**2-1 MyCat概述与基本概念**

## **什么是MyCat**

MyCat 是什么？从定义和分类来看，它是一个开源的分布式数据库系统，前端的用户可以把它看成一个数据库代理，用MySql客户端和命令行工具都可以访问，而其后端则是用MySql原生的协议与多个MySql服务之间进行通信。MyCat的核心功能是分库分表，即将一个大表水平切分成N个小表，然后存放在后端的MySql数据当中。

MyCat发展到目前的版本，已经不是一个单纯的MySql代理了，它的后端支持MySql，Oracle，SqlServer，DB2等主流的数据库，也支持MongoDB这种NoSql数据库。而对于前端的用户来说，无论后端采用哪一种数据库，在MyCat里都是一个传统的数据库，支持标准的SQL语句，对于前端的开发人员来说，可以大大地降低开发难度，提升开发速度。

对于DBA来说，可以这样理解MyCat：

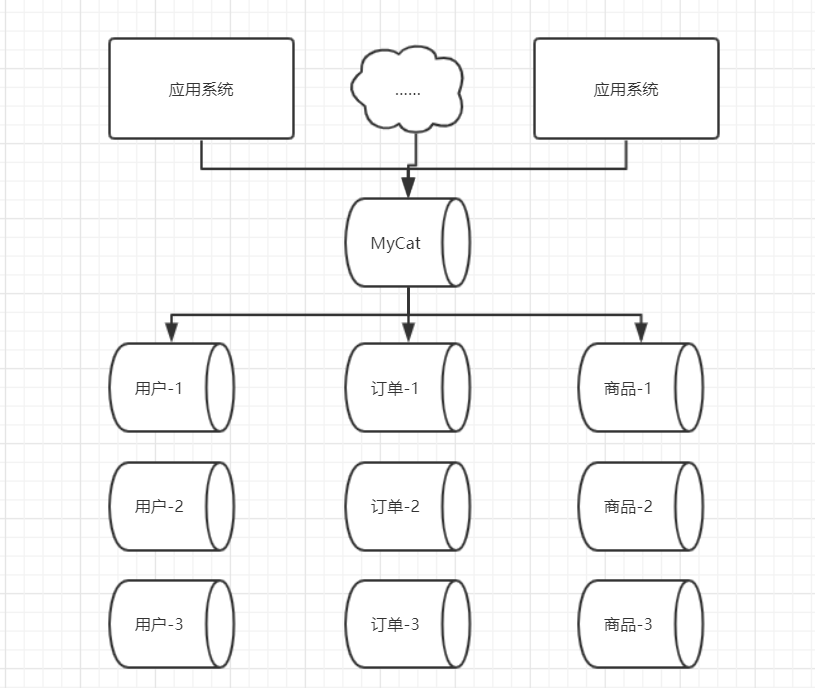
MyCat就是MySql，而MyCat后面连接的MySql，可以理解为MySql中的存储引擎，比如：MyISAM、InnoDB等。所以，MyCat本身不存储数据，数据都是存储在MyCat后面连接的MySql上，数据的可靠性和事务都是MySql保证的。

对于开发人员，可以这样理解MyCat：

MyCat就是一个近似等于MySql的数据库库服务，你可以使用连接MySql的方式连接MyCat。绝大多数情况，你也可以使用常用的ORM框架连接MyCat，但是，对于分片的表，还是建议使用标准SQL语句，这样能够达到最佳的性能。

对于架构师来说，可以这样理解MyCat：

MyCat是一个强大的数据库中间件，不仅仅可以用作读写分离、分库分表，还可以用于容灾备份，云平台建设等，让你的架构具备很强的适应性和灵活性。



## **MyCat的应用场景**

MyCat发展到现在，使用的场景很丰富，常见的典型的应用场景有：

* 单纯的读写分离，此时配置最为简单，支持读写分离，主从切换；
* 分库分表，对于超过1000w的表进行分片，最大支持1000亿的数据；
* 多租户应用，每个应用一个数据库，应用只连接MyCat，程序本身不需要改造；
* 代替HBase，分析大数据

