# 携程异步消息系统实践

携程框架研发部
顾庆

# About Hermes and Me

- 携程、大众点评、百度
  - 框架、中间件、架构
- Hermes
  - 2014.12-
  - 携程消息系统

# Agenda

- 消息队列的优势
- Hermes的整体架构
- 存储设计
- 基于Lease的集群管理

# MQ有广泛的使用场景

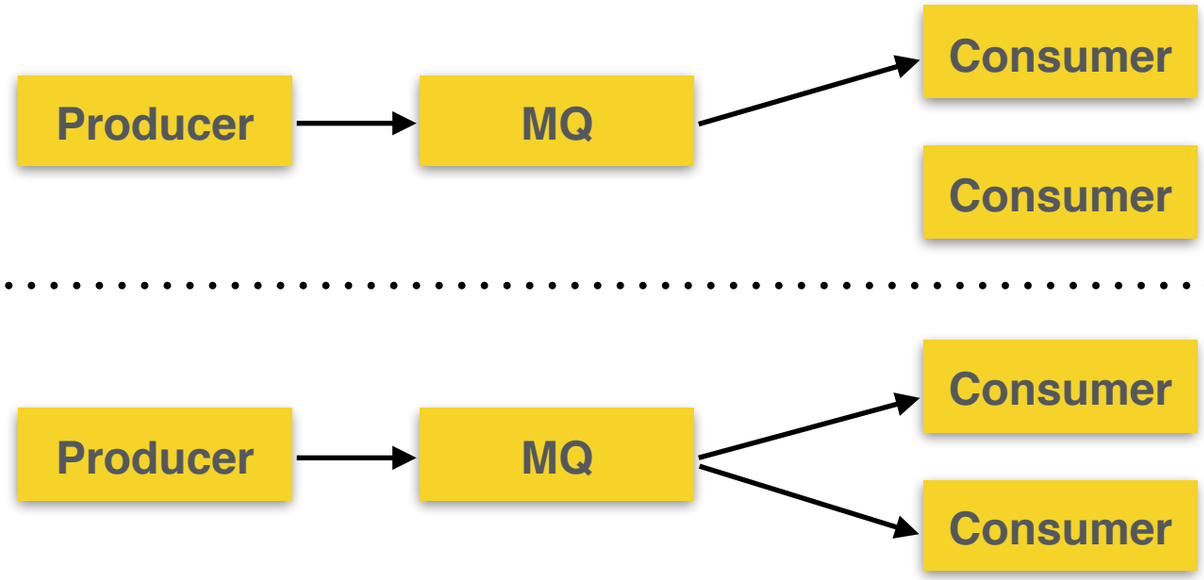
- 索引实时更新


- 支付

# MQ的特点

- 降低系统间的耦合度
  - 异步处理
  - 抵御流量波峰
- 支持大Fan-out

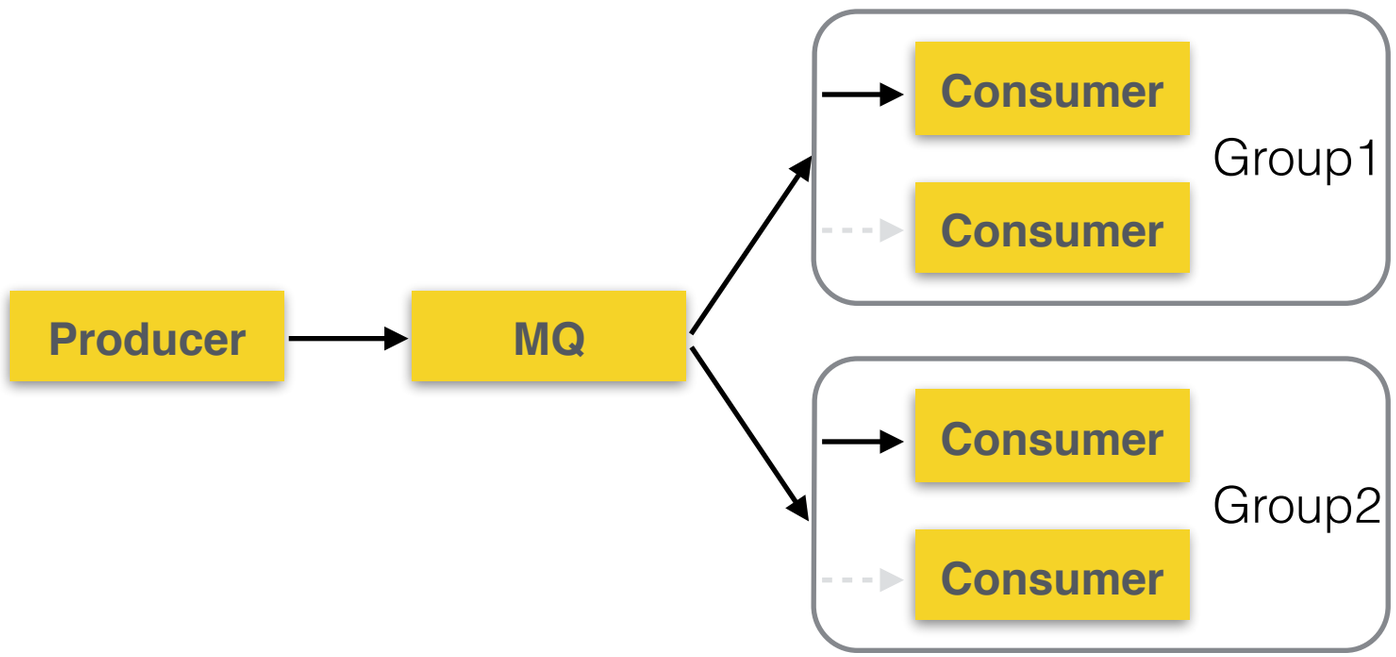
# MQ的基本模型




Queue

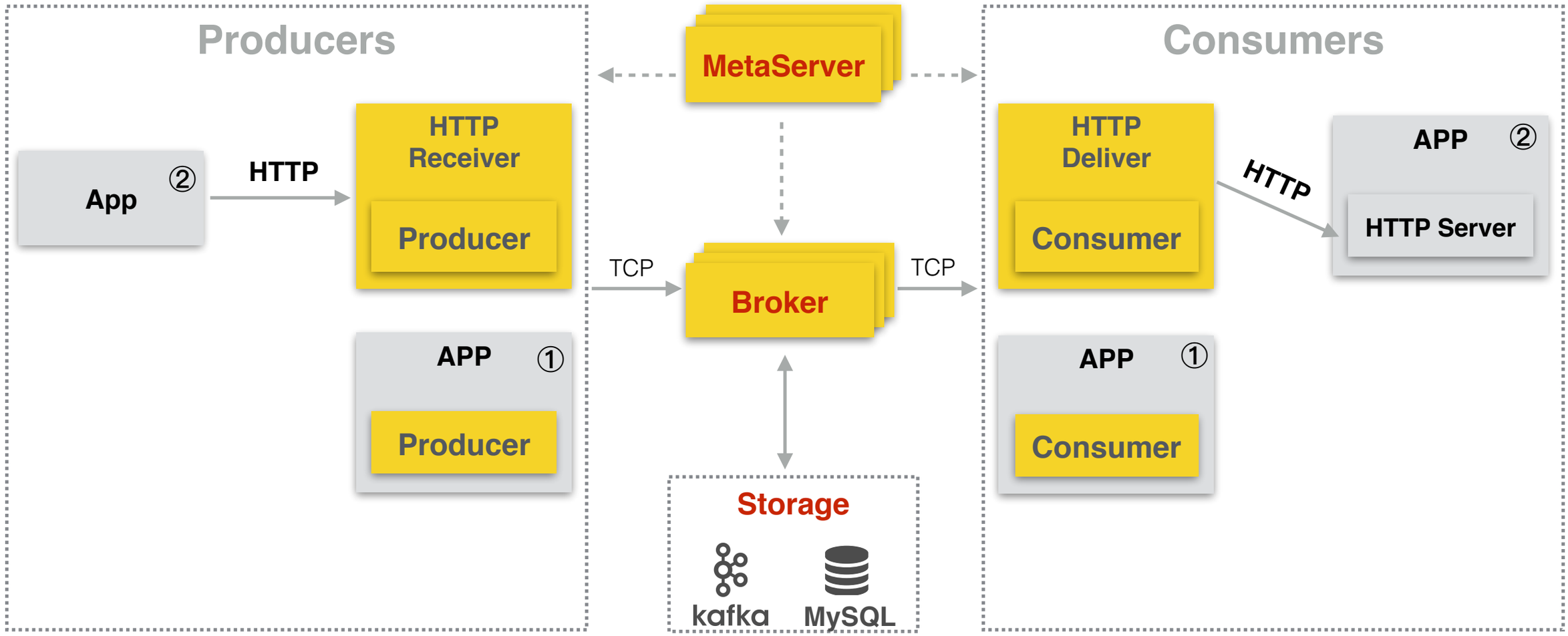



Topic




**Consumer Group**

# Hermes整体架构




① native客户端(Java/C#)　② HTTP客户端

控制流　数据流　Hermes　App

# 两种消息类型

- Hermes-Kafka
  - 高吞吐、高性能
  - 不支持高级的消息队列特性
  - Broker采用ZeroCopy，无法进行深入监控

# 两种消息类型

- Hermes-MySQL
  - 性能足够支撑绝大多数业务
  - 丰富的消息队列特性支持
  - 可以为个性化的业务需求进行定制
  - 更全面和深入的监控治理

