保密等级
公开

Q/DX

青岛鼎信通讯股份有限公司技术文档

Q/DX D121.086-2022

MySQL 分库分表规范

V1.0

2022－06－15 发布　　2022－06－15

青岛鼎信通讯股份有限公司　　发 布

目　　次

前　　言

为保证工程技术本部自动化部对MySQL等关系型数据库分库分表的使用提出规范。

本规范定义了青岛鼎信通讯股份有限公司、青岛鼎信通讯消防安全有限公司、青岛鼎信通讯科技有限公司及相关公司的自动化系统上使用MySQL分库分表的方式和中间件使用进了建议和描述。

本规范适用于公司外发设备上自动化系统对MySQL数据库的验收和维护指导。

本标准由青岛鼎信通讯股份有限公司工程技术本部自动化部软件室起草。

MySQL 分库分表规范

1 范围

本规范定义了青岛鼎信通讯股份有限公司、青岛鼎信通讯消防安全有限公司、青岛鼎信通讯科技有限公司自动化方面使用MySql数据库时分库分表规范。

本规范适用于制造环境中的所有的自动化相关系统，包括SCADA系统、线体总控系统和柔性制造系统。

本规范其他关系型数据库可供参考。

2 规范性引用文件

本规范根据工作实际情况进行整理，并进行规范要求，无其他参考整理。

3 规范概述

规范主要包含以下几方面：

1）分库分表说明

2）分库分表方式

3）中间件模式

4）使用原则

4 分库分表说明

4.1 为什么分库分表

随着公司业务快速发展，数据库中的数据量增大，访问性能变慢，虽然通过机器集群，SQL优化，性能调优等方式在一定程度上可以起到提高性能的作用，但是并不能从根本上解决单库，单表数据量过大的问题,当使用MySQL数据库的时候，单表超出千万数据在性能上会极大降低。

物理服务器的CPU、内存、存储、连接数等资源有限，某个时段大量连接同时执行操作，会导致数据库在处理上遇到性能瓶颈。为了解决这个问题，对大表进行分割，将原来独立的数据库拆分成若干数据库组成，将数据大表拆分成若干数据表，使得单一数据库、单一数据表的数据量变小，从而达到提升数据库性能的目标，然后实施更好的控制和管理，同时使用多台机器的CPU、内存、存储，提供更好的性能。

4.2 什么是分库分表

热数据：指的是需要即时对用户进行分发的数据，即从数据源抓取之后经过数据清洗，需要即时存储到可以快速分发的存储介质（如 Redis等缓存数据库）供 API 或直接面向用户的系统使用。热数据线需要重点保障服务质量和稳定性，为了保证数据的时效性，在数据处理上也是优先级高的数据。热数据可能是临时或短期存储的，后来的数据可能会覆盖已有的数据。

冷数据：指的是不需要即时分发给用户的数据，这些数据甚至可能永远都不会原样分发给用户，但它们需要经过长期的积累，使我们可以从中得出基于此数据更高Level 的分析。冷数据典型的使用场景是供内部数据评估系统做数据准确度的评估分析，同时也可以给算法团队建模使用。设立这个数据线的原则是不影响热数据的服务质量，尤其是时效性和稳定性，同时也满足一些非线上项目的数据使用需求。

分库：从单个数据库拆分成多个数据库的过程，将数据散落在多个数据库中。

分表：从单张表拆分成多张表的过程，将数据散落在多张表中。

总结：把存于一个库的数据分散到多个库中，把存于一个表的数据分散到多个表中，一个库一个表拆分为 N个库N个表。

图1 MySQL分库分表

5 分库分表方式

5.1 垂直分库分表

垂直分库是指按照业务将表进行分类，分布到不同的数据库上面，每个库可以放在不同的服务器上，它的核心理念是专库专用。

垂直分表是数据的访问频次不同，所以可以根据数据的冷热进行表结构的拆分，从而提升性能。

通俗的说叫做“大表拆小表”，拆分是基于关系型数据库中的“列”（字段）进行的。

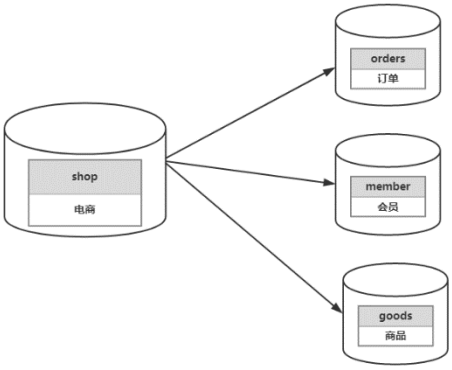

图2 垂直分库

特点：

- 每个库（表）的结构都不一样；
- 每个库（表）的数据都（至少有一列）一样；
- 每个库（表）的并集是全量数据；
- 按拆分字段（多表字段拆成少表字段）；

优点：

- 拆分后业务清晰（专库专用按业务拆分、和微服务概念相似、方便解耦之后的管理及扩展）；
- 高并发的场景下，垂直拆分使用多台服务器的 CPU、I/O、内存能提升性能，同时对单机数据库连接数、一些资源限制也得到了提升；
- 实现动静分离、冷热数据分离设计体现；
- 数据维护简单、按业务不同放到不同机器上；

