# 软考-数据库系统工程师

## 计算机系统知识

### 计算机系统基础知识

#### 硬件及组成

##### 一、计算机的组成

计算机硬件由5大件组成：控制器、运算器、存储器、输入设备、输出设备

###### 运算器（ALU）：

也称算术逻辑单元，对数据进行算术运算和逻辑运算

加法器（累加器）：

专门存放算术或逻辑运算的操作数和运算结果的寄存器。 

程序状态寄存器 PSW：

用来存放两类信息：一类是体现当前指令执行结果的各种状态信息，如有无进位（CY位），有无溢出（OV位），结果正负 （SF位），结果是否为零（ZF位），奇偶标志位（P位）等；另一类是存放控制信息，如允许中断(IF位)，跟踪标志（TF位）等

###### 控制器

是分析和执行指令的部件

指令寄存器

用于保存当前正在执行的指令

指令译码器

分析当前指令的操作码是要做什么

程序计数器

存放下一条指令的地址

定时与控制电路

堆栈和堆栈指针

#### 数据表示

##### 数的进制

十进制：以D表示。如：(123)D或(123)10

二进制：以B表示。如：(1011)B或(1011)2

八进制：以O（大写o）表示。如：(301)O或(301)8

十六进制：以H表示。如：（13E）H或（13E）16

##### 二、进制转换

###### 1.十进制转非十进制

把被转换的十进制整数反复地除以非十进制数，直到商为0，所得的余数（从末位读起）就是这个数的非十进制表示。简称“除\*（\*为非十进制数）取余法”

###### 2.非十进制转十进制

方法：非十进制数按权展开求和

如：（10110）2=1\*24+0\*23+1\*22+1\*21+0\*20=22

(335)8=3\*82+3\*81+5\*80=221

##### 三、原码、反码、补码、移码

###### 1.带符号数的表示

通常的做法是约定一个数的最高位为符号位，若该位为0，则表示正数；若该位为1，则表示负数

###### （1）原码

用最高位表示符号位，数值部分用二进制绝对值表示。

如：+11的原码：00001011，-11的原码：10001011

###### （2）反码

正数的反码和其原码形式相同，负数的反码是除符号位，其他各位逐位取反（即0变1，1变0）

如：+11的反码：00001011，-11的反码：11110100

###### （3）补码

正数的补码和其原码形式相同，负数的补码是原码除符号位以外逐位取反（即0变1，1变0），最后在末尾加1.

如：+11的补码：00001011，-11的补码：11110101

将补码转换为真值：[[X]补]补=[X]原

###### （4）移码（增码）

无论正数、负数，在补码的基础上对符号位取反，一般用做浮点数的阶码，引入的目的是为了保证浮点数的机器零为全0

如：+11的补码：00001011，-11的补码：11110101

+11的移码：10001011，-11的移码：01110101

##### 四、定点数和浮点数

计算机中，通常是用定点数来表示整数和纯小数，分别称为定点整数和定点小数。对于既有整数部分又有小数部分的数，一般用浮点数表示。

###### 1.定点数

定点整数：

小数点的位置固定在最低位的右边，不占位

定点小数：

小数点的位置固定在符号位与最高数值位之间，表示一个纯小数

###### 2.浮点数

用类似科学计算机法来表达，即

N=M\*Re

M称为尾数，R称为基数，e为阶码（指数）

比如：1001.101的规范浮点数表达为1.001101\*23

浮点数利用指数达到了浮动小数点的效果，从而灵活地表达更大范围的实数

#### 校验码

##### 一、编码体系

指一种编码方式中所有合法码字的集合

##### 二、编码效率

合法码字占所有码字的比率就是编码效率。

##### 三、码距

码距是衡量一种编码方式的抗错误能力的一个指标

###### 1.码字的码距

一个编码系统中任意两个合法的编码之间的不同的二进制位的数目叫这两个码字的码距

###### 2.编码系统的码距

该编码系统的任意两个编码之间的距离的最小值称为该编码系统的码距

##### 四、误码

数字信息在传输和存取的过程中，由于各种意外情况的发生，数据可能会发生错误，即所谓误码。

##### 五、奇偶校验

串口通信中使用奇偶校验作为数据校验的方法

使用一位奇偶校验的方法能够检测出1位错误，但无法判断是哪一位出错。

当发生两位同时出错的情况时，奇偶校验也无法检测出来。所以奇偶校验常用于对少量数据的校验，如1个字节。

###### 1.奇校验：

被传输的有效数据中“1”的个数是奇数个，校验位填“0”，否则填“1”

###### 2.偶校验：

被传输的有效数据中“1”的个数是偶数个，校验位填“0”，否则填“1”。

XOR（异或运算）是一种偶校验

##### 六、海明码

海明码是奇偶校验的一种扩充。和奇偶校验的不同之处在于海明码采用多位校验码的方式，在这些多个校验位中的每一位都对不同的信息数据位进行奇偶校验，通过合理地安排每个校验位对原始数据进行校验的位组合，可以达到发现错误、纠正错误的目的（当出现两位错误时，海明码能够查错，但无法纠错）。

海明码可以发现“<=码距-1”位的错误；可以纠正“<码距/2”位的错误，因此，如果要能够纠正n位错误，所需最小的码距应该的2n-1

###### 海明码的原理

在数据中间加入几个校验码，码距均匀拉大，当某一位出错，会引起几个校验位的值发生变化

###### 海明不等式

校验码个数为k，可以表示2k个信息，1个信息用来表示“没有错误”，其余2k-1个表示数据中存在错误，如果满足2k-1>=m+k（m为数据长度，m+k为编码后的数据编码总长度），则在理论上k个校验码就可以判断是哪一位（包括信息码和校验码）出现了问题

###### 海明码的编码规则

校验位依次放在第2i（i=0,1,2,3…）位，其余位置为信息位。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| B7 | B6 | B5 | B4 | B3 | B2 | B1 |
| k3 | k2 | k1 | r2 | k0 | r1 | r0 |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 |

上表4个信息位k0,k1,k2,k3，3个校验位r0,r1,r2

第i个信息位的位数为参与校验它的校验位的位数之和。如上例7＝4+2+1；6＝4+2；5＝4+1；3＝2+1

从上式可得，k3要参与r2，r1和r0的生成，k2参与r2和r1的生成，k1参与r2和r0的生成，k0参与r1和r0的生成。则产生下列式子“⊕”符号为异或运算

r0=k3⊕k1⊕k0🡺 k3⊕k1⊕k0⊕r0=0🡺 B1⊕B3⊕B5⊕B7=0

r1=k3⊕k2⊕k0🡺 k3⊕k1⊕k0⊕r1=0🡺 B2⊕B3⊕B6⊕B7=0

r2=k3⊕k2⊕k1🡺 k3⊕k2⊕k1⊕r2=0🡺 B4⊕B5⊕B6⊕B7=0

若三个校验方程都成立，即方程式右边都等于0，则说明没有出错。若不成立，即方程式右边不等于0，说明有……

从三个方程式右边的值，可以判断哪一位出错。出错位置为从下向上看相应的二进制数值，若三个程式右边的值为100（如下式），说明第4位出错。

B1⊕B3⊕B5⊕B7=0

B2⊕B3⊕B6⊕B7=0

B4⊕B5⊕B6⊕B7=1

##### 七、循环冗余校验码

广泛地在网络通信及磁盘存储时采用

循环冗余校验码的纠错能力取决于K值（信息码长度）和R值（校验码长度），在实践中，K值往往取得非常大，远远大于R的值，提高了编码效率。在这种情况下，循环冗余校验就只能检错不能纠错。

一般来说，R位生成多项式可检测出所有双错、奇数位错和突发错位小于或等于R的突发错误。

###### 1.多项式

在循环冗余（CRC）码中，无一例外地要提到多项式的概念。

一个二进制数可以以一个多项式来表示。如1011表示为多项式X3+X1+X0，如果把这里的X替换为2，这个多项式的值就是该数的值。从这个转换可以看出多项式最高幂次为n，则转换为二进制数有n+1位

2.生成多项式

生成多项式是接受方和发送方的一个约定，也就是一个二进制数，在整个传输过程中，这个数始终保持不变。

###### 3.编码组成

编码的组成是由K位信息码，加上R位的校验码

###### 4.校验码的生成

生成步骤：

1. 将K位数据C（x）左移R位，给校验位留下空间，得到移位后的多项式为C(x)\*XR。
2. 将这移位后的信息多项式除以生成多项式，得到R位的余数多项式。
3. 将余数作为校验码嵌入信息位左移后的空间

例：信息位为1010 0110，约定的生成多项式为1110011即X5+X4+X+1,则：

1010 0110左移5位－〉1010 0110 0000 0

用1010 0110 0000 0和11110011进行模2运算（异或）

余11000

即校验码11000，所以CRC码是1010 0110 11000

### 计算机体系结构

#### 指令系统

指计算机所能执行的全部指令的集合，它描述了计算机内全部的控制信息和“逻辑判断”能力。

一条指令包括：操作码、地址码（操作数）

##### 一、指令系统类别

###### 1.根据地址码区分

立即寻址：直接使用地址码，地址码就是一个数

直接寻址：地址码是一个地址，需去内存中取出该数

间接寻址：地址码是一个地址，内存中仍是一个地址，需再寻址一次

寄存器寻址：地址码是一个寄存器的地址

寄存器间接寻址：地址码是一个寄存器的地址的地址

##### 二、复杂指令集计算机（CISC）与精简指令集计算机（RISC）

###### 1.复杂指令集计算机（CISC）

为提高操作系统的效率，人们最初选择向指令系统中添加更多、更复杂的指令来实现，导致指令集越来越大。这种类型的计算机，称为指令集计算机（CISC）

主要特点

1. 指令数量多：指令系统拥有大量的指令，有100－250条
2. 指令使用频率相关悬殊：最常使用的是一些比较简单的指令，80%的时候使用的是20%的指令
3. 支持很多种寻址方式：通常为5－20种
4. 变长的指令：指令长度不是固定的，变长的指令增加指令译码电路的复杂性。
5. 指令可以对存储器单元中数据直接进行处理：典型的CISC处理器通常都有指令能够直接对内存单元中的数据进行处理，其执行速度较慢

###### 2.精简指令集计算机（RISC）

对指令数目和寻址方式做精简，指令的指令周期相同，采用流水性技术，指令并行执行程度更好，这类是精简指令集计算机（RISC）

主要特点

1. 指令数量少：优先选取使用频率最高的一些简单指令以及一些常用的指令，避免使用复杂指令
2. 指令的寻址方式少：通常只支持寄存器寻址方式、立即数寻址方式以及相对寻址方式。
3. 指令长度固定，指令格式种类少：因为RISC指令数量少，格式相对简单，其指令长度固定，指令之间各字段的划分比较一致，译码相对容易。
4. 只提供了Load/Store指令访问存储器：只提供了从存储器读数Load和把数据定稿存储器Store两条指令，其余所有操作都在CPU的寄存器间进行。
5. 以硬布线逻辑控制为主：为了提高操作的执行速度，通常采用硬布线逻辑来构建控制器。
6. 单周期指令执行：因为简化了指令系统，很容易利用流水线技术使得大部分指令在一个机器周期内完成。
7. 优化的编译器：RISC精简指令集使编译工作简单化。

###### 3.CISC和RISC的简单对比

|  |  |  |
| --- | --- | --- |
|  | CISC | RISC |
| 指令条数 | 多 | 只选取最常见的指令 |
| 指令复杂度 | 高 | 低 |
| 指令长度 | 变化 | 固定 |
| 指令执行周期 | 随指令变化大 | 大多在一个机器同期完成 |
| 指令格式 | 复杂 | 简单 |
| 寻址方式 | 多 | 极少 |
| 涉及访问主存指令 | 多 | 极小，大部分只有两条存指令 |
| 通用寄存器数量 | 一般 | 大量 |
| 译码方式 | 微程序控制 | 硬件电路 |
| 对编译系统要求 | 低 | 高 |

##### 三、流水线技术

在程序执行时，多条指令重叠进程操作的一种任务分解技术。把一个任务分解为若干顺序执行的子任务，不同的子任务由不同的执行机构来负责执行，而这些执行机构可以同时并行工作。

流水线时间

建立时间、排空时间

###### 1.计算机执行时间

假定有某种类型的任务，可分成N个子任务，每个子任务需要时间t，则完成该任务所需的时间为N\*t

若以传统的方式，完成k个任务所需的时间是kNt。

使用流水线技术，花费的时间是Nt0+(k-1)t1.

