# 概述

参考：

<https://blog.csdn.net/seteor/article/details/10532085>

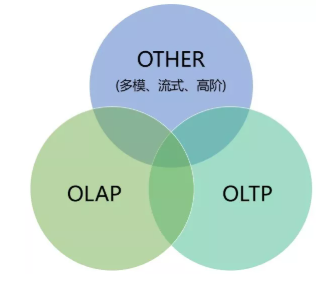
<https://www.cnblogs.com/xiaotengyi/articles/3585465.html>

<https://www.eygle.com/archives/2017/03/oracle_sharding.html>

# 分类

## 数据应用类别

根据数据的使用特征，可简单做如下划分。在选择技术平台之前，我们需要做好这样的定位。



### OLTP

联机事务处理OLTP（On-Line Transaction Processing），OLTP是事件驱动、面向应用的，也称为面向交易的处理过程。其基本特征是前台接收的用户数据可以立即传送到计算中心进行处理，并在很短的时间内给出处理结果，是对用户操作的快速响应。例如银行类、电子商务类的交易系统就是典型的OLTP系统。其具备以下特点：

直接面向应用，数据在系统中产生。

基于交易的处理系统。

每次交易牵涉的数据量很小；对响应时间要求非常高。

用户数量非常庞大，其用户是操作人员，并发度很高。

数据库的各种操作主要基于索引进行。

以SQL作为交互载体。

总体数据量相对较小。

### OLAP

联机实时分析OLAP（On-Line Analytical Processing），OLAP是**面向数据分析**的，也称为面向信息分析处理过程。它使分析人员能够迅速、一致、交互地从各个方面观察信息，以达到深入理解数据的目的。其特征是应对海量数据，支持复杂的分析操作，侧重决策支持，并且提供直观易懂的查询结果。例如数据仓库是其典型的OLAP系统。其具备以下特点：

本身不产生数据，其基础数据来源于生产系统中的操作数据

基于查询的分析系统；复杂查询经常使用多表联结、全表扫描等，牵涉的数量往往十分庞大

每次查询设计的数据量很大，响应时间与具体查询有很大关系

用户数量相对较小，其用户主要是业务人员与管理人员

由于业务问题不固定，数据库的各种操作不能完全基于索引进行

以SQL为主要载体，也支持语言类交互

总体数据量相对较大

OLTP VS OLAP



### OTHER

除了传统的OLTP、OLAP类，近些年来针对数据的使用又有些新特点，我将其归入了“其他”类。

#### 多模

随着业务“互联网化”和“智能化”的发展以及架构 “微服务”和“云化”的发展，应用系统对数据的存储管理提出了新的标准和要求，数据的多样性成为突出的问题。早期数据库主要面对结构化数据的处理场景。后面随着业务的发展，逐渐产生了对非结构化数据的处理需求。包括结构化数据、半结构化(JSON、XML等)数据、文本数据、地理空间数据、图数据、音视频数据等。多模，正是指单一数据库支持多种类型数据的存储与处理。

#### 流式

流式处理(实时计算)，是来源于对数据加工时效性的需求。数据的业务价值随着时间的流失而迅速降低，因此在数据发生后必须尽快对其进行计算和处理。传统基于周期类的处理方式，显然无法满足需求。随着移动互联网、物联网和传感器的发展导致大量的流式数据产生。相应地出现了专有的流式数据处理平台，如Storm、Kafka等。近些年来，很多数据库开始支持流式数据处理，例如MemSQL、PipelineDB。有些专有流式数据处理平台开始提供SQL接口，例如KSQL基于Kafka提供了流式SQL处理引擎。

#### 高阶

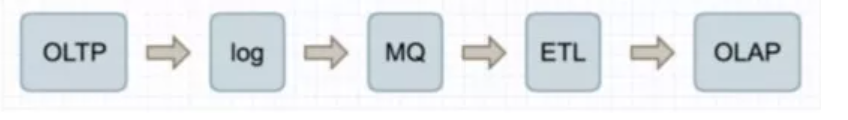
随着对数据使用的深入，数据的使用不再仅仅以简单的增删改查或分组聚合类操作，而对于其更为高阶的使用也逐步引起大家的重视。例如使用机器学习、统计分析和模式识别等算法，对数据进行分析等。

