分布式应用层中间件的设计

李文杰,周剑华

(武汉科技大学 计算机科学与技术学院,湖北 武汉 430065)

摘 要:针对如何在非分布式数据库管理系统中应用分布式特性,提出了分布式数据层中间件DDLM的设计方案。在数据持久化框架和JDBC之间引入一个分库分表的中间件,从而把数据拆分到多个数据库的多个表中,在用户看来这些数据仍然存在于一张表中,从而在应用层透明地解决了海量数据的读写问题。

关键词:分布式数据层;逻辑表;物理表

中图分类号:TP311 **文献标识码**:A **文章编号**:1674-7720(2011)05-0012-03

Design of middleware of distributed database in application layer

Li Wenjie, Zhou Jianhua

(College of Computer Science and Technology, Wuhan University of Science and Technology, Wuhan 430065, China)

Abstract: This paper proposes design scheme for distributed feature middleware without distributed database management systems. Data was split into multiple tables in multiple databases with the middleware between persistence framework and JDBC (Java Database Connectivity), all data is just in one table in user view. It is transparent to solve the problem of reading and writing massive data by the middleware.

Key words: distributed data layer; logic table; physical table

随着互联网应用业务的高速增长,搜索引擎、电子商务、门户网站等大型互联网公司的网络信息流量直线上升,日访问量甚至突破亿次大关,从而产生了海量信息和对这些信息的海量读写,集中式数据库越来越难以满足互联网公司对海量信息的高可靠性、高扩展性的需求。

分布式数据库通过对数据进行垂直分片和水平分片,让数据存储在多个数据库中,能够解决海量数据的存储和管理问题。所谓垂直分片是把一个全局关系的属性集分成若干子集,并在这些子集上作投影运算,每个投影称为垂直分片。属性集数目是一定的,垂直拆分只能适合一定规模的扩展,当对每个垂直分片的访问超过单数据库所能承受的负载时,就需要水平分片。水平分片是按一定的条件把全局关系的所有元组划分成若干不相交的子集,每个子集为关系的一个片段。

目前市场上Oracle、DB2等商用分布式数据库的价格昂贵,一般企业仅仅将商用分布式数据库用来管理企业最核心的数据,而非核心的数据则存放在PostgreSQL、MySql等开源数据库中。然而大多数开源数据库分布式功能不够强大,甚至不具备分布式的功能。为了解决这个问题,本文提出了分布式数据层中间件的设计方案,在应用层把数据垂直、水平拆分到多个数据库、多张表中,使应用层具备了分布式的功能,和底层数据库是否具有分布式特性没有关系,从而使底层的开源数据库能够通过分布式数据层中间件具有分布式的特性。

1 分布式数据层中间件的设计原理

传统的持久化框架是基于JDBC的,如JPA(Java Persistence API)、Hibernate和TopLink等。对象关系映射(ORM)框架是根据对象的属性生成Sql语句,然后调用JDBC API完成数据的持久化操作。Ibatis是个JDBC模板,相当于半自动化ORM映射工具,也是调用JDBC接口来完成对数据的持久化操作的。

所有Java持久化框架对数据库的持久化操作都是直接或者间接地调用JDBC API执行Sql语句来完成对数据的CRUD操作,每条Sql语句通常只操作单数据库。在持久化框架(如Hibernate)和JDBC之间设计一个

分布式数据层中间件 DDLM (Distributed Data Layer MiddleWare),DDLM 层把业务逻辑层的每条 Sql 语句(下文记作逻辑 Sql 语句)按照垂直、水平拆分的策略解释成多个 Sql 语句,解释后的每条 Sql 语句(下文记作物理 Sql 语句)对一个数据源进行操作,从而一条逻辑 Sql 语句被解释成多条物理 Sql 语句,因此 DDLM 具有分布式的特性。

分布式数据层中间件的原理如图 1 所示。图中把持久化层分为四个子层:持久化框架、分布式数据层、JDBC、数据库。例如:JPA 把根据 ORM 映射规则生成的 Sql 语句交给分布式数据层,分布式数据层中间件把 Sql 语句解释为多个物理 Sql 语句交给 JDBC 接口,JDBC 接口完成对数据库的 CRUD 操作。

这样分布式数据层就能够完成原本只有分布式数据库才能完成的垂直分片、水平分片、合并排序等分布式操作。用户不需要使用新的管理工具,只需要利用原有数据库的管理工具与分布式数据层中间件交互。该层把对多个物理数据库的操作透明化。

