



NEW PLAN

软件高级架构师

• — 段 新 征 程 •





大数据处理系统架构分析

■ 大数据带来的三大挑战：

- 如何利用信息技术等手段处理非结构化和半结构化数据
- 传统数据库系统主要处理结构化数据，即具有明确定义的表格和字段，比如银行的交易记录。然而，大数据环境中经常涉及非结构化和半结构化数据，如各种格式的文本文档、图像、音频、视频和社交媒体数据。这些数据不容易以传统的表格形式表示，因此需要新的方法和工具来处理和分析它们。这包括自然语言处理技术、图像识别、文本挖掘等。
- 如何探索大数据复杂性和不确定性及大数据的系统建模
- 大数据通常具有复杂性和不确定性特征。数据量庞大，包含多个维度和特征，涉及多个数据源，导致数据的复杂性增加。此外，大数据往往包含噪音、缺失数据和不确定性，使数据分析更加具有挑战性。解决这一挑战需要开发新的数据挖掘和分析算法，以提取有用的信息并考虑不确定性。例如，金融领域的股票市场数据包含大量的噪音和波动性，需要复杂的模型来预测股价。
- 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
- 大数据往往涉及不同数据源、格式和结构，这引入了数据异构性。决策异构性涉及到不同决策者、利益相关者和利益冲突。这些异构性对大数据的知识发现和决策制定产生了影响。例如，一家医院可能使用多个系统记录病人信息，其中包括电子病历、医疗图像和实验室数据。整合这些异构数据源以支持医疗决策可能涉及解决数据格式不同的问题，同时考虑到不同医生和病人的利益

大数据处理系统架构分析

■ 大数据系统应该具有的特征：

- 鲁棒性和容错性：需要能够处理各种异常情况，例如数据损坏、硬件故障、软件错误等。系统应能够自动恢复或重试失败的操作，并尽量减少对应用程序的影响。使用冗余数据存储机制，例如副本或RAID，来防止数据丢失。
- 低延迟读取和更新能力：需要能够快速读取和更新数据，以满足实时分析和决策的需求。系统应支持各种数据访问模式，例如批量访问、实时访问和随机访问。比如可以使用高速读取硬件设备、缓存机制、索引技术来加快读取和更新速度
- 横向扩容：需要能够根据需要进行横向扩容，以满足不断增长的数据量和计算需求。系统应能够自动添加或删除节点，而无需中断应用程序的运行。
- 通用性：需要能够支持各种数据类型和格式，并能够与各种第三方工具和框架集成。系统应提供开放的 API 和接口，以便开发人员轻松地构建和扩展应用程序。
- 延展性：需要能够支持新的功能和需求，并能够适应不断变化的技术环境。系统应提供良好的设计和架构，以便易于扩展和维护。
- 即席查询能力：是指根据需求和特定目的，临时创建和执行的查询操作。这些查询通常是针对数据库或数据集中的特定信息需求而制定，而非预先计划或事先定义的查询
- 最少维护能力：需要易于维护和管理，以便降低运营成本。系统应提供自动化的监控和管理工具，以便管理员可以轻松地识别和解决问题。
- 可调试性：需要易于调试，以便开发人员可以快速识别和解决问题。系统应提供良好的日志记录和跟踪机制，以便开发人员可以跟踪应用程序的运行状况

