极客时间训练营

系统架构设计师

第19章 大数据架构设计理论与实践

授课：王建平

目录

目录

传统数据处理系统存在的问题

◆传统数据库的数据过载问题。传统应用的数据系统架构设计时，应用直接访问数据库系统。用户访问量增加时，数据库无法支撑日益增长的用户请求的负载，从而导致数据库服务器无法及时响应用户请求，出现超时的错误。关于这个问题的常用解决方法如下：
(1)增加异步处理队列，通过工作处理层批量处理异步处理队列中的数据修改请求。
(2)建立数据库水平分区，通常建立 Key 分区，以主键/唯一键 Hash 值作为 Key。
(3)建立数据库分片或重新分片，通常专门编写脚本来自动完成，且要进行充分测试。
(4)引入读写分离技术，主数据库处理写请求，通过复制机制分发至从数据库。
(5)引入分库分表技术，按照业务上下文边界拆分数据组织结构，拆分单数据库压力。

◆大数据的特点。大数据具有体量大、时效性强的特征，并非构造单调，而是类型多样；处理大数据时，传统数据处理系统因数据过载，来源复杂，类型多样等诸多原因性能低下，需要采用以新式计算架构和智能算法为代表的新技术；大数据的应用重在发掘数据间的相关性，而非传统逻辑上的因果关系；因此，大数据的目的和价值就在于发现新的知识，洞悉并进行科学决策。
现代大数据处理技术。主要分为以下几种：
(1)基于分布式文件系统 Hadoop
(2)使用 Map/Reduce 或 Spark 数据处理技术
(3)使用 Kafka 数据传输消息队列及 Avro 二进制格式
◆大数据的利用过程分为：采集、清洗、统计和挖掘几个过程。

大数据处理系统架构分析

◆大数据带来的三大挑战：
1.如何利用信息技术等手段处理非结构化和半结构化数据
2.如何探索大数据复杂性、不确定性特征描述的刻画方法及大数据的系统建模
3.数据异构性与决策异构性的关系对大数据知识发现与管理决策的影响

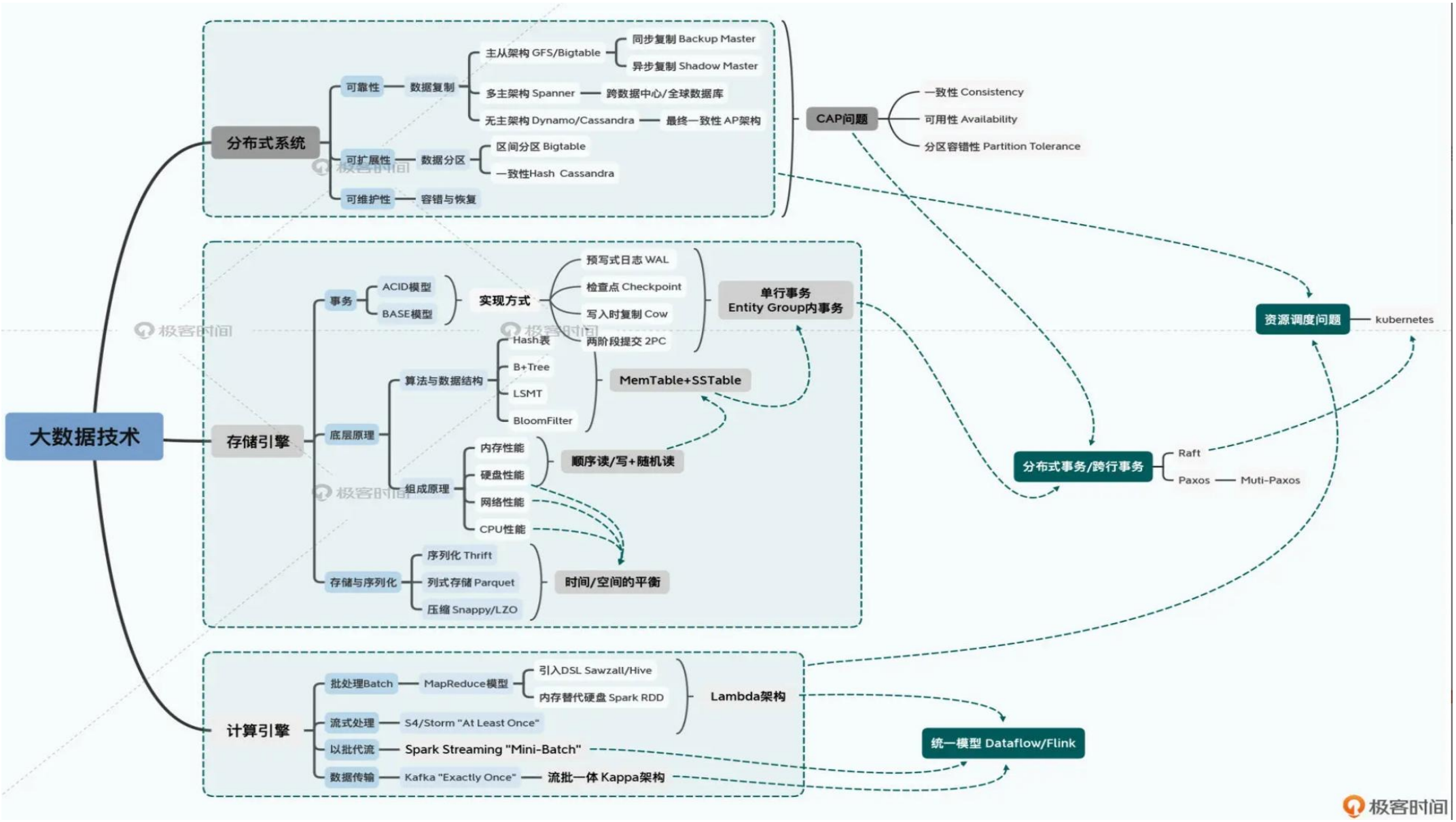
◆大数据处理系统架构特征：（★★）
1.鲁棒性和容错性
2.低延迟读取和更新能力
3.横向扩容
4.通用性
5.延展性
6.即席查询能力
7.最少维护能力
8.可调试性