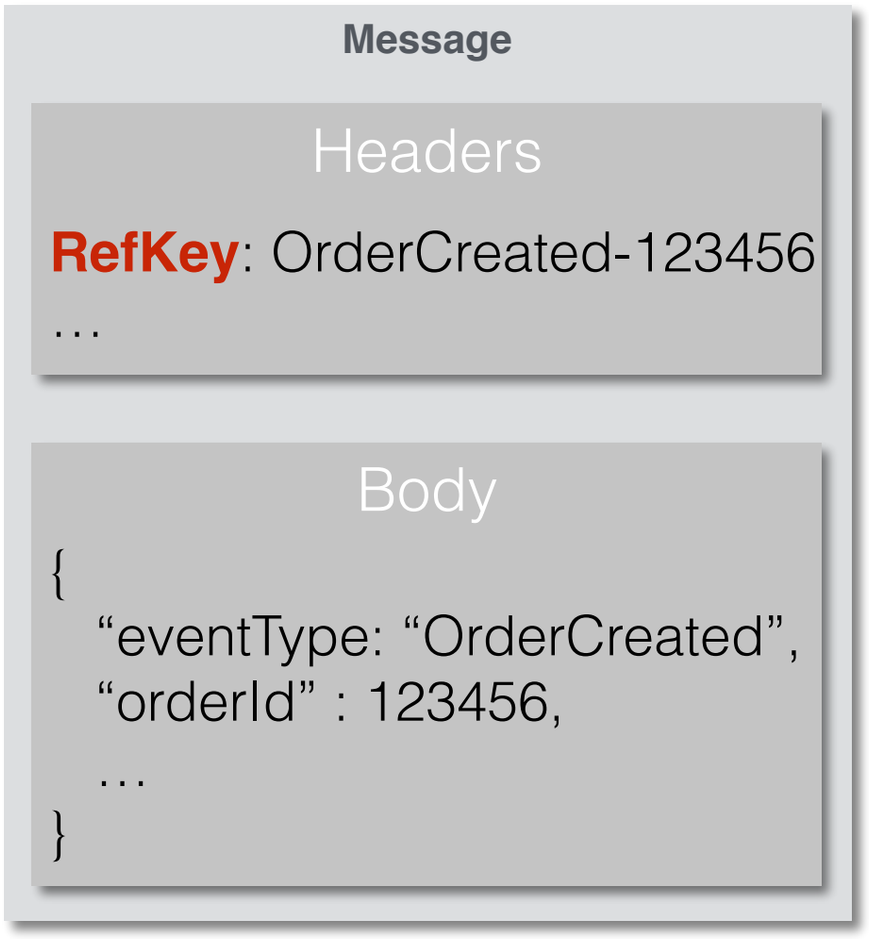
# MQ运营常见问题

有条消息好像没收到，帮我查一下

什么消息?

消息里面有个订单号123456

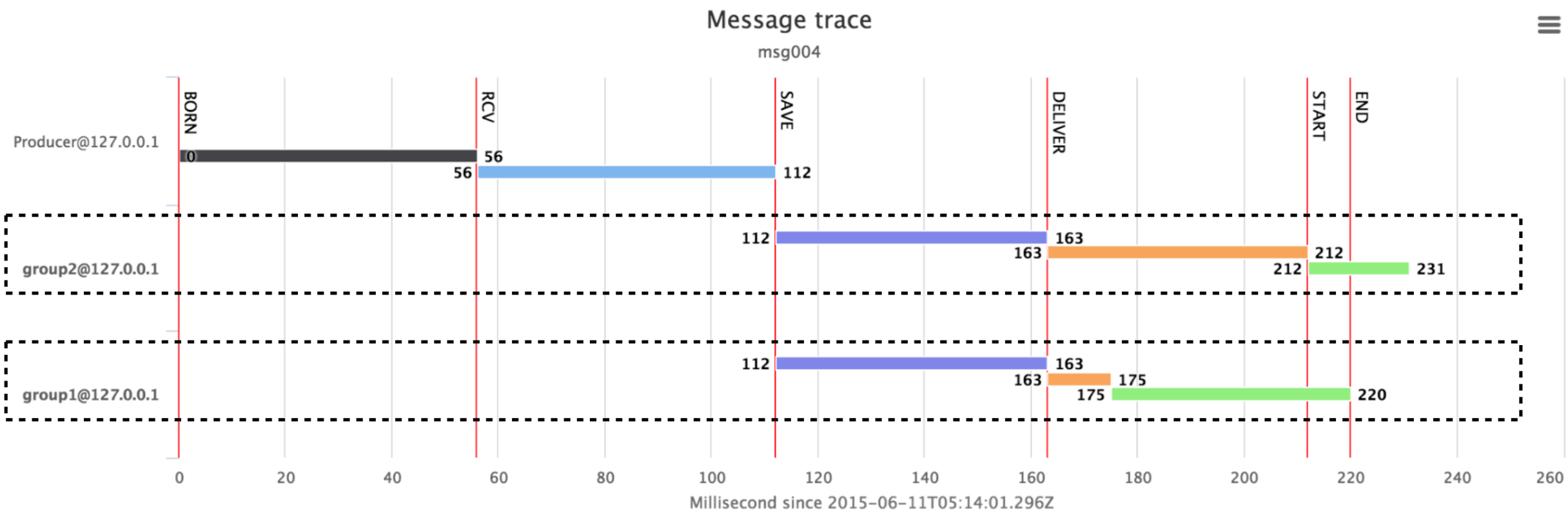
# 消息的ReferenceKey




- 消息在MQ中的“业务ID”
- 和消息一一对应
- 可追踪某条消息在MQ中的所有事件
  - 产生、存储、消费

# 消息追踪

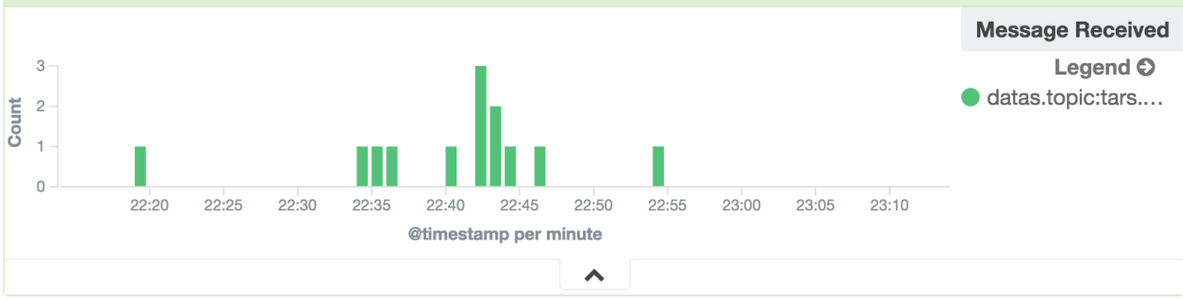
输入 **Ref-Key** msg004 06/11/2015 提交



# 全面的监控治理

Ctrip 携程

生产速度排行 | 消费速度排行 | 消费耗时排行 | 最新消息




Consumer 消费延时

| Consumer 名称 | Partition ID | 延迟数量 | 最新优先消息ID | 消费至 | 最近消费内容 | 最新非优先消息ID | 消费至 | 最近消费内容 | 当前租约分配ip |
|---|---|---|---|---|---|---|---|---|---|
| Consumer Name | Partition I | Delay | | | | | | | |
| tars.deployment.consumer | 0 | 0 | 0 | 0 | | 4547 | 4547 | "ti ... 639 | 10.2.22.44 |