（Nt0为完成任务所需的时间，如果每个子任务所需的时间相同，t0=t1。如果每个子任务所需的时间不同，其时间t1取决于执行顺序中最慢的那一个）

###### 2.流水线的吞吐率

在单位时间内（一般为1秒）流水线所完成的任务数量或输出的结果数量TP=n/Tk （n为任务数，Tk是处理完成n个任务所有的时间）

###### 3.加速比

不采用流水线的执行时间/采用流水线的执行时间

用来衡量并行系统或程序并行化的性能和效果

###### 4.影响流水线的主要因素

**转移指令**：因为前面的转移指令还没有完成，流水线无法确定下一条指令的地址，因些也就无法向流水线中添加这条指令

**共享资源访问的冲突：**后一条指令需要使用的数据，与前一条指令发生冲突，或者相令的指令使用了相同的寄存器

**响应中断：**当有中断请求时，流水线也会停止。对于这种情况有两冲响应方式

精确断点法：立即停止，这种方法能够立即响应中断

不精确断点法：流水线中的指令继续执行，不再新增指令到流水线

#### 存储系统

存储器是计算机系统中的记忆设备，用来存放程序和数据

##### 一、存储器分类

寄存器

Cache（高速缓冲存储器）

主存储器

辅存储器

以上四个存储器从下至上速度越来越快，容量越来越小，成本越来越高

##### 二、存储器的存取方式

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **存取方式** | **读/写装置** | **数据块标志** | **访问特性** | **代表** |
| 顺序存取 | 共享读/写装置 | 无 | 特定线性顺序 | 磁带 |
| 直接存取 | 共享读/写装置 | 数据分块，每块一个唯一标志 | 可直接移到特定数据快 | 磁盘 |
| 随机存取 | 每个可寻址单元专有读/写装置 | 每个可寻址单元均有一个唯一地址 | 随时访问任何一个存储单元 | 主存储器 |
| 相联存取（属随机存取） | 每个可寻址单元专有读/写装置 | 每个可寻址单元均有一个唯一地址 | 根据内容需非地址来选择读写点 | cache |

##### 三、存储器的性能

###### 1.存取时间

对于随机存取而言，就是完成一次读/写所花的时间；对非随机存取，就是将读/写装置移动到目的位置所花的时间

###### 2.存储器带宽

每秒能访问的位数。通常存储器周期是纳秒级(ns)。

计算公式：1/存储器周期\*每周期可访问的字节数。即存储器频率\*每周期可访问的字节数

###### 3.数据传输率

每秒输入/输出的位数。

对于随机存取而言，传输率R=1/存储器周期。即存储器的频率

##### 四、主存储器

###### 主存储器的种类

RAM：随机存储器

可读/写，只能暂存数据，断电后数据丢失

**SRAM:**静态随机存储器，在不断电时信息能够一直保持，读写速度快，生产成本高，多用于容量较小的高速缓冲存储器

**DRAM:**动态随机存储器，需要定时刷新以维持信息不丢失，读写速度较慢，集成度高，生成成本低，多用于容量较大的主存储器

ROM:只读存储器

出厂前用掩膜技术写入，常用于存放BIOS和微程序控制

PROM:可编程ROM

只能够一次写入，需用特殊电子设备进行写入

EPROM：可擦除的PROM

用某种方法可擦去信息，可写入多次

E2PROM:电可擦除EPROM

可以写入，但速度慢。

闪速存储器（Flash Memory）

其特性介于EPROM与E2PROM之间。但不能进行字节级别的删除操作

CAM（相联存储器）

CAM是一种特殊的存储器，是一种基于数据内容进行访问的存储设备。其速度比基于地址进行读写的方式要快

##### 五、辅助存储器

###### 磁带

磁带是一种顺序存取的设备。

特点：存储容量大，价格便宜。适合数据的备份存储。

###### 磁盘

###### 光盘

##### 六、RAID（独立磁盘冗余阵列）

把多个相对便宜的磁盘组合起来，成为一个磁盘组，配合数据分散排列的设计，提升数据的安全性和整个磁盘系统效能。

利用多磁盘数据传输率

通过数据冗余与校验实现可靠性。

###### 1.RAID应用的主要技术

分块技术、交叉技术和重聚技术

###### 2.RAID 0级（无冗余和无校验的数据分块）

RAID 0原理是把连续的数据分散到多个磁盘上存取，数据请求被多个磁盘并行执行，每个磁盘执行属于自己的那部分数据请求。这种数据上的并行操作充分利用总线带宽，显著提高磁盘整体存取性能。

优点：具有最高的I/O性能和最高的磁盘空间利用率，易管理。

缺点：不提供数据冗余，一旦数据损坏，损坏的数据将无法得到恢复。

###### 3.RAID 1级（磁盘镜像阵列）

由磁盘对组成，每个工作盘都有其对应的镜像盘，上面保存着与工作盘完全相同的数据拷贝，具有最高的安全性，但磁盘空间利用率只有50%。RAID 1主要用于存放系统软件、数据及其他重要文件

###### 4.RAID 2级（采用纠错海明码的磁盘阵列）

RAID 2采用海明码纠错技术，用户增加校验盘来提供纠错和验借功能，磁盘驱动器组中的第1、2、4……2n个磁盘驮运器是专门的校验盘，用于校验和纠错，其余的用于存放数据。RAID 2最少要3台磁盘驱动器方能运作。

###### 5.RAID 3级（采用带奇偶校验码的并行传送）

RAID 3把数据分成多个“块”，按照奇偶校验算法存放在N+1个硬盘上，实际数据占用的有效空间为N个硬盘空间的总和，第N+1个硬盘上存储的数据是校验容错信息。

当N+1硬盘中的一个硬盘出现故障时，从其他N个硬盘中可以恢复原始数据。所以RAID3，安全性可以得到保障。

RAID 3比较适合大文件类型且安全性要求较高的应用，如视频编辑、硬盘播放机和大型数据库等。

###### 6.RAID 4级（带奇偶校验的独立磁盘结构）

它与RAID 3很像，不同的是，它对数据的访问是按数据块进行的（一个数据块是一个完整的数据集合，比如一个文件就是一个典型的数据块。一个数据块存储在一个磁盘上），也就是按磁盘进行的，每次是一个盘。RAID 4使用一块磁盘作为奇偶校验盘，每次写操作都需要访问奇偶盘，这时奇偶校验盘成为写操作的瓶颈。

###### 7.RAID 5级（无独立校验盘的奇偶校验码磁盘阵列）

RAID 5把数据和奇偶校验信息存储到组成RAID5的各个磁盘上，并且奇偶校验信息和相对的数据分别存储于不同的磁盘上。

当RAID5的一个磁盘数据损坏后，利用的数据和相应的奇偶校验信息去恢复被损坏的数据。

RAID5磁盘空间利用率较高：（N-1）/N

RAID4和RAID5使用了独立存取技术，阵列中每一个磁盘都相互独立地操作，所以I/O请求可以并行处理。该技术非常适合于I/O请求率高的应用，不太适用于要求高数据传输率的应用。

###### 8.RAID 6级（具有独立的数据硬盘与两个独立的分布式校验方案）

RAID 6技术是在RAID 5的基础上，为进一步加强数据保护而设计的一种RAID方式，是一种扩展RAID 5等级。

与RAID 5的不同之处：除了每个硬盘上都有同级数据XOR(异或)校验区外，还有一个针对每个数据块的XOR校验区。当前盘数据块的校验数据不是存在当前盘而是交错存储的。每个数据块有了两个校验保护，所以RAID 6的数据冗余性能相当好。

但是，由于增加了一个校验，所以写入的效率较RAID 5还差，而且控制系统地更为复杂，第二块的校验区也减少了有效存储空间。

###### 9.RAID 10级

把RAID 0和RAID 1技术结合起来，即RAID1+0是磁盘分段及镜像的结合，结合了RAID0及RAID1的优点。

它采用两组RAID0的磁盘阵列互为镜像，也就是它们之间又成为一个RAID1阵列。

##### 七、Cache（高速缓冲存储器）

高速缓存位于主存与CPU之间的一级存储器，由静态存储芯片（SRAM）组成，容量比较小，速度比主存高得多，接近于CPU的速度，单位成本比内存高。Cache存储了频繁访问内存的数据。

###### 1.Cache原理、命中率、失效率

使用Cache改善系统性能的主要依据是程序的局部性原理

（1）程序局部性原理

时间局部性

当前访问的指令，不久的将来可能还会访问

空间局部性

当前访问了此条指令，有可能马上会访问它附近的指令

（2）命中率

访问的指令能够在Cache中找到，称之为命中。单位时间内在Cache中命中的数量与执行指令的数量比称为命中率

Cache的访问命中率为h（通常1-h就是Cache的失效率），Cache的访问周期时间是t1，主存储器的访问周期时间是t2，则整个系统的平均访存时间就是：t3=h\*t1+(1-h)\*t2

###### 2.Cache存储器的映射机制

分配给Cache的地址存放在一个相联存储器（CAM）中。CPU发生访存请求时，会先让CAM判断所要访问的数据是否在Cache中，如果命中就直接使用。这个判断过程就是Cache地址映射，这个速度应该尽可能快。

常见的映射方法有：

直接映射

是一种多对一的映射关系，但一个主存块只能够复制到Cache的一个特定位置上去。

Cache的行号i和主存的块号j有函数关系：i=j%m （其中m为Cache总行数）

**例：**某Cache容量为16KB（可用14位表示），每行的大小为16B（即可用4位表示），则说明其可分为1024行（可用10位表示）。主存地址的低4位为Cache的行内地址，中间10位为Cache行号。

如果内存地址为1234E8F8H，那么最后4位就是1000（对应十六进制数的最后一位），而中间10位，则应从E8F(1110 1000 1111)中获取，得到10 1000 1111。内存地址为1234E8F8H的单元装入的Cache地址为10 1000 1111 1000

全相联映射

将主存中任一主存块能映射到Cache中任意行（主存块的容量等于Cache行容量）。

根据主存地址不能直接提取Cache页号，而是将主存块标记与Cache各页的标记逐个，直到找到标记符合的页（访问Cache命中），或者全部比较完后仍无符合的标记（访问Cache失败）。

主存块标记与Cache各页的标记逐个比较，所以这种映射方式速度很慢，失掉了高速缓存的作用，这是全相联映射方式的最大缺点。如果让主页标记与各Cache标记同时比较，则成本太高。

组相联映射

是前两种方式的折中方案。它将Cache中的块再分成组，各组之间是直接映像，而组内各块之间则是全相联映像。

主存地址＝区号+组号+组内块号+块内地址号

例：容量为64块的Cache采用组相联方式映射，字块大小为128个字，每4块为一组。若主存容量为4096块，且以字编址，那么主存地址应该为多少位？主存区号为多少位？

