nginx+lua+redis限流实战

先跑通基本环境，再实现具体业务。

基本环境准备

nginx配置文件

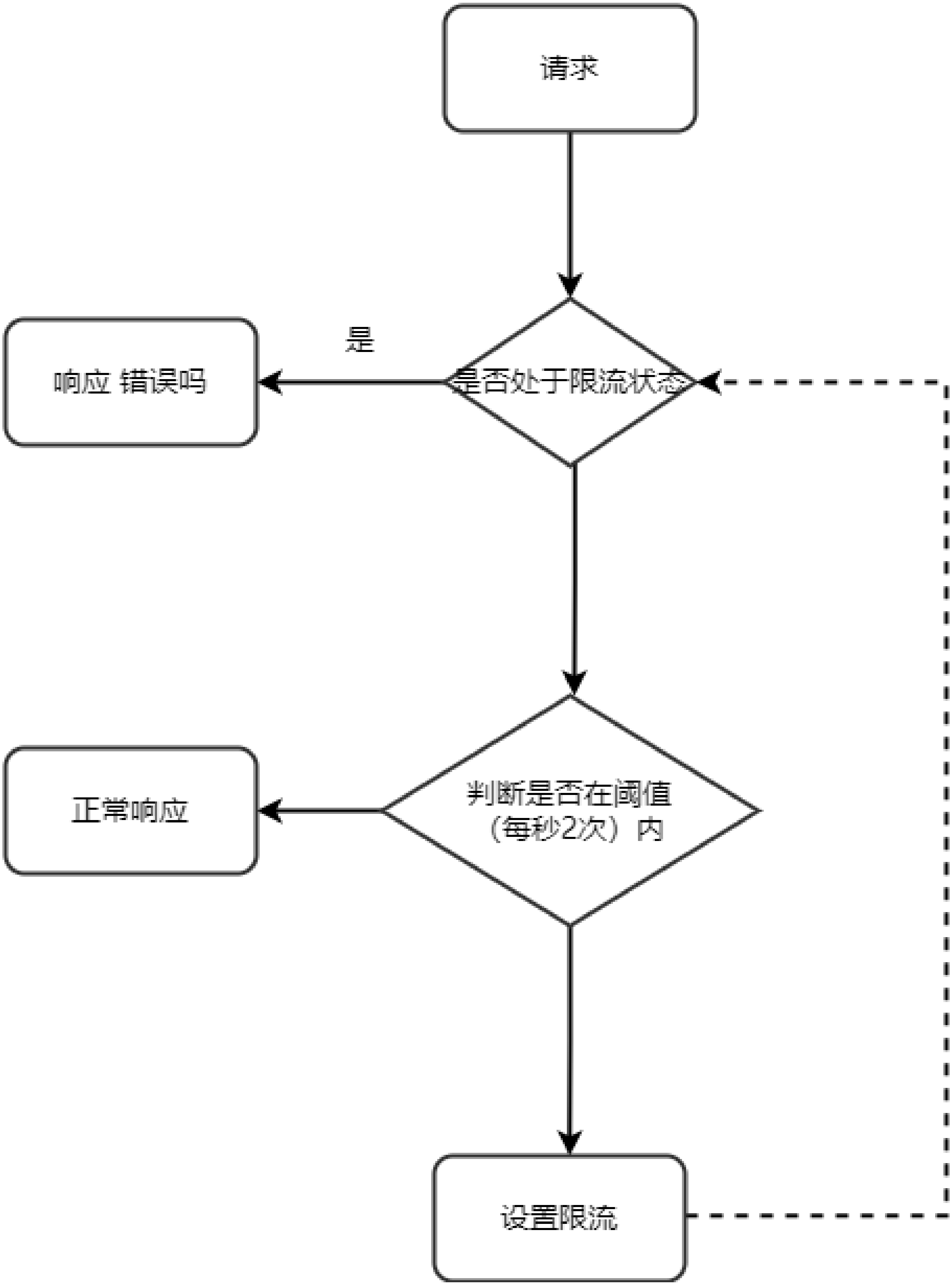
```
[root@localhost conf]# vi nginx-ip-limit.conf

worker_processes 1;
error_log logs/error.log debug;

events {
        worker_connections 1024;
}

http {
        include mime.types;
        default_type application/octet-stream;
        server {
                listen 8083;
                location / {
                        default_type text/html;
                        access_by_lua_file /usr/local/openresty/nginx/lua/ip-limit-access.lua;
                        log_by_lua_file /usr/local/openresty/nginx/lua/ip-limit-log.lua;
                        proxy_pass http://localhost:8080/;
                }
        }
}
```

两个lua文件

```
[root@localhost lua]# cat ip-limit-access.lua
ngx.log(ngx.INFO,"ip limit access");
[root@localhost lua]# cat ip-limit-log.lua
ngx.log(ngx.INFO,"ip limit log");
[root@localhost lua]#
```

限流业务

需求：系统每秒限流2个请求，如果超过 阈值（每秒2个请求），则系统限制10秒内，不能被访问。


请求
是
响应 错误吗
是否处于限流状态
正常响应
判断是否在阈值
（每秒2次）内
设置限流

lua业务代码

```
[root@localhost lua]# cat ip-limit-access.lua
ngx.log(ngx.INFO,"ip limit access");

local redis = require "resty.redis";
local red = redis:new();

--链接redis
red:connect("127.0.0.1",6379);
-- 需要写链接成功的判断。

--判断是否限流
limit = red:get("limit");
if limit == '1' then
        return ngx.exit(503);
end

inc = red:incr("testLimit");
if inc <= 2 then
        red:expire("testLimit",1);
else
        red:set("limit",1);
        red:expire("limit",10);
end


[root@localhost lua]#
```