大型网站的系统特点

- 高并发、大流量

大型网站系统需要面对高并发用户，大流量访问。Google日均PV数35亿，日均IP访问数3亿；腾讯QQ的最大在线用户数1.4亿（2011年数据）；微信用户量已超11亿；2019年天猫双十一交易额突破2500亿。

- 高可用

系统7×24小时不间断服务。大型互联网站的宕机事件通常会成为新闻焦点，例如2010年百度域名被黑客劫持导致不能访问，成为重大新闻热点。

- 海量数据

需要存储、管理海量数据，需要使用大量服务器。Facebook每周上传的照片数目接近10亿，百度收录的网页数目有数百亿，Google有近百万台服务器为全球用户提供服务。

- 用户分布广泛，网络情况复杂

许多大型互联网都是为全球用户提供服务的，用户分布范围广，各地网络情况千差万别。在国内，还有各个运营商网络互通难的问题。而中美光缆的数次故障，也让一些对国外用户依赖较大的网站不得不考虑在海外建立数据中心。

- 安全环境恶劣

由于互联网的开放性，使得互联网站更容易受到攻击，大型网站几 乎每天都会被黑客攻击。任何系统漏洞都会被攻击者利用，造成重大损失。

- 需求快速变更，发布频繁

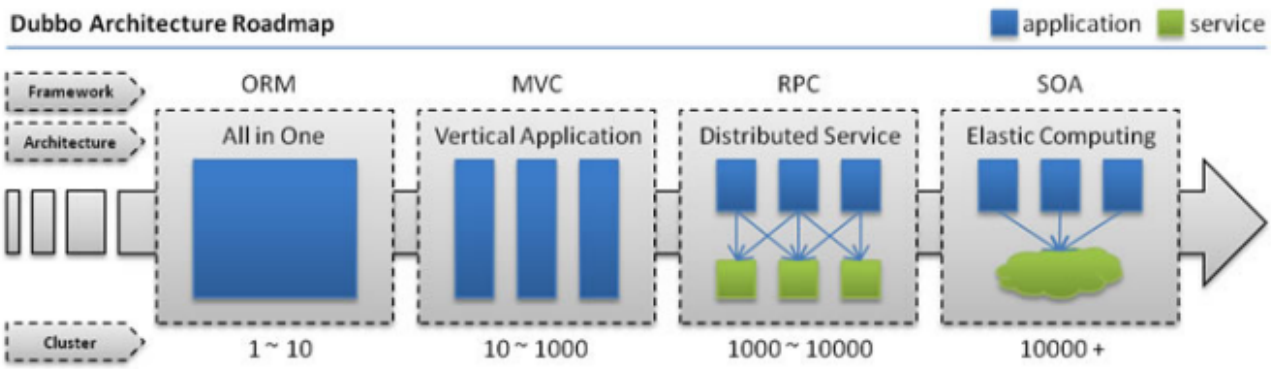
和传统软件的版本发布频率不同，互联网产品为快速适应市场，满足用户需求，其产品发布频率是极高的。Office的产品版本以年为单位发布，而一般大型网站的产品每周都有新版本发布上线，至于中小型网站的发布就更频繁了，有时候一天会发布几十次。

- 渐进式发展

与传统软件产品或企业应用系统一开始就规划好全部的功能和非功能需求不同，几乎所有的大型互联网站都是从一个小网站开始，渐进地发展起来的。

Facebook是伯克扎克同学在哈佛大学的宿舍里开发的；Google的第一台服务器部署在斯坦福大学的实验室里； 阿里巴巴则是在马云家的客厅里诞生的。好的互联网产品都是慢慢运营出来的，不是一开始就开发好的，这也正好与网站架构的发展演化过程对应。

大型网站架构发展历程



- 单一应用架构

当网站流量很小时，只需一个应用，将所有功能都部署在一起，以减少部署节点和成本。此时，用于简化增删改查工作量的数据访问框架(ORM)是关键。

- 垂直应用架构

当访问量逐渐增大，单一应用增加机器带来的加速度越来越小，提升效率的方法之一是将应用拆成互不相干的几个应用，以提升效率。此时，用于加速前端页面开发的Web框架(MVC)是关键。