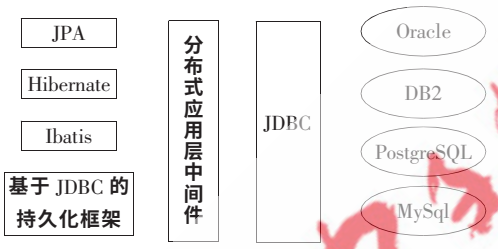

图 1 分布式数据层原理图

2 分布式数据层中间件的设计方案

在 DDLM 设计中,不需进行垂直分片。一个全局关系对应一张数据库表,这样就能满足应用中的大多数需求。而且按照一个关系映射一张表的原则拆分,逻辑简单清晰,简化了数据库模型的设计。DDLM 的研究重点是对表的水平分片以及水平分片后产生的问题的解决。

水平分片把关系模式 R 的记录拆分到 $n(n\geqslant 1)$ 个物理数据库中,每个物理数据库有 $m(m\geqslant 1)$ 张数据表,模式 R 的记录被路由到 $n\times m$ 张模式相同的物理数据库表中。

水平分片后,记录存在于不同的物理数据库,随之产生了两个问题:查询数据时需要合并并且排序、主键需要全局唯一生成。

2.1 分库策略

一个数据库所能存放的表数目会受到文件系统的限制,有必要把一张逻辑表的数据拆分到多个物理数据库中。为了实现此功能,在表模式中添加一个整数类型的 db_num 字段,db_num 字段的值指示了记录(也称作元组)被路由的目标数据库。下面举例说明 db_num 字段的作用:

设关系模式为 R(id,…,db_num,…),该模式对应的表的数据需要被路由到 $N(N\times 1)$ 个物理数据库内,任意一条记录(id_value,…,n,…)存在于第 n 个物理数据库的某张表中($0<n\leqslant N$,n 为 db_num 字段的值)。

2.2 分表策略

数据库表存放记录数量的最大值在理论上可以取很大的值,但在实际应用中通常受到文件系统的限制。当一张表的数据记录数达到一个阈值时,操作该表的速率会急剧下降。在 MySql 数据库中,当表记录数达到 1 000 万条时,查询该表的速率明显地下降。

在同一个数据库建立多张模式相同的表,数据被路由到不同的表中,从而可以很好地解决表记录过多引起速率下降的问题。每条记录要唯一地标示它所在表的编号,因此必须引入某种编码手段存放该记录的编号。有两种常用的策略:(1)用记录的主键标示该记录所在表的编号,也就是数据库表主键拆分策略;(2)特意引进一个日期字段标示记录所在表的编号,也就是数据库表日期字段拆分策略。

2.2.1 数据库表主键拆分策略

假设逻辑表模式 R 的记录在一个数据库中需要分别路由到 $M(M\geqslant 1)$ 张物理表中,设逻辑表 R 的表名为 logic_table_name,物理表的表名分别是 table_1,table_2,…,table_M。

设表 R 的模式为 R(id,…),其中 id 是模式的主键,其数据类型为整数类型。R 的任意一条记录 r(x,…),其主键值为 x,r 被路由到物理表 table_y 中(y 的值为 x 和 M 取模的结果,即:$y=x\%M$)。

随着记录主键值 id 的增加,记录可以非常均匀地路由到 M 张物理表中。然而,如果需要动态增加 M 的值,如 M 的值由 M 增加到 M',则记录就不会均匀地分配到 M' 张物理表中。此时可以采取表日期字段拆分法。

2.2.2 数据库表日期字段拆分

按照表的日期字段拆分数据是另一种常用的拆分策略,当数据量比较大时,暂时无法估算到底需要多少张物理表才能存放一个模式的所有记录,此时可以采取按表日期字段拆分策略。

设数据库表模式为 R(id,column1,…,update_time),update_time 字段是该记录创建时的系统时间,任意一条记录 r(x,column1_vlaue,…,update_time_value)。在应用层读取系统的时间可以计算得到 update_time_value 时间值是一年中的某天 day_of_year,这样就可以把数据拆分到 365(或 366) 张表中,物理表名分别为 table_name_0,table_name_1,…,table_name_day_of_year,…,table_name_365(或 365)。

除了按照取得 update_time_value 的 day_of_year 值,也可以取得 update_time_value 在星期中的某天 day_of_week 和在月的某天 day_of_month。DDLM 中间支持按照时间的各种策略。