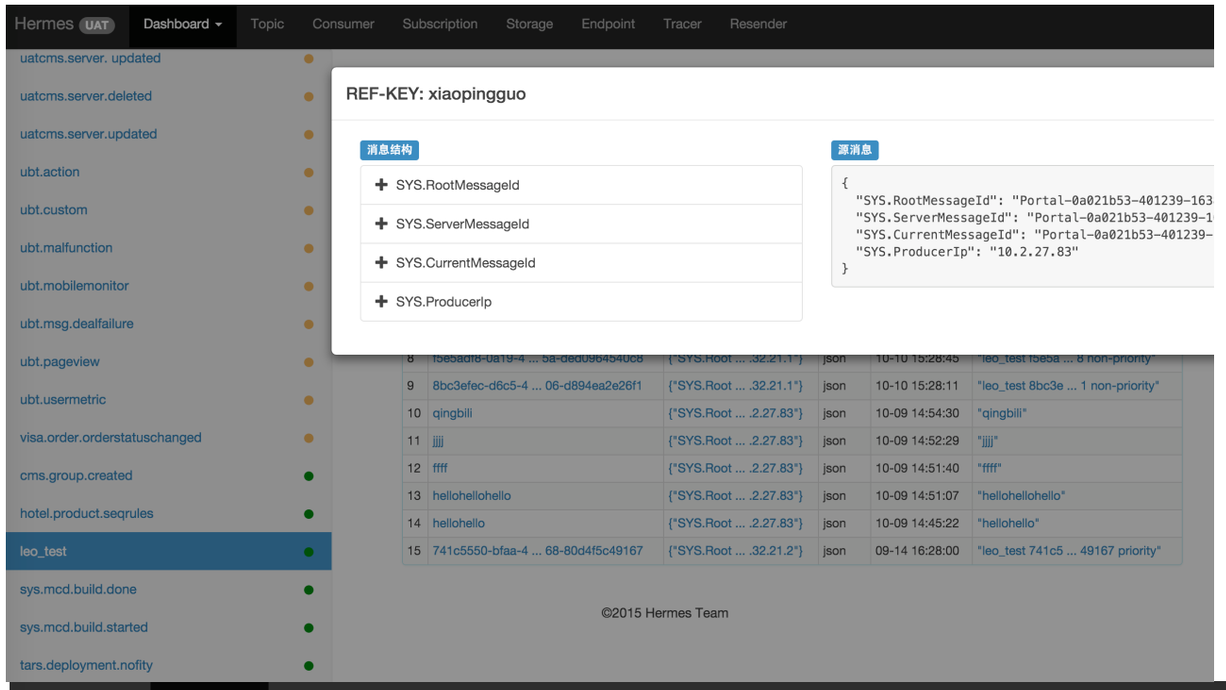
消费延迟排行

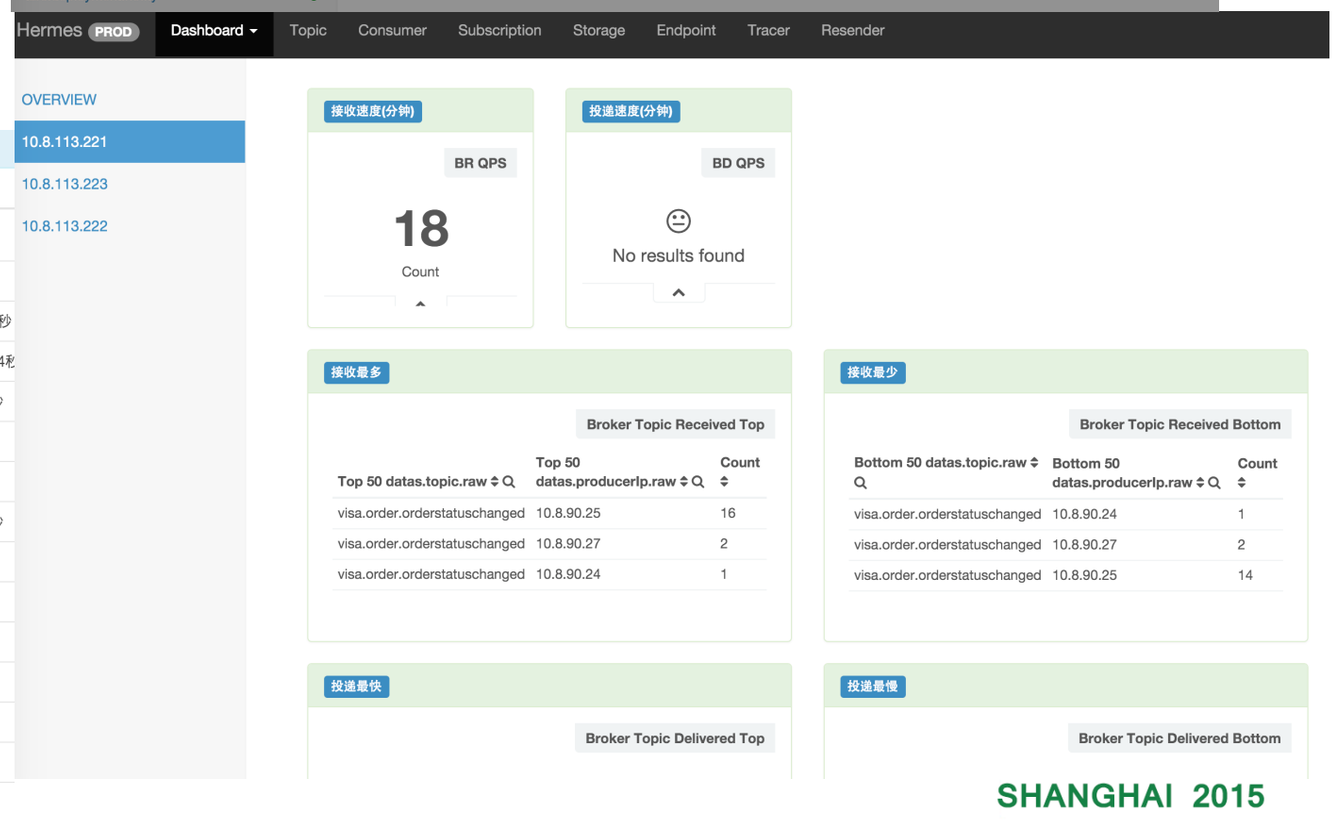
| Topic | Consumer | Delay(条) |
|---|---|---|
| Topic | Consumer | |
| hotel.produc ... mstatuschange | hotel.produc ... tatusrelation | 0 |
| hotel.produc ... mstatuschange | hotel.data.c ... mstatuschange | 635598204 |
| hotel.produc ... mstatuschange | hermes.verify | 0 |
| basebiz.cti. ... perations.add | basebiz.cti.urmp | 2051710 |
| visa.order.o ... statuschanged | cruise.order.visastatus | 125830 |
| cms.app.created | cms.test.consumer | 312 |
| cms.app.updated | cms.test.consumer | 242 |
| uatcms.group.updated | cms.test.consumer | 140 |
| uatcms.group.created | cms.test.consumer | 138 |
| uatcms.group.deleted | cms.test.consumer | 33 |
| leo_test | leo_test_group | 0 |
| sys.mcd.build.done | sys.mcd.buil ... ssor.notifier | 0 |
| cms.app.deleted | cms.test.consumer | 0 |

hotel.product.roomstatuschange

| Partition | Delay |
|---|---|
| 0 | 63688334 |
| 1 | 61137291 |
| 2 | 59973838 |
| 3 | 62606814 |
| 4 | 64284008 |
| 5 | 60976055 |
| 6 | 65324093 |
| 7 | 64917895 |

| 最近生产 | 延时 |
|---|---|
| 2015-09-17 09:52:34 | 23天6小时25秒 |
| 2015-09-18 16:43:30 | 21天23小时9分钟29秒 |
| 2015-09-24 18:18:05 | 15天21小时34分钟54秒 |
| 2015-09-25 10:07:51 | 15天5小时45分钟8秒 |
| 2015-10-08 09:46:19 | 2天6小时6分钟40秒 |
| 2015-10-08 15:16:08 | 2天36分钟51秒 |
| 2015-10-09 14:06:31 | 1天1小时46分钟28秒 |
| 2015-10-09 16:47:52 | 23小时5分钟7秒 |
| 2015-10-09 17:37:11 | 22小时15分钟48秒 |
| 2015-10-09 17:57:56 | 21小时55分钟3秒 |
| 2015-10-09 17:59:56 | 21小时53分钟3秒 |
| 2015-10-10 09:32:52 | 6小时20分钟7秒 |
| 2015-10-10 15:35:08 | 17分钟51秒 |



