**1、(必做)**分别用 100 个字以上的一段话，加上一幅图 (架构图或脑图)，总结自己

对下列技术的关键点思考和经验认识:

1)JVM

2)NIO

3) 并发编程

4)Spring 和 ORM 等框架

5)MySQL 数据库和 SQL

6) 分库分表

7)RPC 和微服务

8) 分布式缓存

9) 分布式消息队列

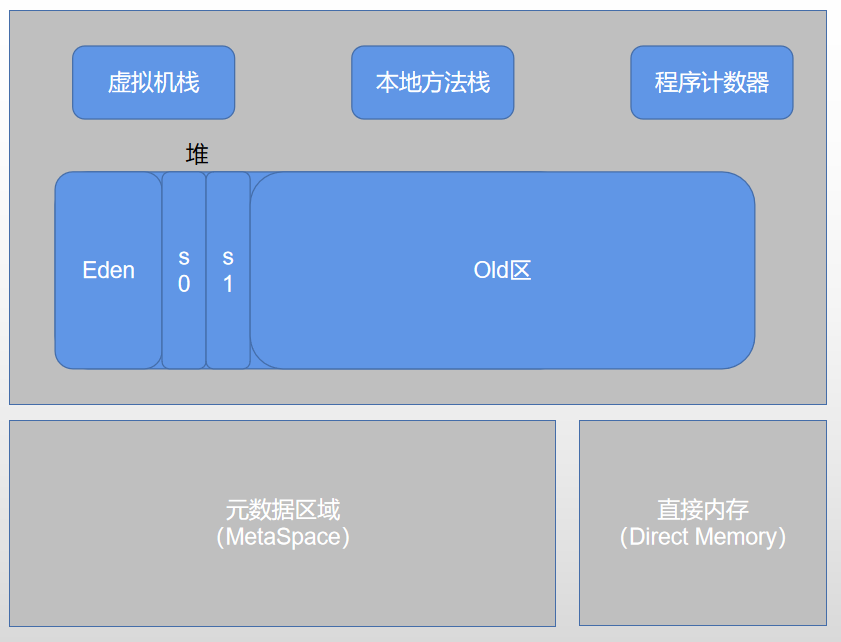
**1)JVM**

jvm分为虚拟机栈、本地方法栈、程序计数器、堆、元数据区域、直接内存；

其中最主要的是堆内存，堆内存中又分 年轻代和老年代，年轻代又分为伊利园区、幸存者区1、幸存者区2，

新创建的对象会先存放在年轻代，对象经过GC的一些垃圾回收算法会收集一部分对象，没有被回收的对象会进入到老年代old区；对象的年龄是有GC的次数决定的；如果对象的很大，新生代无法容纳该对象，那么该对象也会进入到老年代。

老年代的对象通过fullGC 来进行回收对象，如果在执行fullGC之后，还是没有办法给对象分配内存，那么久会抛出OOM的异常。



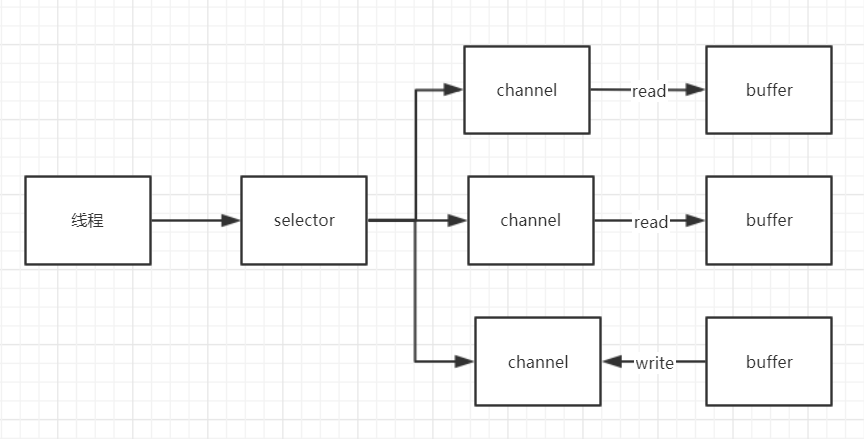
**2)NIO**

NIO主要由三部分组成，channel(通道)、Buffer（缓冲区）、Selector。传统的IO是基于字节流和字符流进行操作，而NIO是基于通道和缓冲区进行操作的。selector用于监听多个多个通道的事件。

channel 主要有FileChannel、DatagramChannel、SocketChannel、ServerSocketChannel

Buffer 用来channel交互，数据从channel读到buffer中，从buffer写到channel中。

selector是一个组件，他可以检查一个或者多个channel实例，并决定哪些channel已经准备好读或者写。通过这种方式一个线程可以管理多个通道，从而事件管理过个网络连接。



