# 分库分表中间件技术选型

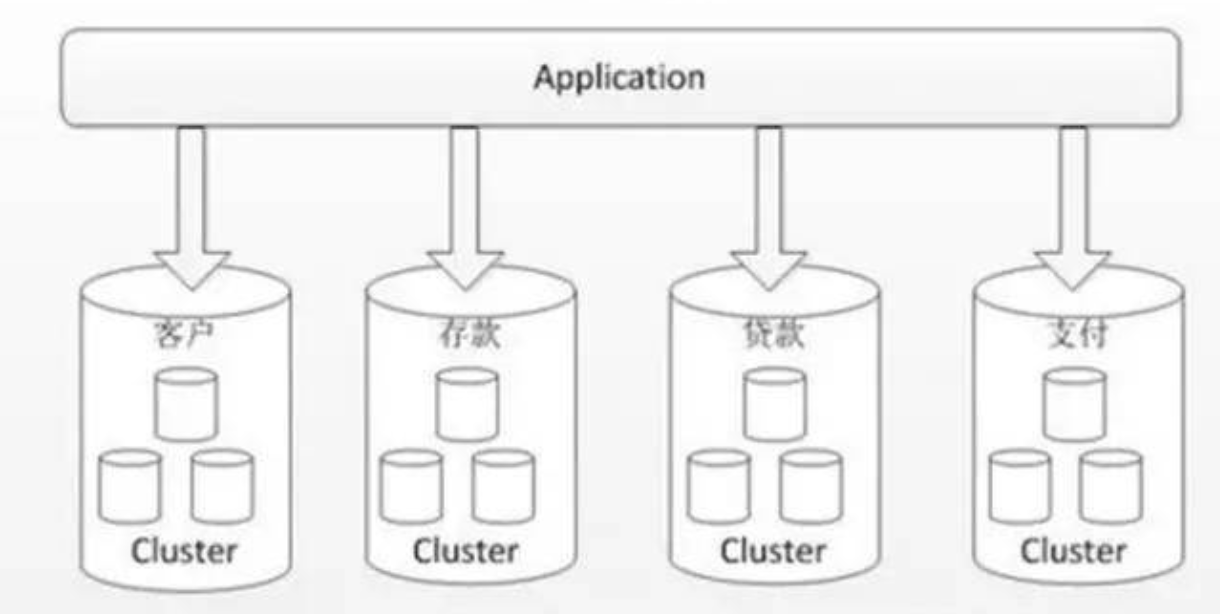
# 分库分表技术介绍

关系型数据库本身比较容易成为系统瓶颈，单机存储容量、连接数、处理能力都有限。当单表的数据量达到 1000W 或 100G 以后，由于查询维度较多，即使添加从库、优化索引，做很多操作时性能仍下降严重。此时就要考虑对其进行切分了，切分的目的就在于减少数据库的负担，缩短查询时间。

数据库分布式核心内容就是数据切分 (Sharding)，以及切分后对数据的定位、整合。数据切分就是将数据分散存储到多个数据库中，使得单一数据库中的数据量变小，通过扩充主机的数量缓解单一数据库的性能问题，从而达到提升数据库操作性能的目的。

