微店分布式数据库平台

中间件：赵继远

提纲

- 数据中间件概述
- VDDS
- VDDS Proxy
- Q&A

1 概述—领域问题

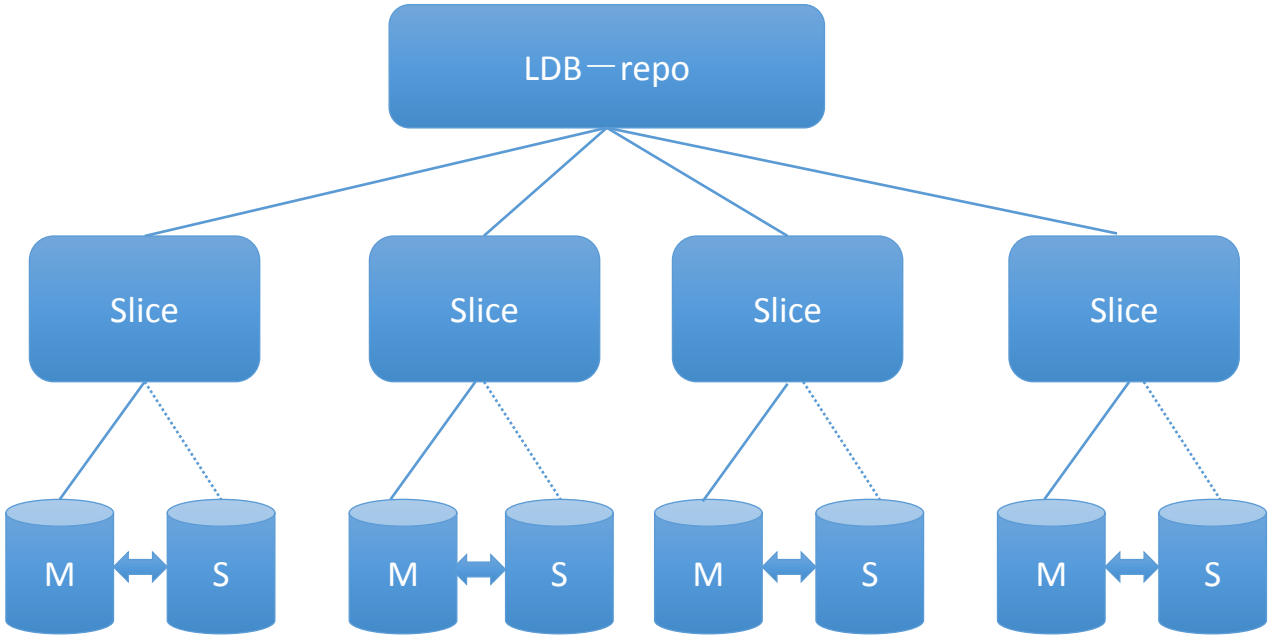
- DB扩展性（读写分离&分库分表）
- 高可用性（监测和主备切换）
- 数据运维（库表变更，配置变更，扩容）

1 概述—client/proxy

Client	
• 维护简单 • 性能好 • 稳定性好	• 不好解决多语言问题
• 解决多语言问题 • 还能提供链接复用，白名单和sql防火墙等额外功能	• 工作量更大，需要网络模块和mysql协议模块 • 独立server运维成本高 • 稳定性要求高，需要考虑down的问题
Proxy	

1 概述—概念

概念	解释
LogicDB（LDB）	逻辑数据库
LogicTable	逻辑表
PhysicalDB	物理数据库
PhysicalTable	物理表
MySQL instance	MySQL 实例，ip＋port
Repo	逻辑数据源配置
Slice	一个高可用组
Rule	分库分表规则
Sequence	唯一序列生成器


LDB—repo
Slice
Slice
Slice
Slice
M
S
M
S
M
S
M
S

2 功能—核心特性

- 实现jdbc接口，可以很好的和orm框架等配合使用。接口标准，易用性很好。

2 功能—核心特性

- 动态配置

  基于配置中心（比如zk）

  以逻辑数据库为单位

  配置变更无需重启应用（修改数据库连接属性，主备库变更等）

- manager

  vdds的配置管理中心，管理vdds元数据。

  对外提供http＋json的接口，可以查询和修改配置

2 功能—核心特性

读写分离

- 动态配置的读写分离策略，对应用透明。

- Slice hint

/*+VDDS({'type':'selectdb','isMaster':true})*/

用来显式制定读写分离策略。只能用在select语句上。

例子如下：

/*+VDDS({'type':'selectdb','isMaster':true})*/ select * from book where id in(5,8,16) or name like '%mysql%’ 表示这个select语句要强制到主库执行。

