Redis环境搭建（19~23）

> 安装Redis 5系列
>
> 在Linux中进行安装
>
> Redis官方是不支持Windows 版本的~~微软维护了一个Windows 版本的Redis 分支.

一、 软件架构与系统评价基础

- **应用 (Application) / 系统 (System)**：指代一个或一组服务器程序。
- **模块 (Module) / 组件 (Component)**：指代应用内部细分的多个功能，每个独立的功能即可称为模块/组件。
- **分布式 (Distributed)**：侧重于**物理层面**，引入多个主机/服务器协同完成工作。
- **集群 (Cluster)**：侧重于**逻辑层面**，同样引入多个主机协同工作。
- **主 (Master) / 从 (Slave)**：典型的分布式结构，一个节点为主，其余为从，从节点的数据从主节点同步而来。
- **中间件 (Middleware)**：指与业务无关的通用功能服务，如数据库、缓存、消息队列等。
- **可用性 (Availability)**：系统的第一要务，指系统整体可用时间占总时间的比例。
  - 计算方式：$可用性 = \frac{系统整体可用时间}{总时间}$。
  - 常见标准：4个9即 **99.99%**，5个9即 **99.999%**。
- **响应时长 (Response Time RT)**：衡量服务器性能，数值越小越好，与具体业务密切相关。
- **吞吐 (Throughput) vs 并发 (Concurrent)**：衡量系统处理请求的能力，是评价性能的一种方式。

二、 分布式系统演化过程小结

阐述了系统为了支撑更多硬件资源而进行的七个演进阶段：

1. **单机架构**：应用程序与数据库服务器部署在同一机器。
2. **数据库和应用分离**：分别部署到不同主机。
3. **引入负载均衡**：应用服务器形成集群，通过负载均衡器分发请求。即便某个主机宕机，其他主机仍可承担服务，提高了**可用性**。
4. **引入读写分离**：数据库采用主从结构，主节点负责**写**，从节点负责**读**并同步主节点数据。

5. **引入缓存（冷热数据分离）**：基于“二八原则”提升处理能力。Redis 常在此阶段扮演缓存角色，但也带来了**数据库和缓存数据一致性**的问题。
6. **引入分库分表**：进一步扩展数据库存储空间。
7. **引入微服务**：按业务功能拆分应用服务器，使其功能更单一、更简单。

核心逻辑：真实的演化过程与业务发展密切相关，业务更重要，技术只是支持。

三、 Redis 的特性与技术优势

- **核心特性**：
  - **内存数据结构**：非关系型键值对存储。Key 永远是 String，Value 支持 String, Hash, List, Set, Sorted Set, Stream 等。
  - **可编程性**：支持简单命令及 **Lua 脚本**批量执行逻辑。
  - **可扩展性**：提供 API 支持使用 C, C++, Rust 编写扩展（如动态链接库 .so 或 .dll）。
  - **持久化**：内存为主，磁盘为辅。重启后可加载磁盘备份恢复数据。
  - **集群 (Clustering)**：支持基于哈希的分片，实现水平扩展，突破单机内存限制。
  - **高可用**：支持主从结构和自动故障转移。
- **Redis 为什么快？**：
  a. **内存存储**：远快于硬盘访问。
  b. **逻辑简单**：操作内存数据结构的逻辑不复杂。
  c. **IO 多路复用**：网络层使用 **epoll** 方式，单线程管理多个 socket。
  d. **单线程模型**：减少线程竞争开销。Redis 核心任务是内存操作，不吃 CPU，故无需多线程充分利用多核。
  e. **C 语言开发**：虽有此说法，但 MySQL 也是 C 开发，因此这并非决定性原因。

四、 Redis 应用场景与限制

- **应用场景**：
  - **实时数据存储 (Database)**：用于对性能要求极高的全量数据存储（如广告搜索），需大量硬件资源。
  - **缓存 (Caching)**：拎出热点数据，数据丢失可从 MySQL 加载。
  - **分布式会话 (Session Storage)**：将 session 集中存在 Redis，解决负载均衡后的会话丢失问题，应用重启会话不丢失。
  - **消息队列 (Streaming)**：实现网络版生产者-消费者模型，优势在于解耦和削峰填谷。