首先根据主存块与Cache块的容量必须一致，得出内存块的大小也是128个字，因此共有128\*4096个字，即219(27\*212)个字，因此需19位主存地址；其中：块内地址号为7位，以表示128个字。一组为4块，则组内块号用2位表示。Cache容量为64块，共分16组，故组号需要4位地址表示。剩余的即为区号，为6位。

###### 3.Cache淘汰算法

当Cache数据已满，并且出现未命中情况时，就要淘汰一些老的数据，更新一些新的数据进入Cache。选择淘汰哪些数据的方法就是淘汰算法。常见的方法有三种：

随机淘汰算法

先进先出淘汰算法（FIFO）

最近最少使用淘汰算法（LRU）

其中平均命中率最高的是LRU算法

###### 4.Cache存储器的写操作（与主存同步的问题）

在使用Cache时，需要保证其数据与主存一致，因此在写Cache时就要考虑与主存间的同步问题，通常使用以下三种算法：

写直达：当Cache写命中时，Cache与主存同时发生写修改

写回：当CPU对Cache写命中时，只修改Cache的内容而不立即写入内存，当此行被换出才写回主存

标记法：数据进入Cache后，有效位置1；当CPU对该数据修改时，数据只写入主存并将该有效位置0.当要从Cache中读取数据时要测试其有效位，若为1则直接从Cache中取数，否则从主存中取数。

#### I/O控制方式

##### 一、程序I/O方式（程序查询方式，已淘汰，基本不用）

由于早期计算机无中断机构，处理机对I/O设备的控制采取程序I/O方式，或称为忙--等待方式，即在处理机向控制器发送一条I/O指令启动输入设备输入数据时，要同时把状态寄存器中的忙/闲标志置为1。然后便不断测试标志。当为1时，表示输入机尚未输完一个字，处理机应继续对该标志测试，直到它为0，表明数据已输入到控制器的数据寄存器中，于是处理机将数据取出送入内存单元，便完成了一个字的I/O。

在程序I/O方式中，由于CPU高速，I/O设备低速致使CPU极大浪费。

##### 二、中断驱动I/O方式

当某进程要启动某个I/O设备时，便由CPU向相应的设备控制器发出一条I/O命令，然后立即返回继续执行原来的任务。设备控制器于是按照命令的要求去控制指定的I/O设备。这时CPU与I/O设备并行操作。

中断驱动方式在I/O设备输入数据的过程中，无需CPU干预，而是当I/O设备准备就绪时“主动”通知CPU。才需CPU花费极短的时间去进行中断处理。从而大大地提高了整个系统的资源利用率及吞吐量，特别是CPU的利用率。但每次中断一次仅能传输一个字（节）。

##### 三、直接存储器访问DMA I/O控制方式

虽然中断方式比程序I/O方式更有效，但它仍是以字（节）为单位进行I/O的，每当完成一个字（节）的I/O时，控制器便要请求一次中断。极其低效的。因此引入了直接存储器访问方式。

该方式的特点是：数据传输的基本单位是数据块；所传送的数据是从设备直接送入内存的，或者相反；仅在传送一个或多个数据块的开始和结束时，才需CPU干预，整块数据的传送是在控制器的控制下完成的。可见DMA方式又是成百倍地减少了CPU对I/O的干预，进一步提高了CPU与I/O设备的并行操作程度

##### 四、I/O通道控制方式

I/O通道是一种特殊的处理机。它具有执行I/O指令的能力，并通过执行通道（I/O）程序来控制I/O操作。但I/O通道又与一般的处理机不同，一是其指令类型单一，只能执行I/O操作有关的指令；二是通道没有自己的内存，与CPU共享内存。

根据信息交换方式，通道分为三种类型：

字节多路通道（Byte Multiplexor Channel）

数组选择通道（Block Selector Channel）

数组多路通道（Block Multiplexor Channel）

### 安全性、可靠性与性能评测

#### 数据安全与保密

##### 一、加密体系

###### 1．对称密码体制

对称密码体制又称为秘密密钥体制（私钥密码体制），加密和解 密采用相同的密钥（或者可以通过一个推导出另一个）。 优点：加密速度快，通常用来加密大批量的数据。 缺点：需要管理的密码多。

常见的对称密钥技术

**DES：**是一种迭代的分组密码，输入/输出都是64位，使用一个 56位的密钥和附加的8位奇偶校验位。攻击DES的主要技术是 穷举。 由于DES的密钥长度较短，为了提高安全性，出现了使用112 位密钥对数据进行三次加密的算法，称为3DES。

**IDEA算法：**其明文和密文都是64位，密钥长度为128位

###### 2 . 非对称密钥技术（公钥算法）

非对称密钥技术是指加密密钥和解密密钥完全不同，并且不可能 从任何一个推导出另一个。

优点：适应开放性的使用环境，可以实现数字签名与验证。

最常见的非对称密钥技术是RSA。它的理论基础是数论中大素数分解极其困难。

使用RSA来加密大量的数据则速度太慢，因此RSA广泛用于密 钥的分发、数字签名中。

##### 身份认证技术与数字签名

数字签名就是只有信息的发送者才能产生的别人无法伪造的一段 数字串，这段数字串同时也是对信息的发送者发送信息真实性的 一个有效证明。 数字签名使用了公钥加密领域的技术实现，用于鉴别数字信息的 方法。一套数字签名通常定义两种运算，一个用于签名，另一个用于验证

###### 1.数字签名算法：

Hash签名  DSS签名  RSA签名

###### 2.数字签名原理：

(1)发送者首先将原文用Hash函数生成128位的消息摘要。

(2)发送者用自己的私钥对摘要再加密，形成数字签名，把加密后 的数字签名附加在要发送的原文后面。

(3)发送者将原文和数字签名同时传给对方。

(4)接收者对收到的信息用Hash函数生成新的摘要，同时用发送者 的公开密钥对消息摘要解密。

(5)将解密后的摘要与新摘要对比，如两者一致，则说明传送过程 中信息没有被破坏或篡改。

##### 数字证书

CA是数字证书的签发机构，它是PKI的核心。CA是负责签发证书、 认证证书、管理已颁发证书的机关。

CA要制定政策和具体步骤来验证、识别用户身份，并对用户 证书进行签名，以确保证书持有者的身份和公钥的拥有权。

CA是可以信任的第三方

###### **数字证书的内容 CA《A》=CA﹛V，SN，AI，CA，UCA，A，UA，Ap，Ta﹜**

V---证书版本号

SN---证书序列号

AI---用于对证书进行签名的算法标识

CA---签发证书的CA机构的名字

UCA---签发证书的CA的惟一标识符

A--用户A的名字

UA---用户A的惟一标识

Ap---用户A的公钥

Ta---证书的有效期

##### 电子商务安全

###### 1.SSL（端口号443）

SSL(安全套接层协议)及其继任者TLS (传输层安全协议)是一种 安全协议，为网络通信及数据完整性提供安全保障。 SSL和TLS是工作在传输层的安全协议, 在传输层对网络连接进 行加密。

SSL协议可分为两层

SSL记录协议（SSL Record Protocol）

它建立在可靠的传输协议（如TCP）之上，为高层协议提供 数据封装、压缩、加密等基本功能的支持。

SSL握手协议（SSL Handshake Protocol）

它建立在SSL记录协议之上，用于在实际的数据传输开始前， 通讯双方进行身份认证、协商加密算法、交换加密密钥等。

###### SET（Secure Electronic Transaction，安全电子交易）协议

SET协议称为安全电子交易协议。 美国Visa和MasterCard两大信用卡组织共同制定了应用于Internet 上的以银行卡为基础进行在线交易的安全标准--SET。

它采用公钥密码体制和X.509数字证书标准，保障网上购物信息的 安全性。

###### 3.HTTPS（安全套接字层上的超文本传输协议）

是以安全为目标的HTTP通道，简单讲是HTTP的安全版。

HTTPS是工作在应用层的协议。

###### 4.PGP

PGP是一个基于RSA公钥加密体系的邮件加密软件。

PGP可用于文件存储的加密。

PGP承认两种不同的证书格式：PGP证书和X.509证书。

#### 可靠性评价

##### **一、可靠性计算**

###### IMG_256串联系统

各个子系统的可靠性分别用R1，R2，…，Rn表示

系统的可靠性为： R=R1×R2 …. ×Rn

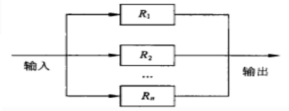
系统的失效率为： λ = λ1 + λ2+…+ λn

λ=1/MTBF（平均无故障时间）

###### 并联系统

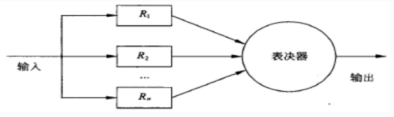
假如一个系统由2个子系统组成，只要有一个子系统能够正常工 作，系统就能正常工作，设系统各个子系统的可靠性用R1，R2 ，…，Rn表示

系统的可靠性为： R= 1- (1- R1 ) ×（1- R2）×…×（1- Rn )

系统的失效率为： （公式中n为子系统数量）

###### 模冗余系统

m模冗余系统由m个（m=2n+1为奇数）相同的子系统和一个 表决器组成，经过表决器表决后，m个子系统中占多数相同结果 的输出作为系统的输出。

m模冗余系统的可靠性为：

公式中的

## 第2章 数据结构

### 线性结构

#### 线性表

##### 线性表

###### 线性表定义

线性表是n个元素的有限序列，通常记为(a1，a2，…，an)。

特点：

存在惟一的表头和表尾。

除了表头外，表中的每一个元素均只有惟一的直接前驱。

除了表尾外，表中的每一个元素均只有惟一的直接后继。

###### 2．线性表的存储结构

(1)顺序存储

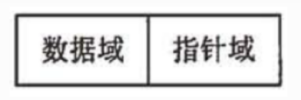
是用一组地址连续的存储单元依次存储线性表中的数据元素，从而使得逻辑关系相邻的两个元素在物理位置上也相邻。

优点：可以随机存取表中的元素

缺点：插入和删除操作需要移动大量的元素。

在线性表的顺序存储结构中，第i个元素ai的存储位置为： LOC(ai)= LOC(a1)+(i-1)×L 其中LOC(a1)是表中第一个元素的存储位置，L是表中每个元素 所占空间的大小。

(2)链式存储

链式存储是指用结点来存储数据元素，结点的空间可以是连续的，也可以是不连续的，因此存储数据元素的同时必须存储元素之间的逻辑关系。

结点空间只有在需要的时候才申请，无须事先分配。

优点：插入和删除操作不需要移动元素，操作方便。

缺点：增加了存储空间开销，不能随机访问任一结点。

其他几种链表结构

1. 双向链表：每个结点包含两个指针，指明直接前驱和直 接后继元素，可在两个方向上遍历链表。
2. 循环链表：表尾结点的指针指向表中的第一个结点，可 在任何位置上开始遍历整个链表。
3. 静态链表：借助数组来描述线性表的链式存储结构。

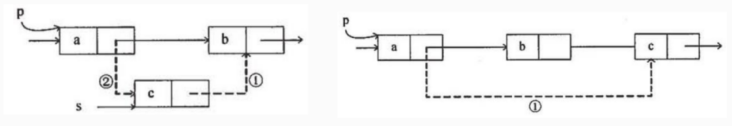
###### 3．线性表的插入和删除运算

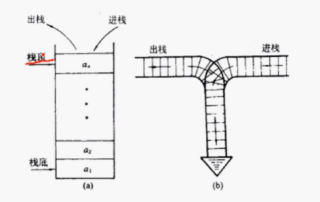
(1)基于顺序存储结构的运算

插入元素前要移动元素以挪出空的存储单元，然后再插入元素。

删除元素时同样需要移动元素，以填充被删除出来的存储单元。

(2)基于链式存储结构的运算

在链式存储结构下进行插入和删除，其实质是对相关指针的修改。



##### 二、栈

栈是只能通过一端来实现数据存储和检索的一种线性表。

栈进行插入和删除操作的一端称为栈顶，另一端称为栈底。

栈的修改是按先进后出的原则进行的。又称为先进后出的线性表。

###### 栈的存储结构

栈的存储结构有顺序存储和链式存储。

栈的顺序存储

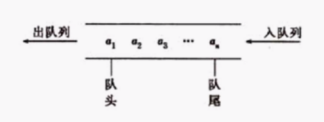
是指用一组地址连续的存储单元依次存储自栈顶到栈底的数据元素，同时附设指针top指示栈顶元素的位置。