Lambda架构

- 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。

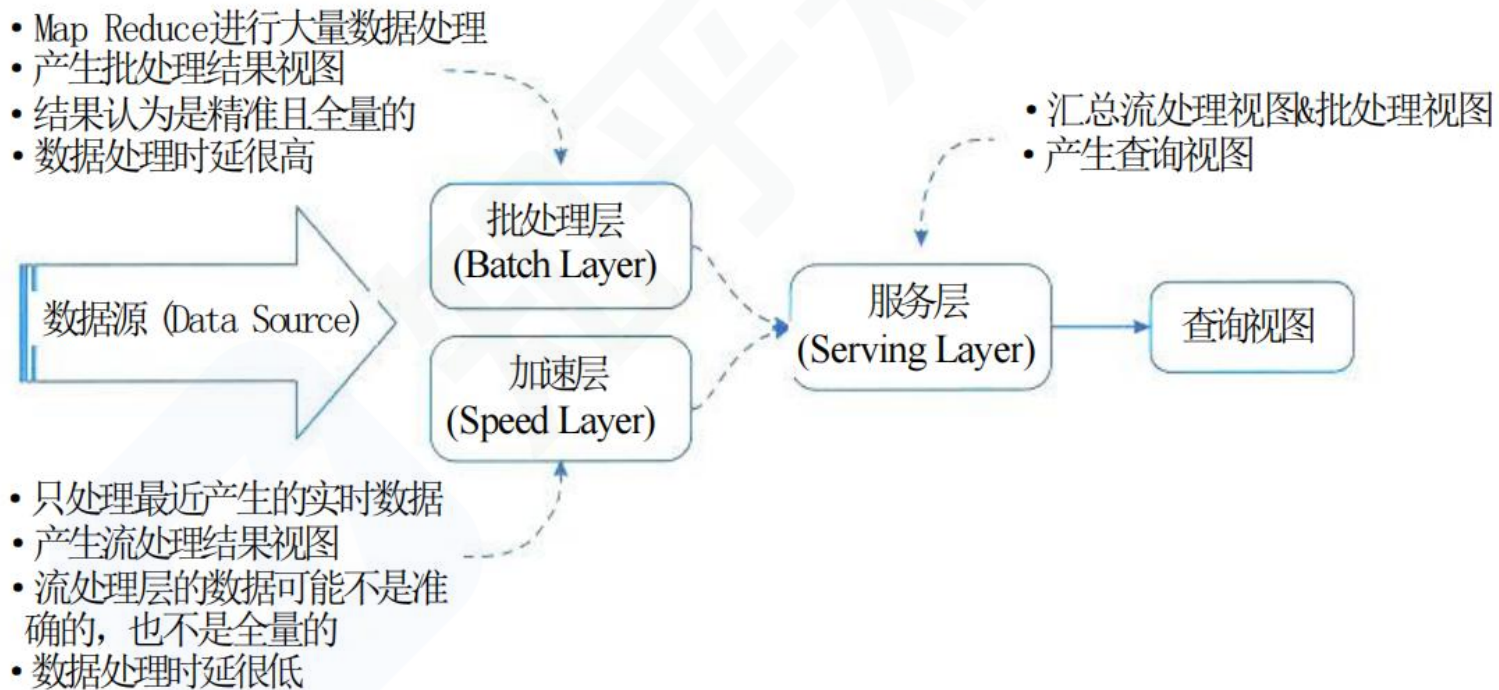


图19-4 Lambda架构

Lambda架构

- 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。
- 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。
- 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。
- 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)可构成加速层(Speed Layer), HBase (或Cassandra)作为服务层, 由Hive创建可查询的视图。

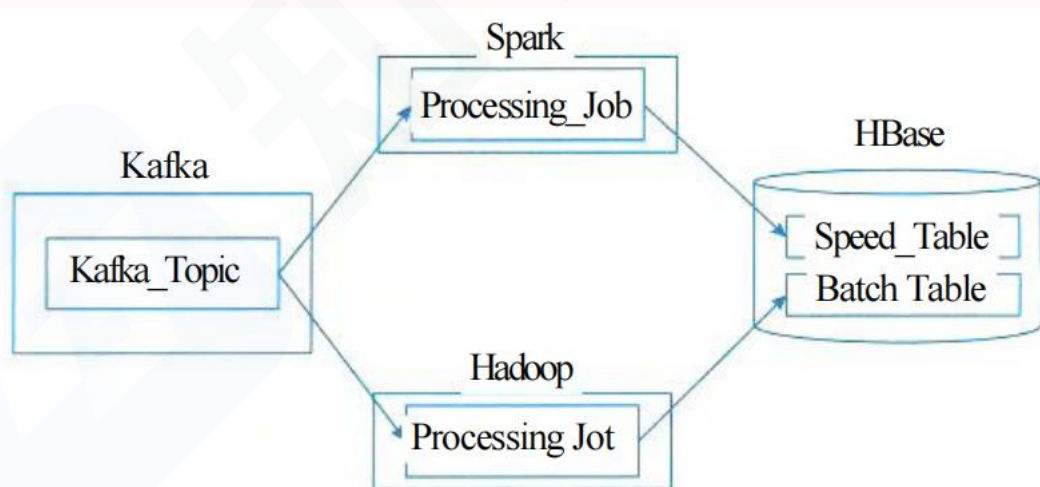


图19-9技术选型

Lambda架构

- Hadoop: 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: **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: 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。

提示:

