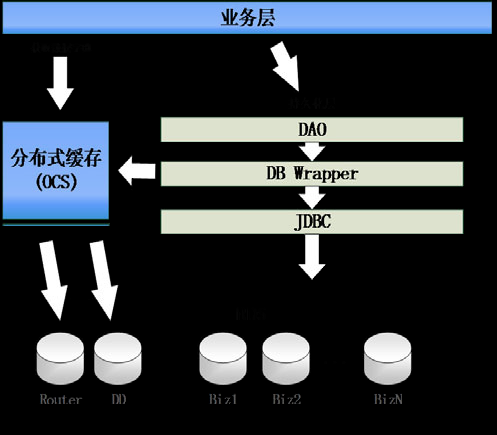
**水平拆分：按用户将数据拆分到多个库的相同表中**

水平拆分的思路，就是将原本存放在单个RDS数据库中的数据，根据业务ID不同，拆分到多个数据库中（参见图2）。拆分后，各库的表数量及表结构都保持一致。水平拆分首先需要确立唯一的业务主表，即其他所有表的数据都与主表ID（前文所说的业务ID）存在直接或间接的主从关系，可以通过主表ID对全部数据做很好的切分。我们选择的业务主表为用户表，其他业务表或表的父表都包含一个用户ID。因此，我们切分的目标就是将不同用户数据存放到不同的数据库中。



图2  水平拆分示意图

确定了拆分规则后，下一步是着手封装Sping数据访问封装层（DBWrapper）。DBWrapper介于DAO与JDBC之间，每个业务DAO进行数据库基本操作，都会经过DBWrapper。它的主要作用是将数据库架构的变化对业务层透明，业务层可以如同操作单个DB一样，调用DBWrapper提供的数据库操作接口，而判断操作哪个数据库的逻辑，则全部交由DBWrapper封装完成（参见图3）。



水平切分是指，以某个字段为依据（例如uid），按照一定规则（例如取模），将一个库（表）上的数据拆分到多个库（表）上，以降低单库（表）大小，达到提升性能的目的的方法，水平切分后，各个库（表）的特点是：

（1）每个库（表）的结构都一样

（2）每个库（表）的数据都不一样，没有交集

（3）所有库（表）的并集是全量数据

垂直拆分是指，将一个属性较多，一行数据较大的表，将不同的属性拆分到不同的表中，以降低单库（表）大小，达到提升性能的目的的方法，垂直切分后，各个库（表）的特点是：

（1）每个库（表）的结构都不一样

（2）一般来说，每个库（表）的属性至少有一列交集，一般是主键

（3）所有库（表）的并集是全量数据

还是以上文提到的用户表为例，如果要垂直拆分，可能拆分结果会是这样的：

user\_base(

uid bigint,

name varchar(16),

pass varchar(16),

age int,

sex tinyint,

flag tinyint,

…);

user\_ext(

uid bigint,

sign varchar(64),

intro varchar(256)

…);

三、垂直切分的依据是什么

当一个表属性很多时，如何来进行垂直拆分呢？如果没有特殊情况，拆分依据主要有几点：

（1）将长度较短，访问频率较高的属性尽量放在一个表里，这个表暂且称为主表

（2）将字段较长，访问频率较低的属性尽量放在一个表里，这个表暂且称为扩展表

如果1和2都满足，还可以考虑第三点：

（3）经常一起访问的属性，也可以放在一个表里

优先考虑1和2，第3点不是必须。另，如果实在属性过多，主表和扩展表都可以有多个。

一般来说，数据量并发量比较大时，数据库的上层都会有一个服务层。需要注意的是，当应用方需要同时访问主表和扩展表中的属性时，服务层不要使用join来连表访问，而应该分两次进行查询：

原因是，大数据高并发互联网场景下，一般来说，吞吐量和扩展性是主要矛盾：

（1）join更消损耗数据库性能

（2）join会让base表和ext表耦合在一起（必须在一个数据库实例上），不利于数据量大时拆分到不同的数据库实例上（机器上）。毕竟减少数据量，提升性能才是垂直拆分的初衷。

四、为什么要这么这么拆分

为何要将字段短，访问频率高的属性放到一个表内？为何这么垂直拆分可以提升性能？因为：

（1）数据库有自己的内存buffer，会将磁盘上的数据load到内存buffer里（暂且理解为进程内缓存吧）

（2）内存buffer缓存数据是以row为单位的

（3）在内存有限的情况下，在数据库内存buffer里缓存短row，就能缓存更多的数据

（4）在数据库内存buffer里缓存访问频率高的row，就能提升缓存命中率，减少磁盘的访问

举个例子就很好理解了：

假设数据库内存buffer为1G，未拆分的user表1行数据大小为1k，那么只能缓存100w行数据。

如果垂直拆分成user\_base和user\_ext，其中：

（1）user\_base访问频率高（例如uid, name, passwd, 以及一些flag等），一行大小为0.1k

（2）user\_ext访问频率低（例如签名, 个人介绍等），一行大小为0.9k

那边内存buffer就就能缓存近乎1000w行user\_base的记录，访问磁盘的概率会大大降低，数据库访问的时延会大大降低，吞吐量会大大增加。

五、总结

（1）水平拆分和垂直拆分都是降低数据量大小，提升数据库性能的常见手段

（2）流量大，数据量大时，数据访问要有service层，并且service层不要通过join来获取主表和扩展表的属性

（3）垂直拆分的依据，尽量把长度较短，访问频率较高的属性放在主表里