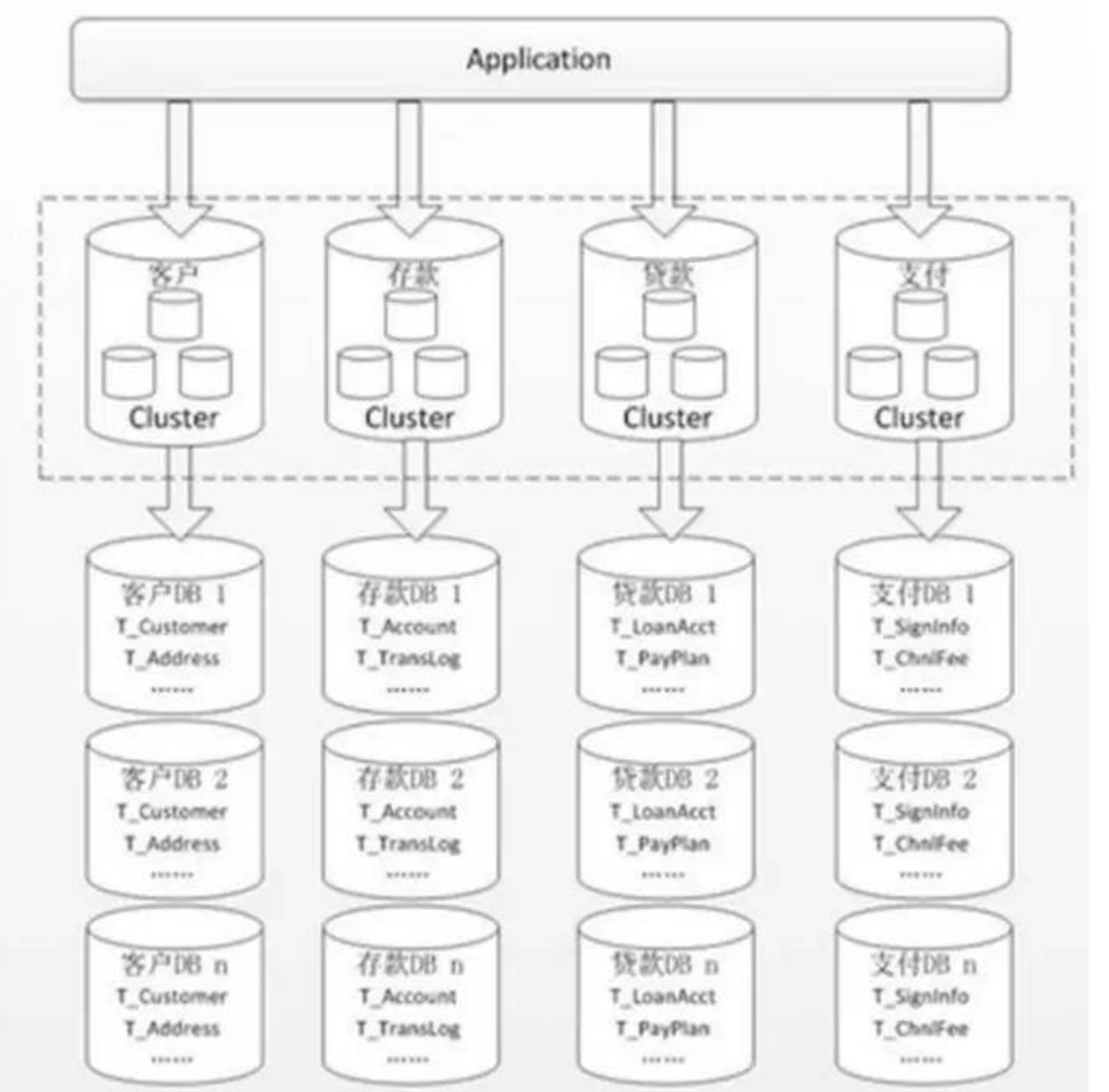
数据切分根据其切分类型一般分为垂直(纵向)切分和水平(横向)切分。

1. 纵向分片



垂直拆分是基于数据库中的“列”，某个表中的字段较多，可以新建一张表，将不经常用的或较大的字段拆分到扩展表中。在字段很多的情况下，通过大表拆小表的可在一定程度上提升数据库性能。

1. 横向分片



当一个应用难以在细粒度的垂直拆分，或者切分后数据量巨大，存在单库读写、存储性能瓶颈，这个时候就需要进行水平切分了。

水平切分分为库内分表和分库分表，是根据表内数据内在的逻辑关系，将同一个表按不同的条件分散到多个数据库或多个表中，每个表中只包含一部分数据，从而使得单个表以及单个数据库的数据量变小，达到分布式的效果。

# 分库分表中间件技术

1. MyCat

Mycat2是Mycat社区开发的一款分布式关系型数据库(中间件)。它支持分布式SQL查询，兼容MySQL通信协议，以Java生态支持多种后端数据库，通过数据分片提高数据查询处理能力。

**功能特点:**

优化查询计划:定制Calcite分布式查询引擎、编译SQL到关系代数表达式、规则优化引擎和代价优化引擎、生成物理执行计划、支持逻辑视图。

SQL支持语法:任意跨库跨表join查询、支持跨库跨表非关联子查询、支持跨库跨表关联子查询、支持跨库跨表Window语法、支持全局二级索引、有限支持存储过程、支持可视化配置。