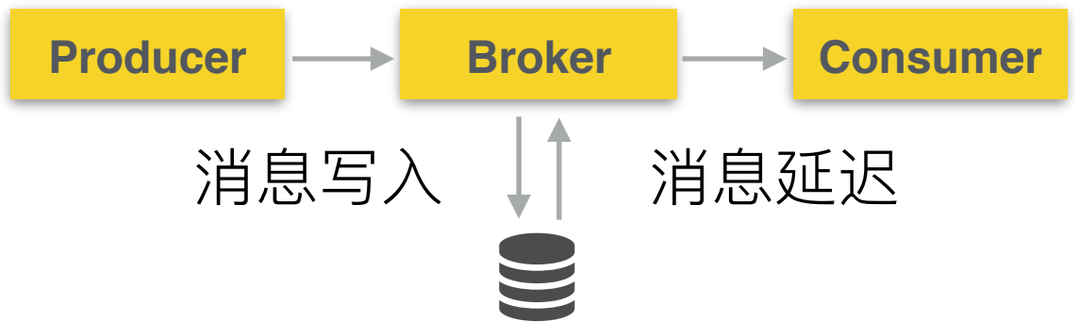



# 如何构建高效的MQ

# 一种有效的设计思路

- 单机如何优化
- 如何扩展到集群
- 如何管理集群

# 单机优化



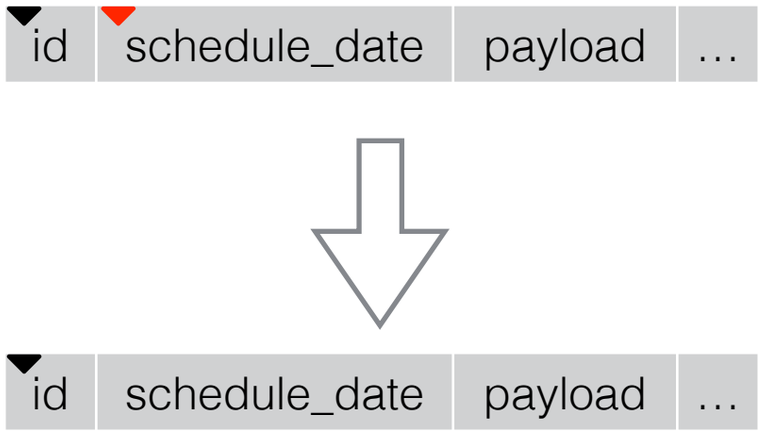
# 消息表设计

| id | payload | … |
|---|---|---|

- Partitioned Table(by id)
  - 高效的数据清理
- Insert only
- 仅id索引

# 重发表设计

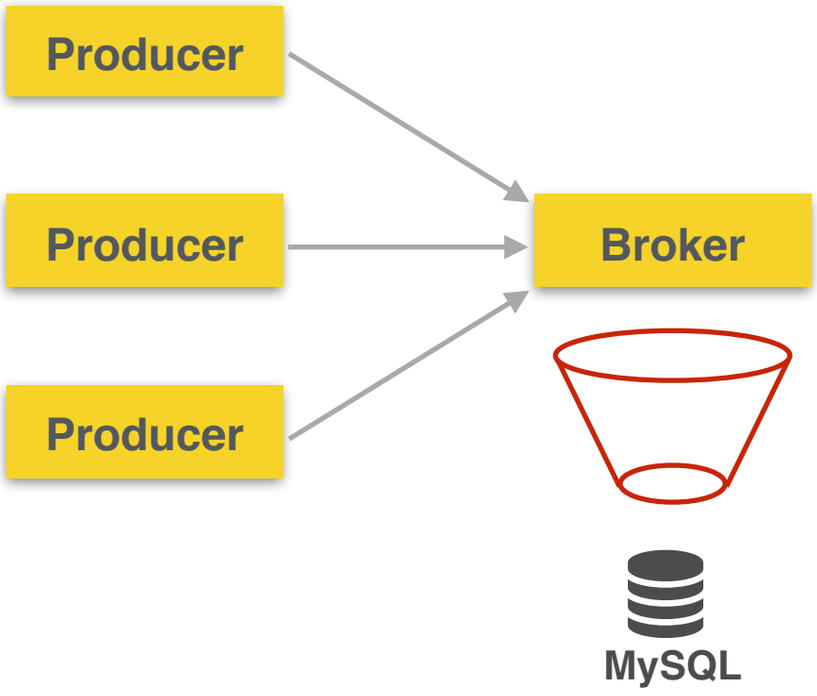
- 消费者指定重发时间
- 消息的重发时间非递增




- schedule_date和id需要联合索引

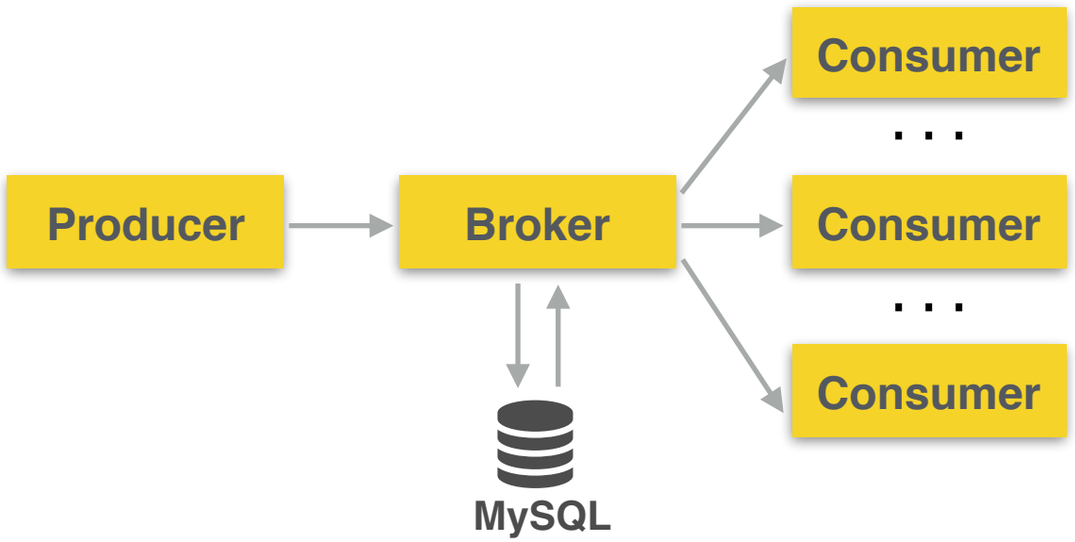
- 重发时间设置固定的延迟
- 仅id索引

# 批量写入



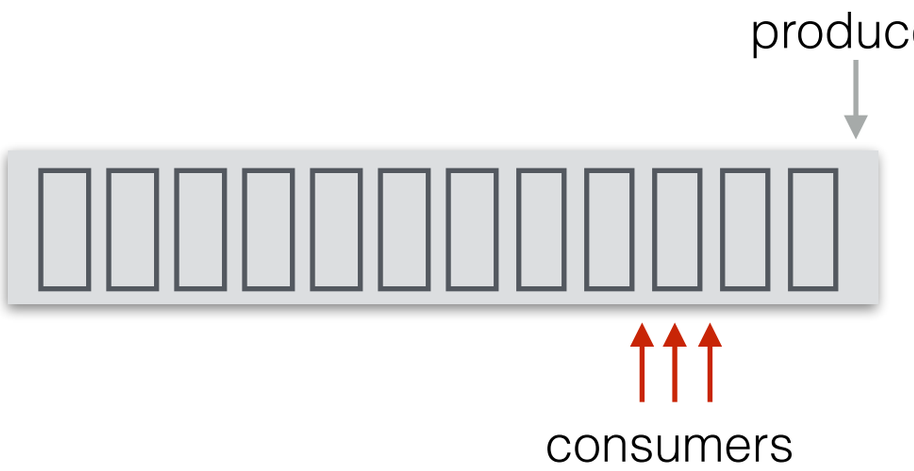

- 5x效率提升
- rewriteBatchedStatements=true

# 减少数据库轮询



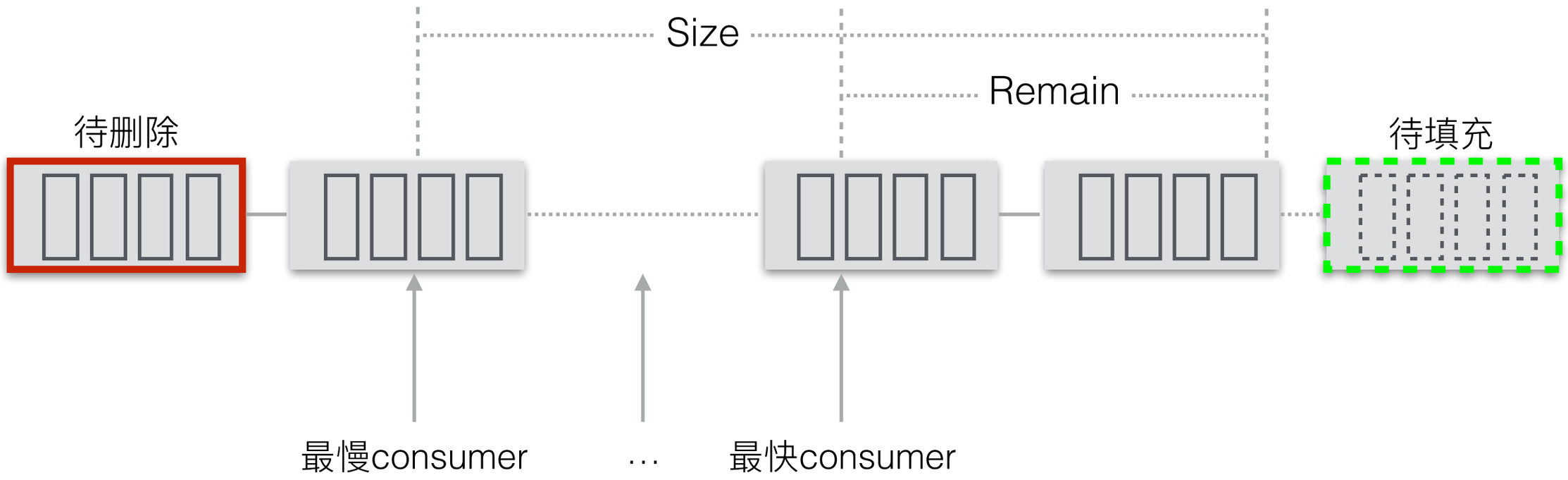

- Broker轮询DB是否有新消息
  - 延迟 vs 开销
  - 很容易导致DB高负载
- 捕获消息写入事件
  - 消息是否会写入其它Broker?

