刀光剑影江湖情，摧枯拉朽浪滔滔。功名利禄拂衣去，山高水远路迢迢。

数据湖初识

近两年，为什么都开始谈论起 Data Lake 这个"新名词"了？

先说说我的想法，其实还是用户需求驱动数据服务，大家开始关注 Data Lake 的根本原因是用户需求发生了质变，过去的数据仓库模式以及相关组件没有办法满足日益进步的用户需求。

数据湖概念的诞生，源自企业面临的一些挑战，如数据应该以何种方式处理和存储。最开始，企业对种类庞杂的应用程序的管理都经历了一个比较自然的演化周期。

那么到底是什么样的需求和挑战驱动了技术的变革，从而导致了新技术的产生呢？

数据湖的定义

Wikipedia上说数据湖是一类存储数据自然/原始格式的系统或存储，通常是对象块或者文件，包括原始系统所产生的原始数据拷贝以及为了各类任务而产生的转换数据，包括来自于关系型数据库中的结构化数据（行和列）、半结构化数据（如CSV、日志、XML、JSON）、非结构化数据（如email、文档、PDF等）和二进制数据（如图像、音频、视频）。

AWS定义数据湖是一个集中式存储库，允许您以任意规模存储所有结构化和非结构化数据。

微软的定义就更加模糊了，并没有明确给出什么是Data Lake，而是取巧的将数据湖的功能作为定义，数据湖包括一切使得开发者、数据科学家、分析师能更简单的存储、处理数据的能力，这些能力使得用户可以存储任意规模、任意类型、任意产生速度的数据，并且可以跨平台、跨语言的做所有类型的分析和处理。

但是随着大数据技术的融合发展，早期的定义可能不再那么准确了，数据湖不断演变，汇集了各种技术，包括数据仓库、实时和高速数据流技术、数据挖掘、深度学习、分布式存储和其他技术。逐渐发展成为一个可以存储所有结构化和非结构化任意规模数据，并可以运行不同类型的大数据工具，对数据进行大数据处理、实时分析和机器学习等操作的统一数据管理平台。

所以说数据仓库不是曾经的那个仓库了，数据湖也不是曾经的那个"大明湖畔的夏雨荷了"，sorry应该不是那一片绿油油的湖了。

趋势

这里聊一个很重要的趋势：数据实时化。

当然这里有很多其他的趋势，比如低成本化、设计云原生化等，但总体上我还是认为数据实时化是近几年来最热门、最明显且最容易让人看到收益的一个趋势。

数据仓库过去的模式大家可能都很了解，将整个数据仓库划分为 ODS、DWD、DWS，使用 Hive 作为数据存储的介质，使用 Spark 或者 MR 来做数据清洗的计算。

这样的数据仓库设计很清晰，数据也比较容易管理，所以大家开开心心地使用这套理论和做法将近 10 年左右。

在这 10 年的时间里，主流的互联网公司在数据技术上的玩法并没有多大的改变，比如推荐需要用到的用户画像、电商里商品的标签、好友传播时用的图、金融风控数据体系，站在更高的一个角度看，我们会发现，十年前做的事情，比如用户画像表，如果你现在去做推荐服务，还是需要这个表。这样会产生一个什么现象？十年的互联网行业的人才积累、知识积累、经验积累，让我们可以更加容易地去做一些事情，比如十年前很难招聘到的懂推荐数据的人才，水平在如今也就是一个行业的平均值罢了。

既然这些事情变得更好做了，人才更多了，我们就期望在事情上做的更精致。因为从业务上讲，我去推荐短视频，让用户购买东西，这个需求是没有止境的，是可以永远做下去的。所以以前我可能是 T+1 才能知道用户喜欢什么，现在这个需求很容易就达到之后，我希望用户进来 10s 之后的行为就告诉我这个用户的喜好；以前可能做一些粗粒度的运营，比如全人群投放等，现在可能要转化思路，做更加精细化的运营，给每个用户提供个性化定制的结果。