典型真题

以下不属于大数据处理系统架构特征的是（）。
A.鲁棒性
B.容错性
C.纵向扩容
D.即席查询能力
参考答案：C

目录

大数据处理系统架构技术全景图（教程无）



Lambda架构设计

全量数据进行一次性处理 → 批处理　　高吞吐、高延迟

数据经常实时更新 → 流处理　　低吞吐、低延迟

Lambda架构设计

◆Lambda架构设计目的在于提供一个能满足大数据系统关键特性的架构，包括高容错、低延迟、可扩展等。其整合离线计算与实时计算，融合不可变性、读写分离和复杂性隔离等原则。Lambda是用于同时处理离线和实时数据的，可容错的，可扩展的分布式系统。它具备强鲁棒性，提供低延迟和持续更新。
◆ Lambda架构应用场景：机器学习、物联网、流处理。
◆如图所示，Lambda架构可分解为三层，即批处理层、加速层和服务层。（★★★）

(1)批处理层(Batch Layer): 存储数据集，Batch Layer在数据集上预先计算查询函数，并构建查询所对应的View。Batch Layer可以很好地处理离线数据，但有很多场景数据不断实时生成，并且需要实时查询处理。Speed Layer正是用来处理增量的实时数据。

Batch Layer有两个核心功能：存储数据集和生成Batch View。
该层负责管理主数据集。主数据集中的数据必须具有以下三个属性：
(1)数据是原始的。
(2)数据是不可变的。
(3)数据永远是真实的。

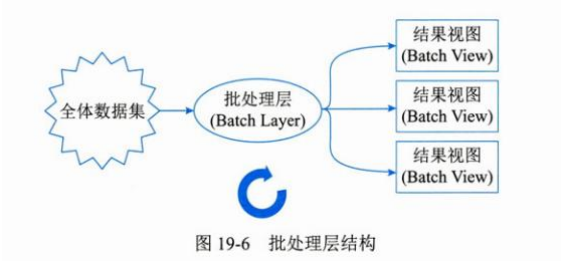
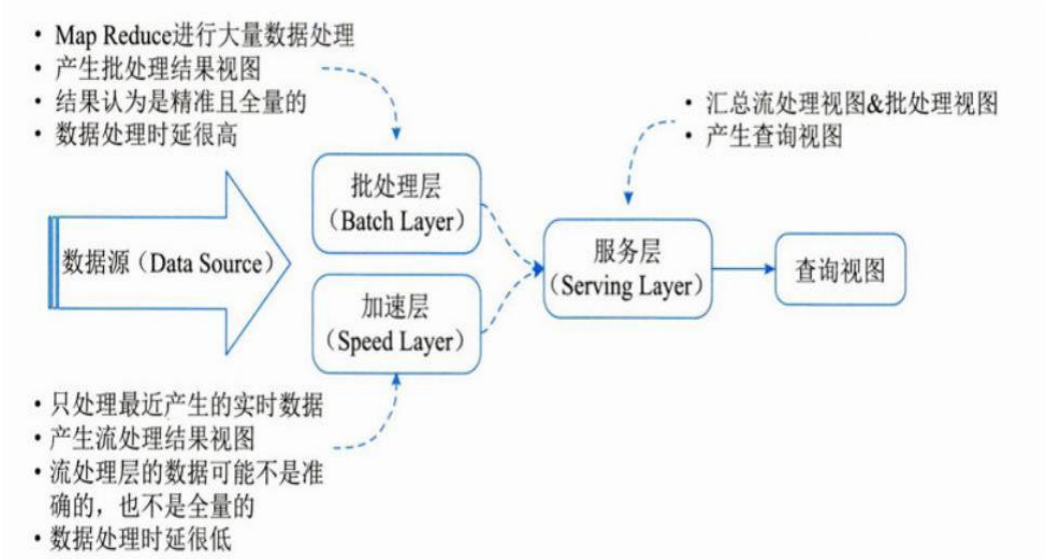