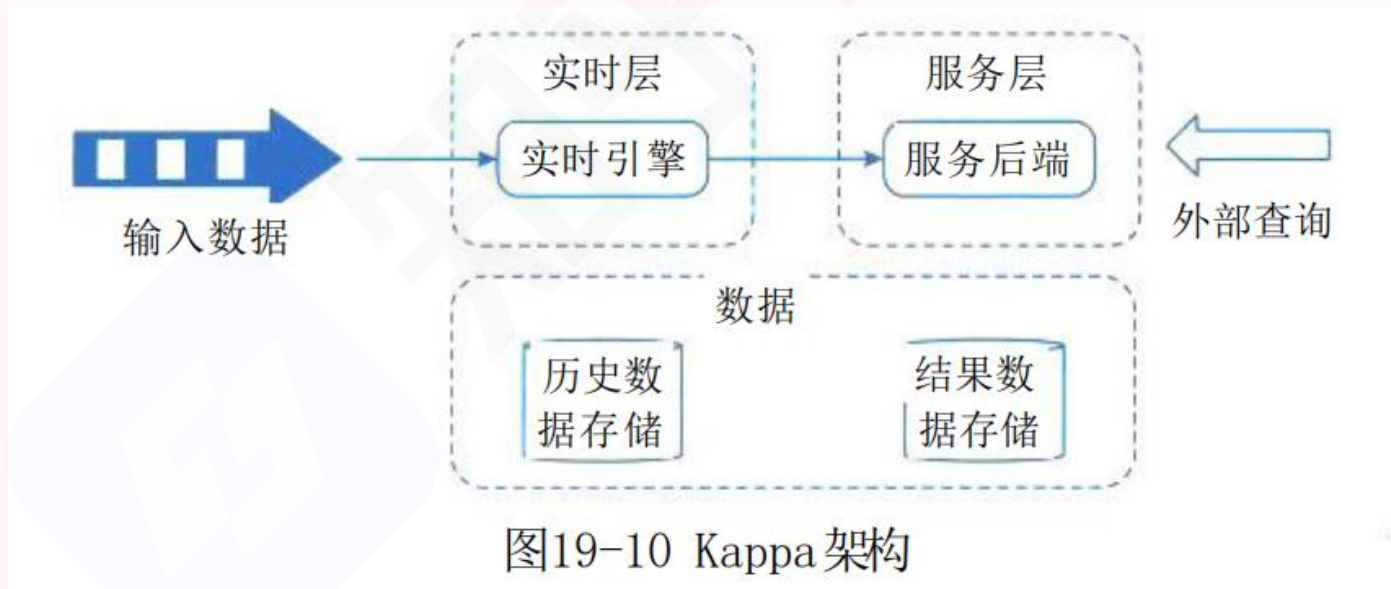
- Hadoop是一个开源的分布式计算框架, 用于存储和处理大规模数据。它包括两个主要组件: Hadoop分布式文件系统 (HDFS) 和MapReduce。Hadoop的设计目标是分布式存储和计算, 使其能够处理大数据。
- HDFS用于大数据存储, 分布式存储在Hadoop集群的多个节点上, 只用来存储数据, 不提供分析和查询
- Hive是一个数据仓库工具, 用于SQL查询和数据分析
- Hbase是一个分布式的NoSQL数据库, 构建在HDFS之上, 用于实时数据存储和访问。

这些组件通常一起使用, 以满足大数据处理的不同需求。例如, 数据可以存储在HDFS中, 然后使用Hive执行查询和分析, 同时使用HBase存储实时数据以供应用程序访问。根据具体需求, 这些组件可以组合使用, 以构建全面的大数据解决方案。

- Lambda架构的优点: **容错性好、查询灵活度高、易伸缩、易扩展**。
- Lambda架构的缺点: **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。

Kappa架构

- Kappa架构的原理就是：在Lambda的基础上进行了优化，**删除了Batch Layer 的架构**，将数据通道以消息队列进行替代。因此对于Kappa架构来说，依旧以流处理为主，但是数据却在数据湖层面进行了存储，**当需要进行离线分析或者再次计算的时候，则将数据湖的数据再次经过消息队列重播一次则可。**
- 如图所示，输入数据**直接由实时层的实时数据处理引擎对源源不断的源数据进行处理**，再由服务层的服务后端进一步处理以提供上层的业务查询。而中间结果的数据都是需要存储的，这些数据包括历史数据与结果数据，统一存储在存储介质中。



Kappa架构

- 从使用场景上来看，Kappa架构与Lambda相比，主要有**两点区别**：
 - 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；
 - 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**。
- 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：
 - 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。
 - 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。
 - 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。

Lambda架构与Kappa架构的对比和设计选择

根据两种架构对比分析，将**业务需求、技术要求、系统复杂度、开发维护成本和历史数据处理能力**作为选择考虑因素。而计算开销虽然存在一定差别，但是相差不是很大，所以不作为考虑因素。

