21 | 类似“点击流”这样的海量数据应该如何存储?

李玥 · 后端存储实战课

你好，我是李玥。

对于大部分互联网公司来说，数据量最大的几类数据是：点击流数据、监控数据和日志数据。这里面“点击流”指的是在 App、小程序和 Web 页面上的埋点数据，这些埋点数据记录用户的行为，比如你打开了哪个页面，点击了哪个按钮，在哪个商品上停留了多久等等这些。

当然你不用太担心自己的隐私问题，记录的这些行为数据不是为了监控用户，主要目的是为了从统计上分析群体用户的行为，从而改进产品和运营。比如，某件商品看的人很多，停留时间很长，最后下单购买的人却很少，那采销人员就要考虑是不是这件商品的定价太高了。

除了点击流数据以外，监控和日志数据都是大家常用的，我就不再多解释了。

这类数据都是真正“海量”的数据，相比于订单、商品这类业务的数据，数据量要多出 2～3 个数量级。每天产生的数据量就可能会超过 TB（1 TB = 1024 GB）级别，经过一段时间累积下来，有些数据会达到 PB（1 PB = 1024 TB）级别。

这种量级的数据，在大数据技术出现之前，是没法保存和处理的，只能是通过抽样的方法来凑合着做分析。Hadoop 等大数据技术出现以后，才使得存储和计算海量数据成为可能。

今天这节课，我们来说说，应该选择什么样的存储系统，来保存像“点击流”这样的海量数据。

使用 Kafka 存储海量原始数据

早期对于这类海量原始数据，都倾向于**先计算再存储**。也就是，在接收原始数据的服务中，先进行一些数据过滤、聚合等初步的计算，将数据先收敛一下，再落存储。这样可以降低存储系统的写入压力，也能节省磁盘空间。

这几年，随着存储设备越来越便宜，并且，数据的价值被不断地重新挖掘，更多的大厂都倾向于**先存储再计算**，直接保存海量的原始数据，再对数据进行实时或者批量计算。这种方案，除了贵以外都是优点：

- 不需要二次分发就可以同时给多个流和批计算任务提供数据；
- 如果计算任务出错，可以随时回滚重新计算；
- 如果对数据有新的分析需求，上线后直接就可以用历史数据计算出结果，而不用去等新数据。