栈的链式存储

用链表作为存储结构的栈也称为链栈。由于栈中元素的插入和删除仅在栈顶一端进行，因此不必设置头结点，链表的头指针就是栈顶指针。

##### 三、队列

队列是一种先进先出(FIFO)的线性表，它只允许在表的一端插入元素，而在表的另一端删除元素。

在队列中，允许插入元素的一端称为队尾(rear)，允许删除元 素的一端称为队头(front)。

###### 队列的存储结构

队列的存储结构有顺序存储和链式存储两种。

队列的顺序存储结构

是利用一组地址连续的存储单元存放队列中的元素。由于队列中元素的插入和删除限定在队列的两端进行，因此设置队头指针和队尾指针，分别指示当前的队首元素和队尾元素。

队列链式存储

用链表表示的队列简称为链队列。

为了便于操作，给链队列添加一个头结点，并令头指针指向头结点。队列为空的判定条件是：头指针和尾指针的值相同，且均指向头结点。

##### 四、串

串是仅由字符构成的有限序列，是取值范围受限的线性表。

一般记为S=‘a1 a2...an’，其中S是串名，a1a2…an是串值。

###### 1.串的几个基本概念

空串

长度为零的串，空串不包含任何字符。

空格串

由一个或多个空格组成的串。

子串

由串中任意长度的连续字符构成的序列。含有子串的串称为主串。子串在主串中的位置指子串首次出现时，该子串的第一个字符在主串中的位置。空串是任意串的子串。

如:主串：abcbcc 子串：cb

串相等

指两个串长度相等且对应位置上的字符也相同。

串比较

两个串比较大小时以字符的 ASCII码值作为依据。比较操作从两个串的 第一个字符开始进行，字符的ASCII码值大 者所在的串为大，若其中一个串先结束，则以串长较大者为大。

几个关键的ASCII码

a 97

A 65

0(数字) 48

###### 2.串的存储结构

每个字符串的最后要增加个串结束标志 \0 。

(1) 串的顺序存储

用一组地址连续的存储单元来存储串值的字符序列。

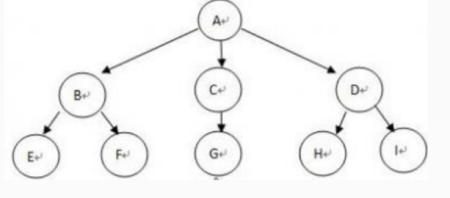
(2)串的链式存储

当用链表存储串中的字符时，每个结点中可以存储一个字符，也可以存储多个字符，要考虑存储密度问题。

### 树和二叉树

#### 树

树是n(n≥0)个结点的有限集合，n=0时称为空树，

在任一非空树中有且仅有一个称为根的结点。其余的结点可分为m(m≥0)个互不相交的子集T1，T2…，Tm， 其中每个子集本身又是一棵树，并称其为根结点的子树。

##### 一、树的基本概念

###### 双亲和孩子

###### 兄弟：

具有相同双亲的结点互为兄弟。

###### 结点的度：

一个结点的子树的个数记为该结点的度。

###### 树的度：

树中各结点的度的最大值

###### 叶子结点：

也称为终端结点，指度为零的结点。

###### 内部结点：

度不为零的结点称为分支结点或非终端结点。除 根结点之外，分支结点也称为内部结点。

###### 结点的层次：

根为第一层，根的孩子为第二层，依此类推。

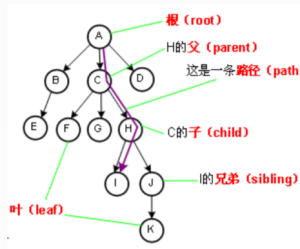
###### 树的高度：

一棵树的最大层次数记为树的高度（或深度）。

###### 有序（无序）树：

若将树中的结点的各子树看成是从左到右 具有次序的，即不能交换，则称该树为有序树，否则称为无序树。

###### 森林：

是m(m≥O)棵互不相交的树的集合

##### 二、树的存储结构

###### 标准存储结构

结点的数据

指向子结点的指针

###### 带逆存储结构

结点的数据

指向子结点的指针

指向其父结点的指针

##### 三、树的遍历

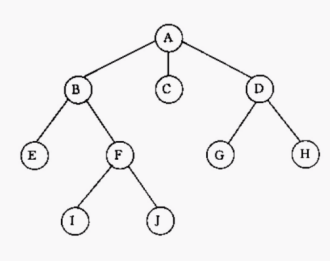
遍历是指对树中所有结点信息的访问，即依次对树中每个结点访问一次且仅访问一次。

如下图：

前序遍历A B E F I J C D G H

后序遍历E I J F B C G H D A

层次遍历A B C D E F G H I J

树是没有中序遍历的，中序遍历使用于二叉树

#### 二叉树

二叉树(BinaryTree)是n(n≥o)个结点的有限集合，它或者是空树 (n=0)，或者是由一个根结点及两棵互不相交的、分别称为左 子树和右子树的二叉树所组成。

##### 二叉树与树的区别：

二叉树的结点的最大度为2，而树中不限制结点的度。

二叉树的结点的子树要区分左子树和右子树

二叉树可以为空，即没有结点，树至少有一个结点。

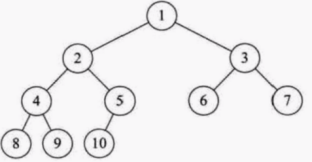
##### **一、二叉树的性质**

(1)二叉树第i层上的结点数目最多为2 i-1 (i≥1)。

(2)深度为k的二叉树至多有2 k -1个结点(k≥1)。

(3)在任意一棵二叉树中，若终端结点数为n0，度为2的结点 数为n2，则n0=n2+1。

(4)具有n个结点的完全二叉树的深度为⌊log2n」+1。（“⌊」”该符号为取下限整数）

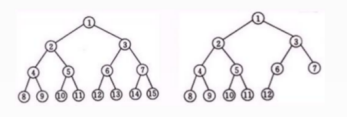
(5)对一棵有n个结点的完全二叉树的结点按层次自左至右进行 编号，则对任一结点i有：

若i=1，则结点 i 是二叉树的根，无双亲，若i>1，则其双 亲为⌊ i/2 」 。

若 2i>n，则结点i无左孩子，否则其左孩子为 2i。

若 2i+1>n，则结点i无右孩子，否则其右孩子为 2i+1。

（6）若深度为k的二叉树有2 k -1个结点，则称其为满二叉树。 深度为k、有n个结点的二叉树，当且仅当其每一个结点都与 深度为k的满二叉树编号从1至n的结点一一对应时，称之为完全二叉树。



##### 二、二叉树的存储结构

###### 1.顺序存储结构

对完全二叉树既简单又节省空间，而对于一般二叉树则不适用。

###### 2.链式存储结构

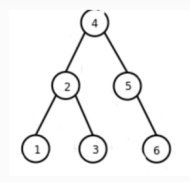
由于二叉树中结点包含有数据元素、左子树根、右子树根及双亲等信息，因此可以用三叉链表或二叉链袭来存储二叉树。链表的头指针指向二叉树的根结点。

###### **3.二叉树的遍历**

前序遍历 4 2 1 3 5 6 （先访问根、再访问左子树、最后右子树）

中序遍历 1 2 3 4 5 6 （先左子树、再访问根、最后右子树）

后序遍历 1 3 2 6 5 4 （先左子树、后右子树、最后根）

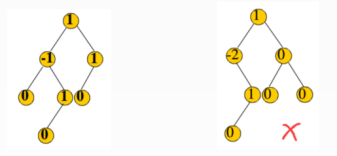


#### PSHULIBTQQS}M474B~90BTW二叉排序树（二叉查找树）

定义：或者是一棵空树，或者是具有下列性质的二叉树：

1. 若左子树不空，则左子树上所有结点的值均小于它的根结点的值；
2. 若右子树不空，则右子树上所有结点的值均大于或等于它的根结点的值；
3. 左、右子树也分别为二叉排序树；

#### 平衡二叉树（AVL树）

具有以下性质：它是一棵空树或它的左右两个子树的高度差的绝对值不超过1，并且左右两个子树都是 一棵平衡二叉树。

#### 线索树

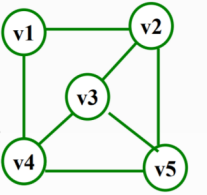
n个结点的二叉链表中含有n+1 (2n-(n-1)=n+1)个空指针域。利用二叉链表中的空指针域，存放指向结点在某种遍历次序下的前趋和后继结点的指针（这种附加的指针称为"线索"）。

#### 最优二叉树

给定n个权值作为n的叶子结点，构造一棵二叉树，若带权路径长度达到最小，称这样的二叉树为最优二叉树，也称为哈夫曼树(Huffman tree)。哈夫曼树是带权路径长度最短的树，权值较大的结点离根较近。

### 图

#### 图的定义

图G是由两个集合V和E构成的二元组，记作G=(V，E)，其中V是图中顶点的非空有限集合，E是图中边的有限集合。

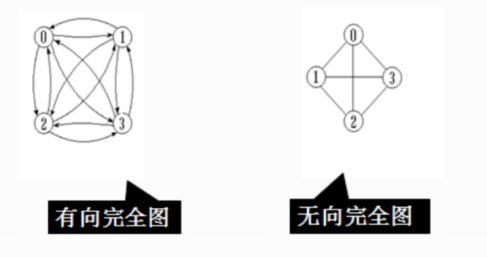
**有向图：**图G中的每条边都是有方向的，顶点间的关系用 <vi，vj>表示；

**无向图：**图G中的每条边都是无方向的；顶点 间的关系用(vi，vj)表示；

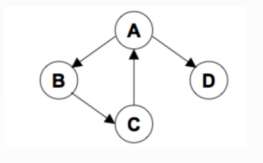
**完全图：**图G任意两个顶点都有一条边相连接；

**有向完全图：**n 个顶点的有向图有n(n-1) 条边。

**无向完全图：**n 个顶点的无向图有 n(n-1)/2 条边。



##### 一、度

顶点v的度是与它相关联的边的条数。记作TD(v)。

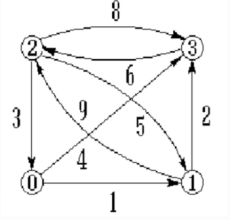
###### 入度

是以 v 为终点的有向边的条数, 记作 ID(v);

###### 出度

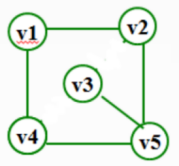
是以 v 为始点的有向边的条数, 记作 OD(v)。 在有向图中, 顶点的度等于该顶点的入度与出度之和。

##### 带权图

即边上带权的图。其中权是指每条边标上的具有某种含义的数值 （即与边相关的数）。

##### 三、连通图

在无向图中, 若从顶点v1到顶点v2有路径, 则称顶点v1与v2是连通的。如果图中任意一对顶点都是连通的，则称此图是连通图。无向图 G=(V,E) 是连通的，那么边的数目大于等于顶点的数目减 1