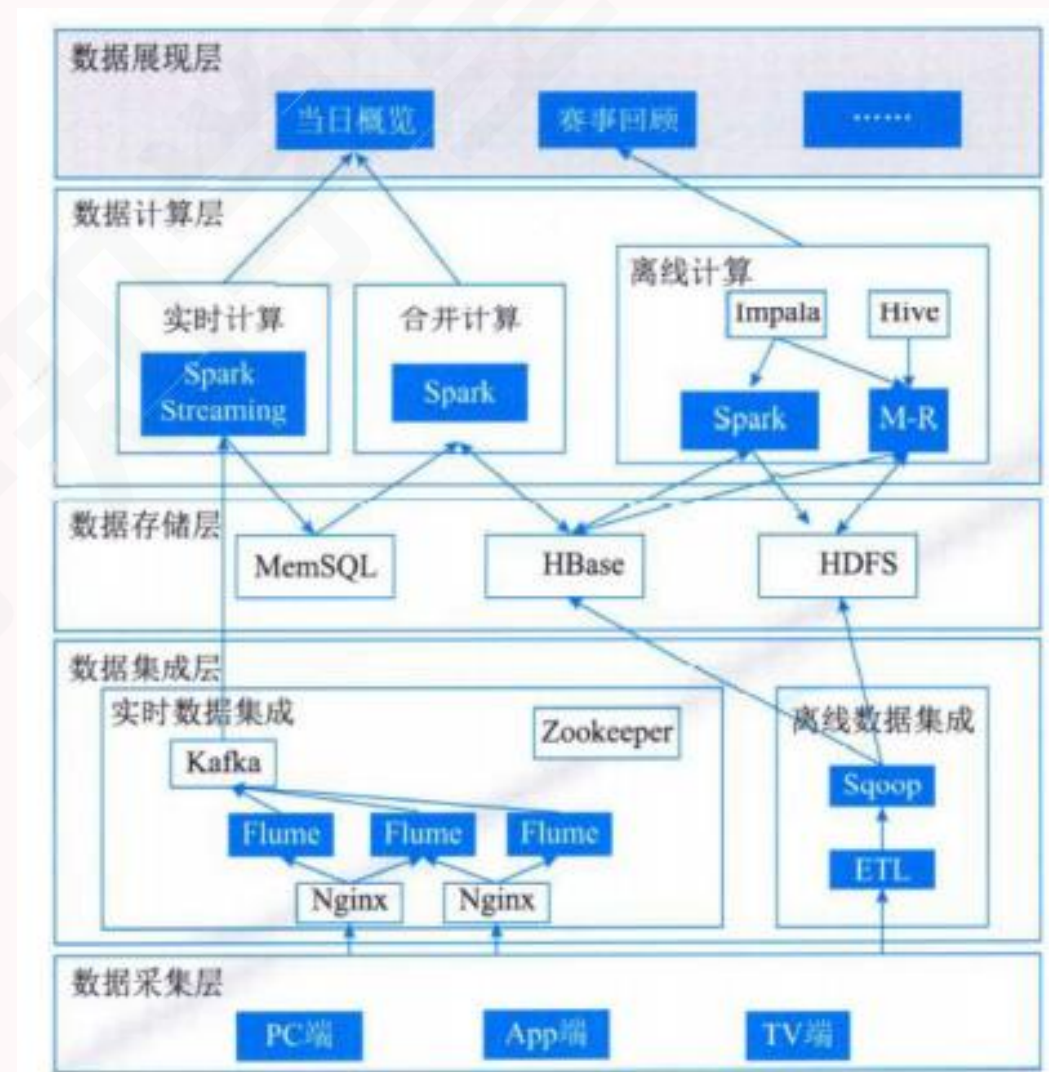
- 业务需求与技术要求：如果业务**对于Hadoop、Spark、Strom(一种开源的实时流数据处理系统，它是一个分布式计算框架，专注于实时数据流处理)等关键技术有强制性依赖**，选择Lambda架构可能较为合适；如果处理数据偏好于流式计算，又依赖Flink计算引擎，那么选择Kappa架构可能更为合适。
- 复杂度：如果项目中需要频繁地对算法模型参数进行修改，Lambda 架构需要反复修改两套代码，则显然不如Kappa架构简单方便。同时，如果算法模型支持同时执行批处理和流式计算，或者希望用一份代码进行数据处理，那么可以选择Kappa架构。
- 开发维护成本：Lambda架构需要有一定程度的开发维护成本，包括两套系统的开发、部署、测试、维护，适合有足够经济、技术和人力资源的开发者。而Kappa架构只需要维护一套系统。
- 历史数据处理能力：有些情况下，项目会频繁接触海量数据集进行分析，应该选择Lambda架构。如果**始终使用小规模数据集，流处理系统完全可以使用**，则应该选择Kappa架构。

