# 背景

# 概述

### 定义

数据湖是目前比较热的一个概念，许多企业都在构建或者计划构建自己的数据湖。但是在计划构建数据湖之前，搞清楚什么是数据湖，明确一个数据湖项目的基本组成，进而设计数据湖的基本架构，对于数据湖的构建至关重要。关于什么是数据湖？有不同的定义。

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。

关于数据湖的定义其实很多，但是基本上都围绕着以下几个特性展开。

1. 数据湖需要提供足够用的数据存储能力，这个存储保存了一个企业/组织中的所有数据。
2. 数据湖可以存储海量的任意类型的数据，包括结构化、半结构化和非结构化数据。
3. 数据湖中的数据是原始数据，是业务数据的完整副本。数据湖中的数据保持了他们在业务系统中原来的样子。
4. 数据湖需要具备完善的数据管理能力（完善的元数据），可以管理各类数据相关的要素，包括数据源、数据格式、连接信息、数据schema、权限管理等。
5. 数据湖需要具备多样化的分析能力，包括但不限于批处理、流式计算、交互式分析以及机器学习；同时，还需要提供一定的任务调度和管理能力。
6. 数据湖需要具备完善的数据生命周期管理能力。不光需要存储原始数据，还需要能够保存各类分析处理的中间结果，并完整的记录数据的分析处理过程，能帮助用户完整详细追溯任意一条数据的产生过程。
7. 数据湖需要具备完善的数据获取和数据发布能力。数据湖需要能支撑各种各样的数据源，并能从相关的数据源中获取全量/增量数据；然后规范存储。数据湖能将数据分析处理的结果推送到合适的存储引擎中，满足不同的应用访问需求。
8. 对于大数据的支持，包括超大规模存储以及可扩展的大规模数据处理能力。

综上，个人认为数据湖应该是一种不断演进中、可扩展的大数据存储、处理、分析的基础设施；以数据为导向，实现任意来源、任意速度、任意规模、任意类型数据的全量获取、全量存储、多模式处理与全生命周期管理；并通过与各类外部异构数据源的交互集成，支持各类企业级应用。

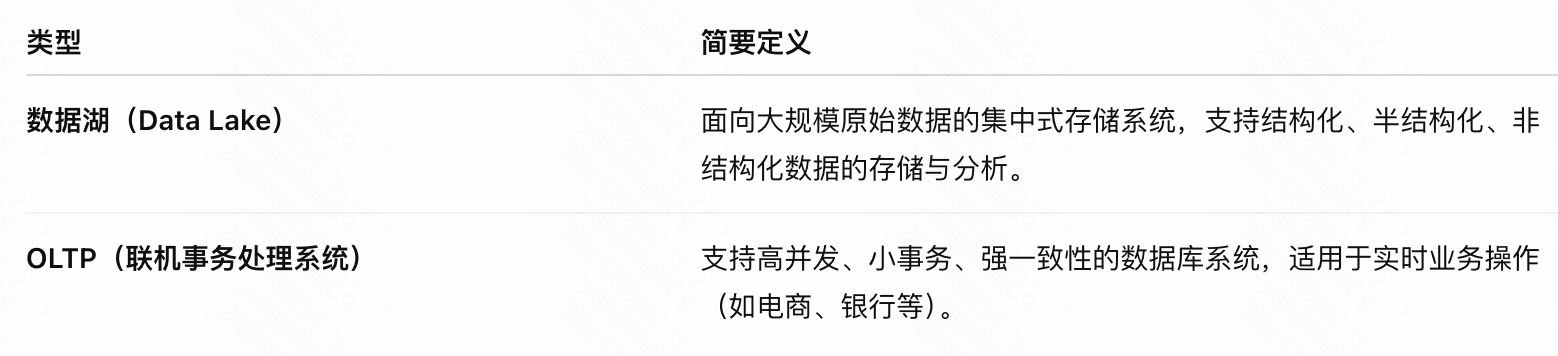
## 对比

参考：

[数据仓库、数据湖、湖仓一体，究竟有什么区别？](https://www.zhihu.com/question/423157844/answer/3080312596?share_code=19TXBVswqTXLo&utm_psn=1906063697627055809)