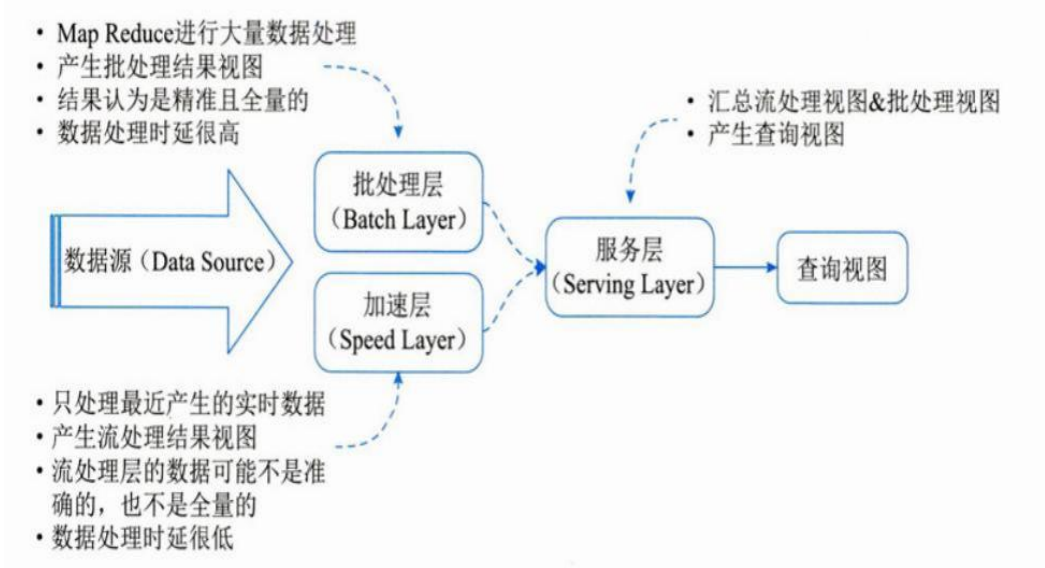
图 19-6　批处理层结构



Lambda架构设计

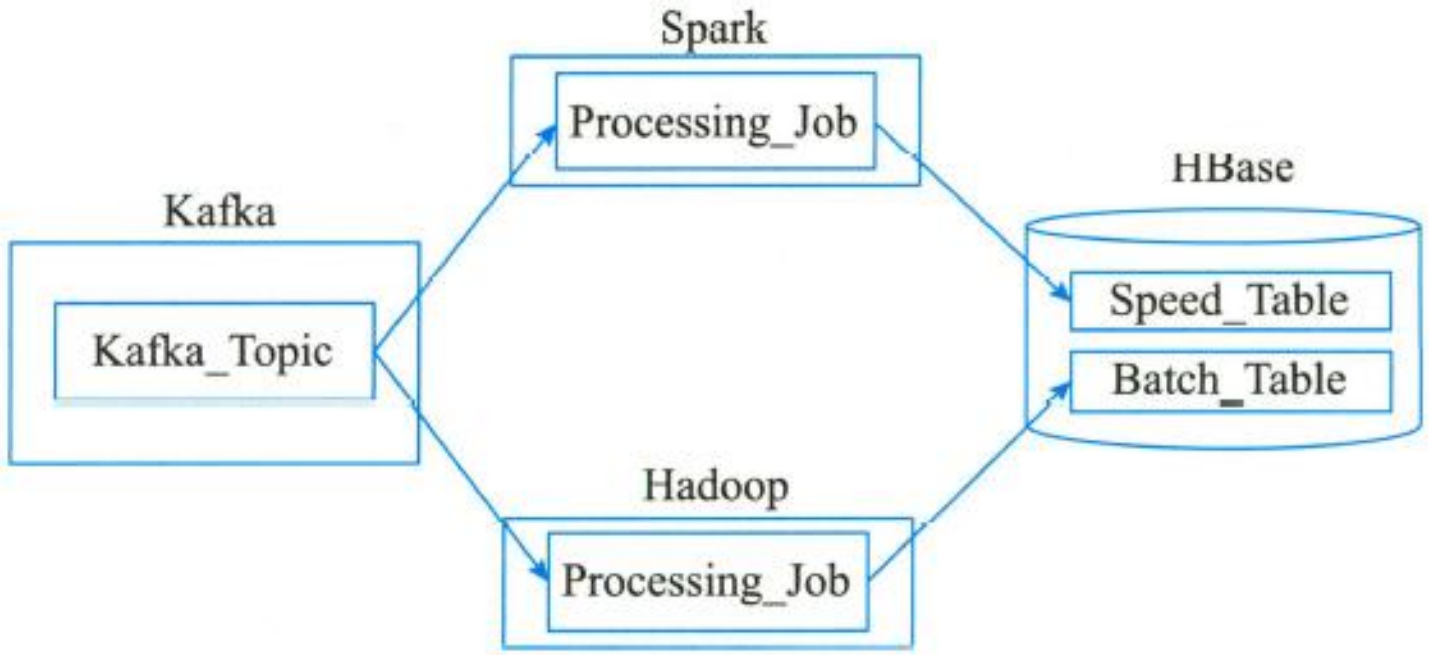
(2)加速层(Speed Layer):Batch Layer处理的是全体数据集，而Speed Layer处理的是最近的增量数据流。Speed Layer 为了效率，在接收到新的数据后会不断更新Real-time View, 而Batch Layer 是根据全体离线数据集直接得到Batch View。

