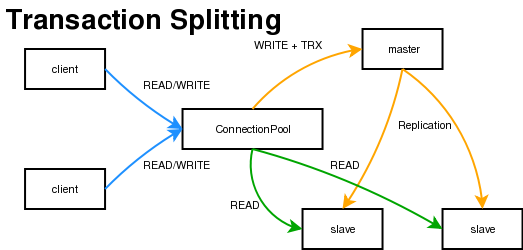
1. 业界可用的mysql读写分离技术
2. Mysql Proxy：处于客户端应用程序和MySQL服务器之间，通过截断、改变并转发客户端和后端数据库之间的通信来实现其功能，通过lua脚本来控制连接转发的机制。

优点：配置简单，一个lua脚本就全部搞定。

缺点：不够稳定，单节点经常出现问题，并且压力会非常大，目前没有正式版，官网都不推荐成产使用。

原理图：



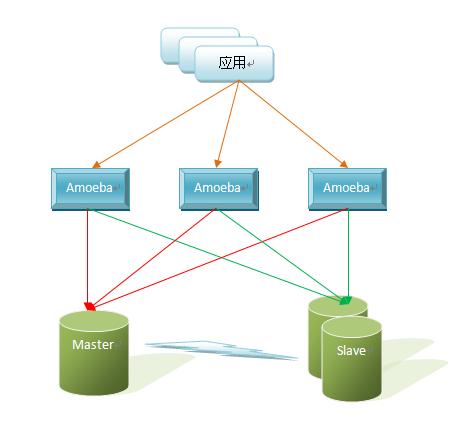
官网：https://downloads.mysql.com/archives/proxy/

1. Amoeba : 处于应用层和数据库之间，将应用传递过来的SQL语句经过分析后，将写的语句交给Master库执行，将读的语句路由到Slave库执行，也可做master读，看配置，已经做了多个读的负载均衡(轮询算法)，也可以进行水平扩展，以分担压力和提升可用性。

优点：读写分离，垂直分库，水平分库，容错，故障恢复功能和稳定性都非常的不错，产品线。

缺点：**Amoeba不支持事务，Amoeba不支持跨库join和排序，Insert语句必须指定表的列名，Amoeba不支持分库分表，Amoeba不支持大数据量的查询(10万+)，Amoeba需要更严格的SQL语句规范，不支持分布式事务。**

原理图：



官网：<https://sourceforge.net/projects/amoeba/>

1. 使用程序实现
2. 直接在程序中配置多份数据库，直接指定读写库【×】

工作量会很大，并且代码无法重复使用，显得很臃肿很乱。

1. 使用AOP在dao层根据方法名前缀进行切入数据源【×】

在同一个service中如果有读有写则无法使用事务，因为垮了数据源。

1. 使用AOP在service层根据方法名前缀进行切入数据源【√】

在同一个service中如果有读有写可以使用事务，但是只能使用其中一个数据源，读和写都在主库，但是目前项目中大多数都是读写都存在，所以其实起不到多大作用，但是没有风险。

1. 其他技术对比

<https://blog.csdn.net/kobejayandy/article/details/60869530>

1. Mycat技术可以选用

http://www.mycat.io/