为了最大化地拆分数据,DDLM 还提供以上策略的二级拆分。

2.3 数据合并排序策略

分库分表后，一张逻辑表 table_name 的数据存储在不同的物理表中，在对表进行查询、删除和更新时，一条 Sql 语句可能会同时对一张或者多张物理表的数据产生影响。对于删除、更新操作，分别针对每个物理数据库执行对应的删除、更新语句，然而对于查询语句涉及到多个物理数据库时，不能简单地针对每个数据库执行查询语句，还需要合并所有的查询结果并且排序。下面举例说明查询合并以及排序策略。

假设物理表表名分别为 table_name0，table_name1，…，table_name$N$，同时设 Sql 语句为 SELECT*FROM talbe_name WHERE update_time=today OR update_time=yesterday ORDER By id LIMIT a，b（其中 a，b 为自然数），又假设该逻辑表 table_name 是按 update_time 日期字段水平分片的，则 Sql 的查询会涉及到物理表中的两张表记为 table_nameX（$0 \leqslant X \leqslant N$），table_name_$Y$（$0 \leqslant X \leqslant N$）。该 Sql 的执行流程如下：

(1) 对表 table_nameX 查询操作 SELECT*FROM table_nameX WHERE update_time=today 得到结果集 ResultSet1，并对表 table_nameY 执行和 SELECT*FROM table_nameY WHERE update_time=yesterday 得到结果集 ResultSet2。

(2) 从结果 ResultSet1 和 ResultSet2 读取数据存放在一个集合 Result 中，按照 id 字段排序。

(3)在集合 Result 中，读取 id 分布在区间[a，$a+b$]上的记录作为返回结果。

通过上述查询合并排序策略，当在查询过程中涉及到多张物理表时，能够分别读取多张物理数据库表的数据，然后在内存中对数据分别执行合并、排序和分页操作。合并排序需要一定的时间和空间，所以在查询时，尽量不要同时涉及到两个或者以上的数据库。

2.4 主键生成策略设计

在 DDLM 中，数据被路由到多个数据库的多张表中，为了确保主键的全局唯一性，不能借助于数据库管理系统 DBMS 来生成主键，因为 DBMS 生成的主键只在当前数据库中具有唯一性，不能确保主键的全局唯一性。有两种策略可以生成具有全局唯一性的主键：(1)采用通用的 UUID 生成策略，UUID 是借助主机的时间戳、IP 地址和网卡 Mac 地址等生成分布式唯一标示符的算法，但是该策略生成的唯一标示符需要用 32 个字符来存储，非常浪费空间；(2) 借助分库分表的信息生成主键，该策略非常有效地利用了分库分表的路由信息，巧妙地生成全局唯一主键。下面将详细地介绍该策略。

假设一张逻辑表 logic_name 的数据分别存储在数据库 db_1，db_2，…，db_s（s 为大于 1 的正整数）中，每个数据库中有相同的表 table_1，table_2，…，table_t（t 为大于 1 的正整数）。用三位作为数据库的编号、三位作为表的编号以及一个随机字段来构成全局唯一主键。数学表达式为 xxxyyym…m，xxx 为数据库的编号，yyy 为表的编号，m…m 为随机数。该主键生成策略有两个优点：(1)实现方便，通常一张逻辑表的数据不会多得需要被路由到 1 000 个物理数据库以上，也不会路由到 1 000 张表以上；(2)主键本身就包含有路由信息。使用此策略，由主键信息就能路由该记录，而不必查询配置信息。假设一条数据库记录的主键为 10020012345，取出前三位为 100，则该记录应该路由到编号为 100 的数据库（记为 db100），取出 4~6 位为 200，则该记录应该路由到 db100 的编号为 200 的表。

DDLM 在应用层透明地把逻辑数据库表的数据拆分到多个物理数据库的多张表中，同时提供合并查询排序、主键生成等功能，从而可以在不支持分布特性的数据库管理系统应用分布式特性。

参考文献

[1] 林昊.分布式 Java 应用：基础与实践[M].北京：电子工业出版社，2010.

[2] 何坤.基于内存数据库的分布式数据库架构[J].程序员，2010(7)：116.

[3] 潘群华，吴秋云，陈宏盛.分布式数据库系统中数据一致性的维护方法[J].计算机工程，2002(9)：12-15.

[4] 习周龙.分布式数据库管理系统实现技术[M].北京：科学出版社，1999.

[5] 赵致格.数据库系统与应用[M].北京：高等教育出版社，1994.

（收稿日期：2010-10-10）

作者简介：

周剑华，男，1985 年生，硕士研究生，主要研究方向：分布式系统。

李文杰，女，1969 年生，副教授，主要研究方向：数据库与数据挖掘，基于网络的计算机应用等。