### 数据库/OLTP

1. 定义



2、核心区别



3、适用场景



系统设计出发点：



举例：



4、关系

不是互斥的，反而是互补的。

现代数仓/数据湖架构中，常见如下组合方式：

1）用 OLTP 系统（如 MySQL / TiDB）做实时交易业务处理

2）用 CDC 工具同步数据到 Kafka / Hudi / Iceberg 构建数据湖

3）用数据湖支撑 BI、报表、机器学习等离线/准实时分析需求

### 分布式存储/HDFS/S3

1、核心区别



2、分布式存储是数据湖的底座

数据湖是一整套生态系统，而分布式存储是其中的“地基”。你可以这样理解：

数据湖 = 数据湖引擎（Hudi/Iceberg）+ 元数据管理 + 数据目录 + 分布式存储（如 HDFS/S3）



例如：

Hudi 数据湖：

1）存储层：使用 HDFS 或 S3 存储 Parquet、Avro 文件

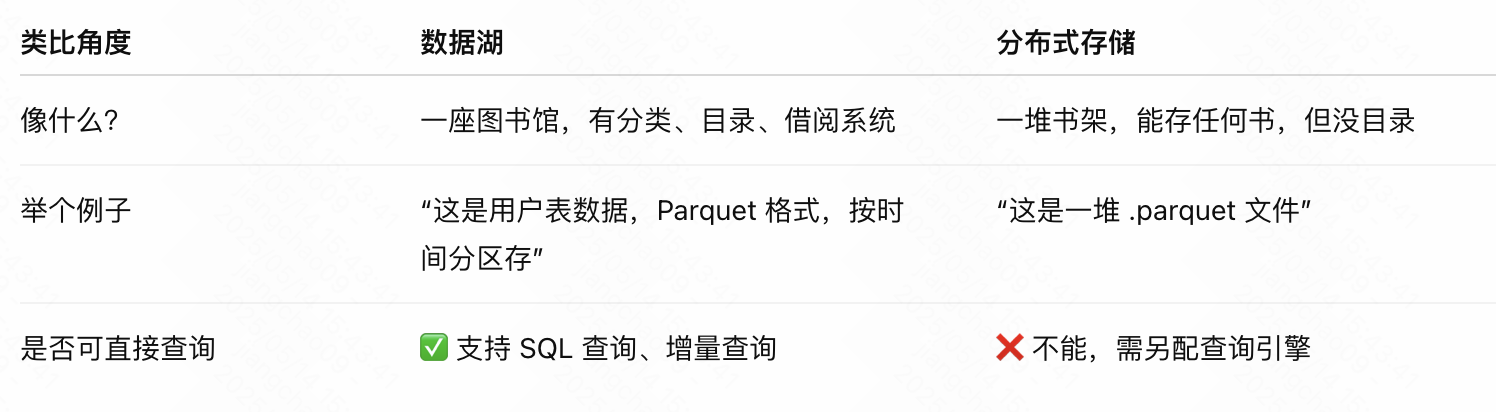
2）元数据层：.hoodie 目录记录 schema、时间线

3）计算层：Spark / Flink 写入和分析

如果没有分布式存储，数据湖就无法存储大数据文件。

如果没有数据湖引擎，HDFS 就只是文件，不知道哪些是表、哪些是数据行。

举例：



3、为什么不能把数据湖当作HDFS？

因为HDFS/S3没有：

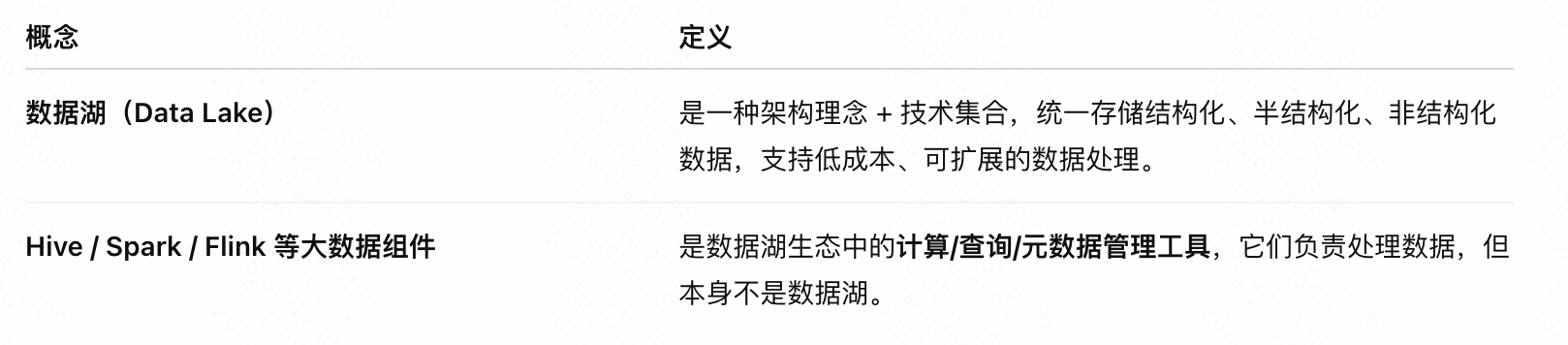


4、区别联系



### 大数据组件/Hive/Spark/Flink

1、定义