###### 强连通图

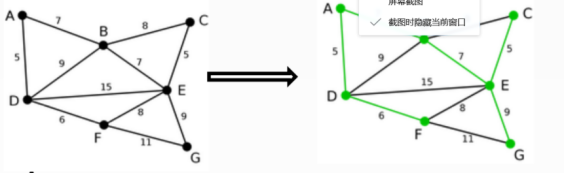
在有向图中，若对于每一对顶点vi和vj，都存在一条从vi到vj和从vj到vi的路径，则称此图是强连通图

##### 四、生成树（最小生成树）

是一个极小连通子图，它含有图中全部n个顶点，但只有n-1 条边。

如果在生成树上添加1条边，必定构成一个环。

若图中有n个顶点，却少于n-1条边 ，必为非连通图。

最小生成树：走遍所有节点，权值和是最小的，称为最小生成树

###### 最小生成树算法：

1.选择权值最小的边连起来

2.再选择第2小（或等于最小的）的边连起来

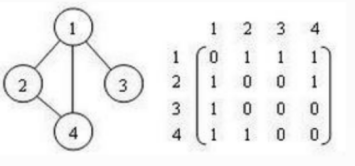
3.依次循环，但不能选择会构成环的路径

#### 图的存储结构

图的存储结构有：邻接矩阵、邻接表、十字链表、邻接多重表

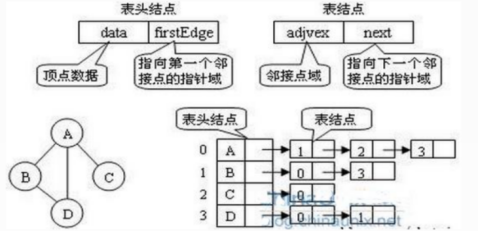
##### 一、邻接矩阵

对于一个具有n个结点的图，可以使用n\*n的矩阵(二维数组)来表示它们间的邻接关系。



##### 二、邻接表

邻接表由表头结点和表结点两部分组成，其中图中每个顶点均对应一个存储在数组中的表头结点。如这个表头结点所对应的 顶点存在相邻顶点，则把相邻顶点依次存放于表头结点所指向 的单向链表中。



#### 5IHUF@TWUS9B4SW6NDHTXAI图的遍历

##### 一、深度优先搜索( DFS )

访问起始点 v;

若v的第1个邻接点没访问过，深度遍历此邻接点；

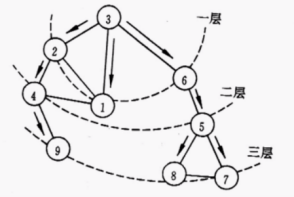
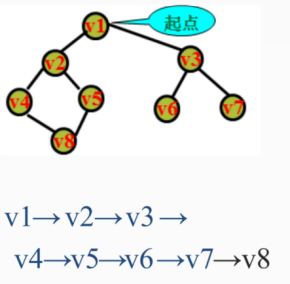
若当前邻接点已访问过，再找v的第2个邻接点重新遍历。

##### 二、广度优先搜索( BFS )

在访问了起始点v之后，依次访问 v的邻接点；

然后再依次（顺序）访问这些点（下一层）中未被访问过的邻接点；

直到所有顶点都被访问过为止。



## 第3章 操作系统

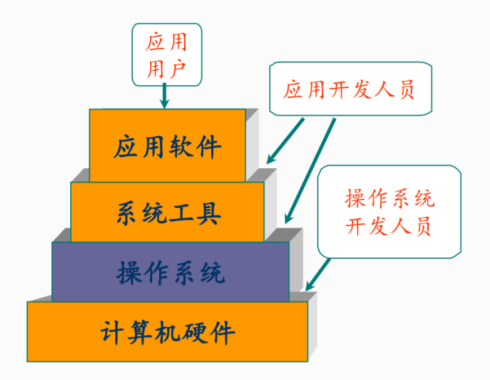
### 操作系统基础知识

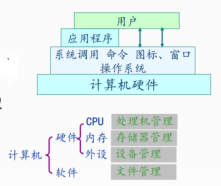
#### 一、操作系统定义

操作系统是直接控制和管理计算机硬件、软件资源，合理地对 各类作业进行调度，以方便用户使用的程序集合。



#### OS在计算机中的地位





#### 三、OS的作用

作为用户和计算机间的接口

作为计算机系统资源的管理者

实现了对计算机资源的抽象

#### 四、操作系统分类

批处理操作系统

分时操作系统

实时操作系统

网络操作系统

分布式操作系统

#### 五、操作系统的功能

处理机管理功能

存储器管理功能

设备管理功能

文件管理功能

用户接口

OS定义： OS是直接控制和管理计算机硬件、软件资源，合理地 对各类作业进行调度，以方便用户使用的程序集合

### 处理机管理（进程管理）

#### 一、进程的定义

进程：程序关于某个数据集合的一次执行过程。

##### 1.进程的特征（与程序比较）

###### (1) 结构特征

进程控制块(PCB) + 程序 + 数据 = 进程实体

###### (2) 动态性--最基本特征

进程：进程实体的一次执行过程，有生命周期。

程序：程序是一组有序指令的集合，是静态的概念。

##### 2.进程的三种基本状态

###### (1)就绪状态(Ready)

进程已获得除CPU之外的所有必需的资源，一旦得到CPU控制权 ，立即可以运行。

###### (2)运行状态(Running)

进程已获得运行所必需的资源，它正在处理机上执行。

###### (3)阻塞状态(Blocked)

正在执行的进程由于发生某事件而暂时无法执行时，便放弃处 理机而处于暂停状态，称该进程处于阻塞状态或等待状态。

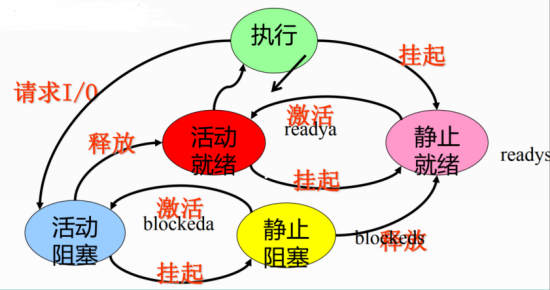
###### 进程的三种基本状态以及各状态之间的转换



就绪队列、阻塞队列

##### 3.进程的五种状态

引入挂起状态后，增加了挂起状态(静止状态)到非挂起状态( 活动状态)的转换，或者相反。



#### 二、进程互斥与同步

##### 1.进程间两种形式的制约关系

(1) 间接相互制约关系 --- 源于资源共享

(2) 直接相互制约关系 --- 源于进程合作

##### 2.临界资源

###### 临界资源（Critical Resource）

把一段时间内只允许一个进程访问的资源称为临界资源或独占资源

###### 临界区（Critical Section）

每个进程中访问临界资源的那段代码称为临界区。只要保证进程不同时进入到临界区，就能够保证进程互斥地访问临界资源。

#### 三、 信号量机制

信号量（Semaphore）是OS提供的管理公有资源的有效手段。

信号量是一个整数，当信号量大于等于零时，代表可供并发进程使用的资源数量，当信号量小于零时，表示处于阻塞态 进程的个数。

##### Wait 操作（也称为P操作）：

申请资源，减量操作，S.value:=S.value-1

当S.value<0时，表示资源分配完，进行自我阻塞。

##### Signal操作（也称为V操作）：

释放资源，增量操作，S.value:=S.value+1

当S.value≤=0，唤醒S.L链表中的等待进程。

#### 四、信号量的应用

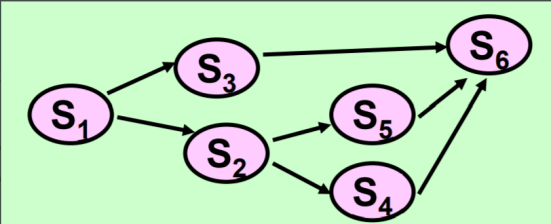
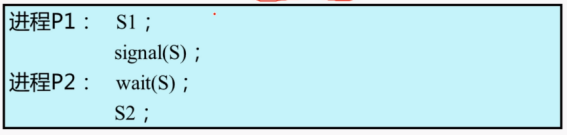
##### 1. 利用信号量实现进程互斥（模式）

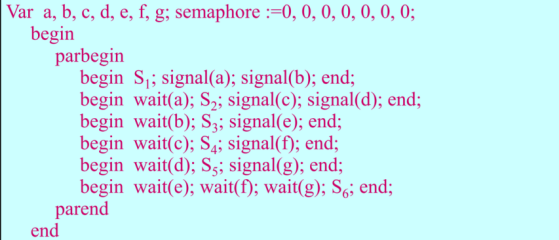
为使多个进程互斥的访问某临界资源，须为该资源设置一互斥信号量mutex，并设其初始值为1，然后将各进程访问资源的临界区CS置于wait(mutex)和signal(mutex)之间即可。

##### 2. 利用信号量实现前驱关系（模式）

设有两个并发执行的进程P1和P2，P1中有语句S1，P2中有语句S2，希望在S1执行后再执行S2。

使进程P1和P2共享一个公用信号量S，并赋予其初值为0。





##### 3. 利用记录型信号量实现同步（模式）

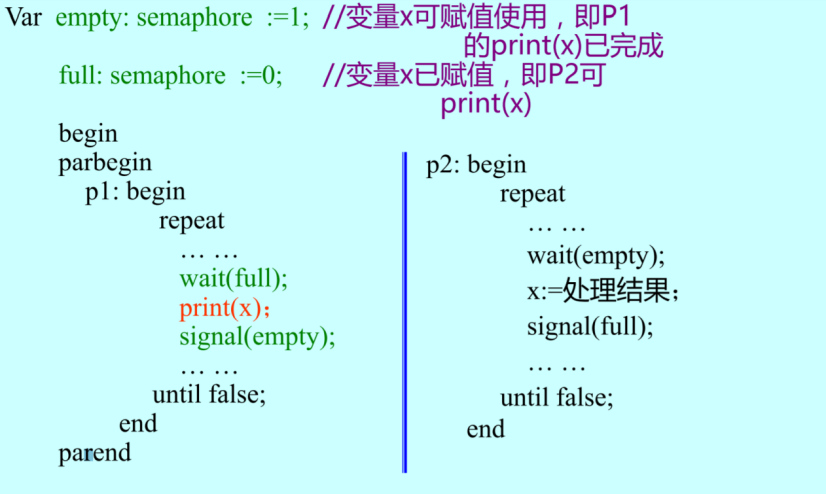
P1，p2两进程因合作完成一项任务而共用一个变量x。

进程p2将处理结果送入x；进程P1将x的结果打印。

即：p2：x=处理结果；

p1：Print(x)；

如何实现该合作关系?