## 数据处理模式

面对上述复杂多变的应用场景，数据应用的多种类别，是由单一平台处理，还是由不同平台来处理呢？一般来说，专有系统的性能将比通用系统性能高一到两个数量级，因而不同的业务应采用不同的系统。但正如古人说“天下大势、分久必合、合久必分”，在数据处理领域也有一种趋势，由单一平台来处理。这里选择的核心在于如何来辩证看待需求和技术。它们是一对矛盾体，当这对矛盾缓和时，数据处理领域将更趋向于整合；而当这对矛盾尖锐时，数据处理领域将趋于分散。就软硬件技术发展现状和当前需求来看，未来整合的趋势更为明显。集成数据平台将能满足绝大多数用户的场景，只有极少数企业需要使用专有系统来实现其特殊的需求。

### 分散式（专有平台）

目前比较常规的方式，是采用多个专有平台，来针对不同场景进行数据处理。因此是跨平台的，因此是有个数据传输的过程。这之中会带来两个问题：数据同步、数据冗余。数据同步的核心是数据时效性问题，过期的数据往往会丧失价值。常见的做法如下：



OLTP系统中的数据变化，通过日志的形式暴露出来；通过消息队列解耦传输；后端的ETL消费拉取，将数据同步到OLAP中。整个链条较长，对于时效性要求较高的场景是个考验。此外，数据在链条中流动，是存在多份的数据冗余保存。在常规的高可用环境下，数据会进一步保存多份。因此这里面隐藏了比较大的技术、人力成本以及数据同步成本。而且横跨如此之多的技术栈、数据库产品，每个技术栈背后又需要单独的团队支持和维护，如DBA、大数据、基础架构等。这些都蕴含着巨大的人力、技术、时间、运维成本。正是出于在满足各种业务需求的同时，提高时效性，减低数据冗余、缩短链条等，收敛技术栈就变得很重要。这也是通用类平台解决方案，诞生的出发点。

### 集中式/HTAP（通用平台）

用户厌倦了为不同的数据处理采用不同的数据处理系统，更倾向于采用集成数据处理平台来处理企业的各种数据类型。对于融合了联机事务处理和联机实时分析的场景，也就是HTAP。

在互联网浪潮出现之前，企业的数据量普遍不大，特别是核心的业务数据，通常一个单机的数据库就可以保存。那时候的存储并不需要复杂的架构，所有的线上请求 (OLTP, Online Transactional Processing) 和后台分析 (OLAP, Online Analytical Processing) 都跑在同一个数据库实例上。

随着互联网的发展，企业的业务数据量不断增多，单机数据库的容量限制制约了其在海量数据场景下的使用。因此在实际应用中，为了面对各种需求，OLTP、OLAP 在技术上分道扬镳，在很多企业架构中，这两类任务处理由不同团队完成。当物联网大数据应用不断深入，具有海量的传感器数据要求实时更新和查询，对数据库的性能要求也越来越高，此时，新的问题随之出现：

1、OLAP 和 OLTP 系统间通常会有几分钟甚至几小时的时延，OLAP 数据库和 OLTP 数据库之间的一致性无法保证，难以满足对分析的实时性要求很高的业务场景。

2、企业需要维护不同的数据库以便支持两类不同的任务，管理和维护成本高。

因此，能够统一支持事务处理和工作负载分析的数据库成为众多企业的需求。在此背景下，由Gartner提出的HTAP（混合事务/分析处理，Hybrid Transactional/Analytical Processing）成为希望。基于创新的计算存储框架，HTAP数据库能够在一份数据上同时支撑业务系统运行和OLAP场景，避免在传统架构中，在线与离线数据库之间大量的数据交互。此外，HTAP基于分布式架构，支持弹性扩容，可按需扩展吞吐或存储，轻松应对高并发、海量数据场景。