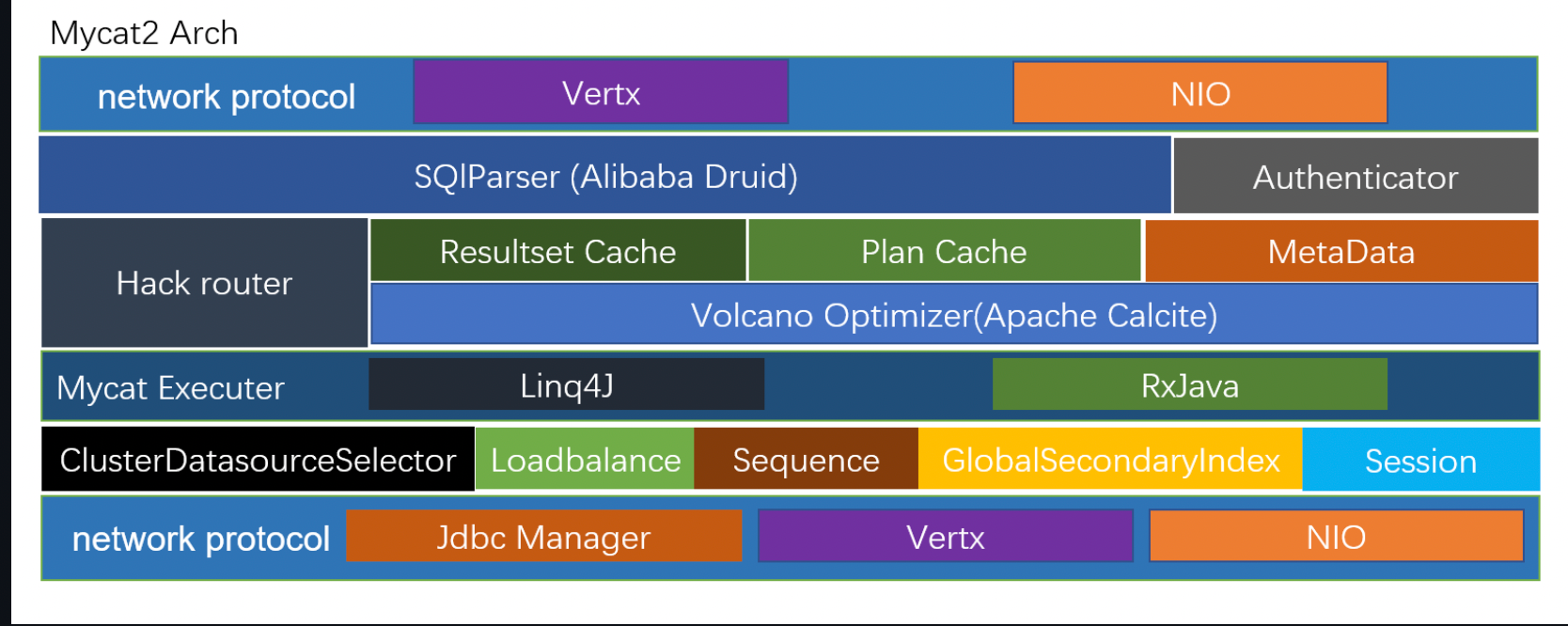
高性能：支持并行拉取结果集、支持自动调动后端结果集、支持多种路由注释、优化器注释