技术演进——实时化

数据实时化没问题，但是对应到技术上是什么情况呢？是不是我们要在实时领域也搭一套类似离线数据仓库的数据体系和模式？

是的，很多公司确实是将实时数据流划分为了不同层级——也就是我们说的实时数仓，整体层级的划分思路和离线仓库类似，但是实时数据的载体就不是 Hive 或者 Hdfs 了，而是要选择更加实时的消息队列，比如 Kafka，这样就带来了很多问题，比如：

- 消息队列的存储时间有限
- 消息队列没有查询分析的功能
- 回溯效率比文件系统更差

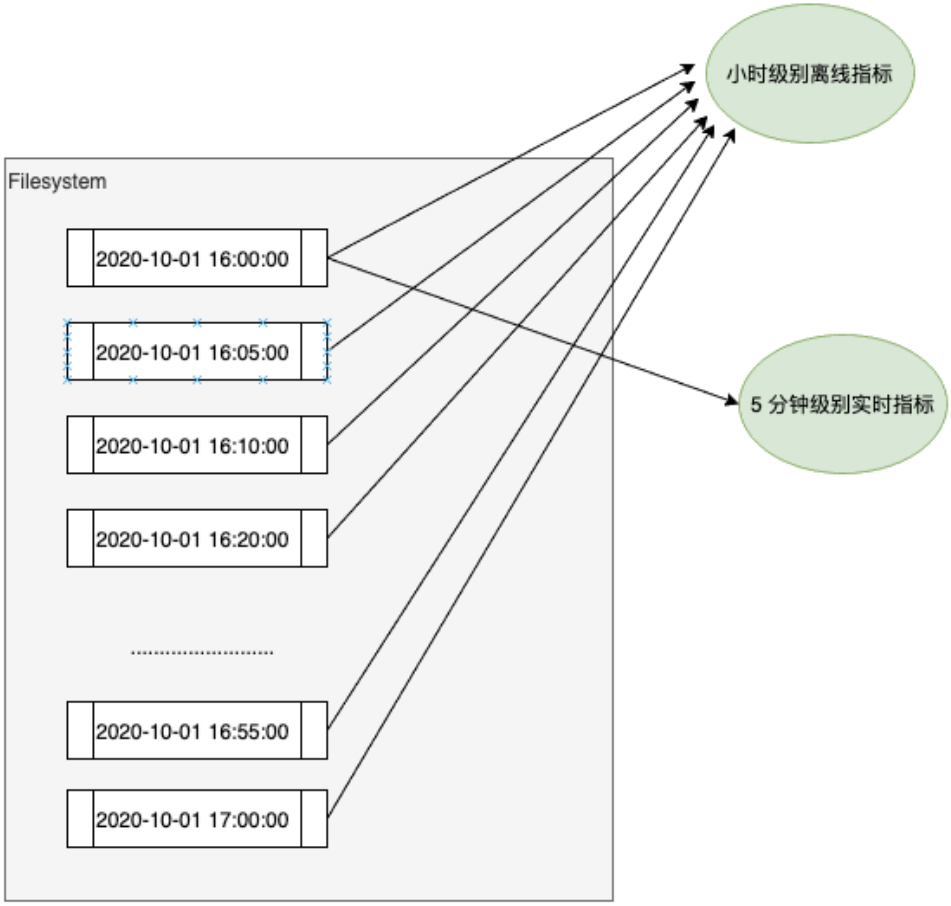
除了实时数据载体的问题，还有引入实时数仓后，和离线数仓的统一的问题，

- 比如实时数仓的数据治理、权限管理，是不是要单独做一套？
- 如何统一实时数据和离线数据的计算口径？
- 两套数据系统的资源浪费严重，成本提高？

举一个比较现实的例子，假设我们构造了一个实时计算指标，在发现计算错误后我们需要修正昨天的实时数据，这种情况下一般是另外写一个离线任务，从离线数仓中获取数据，再重新计算一遍，写入到存储里。这样的做法意味着我们在每写一个实时需求的同时，都要再写一个离线任务，这样的成本对于一个工程师是巨大的。