# 消息Buffer



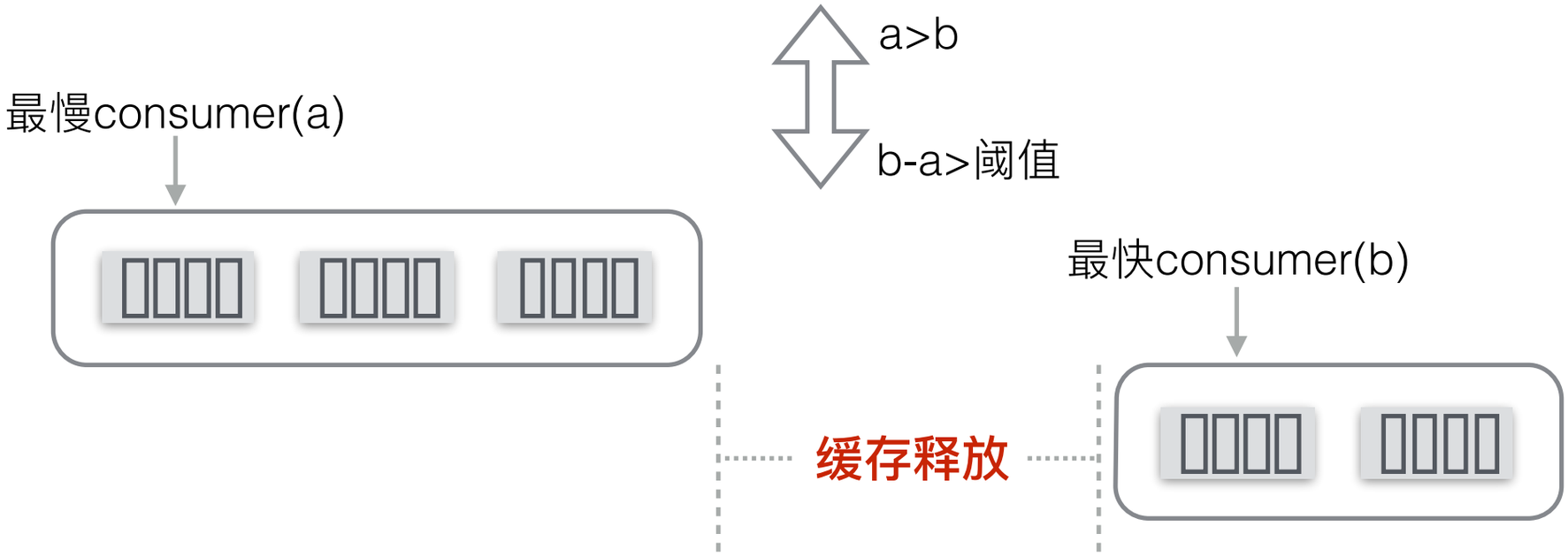

- 消费的消息邻近
  - 缓存命中率高
- 降低DB开销
- 降低消息延迟

# 消息Buffer

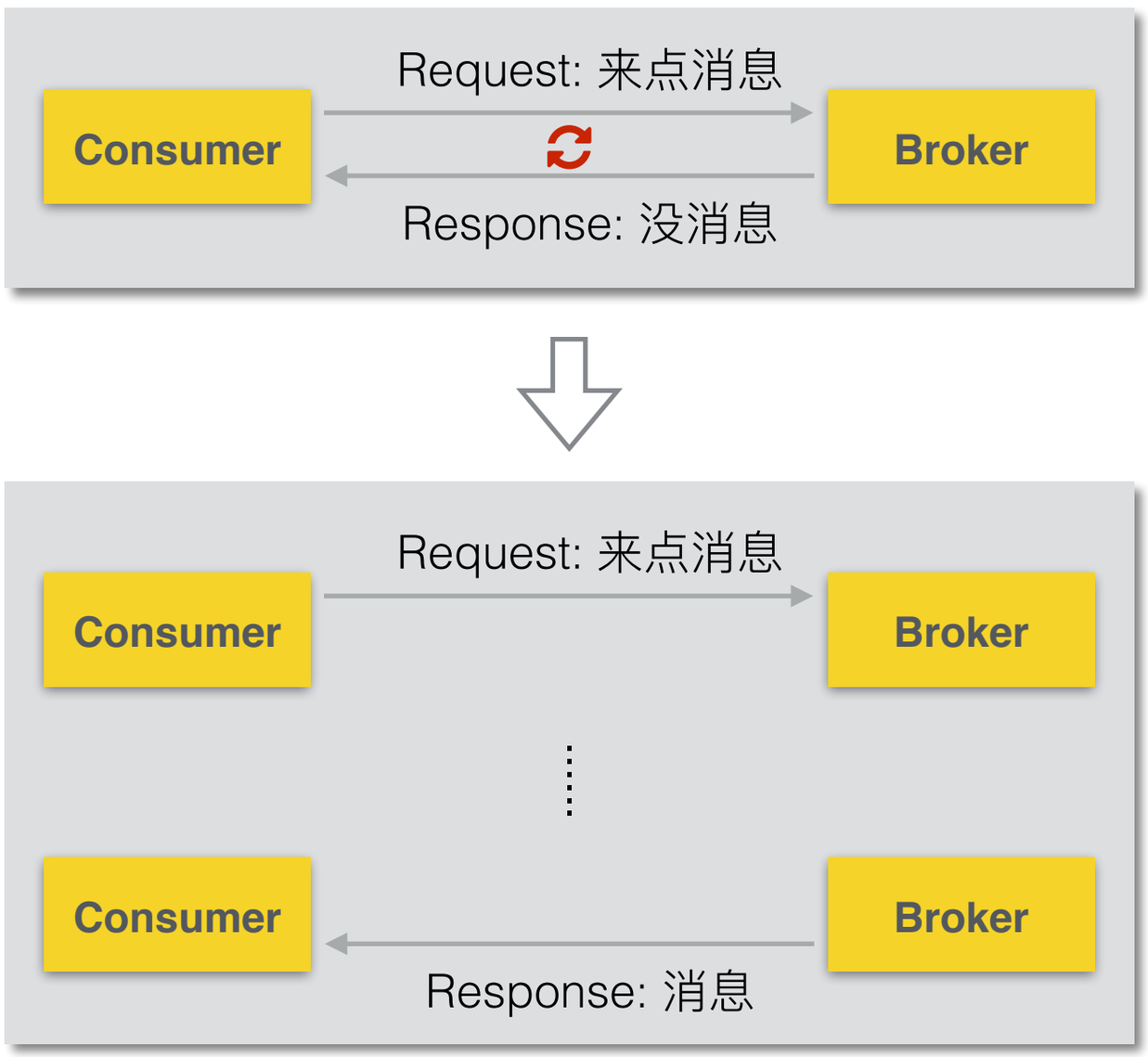



- 定时扫描并预加载消息
- 当Remain < 阈值时加载
- 当Size > 阈值时分裂Buffer

# Buffer的分裂合并

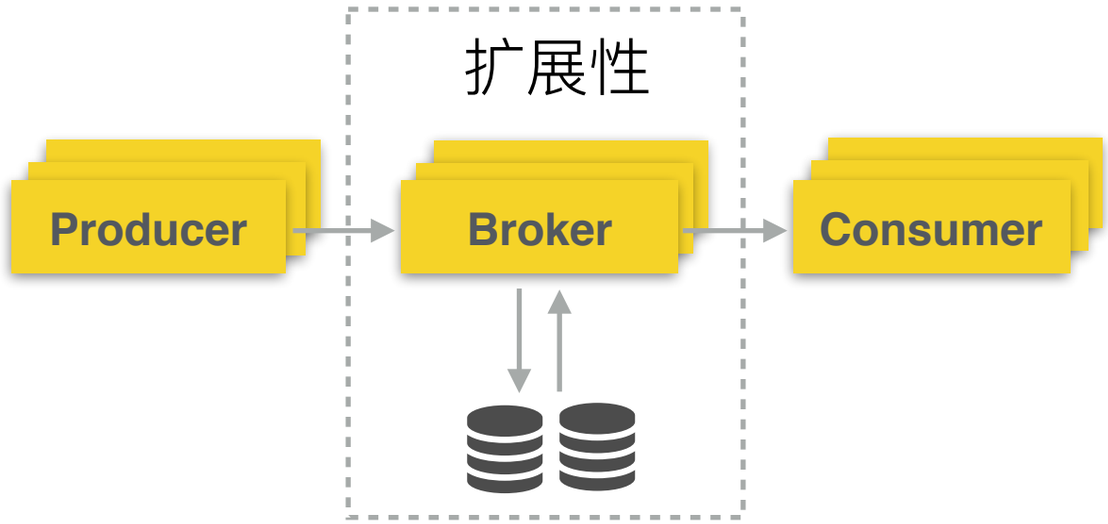



# 消费者Long Polling



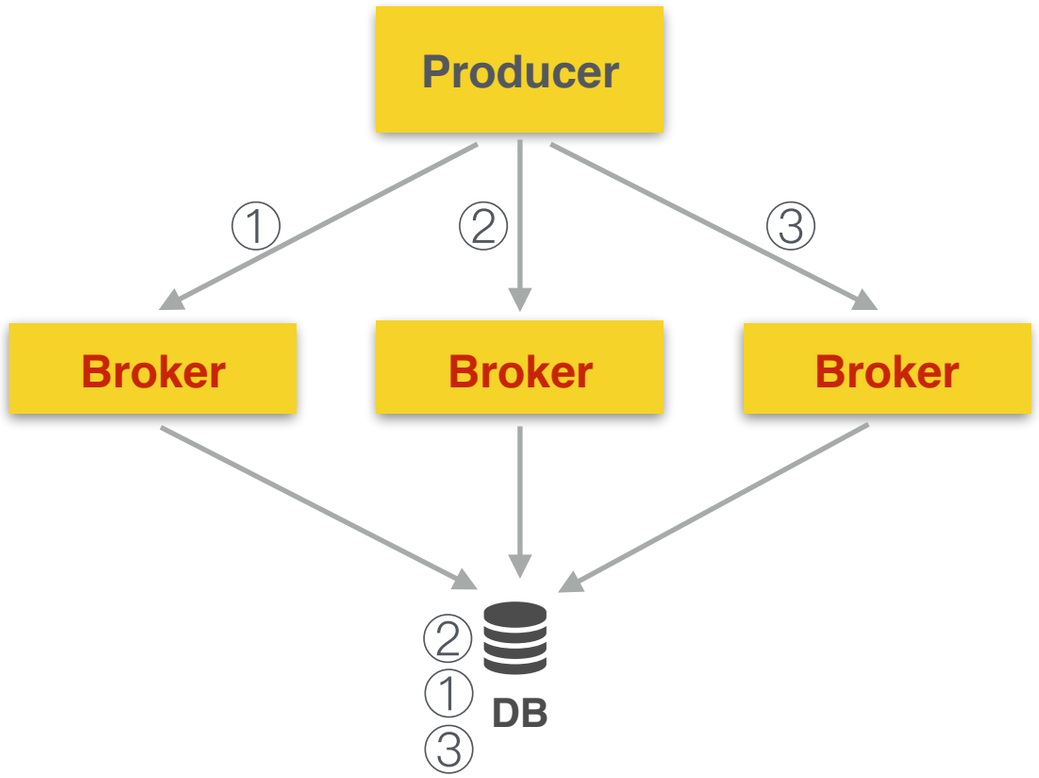

- Push vs Pull
- 消息低延迟要求快速轮询
- LongPolling
  - 降低消息延迟
  - 降低Broker负载

# 单机到集群



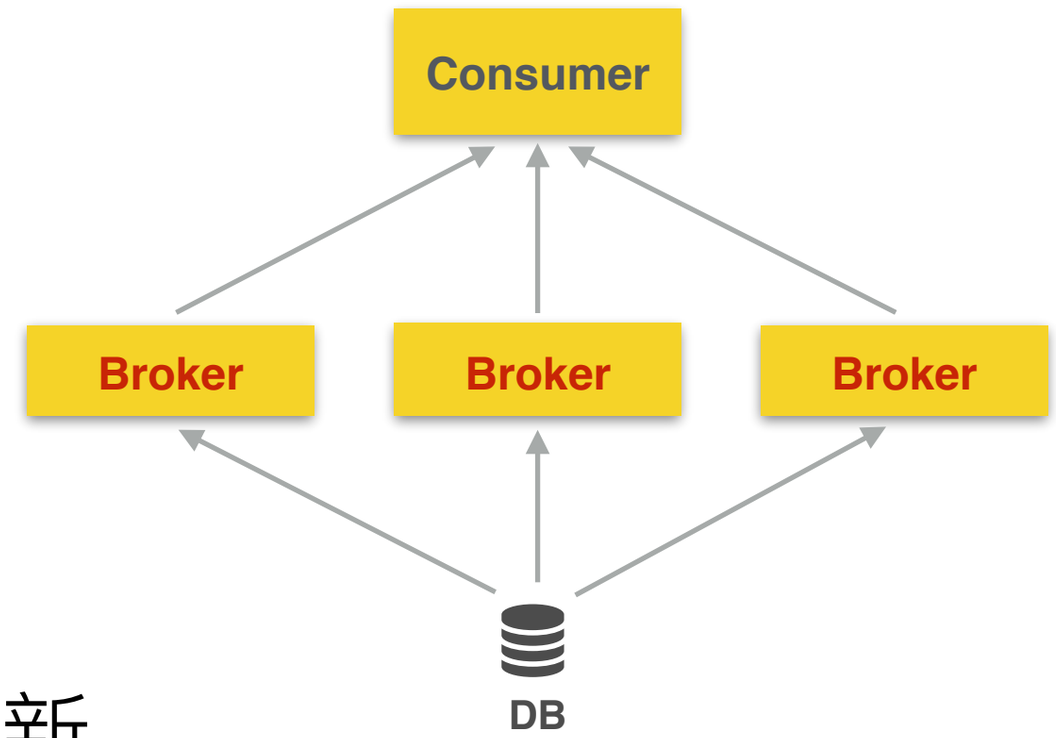
# Broker直接扩展

- 消息顺序无法保证