参数优化：对请求的sql进行参数化、缓存物理执行计划、相同参数化sql的请求、将免去一些分析优化过程

支持云原生协议：前端协议MySQL网络通信协议、MySQL原生网络协议异步非阻塞、JDBC接口支持多种数据库、生成物理执行计划。

支持定制任意多字段路由：提供分片算法接口、优化器简化过滤条件、分片信息与关系表达式结合生成执行sql。

**架构视图**：



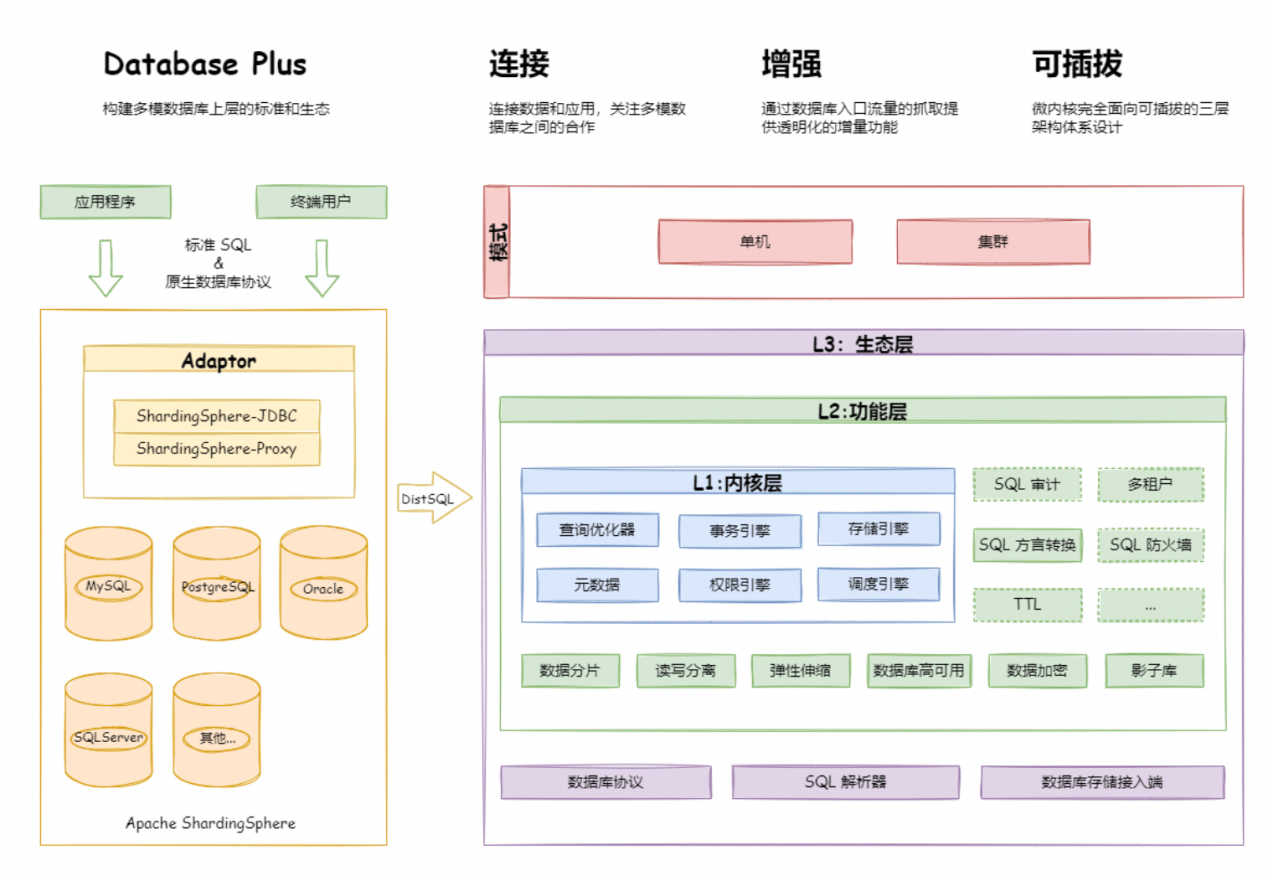
1. Apache ShardingSphere

Apache ShardingSphere 是一款分布式的数据库生态系统， 可以将任意数据库转换为分布式数据库，并通过数据分片、弹性伸缩、加密等能力对原有数据库进行增强。

Apache ShardingSphere 设计哲学为 Database Plus，旨在构建异构数据库上层的标准和生态。 它关注如何充分合理地利用数据库的计算和存储能力，而并非实现一个全新的数据库。 它站在数据库的上层视角，关注它们之间的协作多于数据库自身。

Apache ShardingSphere 由 ShardingSphere-JDBC 和 ShardingSphere-Proxy 这 2 款既能够独立部署，又支持混合部署配合使用的产品组成。 它们均提供标准化的基于数据库作为存储节点的增量功能，可适用于如 Java 同构、异构语言、云原生等各种多样化的应用场景。

**生态架构**:



**产品功能**:

数据分片: 数据分片，是应对海量数据存储与计算的有效手段。ShardingSphere 基于底层数据库提供分布式数据库解决方案，可以水平扩展计算和存储。

分布式事务:事务能力，是保障数据库完整、安全的关键技术，也是数据库的核心技术。基于 XA 和 BASE 的混合事务引擎，ShardingSphere 提供在独立数据库上的分布式事务功能，保证跨数据源的数据安全。

读写分离:读写分离，是应对高压力业务访问的手段。基于对 SQL 语义理解及对底层数据库拓扑感知能力，ShardingSphere 提供灵活的读写流量拆分和读流量负载均衡。