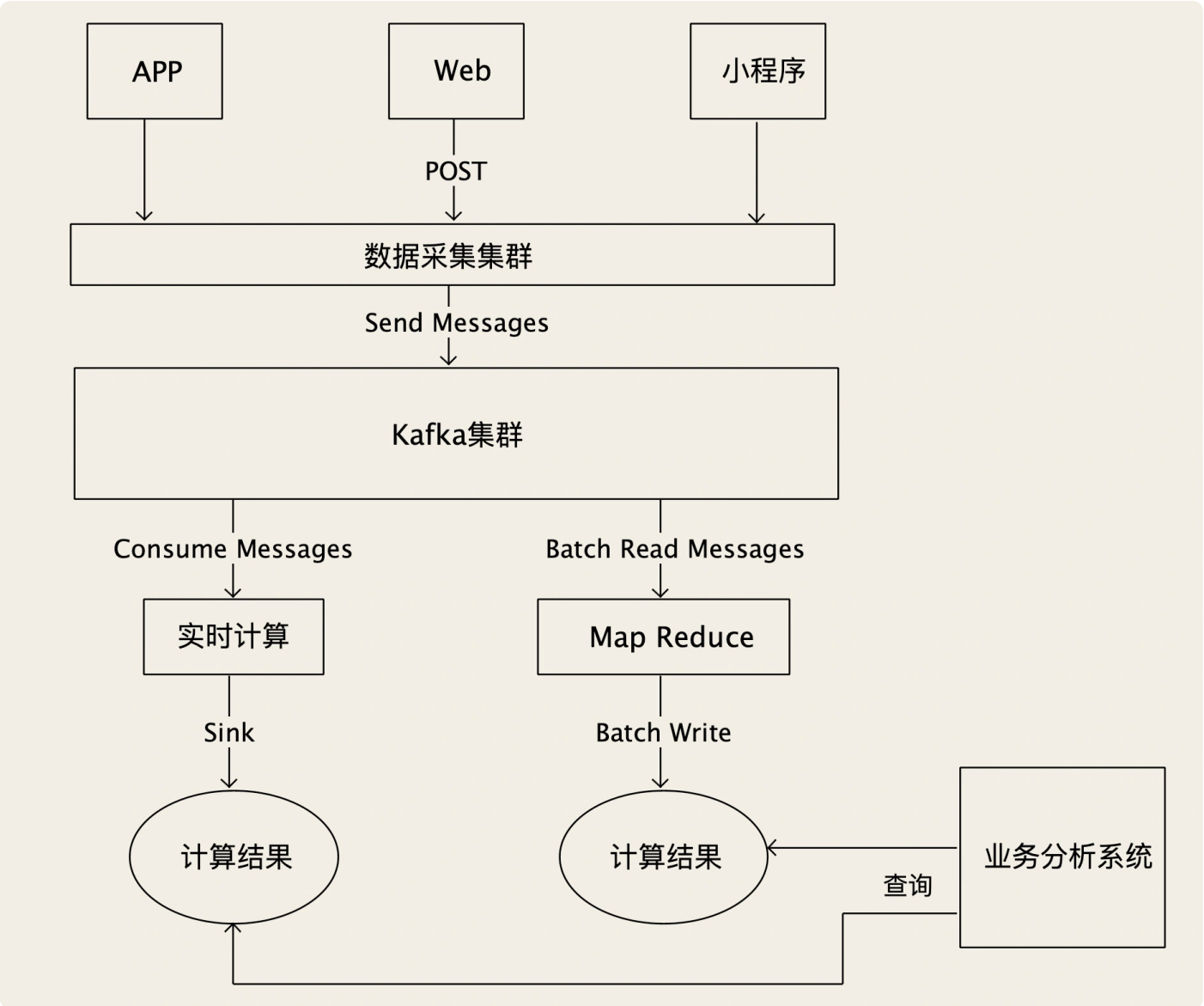
但是，这种方式对保存原始数据的存储系统要求就很高了：既要有足够大的容量，能水平扩容，还要读写都足够快，跟得上数据生产的写入速度，还要给下游计算提供低延迟的读服务。什么样的存储能满足这样的要求呢？这里我给出几种常用的解决方案。

第一种方案是，使用 Kafka 来存储。有的同学会问了，Kafka 不是一个消息队列么，怎么成了存储系统了？那我告诉你，**现代的消息队列，本质上就是分布式的流数据存储系统。**

如果你感兴趣的话，你可以仔细去研究一下 Kafka，它的数据是如何存储、分片、复制的？它是如何保证高可用，如何保证数据一致性的？那你会发现它和我们之前讲过的那些分布式存储系统，并没有什么太大的区别。唯一的区别就是，它的查询语言（生产和消费消息）和存储引擎的数据结构（Commit Log）比一般的存储系统要简单很多。但也正是因为这个原因，使得

Kafka 的读写性能远远好于其他的存储系统。Kafka 官方给自己的定位也是“分布式流数据平台”，不只是一个 MQ。

Kafka 提供“无限”的消息堆积能力，具有超高的吞吐量，可以满足我们保存原始数据的大部分要求。写入点击流数据的时候，每个原始数据采集服务作为一个生产者，把数据发给 Kafka 就可以了。下游的计算任务，可以作为消费者订阅消息，也可以按照时间或者位点来读取数据。并且，Kafka 作为事实标准，和大部分大数据生态圈的开源软件都有非常好的兼容性和集成度，像 Flink、Spark 等大多计算平台都提供了直接接入 Kafka 的组件。



当然，Kafka 也不是万能的，你可能注意到了，我刚刚讲 Kafka 提供“无限”的消息堆积能力，我在这个“无限”上打了个引号，这里面还是有一些限制需要注意的。Kafka 也支持把数据分片，这个在 Kafka 中叫 Partition，每个分片可以分布到不同的存储节点上。

写入数据的时候，可以均匀地写到这些分片上，理论上只要分片足够多，存储容量就可以是“无限”的。但是，单个分片总要落到某一个节点上，而单节点的存储容量毕竟是有限的，随着时间推移，单个分片总有写满的时候。

即使它支持扩容分片数量，也没办法像其他分布式存储系统那样，重新分配数据，把已有分片上的数据迁移一部分到新的分片上。所以扩容分片也解决不了已有分片写满的问题。而 Kafka 又不支持按照时间维度去分片，所以，**受制于单节点的存储容量，Kafka 实际能存储的数据容量并不是无限的**。

Kafka 之外还有哪些解决方案?

