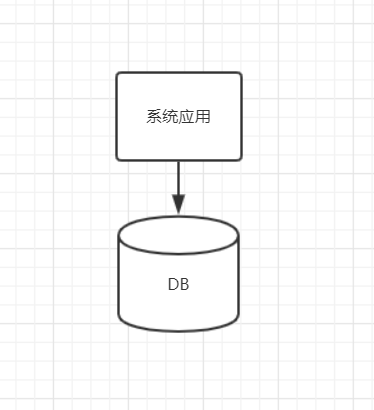
**1-2 如何正确使用数据库读写分离**

## **背景**

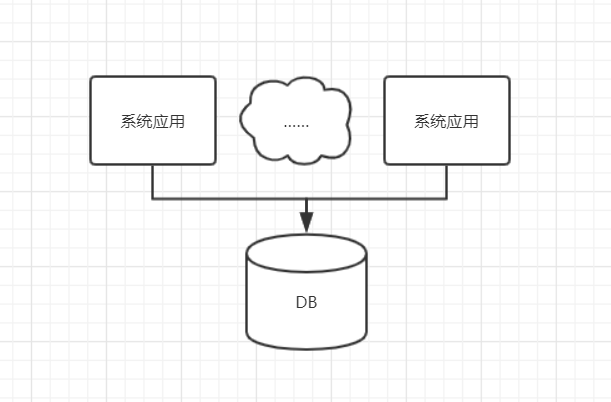
在应用系统发展的初期，我们并不知道以后会发展成什么样的规模，所以一开始不会考虑复杂的系统架构，复杂的系统架构费时费力，开发周期长，与系统发展初期这样的一个定位是不吻合的。所以，我们都会采用简单的架构，随着业务不断的发展，访问量不断升高，我们再对系统进行架构方面的优化。

## **架构演进**

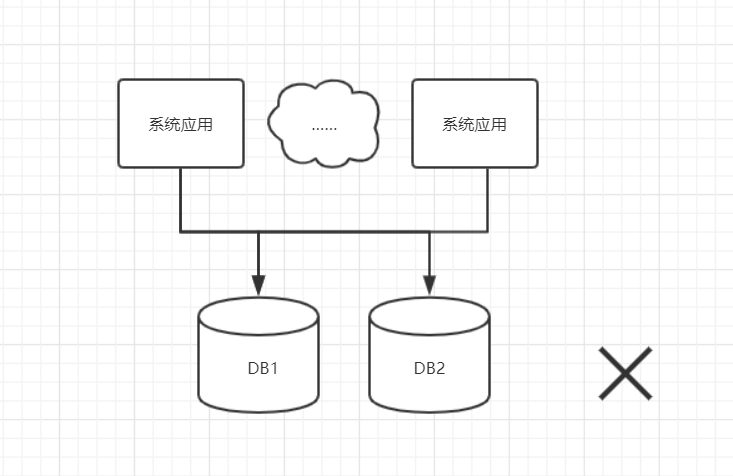
系统建立初期，我们的架构都非常的简单，主要满足业务的正常运行，如图：



但是随着访问量的升高，人们对系统的可靠性有了更高的要求，所以，我们为了避免单点故障，对系统应用层进行了横向的扩展，如图：

这样，保证了系统应用层的高可用，在发生宕机，或者系统升级时，系统对外还是可用的。而且在访问量升高的时候，系统应用层的压力也会得到分摊，使得每一个单体的系统应用的压力在一个合理的区间范围内。

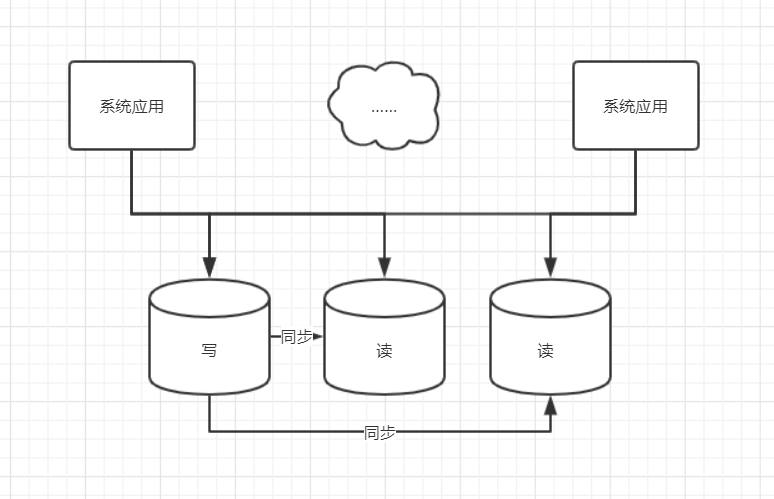
但是，随着访问量的升高，所有的压力都将集中到数据库这一层。那么数据库这一层，我们要怎么处理呢？能不能像系统应用层那样进行扩展呢？答案当然时否定的，我们想象一下，如果数据库层也像系统应用层那样，进行横向扩展，如图：