简而言之：

数据湖是一个系统架构，Hive 等是工具，是“使用者”或“构建者”之一。

2、Hive 与数据湖的区别与联系

Apache Hive 是一个基于 Hadoop 的数据仓库系统，用于在 HDFS 上运行 SQL 查询（HiveQL），本质是“将结构化 SQL 查询翻译为 MapReduce 或 Spark 作业”，让工程师用 SQL 操作大数据。

区别：



3、Hive 是数据湖的重要组件之一（不是对立）

虽然 Hive 不是“数据湖”，但它在现代数据湖架构中仍然扮演着元数据管理和查询引擎的角色。

数据湖可以使用 Hive 来：

1）作为 Catalog 元数据管理工具（Hive Metastore）

Hudi、Iceberg 等都能注册为 Hive 表，实现兼容查询

可以用 Spark SQL、Hive SQL、Trino 查询数据湖表

2）运行批量分析查询

Hive 是一个通用 SQL 执行框架，配合数据湖使用可做报表、ETL

3）支持 BI 工具对接

BI 工具可以通过 HiveServer2 连接到 Hudi / Iceberg 表

举例：Hive+Hudi架构



4、Hive 生态是“旧式数仓”，数据湖是“新式数仓”



数据湖不是为了替代 Hive，而是将 Hive 的能力提升到面向“海量数据+实时+多格式”的现代架构中。

5、总结

Hive 是传统大数据生态中的“SQL + 元数据”工具，而数据湖是一种面向未来的数据架构。两者并不冲突，Hive 可以成为数据湖的一部分，也可以用来查询和管理数据湖。