表19-1 Lambda架构和Kappa 架构对比

对比内容	Lambda 架构	Kappa架构
复杂度与开发、维护成本	需要维护两套系统(引擎), 复杂度高, 开发、维护成本高	只需要维护一套系统(引擎), 复杂度低, 开发、维护成本低
计算开销	需要一直运行批处理和实时计算, 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	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

大数据架构设计案例分析-Lambda架构在某网奥运中的大数据应用

- Lambda架构实时处理层采用增量计算实时数据的方式，可以在集群规模不变的前提下，秒级分析出当日概览所需要的信息。赛事回顾模块需要展现自定义时间段内的历史最高在线人数、逐日播放走势、直播最高在线人数和点播视频排行等海量数据的统计信息，由于奥运期间产生的数据通常不需要被经常索引、更新，因此要求采用不可变方式存储所有的历史数据，以保证历史数据的准确性。
- Lambda架构的批处理层采用不可变存储模型，**不断地往主数据集后追加新的数据**，恰好可以满足对奥运数据的大规模统计分析要求。具体架构如下图：
- 提示：ETL代表提取、转换、加载（Extract, Transform, Load）。它是一种数据集成过程，涉及从源系统中提取数据，将其转换为适合分析的格式，并加载到目标系统（例如数据仓库或数据湖）中。
- 提示：Sqoop 是一款开源工具，用于在 Apache Hadoop 和关系数据库之间进行数据传输
- 提示：M-R就是MapReduce
- 提示：Impala 是一款开源的、高性能的、面向列的 SQL 查询引擎，用于分析存储在 Apache Hadoop 集群中的大量数据