所以，需要长时间（几个月 – 几年）保存的海量数据，就不适合用 Kafka 存储。这种情况下，只能退而求其次，使用第二种方案了。

第二种方案是，使用 HDFS 来存储。使用 HDFS 存储数据也很简单，就是把原始数据写成一个一个文本文件，保存到 HDFS 中。我们需要按照时间和业务属性来组织目录结构和文件名，以便于下游计算程序来读取，比如说：**“click/20200808/Beijing_0001.csv”**，代表 2020 年 8 月 8 日，从北京地区用户收集到的点击流数据，这个是当天的第一个文件。

对于保存海量的原始数据这个特定的场景来说，HDFS 的吞吐量是远不如 Kafka 的。按照平均到每个节点上计算，Kafka 的吞吐能力很容易达到每秒钟大几百兆，而 HDFS 只能达到百兆左右。这就意味着，要达到相同的吞吐能力，使用 HDFS 就要比使用 Kafka，多用几倍的服务器数量。

但 HDFS 也有它的优势，第一个优势就是，它能提供真正无限的存储容量，如果存储空间不够了，水平扩容就可以解决。另外一个优势是，HDFS 能提供比 Kafka 更强的数据查询能力。Kafka 只能按照时间或者位点来提取数据，而 HDFS 配合 Hive 直接就可以支持用 SQL 对数据进行查询，虽然说查询的性能比较差，但查询能力要比 Kafka 强大太多了。

以上这两种方案因为都有各自的优势和不足，在实际生产中，都有不少的应用，你可以根据业务的情况来选择。那有没有兼顾这二者优势的方案呢？最好能做到，既有超高的吞吐能力，又能无限扩容，同时还能提供更好的查询能力，有这样的好事儿么？

我个人的判断是，目前还没有可用大规模于生产的，成熟的解决方案，但未来应该会有的。目前已经有一些的开源项目，都致力于解决这方面的问题，你可以关注一下。

一类是**分布式流数据存储**，比较活跃的项目有Pravega和 Pulsar 的存储引擎Apache BookKeeper。我所在的团队也在这个方向上持续探索中，也开源了我们的流数据存储项目JournalKeeper，也欢迎你关注和参与进来。这些分布式流数据存储系统，走的是类似 Kafka 这种流存储的路线，在高吞吐量的基础上，提供真正无限的扩容能力，更好的查询能力。

还有一类是**时序数据库（Time Series Databases）**，比较活跃的项目有InfluxDB和OpenTSDB等。这些时序数据库，不仅有非常好的读写性能，还提供很方便的查询和聚合数据的能力。但是，它们不是什么数据都可以存的，它们专注于类似监控数据这样，有时间特征并且数据内容都是数值的数据。如果你有存储海量监控数据的需求，可以关注一下这些项目。

小结

在互联网行业，点击流、监控和日志这几类数据，是海量数据中的海量数据。对于这类数据，一般的处理方式都是先存储再计算，计算结果保存到特定的数据库中，供业务系统查询。

所以，对于海量原始数据的存储系统，我们要求的是超高的写入和读取性能，和近乎无限的容量，对于数据的查询能力要求不高。生产上，可以选择 Kafka 或者是 HDFS，Kafka 的优点是读写性能更好，单节点能支持更高的吞吐量。而 HDFS 则能提供真正无限的存储容量，并且对查询更友好。

未来会有一些开源的流数据存储系统和时序数据库逐步成熟，并陆续应用到生产系统中去，你可以持续关注这些项目。

思考题

课后请你想一下，为什么 Kafka 能做到几倍于 HDFS 的吞吐能力，技术上的根本原因是什么？欢迎你在留言区与我讨论。

感谢你的阅读，如果你觉得今天的内容对你有帮助，也欢迎把它分享给你的朋友。

精选留言 (14)