Hive 是工具，数据湖是平台

Hive 构建于 HDFS 之上，数据湖构建在 Hive / Spark / Flink 等生态之上

数据湖吸收了 Hive 的一部分理念，但更具现代化、灵活性和实时能力

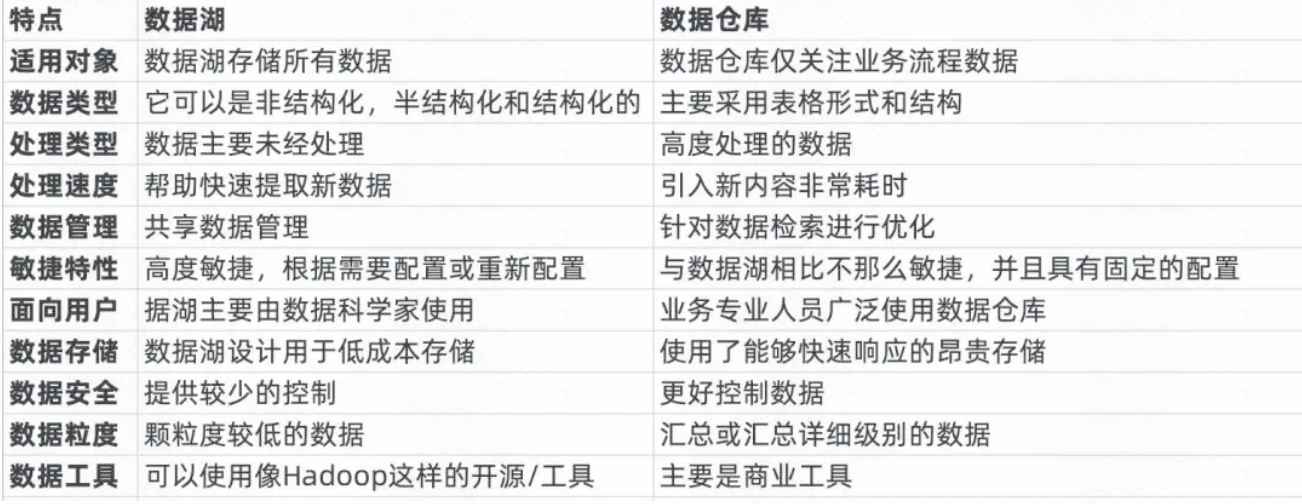
### 数据仓库

数据仓库(Data Warehouse)，也称为企业数据仓库，它是一个面向主题的、集成的、相对稳定的、反映历史变化的数据集合存储系统，它将来自不同来源的结构化数据聚合起来，用于业务智能领域的比较和分析，数据仓库是包含多种数据的存储库，并且是高度建模的。

**数据仓库 VS 数据湖**

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，但专家们一致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，并且提供了更高的敏捷性。值得一提的是，数据湖非常适合使用机器学习和深度学习来执行各种任务，比如数据挖掘和数据分析，以及提取非结构化。

区别：



**湖仓一体**

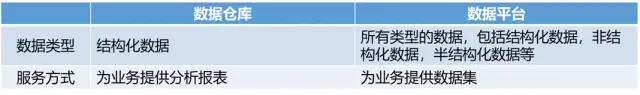
数据仓库和数据湖的结合就是湖仓一体，湖仓一体可以理解为把数据湖这个大杂间分了很多的区，每个区是一个应用站点，有的站点做BI，有的站点做大数据处理。

湖仓一体的架构，最终想要实现的，就是通过把数据湖作为中央存储库，围绕数据湖建立各种提供服务的站点，比如数据仓库，供业务分析和接入BI使用；再比如供机器学习使用的站点；供大数据处理的站点等等，最终实现随心所欲地使用数据湖中的数据。

说明：

这里的湖仓一体与我们的关系型数据库（OLTP）不同，一般是关系型数据库的数据作为接入层通过Kafka等同步到数据湖的存储介质，然后再经过加工到数据仓库。

### 数据平台

由于数据仓库具有历史性的特性，其中存储的数据大多是结构化数据；而数据平台的出现解决了数据仓库不能处理非结构化数据和报表开发周期长的问题。  


通过以上的论述，我们发现数据平台和数据湖好像存在诸多相似性，这二者之间的区别，从个人角度理解上分析应该是数据加工的角度不同，数据湖更着重于对原始数据的存储，而数据平台则同数据仓库一样，需对原始数据进行清洗、转换等数据处理后按照统一的标准规范进行存储。

### 数据中台

数据中台通过对企业内外部多源异构的数据采集、治理、建模、分析和应用，使数据对内优化管理提高业务价值，对外进行数据合作让业务价值得到释放，使之成为企业数据资产管理中枢。数据中台建立后，会形成数据API服务，为企业和客户提供高效各种数据服务。

数据中台对一个企业的数字化转型和可持续发展起着至关重要的作用。数据中台为解耦而生，企业建设数据中台的最大意义就是应用与数据之间的解藕，这样企业就可以不受限制地按需构建满足业务需求的数据应用。

构建了开放、灵活、可扩展的企业级统一数据管理和分析平台，将企业内、外部数据随需关联，打破了数据的系统界限。

利用大数据智能分析、数据可视化等技术，实现了数据共享、日常报表自动生成、快速和智能分析，满足企业各级部门之间的数据分析应用需求。

深度挖掘数据价值，助力企业数字化转型落地。实现了数据的目录、模型、标准、认责、安全、可视化、共享等管理，实现数据集中存储、处理、分类与管理，建立大数据分析工具库、算法服务库，实现报表生成自动化、数据分析敏捷化、数据挖掘可视化，实现数据质量评估、落地管理流程。

**数据仓库 VS 数据中台**

数据仓库和传统的数据平台，其出发点为一个支撑性的技术系统，即一定要先考虑我具有什么数据，然后我才能干什么，因此特别强调数据质量和元数据管理；而数据中台的第一出发点不是[数据](http://mp.weixin.qq.com/s?__biz=MzIxMTE0ODU5NQ==&mid=2650243693&idx=1&sn=b721eaccd31b1f81cff1a63075c33d69&chksm=8f5ae831b82d61275943777963e4691848985ba4743f8418a1479af7baabd9af4abd3032314f&scene=21" \l "wechat_redirect" \t "https://mp.weixin.qq.com/_blank)而是业务，一开始不用看你系统里面有什么数据，而是去解决你的业务问题需要什么样的数据服务。

