
         1) 该选项虽然有默认值，但是线上环境最好填写，要不然集群注册会有问题
         2) 属于同一个主备配置的broker的brokerName要一样
         3) 该选项默认取当前机器地址，如果取不到默认是"DEFAULT_BROKER"
      vi. brokerClusterName=DEFAULT_CLUSTER（可选）
         1) 该选项默认是"DEFAULT_CLUSTER"
         2) 当需要通过同一个namesrv管理多个集群的时候，不同集群配置不同的值
      vii. brokerId=0（必填）
         1) 该选项默认是0，代表主
         2) 当配置主备的时候，备库需要递增，例如1,2等
      viii. defaultTopicQueueNums=8（可选）
         1) 默认8，表示默认为每个topic创建的queue的数量
      ix. autoCreateTopicEnable=true
         1) 当topic不存在的时候自动创建topic，默认为true
         2) 线上最好关闭，有利于管理topic
      x. autoCreateSubscriptionGroup=true
         1) 当订阅组不存在的时候，自动创建，默认为true
         2) 线上最好关闭，便于管理消息订阅组
      xi. rejectTransactionMessage=false
         1) 是否要拒绝事务消息，默认为false
         2) 当broker不希望支持事务消息的时候，可以设置为true
      xii. fetchNamesrvAddrByAddressServer=false
         1) 是否通过域名系统获得namesrv的地址，默认为false
         2) 当为了提高namesrv地址的灵活性，可以设置为true，当打开该选项的时候，上边namesrvAddr配置中所述的都可以不配置
   b. store存储配置
      i. storePathRootDir=/home/admin/store（可选）
         1) 默认地址为java属性的user.home + "store"，也就是存储在当前用户目录的store目录下
      ii. storePathCommitLog=/home/admin/store/commitlog
         1) 默认地址为当前用户目录user.home + "store" + "commitlog"
      iii. flushIntervalCommitLog=1000（可选）
         1) commitlog刷盘的间隔，默认为1000毫秒，即1秒
      iv. flushCommitLogTimed=false（可选）
         1) 是否是定时刷盘，默认为false，也就是实时刷盘，实时刷盘是指有数据写入就会触发刷盘逻辑，如果满足刷页条件就刷盘
      v. deleteWhen="04"（可选）
         1) 何时触发删除文件，默认是凌晨4点删除文件
         2) 该时间是服务器时间，配置服务器压力最低的时间就可以
      vi. fileReservedTime=72（可选）
         1) 文件保留时间，单位小时
      vii. maxMessageSize=524288（可选）
         1) 默认值是1024 * 512 = 524288，也就是512k
      viii. maxTransferBytesOnMessageInMemory=（可选）
         1) 最大被拉取的消息字节数，消息在内存，默认256k
      ix. maxTransferCountOnMessageInMemory=32（可选）
         1) 最大被拉取的消息个数，消息在内存，默认32个
      x. maxTransferBytesOnMessageInDisk=（可选）
         1) 最大被拉取的消息字节数，消息在磁盘，默认64k
      xi. maxTransferCountOnMessageInDisk=8（可选）
         1) 最大被拉取的消息个数，消息在磁盘，默认8个
      xii. accessMessageInMemoryMaxRatio=40（可选）
         1) 命中消息在内存的最大比例
      xiii. messageIndexEnable=true（可选）
         1) 是否开启消息索引功能
      xiv. messageIndexSafe=false