技术演进——降低成本化

实时系统的成本太大了，这也是让很多公司对实时需求望而生畏的原因之一。所以这样去建设实时数仓的思路肯定不行啊，等于我要招两倍的人才（可能还不止），花两倍的时间，才能做一个让我的业务可能只提升 10% 的功能。从技术的角度来看，是这两套系统的技术栈不一样造成了工程无法统一。那么，Data Lake 就是用来解决这样一个问题，**比如我一个离线任务，能不能既产生实时指标，也产生离线指标**，类似下图这样：



满足上面最重要的一个前提就是我的数据源是实时的，这样对我们的大数据存储主要就是HDFS 和 S3 又提出了新的挑战——数据实时更新，如果原有技术或者组件不能满足需求，新的技术在需求的驱动下就此诞生。

除了计算层面上，在数据管理上，比如中间表的 schema 管理，数据权限管理，能否做到统一，在架构上实现统一后，我们在应对实时需求时，可以将实时离线的冗余程度降到最低，甚至能够做到几乎没有多余成本。

这块我们也在积极探索，国内互联网公司的主流做法还是停留在 【技术演进——降低成本化】 的阶段，相信随着大家的努力，很快就会出现优秀且成功的实践。

技术演进——去结构化

Pentaho的CTO James Dixon 在2011年提出了"Data Lake"的概念。在面对大数据挑战时，他声称：不要想着数据的"仓库"概念，想想数据 的“湖”概念。数据“仓库”概念和数据湖概念的重大区别是：数据仓库中数据在进入仓库之前需要是事先归类，以便于未来的分析。这在OLAP时代很常见，但是对于离线分析却没有任何意义，不如把大量的原始数据线保存下来，而现在廉价的存储提供了这个可能。

数据仓库是高度结构化的架构，数据在转换之前是无法加载到数据仓库的，用户可以直接获得分析数据。而在数据湖中，数据直接加载到数据湖中，然后根据分析的需要再转换数据。

这里看到数据仓库这种Schema on write 已经不满足日益变化的需求了，数据湖是Schema on read，但是个人觉得与其说是Schema 还不如说是对数据的态度变了，以前我们只将对当前有用的数据抽取到数仓，而现在我们希望数据湖可以容纳所有的数据，即使当前用不到，但是当想用的时候就有数据可以用。

数据湖与数据仓库的区别

数据仓库是一种成熟稳定的技术架构。它们存储经过ETL 处理的结构化数据，以便完成整决策支持的过程。数据仓库将数据组合为一种聚合、摘要形式，以在企业范围内使用，**并在执行数据写入操作时写入元数据和模式定义。**数据仓库通常拥有固定的配置；它们是高度结构化的，因此不太灵活和敏捷。数据仓库成本与在存储前处理所有数据相关，而且大容量存储的费用相对较高。

相较而言，数据湖是较新的技术，拥有不断演变的架构。数据湖存储任何形式（包括结构化和非结构化）和任何格式（包括文本、音频、视频和图像）的原始数据。根据定义，数据湖不会接受数据治理，但专家们都认为良好的数据管理对预防数据湖转变为数据沼泽不可或缺。数据湖在数据读取期间创建模式，与数据仓库相比，数据湖缺乏结构性，而且更灵活；它们还提供了更高的敏捷性。在检索数据之前无需执行任何处理，而且数据湖特意使用了更加便宜的存储。