在具体的技术处理环节，二者也有明显不同，数据的预处理流程正在从传统的ETL结构向ELT结构转变。传统的数据仓库集成处理架构是ETL结构，这是构建数据仓库的重要一环，即用户从数据源抽取出所需的数据，经过数据清洗，将数据加载到数据仓库中去。而大数据背景下的架构体系是ELT结构，其根据上层的应用需求，随时从数据中台中抽取想要的原始数据进行建模分析。



# 原理

数据湖（Data Lake）是一种现代化的数据架构，用于集中存储各种结构化、半结构化、非结构化的数据，并支持大规模的存储、管理、分析和处理。它主要解决的是传统数据仓库难以处理多种数据类型、扩展性差以及数据更新不灵活等问题。

数据湖的原理可以从以下几个方面来分析：

## 基本原理

1、统一存储不同类型的数据

数据湖的一个关键特性就是可以存储结构化、半结构化、非结构化数据，不受数据格式的限制。这使得数据湖成为一个可以存储各种来源数据的中央数据仓库。例如：

结构化数据：关系数据库中的表数据、日志文件中的 CSV、JSON 数据等

半结构化数据：如 JSON、XML、Parquet 文件等

非结构化数据：图像、音频、视频、文档等

2、无模式（Schema-on-Read）

与传统的关系数据库不同，数据湖通常采用无模式存储（Schema-on-Read）。这意味着数据的模式（Schema）是在数据被读取时才定义，而不是在数据写入时定义。这种方式使得数据湖能够存储任何格式的数据，无需预定义结构。

在数据写入时不需要强制执行任何格式或模式。

在分析数据时，可以根据需求对数据进行模式的定义或映射。

3、分布式存储系统

数据湖通常使用分布式存储技术（如 Hadoop HDFS、AWS S3、Azure Data Lake、Google Cloud Storage 等），能够高效地存储海量的数据。分布式存储提供了以下几个优势：

横向扩展性：随着数据量的增加，可以通过增加节点来扩展存储容量。

高可用性：分布式存储通过复制机制保证数据的高可用性，避免单点故障。

成本效益：使用分布式存储的硬件可以是低成本的 commodity 硬件，降低存储成本。

4、数据存储与计算分离

数据湖通常采用存储与计算分离的架构。例如，数据存储可以在 HDFS 或 S3 中，而计算和查询则由 Spark、Flink、Hive 等工具来执行。这种设计使得计算和存储能够独立扩展，提高了灵活性和成本效益。

5、元数据管理

数据湖依赖于强大的元数据管理系统来管理数据的目录、结构和访问权限。元数据是数据湖的核心，通常通过以下方式进行管理：

元数据层：记录数据文件的路径、格式、数据源、分区等信息。

数据目录（Catalog）：例如 Apache Hive Metastore，提供数据表、数据列、表格关系等元数据管理。

元数据管理是确保数据湖中数据能被快速检索和高效查询的关键。

## 架构

数据湖架构可以分为以下几个层级：

### 数据接入层（Ingestion Layer）

这一层负责从各种数据源获取数据并将其存储到数据湖中。数据源可以是实时数据流、数据库、日志、文件等。常见的工具包括：

Kafka：用于处理实时数据流。

Flume：用于从日志文件中采集数据。

Apache Nifi：用于数据流的自动化调度和管理。

说明：

这里就与数据库息息相关了，数据库的CDC可以通过这个链路将其中的数据同步到数据湖中。

### 存储层（Storage Layer）

存储层是数据湖的基础设施部分，它通过分布式文件系统（如 HDFS、AWS S3、Azure Blob Storage 等）存储数据。数据在这里是以原始文件的形式存储的，数据湖通常采用列式存储格式（如 Parquet、ORC），以优化查询性能。