(3)服务层(Serving Layer):Serving Layer 用于合并Batch View 和Real-time View中的结果数据集到最终数据集。用于响应用户的查询请求。



Lambda架构

◆如图所示，在这种Lambda架构实现中，Hadoop(HDFS) 用于存储主数据集，Spark(或Storm)可构成速度层(Speed Layer),HBase (或Cassandra) 作为服务层，由Hive创建可查询的视图。



Lambda架构设计

◆ Hadoop是被设计成适合运行在通用硬件上的分布式文件系统。HDFS是一个具有高度容错性的系统，能提供高吞吐量的数据访问，非常适合大规模数据集上的应用。HDFS放宽了一些约束，以达到流式读取文件系统数据的目的。

◆ Apache Spark是专为大规模数据处理而设计的快速通用的计算引擎。Spark中间输出结果可以保存在内存中，从而不再需要读写HDFS, 因此Spark 能更好地适用于数据挖掘与机器学习等需要迭代的Map Reduce算法。

◆ HBase-Hadoop Database, 是一个高可靠性、高性能、面向列、可伸缩的分布式存储系统，利用HBase技术可在廉价PCServer上搭建起大规模结构化存储集群。

◆ Lambda架构的优点：容错性好、查询灵活度高、易伸缩、易扩展。
◆缺点：全场景覆盖带来的编码开销。针对具体场景重新离线训练一遍益处不大。重新部署和迁移成本很高。

Lambda架构设计

◆事件溯源(Event Sourcing)与Lambda架构
Event Sourcing本质上是一种数据持久化的方式，其由三个核心观点构成：
(1)整个系统以事件为驱动，所有业务都由事件驱动来完成。
(2)事件是核心，系统的数据以事件为基础，事件要保存在某种存储上。
(3)业务数据只是一些由事件产生的视图，不一定要保存到数据库中。
◆Lambda架构中数据集的存储使用的概念与Event Sourcing中的思想完全一致，二者都是在使用统一的数据模型对数据处理事件本身进行定义。这样在发生错误的时候，能够通过模型找到错误发生的原因，对这一事件进行重新计算以丢弃错误信息，恢复到系统应该的正确状态，以此实现了系统的容错性。

◆ CQRS与Lambda架构
◆CQRS架构分离了对于数据进行的读操作(查询)和写(修改)操作。其将能够改变数据模型状态的命令和对于模型状态的查询操作实现了分离。这是领域驱动设计的一个架构模式，主要用来解决数据库报表的输出处理方式。
◆Lambda架构中，数据的修改通过批处理和流处理实现，通过写操作将数据转换成查询时所对应的View。在Lambda架构中，对数据进行查询时，实际上是通过读取View直接得到结果，读出所需的内容。这实际上是一种形式的读写分离。

Kappa架构

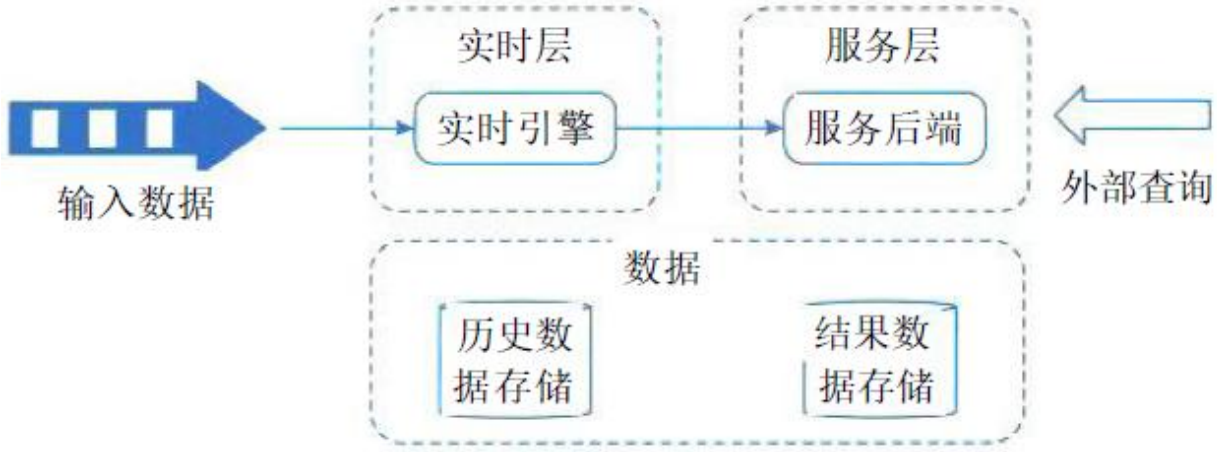
Kappa架构

◆Kappa架构是在Lambda的基础上进行了优化，删除了BatchLayer的架构，将数据通道以消息队列进行替代。Kappa架构分为实时层、服务层。（★★★）

(1)实时层。该层核心功能是处理输入数据，生成实时视图。具体来说是采用流式处理引擎逐条处理输入数据，生成实时视图。架构实现方式是采用Apache Kafka回访数据，然后采用Flink或Spark Streaming进行处理。