缺点：

- 如果单表的数据量大、读写压力大；
- 受某种业务来决定、或者被限制。也就是说一个业务往往会影响到数据库的瓶颈（性能问题）；
- 部分业务无法关联 Join、只能通过程序接口去调用，应用层需要很大的改造，只能通过聚合的方式来实现、提供了开发复杂度和难度（例如产品信息、订单信息、员工信息等）；

5.2 水平分库分表

当一个应用无法更细颗粒度的垂直切分，或者说切分后数据量行数巨大，存在单库读写、存储性能瓶颈此时需要水平分库，经过水平切分的优化，往往能解决单库存储量和性能的瓶颈。但由于同一个表被分配在不同的数据库，需要额外进行数据操作的路由工作，因此会提升了系统复杂度。

水平分库是把同一个表的数据按一定规则（例如取模）拆到不同的数据库中，每个库可以放在不同的服务器上。

水平分表是在同一个数据库内，把同一个表的数据按一定规则拆到多个表中。

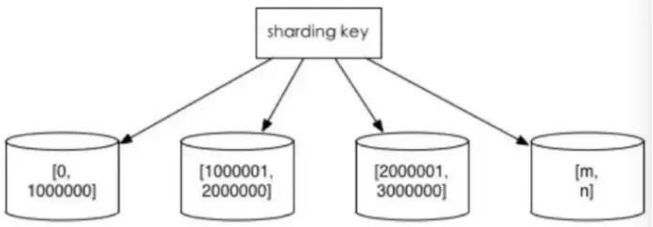

图3 水平分表

特点：

- 每个库（表）的结构都一样；
- 每个库（表）的数据都不一样；
- 每个库（表）的并集是全量数据；
- 总结：按数据（内容）拆分；

优点：

- 单库（表）的数据保持在一定的量（减少），有助于性能提高，水平无线扩展；
- 能够较好的应对高并发，同时可以将热点数据打散，提高了系统的稳定性和负载能力；
- 切分的表的结构相同、程序改造较少；

缺点：

- 数据的扩容难度大、维护量大；
- 拆分规则抽象麻烦；
- 分片事务的一致性的问题部分业务无法关联 Join、需要通过 Java 程序接口去调用；

6 中间件模式

6.1 Proxy 模式

把中间件作为一个独立的服务，它将接收到的SQL 语句做了一些特定的分析：如分片分析、路由分析、读写分离分析、缓存分析等，然后将此 SQL 发往后端的真实数据库，并将返回的结果做适当的处理，最终再返回给用户。

中间件有：

- MyCat 开源面向企业应用开发的大数据库集群；
- ShardingSphere-Proxy 透明数据库代理支持 MySQL 和 PostgreSQL；
- Cobar 阿里开源中间件；
- Atlas 360 公司基于 MySQL 官方中间件 MySQL-Proxy 二次开发；
- DRDS 阿里分布式关系型数据库；
- MySQL-Proxy MySQL 官方中间件；
- Heisenberg 百度 MySQL 分库分表中间件服务器；
- Oceanus 跨平台、58 同城数据库中间件；
- Vitess 云原生数据库系统，源于 YouTube；
- Oneproxy 基于 Mysql-Proxy 二次开发源于支付宝；

6.2 Client 模式

中间件在Driver或者连接池的基础之上，增加了一层封装。我们使用时，以Java开发为例，需要先引入一个Jar包。中间件接收持久层产生的SQL，同样对SQL进行分析等操作，然后才落实到具体的库上。

中间件有：

- ShardingSphere-JDBC Apache ShardingSphere 的一个独立的产品无需额外部署和依赖；
- TDDL 阿里类 JDBC 数据库中间件；
- Zebra 美团一个基于 JDBC API 协议上开发出的高可用、高性能的数据库访问层解决方案；
- Zdal 支付宝采用标准的 JDBC 规范自主研发的数据中间件产品；
- ShardingCore .Net EFCore 下高性能、轻量级针对分表分库读写分离的解决方案；

7 使用原则

MySQL为关系型数据库，数据库表之间的关系从一定的角度上映射了业务逻辑，任何分库分表的行为都会在某种程度上提升业务逻辑的复杂度，数据库除了承载数据的存储和访问外，协助业务更好地实现需求和逻辑也是其重要工作之一。

分库分表会带来数据的合并，查询或更新条件的分离，以及事务的分离等多种结果，业务实现的复杂程度往往会翻倍或指数级上升。在分库分表前，不能为分而分，而应该尽量去做其他力所能及的事情，例如升级硬盘、内存、CPU、网络、数据库等，进行读写分离及负载均衡。

基础原则：

- 单表超过 500 万条数据进行分表；
- 按照业务需求应用划分，类似于微服务思想，进行分库（安全性和可用性的考虑）；
- 数据库被切分后，不能再依赖数据库自身的主键生成机制，使用 UUID 或雪花算法生成分布式 ID；
- 可以考虑按年分库、按月分表的模式，数据量大时可考虑按日分表;
- JAVA 语言建议使用中间件 ShardingSphere-JDBC、ShardingSphere-Proxy、MyCat；
- .Net 语言建议使用中间件 ShardingSphere-Proxy、MyCat、ShardingCore（.NetCore）；

版本记录

版本编号/修改状态	拟制人/修改人	审核人	批准人	备注
V1.0	李健			