### 数据处理层（Processing Layer）

数据处理层是数据湖的核心计算层，负责处理存储的数据。它通常依赖分布式计算框架来进行批处理、流处理、ETL（抽取、转换、加载）等操作。常见的处理工具包括：

Apache Spark：支持大规模数据处理和分析，适合批处理和流处理。

Flink：主要用于实时流处理。

Hadoop MapReduce：传统的大数据处理框架，主要用于批处理。

### 分析层（Analytics Layer）

在这一层，数据湖中的数据可以被查询、分析和可视化。数据湖能够支持通过 SQL、机器学习、BI 工具等对数据进行分析。常见的工具包括：

Presto：一个分布式 SQL 查询引擎，能够快速查询数据湖中的数据。

Apache Hive：可以将 SQL 查询扩展到大规模数据集，支持数据湖的数据查询。

Trino：原名 PrestoSQL，支持大规模数据查询。

### 访问与安全层（Access and Security Layer）

这一层负责管理数据的访问权限、审计、认证、授权等安全控制。数据湖中通常会集成多种安全工具来保护数据的安全性，包括：

Apache Ranger：提供细粒度的访问控制。

Kerberos：用于安全认证。

Sentry：数据访问控制和安全策略。

## 组件关系

Hudi、Iceberg 等数据湖引擎（Data Lake Engine）在数据湖架构中处于数据处理层和元数据管理层的关键角色，它们是实现数据湖管理、更新和查询的核心组件。它们提供了一些关键的功能，使得数据湖不仅能存储原始数据，还能支持数据的高效管理、版本控制、ACID 事务等。

为了更清晰地展示这些引擎在数据湖架构中的位置，我们可以根据以下几个方面进行分析：

### Hudi/Iceberg等的功能定位

1）数据管理和版本控制

Hudi、Iceberg 和 Delta Lake 等数据湖引擎为数据湖提供了：

ACID 事务支持：允许对数据执行原子操作，如插入、更新和删除（如 Hudi 支持 upsert 操作）。

版本控制：管理数据版本，支持时间旅行（Time Travel）查询，回溯到历史状态。

增量查询：支持增量拉取，能够高效地处理新增或变更的数据。

2）元数据管理

这些引擎通过元数据存储来管理数据的分区、表结构、时间线、版本等。它们通常将元数据存储在独立的元数据存储系统（如 Hive Metastore、Iceberg Catalog）中，以便进行高效的查询。

3）数据湖查询优化

通过将数据存储格式（如 Parquet、ORC）与表结构紧密结合，Hudi、Iceberg 等可以优化查询性能。它们通过构建索引、文件合并等手段来提升查询速度。

### Hudi/Iceberg在数据湖架构中的具体位置

在数据湖的整体架构中，Hudi、Iceberg 等引擎主要处于以下两个层级：

1）数据存储层

数据湖存储格式管理：Hudi、Iceberg 等引擎为数据提供了强大的存储格式支持（如 Parquet），并在存储文件中管理数据分区、版本和索引等。数据湖引擎决定了数据是如何存储的，包括文件的分布、压缩、格式和增量更新。

高效的增量存储：通过合并小文件、压缩文件、构建索引等手段，Hudi 和 Iceberg 确保数据存储高效且便于查询。

2）数据处理与计算层

执行计算与查询：在数据处理层，Hudi、Iceberg 等引擎与分布式计算框架（如 Spark、Flink）结合，提供数据的批处理、流处理和交互式查询。它们通过提供事务支持和增量处理能力，简化了大规模数据集的管理与计算。

SQL 查询与事务管理：Hudi、Iceberg 等通过与 SQL 查询引擎（如 Hive、Presto、Trino 等）集成，提供 SQL 查询能力。同时，它们的事务机制允许对数据执行可靠的操作（如更新和删除），从而确保数据一致性。