(2)服务层。该层核心功能是使用实时视图中的结果数据集响应用户请求。实践中使用数据湖中的存储作为服务层。

◆Kappa架构本质上是通过改进Lambda架构中的加速层，使它既能够进行实时数据处理，同时也有能力在业务逻辑更新的情况下重新处理以前处理过的历史数据。



常见Kappa架构变形

从使用场景上来看，Kappa架构与Lambda相比，主要有两点区别：（★）

(1)Kappa不是Lambda的替代架构，而是其简化版本，Kappa放弃了对批处理的支持，更擅长业务本身为增量数据写入场景的分析需求，例如各种时序数据场景，天然存在时间窗口的概念，流式计算直接满足其实时计算和历史补偿任务需求；

(2)Lambda直接支持批处理，因此更适合对历史数据分析查询的场景，比如数据分析师需要按任意条件组合对历史数据进行探索性的分析，并且有一定的实时性需求，期望尽快得到分析结果，批处理可以更直接高效地满足这些需求。

Kappa架构

Kappa架构的优点和缺点（★★★）
◆Kappa架构的优点在于将实时和离线代码统一起来，方便维护而且统一了数据口径的问题，避免了Lambda架构中与离线数据合并的问题，查询历史数据的时候只需要重放存储的历史数据即可。

◆Kappa的缺点也很明显：
(1)消息中间件缓存的数据量和回溯数据有性能瓶颈。通常算法需要过去180天的数据，如果都存在消息中间件，无疑有非常大的压力。同时，一次性回溯订正180天级别的数据，对实时计算的资源消耗也非常大。
(2)在实时数据处理时，遇到大量不同的实时流进行关联时，非常依赖实时计算系统的能力，很可能因为数据流先后顺序问题，导致数据丢失。
(3)Kappa在抛弃了离线数据处理模块的时候，同时抛弃了离线计算更加稳定可靠的特点。
Lambda虽然保证了离线计算的稳定性，但双系统的维护成本高且两套代码带来后期运维困难。

常见Kappa架构变形

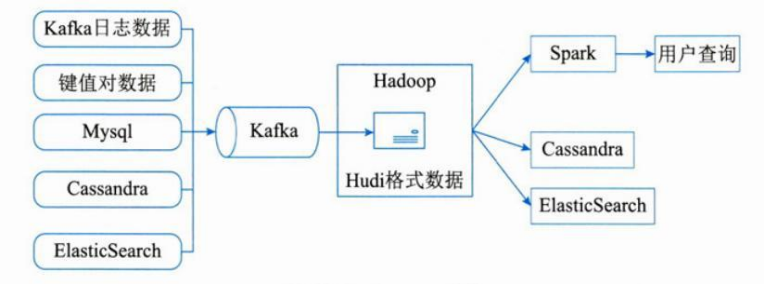
Kappa 架构常见变形是 Kappa+架构，以及混合分析系统 Kappa 架构.

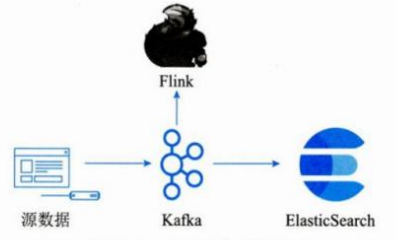
◆ Kappa+是流式数据处理架构，它的核心思想是让流计算框架直接读HDFS里的数据仓库数据，一并实现实时计算和历史数据backfll计算，不需要为backfll作业长期保存日志或者把数据拷贝回消息队列。开发了Apache hudi框架来存储数据仓库数据， hudi支持更新、删除已有parquet数据，也支持增量消费数据更新部分。

◆如图所示，将不同来源的数据通过Kafka导入到Hadoop中，通过HDFS来存储中间数据，再通过spark对数据进行分析处理，最后交由上层业务进行查询。

◆混合分析系统的Kappa架构：Lambda和Kappa架构都还有展示层的困难点，结果视图如何支持热点数据查询分析，一个解决方案是在Kappa基础上衍生数据分析流程。

◆在基于使用Kafka +Flink构建Kappa流计算数据架构，针对Kappa架构分析能力不足的问题，再利用Kafka对接组合Elastic-Search实时分析引擎，部分弥补其数据分析能力。但是ElasticSearch也只适合对合理数据量级的热点数据进行索引，无法覆盖所有批处理相关的分析需求，这种混合架构某种意义上属于Kappa和Lambda间的折中方案。





Lambda架构与Kappa架构的特性对比

◆Lambda 架构与 Kappa 架构的抉择。两种架构特性对比（★★★）

对比内容	Lambda 架构	Kappa 架构
复杂度与开发维护成本	维护两套系统（引擎）， 复杂度高，成本高	维护一套系统（引擎） 复杂度低，成本低
计算开销	周期性批处理计算，持续实时计算 计算开销大	必要时进行全量计算 计算开销相对较小
实时性	满足实时性	满足实时性
历史数据处理能力	批式全量处理，吞吐量大 历史数据处理能力强 批视图与实时视图存在冲突可能	流式全量处理，吞吐量相对较低 历史数据处理能力相对较弱