数据迁移:数据迁移，是打通数据生态的关键能力。ShardingSphere 提供跨数据源的数据迁移能力，并可支持重分片扩展。

联邦查询:联邦查询，是面对复杂数据环境下利用数据的有效手段。ShardingSphere 提供跨数据源的复杂查询分析能力，实现跨源的数据关联与聚合。

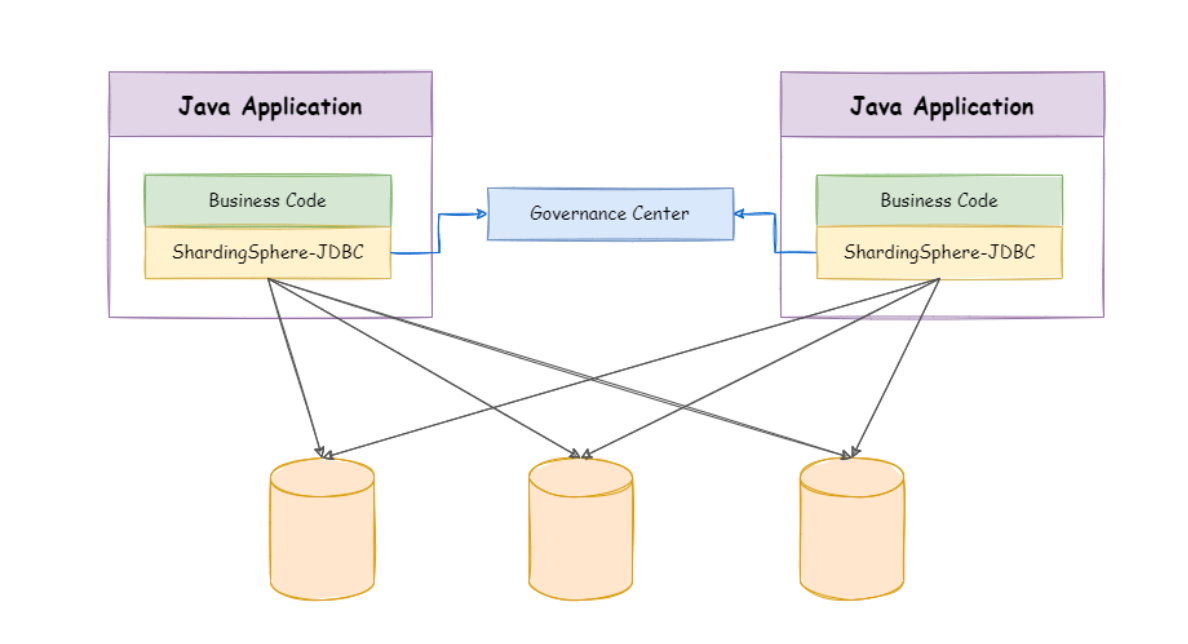
数据加密:数据加密，是保证数据安全的基本手段。ShardingSphere 提供完整、透明、安全、低成本的数据加密解决方案。

