# 综合面试内容扩展

## 一、数据库

### 34.数据库的两种独立性之逻辑独立性

模式描述的是数据的全局逻辑结构，外模式描述的是数据的局部逻辑结构。

对应于同一个模式可以有任意多个外模式。对于每一个外模式，数据库系统都有一个外模式/模式映像，它定义了该外模式与模式之间的对应关系。这些映像定义通常包含在各模式的描述中。 当模式改变时，由数据库管理员对各个外模式/模式映像作相应的改变，可以使外模式保持不变。应用程序是依据数据的外模式编写的，从而应用程序可以不必修改，保证了数据与程序的逻辑独立性。

### 36.什么是数据库控制语言，试举例说明

数据控制语言（DCL）是用来设置或者更改数据库用户或角色权限的语句，这些语句包括GRANT、DENY、REVOKE等语句，在默认状态下，只有sysadmin、dbcreator、db_owner或db_securityadmin等角色的成员才有权利执行数据控制语言。

- GRANT语句是授权语句，它可以把语句权限或者对象权限授予给其他用户和角色。
- DENY语句用于拒绝给当前数据库内的用户或者角色授予权限，并防止用户或角色通过其组或角色成员继承权限。
- REVOKE语句是与GRANT语句相反的语句，它能够将以前在当前数据库内的用户或者角色上授予或拒绝的权限删除，但是该语句并不影响用户或者角色从其他角色中作为成员继承过来的权限。

## 二、计算机网络

### 1.比较TCP与UDP

- TCP面向连接（如打电话要先拨号建立连接）；
  UDP是无连接的，即发送数据之前不需要建立连接。
- TCP提供可靠的服务。也就是说，通过TCP连接传送的数据，无差错，不丢失，不重复，且按序到达；
  UDP尽最大努力交付，即不保证可靠交付。

注：TCP通过校验和，重传控制，序号标识，滑动窗口、确认应答实现可靠传输。如丢包时的重发控制，还可以对次序乱掉的分包进行顺序控制。

- UDP具有较好的实时性，工作效率比TCP高，适用于对高速传输和实时性有较高的通信或广播通信。
- 每一条TCP连接只能是点到点的；
  UDP支持一对一，一对多，多对一和多对多的交互通信。
- TCP面向字节流，实际上是TCP把数据看成一连串无结构的字节流；
  UDP是面向报文的，UDP没有拥塞控制，因此网络出现拥塞不会使源主机的发送速率降低（对实时应用很有用，如IP电话，实时视频会议等）

注：

1.面向报文（UDP）和面向字节流（TCP）的区别

面向报文的传输方式是应用层交给UDP多长的报文，UDP就照样发送，即一次发送一个报文。因此，应用程序必须选择合适大小的报文。若报文太长，则IP层需要分片，降低效率。UDP对应用层交下来的报文，既不合

并，也不拆分，而是保留这些报文的边界。这也就是说，应用层交给UDP多长的报文，UDP就照样发送，即一次发送一个报文。

面向字节流的话，虽然应用程序和TCP的交互是一次一个数据块（大小不等），但TCP把应用程序看成是一连串的无结构的字节流。TCP有一个缓冲，当应用程序传送的数据块太长，TCP就可以把它划分短一些再传送。如果应用程序一次只发送一个字节，TCP也可以等待积累有足够多的字节后再构成报文段发送出去。

2.什么是报文?

例如一个 100kb 的 HTML 文档需要传送到另外一台计算机，并不会整个文档直接传送过去，可能会切割成几个部分，比如四个分别为 25kb 的数据段。而每个数据段再加上一个 TCP 首部，就组成了 TCP 报文。一共四个 TCP 报文，发送到另外一个端。另外一端收到数据包，然后再剔除 TCP 首部，组装起来。等到四个数据包都收到了，就能还原出来一个完整的 HTML 文档了。

在 OSI 的七层协议中，第二层（数据链路层）的数据叫「Frame」，第三层（网络层）上的数据叫「Packet」，第四层（传输层）的数据叫「Segment」。

TCP 报文 (Segment)，包括首部和数据部分。


## 6.什么是码元？什么是码元长度?

在数字通信中常常用时间间隔相同的符号来表示一个二进制数字，这样的时间间隔内的信号称为(二进制）码元。而这个间隔被称为码元长度。

通俗点说，可以把一个码元看做一个存放一定信息量的包，如果只存放1bit，那么波特率等于比特率，但是一般不止存放1bit。

如一串二进制信息为101010101 当一个码元携带的信息量为1bit时，那么就有9个码元，其波特率相当于比特率，如果每三个一组101，010，101，这时就可以使用8种振幅来表示某个码元，这里相当于一个码元就包含了3bit，这里码元的离散取值数目就是8。