1) 是否使用安全的消息索引功能，即可靠模式。可靠模式下，异常宕机恢复慢，非可靠模式下，异常宕机恢复快

xv. haMasterAddress=127.0.0.1（可选）

1) 如果不设置，则从NameServer获取Master HA服务地址

xvi. brokerRole=ASYNC_MASTER（可选）

1) 默认是ASYNC_MASTER，异步复制master
2) 还有SYNC_MASTER - 同步双写master；SLAVE - slave服务器

xvii. flushDiskType=ASYNC_FLUSH（可选）

1) 默认是ASYNC_FLUSH，异步刷盘
2) 还有SYNC_FLUSH，同步刷盘

xviii. cleanFileForciblyEnable=true

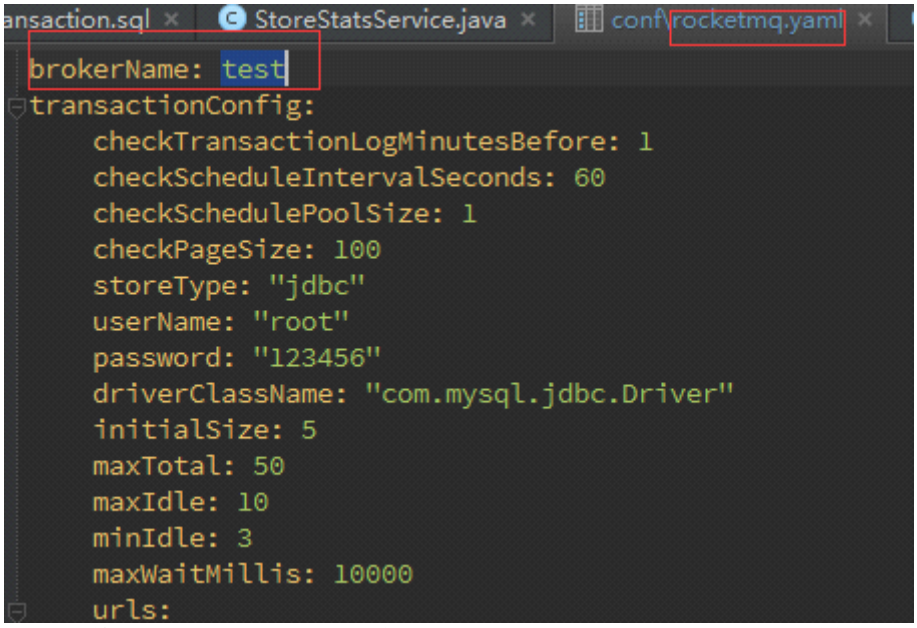
1) 磁盘空间超过90%警戒水位，自动开始删除文件

安装

2015年10月22日 15:12

1. 设置环境变量
   a. export ROCKETMQ_HOME=/home/rocketmq
   b. 将bin目录下的文件设置为可执行，chmod +x mq*
2. 启动namesrv
   a. 配置jvm参数，在runserver.sh中
   b. 运行 nohup mqnamesrv&
   c. **线上日志要配置监控**
3. 启动broker
   a. 配置jvm参数，在runbroker.sh中
   b. 调整配置文件，根据需求是多m还是ms结构，其中nameSrv的地址可以配置在配置文件中，也可以写到运行命令中
   c. 运行 nohup sh mqbroker -n "127.0.0.1:9876" -c ../conf/2m-noslave/broker-a.properties &
   d. 事务启动运行 nohup sh mqbroker -n "127.0.0.1:9876" -c ../conf/2m-noslave/broker-a.properties -tc ../conf/rocketmq.yaml &
   e. 其中事务消息需要去数据库建表

```
CREATE TABLE `t_transaction` (
  `id` bigint(20) unsigned NOT NULL AUTO_INCREMENT,
  `broker_name` varchar(50) NOT NULL,
  `offset` bigint(20) unsigned NOT NULL,
  `producer_group` varchar(50) NOT NULL,
  `gmt_create` datetime NOT NULL,
  PRIMARY KEY (`id`),
  KEY `bpo_idx` (`broker_name`,`producer_group`,`offset`),
  KEY `bpg_idx` (`broker_name`,`producer_group`,`gmt_create`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8
```