**3) 并发编程**

并发编程主要只是点包括

线程之间的通讯：1、Object方法 wait、notify、notifyAll 2、Thread方法 sleep、yield、join

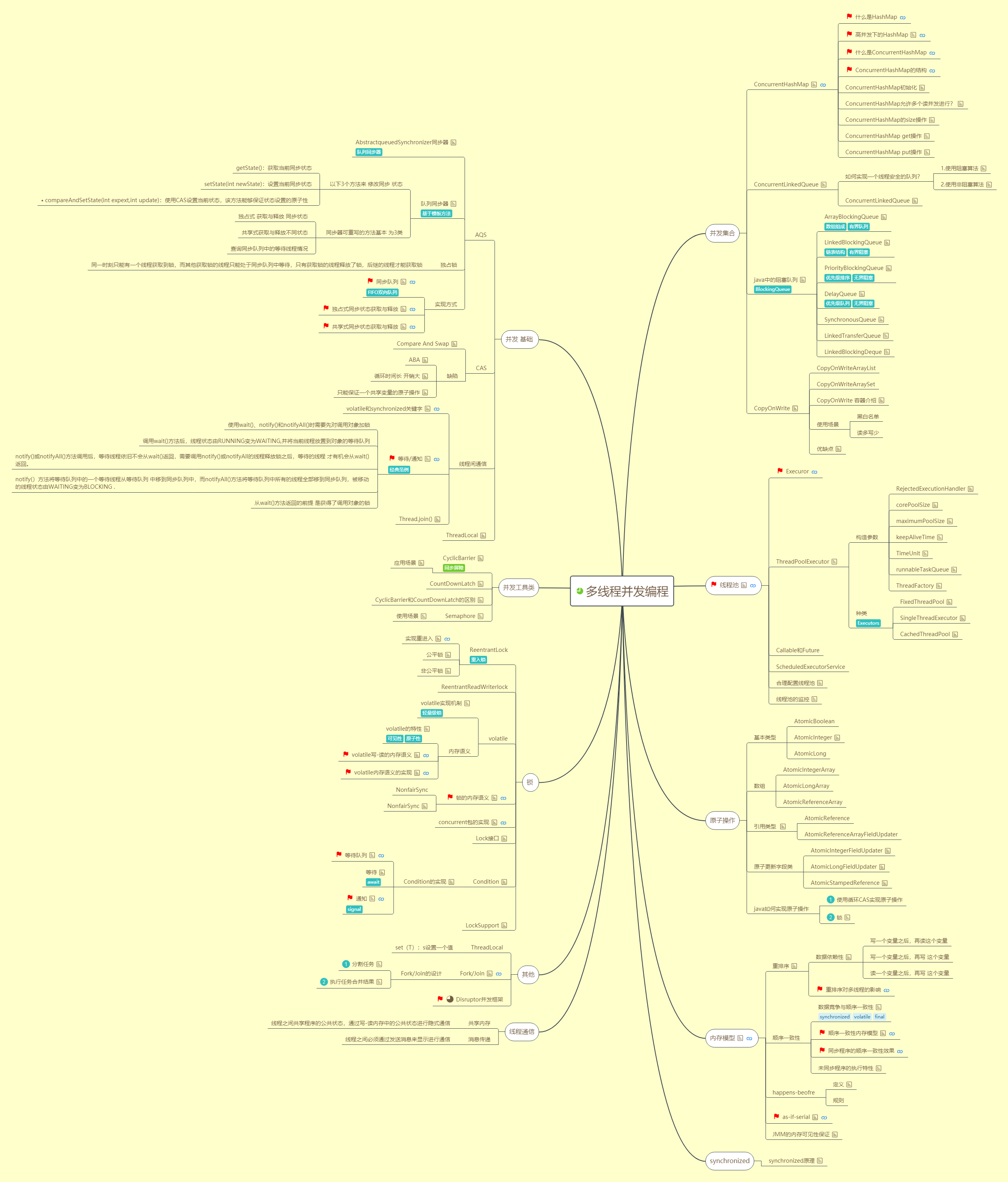
valitile：线程访问共享变量。如果一个字段被声明为volatile，java线程内存模型确保所有的线程看到的这个变量的值都是一样的。