此类通用平台方案具备下面优点：

通过数据整合避免信息孤岛，便于共享和统一数据管理。

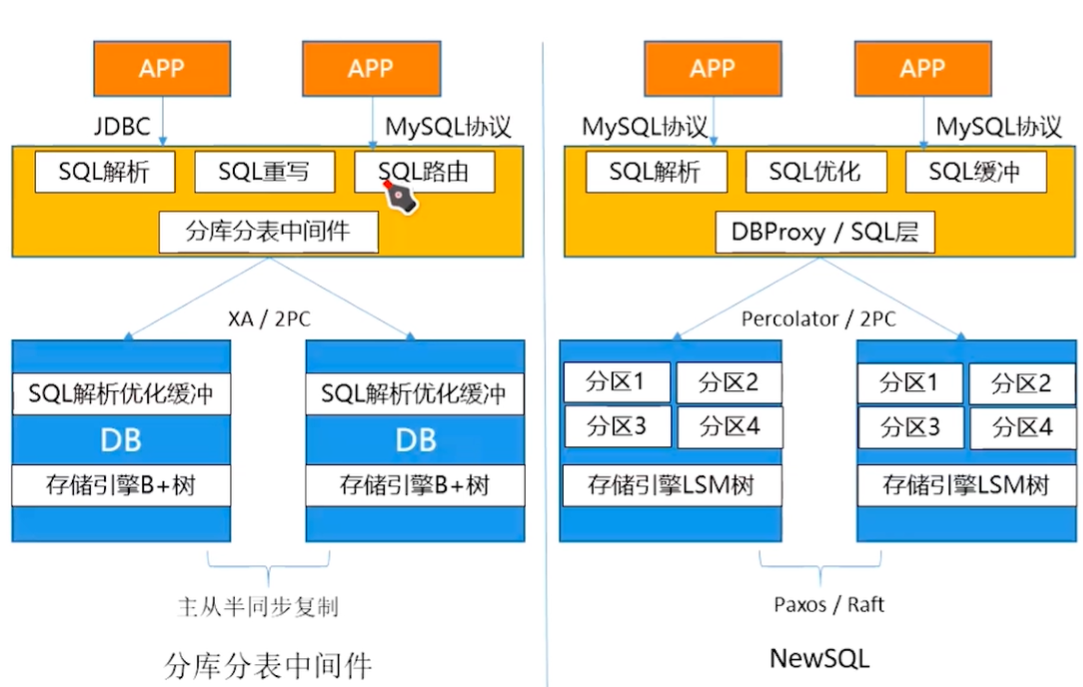
基于SQL的数据集成平台可提供良好的数据独立性，使应用能专注于业务逻辑，不用关心数据的底层操作细节。

集成数据平台能提供更好的实时性和更全的数据，为业务提供更快更准的分析和决策。

能够避免各种系统之间的胶合，企业总体技术架构简单，不需要复杂的数据导入/导出等，易于管理和维护。

便于人才培养和知识共享，无须为各种专有系统培养开发、运维和管理人才。

## 分片方式



### 分库分表中间件

中间件模式SQL解析、执行计划优化等在中间件与数据库中重复工作，效率相对比较低。

### NewSQL

#### 背景

NewSQL数据库的分布式事务相比较于XA进行了优化，性能更高；新架构NewSQL数据库存储设计即为基于Paxos（或Raft）协议的多副本，相比较于传统数据库主从模式（半同步转异步后也存在丢数问题），实现了真正的高可用、高可靠（RTO<30s，RTO=0）。

NewSQL数据库天生支持数据分片，数据的迁移、扩容都是自动化的，大大减轻了DBA的工作，同时对应用透明，无需在SQL指定分库分表键。

这些是NewSQL数据库产品主要宣传的优点，两种架构成熟度都低于传统关系型数据库，SQL功能支持以及事务一致性、可靠性等都有待提高。

#### 概述

## 共享介质

### 共享存储(主备)

### 共享存储/clusting(集群)

Share-memory：多个CPU共享同一片内存，CPU之间通过内部通讯机制进行通讯

Share-disk：每一个CPU使用自己的私有内存区域，通过内部通讯机制直接访问所有磁盘系统

Share memory体系结构的CPU之间通过主存进行通讯，具有很高的效率。但当更多的CPU被添加到主机上时，内存竞争contetion就成为瓶颈，CPU越多，瓶颈越厉害。Share-disk也存在同样的问题，因为磁盘系统由Interconnection Network连接在一起。

Share-memory和share-disk的基本问题是interference：当添加更多的CPU，系统反而减慢，因为增加了对内存访问（memory access）和网络带宽(network bandwidth)的竞争。这样shared noting体系得到了广泛的推广。

### Share-nothing

Share-noting：每一个CPU都有私有共享区域和私有磁盘空间，而且2个CPU不能访问相同磁盘空间，CPU之间的通讯通过网络连接。

Shared nothing体系是数据库稳定增长，当随着事务数量不断增加，增加额外的CPU和主存就可以保证每个事务处理时间不变。

总的来说，share nothing降低了竞争资源的等待时间，从而提高了性能。反过来，如果一个数据库应用系统要获得良好的可拓展的性能，它从设计和优化上就要考虑shared nothing体系结构。Share nothing neans few connection，它在oracle数据库设计和优化上有很多相同之处。

Share nothing对数据库应用主要体现在多用户并行访问系统时候，优化数据库操作的response time上。如果数据库操作能够顺利获得所需要的资源，不发生等待事件，自然可以减少response time，同时也体现在操作尽量少占用资源上，避免浪费时间在无用功上。

### Share-nothing certificate-based

# 应用