2 功能—核心特性

- 分库分表（基于动态配置，灵活，扩展性好）

支持数值型，字符型和日期类型切分，支持多字段切分。后续还会支持应对热点数据等。

支持多库多表，既分库又分表。

支持hint的方式处理特殊case。

2 功能—核心特性

Route Hint——提供了绕过rule的配置，直接指定sql路由信息的方式。

```
/*+VDDS({'type':'direct','dbSliceID':'xxx_group','vtab':'logic','realtabs':['rtab_0','rtab_1']})*/
```

type 固定为direct

dbSliceID 为用户指定要执行sql的分片id

vtab 为逻辑表名

realtabs 为实际要执行的物理表。

2 功能—核心特性

Route Hint——例子如下：

/*+VDDS({'type':'direct','dbSliceID':'group0','vtab':'book','realtabs':['book0','book1']})*/ select * from book where id in(5,8,16) and name like '%mysql%'

SELECT * FROM book0 WHERE id IN (5, 8, 16) AND name LIKE '%mysql%'

SELECT * FROM book1 WHERE id IN (5, 8, 16) AND name LIKE '%mysql%’

2 功能—核心特性

全局唯一序列生成器sequence

- 为什么需要sequence

  在分库分表场景下，mysql本身的自增主键无法保证全局唯一。

  vdds的sequence就是为了解决这个问题而生的。

- 怎么使用

  a. vdds的配置里先做好相应的设置（配置好sequence和主键）

  b. insert 时就像使用自增主键一样。（注意：如果insert语句包含主键，sequence就不会生成了）

2 功能—核心特性

批量插入—导数据

- 当前支持insert 多values的形式或JDBC的addbatch方式。
- 目前正在开发load data的方式

2 功能—核心特性

- OLTP支持好。OLTP类应用基本都能支持。目前OLAP类的支持还不足（复杂查询支持的还不好，后面会做）。
- 和dba的运维系统配合解决数据库高可用问题。
- 可以做到应用不停机做扩容。简化业务方扩容的烦恼。（目前是我们半自动化做，后面会往全自动化方向走）