Lambda架构与Kappa架构设计选择

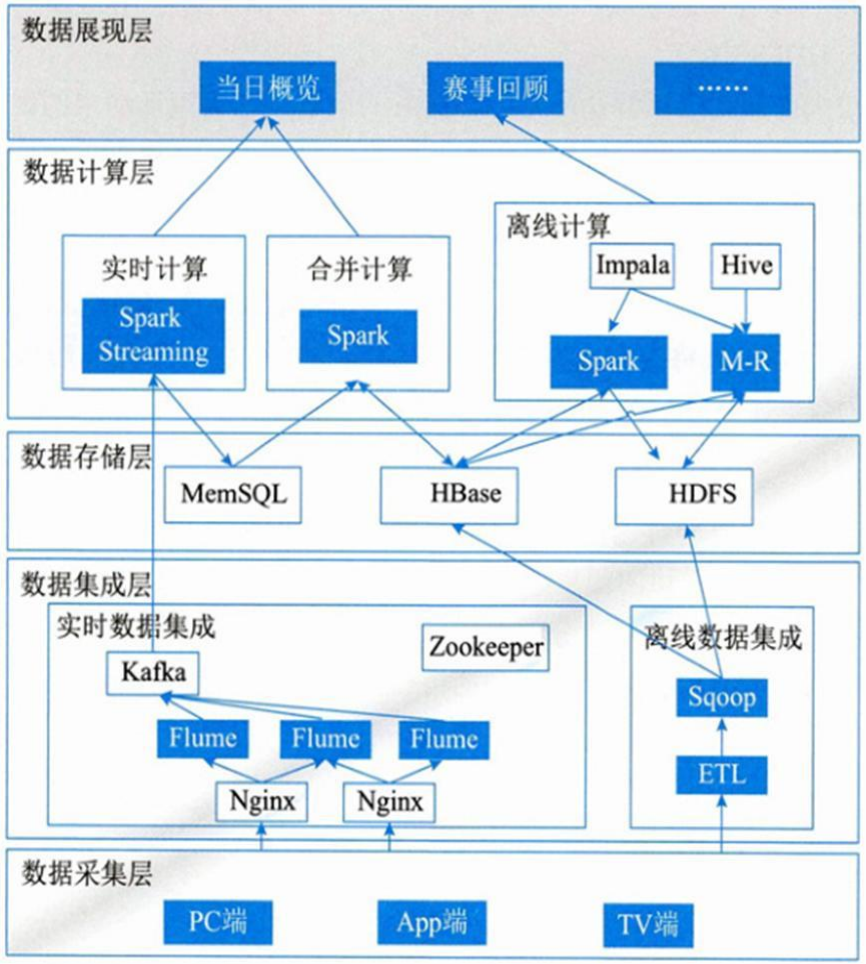
◆对于两种架构设计的选择可以从以下几个方面考虑（★★★）

设计考虑	Lambda 架构	Kappa 架构
业务需求与技术要求	依赖 Hadoop、Spark、Storm 技术	依赖 Flink 计算引擎，偏流式计算
复杂度	实时处理和离线处理结果可能不一致	频繁修改算法模型参数
开发维护成本	成本预算充足	成本预算有限
历史数据处理能力	频繁使用海量历史数据	仅使用小规模数据集

目录

大数据架构设计案例分析

1．大规模视频网络。某网采用以 Lambda 架构搭建的大数据平台处理里约奥运会大规模视频网络观看数据，具体平台架构设计



数据计算层可以分为离线计算、实时计算、合并计算三个部分。

（1）离线计算部分：用于存储持续增长的批量离线数据，并且会周期性使用 Spark 和Map/Reduce 进行批处理，将批处理结果更新到批视图之后使用 Impala 或者 Hive 建立数据仓库，将结果写入 HDFS 中。