也需要修改rocketmq.yaml配置中的brokerName，跟对应的broker一样

测试

2015年10月22日 16:57

1. 运行例子
    a. sh ${ROCKETMQ_HOME}/bin/tools.sh com.alibaba.rocketmq.example.quickstart.Producer
    b. sh ${ROCKETMQ_HOME}/bin/tools.sh com.alibaba.rocketmq.example.transaction. TransactionProducer
    c. sh ${ROCKETMQ_HOME}/bin/tools.sh com.alibaba.rocketmq.example.quickstart.Consumer

运维

2015年10月22日 17:09

1. The most commonly used mqadmin commands are:
   updateTopic Update or create topic
   deleteTopic Delete topic from broker and NameServer.
   updateSubGroup Update or create subscription group
   deleteSubGroup Delete subscription group from broker.
   updateBrokerConfig Update broker's config
   topicRoute Examine topic route info
   topicStatus Examine topic Status info
   brokerStatus Fetch broker runtime status data
   queryMsgById Query Message by Id
   queryMsgByKey Query Message by Key
   queryMsgByOffset Query Message by offset
   printMsg Print Message Detail
   producerConnection Query producer's socket connection and client version
   consumerConnection Query consumer's socket connection, client version and subscription
   consumerProgress Query consumers's progress, speed
   consumerStatus Query consumer's internal data structure
   cloneGroupOffset clone offset from other group.
   clusterList List all of clusters
   topicList Fetch all topic list from name server
   updateKvConfig Create or update KV config.
   deleteKvConfig Delete KV config.
   wipeWritePerm Wipe write perm of broker in all name server
   resetOffsetByTime Reset consumer offset by timestamp(without client restart).
   updateOrderConf Create or update or delete order conf
   cleanExpiredCQ Clean expired ConsumeQueue on broker.
   startMonitoring Start Monitoring
   checkMsg Check Message Store
   statsAll Topic and Consumer tps stats
   syncDocs Synchronize wiki and issue to github.com

   See 'mqadmin help <command>' for more information on a specific command.

2. 按照msgId查询消息：mqadmin queryMsgById -i 0A00020F00002A9F000000000000D352-n 127.0.0.1:9876
3. 安装web管理界面
   a. 下载代码：git clone https://github.com/tgou/rocketmq-console
   b. 配置config.properties中的namesrv地址
   c. 打包：mvn clean package -Dmaven.test.skip=true
   d. 放到web容器中运行

事务设计

2015年10月19日 16:12

1. 事务日志表设计

属性名	类型	长度	备注
id	long		表主键，应用中不使用
broker_name	varchar	50	配置文件里的brokerName，一定要配置，而且不要改
offset	bigint		broker内commitLog的offset
producer_group	varchar	50	生产者配置的group，不要轻易修改，如果更改也要进行版本兼容，要不然事务回查会有问题
gmt_create	datetime		日志创建时间

2. 设计要点

- prepare事务日志插入
  - dispatch的时候如果是prepare的事务消息，插入db（可以批量提高性能，需要考虑db不可用如何处理）
  - 位置在DefaultMessageStore的doDispatch方法中
- rollback事务日志删除
  - dispatch的时候如果是rollback事务消息，删除db事务日志
  - 位置在DefaultMessageStore的doDispatch方法中
- commit事务日志删除
  - 同上
- check事务回查
  - 遍历当前broker的XX秒之内的prepare事务日志，并通过offset拿到该消息的详情，通过producer_group拿到机器ip进行回查
  - 每个broker单线程定时程序回查prepare事务日志
- 事务日志redo
  - 由于requestDispatch中的消息消费顺序跟commitLog的顺序是一样的，所以在每次事务日志成功写入db之后更新当前事务日志的checkpoint（此处可以进行批量处理之后再记录checkpoint）
  - 事务日志回滚的时候，拿到所有checkpoint的最小值进行recovery，重新走dispatch流程
- 事务消息恢复
  - 由于recover消息的代码是在BrokerController的load期间进行的，所以初始化db的代码要在load的时候加载，而不能放到start的时候加载