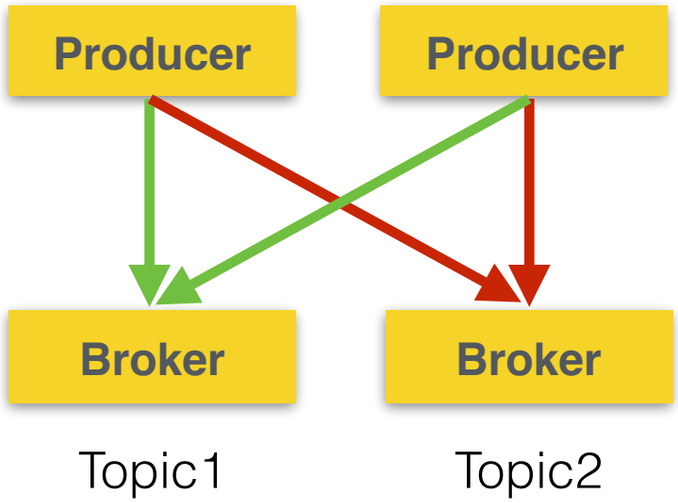
# Broker直接扩展

- 单机优化不再有效
  - 消息轮询
  - 消息缓存
- Consumer Offset无法高效更新



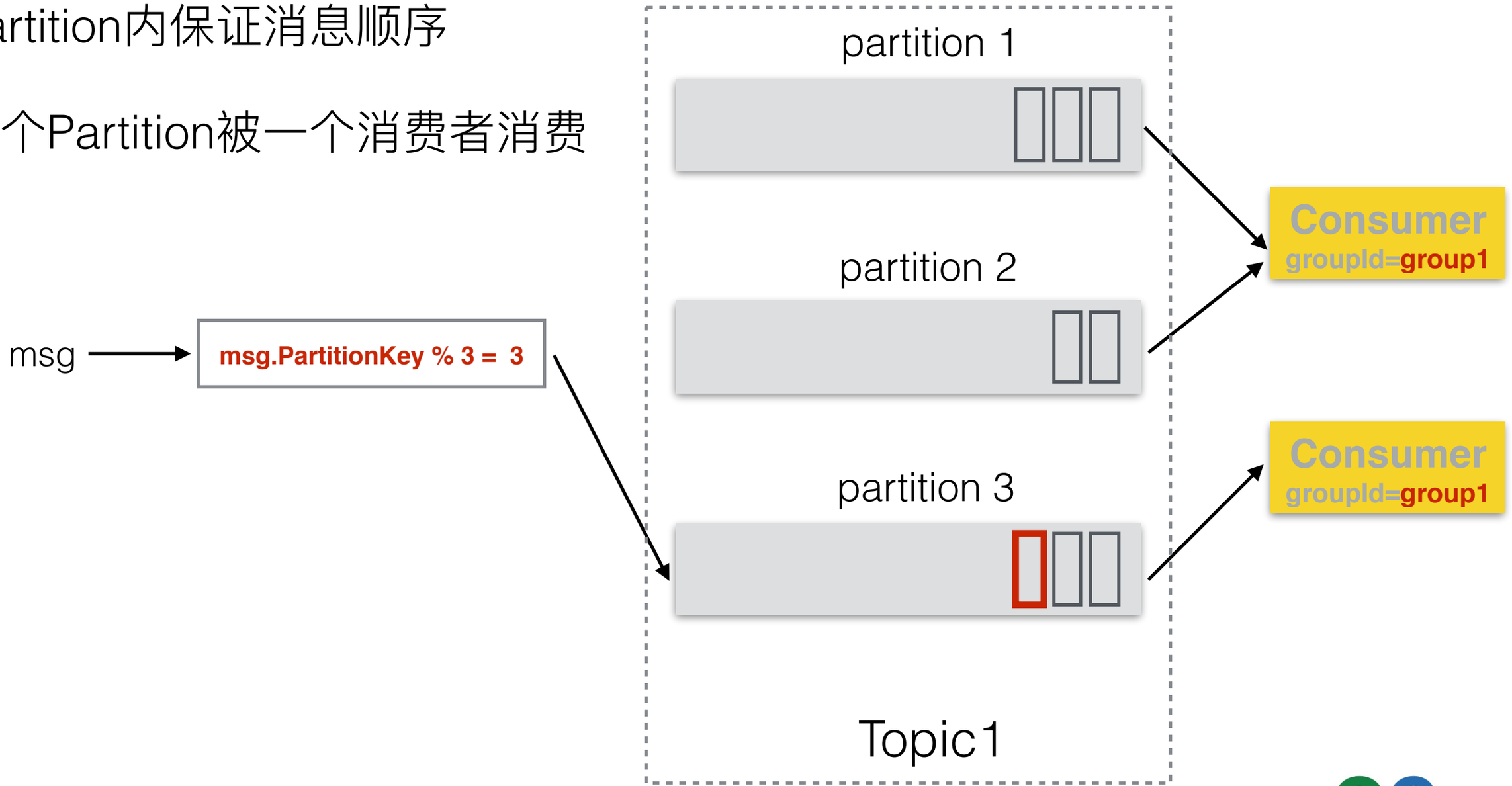
# Topic粘滞到Broker

- 消息顺序及单机优化继续有效
- Topic吞吐 < 单个Broker吞吐



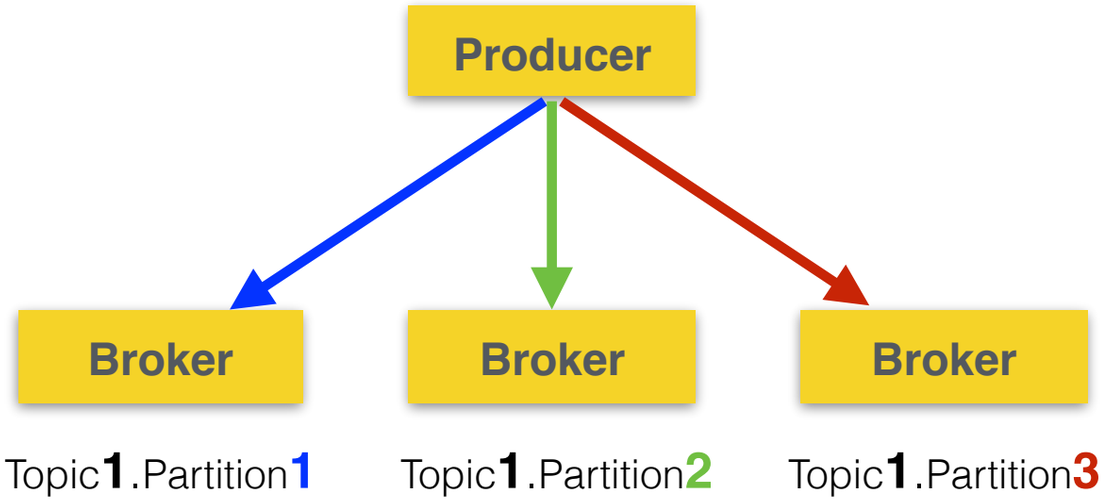
# Topic Partition

- Partition内保证消息顺序
- 一个Partition被一个消费者消费

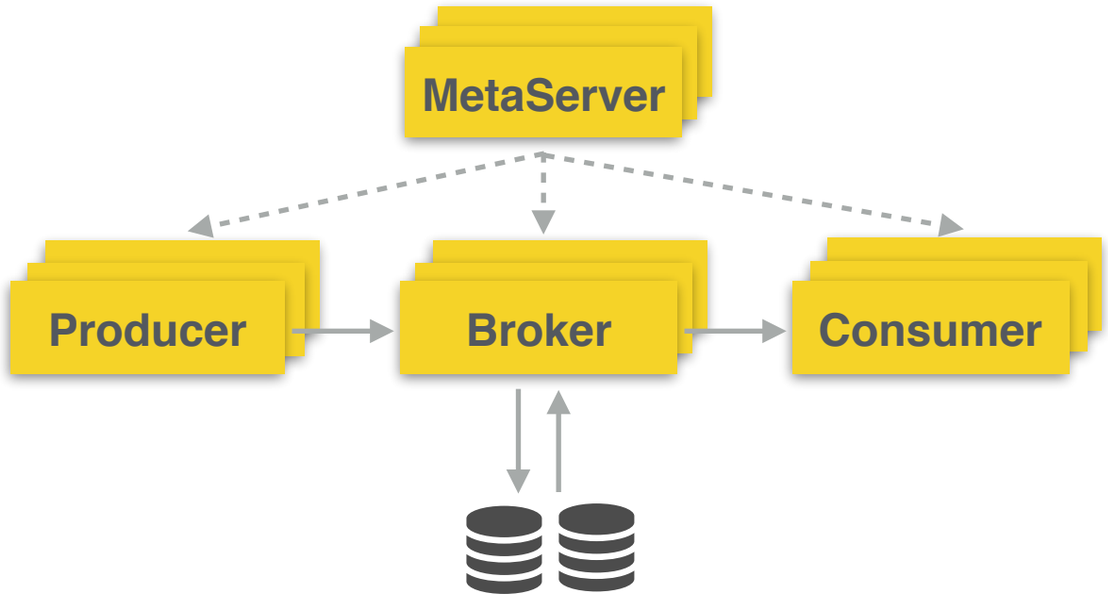


# Topic Partition

- Topic.Partition粘滞到Broker
- 单机优化有效
- 粒度更细，更易于做负载均衡
- 如何分配Partition到Broker?