#### 五、死锁

是指多个进程在运行过程中因争夺资源而造成的一种僵局，当进程处于这种状态时，若无外力作用，它们都将无法再向前推进

##### 1.产生死锁的原因

###### (1)竞争资源

当系统中供多个进程共享的资源如打印机、公用队列等，其数目不足以满足诸进程的需要时，会引起诸进程对资源的竞争而产生死锁。

###### (2)进程间推进顺序非法

进程在运行过程中，请求和释放资源的顺序不当，也同样会导致产生死锁。

##### 2.竞争资源引起进程死锁

可把系统中的资源分为两类：

a.可剥夺性资源：资源分配给进程后可以被高优先级的进程剥夺。如CPU、主存。 可剥夺性资源不会引起死锁。

b.不可剥夺性资源：分配给进程后只能在进程用完后才释放的资源。如磁带机、打印机等。

只有不可剥夺性资源才会产生死锁

##### 3.产生死锁的必要条件

###### (1)互斥条件

进程访问的是临界资源，即在一段时间内某资源只由一个进程占用。如果此时还有其他进程请求该资源，则请求者只能等待，直至占有该资源的进程用完释放。

###### (2)请求和保持条件

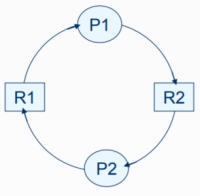
一进程在请求新的资源的同时，保持对已分配资源的占有。

###### (3)不剥夺条件

指进程已获得的资源，在未使用完之前，不能被剥夺，只能在使用完时由自己释放

###### (4)环路等待条件

指在发生死锁时，必然存在一个进程--资源的环形链。 即进程集合{P0,P1,P2,…,Pn}中的P0正在等待一个P1占用的资源；P1正在等待一个P2占用的资源，……， Pn正在等待一个已被P0占用的资源。



##### 处理死锁的基本方法

在系统中已经出现死锁后，则应及时检测到死锁的发生，并采取适当措施来解除死锁。

目前处理死锁的方法可归结为四种：

###### 预防死锁

是一种较简单和直观的事先预防方法。该方法是通过设置某些限制条件，去破坏产生死锁的四个必要条件的一个或几个，来预防发生死锁。 

摒弃“请求和保持”条件 

摒弃“不剥夺”条件 

摒弃“环路等待”条件

###### 避免死锁

该方法同样是属于事先预防的策略，这种方法不是预先加上各种限制条件以预防产生死锁的可能性，而是用某种方法去防止系统进入不安全状态，使死锁不致于最终发生。 

银行家算法避免死锁

###### 检测死锁

这种方法并不须事先采取任何限制性措施，也不必检查系 统是否已经进入不安全区。

此方法允许系统在运行过程中发生死锁，但可通过系统所设置的检测机构，及时的检测出死锁的发生，并精确的确定与死锁有关的进程和资源；然后采取适当的措施，从系统中将已发生的死锁清除掉。

###### 解除死锁

是与死锁检测相配套的一种措施。当检测到系统中已发生死锁时，须将进程从死锁状态中解脱出来。 

常用的实施方法是撤销或挂起一些进程，以便回收一些资 源，再将这些资源分配给已处于阻塞状态的进程，使之转为就绪状态，以继续运行。 

死锁的检测与解除措施，有可能使系统获得较好的资源利用率和吞吐量，但在实现上难度也最大。

###### 系统保证不发生死锁的资源数的计算方式

并发进程数\*（进程需要资源数-1）+1

### 存储管理

存储管理主要是指对内存的管理，负责内存分配和回收，内存的保护和扩充。

存储管理的目的是尽量提高内存的使用效率。

内存的分配方式有两种：连续的分配方式、离散的分配方式

#### 连续分配方式

指为一个用户程序分配一个连续的内存空间。

##### 1.单一连续分配

针对单线程操作系统,如DOS.将用户内存全部分配给单独的应用程序

##### 2.固定分区分配

将用户内存分成若干大小不同的区域,当有对内存需求不同的作业来时,将它分配到合适的内存区域。

它有可能出现内存空间的浪费，浪费的空间称为内碎片。

##### 3.动态分区分配

为把一个新作业装入内存，需按照一定的分配算法，从空闲分区表或空闲分区链中选出一分区分配给该作业。

常用的分配算法：

a.首次适应算法

b.循环首次适应算法

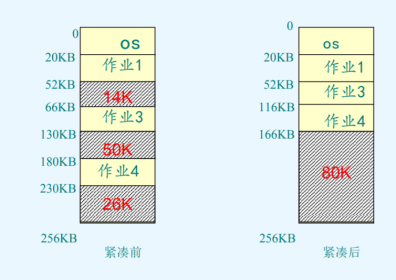
c.最佳适应算法

d.最坏适应算法

##### 4.可重定位分区分配

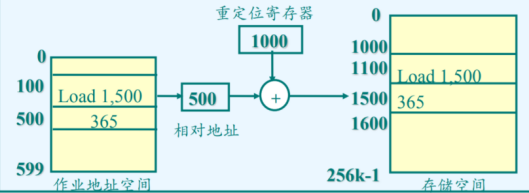
如果在系统中只有若干个小分区，即使它们的容量总和大于要装入的程序，但由于这些分区不相邻，也无法将程序装入内存。

解决方法：将内存中的所有作业 进行移动，使它们全部邻接，这样把原来分散的小分区拼接成大分区，这种方法称为“拼接”或 “紧凑”。



##### 动态重定位的实现

在动态运行时装入的方式，将相对地址转换为物理地址的工作在程序指令真正要执行时才进行。地址转换需要重定位寄存器的支持。程序执行时访问的内存地址是相对地址与重定位寄存器中的地址相加而成。



##### 5.对换与覆盖技术

###### （1）覆盖技术

一个作业的若干程序段或数据段的某些部分共享内存空间

###### （2）对换技术

把内存中暂时不能运行的进程或者暂时不用的程序和数据，调到外存上，以便腾出足够的内存空间，再把已具备运行条件的进程 和进程所需要的程序和数据调入内存。

对换的分类： 

整体对换(或进程对换)：以整个进程为单位 

页面对换或分段对换：以页或段为单位

#### 二、离散分配方式

连续分配方式会形成“碎片”，虽然可以通过“紧凑”解决，但开销大。如果允许将一个进程直接分散地装入许多不相 邻的分区中，则无需“紧凑”，由此产生离散分配方式。

分类： 

分页存储管理方式：离散分配的基本单位是页 

分段存储管理方式：离散分配的基本单位是段

##### 1.基本分页存储管理方式

###### 2PPF73M(NFH85VF`5AMSZ09(1)页面

分页式存储管理的原理: 将一个进程的逻辑地址空间分成若干个大小相等的片称为页面或页，并为各页加以编号，从 0开始。同时把内存空间分成与页面相同大小 的若干个存储块，称为块或页框。

在为进程分配内存时，以块为单位将进程的若干个页分别装入到多个可以不相邻的物理块中。进程的最后一页经常装不满一块而形成“页内碎片”。

###### (2)地址变换

若给定一个逻辑地址空间中的地址为A，页面大小为L，则页号P = INT[A/L] 页内地址d = [A] MOD L

例如：系统页面大小为1KB，设A=2170D，则 P=2，d=122

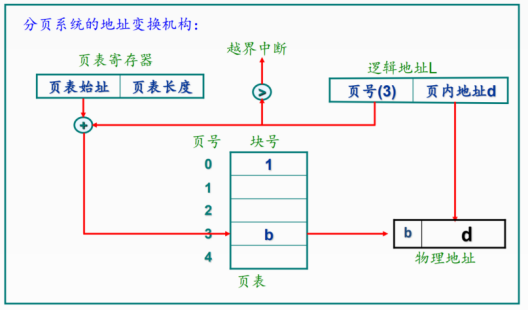
###### SL0T04%U~LB~0@H]78G_KEF(3)基本分页式存储管理的实现

进程的每一页离散地存储在内存的任一存储块中，为方便查找，系统为每一进程建立一张页面映像表，简称页表。

页表实现了从页号到物理块号的地址映射。

为了能将用户地址空间的逻辑地址变换为内存空间的物理地址 ，在系统中必须设置地址变换机构。

地址变换机构实现从逻辑地址到物理地址的转换，由于页内地址与物理地址是一一对应的，因此，地址变换机构的任务是借助于页表，将逻辑地址中的页号转换为内存中的物理块号。



###### (4)具有快表的地址变换机构

由于页表是存放在内存中的，CPU在每存取一个数据时，需要两次访问内存： 

第一次：访问页表，找到指定页的物理块号，将块号与页内 偏移量拼接形成物理地址。 

第二次：从第一次所得地址中获得所需数据，或向此地址中写入数据。 存储器利用率提高，处理器处理速度降低。

解决方法：在地址变换机构中，增设一个具有并行查寻能力的特殊高速缓冲寄存器，称为“联想存储器”或“快表”。

##### 2.基本分段式存储管理的实现

###### (1)段表

为使程序正常运行，须在系统中为每个进程建立一张段映射表 ，简称“段表”。每个段在表中占有一个表项。

段表结构：段号；段在内存中的起始地址(基址)；段长。

段表可以存放在寄存器中，但更多的是存放在内存中。

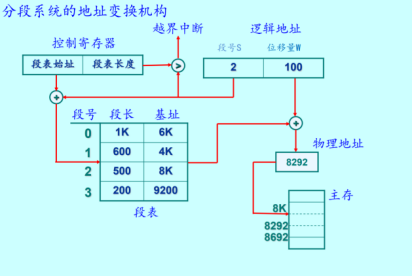
段表用于实现从逻辑段到物理内存区的映射。

###### (2)地址变换机构

在系统中设置段表寄存器，用于存放段表始址和段表长度，以实现从进程的逻辑地址到物理地址的变换。

当段表存放在内存中时，每访问一个数据，都需访问两次内存，降低了计算机的速率。

解决方法：设置联想寄存器，用于保存最近常用的段表项。



##### 3.分页和分段的主要区别

###### 相似点：

采用离散分配方式，通过地址映射机构实现地址变换

###### 不同点：

页是信息的物理单位，分页是为了满足系统的需要；段是信息的逻辑单位，含有意义相对完整的信息，是为了满足用户的需要。

页的大小固定且由系统确定，由系统把逻辑地址分为页号和页内 地址，由机器硬件实现；段的长度不固定，取决于用户程序，编 译程序对源程序编译时根据信息的性质划分。 

分页的作业地址空间是一维的；分段的作业地址空间是二维的。

##### 4.段页式存储管理

分段和分页存储管理方式各有优缺点。把两者结合成一种新的存储管理方式--段页式存储管理方式，具有两者的长处。

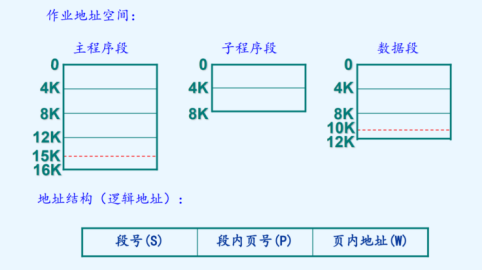
###### (1)基本原理

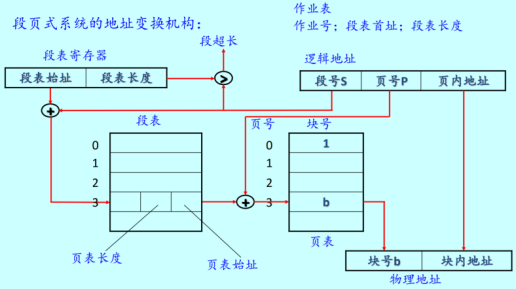
先将用户程序分成若干段，再把每个段分成若干页，并为每个段赋予一个段名。

基本段页式存储管理：把作业的所有段装入内存方可运行。

请求段页式存储管理：没必要把整个作业装入内存，可把作业的几段或几页装入内存即可运行。

段页式系统地址结构：段号；段内页号；页内地址。





在段页式系统中，为了获得一条指令或数据，需访问三次内存：

第一次：访问内存中的段表，取得页表始址

第二次：访问内存中的页表，取得该页所在的物理块号，将块号与页内地址形成物理地址

第三次：根据第二次所得的地址，取出指令或数据

缺点：访存次数增加两倍

解决方法：增设高速缓冲寄存器

##### 5.页面置换算法

最佳置换算法是一种理想化的算法，具有最好的性能，但难于实现。先进先出置换算法最直观，但可能性能最差，故应用极少。

###### (1)最佳置换算法

最佳置换算法其所选择的被淘汰页面，将是以后永不再用的，或许是在最长( 未来)时间内不再被访问的页面。

优点：保证获得最低的缺页率

缺点：无法预知一个进程在内存的若干个页面，哪个在未来最长时间内不再被访问。

算法无法实现，但可评价其他算法。

###### (2)先进先出置换算法

算法总是淘汰最先进入内存的页面，即选择在内存中驻留时间最久的页面予以淘汰。

算法实现简单，只需把一个进程已调入内存的页面，按先后次序链接成一个队列，并设置一个指针(替换指针)，使它总是指向最老的页面。

算法与进程的实际运行规律不相适应，因为进程中的某些页面经常被访问，但先进先出置换算法不能保证这些页面不被淘汰。

Belady现象

如果对—个进程未分配它所要求的全部页面，有时就会出现分配的页面数增多但缺页率反而提高的异常现象。发生在FIFO（先进先出）置换算法。

###### (3)最近最久未使用(LRU)置换算法

算法根据页面调入内存后的使用情况进行决策。由于无法预测各页面将来的使用情况，只能利用“最近的过去”作为“最近的将来”的近似，因此，LRU置换算法是选择最近最久未使用的页面予以淘汰。

