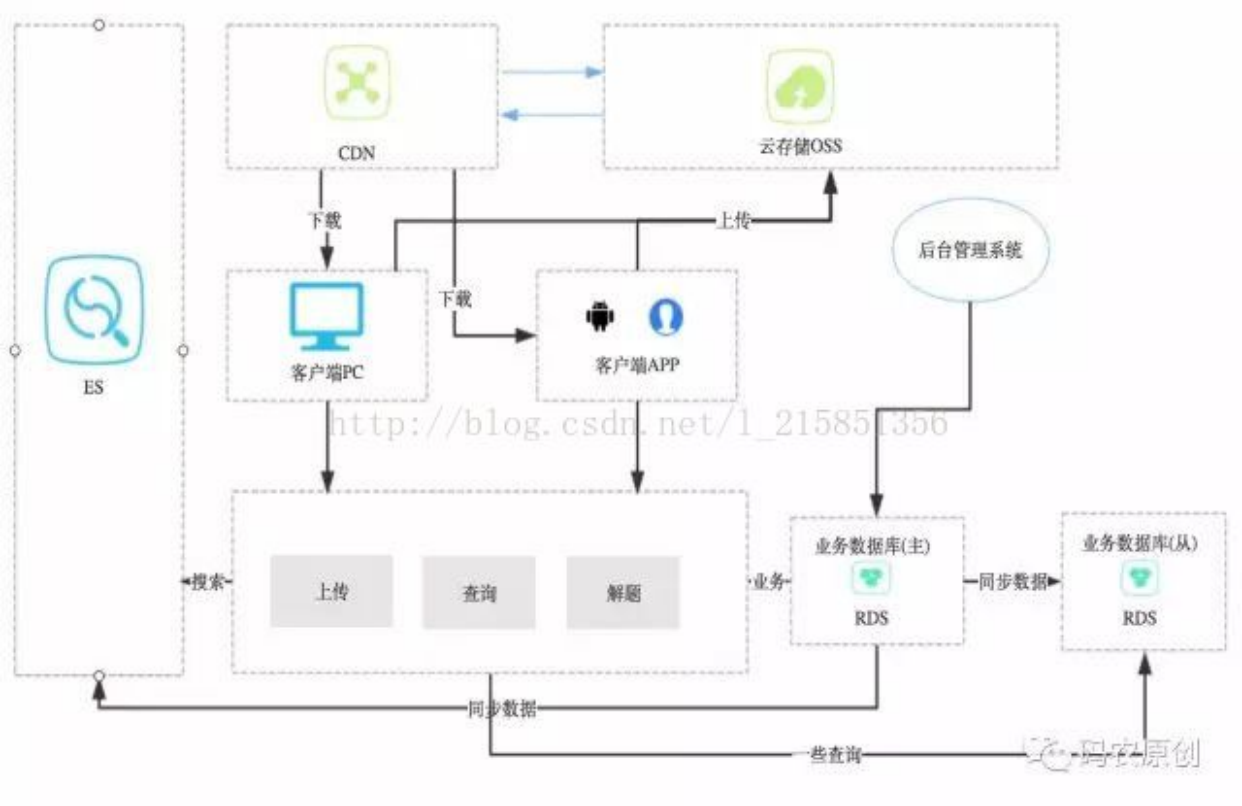
这种模式主要解决单机数据库压力过大，从而导致业务缓慢甚至超时，查询响应时间变长的问题，也包括需要大量数据库服务器计算资源的查询请求。这个可以说是单库单应用模式的升级版本，也是技术架构迭代演进过程中的必经之路。
这种模式的一般设计见下图：



如上图所示，这种模式较单库单应用模式与内容分发模式多了几个部分，一个是业务数据库的主从分离，一个是引入了ES，为什么要这样？都解决了哪些痛点，下面具体结合业务需求场景进行叙述。

场景一：全文关键词检索

我想这个需求，绝大多数应用都会有，如果使用传统的数据库技术，大部分可能都会使用like这种SQL语句，高级一点可能是先分词，然后通过分词index相关的记录。SQL语句的性能问题与全表扫描机制导致了非常严重的性能问题，现在基本上很少见到。

这里的ES是ElasticSearch的缩写，是一种查询引擎，类似的还有Solr等，都差不多的技术，ES较Solr配置简单、使用方便，所以这里选用了它。另外，ES支持横向扩展，理论上没有性能的瓶颈。同时，还支持各种插件、自定义分词器等，可扩展性较强。在这里，使用ES不仅可以替代数据库完成全文检索功能，还可以实现诸如分页、排序、分组、分面等功能。具体的，请同学们自行学**之。那怎么使用呢？一个一般的流程是这样的：

1. 服务端把一条业务数据落库
2. 服务端异步把该条数据发送到ES
3. ES把该条记录按照规则、配置放入自己的索引库
4. 客户端查询的时候，由服务端把这个请求发送到ES，得到数据后，根据需求拼装、组合数据，返回给客户端

实际中怎么用，还请同学们根据实际情况做组合、取舍。

场景二：大量的普通查询

这个场景是指我们的业务中的大部分辅助性的查询，如：取钱的时候先查询一下余额，根据用户的ID查询用户的记录，取得该用户最新的一条取钱记录等。我们肯定是要天天要用的，而且用的还非常多。同时呢，我们的写入请求也是非常多的，导致大量的写入、查询操作压向同一数据库，然后，数据库挂了，系统挂了，领导生气了，被开除了，还不起房贷了，露宿街头了，老婆跟别人跑了，.....

不敢想，所以要求我们必须分散数据库的压力，一个业界较成熟的方案就是数据库的读写分离，写的时候入主库，读的时候读从库。这样就把压力分散到不同的数据库了，如果一个读库性能不行，扛不住的话，可以一主多从，横向扩展。可谓是一剂良药啊！那怎么使用呢？一个一般的流程是这样的：

1. 服务端把一条业务数据落库
2. 数据库同步或异步或半同步把该条数据复制到从库
3. 服务端读数据的时候直接去从库读相应的数据

比较简单吧，一些聪明的、爱思考的、上进的同学可能发现问题了，也包括上面介绍的场景一，就是延迟问题，如：数据还没有到从库，我就马上读，那么是读不到的，会发生问题的。

对于这个问题，各家公司解决的思路不一样，方法不尽相同。一个普遍的解决方案是：读不到就读主库，当然这么说也是有前提条件的，但具体的方案这里就不一一展开了，我可能会

在接下来的分享中详解各种方案。

另外，关于数据库的复制模式，还请同学们自行学**，太多了，这里说不清。该总结一下这种模式的优缺点的了，如下：

优点：减少数据库的压力，理论上提供无限高的读性能，间接提高业务(写)的性能，专用的查询、索引、全文(分词)解决方案。
缺点：数据延迟，数据一致性的保证。