由此可得波特率和比特率的关系，若码元的离散取值数目是L，波特率是B，数据率（比特率）为C，则 $C = Blog_2L.$（当L=2时，C=B）

## 9.网络时延由哪几部分组成？各产生于何处?

《计算机网络》（第七版）书本P22 & P23

## 24.DNS的递归查询与迭代查询

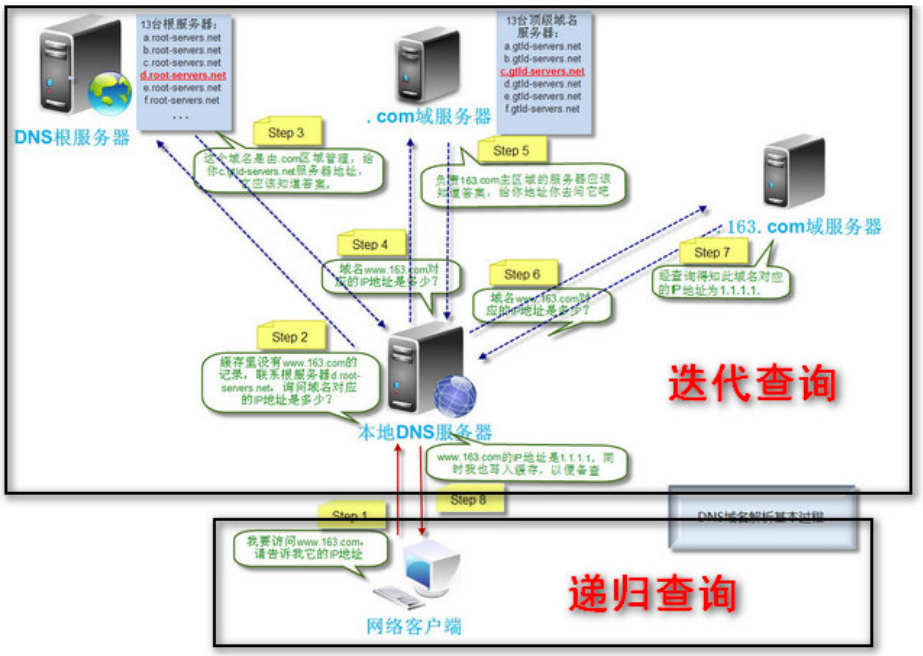
（1）递归查询

递归查询是一种DNS 服务器的查询模式，在该模式下DNS服务器接收到客户机请求，必须使用一个准确的查询结果回复客户机。如果DNS 服务器本地没有存储查询DNS信息，那么该服务器会询问其他服务器，并将返回的查询结果提交给客户机。

（2）迭代查询

DNS服务器另外一种查询方式为迭代查询，DNS服务器会向客户机提供其他能够解析查询请求的DNS服务器地址，当客户机发送查询请求时，DNS服务器并不直接回复查询结果，而是告诉客户机另一台DNS 服务器地址，客户机再向这台DNS服务器提交请求，依次循环直到返回查询的结果为止。

两种过程的示意图:




## 25.ARP协议及其过程

《计算机网络》（第七版）书本P20

## 30.多路复用与多路分解

我们知道，在网络上主机与主机之间的通信，实质上是主机上运行的应用进程之间的通信。例如，当你通过Http上网浏览网页时，实质上是你所访问的主机的服务器进程与你本机的浏览器进程在进行通信。试想一下，当你在上网的同时，还挂着QQ，还使用ftp下载大文件，这时就有三个网络上的进程与你的主机上的三个进程进行通信，那么系统是怎么样正确地把接收到的数据定位到指定的进程中的呢？也就是说，系统是怎么把从ftp服务器发送过来的数据交付到ftp客户端，而不把这些数据交付到你的QQ上的呢？反过来考虑，系统又是如何精确地把来自各个应用进程的数据发到网络上指定上的主机（服务器）上的对应进程的呢？这就是多路分解与多路复用的作用了。

为了说明这个问题，先来补充一下操作系统方面的知识，以Linux对文件和设备的管理和使用方式为例。

为了方便资源的使用，提高机器的性能、利用率和稳定性等等原因，我们的计算机都有一层软件叫做操作系统，它用于帮我们管理计算机可以使用的资源，当我们的程序要使用一个资源的时候，可以向操作系统申请，再由操作系统为我们的程序分配和管理资源。通常当我们要访问一个内核设备或文件时，程序可以调用系统函数，系统就会为我们打开设备或文件，然后返回一个文件描述符fd（或称为ID，是一个整数），我们要访问该设备或文件，只能通过该文件描述符。可以认为该编号对应着打开的文件或设备。