- 分布式服务架构

当垂直应用越来越多，应用之间交互不可避免，将核心业务抽取出来，作为独立的服务，逐渐形成稳定的服务中心，使前端应用能更快速的响应多变的市场需求。此时，用于提高业务复用及整合的分布式服务框架(RPC)是关键。

从NoSQL说起

NoSQL是Not only SQL的缩写，大意为“不只是SQL”，说明这项技术是**传统关系型数据库的补充**而非替代。在整个NoSQL技术栈中**MemCache**、**Redis**、**MongoDB**被称为NoSQL三剑客。那么时代为什么需要NoSQL数据库呢？我们来做个对比：

	关系型数据库	**NoSQL数据库**
数据存储位置	硬盘	内存
数据结构	高度组织化结构化数据	没有预定义的模式
数据操作方式	SQL	所有数据都是键值对，没有声明性查询语言
事务控制	严格的基础事务ACID原则	基于乐观锁的松散事务控制
访问控制	细粒度的用户访问权限控制	简单的基于IP绑定或密码的访问控制
外键	支持	不支持
索引	支持	不支持

所以NoSQL数据库的最大优势体现为：高性能、高可用性和可伸缩性。

Redis简介

非关系型数据库 Redis 也叫缓存数据库 也是 内存数据库

- 基本信息

```
Redis是一个开源（BSD许可）的，内存中的数据结构存储系统，它可以用作数据库、缓存和消息中间件。 它支持多种类型的数据结构，如字符串（strings），散列（hashes），列表（lists），集合（sets），有序集合（sorted sets）与范围查询，bitmaps，hyperloglogs和地理空间（geospatial） 索引半径查询。 Redis 内置了复制（replication），LUA脚本（Lua scripting）， LRU驱动事件（LRU eviction），事务（transactions）和不同级别的 磁盘持久化（persistence），并通过Redis哨兵（Sentinel）和自动分区（Cluster）提供高可用性（high availability）。
```

Redis的应用场景

- 缓存

使用Redis可以建立性能非常出色的缓存服务器，查询请求先在Redis中查找所需要的数据，如果能够查询到（命中）则直接返回，大大减轻关系型数据库的压力。

- 数据临时存储位置

使用token（令牌）作为用户登录系统时的身份标识，这个token就可以在Redis中临时存储。

Redis有什么好处

- 查询快
- 还可以设置数据的过期时间

  现在很多公司 都用redis 因为 redis能抗住 高并发

  高并发 = 大量的用户同时访问 场景： 火车票刚开的时候抢票， 618、双11、双12，世界总决赛直播平台

Redis命令

切换数据库

```
# Redis默认有16个数据库，使用select进行切换，数据库索引从0开始
select 0
```

KEY操作

```
# 查看所有的key
keys *
```

string

```
SET KEY VALUE [EX SECONDS] [PX MILLISECONDS] [NX|XX]
    给KEY设置一个string类型的值。
    EX参数用于设置存活的秒数。
    PX参数用于设置存活的毫秒数。
    NX参数表示当前命令中指定的KEY不存在才行。
    XX参数表示当前命令中指定的KEY存在才行。
GET KEY
    根据key得到值，只能用于string类型。
APPEND KEY VALUE
    把指定的value追加到KEY对应的原来的值后面，返回值是追加后字符串长度
STRLEN KEY
    直接返回字符串长度
INCR KEY
    自增1（要求：参与运算的数据必须是整数且不能超过整数Integer范围）
DECR KEY
    自减1（要求：参与运算的数据必须是整数且不能超过整数Integer范围）
INCRBY KEY INCREMENT
    原值+INCREMENT（要求：参与运算的数据必须是整数且不能超过整数Integer范围）
DECRBY KEY DECREMENT
    原值-DECREMENT（要求：参与运算的数据必须是整数且不能超过整数Integer范围）
GETRANGE KEY START END
    从字符串中取指定的一段，索引从0开始 START是开始取值的索引 END是结束取值的索引
SETRANGE KEY OFFSET VALUE
    从offset（从0开始的索引）开始使用VALUE进行替换 包含offset位置
SETEX KEY SECONDS VALUE
    设置KEY,VALUE时指定存在秒数
SETNX KEY VALUE
    新建字符串类型的键值对
MSET KEY VALUE [KEY VALUE ...]
    一次性设置一组多个键值对
MGET KEY [KEY ...]
    一次性指定多个KEY，返回它们对应的值，没有值的KEY返回值是(nil)
MSETNX KEY VALUE [KEY VALUE ...]
    一次性新建多个值
GETSET KEY VALUE
    设置新值，同时能够将旧值返回
```