### 数据湖架构中各层次与 Hudi、Iceberg 的关系

1）数据接入层（Ingestion Layer）：

数据从不同的数据源（如数据库、实时流、日志、文件等）流入数据湖。数据可以通过 Kafka、Flume、NiFi 等工具进行接入。

Hudi和Iceberg可以作为数据接入和存储的目标，使用它们提供的 API 直接将数据流式或批量写入。

2）存储层（Storage Layer）：

数据湖的存储通常依赖于 HDFS、S3、Azure Data Lake 等分布式存储。

Hudi、Iceberg等提供数据存储格式（如 Parquet 或 ORC）和表管理能力，它们为分布式存储提供高效的管理功能。

存储层不仅仅是存储数据文件，它还通过引擎提供数据版本、增量更新等能力。

3）数据处理层（Processing Layer）：

Hudi和Iceberg在这一层与分布式计算框架（如 Spark、Flink）协同工作。

Hudi和Iceberg负责执行增量数据处理、数据的合并与清理，确保数据一致性和高效的查询性能。

4）分析层（Analytics Layer）：

数据湖中的数据通常通过 SQL 查询引擎（如 Presto、Hive、Trino）进行分析。

Hudi和 Iceberg提供了与这些引擎的集成支持，使得用户能够用 SQL 查询存储在数据湖中的数据，并提供 ACID 事务支持、时间旅行查询等高级功能。

5）访问与安全层（Access and Security Layer）：

数据湖需要确保数据的安全性、权限控制和访问管理。

虽然Hudi和Iceberg主要专注于数据存储与处理，但它们依赖底层的安全管理机制（如 Hadoop 安全、Kerberos、AWS IAM、Azure AD）来保证数据访问的安全性。

### 架构图示例（数据湖中的 Hudi、Iceberg 位置）



### Hudi 和 Iceberg 的作用

1）数据湖引擎（Hudi、Iceberg 等）：

存储层管理：它们定义了数据湖中数据的存储结构、版本控制、分区管理、增量更新等。

事务与增量处理：提供 ACID 事务语义，确保数据一致性，支持增量数据的高效存储和查询。

与计算引擎集成：通过与 Spark、Flink、Hive 等引擎的集成，实现对数据湖的处理和查询。

2）角色定位：

Hudi：擅长处理增量数据、支持流式数据写入，并提供高效的文件合并机制，适用于需要频繁更新和插入的场景。

Iceberg：专注于高效的大规模数据表管理，支持 ACID 事务和复杂的查询场景，适用于高吞吐量的分析型场景。

总结：

Hudi、Iceberg 等数据湖引擎为数据湖架构提供了强大的数据管理能力，帮助解决数据的版本控制、增量更新、ACID 事务等问题。它们在数据存储层和数据处理层中发挥着关键作用，通过与 Spark、Flink 等计算框架的结合，实现高效的大数据处理和查询。这使得数据湖不仅仅是一个存储平台，还能成为一个支持高效数据治理和分析的平台。

## 优缺点

* 优点

1、灵活的数据存储与处理：能够存储多种格式的数据（结构化、半结构化、非结构化），并支持多种分析需求（批量、实时）。

2、成本效益：使用廉价的硬件（commodity hardware）和分布式存储来存储海量数据，相比传统关系型数据库更具成本优势。

3、可扩展性：可以处理PB级甚至更大的数据，横向扩展能力强。

4、支持大规模数据分析：通过分布式计算平台（如 Spark、Flink），可以对海量数据进行高效处理和分析。

* 缺点：

1、数据治理复杂：数据湖中的数据往往没有统一的结构和清晰的治理标准，数据的质量控制、标注和管理会更为复杂。

2、查询性能差：对于没有经过处理的数据，查询可能会比较慢，尤其是在数据量庞大时。

3、缺乏数据一致性：数据湖一般采用最终一致性模型，适用于大量数据处理，但对于实时的事务一致性要求不高。

4、数据冗余：由于数据湖不强制数据模式，数据可能会存储很多冗余数据，导致存储管理和处理效率下降。

## 总结

数据湖的核心原理是：

集中存储多种格式的数据：包括结构化、半结构化、非结构化数据。

无模式管理：支持 Schema-on-Read，使得数据可以灵活地存储，并根据需要定义查询时的数据结构。

分布式存储：通过分布式文件系统（如 HDFS、S3等）提供扩展性、可靠性和低成本的存储能力。

大数据计算引擎：利用 Spark、Flink 等工具进行大规模数据处理。

总之，数据湖是一种非常强大的架构，它适用于需要存储海量数据并进行复杂分析的场景，如数据科学、机器学习、大数据分析等领域。

# 产品

## 开源组建

### Iceberg

### Hudi

### DeltaLake

### TiDB

## 区别



## 适用场景

* Apache Hudi 适用场景

Hudi 的优势在于其内建的 增量处理能力 和灵活的 实时写入场景支持（MOR）：



* Apache Iceberg 适用场景

Iceberg 更强调稳定性、多引擎兼容、可扩展性和并发处理能力，适合现代数据仓库 / 数据湖一体化架构：



* Delta Lake 适用场景

Delta Lake 是 Databricks 主导的项目，与 Spark 深度绑定，强调性能和事务一致性。



## 技术选型



## 总结



# 未来规划