而当我们的程序要使用网络时，要使用到对应的操作系统内核的操作和网卡设备，所以我们可以向操作系统申请，然后系统会为我们创建一个套接字Socket，并返回这个Socket的ID，以后我们的程序要使用网络资源，只要向这个Socket的编号ID操作即可。而我们的每一个网络通信的进程至少对应着一个Socket。向Socket的ID中写数据，相当于向网络发送数据，向Socket中读数据，相当于接收数据。而且这些套接字都有唯一标识符——端口号。

有了上面的了解后，再来说说什么是多路分解和多路复用。

每个运输层的报文段中设置了几个字段，包括源端口号和目的端口号等。多路分解就是，在接收端，运输层检查这些字段并标识出接收套接字，然后将该报文定向到该套接字。其工作方式可以简单地认为是这样的，主机上的每个套接字被分配一个端口号，当报文到达主机时，运输层检查报文段中的目的端口号，并将其定向到相应的套接字。

多路复用就是从源主机的不同套接字中收集数据块，并为每个数据块封装上首部信息从而生成报文段，然后将

报文段传递到网络层中去。

## 32.什么是虚电路网络，什么是数据报网络?

TCP的特性就是面向连接的，是可靠传输，可以差错控制和流量控制，TCP的数据传送是建立在虚电路的基础上的。

虚电路，应该是指“意”的虚。就是存在那么一条电路，逻辑上好像是固定的存在的，但事实它是随着会话的不同而使用不同的路径。也就是说没有一条固定的路径。它是在通信过程中灵活地变动的。具体使用哪条路径以及如何操作，通过三次握手来进行建立。在发握手信号的过程中，知道了发送方与接收方，然后建立一条路径。这条路径是一条逻辑上的电路。以后的传输就由此来进行。

在数据报方式中，每个分组被称为一个数据报。若干个数据报构成一次要传送的报文或数据块。每个数据报自身携带有足够的信息。它的传送是被单独处理的。一个节点接收到一个数据报后，根据数据报中的地址信息和节点所存储的路由信息，找出一个合适的出路，把数据报原样地发送到下一个节点。

当端系统要发迭一个报文时，将报文拆成若干个带有序号和地址信息的数据报，依次发给网络节点。此后，各个数据报所走的路径就可能不同了，因为各个节点在随时根据网络的流量、故障等情况选择路由。由于各行其道，各数据报不能保证按顺序到达目的节点，有些数据报甚至还可能在途中丢失。在整个传送过程中，不必建立虚电路，但要为每个数据报作路由选择。

## 35.计算机网络有哪几种校验算法?

**(1).奇偶校验**

奇偶校验就是在发送的每一个字节后都加上一位，使得每个字节中1的个数为奇数个或偶数个。比如我们要发送的字节是0x1a，二进制表示为0001 1010。

- 采用奇校验，则在数据后补上个0，数据变为0001 1010 0，数据中1的个数为奇数个（3个）
- 采用偶校验，则在数据后补上个1，数据变为0001 1010 1，数据中1的个数为偶数个（4个）

接收方通过计算数据中1个数是否满足奇偶性来确定数据是否有错。

奇偶校验的缺点也很明显，首先，它对错误的检测概率大约只有50%。也就是只有一半的错误它能够检测出来。另外，每传输一个字节都要附加一位校验位，对传输效率的影响很大。因此，在高速数据通讯中很少采用奇偶校验。奇偶校验优点也很明显，它很简单，因此可以用硬件来实现，这样可以减少软件的负担。因此，奇偶校验也被广泛的应用着。

**(2).累加和校验**

累加和校验实现方式有很多种，最常用的一种是在一次通讯数据包的最后加入一个字节的校验数据。这个字节内容为前面数据包中全部数据的忽略进位的按字节累加和。比如下面的例子:

- 我们要传输的信息为：6、23、4
- 加上校验和后的数据包：6、23、4、33

这里 33 为前三个字节的校验和。接收方收到全部数据后对前三个数据进行同样的累加计算，如果累加和与最后一个字节相同的话就认为传输的数据没有错误。

累加和校验由于实现起来非常简单，也被广泛的采用。但是这种校验方式的检错能力也比较一般，对于单字节的校验和大概有1/256 的概率将原本是错误的通讯数据误判为正确数据。之所以这里介绍这种校验，是因为CRC校验在传输数据的形式上与累加和校验是相同的，都可以表示为：通讯数据 校验字节（也可能是多个字节）