list

```
LPUSH key value [value ...]
    针对key指定的list，从左边放入元素
RPUSH key value [value ...]
    针对key指定的list，从右边放入元素
LRANGE key start stop
    根据list集合的索引打印元素数据
    正着数：0,1,2,3,...
    倒着数：-1,-2,-3,...
LLEN key
    返回list集合的长度
LPOP key
    从左边弹出一个元素。 弹出=返回+删除。
```

```
RPOP key
    从右边弹出一个元素。
RPOPLPUSH source destination
    从source中RPOP一个元素，LPUSH到destination中
LINDEX key index
    根据索引从集合中取值
LINSERT key BEFORE|AFTER pivot value
    在pivot指定的值前面或后面插入value
    如果pivot值有重复的，那么就从左往右数，以第一个遇到的pivot为基准
    BEFORE表示放在pivot前面
    AFTER表示放在pivot后面
LPUSHX key value
    只能针对存在的list执行LPUSH
LREM key count value
    根据count指定的数量从key对应的list中删除value
    具体执行时从左往右删除，遇到一个删一个，删完为止
LSET key index value
    把指定索引位置的元素替换为另一个值
LTRIM key start stop
    仅保留指定区间的数据，两边的数据被删除
```

set

```
SADD key member [member ...]
    给key指定的set集合中存入数据，set会自动去重
SMEMBERS key
    返回可以指定的set集合中所有的元素
SCARD key
    返回集合中元素的数量
SISMEMBER key member
    检查当前指定member是否是集合中的元素
    返回1：表示是集合中的元素
    返回0：表示不是集合中的元素
SREM key member [member ...]
    从集合中删除元素
SINTER key [key ...]
    将指定的集合进行“交集”操作
    集合A：a,b,c
    集合B：b,c,d
    交集：b,c
SINTERSTORE destination key [key ...]
    取交集后存入destination
SDIFF key [key ...]
    将指定的集合执行“差集”操作
    集合A：a,b,c
    集合B：b,c,d
    A对B执行diff：a
    相当于：A-交集部分
SDIFFSTORE destination key [key ...]
SUNION key [key ...]
    将指定的集合执行“并集”操作
    集合A：a,b,c
```

```
    集合B: b,c,d
    并集: a,b,c,d
SUNIONSTORE destination key [key ...]
SMOVE source destination member
    把member从source移动到destination
SSCAN key cursor [MATCH pattern] [COUNT count]
    基于游标的遍历。cursor是游标值，第一次显示第一块内容时，游标取值为0；根据后续返回的新的游标值获取下一块数据。直到游标值变成0，说明数据遍历完成。
SRANDMEMBER key [count]
    从集合中随机返回count个数量的元素，count不指定就返回1个（数据有可能重复出现）
SPOP key [count]
    从集合中随机弹出count个数量的元素，count不指定就弹出1个（保证不会有重复数据出现）
```

hash

```
HSET key field value
    插入新数据返回1 修改旧数据返回0
    同样的key,同样的field,新的value，添加这个对应的第二个内容
HGETALL key
HGET key field
     获取这个value
HLEN key
HKEYS key
HVALS key
HEXISTS key field
HDEL key field [field ...]
HINCRBY key field increment
HMGET key field [field ...]
HMSET key field value [field value ...]
HSETNX key field value
    要求field是新建的
```