2 功能—使用限制

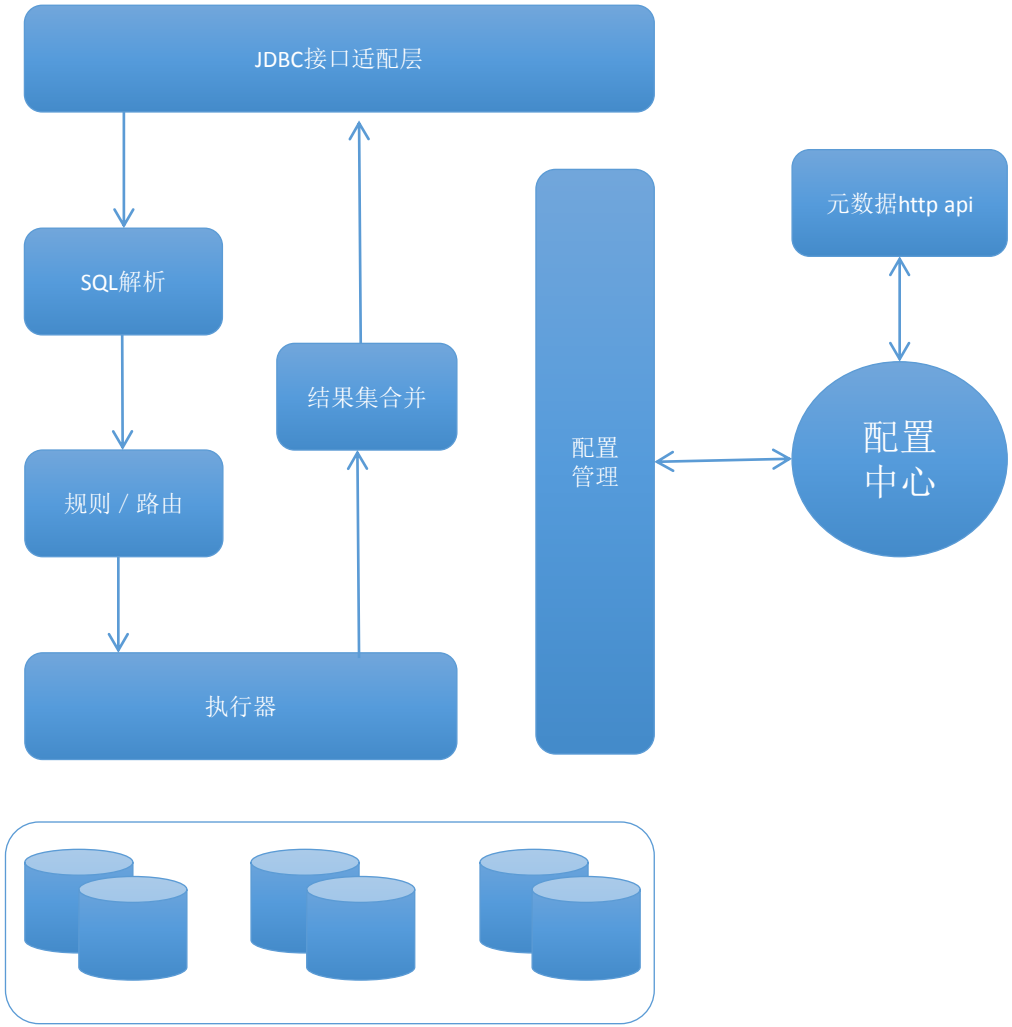
- 分布式事务

  vdds 本身不支持分布式事务，只支持单库事务。

- 复杂SQL

  查询涉及多个分表的，目前vdds只支持简单merge，不支持二次计算（分布式join，自查询）。

2 VDDS 内部实现



2 实现一jdbc接口&sql解析

接口适配层：

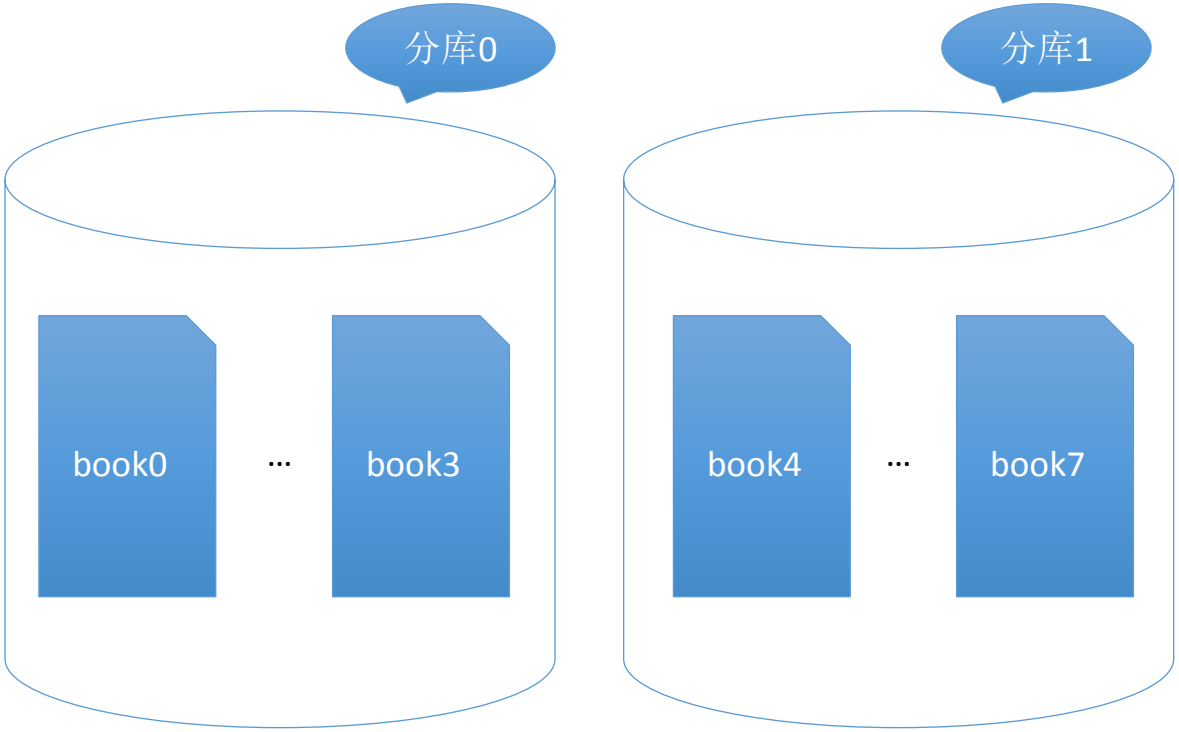
- 实现jdbc DataSource，connection，statement，resultset等接口。
- 通用性好，可以和各种orm配合使用