那么，如果系统应用层产生了一条数据，这条数据应该插入到DB1还是DB2呢？假设插入了DB1，那么这条数据被读取时，应用层怎么知道从哪个数据库读取这条数据呢？问题是不是很复杂，如果数据库不进行扩展，那么一台数据库是承载不了这么大的访问量的，那我们怎么办呢？

## **数据库读写分离**

办法总比问题多，随着互联网技术的发展，以及一代代互联网人对互联网的深入研究，人们发现在互联网的系统应用是一个读多写少的应用，比如咱们课程中的电商系统，商品浏览的次数是比下单要多的。数据库承载压力大，主要是由这些读的请求造成的，那么我们是不是可以把读操作和写操作分开，让所有读的请求落到专门负责读的数据库上，所有写的操作落到专门负责写的数据库上，写库的数据同步到读库上，这样保证所有的数据修改都可以在读取时，从读库获得，系统的架构如图所示：



如果系统的读请求比较多的话，读库可以多部署几台，这样读请求就可以均摊到多台读库上，降低每一个读库上的压力。但是在写数据的时候，数据要落在一个确定，且唯一的写库中。上图中，咱们的写库只有一个，你当然可以部署多个写库，但是数据怎么分片是一个十分重要的问题，这个问题我们在后续的课程中会给大家做介绍。目前仅以一个写库为例，比如：商户发布商品时，将这个商品的数据落在了写库上，同时，写库将这条数据同步给两个读库，买家在网站浏览商品时，会从读库将这个商品数据读取。至于从哪个读库取出数据，那就要看这个请求在当时的路由情况了。

总之，将大量的读操作从数据库中剥离，让读操作从专用的读数据库中读取数据，大大缓解了数据库的访问压力，也使得读取数据的响应速度得到了大大的提升。那么读写分离有什么弊端吗？是不是所有的场景都适用读写分离这种架构呢？

## **读写分离的弊端**

读写分离给我们带来的好处是很多的，我们对比一下原始的架构和读写分离的架构，从数据流上看，他们的区别是，数据从写入到数据库，到从数据库取出，读写分离的架构多了一个同步的操作。大家想一想，同步操作的时间是多少，延迟如果太大对系统有没有影响，如果同步挂了怎么办？老师举一个亲身经历的案例，那时老师在做个人中心的订单列表页，这个功能挺简单，只需要把订单数据取出来，在页面上展示就可以了。但是在做的时候，订单以及订单相关的数据都是从读库取出的，其中就包括支付状态，这个用户非常敏感的字段。就在某一天的某一个时段，突然接到了用户大量的投诉，说用户已经付了钱了，但是订单的状态还是未支付。我也觉得很奇怪，马上要了一个订单号，去数据库里查询，发现订单状态就是未支付呀，没有问题，过了一会，为了保险起见，我还是去写库再查一下这个订单吧，发现写库的订单状态确实是已支付，这下完了，写库和读取的数据不一致，我马上通知DBA，让他去查数据库，他的反馈是同步挂掉了。

大家看到了吧，这就是读写分离的弊端，当同步挂掉，或者同步延迟比较大时，写库和读库的数据不一致，这个数据的不一致，用户能不能接受，订单支付状态这个不一致当然是不能接受的了，其他的业务场景能不能接受呢？这个要对不同的业务场景做具体的分析了。

## **如何正确的使用读写分离**

一些对数据实时性要求不高的业务场景，可以考虑使用读写分离。但是对数据实时性要求比较高的场景，比如订单支付状态，还是不建议采用读写分离的，或者你在写程序时，老老实实的从写库去读取数据。我也咨询过专门做数据同步的机构，他们给出的建议是，如果你做数据的同步，你的网络延迟应该在5ms以内，这个对网络环境要求是非常高的，大家可以ping一下你网络中的其他机器，看看能不能达到这个标准。如果你的网络环境很好，达到了要求，那么使用读写分离是没有问题的，数据几乎是实时同步到读库，根本感觉不到延迟。

读写分离呢，就给大家介绍到这，大家在使用的时候，还是要从业务出发，看看你的业务是否适合使用读写分离，每种技术架构都有自己的优缺点，没有好不好，只有适合不适合。只有适合业务的架构才是好的架构。