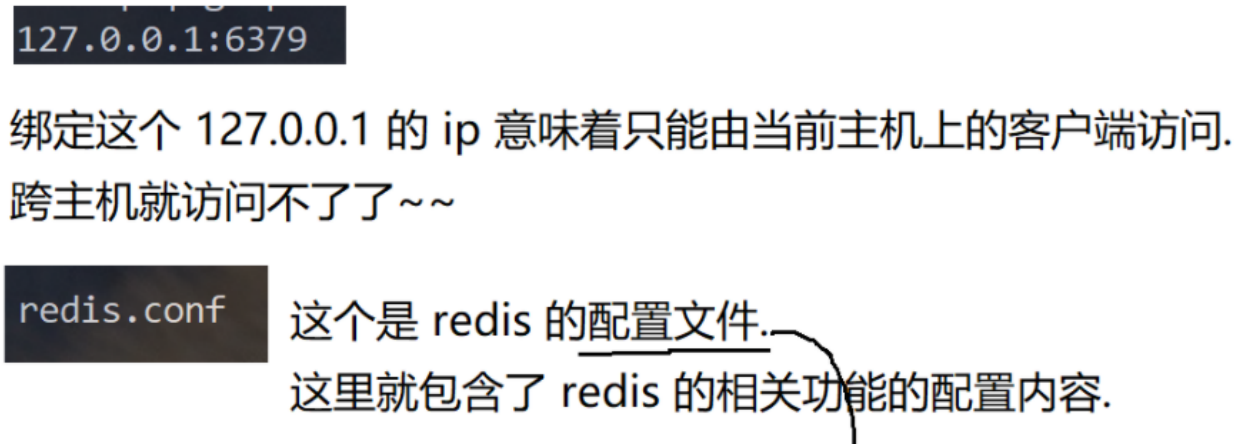
- **局限性**：Redis 不适合存储超大规模的数据。
- **学习总结**：要通过业务理解分布式系统，技术要通过博客、高质量作业和项目理解透彻。

五、 安装与环境配置

- **系统支持**：官方不支持 Windows，建议在 Linux（Ubuntu, Centos）或 Docker 中安装。
- **Ubuntu 安装步骤**：
  - 切换 root 用户。
  - 使用apt命令来搜索redis相关的软件包. `apt search redis`

    > redis是否要配置密码??
    >
    > 不要!!!
    >
    > 虽然没有密码，但是非常安全!!
    >
    > 只要咱们的数据不值钱，就是非常安全的!!

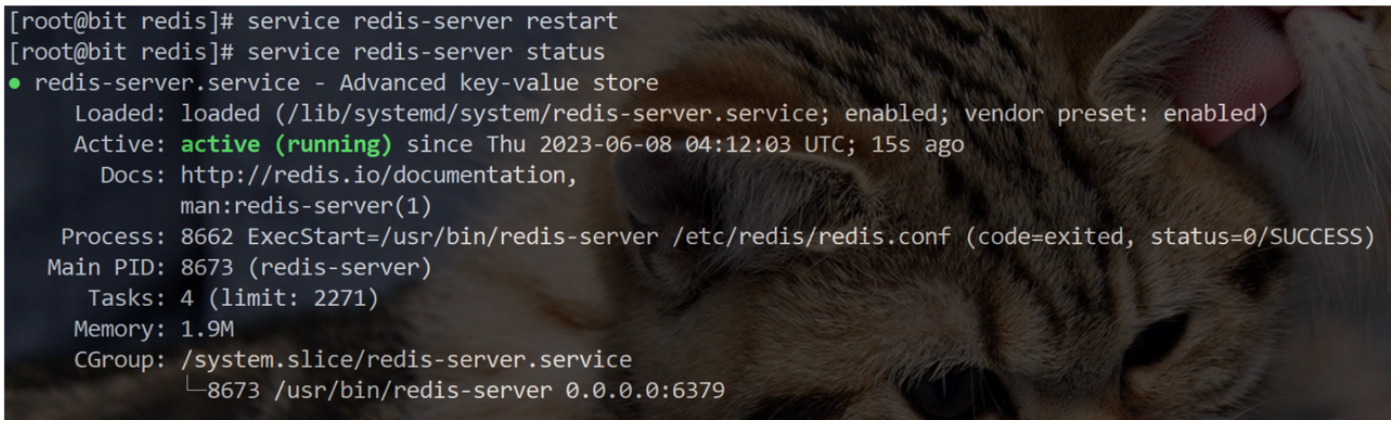
  - `apt install redis` 安装软件。
  - 修改配置文件 `redis.conf` （如修改 `bind 0.0.0.0` 及关闭保护模式）。