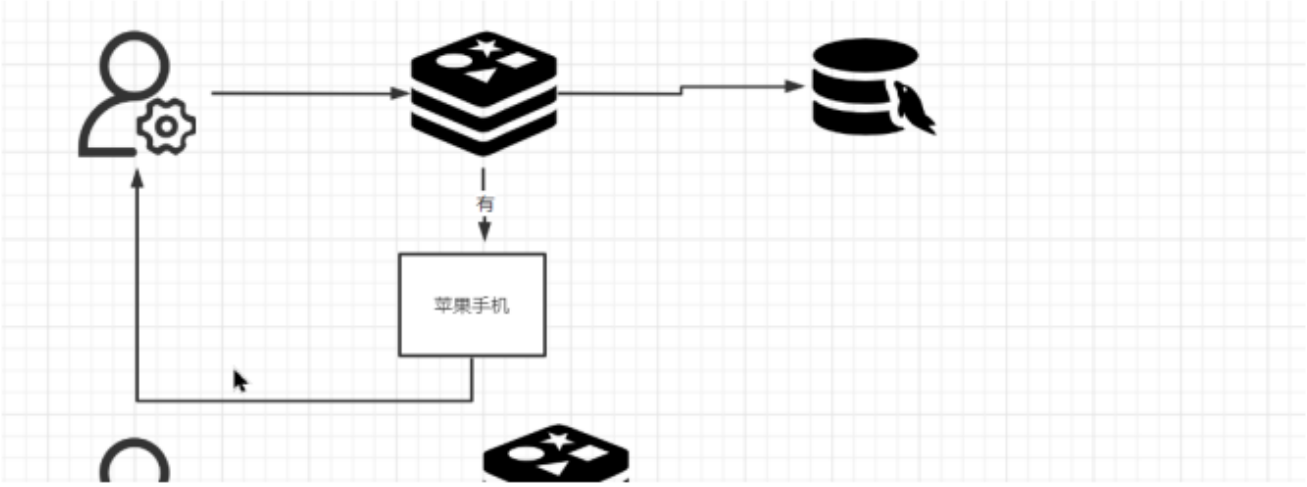
zset

```
ZADD key [NX|XX] [CH] [INCR] score member [score member ...]
ZRANGE key start stop [WITHSCORES]
ZCARD key
ZSCORE key member
ZINCRBY key increment member
ZRANGEBYSCORE key min max [WITHSCORES] [LIMIT offset count]
    在分数的指定区间内返回数据
    min参数可以通过
    -inf 表示负无穷
    max参数可以通过
    +inf 表示正无穷
    默认是闭区间
    可以通过（min（max 形式指定开区间，例如：（50 （80
ZRANK key member
    先对分数进行升序排序，返回member的排名。排名从0开始
ZREM key member [member ...]
```

代码里面的用法【了解】

在服务器上， 先请求 redis 判断 redis有没有，有的话就直接给用户，没有 再去mysql 查， 查完 先往redis写一份 然后在把信息给用户

为什么先往redis写， 因为 可能有其他用户同时来拿这个数据，这个时候你先写进去 第二个用户，来拿的时候 就拿到这个数据



面试问题

看你简历写了redis 能说一下么

你怎么说呢。 redis是非关系数据库，数据是key跟value主要使用的数据类型是string，它查询速度特别快，(为什么快，redis不是key和value，他可以直接根据key找到这个value就像在mysql通过主键查询一样很快，这个地方我了解这么多)

不要让他问你，你要反客为主： 直接说 我大概知道redis跟mysql的一点区别

我知道redis是非关系型数据库，数据是存在内存的，mysql是存在硬盘的，mysql支持索引和外键，但是redis不支持

redis有五个数据类型， string list set zset hash

string是字符串 hash 一个key 在对应 key和value list是有序列表 set是无序列表 zset是有分数的无序列表

因为string在项目中用的比较多，我还熟悉了一下 string命令的命令，

比如 放进去用set key value 取出来呢 get 这个key就可以了

项目中 :

登录的时候会有一个令牌验证，这个token就是存在redis里面的，以后做其他操作的时候，直接在redis拿这个token就好了。

后端令牌验证 ,你登录之后吧，会给你一个 token： 414125etdggdsczx23432532wtFJFS@#￥@fafsasf 以后呢

你做任何操作都带着这个token去操作 就不需要继续登录

redis的其他问题，数据持久化，它持久化不如mysql，如果面试问你持久化，你说 我知道这个地方 它不如mysql数据库，但是其他具体的我不了解，这个工作中是开发用的，我没有用过。