李玥 置顶
2020-04-14

Hi，我是李玥。

这里回顾一下上节课的思考题：

我们整个切换的方案中，只有一个步骤是不可逆的，就是由双写切换为单写新库这一步。如果说不计成本，如何修改我们的迁移方案，让这一步也能做到快速回滚？

答案：

双写切新库单写这一步不可逆的主要原因是，一旦切为新库单写，旧库的数据就和新库不一致了，这时候就没法再切回旧库了。所以，问题的关键是，切到新库单写后，需要保证旧库的数据和新库保持同步。那我们的双写就要增加一种过渡状态：就是从双写以旧库为准，过渡到双写以新库为准。然后把比对和补偿程序反过来，用新库的数据补偿旧库的数据。这样就可以做到，一旦出问题，再切回到旧库上了。

共 3 条评论>

42

每天晒白牙
2020-04-14

Kafka 性能高的原因
1.采用批处理的方式提升吞吐量
2.利用了磁盘文件顺序读写性能高的特点在设计存储
3.利用了操作系统的 PageCache 做缓存，减少 IO
4.采用零拷贝技术加速消费流程
来自老师第一个专栏

作者回复: 哈哈，现学现卖。

共 2 条评论>

53

nfx
2020-04-19

虽然hdfs和kafka都可以用来做存储, 但 kafka在使用方面像磁带; hdfs更像硬盘

作者回复: 这个比喻非常形象哈。

共 2 条评论> 22

seg-上海
2020-04-27

老师，文中提到“Kafka 实际能存储的数据容量并不是无限的”，那如果不断的加新的broker，然后同时新增分片，是不是可以做到无限的扩展存储呢

作者回复: 理论上是可以的，但实际操作起来就比较麻烦。
你要知道哪些分区已经满了，哪些没满，指定去写那些还有空间的分区。额外增加了管理成本，对数据采集系统也不友好。

9

xzy
2020-11-07

pulsar 采用存算分离的结构，能够快速扩容计算节点和存储节点。且扩容后，新的计算节点和存储节点能够快速承担起计算和存储的责任，因此我觉得，pulsar 能够做到无限存储且查询能力和kafka 类似。

4

滴流乱转小胖儿
2020-04-14

老师你好，对于大数据存储，使用Elasticsearch是否可以?

共 4 条评论> 2

夜辉
2021-04-11

本质上因为Kafka采用复制状态机 (Replication State Machine)模型
同时底层对各种IO操作进行了优化

1

djfhchdh
2020-09-21

kafka的零拷贝，顺序读写

1

黄海峰
2020-04-14

求更多介绍分布式流数据存储和时序数据库。。。尤其是你们有在研发journalkeeper，求干货，怎么高吞吐无限扩容及强大查询。。。

1

ifelse
2022-12-14 来自浙江

对于海量原始数据的存储系统，我们要求的是超高的写入和读取性能，和近乎无限的容量，对于数据的查询能力要求不高。--记下来

Geek_a78004
2021-07-19

老师您好，在读少写多的场景下，cassandra是不是更有利呢?

凯文小猪
2021-04-20

kafka性能好有以下几点：
1. 本身是攒起来一批批发的 所以会有个batch.size 底层是bytebuffer打包做的
2. 本身操作系统就是write back 这部分和文件系统本身做法是一样的 当然对操作系统很友好
3. 做了一些index文件做了mmap log文件存的是文件内偏移量 这一部分网上资料一大堆

J.Smile
2021-04-13

“受制于单节点的存储容量，Kafka 实际能存储的数据容量并不是无限的。”

老师好，我理解单节点存储容量如果达到上限，可以增加broker机器，然后将容量达到上限的那个broker的一部分分区重分配到新增的broker上就可以把数据移走了，那么单节点上限应该就解决了。

独酌相思解千愁

2020-09-23

老师，问个题外话。怎么探测热点数据呢。比如数，目前数据库中有很多数据，某些时间可能在一段时间访问次数很多。怎么探测这个呢。类似于微博大V发了一条微博，或者某时间出现某个爆炸性新闻。