影子库:在全链路压测场景下，ShardingSphere 支持不同工作负载下的数据隔离，避免测试数据污染生产环境。

**部署形态:**

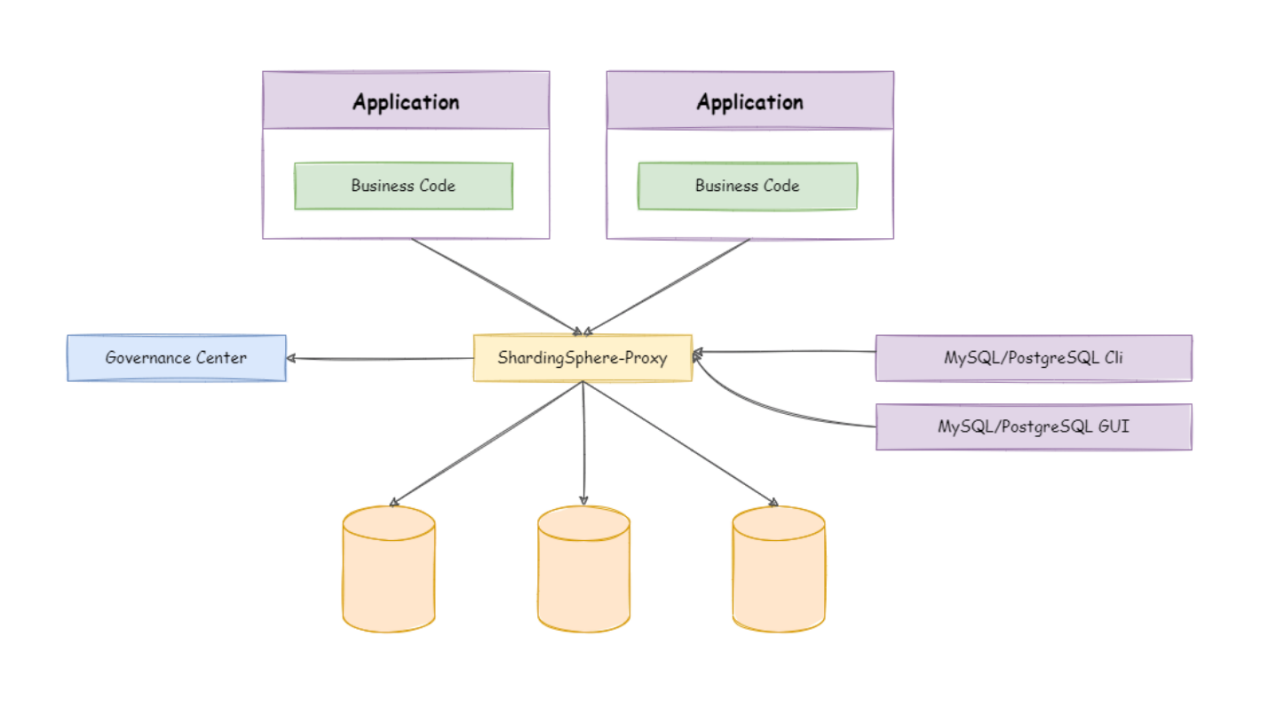
ShardingShpere-jdbc独立部署:

ShardingSphere-JDBC 定位为轻量级 Java 框架，在 Java 的 JDBC 层提供的额外服务。 它使用客户端直连数据库，以 jar 包形式提供服务，无需额外部署和依赖，可理解为增强版的 JDBC 驱动，完全兼容 JDBC 和各种 ORM 框架。



ShardingShpere-proxy独立部署：

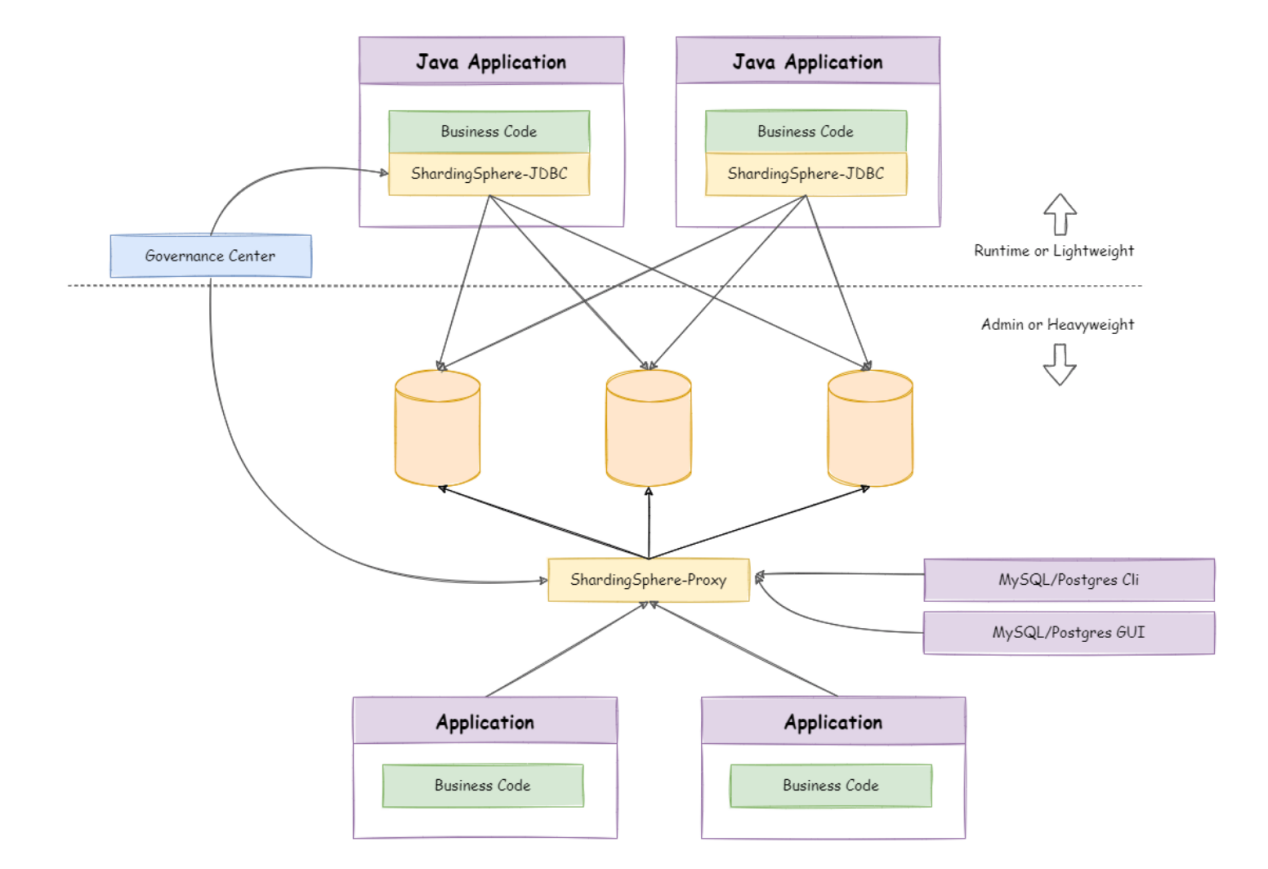
ShardingSphere-Proxy 定位为透明化的数据库代理端，通过实现数据库二进制协议，对异构语言提供支持。 目前提供 MySQL 和 PostgreSQL 协议，透明化数据库操作，对 DBA 更加友好。