# 集群管理



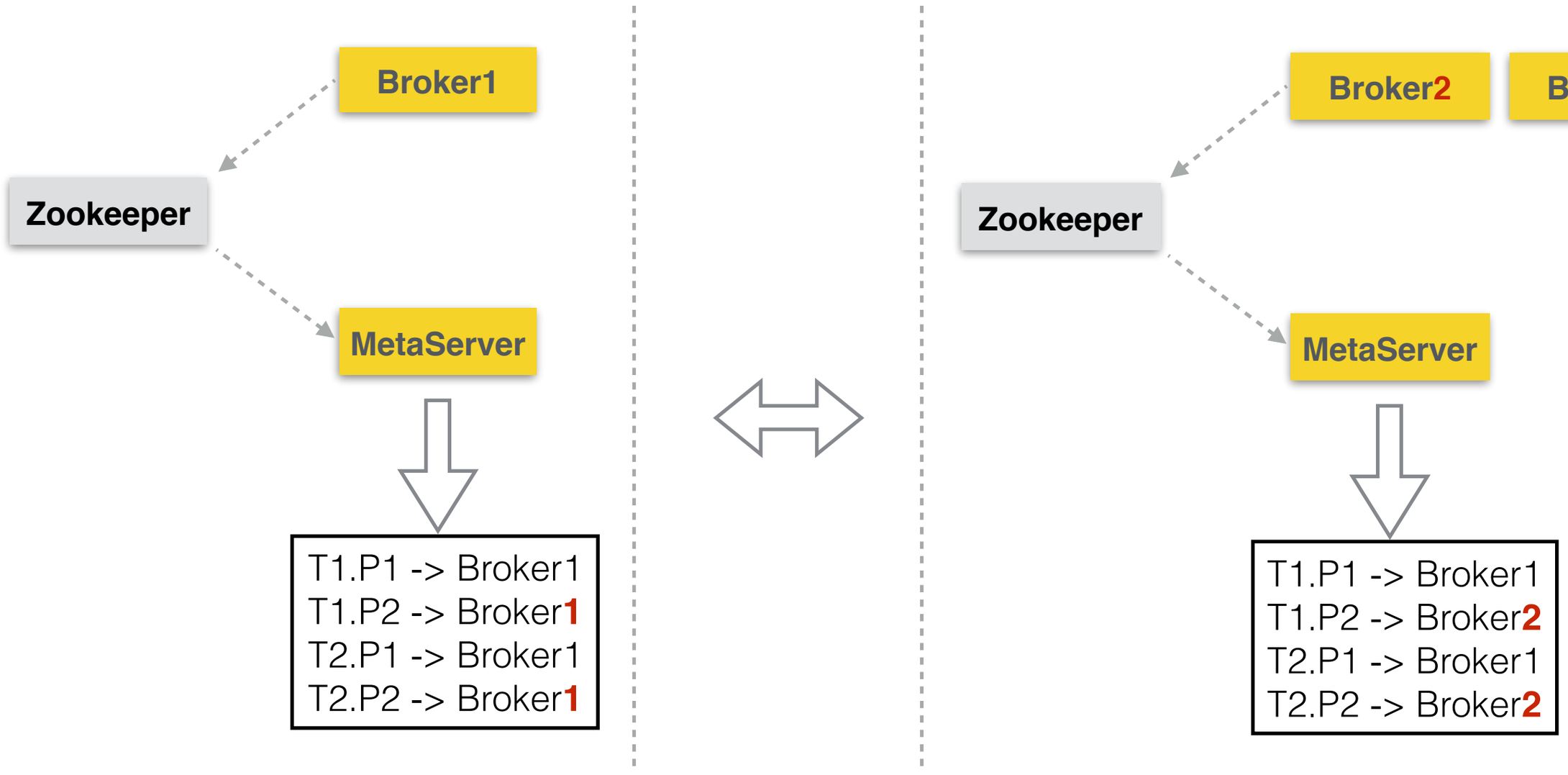
# 集群管理的基本问题

- Broker的加入/退出
- Consumer的加入/退出
- Topic.Partition的动态分配

# 基于Lease的集群管理

- 有时间限制的Lock
  - 不续租则到期释放
- 根据Lease生成消息的”路由表”
  - Producer->Broker->Consumer

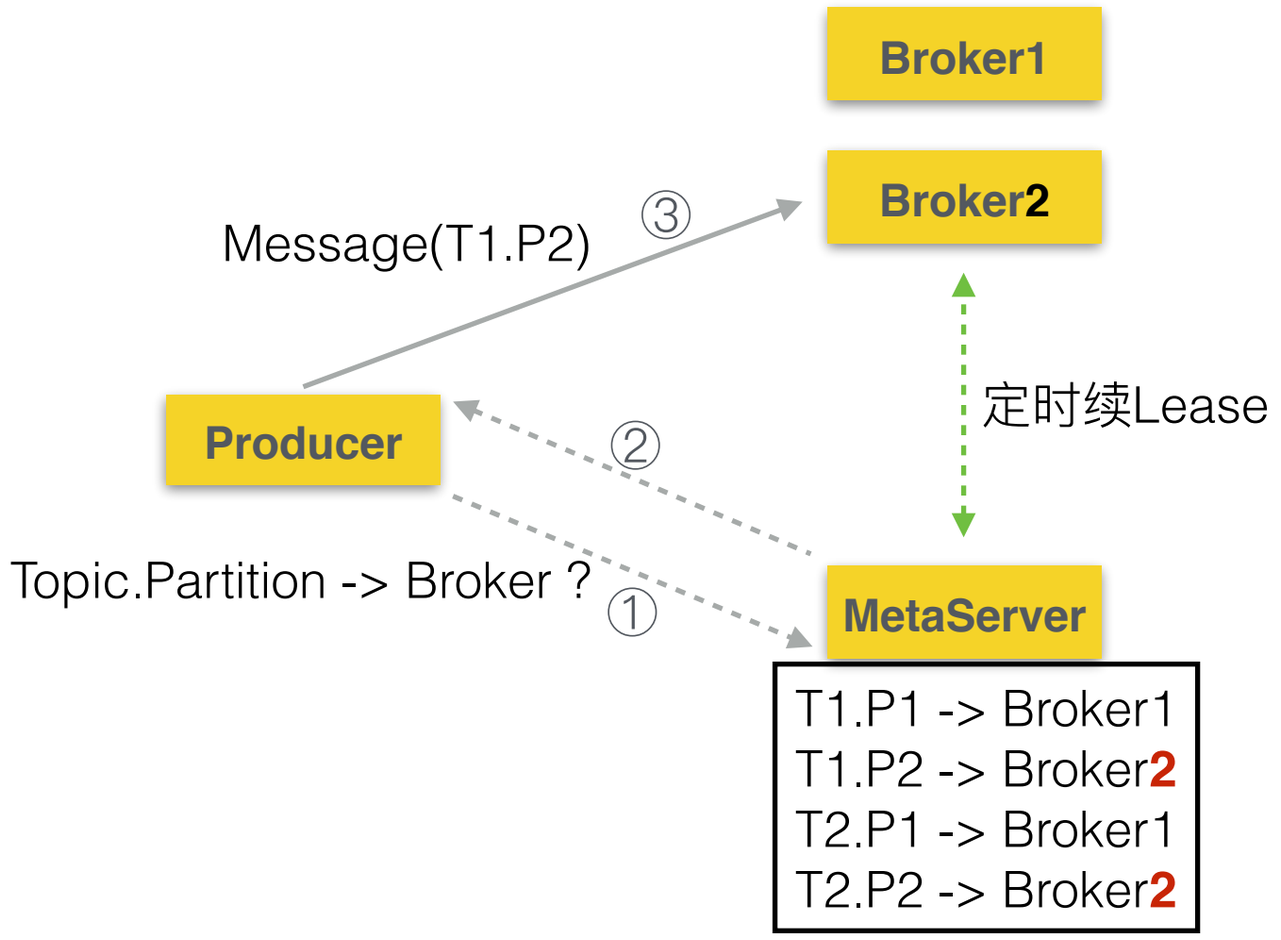
# Broker加入/退出



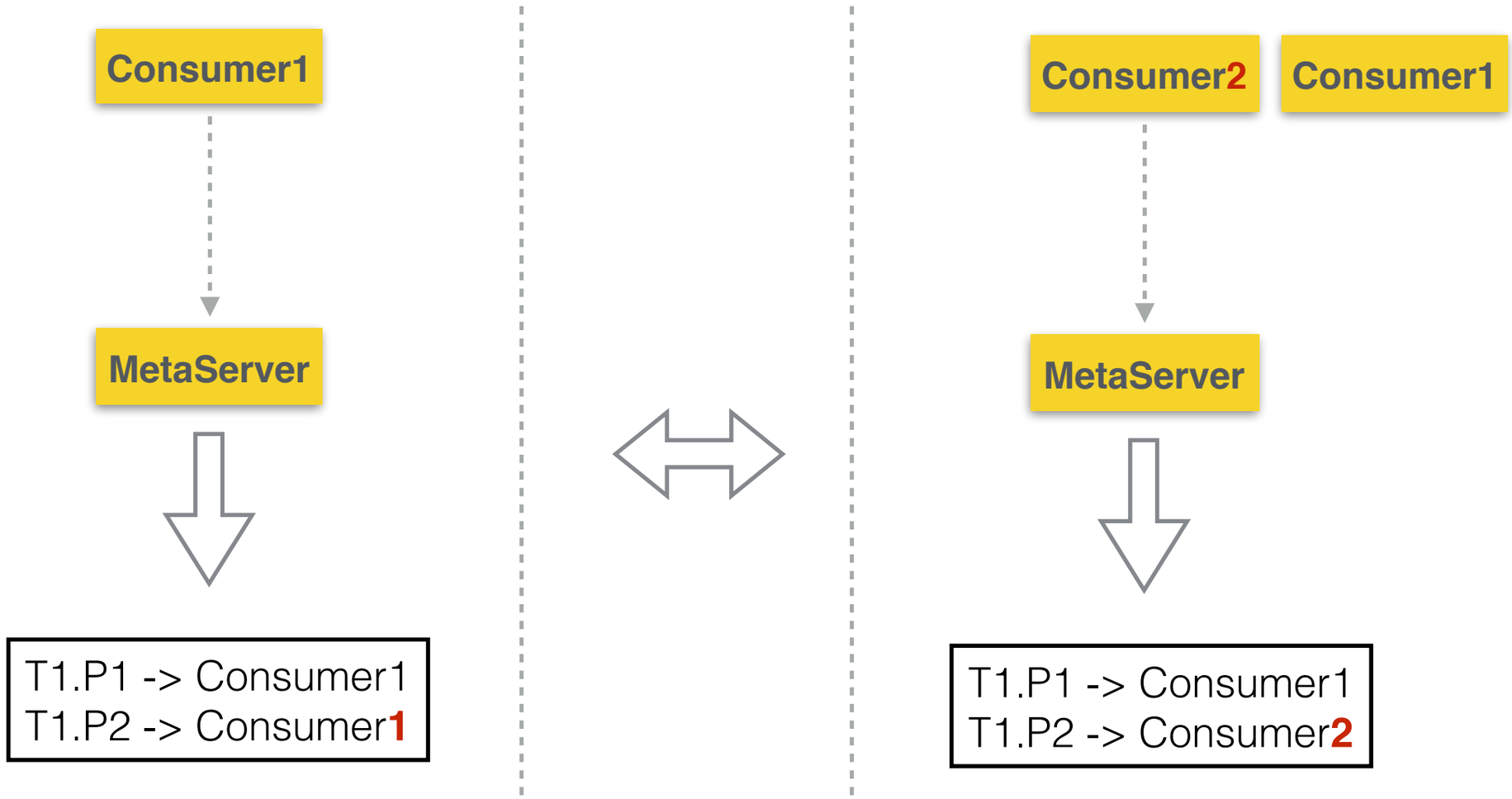

- MetaServer通过ZK发现Broker
- 重新分配Topic.Partition到Broker
- 发生变更的Lease不再允许续租

# 消息发送

- 定时刷新”路由”
- 发送到指定Broker
- 被拒绝则刷新”路由”



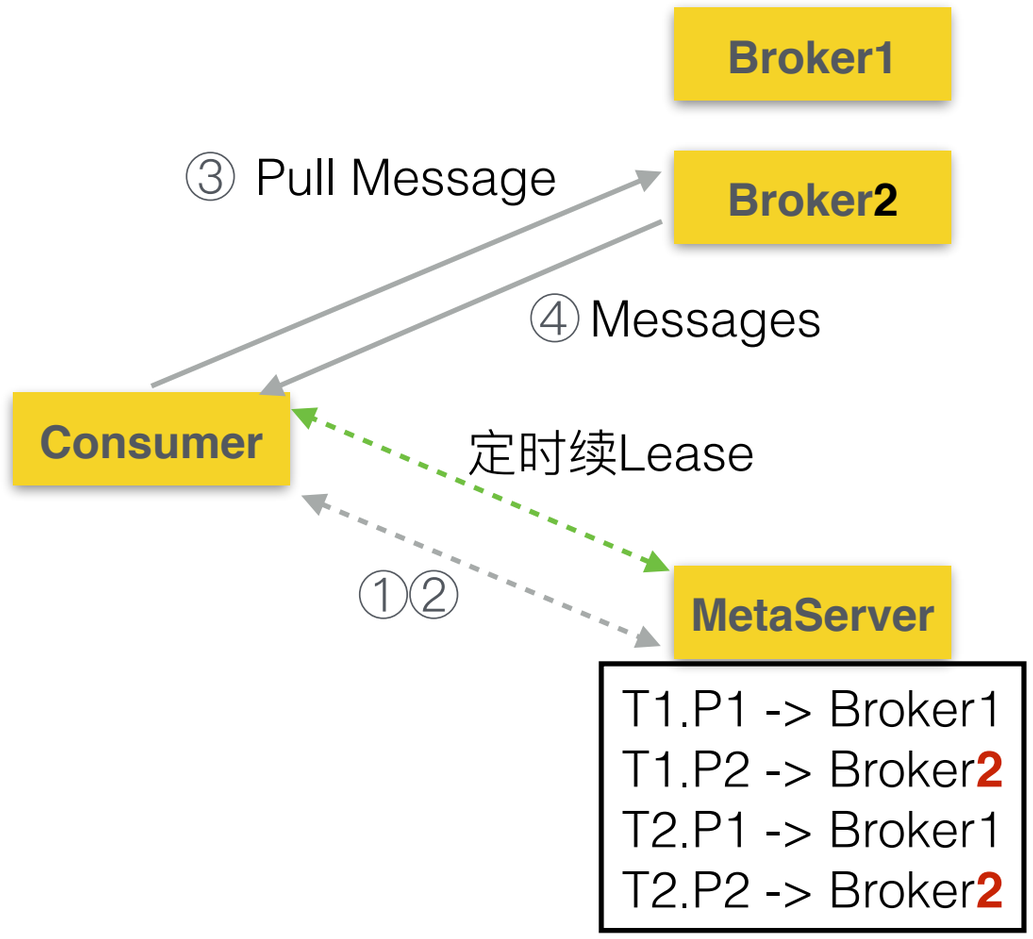
# Consumer加入/退出




- MetaServer通过Lease请求发现Consumer
- 重新分配Topic.Partition到Consumer
- 发生变更的Lease不再允许续租