（2）实时计算部分：采用 Spark Streaming，只处理实时增量数据，将处理后的结果更新到实时视图。

（3）合并部分：合并批视图和实时视图中的结果，生成最终数据集，将最终数据集写入 HBase数据库中用于响应用户的查询请求。

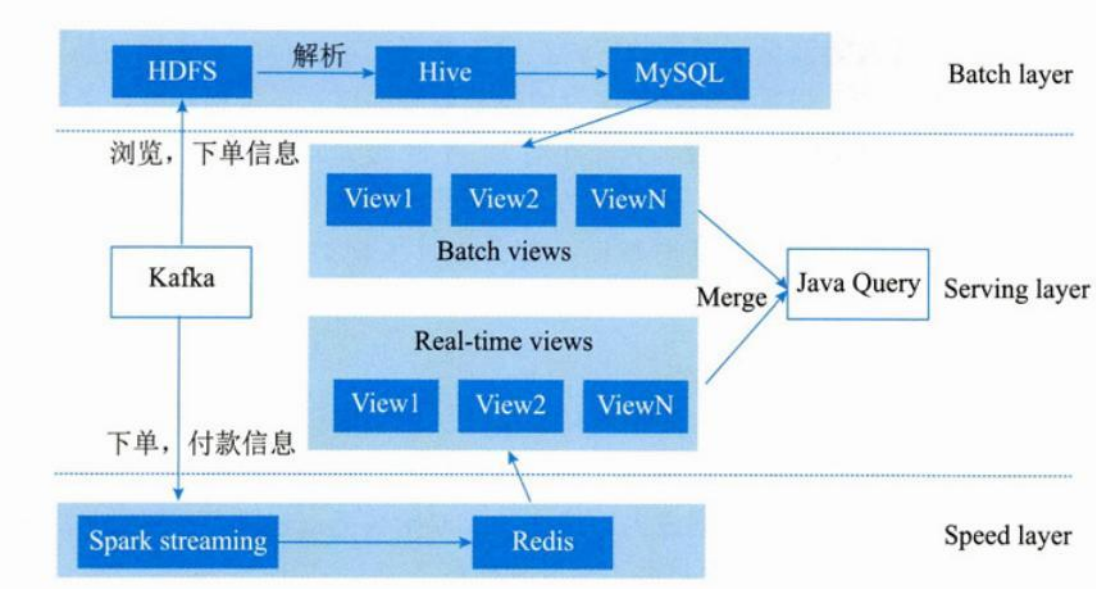
大数据架构设计案例分析

2．广告平台。某网基于 Lambda 架构的广告平台，分为批处理层（Batch layer）、加速层（Speed layer）、服务层（Serving layer）

（1）批处理层：每天凌晨将 Kafka 中浏览、下单等消息同步到 HDFS 中，将 HDFS 中数据解析为 Hive 表，然后使用 HQL 或 Spark SQL 计算分区统计结果 Hive 表，将 Hive 表转储到 MySQL 中作为批视图。

（2）加速层：使用 Spark Streaming 实时监听 Kafka 下单、付款等消息，计算每个追踪链接维度的实时数据，将实时计算结果存储在 Redis 中作为实时视图。

（3）服务层：采用 Java Web 服务，对外提供 HTTP 接口，Java Web 服务读取 MySQL 批视图表和 Redis 实时视图表。



大数据架构设计案例分析

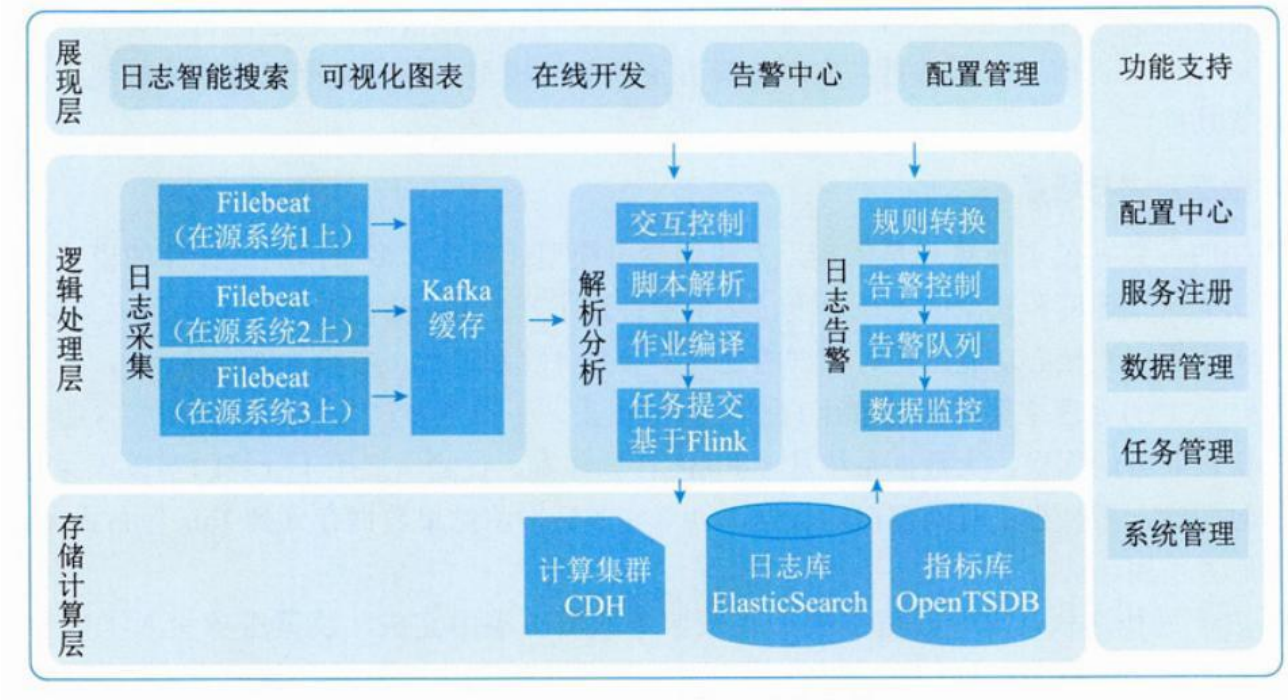
3．某证券公司智能决策大数据系统。该系统是一个基于 Kappa 架构的的实时日志分析平台，具体的实时处理过程如下：

