**幂等性**原本是数学上的概念，即使公式：f(x)=f(f(x)) 能够成立的数学性质。用在编程领域，则意为对同一个系统，使用同样的条件，一次请求和重复的多次请求对系统资源的影响是一致的。

**幂等性适用领域**

试想这样的一种场景：在电商平台上支付后，因为网络原因导致系统提示你支付失败，于是你又重新付款了一次，等完成后检查网银发现被系统扣了两次款，这是一种什么样的体验？

## 幂等的常用思路

#### 1. MVCC：####

多版本并发控制，乐观锁的一种实现，在数据更新时需要去比较持有数据的版本号，版本号不一致的操作无法成功。例如博客点赞次数自动+1的接口：

public boolean addCount(Long id, Long version);

update blogTable set count= count+1,version=version+1 where id=321 and version=123

每一个version只有一次执行成功的机会，一旦失败必须重新获取。

#### 2. 去重表：####

利用数据库表单的特性来实现幂等，常用的一个思路是在表上构建唯一性索引，保证某一类数据一旦执行完毕，后续同样的请求再也无法成功写入。

例子还是上述的博客点赞问题，要想防止一个人重复点赞，可以设计一张表，将博客id与用户id绑定建立唯一索引，每当用户点赞时就往表中写入一条数据，这样重复点赞的数据就无法写入。

#### 3. TOKEN机制：####

这种机制就比较重要了，适用范围较广，有多种不同的实现方式。其核心思想是为每一次操作生成一个唯一性的凭证，也就是token。一个token在操作的每一个阶段只有一次执行权，一旦执行成功则保存执行结果。对重复的请求，返回同一个结果。

以电商平台为例子，电商平台上的订单id就是最适合的token。当用户下单时，会经历多个环节，比如生成订单，减库存，减优惠券等等。

每一个环节执行时都先检测一下该订单id是否已经执行过这一步骤，对未执行的请求，执行操作并缓存结果，而对已经执行过的id，则直接返回之前的执行结果，不做任何操作。这样可以在最大程度上避免操作的重复执行问题，缓存起来的执行结果也能用于事务的控制等。

作者：GameKing  
链接：https://www.jianshu.com/p/475589f5cd7b  
來源：简书  
简书著作权归作者所有，任何形式的转载都请联系作者获得授权并注明出处。