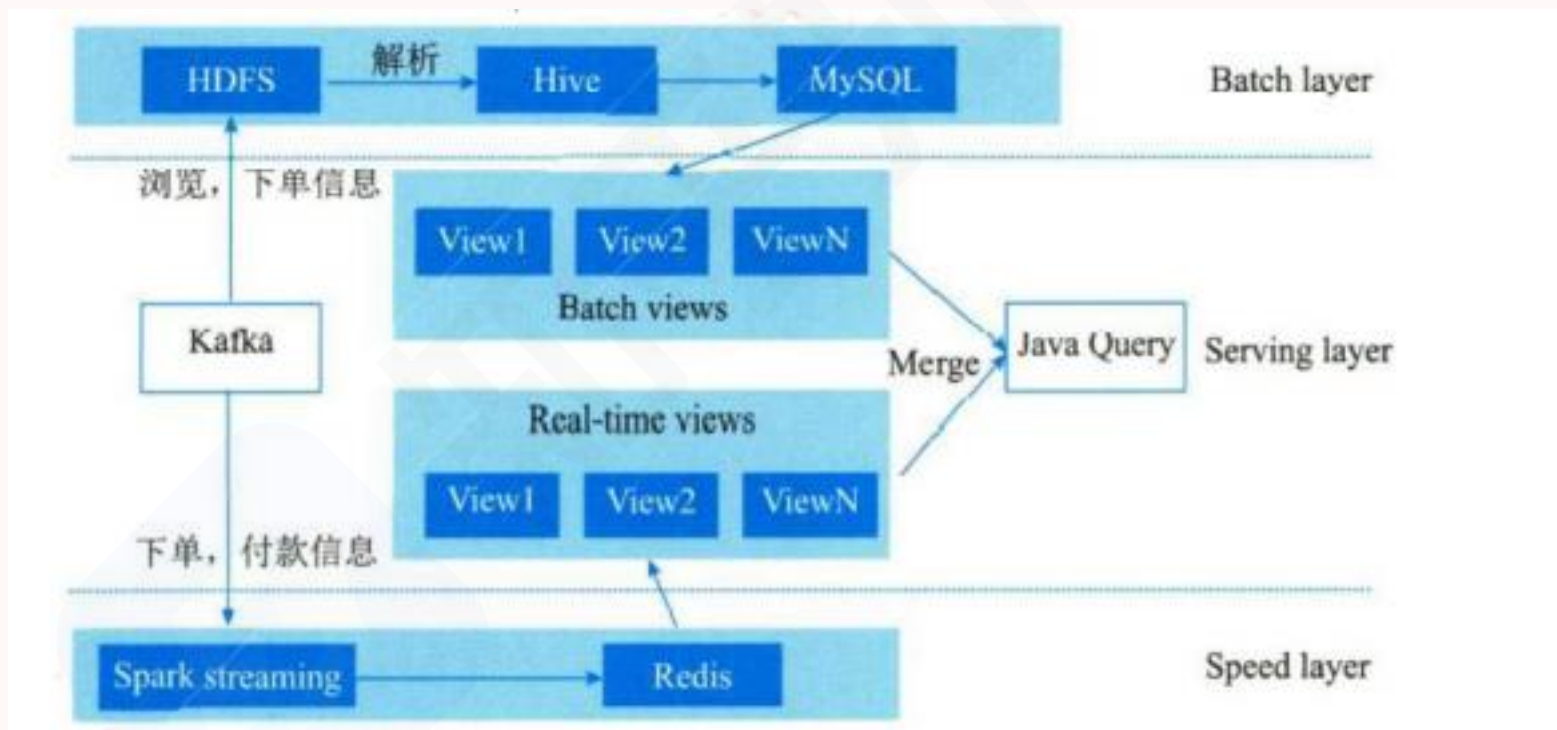


大数据架构设计案例分析-Lambda架构在某网广告平台的应用与演进

某网广告平台展示的数据指标包含两类：曝光类(包括曝光数、点击数、点击单价、花费)，转化类(包括转化下单数、转化下单金额、转化付款数、转化付款金额)。前一类的数据主要由流量方以接口的方式提供(比如对接的腾讯广点通平台)，后一类则是某网特有的数据，通过买家的浏览、下单、付款日志算出来。