# 消息接收

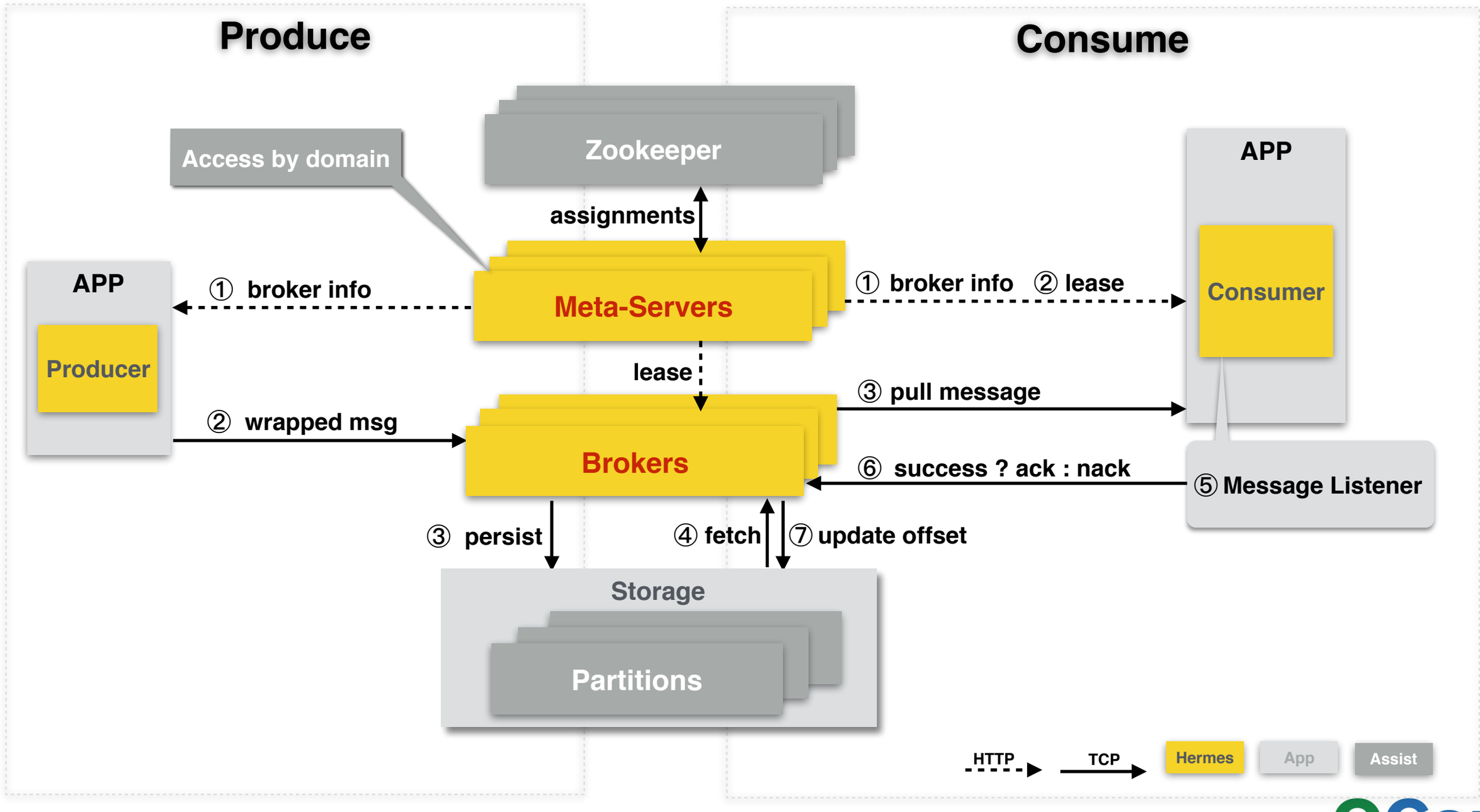
- 定时刷新"路由"
- 从指定Broker"拖"消息
- 续不了Lease则停止消费



# 集群管理总结

- Consumer不连接ZK
- 通过MetaServer竞争Lease
- MetaServer对集群有灵活的控制能力

# 消息收发全过程


Produce
Consume
Access by domain
Zookeeper
assignments
APP
Producer
① broker info
Meta-Servers
① broker info
② lease
APP
Consumer
lease
② wrapped msg
Brokers
③ pull message
⑥ success ? ack : nack
⑤ Message Listener
③ persist
④ fetch
⑦ update offset
Storage
Partitions
HTTP
TCP
Hermes
App
Assist

# 总结

- 消息写入
  - 批量、InsertOnly、索引
- 消息投递
  - Partition Stick、写入事件截获、预取、Long Polling
- 集群管理
  - Lease

# Thanks!