线程池：线程池通过复用可以大大减少线程频繁的创建和销毁带来的性能上的损耗。

ThreadLocal:线程本地变量，每一个线程都有一个副本

Future：要想拿到异步任务的返回值，使用future

CountDownLatch和CyclicBarrier:CountDownLatch 在主线程里await阻塞并做聚合，不可以复用；CyclicBarrier 直接在子线程做await阻塞，回调聚合，可以复用



**4)Spring 和 ORM 等框架**

Spring

AOP 切面编程

IOC 控制翻转

spring Bean生命周期(创建对象、属性赋值、初始化、注销接口注册)

Springboot

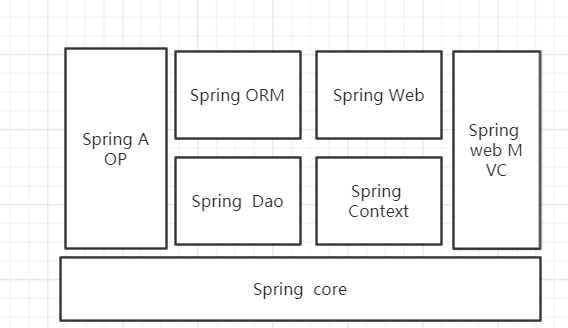
简化配置

ORM

mybatis 和hibernate

mybatis 使用原生的sql，对dba比较友好；但比较繁琐，可以使用插件生成

hibernate 简单场景不用写sql，对dba不友好



**5)MySQL 数据库和 SQL**

**Mysql**

mysql事务(表级锁 和行级锁)，默认事务 可以重复读

mysql主从复制

mysql主库写binlog，从库读relaylog

mysql读写分离