第一版架构：

第一版采用了**典型的Lambda架构形式**，架构图如图所示。



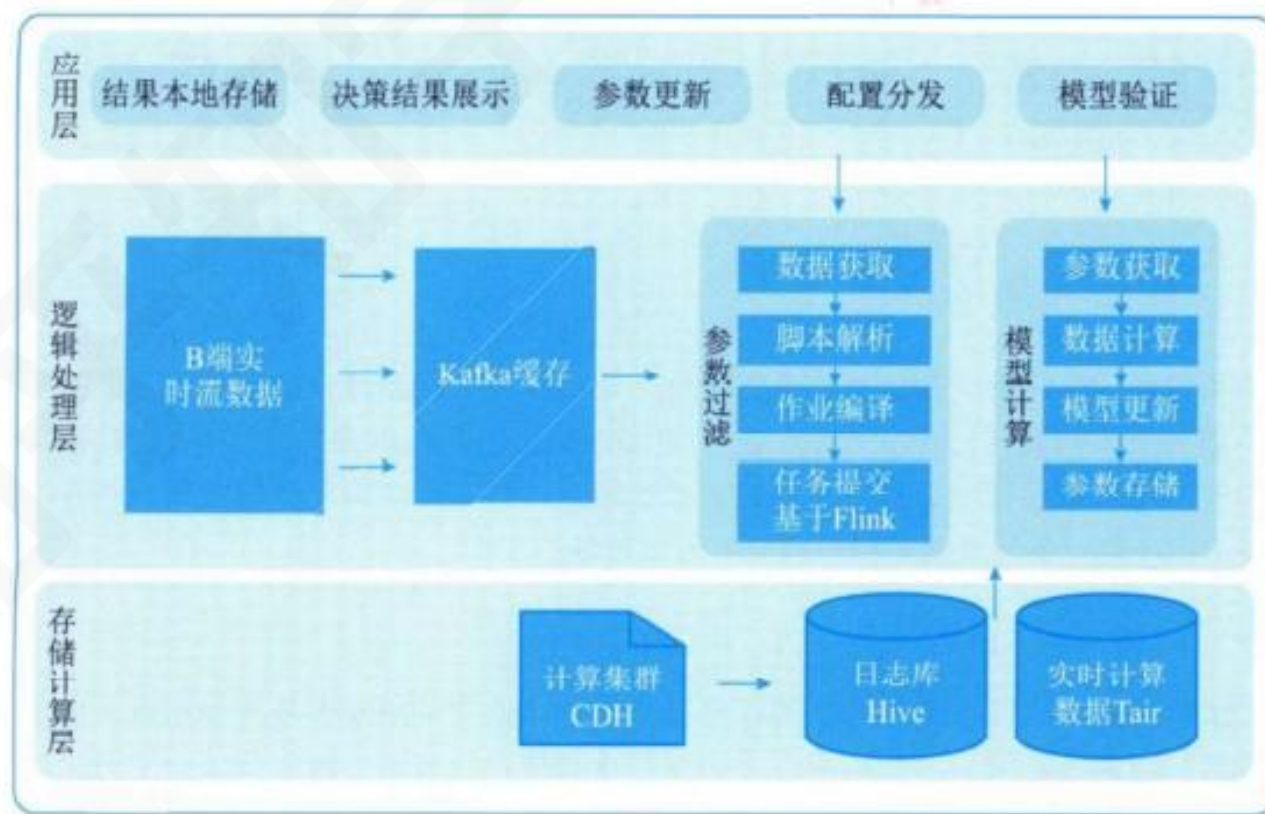
大数据架构设计案例分析-某电商智能决策大数据系统

实时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Flink可实时处理流数据**，并将其存储到Hive与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，即**B端系统会实时收集用户的点击**，下单以及广告的曝光和出价数据并输出到Kafka缓存。

(2)数据的清洗与聚合，即基于大数据计算集群**Flink计算框架**，**实时读取Kafka中的实时流数据**，**过滤出需要参与计算的字段**，根据业务需求，聚合指定时间端的数据并转换成指标。

(3)数据存储，**即将Flink计算得到数据存储到Hive日志库中**，**需要参与模型计算计算的字段存储到Tair分布式缓存中**。当需要进行模型计算时，决策服务会从Tair中读取数据，进行模型的计算，得到新的决策参数和模型。决策服务基于微服务架构，客户端部署在业务方系统中，服务端主要用于计算决策参数和模型，当服务端计算得到新的参数，此时会通过Zookeeper通知部署到业务方系统的客户端，客户端此时会拉取新的参数并存储到本地，并且客户端提供了获取参数的接口，业务方可以无感知调用。



2023年案例分析第一题回忆版

某网作为某电视台在互联网上的大型门户入口，某一年成为某奥运会中国大陆地区的特权转播商，独家全程直播了某奥运会全部的赛事，积累了庞大稳定的用户群，这些用户在使用各类服务过程中产生了大量数据，对这些海量数据进行分析与挖掘，将会对节目的传播及商业模式变现起到重要的作用。

该奥运期间需要对增量数据在当日概览和赛事回顾两个层面上进行分析。其中，当日概览模块需要秒级刷新直播在线人数、网站的综合浏览量、页面停留时间、视频的播放次数和平均播放时间等千万级数据量的实时信息，而传统的分布式架构采用重新计算的方式分析实时数据，在不扩充以往集群规模的情况下，无法在几秒内分析出重要的信息。赛事回顾模块需要展现自定义时间段内的历史最高在线人数、逐日播放走势、直播最高在线人数和点播视频排行等海量数据的统计信息，由于该奥运期间产生的数据通常不需要被经常索引、更新，因此要求采用不可变方式存储所有的历史数据，以保证历史数据的准确性。

【问题1】(8分)

请根据Lambda架构和Kappa架构特点，填写以下表格。

对比内容	Lambda架构	Kappa架构
复杂度与开发、维护成本	需要维护（1）套系统（引擎），复杂度（3），开发、维护成本（3）	只需要维护（2）套系统（引擎），复杂度（4），开发、维护成本（4）
计算开销	需要一直运行批处理和实时计算，计算开销大	（5）
实时性	满足实时性	（6）
历史数据处理能力	批式全量处理，吞吐量（7），历史数据处理能力强	流式全量处理，吞吐量相对较低，历史数据处理能力（8）

2023年案例分析第一题回忆版

【问题2】(9分)

下图1给出了某网奥运的大数据架构图，请根据下面的 (a)~(n) 的相关技术；判断这此技术属于架构图的哪个部分，补充完善下图1的(1)-(9)的空白处。

- | | |
|--------------------|----------|
| (a)Nginx | (b)Hbase |
| (c)Spark Streaming | (d)Spark |
| (e)MapReduce | (f)ETL |
| (g)MemSQL | (h)HDFS |
| (i)Sqoop | (j)Flume |
| (k)数据存储层 | (l)kafka |
| (m)业务逻辑层 | (n)数据采集层 |