该算法赋予每个页面一个访问字段，用来记录一个页面自上次被访问以来所经历的时间t，当需淘汰一个页面时，选择现有页面中其t值最大的，即最近最久未使用的页面予以淘汰。

### 设备管理

计算机系统的一个重要组成部分是I/O系统。

I/O系统包括:  输入、输出设备  存储功能的设备  设备控制器

#### 一、设备管理的概念

**设备管理程序提供下述功能:**

提供和进程管理系统的接口

进行设备分配

实现设备和设备之间、设备和CPU之间的并行操作

进行缓冲区管理。

#### 二、 I/O控制方式

(1) 程序I/O方式

(2) 中断控制I/O方式

(3) 直接存储器访问(DMA) 方式

(4) I/O通道控制方式 

字节多路通道 

选择通道 

成组多路通道

#### 三、缓冲管理

**引入缓冲区的主要原因归结为以下几点： **

缓和CPU与I/O设备间速度不匹配的矛盾。 

减少对CPU的中断频率，放宽对CPU中断响应时间的限制 

提高CPU和I/O设备之间的并行性。

##### (1) 单缓冲（Single Buffer）

在单缓冲情况下，每当用户进程发出一I/O请求时，OS便在主存中为之分配一缓冲区。

在字符设备输入时，缓冲区用于暂存用户输入的一行数据，在输入期间，用户进程被挂起以等待数据输入完毕；在输出时，用户进程将一行数据输入到缓冲区后，继续执行处理。当用户进程已有第二行数据输出时，如果第一行数据尚未被提取完毕，则此时用户进程应阻塞。

##### (2)双缓冲（Double Buffer）

为了加快输入和输出速度，提高设备利用率，人们又引入了双缓冲区机制，也称为缓冲对换（Buffer Swapping）。在设备输入时，先将数据送入第一缓冲区，装满后便转向第二缓冲区。此时OS可以从第一缓冲区中移出数据，并送入用户进程。

##### (3)循环缓冲区

循环缓冲有多个大小相同的缓冲区，

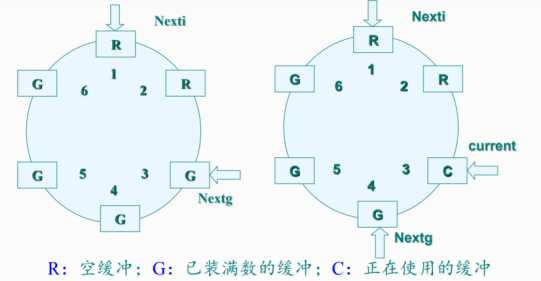
缓冲区有三种类型： 

用于装输入数据的空缓冲区R 

已装满数据的缓冲区G 

计算进程正在使用的现行工作缓冲区C

循环缓冲的组成示意图:



##### (4)缓冲池

对于既可输入又可输出的公用缓冲池，至少应含三种类型缓冲区：

1. 空缓冲区；
2. 装满输入数据的缓冲区；
3. 装满输出数据的缓冲区；

#### 四、设备的分配

##### 1、设备分配原则

###### 静态分配

一次性分配进程所需要的所有设备

###### 动态分配

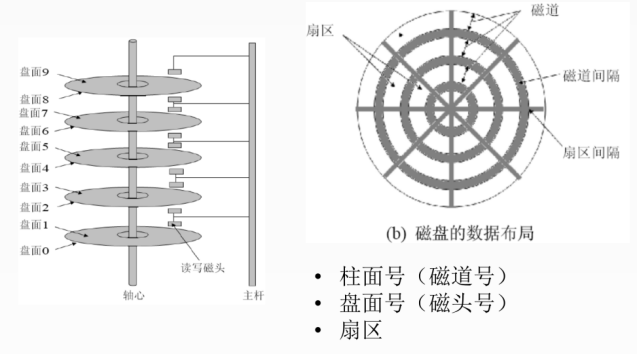
进程在运行过程中,根据系统申请需要实时分配相应的设备

##### 2、设备分配策略 

先请求先分配 

优先级高者先分配

#### 五、磁盘管理



##### 1.磁盘的访问时间 

###### 寻道时间Ts

把磁臂从当前位置移到指定磁道上所经历的时间 

###### 旋转延迟时间Tr

指定扇区移动到磁头下面所经历的时间。 

###### 传输时间Tt

数据从磁盘读出或向磁盘写入数据所经历的时间

在访问时间中，寻道时间和旋转延迟时间，通常是占据了访问时间的大头。适当地集中数据（不要太零散）传输，将有利于提高传输效率。

##### 2.磁盘调度算法 

###### 先来先服务（FCFS） 

根据进程请求访问磁盘的先后次序进行调度。 

优点：公平、简单，且每个进程的请求都能依次得到处理，不会出现某一进程的请求长期得不到满足的情况。 

缺点：未对寻道进行优化，致使平均寻道时间可能较长。仅适用于请求磁盘I/O的进程数目较少的场合。

###### 最短寻道时间优先（SSTF） 

优先满足访问磁道与当前磁头所在磁道距离最近的进程，以使每次的寻道时间最短。

问题：可能导致某些进程发生“饥饿”。因为只要不断有所要访问的磁道与磁头当前所在磁道的距离较近的新进程到达，就会出现“老进程饥饿”现象。这种调度算法不能保证平均寻道时间最短。

###### 扫描（SCAN）算法（电梯调度算法） 

SCAN算法中磁头移动的规律似电梯的运行，又称为电梯调度算法 。算法既能获得较好的寻道性能，又能防止进程饥饿，被广泛用于 大、中、小型机和网络中的磁盘调度。

问题：当磁头刚从里向外移动过某一磁道时，恰有一进程请求访问此磁道，这时该进程必须等待，待磁头从里向外，然后再从外向里扫描完所有要访问的磁道后，才处理该进程的请求，致使该进程的请求被严重地推迟。

###### 循环扫描CSCAN

为了减少请求进程的延迟，CSCAN算法规定磁头单向移动。若规定只自里向外移动，当磁头移到最外的被访问磁道时，磁头立即返回到最里的欲访磁道，即将最小磁道号紧接着最大磁道号构成循环，进行扫描。

#### 六、虚设备与SPOOLing技术

为缓和CPU的高速性与I/O设备低速性间的矛盾而引入了脱机输入、脱机输出技术。该技术是利用专门的外围控制机，将低速设备上的数据传送到高速磁盘上；或者相反。

这样就可以在主机的直接控制下实现脱机输入输出。此时外围操作与CPU对数据的处理同时进行，我们把这种在联机情况下实现的同时外围操作称为SPOOLing（Simultaneaus Periphernal Operating On—Line），或称为假脱机操作。

##### 1.SPOOLing系统的组成： 

###### 输入井和输出井。

是磁盘上开辟的两个大存储空间。

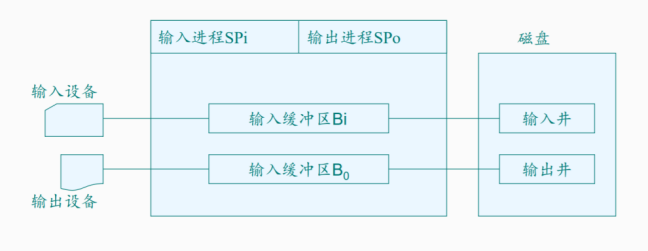


###### 输入缓冲区和输出缓冲区。

在内存中开辟两个缓冲区，输入缓冲区暂存由输入设备送来的数据，后送输入井；输出缓冲区暂存从输出井送来的数 据，后送输出设备。 

###### 输入进程和输出进程。

利用两个进程模拟脱机I/O时的外围处理机。



进程SPi模拟脱机输入时的外围控制机，将数据从输入机，通过输入缓冲区再送到输入井。CPU直接从输入井读数据入内存。 进程SPo模拟脱机输出时的外围控制机，把输出的数据从内存送到输出井，再待输出设备空闲时，将输出井中的数据，经过输出缓冲区送到输出设备上。

##### 2.SPOOLing系统的特点 

提高了I/O的速度。利用输入输出井模拟成脱机输入输出，缓和了CPU和I/O设备速度不匹配的矛盾。 

将独占设备改造为共享设备 

实现了虚拟设备功能。多个进程同时使用一台独占设备，虚拟成了多台设备。

### 文件管理

#### 一、文件和文件系统

文件是指具有文件名的若干相关元素的集合。 

现代OS中通过文件系统来组织和管理计算机中存储的数据； 

文件系统包括两方面：  负责管理文件的系统软件  被管理的对象--文件

##### 1.文件的结构

###### （1）文件的逻辑结构。

从用户观点出发所观察到的文件组织形式，是用户可以直接处理的数据及其结构，它独立于文件的物理特性，又称为文件组织。 

有结构文件

是指由一个以上的记录构成的文件，又把它称为记录式文件；根据记录的长度可分为定长记录文件；不定 长记录文件。

根据记录的组织方式分为下列文件： 

**顺序文件。**由一系列记录按某种顺序排列所形成的文件。通常是定长记录。 

**索引文件。**当记录可变长时，通常为之建立一张索引表，并为每个记录设置一个表项以加快对记录检索的速度。 

**索引顺序文件。**上述两种方式的结合。为文件建立一张索引表，为每一组记录中的第一个记录设置一个表项。 

**直接文件**

无结构文件

这是指由字符流构成的文件，故又称为是流式文件。

如果说大量的数据结构和数据库，是采用有结构的文件形式的话，则大量的源程序、可执行文件、库函数等，所采用的就是无结构的文件形式，即流式文件。其长度以字节为单位。对流式文件的访问，则是采用读写指针来指出下一个要访问的字符。 

UNIX 系统中，所有的文件都被看做是流式文件。

###### （2）文件的物理结构。

又称为文件的存储结构，是指文件在外存上的存储组织形式。与存储介质的存储性能和采用的外存分配方式有关。

由于磁盘具有可直接访问的特性，故当利用磁盘来存放文件时，具有很大的灵活性。

常用的外存分配方法有：连续分配、链接分配、索引分配，在一个系统通常只采用一种方法。

连续分配 

连续分配要求为每一个文件分配一组相邻的盘块。在采用该方式时，可把逻辑文件中的记录顺序的存储到邻接的各物 理块中，这样所形成的文件结构成为顺序文件结构，此时的物理文件称为顺序文件。这种分配方式保证了逻辑文件中的 记录顺序与存储器中文件占用盘块的顺序的一致性。 

