【技术干货】数据蜂巢架构演进之路

原创：贺思远 京东技术 2018-05-31

来这里找志同道合的小伙伴！

背景

各业务系统为使用mysql的业务数据，重复开发出多套数据同步工具，一方面难以管理，另外部分工具性能也偏差。需要一个统一为mysql数据提供同步服务的平台。该平台需支持离线同步，实时订阅，实时同步三大基本功能。

架构

一、功能整合

1、各功能如何实现?

离线同步：可理解为将根据一个sql查询出的数据同步到其它目标存储上；

实时订阅：通过实时解析mysql-binlog,将数据的变动封装成事件存于消息队列，供用户订阅消费；

实时同步：提供一些常见的订阅客户端料现，实时消费消息，将数据的变动应用于目标存储上。

2、如何将三个功能集成在一个平台架构下?

将离线同步，实时订阅，实时同步三个需求抽象为三种作业，分别为BatchJob,StreamJob,PieJob。

i. BatchJob参考Sqoop的模式，将需同步的数据先根据指定的规则进行分片，然后将作业根据分片拆分成多个任务，每个任务只同步本分片的数据，多个任务可同时运行，以加快同步效率；

ii. 以BatchJob的模式为基础，StreamJob也可根据需要采集的mysql实例分成多个任务，每个任务负责采集解析一个mysql的binlog，并将解析后的事件封装成消息存于本地供订阅者消费；

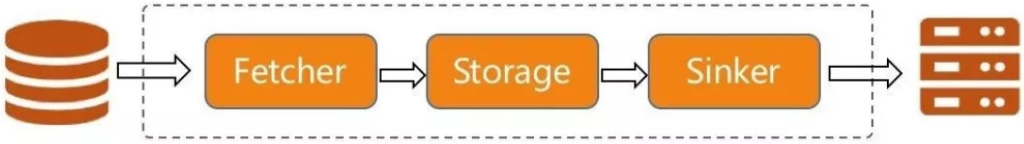
iii. PieJob是对订阅客户端的封装，每一个订阅客户端即可看作一个任务。

三种不同的作业最终都可以通过分片分成多个任务去运行，使用统一的模型。

二、任务细节

以下为各个Job进行分片后生成的Task内部具体实现细节

1、BatchTask



Fetcher负责抽取数据，Sinker负责写入数据，Storage为缓存层。

2、StreamTask

RelayLogTask负责拉取binlog；HHLTask负责解析Binlog,并将解析出的数据变更事件封装为易使用的消息体，最后存入hhl中。

hhl的实现借鉴了Kafka，可看作一个简易版的消息队列。消息使用protobuf序列化，压缩后顺序写入文件。同时提供了指定大小的索引块。

在StreamTask中提供了一个ClientServer负责处理客户端的订阅请求，细节如下图：

当收到指定位点的订阅请求后先通过索引快速定位对应的数据块，然后扫描数据块定位对应的消息，将该位点之后的所有消息通过指定的过滤器过滤，最终推送给客户端。

消息订阅的服务端并不维护客户订阅的状态，即不存储客户端的位点，交由客户端自行处理。服务端只负责将指定的位点之后的消息不断的推送给用户。

3、PieTask

PieTask实际是对客户端的封装，这里主要介绍一下客户端的实现。

客户端采用并发处理的模式，connector负责接收消息，paritioner负责分发消息交给不同的Processor(线程)处理。

因客户端需自己记录当前处理的位点，但又要保障在并发场景下记录的位点之前的消息都已被正确处理。为了减少线程间阻塞，使用了环形数组的提交方式（记录位点）。

三、集群

使用Master-Slave结构，如下图所示：

Master这里称之为Queen；Slave这里称之为Bee。

Queen负责作业的分片，调度；Bee负责任务的具体执行（任务由作业分片后得到）。

1、高可用

i.Mysql：mysql的高可用由dba维护，但mysql主从切换后对应的位点会不同，此处通过监测serverId的变更来发现主从切换，主机切换后通过时间在新实例上查找对应位点；

ii.Queen：通过zookeeper来实现Active与StandBy的切换

iii.Bee：宕机后Queen会将该主机上运行的所有任务切换到其它机器上

2、数据本地性

每一台Bee都有自己的机柜，机架，机房，分组信息。作业运行时可以指定自己的喜好，任务会优先分到指定的机器分组上

3、负载均衡

Bee在运行时会通过心跳汇报自己负载情况，当一个任务需要调度时，Queen会在满 足数据本地性的前提下优先将任务分发到负载低的主机上。

演进

一、HHL文件丢失

Binlog采集解析后的消息存于本地hhl文件中，一但主机发生HA切换后，之前的消息会全部丢失。

方案一：复本，缺点：占用大量磁盘资源，实现逻辑复杂，放弃使用；

方案二：数据补全，因本身mysql为满足运维需要,binlog会存储N天，丢失消息完全可以重新抽取解析binlog获得，此时不再需要对消息做复本，丢失的消息如果被请求可以重新生成，原理如下图所示：

Main为StreamTask启动后的立刻启动的线程组，启动位置为虚线所示，catchup线程组为请求丢失消息的客户端进行数据补全。

二、元数据

Binlog中并不记录字段名等相关信息，导致生成的消息只有数据，没有结构。

方案一：通过查询数据库获得，缺点：在解析存在延迟情况下，表结构可能不正确，弃用；

方案二：快照，StreamJob在初次启动时会对mysql中所有的表做一份快照，此后在运行期间当解析到DDL操作时会将原快照取出生成一个新的复本，并在该复本上应用对应的DDL操作，最后生成一个新快照。保证任何时刻的binlog都可以找到其对应的元数据。

同时每个StreamTask会提供一个元数据服务，消息在传输时不存储字段等信息，客户端需要时直接请求元数据服务即可，以减少带宽占用。

三、资源隔离

第一版采用的是分布式线程池的模式，同一个Bee上跑的多个任务在一个进程内以多线程的形式存在。但因系统为满足各种需求，提供了自己定义业务逻辑，上传jar包的方式提交作业，部分用户作业结束后忘记释放相关资源，以及一些作业占用资源过多影响其它任务的执行。随后弃用线程池，使用进程池。用子进程保障不同的作业之间有一个资源隔离。架构图如下：

四、子集群

原架构高度依赖Queen，当Bee与Queen断开连接后任务会立刻停止（防止Queen重新调度，进行多次执行）。但在库房环境下，当库房内部网络正常但与Queen网络不通时，因为库房的Bee全为同一分组，同时库房的任务只能分到对应库房内，此时同步任务将无法运行。为适应该场景，使用了子集群方案，具有特定分组信息的Bee启动时会和同一分组的机器先自发组建子集群，并推选Master,随后由子集群的Master与Queen进行交互。此时，对于需要发送给子集群的作业，Queen下放调度权限给子集群，由子集群的master负责调度。而Queen只负责作业状态的监控及生命周期的全局控制。同时该版本去除了对Zookeeper的依赖。架构图如下：

篇幅有限，更多细节敬请期待……

----------END----------

下面的内容同样精彩

点击图片即可阅读

京东技术 | 关注技术的公众号

长按，识别二维码，加关注