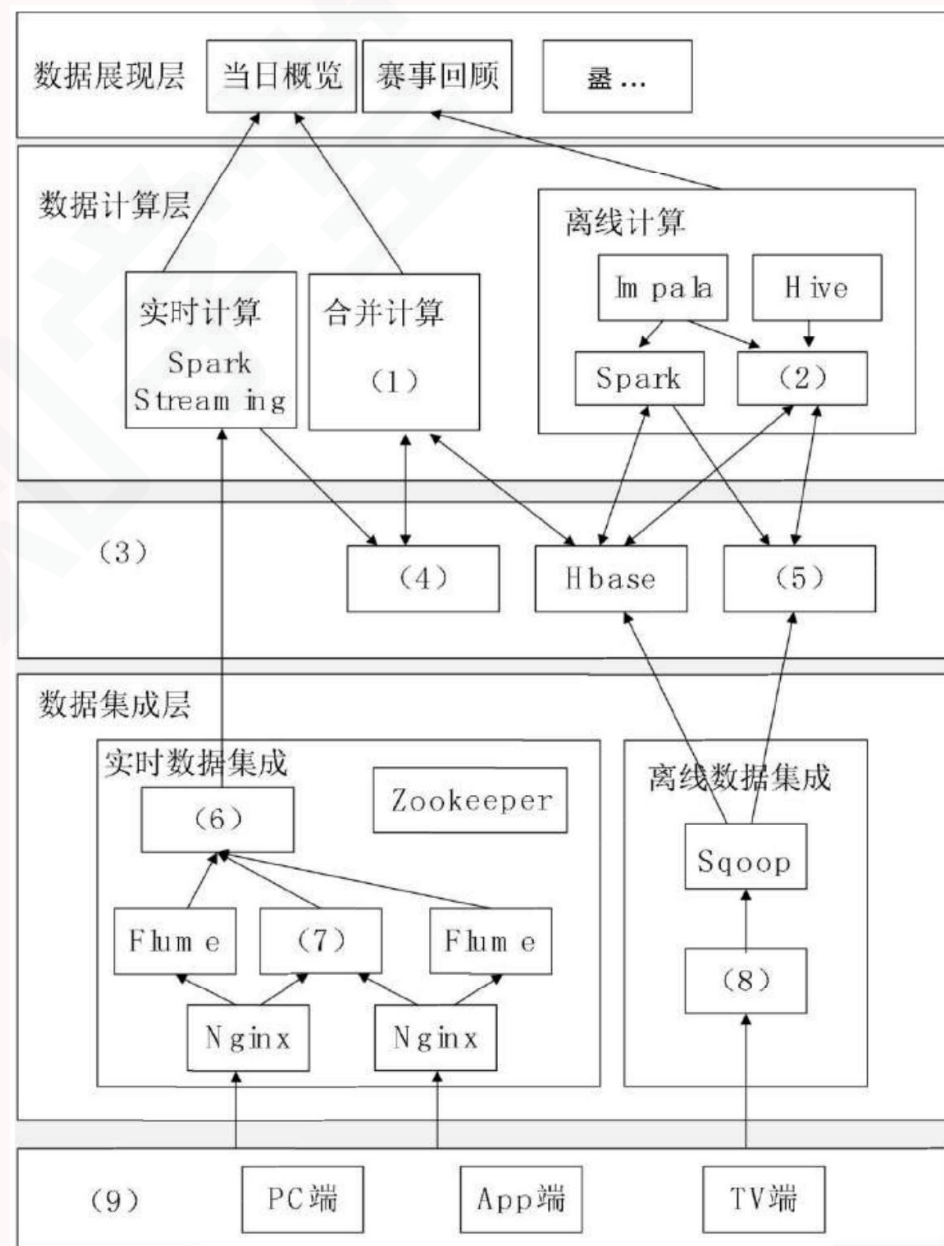
【问题3】(8分， 每空2分)

大数据的架构包括了Lambda架构和Kappa架构， Lambda架构分解为三层：即 (1)、(2) 和(3)；

Kappa架构不同于Lambda同时计算流计算和批计算并合并视图

,Kappa只会通过流计算一条的数据链路计算并产生视图。

请问该系统的大数据架构是基于哪种架构搭建的大数据平台处理奥运会大规模视频网络观看数据。



2023年案例分析第一题回忆版

【问题1】(8分)

1.两, 2.一, 3.高, 4.低, 5.只需要运行实时计算, 计算开销小;6.满足实时性,7.大,8.弱

【问题2】(9分)

1.spark;2.mapreduce;3.数据存储层;4.MemSQL;5.HDFS;6,kafka;7.flume;8.ETL;9.数据采集层

【问题3】(8分)

批处理层、加速层和服务层, lambda



T H E E N D

功不唐捐，玉汝于成！

• 开 启 新 征 程 •