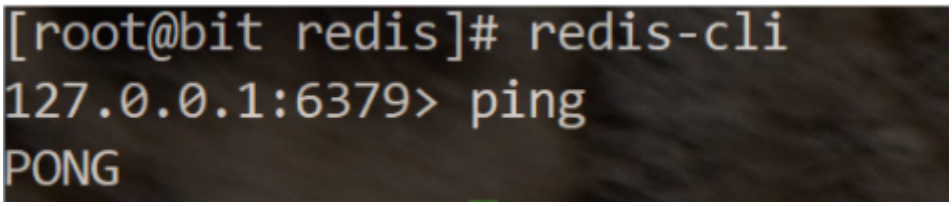
> 啥叫配置文件??
>
> 很多软件都是有配置文件的.
>
> 一个大的软件，里面包含很多的功能，有很多可以定制化的操作.就可以通过配置文件选择开启/关闭/设定某些功能

  - 重启服务并使用 `redis-cli` 测试连接。

```
service redis-server restart
```



- 使用redis自带的客户端来连接服务器.

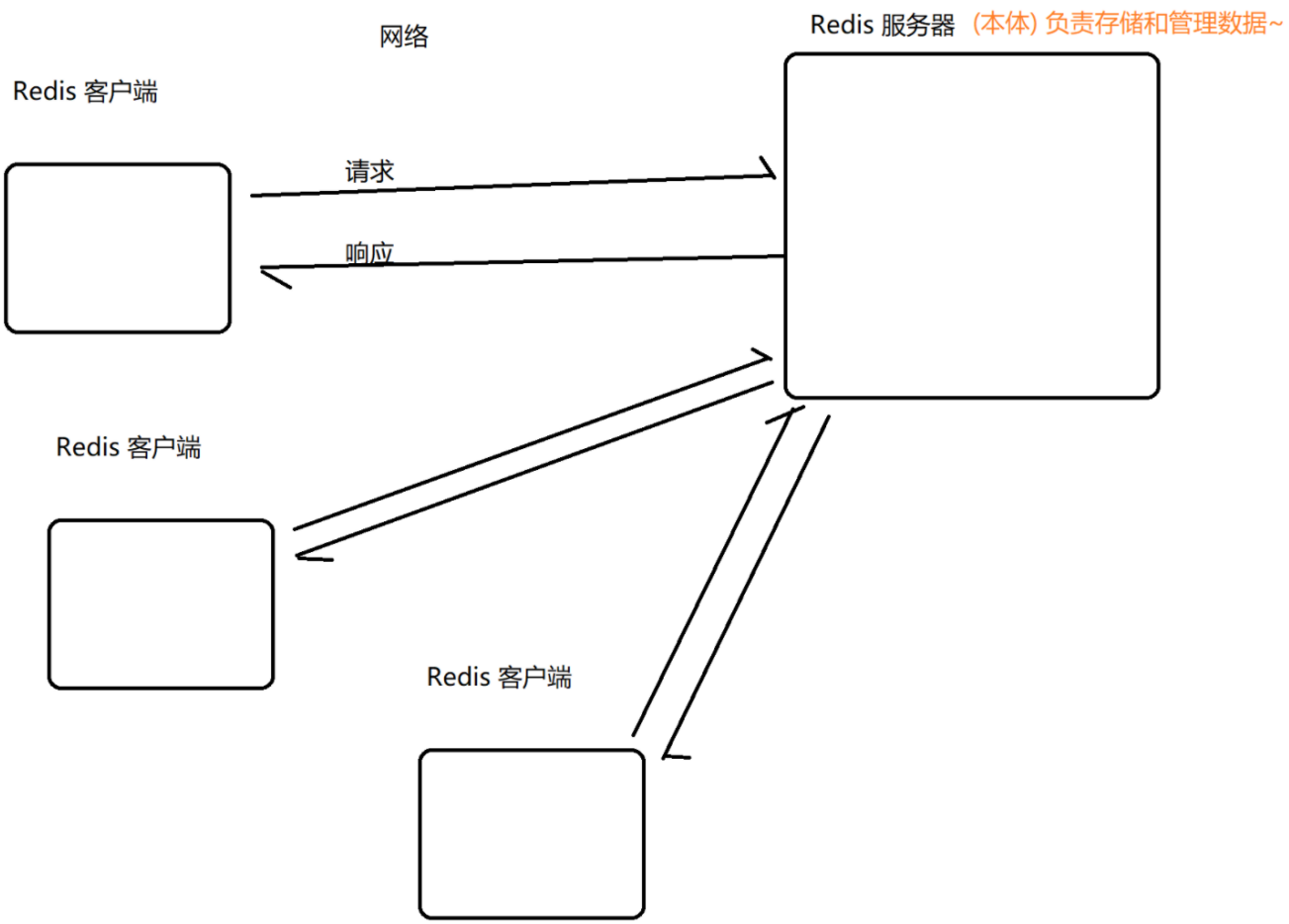


- **Centos 安装要点**：
  - Centos 8 默认版本为 5，Centos 7 为 3（较老，需 scl 源）。
  - **后台进程**：服务器程序通常以“守护进程”方式运行，不受终端关闭影响。
  - **启动/停止**：使用命令启动并指定配置文件；停止时需查询 `pid` 后进行 `kill`。

六、Redis客户端介绍

> redis也是一个客户端-服务器结构的程序

redis 客户端和服务器可以在同一个主机上，也可以在不同主机上.(当前阶段，我们一般只有一台机器，此时客户端和服务器就是在同一个机器上)



Redis的客户端也有很多种形态.

1. **自带了命令行客户端**

`redis-cli`

```
[root@bit ~]# redis-cli
127.0.0.1:6379>
[root@bit ~]# redis-cli -h 127.0.0.1 -p 6379
127.0.0.1:6379>