（1）日志采集：用统一的数据处理引擎 Filebeat 实时采集日志并推送给 Kafka 缓存。

（2）日志清洗解析：利用基于大数据计算集群的 Flink 计算框架实时读取 Kafka 消息并进行清洗，解析日志文本转换成指标。

（3）日志存储：日志转储到 ElasticSearch 日志库，指标转储到 OpenTSDB 指标库。

（4）日志监控：单独设置告警消息队列，保持监控消息时序管理和实时推送。



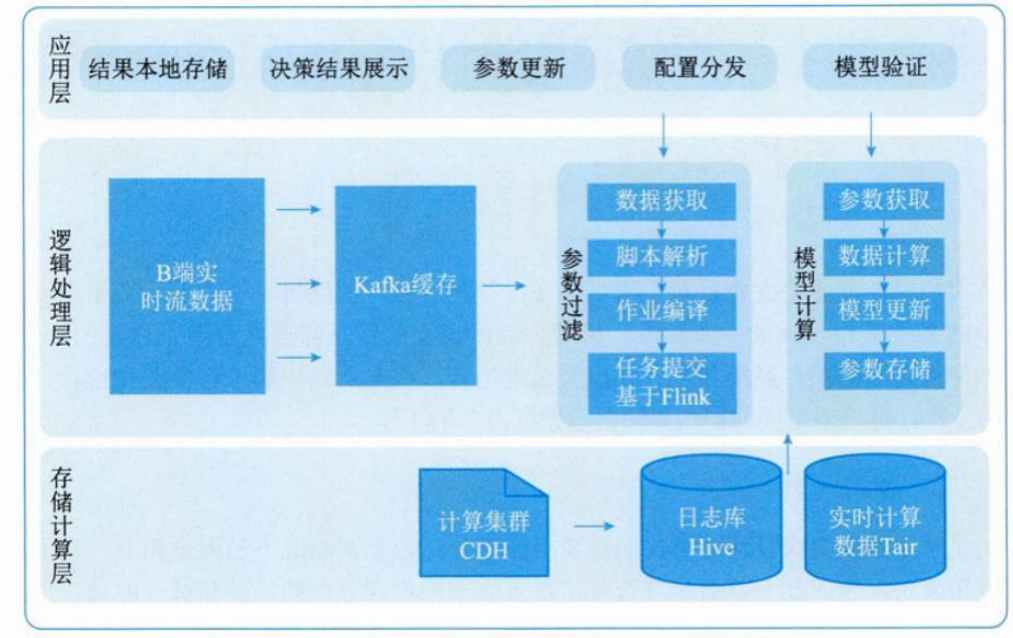
大数据架构设计案例分析

4.电商智能决策大数据系统。该智能决策大数据平台基于 Kappa 架构，使用统一的数据处理引擎 Funk 可实时处理流数据，并将其存储到数据仓库工具 Hive 与分布式缓存Tair 中，以供后续决策服务的使用。实时处理的过程如下：

（1）数据采集：B 端实时采集用户点击、下单、广告曝光、出价等数据然后推送给 Kafka 缓存。

（2）数据清洗聚合：由 Flink 实时读取 Kafka 消息，按需过滤参与业务需求的指标，将聚合时间段的数据转换成指标。

（3）数据存储：Flink 将计算结果转储至 Hive 日志库，将模型需要的参数转储至实时计算数据库 Tair 缓存，然后后续决策服务从 Tair 中获取数据进行模型训练。



典型真题

Lambda 架构分为三层：（1）的核心功能是存储主数据集。（ 2）的核心功能是处理增量实时数据，生成实时视图，快速执行即席查询。（3 ）的核心功能是响应用户请求，合并批视图和实时视图中的结果数据集得到最终数据集。

（1）A. 批处理层.　B. 流处理层　C. 加速层　D.存储层

（2）A. 批处理层　B. 服务层　C. 加速层.　D.视图层

（3）A. 视图层　B. 流处理层　C. 服务层.　D.存储层

典型真题

解析：Lambda 架构分为三层：
（1）批处理层。该层核心功能是存储主数据集，主数据集数据具有原始，不可变，真实的特征。批处理层周期性将增量数据转储至主数据集，并在主数据集上执行批处理，生成批视图。架构实现方面可以使用 Hadoop HDFS 或 HBase 存储主数据集，再利用 Spark 或 Map/Reduce 执行周期批处理，之后使用 Map/Reduce 创建批视图。
（2）加速层。该层的核心功能是处理增量实时数据，生成实时视图，快速执行即席查询。架构实现方面可以使用 Hadoop HDFS 或 HBase 存储实时数据，利用 Spark 或 Storm 实现实时数据处理和实时视图。
（3）服务层。该层的核心功能是响应用户请求，合并批视图和实时视图中的结果数据集得到最终数据集。具体来说就是接收用户请求，通过索引加速访问批视图，直接访问实时视图，然后合并两个视图的结果数据集生成最终数据集，响应用户请求。架构实现方面可以使用 HBase 或Cassandra 作为服务层，通过 Hive 创建可查询的视图。
答案：A、C、C

THANKS