数据湖	数据仓库
能处理所有类型的数据，如结构化数据，非结构化数据，半结构化数据等，数据的类型依赖于数据源系统的原始数据格式。	只能处理结构化数据进行处理，而且这些数据必须与数据仓库事先定义的模型吻合。
读取的时候设计schema，存储原始原始数据	写入时设计数据仓库，存储处理后的原始数据
拥有足够强的计算能力用于处理和分析所有类型的数据，分析后的数据会被存储起来供用户使用。	处理结构化数据，将它们或者转化为多维数据，或者转换为报表，以满足后续的高级报表及数据分析需求。
数据湖通常包含更多的相关的信息，这些信息有很高概率会被访问，并且能够为企业挖掘新的运营需求。	数据仓库通常用于存储和维护长期数据，因此数据可以按需访问。

数据湖与数据仓库的差别很明显。 然而，在企业中两者的作用是互补的，**不应认为数据湖的出现是为了取代数据仓库，毕竟两者的作用是截然不同的**

1. 数据价值性：数仓中保存的都是结构化处理后的数据，而数据湖中可以保存原始数据也可以保存结构化处理后的数据，保证用户能获取到各个阶段的数据。因为数据的价值跟不同的业务和用户强相关，有可能对于A用户没有意义的数据，但是对于B用户来说意义巨大，所以都需要保存在数据湖中。
2. 数据实时性：数据湖支持对实时和高速数据流执行 ETL 功能，这有助于将来自 IoT 设备的传感器数据与其他数据源一起融合到数据湖中。形象的来看，数据湖架构保证了多个数据源的集成，**并且不限制schema，保证了数据的精确度。**数据湖可以满足实时分析的需要，同时也可以作为数据仓库满足批处理数据挖掘的需要。数据湖还为数据科学家从数据中发现更多的灵感提供了可能。
3. 数据保真性：数据湖中对于业务系统中的数据都会存储一份“一模一样”的完整拷贝。与数据仓库不同的地方在于，数据湖中必须要保存一份原始数据，无论是数据格式、数据模式、数据内容都不应该被修改。在这方面，数据湖强调的是对于业务数据“原汁原味”的保存。同时，数据湖应该能够存储任意类型/格式的数据。
4. 数据灵活性：数据湖提供灵活的，面向任务的数据绑定，不需要提前定义数据模型，"写入型schema" 和"读取型schema"，其实本质上来讲是数据schema的设计发生在哪个阶段的问题。对于任何数据应用来说，其实schema的设计都是必不可少的，即使是MongoDB等一些强调“无模式”的数据库，其最佳实践里依然建议记录尽量采用相同/相似的结构。

数据仓库强调的"写入型schema"背后隐含的逻辑是数据在写入之前，就需要根据业务的访问方式确定数据的schema，然后按照既定schema，完成数据导入，带来的好处是数据与业务的良好适配；但是这也意味着数仓的前期拥有成本会比较高，特别是当业务模式不清晰、业务还处于探索阶段时，数仓的灵活性不够。

数据湖强调的"读取型schema"背后的潜在逻辑则是认为业务的不确定性是常态：我们无法预期业务的变化，那么我们就保持一定的灵活性，将设计去延后，让整个基础设施具备使数据“按需”贴合业务的能力。因此，个人认为“保真性”和“灵活性”是一脉相承的：既然没办法预估业务的变化，那么索性保持数据最为原始的状态，一旦需要时，可以根据需求对数据进行加工处理。因此，数据湖更加适合创新型企业、业务高速变化发展的企业。同时，数据湖的用户也相应的要求更高，数据科学家、业务分析师（配合一定的可视化工具）是数据湖的目标客户。

总结

离线架构大行其道数十年，互联网数十年技术积淀和业务发展对数据又提出新要求，实时计算技术的发展满足了人们对数据实时性的要求，但未能满足互联网人对低成本高性能的执着追逐，技术的浪潮一波接一波，如果你错过了朝阳和高山，请不要错过了星辰和大海

当然，对于数据湖架构的批评也是不绝于耳。有人批评说，汇集各种杂乱的数据，应该就是数据沼泽。Martin Fowler也对数据湖中数据的安全性和私密性提出了质疑，历史见证了每一次新技术的诞生总是遇到万般挫折与质疑，但是它何曾让你失望过。