在咱们的课程中，主要给大家讲解读写分离和分库分表，这也是MyCat应用最多，最基础的功能。

## **MyCat中的基本概念**

MyCat是一个数据库的中间件，介于应用与数据库之间，是进行数据处理和交互的中间服务。正是由于它是一个数据库代理的中间件，它里边也有很多新的概念，这些概念先给大家介绍一下，大家有个印象，对后面的学习也很有帮助。

### **逻辑库（Schema）**

在实际的开发中，开发人员不需要知道数据库中间件的存在，开发人员只需要有数据库的概念就可以了。所以数据库中间件可以被看做是一个或者多个数据库集群构成的逻辑库。例如：上图中的例子，我们可以理解为系统先做了垂直切分，被分为了3个库，用户库，订单库，商品库，而这3个库就被称为逻辑库。

### **逻辑表（table）**

既然有逻辑库，那么就有逻辑表，对于应用系统来说，读写数据的表，就是逻辑表。而逻辑表中的数据，则是被水平切分后，分布在不同的分片库中。如上图所示：假设用户库中有一张用户表，这个用户表就被称为逻辑表，而用户表又被水平切分为3个表，每一个表中都存储一部分用户数据。业务系统在进行用户数据的读写时，只需要操作逻辑表就可以了，后面的分片细节则由MyCat进行操作，这些对于业务开发人员来说时完全透明的。当然，有些表的数据量没有那么大，完全不需要进行分片，只在一个物理的数据库表中即可。

凡是我们做的数据水平切分的表，我们把它叫做分片表。而数据量比较小，没有进行分片的表，我们叫它非分片表。

在真实的业务系统中，往往存在着大量的字典表，这些表的数据基本上很少变动，比如：订单状态。我们查询的时候，往往需要关联字典表去查询，比如：查询订单时，需要把订单状态关联查出，如果订单表做了分片，分布在不同的数据库中，而订单状态表由于数据量小，没有做分片，那么我们查询的时候就要跨库关联查询订单状态，增加了不必要的麻烦，不如我们干脆把订单状态表冗余到所有的订单分片库中，这样关联查询就不需要跨库了。我们把这种通过数据冗余方式复制到所有的分片库中的表，叫做全局表。

### **分片节点（dataNode）**

数据被切分后，一张大表被分到不同的分片数据库上面，每个分片表所在的数据库就叫做分片节点。

### **节点主机（dataHost）**

数据切分后，每一个分片节点不一定都会占用一个真正的物理主机，会存在多个分片节点在同一个物理主机上的情况，这些分片节点所在的主机就叫做节点主机。为了避免单节点并发数的限制，尽量将读写压力高的分片节点放在不同的节点主机上。

### **分片规则（rule）**

一个大表被拆分成多个分片表，就需要一定的规则，按照某种业务逻辑，将数据分到一个确定的分片当中，这个规则就叫做分片规则。数据切分选择合适的分片规则非常重要，这将影响到后的数据处理难度，结合业务，选择合适的分片规则，是对架构师的一个重大考验。对于架构师来说，选择分片规则是一个艰难的，难以抉择的过程。

### **全局序列号（sequence）**

大家有没有想过，数据切分以后，数据库表的中的id怎么办？原来在一张表的时候，我们采用id自增，但是数据分布到多个库怎么办？比如：向用户表插入数据，第一条记录插入了用户库1，它的id为1；第二条记录插入了用户库2，如果是自增，它的id也为1。这样id就混乱了，我们也无法确定一条数据的唯一标识了。这时，我们需要借助外部的机制保证数据的唯一标识，这种保证数据唯一标识的机制，我们叫做全局序列号。

## **总结**

通过上面我们对MyCat的介绍，我相信大家对MyCat有了一个整体的认识。在接下来的视频课程中，我们就来体验一下MyCat的读写分离和分库分表的强大功能。