分散式数据存储设计

一、应该考虑分散式数据存储设计

企业内的生产线类系统，一开始都对性能考虑不足，在设计中基本上都采用单一数据库来支持业务，随着业务规模的扩大，对数据库性能要求提升时，企业会采购更昂贵的软硬件产品来支持更大的访问压力，当实在无法缓解时，会考虑在数据库端采用分区等方法来分散硬件压力，但是数据库本身从逻辑上还是单一的实例。

由于所有数据服务都在集中在少数的关键硬件上，就会造成几方面的问题，一是极高性能的硬件成本太高，二是所有的鸡蛋放在了同一个篮子里，服务故障造成的灾难影响面太大，三是硬件的升级总是跟不上数据量、访问量的提升速度，整个系统总是慢吞吞的。

要解决这个问题，可以采用的第一个设计思路，是按业务领域划分为若干个系统，每个系统之间再做服务、数据、流程的集成。这种思路已经广为采用，优劣各人自知，不再多说，本文想讨论的是另一个更为大胆激进的想法：分散式客户数据。

Facebook、twitter、微博这种社交网站面对的访问量远远大于企业应用，与企业应用相比而言，社交网站还要考虑如“突然大批的用户涌来怎么办？”、“某个普通用户突然火了怎么办？”这种突发性能要求的处理。

他们解决方案是在软件逻辑上就支持分散存储，即建立数量不限的数据库，数据被按一定的策略分散保存在不同的数据库硬件里，这样在访问不同的数据时，可以按策略来判断应该去哪个数据库里去寻找数据。由于数据库是独立的，只需要增加一些低端的硬件，就可以达到平滑升级性能的目的。

为了与逻辑上集中物理上分布的表分区等技术相区别，我以“分散”一词来表示这种在逻辑上的多数存储方案。

数据分散的策略有哪些？主要是三种：

一是按业务（产品）领域划分。这种方式其实与分为多个子系统没有区别，但主要业务一般也是数据量最大、访问量最大的，按这种划分的话，可能还是无法解决最主要的问题。

二是按业务时间划分。这种方法对于一些偏重近期的业务会有作用，如股票交易，但是对于依靠长期信息的业务则不实用，比如长期保险业务中，保单生效越长，反而越有可能发生理赔、给付等，而且在这些业务处理过程中，要依据业务对象的所有信息进行处理，如果数据是按时间分散保存的，软件系统逻辑上则有较大幅度的复杂度增加。

三是按主业务对象的“群落”来划分。比如把客户分群，某一个客户的所有相关业务信息（比如他的博文、有关评论等）不可分散，按群放在不同的物理数据源中。这样，当系统存取某个客户的信息时，可以按群落找到客户所在的物理数据库，之后的逻辑则与集中式数据库的处理大体类似。

如何得知客户所在群落呢？对于不存在个别突发的访问增长的系统，可以暂时采用平均式的分群方案，用某种约定算法求得客户群号，比如说按客户证件号的前几位。但是这种分类一但固定了就不容易更改，例如，一开始用证件号第一位平均分为10个数据库，但如果业务增加到10个数据库也顶不住时，要修改这个分群就很困难了。

另一个办法就是采用一个中心数据库来记录用户所在群落，这个中心数据库里只需要一张表、最少三个字段就够了——用户标识、群落号或物理数据源地址、可访问状态。当系统要开始处理一个用户的业务时，需要向这个数据库进行询问，得到群号后就可以转向所在数据源进行业务处理。

这种办法的好处很大，比如微博这种社会网络上，常常会有人突然火了，前一天还是无名小卒，一百多个粉丝，每天两篇微博，被与其他五十万同样的屁民放在某个刀片机上的mysql里默默无闻着，谁知道第二天他可能运气太好（或是运气太坏）赶上了个什么热点时件，突然有几十万人来关注他，那个机器立即就顶不住了。在这种情况下，就用数据迁移程序把这个人的有关信息全部移到另一台小型机上去跑，迁移完成之后修改中心数据库，压力立即就调整了。

这种迁移甚至可以做成自动的，发现哪个用户火了就自动启动，按策略移到预备的空闲计算资源上去。

这样的系统架构下，压力最大的地方就成了中心数据库，但是第一，这个数据库中里没有业务数据，数据结构很简单，极容易优化索引与性能，就是买好硬件，所需的投资也有限。其次，用户分群策略可以结合固定规则与特殊指定分群，中心数据库可以只保存特殊指定分群的记录，如果没有记录则是按固定规则（证件号第一位？）来分。再次，应用服务器也可以按缓存访问过的用户分群信息，只在按固定规则断言数据源后访问失败、而且没有缓存过或按缓存访问失败的情况下才访问中心数据库，得到用户的最新群落位置。

不可否认，这种模式会对全局性数据访问需求造成技术困难，例如需要统计全系统的一些数据时，就需要协调所有的数据库进行分别运算，最后再把结果汇总，这肯定需要一些全新的开发与运维架构来支持，但对于巨量数据的系统来说，统计结果的绝对精确性已经让位于系统可以正常运行的要求。