**(3).循环冗余CRC**

CRC算法的基本思想是将传输的数据当做一个位数很长的数。将这个数除以另一个数。得到的余数作为校验数据附加到原数据后面。例如:

6、23、2 可以看做一个2进制数： 00000110 00010111 00000100

假如被除数选9，二进制表示为：1001

则除法运算可以表示为：

```
                1010, 1101, 0011, 1001
          ____________________________________
1001 )0000, 0110, 0001, 0111, 0000, 0010
          100, 1
          -----
            1, 100
            1, 001
            -----
               111, 0
               100, 1
               ------
                10, 11
                10, 01
                ------
                  1011
                  1001
                  ------
                    10, 000
                     1, 001
                     ------
                       1110
                       1001
                       ------
                          101, 0
                          100, 1
                          -----
                              1010
                              1001
                              ----
                              0001
```

可以看到，最后的余数为1。如果我们将这个余数作为校验和的话，传输的数据则是：6、23、2、1

CRC 算法和这个过程有点类似，不过采用的不是上面例子中的通常的这种除法。循环冗余校验（英语：Cyclic redundancy check，通称“CRC”）是一种根据网络数据包或电脑文件等数据产生简短固定位数校验码的一种散列函数，主要用来检测或校验数据传输或者保存后可能出现的错误。生成的数字在传输或者存储之前计算出来并且附加到数据后面，然后接收方进行检验确定数据是否发生变化。一般来说，循环冗余校验的值都是32位的整数。

## 36.子网掩码和默认网关是什么以及它们的作用?

### (1).子网掩码

网络上，数据从一个地方传到另外一个地方，是依靠IP寻址。从逻辑上来讲，是两步的。

- 第一步，从 IP 中找到所属的网络，好比是去找这个人是哪个小区的；
- 第二步，再从 IP 中找到主机在这个网络中的位置，好比是在小区里面找到这个人。

第一步中的网络，就称之为「子网」（Subnet）。从逻辑上来讲，一般同一子网（Subnet）是使用相同的网关。就好比，一个小区的入口。IPv4的IP地址是32位的，形式如xxx. xxx. xxx. xxx，每一个 xxx取值都是 0 - 255。

到底是前三个 xxx 相同，就代表同一个子网，还是前两个，还是其他？这个并不一定。就好比小区有大有小，有的小区有上千户人家，有的小区只有区区几个。

所以，就引入「子网掩码」（Subnet Mask）来标识该子网的大小。我们一般看到的 IP 地址是十进制的编码，所以如果换一个视角，从二进制的角度看，每一个 IP 地址就是 32 位 1 或 0。

子网掩码，就是用来告诉这个子网的覆盖区间。这32位中，前多少位是网络段？当然，余下的就是主机段。举典型的例子：IP 中前 24 位代表子网号，后 8 位代表主机号。所以子网掩码就是 24 个 1（代表前 24 位是子网部分），加 8 个 0（后 8 位是主机部分）。如果沿用 IP 的标识方式，就是 255.255.255.0 。每一个 255 对应 8 个二进制 1，最后一个 0 对应 8 个二进制 0。该子网可以容纳最多 256 台主机，也就是主机号从 0 到 255。当然，实际

情况没有这么多，有一些特殊数字有保留用处（广播、网关等）。

子网掩码的作用：

子网掩码是用来判断任意两台计算机的ip地址是否属于同一子网络的根据，并将某个IP地址划分成网络地址和主机地址两部分，最为简单的理解就是两台计算机各自的ip地址与子网掩码进行与运算后，得出的结果是相同的，则说明这两台计算机是处于同一个子网络上的，可以进行直接的通讯。

计算过程是这样的，将IP地址和子网掩码都换算成二进制，然后进行与运算，结果就是网络地址。与运算如下所示，上下对齐，1位1位的算，1 & 1 = 1，其余组合都为0。

```
  1 0 1 0
  1 1 0 0
与运算
结果为 1 0 0 0
```

例如:计算IP地址为：202.99.160.50，子网掩码：255.255.255.0的网络地址步骤如下:

1)将IP地址和子网掩码分别换算成二进制

202.99.160.50 换算成二进制为 11001010 01100011 10100000 00110010

255.255.255.0 换算成二进制为 11111111 11111111 11111111 00000000

2)将二者进行与运算

```
  11001010 01100011 10100000 00110010
  11111111 11111111 11111111 00000000
与运算
  11001010 01100011 10100000 00000000
```

3)将运算结果换算成十进制,这就是网络地址.