ShardingSphere-jdbc的Master-Slave 功能可以自动进行sql解析和事务管理，自动实现读写分离

mysql高可用

1、多个实例不在同一个机架

2、跨机房区域部署

3、两地三中心容灾高可用方案

**sql优化**

写入优化

模糊查询

连接查询

索引失效问题



**6) 分库分表**

垂直拆分：

垂直拆分（拆库）：将一个数据库，拆分成多个提供不通业务数据处理能力的数据库。

垂直拆分(拆表)：单表数据量大，需要对单表拆表。

水平拆分

按住键分库分表

按时间分库分表

按条件制定分库分表

中间件

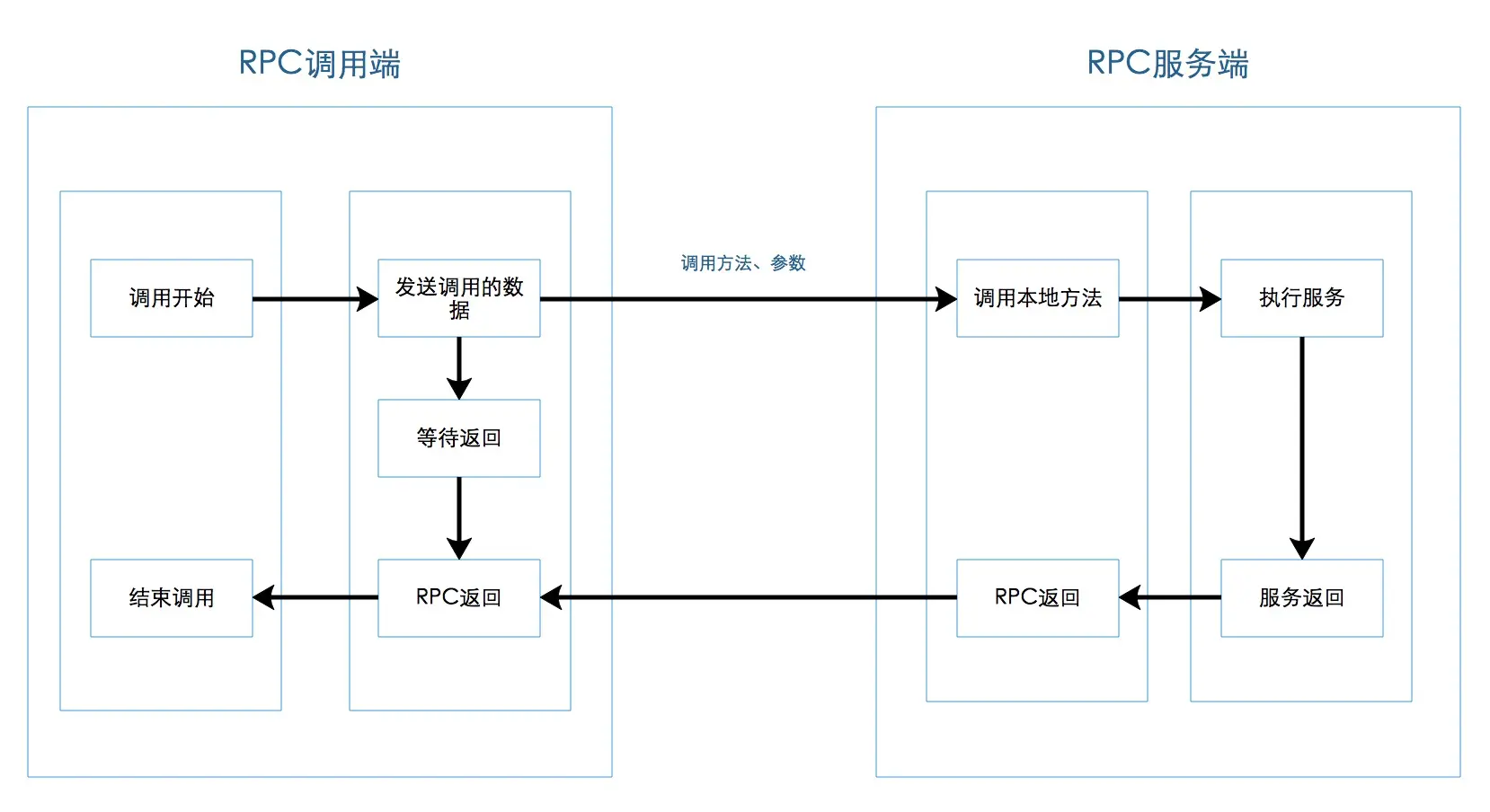
java：TDDL、Apache ShardingSphere-jdbc

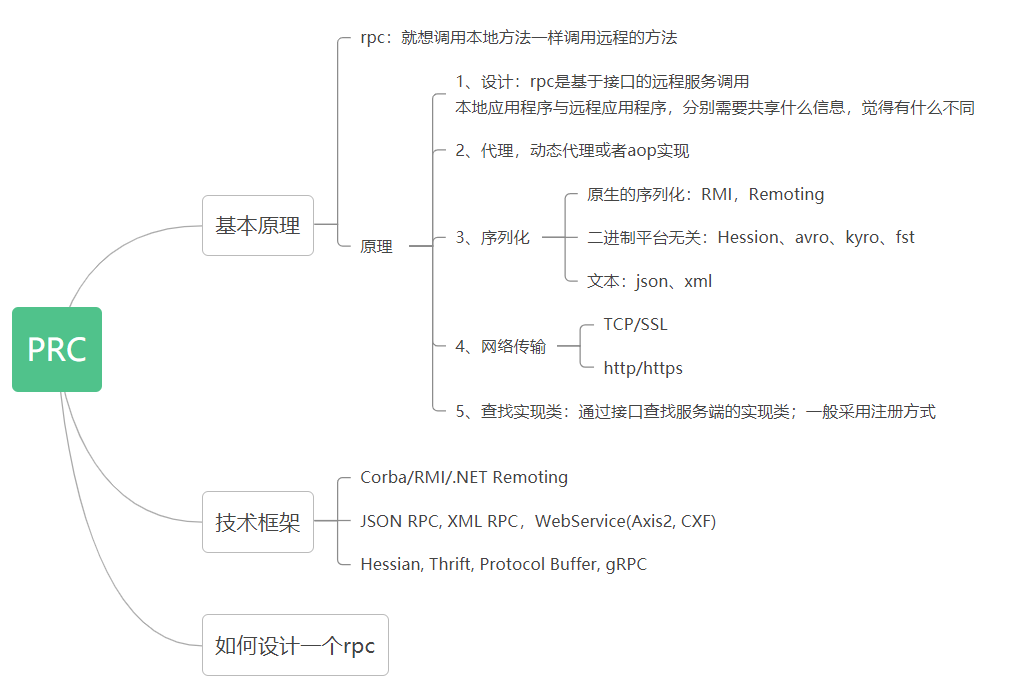
中间件：mycat、vitness、kingshard



**7)RPC 和微服务**

rpc 是远程过程调用，通过RPC我们可以像调用本地方法一样调用别的机器上的方法，用户将无感服务器与服务器之间的通讯，RPC在微服务当中起到相当大的作用，当然RPC不是微服务必须的一种方式，有别的方式也可以实现这种远程调用例如RESTful API就可以实现远程调用。