```

2. **图形化界面的客户端**

> (桌面程序,web程序)

像这样的图形化程序，依赖 windows 系统.而未来在实际工作中，你用来办公的windows系统，连接到服务器可能会有诸多限制，你的 windows 上的图形化界面客户端能不能连上你们的服务器里的redis，是个未知数(和mysql同理)

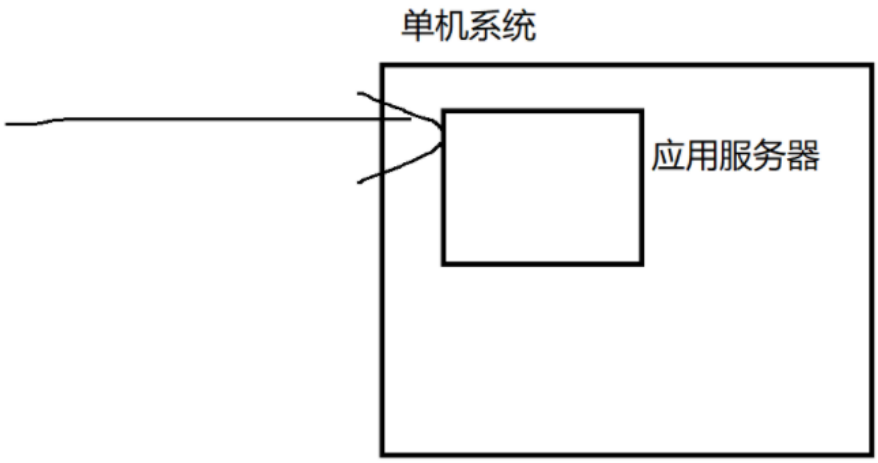
中间可能会经历很多的跳板机，堡垒机，权限校验

3. **基于redis的api自行开发客户端[工作中最主要的形态]**

    非常类似于mysql的C语言API和JDBC

Redis的使用形态

> 咱们谈到的redis的快，是相对于mysql这样的关系型数据库的
>
> 但是如果是直接和内存中的操作变量相比，就没有优势了，甚至更慢了!!



> 比如，应用程序要存储一些数据. 存储一下用户点赞数.
>
> 视频id，点赞个数键值对格式来存储~~
>
> 那么是用一个redis来存，还是直接在内存中搞个hash map来存呢??
>
> 使用hash map是直接操作内存.使用redis是先通过网络!!再操作内存

要对症下药，具体问题具体分析

上述场景中，是否要使用redis?

要结合实际的需求来确定!!!

引入redis的缺点，会更慢但是有了redis之后，就可以把数据单独存储.后续应用服务器重启，不会影响到数据内容

未来要扩展成分布式系统，使用redis是更佳的

是否引入一个技术，一定要想清楚来龙去脉，想清楚能够解决啥问题，引入了啥问题~~