11001010 01100011 10100000 00000000换算成十进制就是202.99.160.0

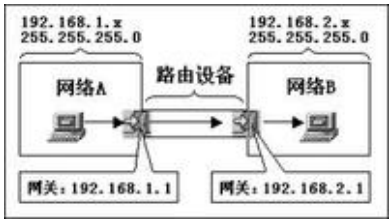
**(2).**默认网关

什么是网关?

那么网关到底是什么呢？网关实质上是一个网络通向其他网络的IP地址。比如有网络A和网络B，网络A的IP地址范围为“192.168.1.1 ~ 192.168.1.254”，子网掩码为255.255.255.0；网络B的IP地址范围为“192.168.2.1 ~ 192.168.2.254”，子网掩码为255.255.255.0。在没有路由器的情况下，两个网络之间是不能进行TCP/IP通信的，即使是两个网络连接在同一台交换机（或集线器）上，TCP/IP协议也会根据子网掩码（255.255.255.0）判定两个网络中的主机处在不同的网络里。而要实现这两个网络之间的通信，则必须通过网关。如果网络A中的主机发现数据包的目的主机不在本地网络中，就把数据包转发给它自己的网关，再由网关转发给网络B的网关，网络B的网关再转发给网络B的某个主机（如附图所示）。网络A向网络B转发数据包的过程。

所以说，只有设置好网关的IP地址，TCP/IP协议才能实现不同网络之间的相互通信。那么这个IP地址是哪台机器的IP地址呢？网关的IP地址是具有路由功能的设备的IP地址，具有路由功能的设备有路由器、启用了路由协议的服务器（实质上相当于一台路由器）、代理服务器（也相当于一台路由器）。

在和Novell NetWare网络交互操作的上下文中，网关在Windows网络中使用的服务器信息块 (SMB) 协议以及NetWare网络使用的NetWare核心协议(NCP)之间起着桥梁的作用。网关也被称为IP路由器。



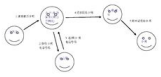

举例说明

假设你的名字叫小不点(很小)，你住在一个大院子里，你的邻居有很多小伙伴，父母是你的网关。当你想跟院子里的某个小伙伴玩，只要你在院子里大喊一声他的名字，他听到了就会回应你，并且跑出来跟你玩。

但是你家长不允许你走出大门，你想与外界发生的一切联系，都必须由父母（网关）用电话帮助你联系。假如你想找你的同学小明聊天，小明家住在很远的另外一个院子里，他家里也有父母（小明的网关）。但是你不知道小

明家的电话号码，不过你的班主任老师有一份你们班全体同学的名单和电话号码对照表，你的老师就是你的DNS服务器。于是你在家里和父母有了下面的对话：

- 1.小不点：妈妈(或爸爸),我想找班主任查一下小明的电话号码行吗?
- 2.家长：好，你等着。（接着你家长给你的班主任挂了一个电话，问清楚了小明的电话）问到了，他家的号码是211.99.99.99
- 3.小不点：太好了！妈(或爸),我想找小明，你再帮我联系一下小明吧。
- 4.家长：没问题。（接着家长向电话局发出了请求接通小明家电话的请求，最后一关当然是被转接到了小明家家长那里，然后他家长把电话给转到小明）.
  就这样你和小明取得了联系。



什么是默认网关?

如果搞清了什么是网关，默认网关也就好理解了。就好像一个房间可以有多扇门一样，一台主机可以有多个网关。默认网关的意思是一台主机如果找不到可用的网关，就把数据包发给默认指定的网关，由这个网关来处理数据包。现在主机使用的网关，一般指的是默认网关。

如何设置默认网关

一台电脑的默认网关是不可以随随便便指定的，必须正确地指定，否则一台电脑就会将数据包发给不是网关的电脑，从而无法与其他网络的电脑通信。

## 37.网络中数据的分片与重组发生在什么时候?

**(1).IP数据报分片的原因**

在TCP/IP分层中，数据链路层用 MTU (Maximum Transmission Unit，最大传输单元)来限制所能传输的数据包大小，MTU是指一次传送的数据最大长度，不包括数据链路层数据帧的帧头，如以太网的MTU为1500字节，实际上数据帧的最大长度为1512字节，其中以太网数据帧的帧头为12字节。

**(2).分片的思想**

当发送的IP数据报的大小超过了MTU时，IP层就需要对数据进行分片，否则数据将无法发送成功。IP分片发生在IP层 ，不仅源端主机会进行分片，中间的路由器也有可能分片，因为不同的网络的MTU是不一样的，如果传输路径上的某个网络的MTU比源端网络的MTU要小，路由器就可能对IP数据报再次进行分片。而分片数据的重组只会发生在目的端的IP层。