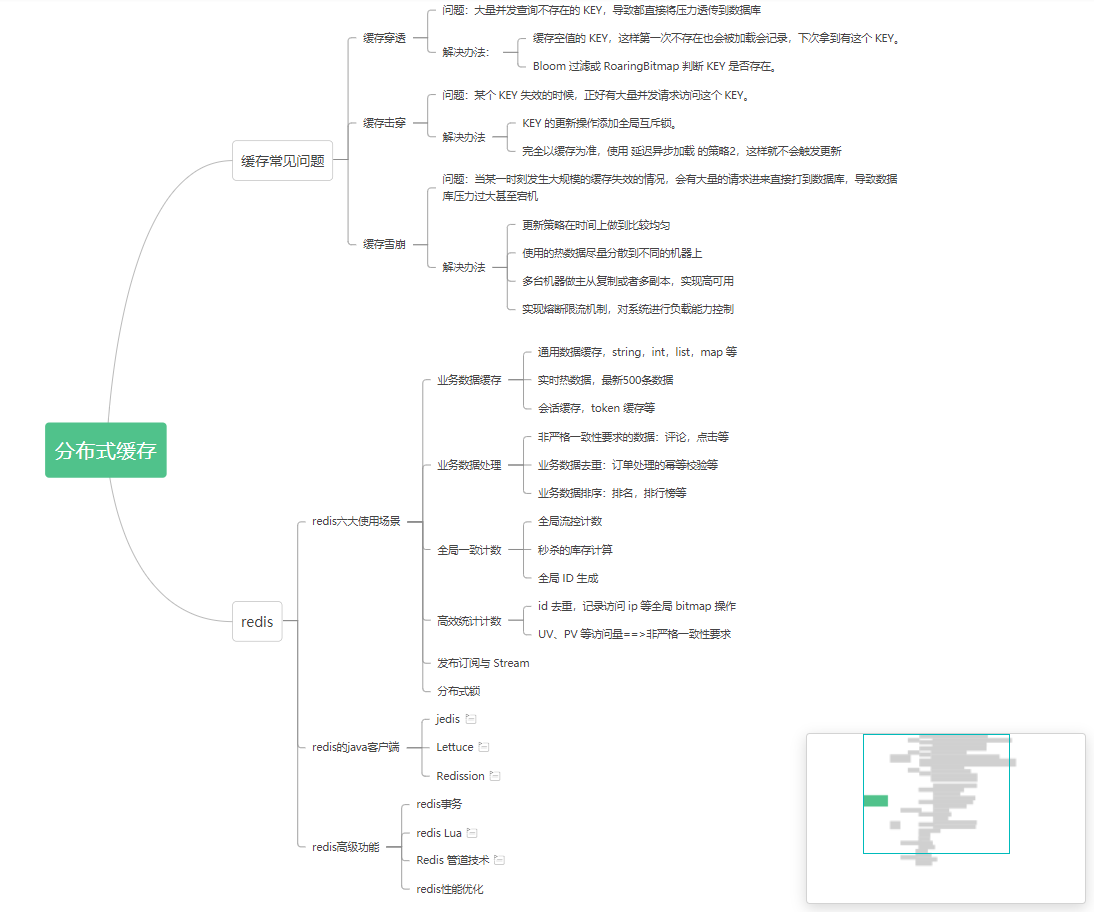




**8) 分布式缓存**

随着系统负载原来越大，单台reids已无法满足系统负载、安全等要求。redis 分布式缓存可以加大服务吞吐量，支持动态扩展reids集群，支持自定义每台reids服务器负载、支持宕机请求自动转移。

Redis 是一种流行的开源内存数据存储，可用作数据库、缓存或消息代理。由于是从内存而非磁盘加载数据，Redis 比许多传统的数据库解决方案更快。



**9) 分布式消息队列**

**Kafka的特性:**

高吞吐量、低延迟：kafka每秒可以处理几十万条消息，它的延迟最低只有几毫秒，每个topic可以分多个partition, consumer group 对partition进行consume操作。

可扩展性：kafka集群支持热扩展

持久性、可靠性：消息被持久化到本地磁盘，并且支持数据备份防止数据丢失

容错性：允许集群中节点失败（若副本数量为n,则允许n-1个节点失败）

高并发：支持数千个客户端同时读写