SQL解析：

- 复用cobar的sql parser
- 根据我们的项目需要做了修改

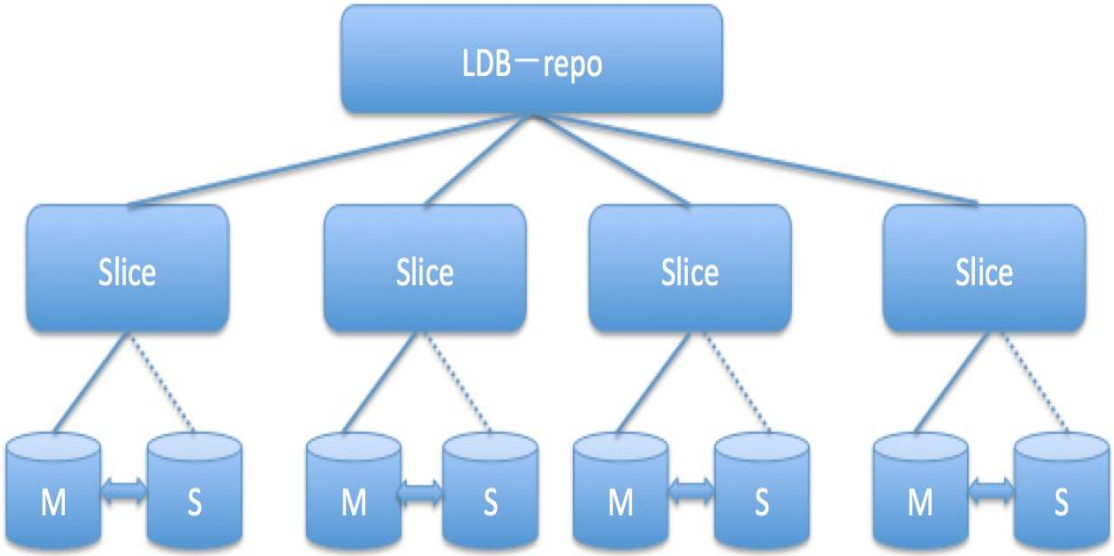
2 实现—rule

Rule决定了数据的分布规则——id % 128 / 16



2 实现—repo

- 管理与后端mysql的数据库连接（hikaricp）
- 读写分离



一个mysql实例上可以又多个分库，但是最好不要超过8个。

2 实现—Sequence原理

- 基于数据库表来存放序列值。
- 为了提高性能，会将序列值缓存在内存中，减少和数据库的交互。

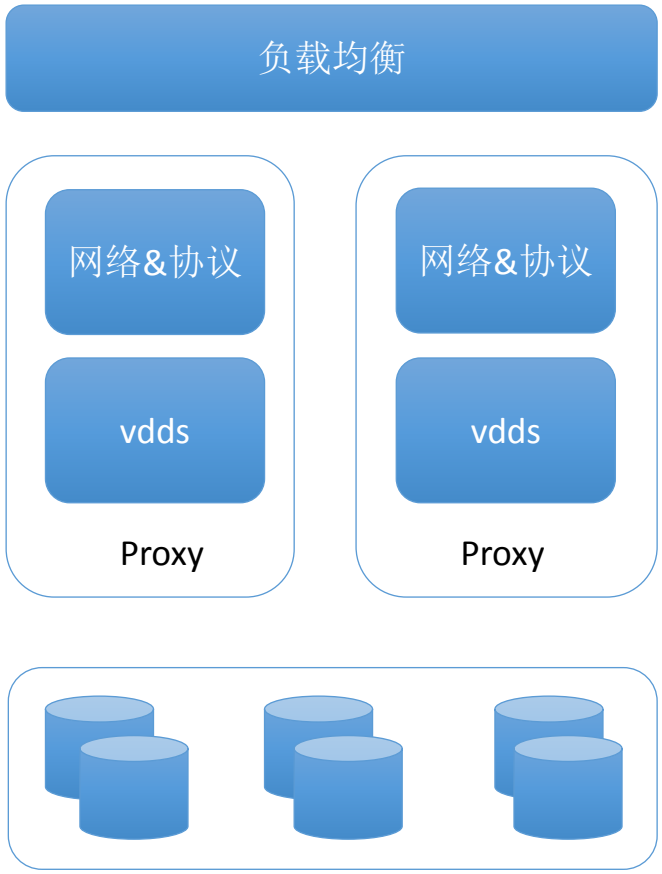
2 VDDS—后续规划

- OLAP分析类业务支持，更复杂的sql支持。并提供一些复杂查询的优化解决方案（包括类mpp解决方案）。
- 更多分库分表策略的支持。应对热点数据等。
- 扩容做到自动化，减少人力投入。
- 更多存储的支持。（目前只支持mysql，后续根据需要考虑支持nosql等）

3. Proxy核心功能

3. Proxy使用

3. Proxy 实现



3. Proxy后续规划

Q&A

Thanks!