混合部署：

ShardingSphere-JDBC 采用无中心化架构，与应用程序共享资源，适用于 Java 开发的高性能的轻量级 OLTP 应用； ShardingSphere-Proxy 提供静态入口以及异构语言的支持，独立于应用程序部署，适用于 OLAP 应用以及对分片数据库进行管理和运维的场景。

Apache ShardingSphere 是多接入端共同组成的生态圈。 通过混合使用 ShardingSphere-JDBC 和 ShardingSphere-Proxy，并采用同一注册中心统一配置分片策略，能够灵活的搭建适用于各种场景的应用系统，使得架构师更加自由地调整适合于当前业务的最佳系统架构。



# 分库分表中间件对比

|  |  |  |
| --- | --- | --- |
|  | MyCat2 | ShardingShpere |
| 数据库支持 | 仅支持Mysql | 使用ShrdingShpere-jdbc支持任意数据库,使用ShardingShpere-Proxy支持Mysql和PostgreSQL。 |
| 异构语言 | 任意 | 使用ShardingShpere-jdbc仅支持java，使用ShardingShpere-proxy支持任意语言 |
| 运行模式 | 单机、集群 | 单机、集群 |
| 部署模式 | 仅支持proxy模式部署 | 支持jdbc部署模式和proxy模式 |
| 上手难度 | 较困难 | 较简单 |
| 绑定表 | 不支持 | 支持 |
| 广播表 | 不支持 | 支持 |
| 自定义分片算法 | 支持 | 支持 |
| 聚合查询 | 支持 | 支持 |
| 影子库 | 不支持 | 支持 |
| 数据迁移 | 不支持 | 支持 |
| 数据加密 | 不支持 | 支持 |
| 数据脱敏 | 不支持 | 支持 |
| 读写分离 | 支持 | 支持 |
| 可观测性 | 较弱 | 支持较好 |
| 事务支持 | XA事务 | XA事务 |
| 功能商业化 | 无，所有功能全部开源提供 | 可具有商业化定制功能 |
| 社区活跃 | 几乎已经死掉，版本难产，文档较少 | 非常活跃，版本一直保持迭代，提供全量的官方文档支持 |

# 选型建议

1. 从功能方面来说ShardingShpere相较于Mycat来说强大很多，而且版本一直保持迭代，一直在对已有功能进行改进升级、并增加新的功能。
2. 从使用方面来说ShardingShpere可以根据自己产品选择jdbc或者proxy模式甚至或者部署，而Mycay只支持proxy模式不是，并且上手难度较高。
3. 从技术支持方面来说ShardingShpere保持较高的活跃度，而且有专门的微信等渠道，并且可以提供商业务版本支持。

建议选择ShardingShpere产品。