随着文件的建立与删除不断进行，将产生很多外存的碎片，利用紧凑方法也可消除碎片。

链接分配

采用链接分配方式时，可通过在每个盘块上的链接指针，将同属于一个文件的多个离散的盘块链接成一个链表，把这样形成的文件称为链接文件。

链接分配方式虽然解决了连续分配方式所存在的问题，但又出现了另外两个问题： 

不能支持高效的直接存取。要对一个文件进行直接存取，需首先在FAT（在内存中的一种数据结构）中顺序的查找许多盘块号。 

FAT需占用较大的内存空间。当磁盘容量较大时，FAT可能要占用数MB以上的内存空间。这是令人难以忍受的。

索引分配

索引分配方式示意图 

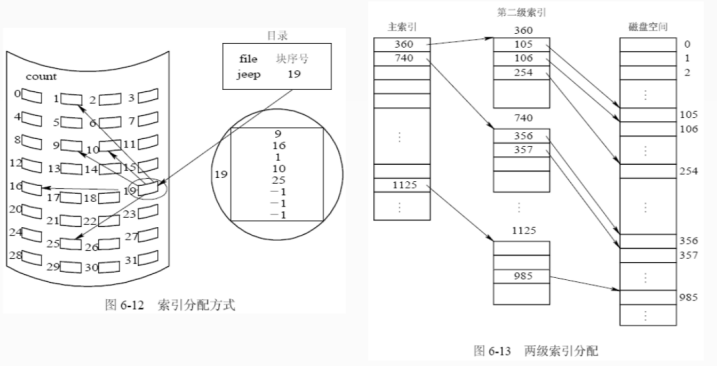
单级索引方式 

多级索引方式 

混合索引方式

索引分配方式的问题：

可能要花费较多的外存空间。每当建立一个文件时，便须为之分配一个索引块，将分配给该文件的所有盘块号记录于其中。



##### 2.存储空间的管理

###### 空闲表法 和 空闲链表法

###### 位示图法

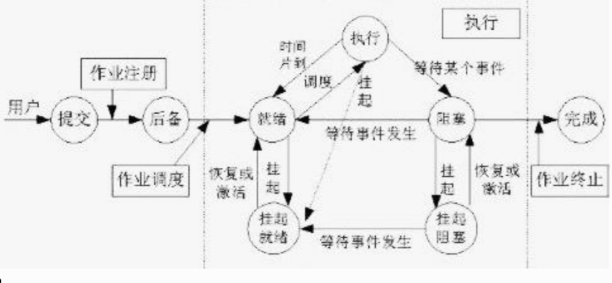


###### 成组链接法

### 作业管理

#### 一、作业状态

一个批处理型作业，从进入系统并驻留在外存的后备队列上开始，直至作业运行完毕，要经历提交、后备、执行和完成4个状态



#### 二、处理机调度

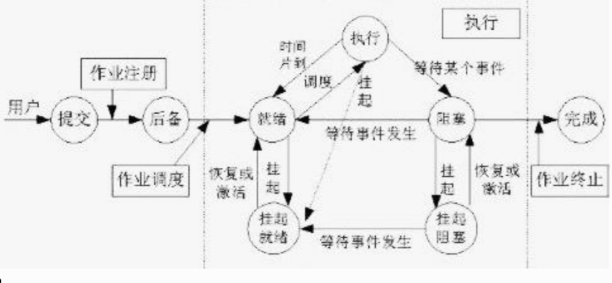
##### 1. 高级调度（High Scheduling）

也称为作业调度，是指在后备队列中选择一个或一批作业，为它们建立进程，分配必要的资源，使它们能够运行。

在批处理系统中，因作业进入系统后先驻留在外存，故需要有作业调度。 

在分时系统中为做到及时响应，命令或数据被直接送入内存，故不需作业调度。 

在实时系统中，不需作业调度。



##### 中级调度（Intermediate-Level Scheduling）

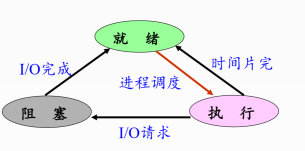
是为了提高内存利用率和系统吞吐量。

应使那些暂时不能运行的进程不再占用宝贵的内存资源，而将它们调到外存去等待，把此时的进程状态称为就绪驻外存状态或挂起状态。

##### 4.低级调度（Low Level Scheduling）

也称为进程调度或短程调度，用来决定就绪队列中的哪个进程应获得处理机。

为最基本的一种调度，三种类型OS中都必须有进程调度。



#### 三、调度算法

##### (1)先来先服务

##### (2)短作业（进程）优先调度算法

##### (3)高优先权优先调度算法

##### (4)高响应比优先调度算法

#### 四、用户接口

操作系统接口：命令接口、程序接口

## 程序语言基础

### 程序语言概述

#### 一、程序设计语言基本概念

##### 1．低级语言和高级语言

###### (1)低级语言

机器语言和汇编语言称为低级语言。 

机器语言指0、1组成的机器指令序列 

汇编语言指用符号表示指令的语言 MOV AX,2 MOV BX,3 ADD AX,BX

###### (2)高级语言

高级语言是从人类的逻辑思维角度出发、面向各类应用的程序语言，抽象程度大大提高，需要编译成特定机器上的目标代码才能执行。这类语言与人们使用的自然语言比较接近，大大提高了程序设计的效率。

##### 2．编译程序和解释程序

高级语言或汇编语言编写的程序称为源程序，源程序不能直接在计算机上执行。 

如果源程序是汇编语言编写的，则需要一个称为汇编程序 的翻译程序将其翻译成目标程序，然后才能执行。 

如果源程序是为高级语言时，这个翻译程序称为编译程序 

按源程序中语句的执行顺序，逐条翻译并立即执行相关功 能的处理程序、称为解释程序。

###### 程序的执行方式：

（1）编译执行：

按编译方式在计算机上执行用高级语言编写的程序，需经过两个阶段：

编译阶段，把源程序翻译为目标程序；

运行阶段，真正执行此目标程序。 

优点：执行效率高、占用资源小。 

缺点：兼容性差。

（2）解释执行：

源程序的每个语句一经解释就立即执行。 

优点：可移植性较好、开发速度较快、与用户通信方便。 

缺点：效率低。

### 语言处理程序基础

#### 一、表达式

##### 1.前缀表达式

也被称为波兰表示法，其特点是将操作符置于操作数之前，如： - × + 3 4 5 6 。 

从右至左扫描表达式，遇到数字时，将数字压入堆栈，遇到运算符时，弹出栈顶的两个数，用运算符对它们做相应的计算（栈顶元素 op 次顶元素），并将结果入栈；重复上述过程直到表达式最左端，最后运算得出的值即为表达式的结果。

##### 2.中缀表达式

即我们常用的表示方法，(3 + 4) × 5 - 6 。 

中缀表达式是一种通用的算术或逻辑公式表示方法，操作符 以中缀形式处于操作数的中间。中缀表达式是人们常用的算 术表示方法。 虽然人的大脑很容易理解与分析中缀表达式，但对计算机来 说中缀表达式却是很复杂的，因此计算表达式的值时，通常 需要先将中缀表达式转换为前缀或后缀表达式，然后再进行 求值。

##### 3.后缀表达式

又被称为逆波兰法，其特点是将操作符置于操作数之后，3 4 + 5 × 6 - 。

与前缀表达式类似，只是顺序是从左至右： 从左至右扫描表达式，遇到数字时，将数字压入堆栈，遇到运算符时，弹出栈顶的两个数，用运算符对它们做相应的计算（次顶元素 op 栈顶元素），并将结果入栈；重复上述过程直到表达式最右端，最后运算得出的值即为表达式的结果。

#### 二、操作符的优先级 

指针最优，单目运算优于双目运算。如正负号。 

先乘除（模），后加减。 

先算术运算，后移位运算，最后位运算。 1 << 3 + 2 & 7等价于 (1 << (3 + 2))&7 

逻辑运算最后计算。



#### 三、语句间的结构 

顺序语句 

选择语句 

循环语句

#### 四、过程控制

int Function1 (int x,int y) { ....... }

##### 参数传递的方式 

###### 传值调用

数据传送是单向的



###### 引用调用(地址调用)

数据传送是双向的

## 第5章 计算机网络基础

### 网络概述

#### 一、计算机网络定义

计算机网络是指将地理位置不同的具有独立功能的多台计算机及其外部设备，通过通信线路连接起来，在网络操作系统、网络管理软件及网络通信协议的管理和协调下，实现资源共享和信息传递的计算机系统。

#### 二、计算机网络的分类

##### 1.按传输距离分 

局域网(LAN) 一般分布在几米到几公里范围。 

城域网(MAN) 一般分布在几公里到几十公里或一个城区。 

广域网(WAN) 一般分布在数十公里到几千公里

##### 2.按工作模式分为 

对等网络 

基于服务器的网络

##### 3.按传输介质分为 

###### （1）有线网络

分为双绞线网络、同轴电缆网络、光纤网络、光纤同轴混合网络等。 

双绞线

把两根互相绝缘的铜导线并排放在一起，然后用规则的方法绞合起来就构成了双绞线。

绞合可减少对相邻导线的电磁干扰。

**布线标准**

EIA/TIA的布线标准中规定了两种双绞线的线序，分别是T568A与T568B

T568A线序为：白绿 绿 白橙 蓝 白蓝 橙 白棕 棕

T568B线序为：白橙 橙 白绿 蓝 白蓝 绿 白棕 棕

直通线与交叉线：

直通线：是指线缆两端的线序排列完全相同的网线(两端全部使用T568A，或者两端全部使用T568B)。 

交叉线：是指线缆两端的线序一端按照T568A标准连接，另一端按照T568B标准连接。



同轴电缆

由内导体铜质芯线、绝缘层、网状编织的外导体屏蔽层以及保护塑料外层所组成，广泛用于传输较高速率的数据。

光纤

是光导纤维的简写，是一种由玻璃或塑料制成的纤维，可作为光传导工具。传输原理是光的全反射。

光纤按传输方式可分为 

多模光纤

多模光纤是在给定的工作波长上能以多个模式同时传输的光纤。

多模光纤的光源为发光二极管，发出的可见光定向性较差，光以不同的角度进入纤芯。存在许多条不同角度入射的光线在一条光纤中传输，多模光纤适合于近距离传输

单模光纤

单模光纤的直径减小到只有一个光的波长大小，它可使光线沿直线传播，而不会产生多次反射。传输频带宽，传输容量大。单模光纤的光源使用定向型很好的激光二极管。单模光纤的损耗较小，传输距离远。

###### 无线网络

分为无线电、微波、红外等。

无线电

无线电的频带是有限的。上限为在300GHz（吉赫弦），而下限频率在各类规范中不统一，常见的说法有三种分别为3kHz～ 300GHz（国际电信联盟规定）、9kHz～300GHz、10kHz～ 300GHz。 

中波主要沿地面传播，绕射能力强，适用于广播和海上通信。 

短波具有较强的电离层反射能力，适用于环球通信 

超短波和微波的绕射能力较差，可作为视距或超视距中继通信。

微波

微波是指频率在0.3GHz～300GHz范围的电磁波，目前主要是使用2GHz～40GHz的频率范围。

陆地微波系统的主要用途是完成远距离远程通信服务和楼宇间建立短距离的点对点通信。

红外线

红外线的主要特点：(1)不能穿透坚实的物体，防窃听的安全性要比无线电系统好。 (2)几乎不会受到电气、天电、人为干扰，抗干扰性强。 (3)红外线通信机体积小，重量轻，结构简单，价格低廉。 (4)必须在直视距离内通信，且传播受天气的影响。

激光

激光通信系统组成设备包括发送和接收两个部分。

激光通信的优点：通信容量大。、保密性强。 结构轻便，设备经济。

激光通信的缺点：距离限于视距（数公里至数十公里范围），易受气候影响。瞄准困难。激光束有极高的方向性，这给发射和接收点之间 的瞄准带来不少困难。

#### 三、计算机网络的组成

计算机网络由资源子网和通信子网组成。 1.服务器 2.工作站 3.通信设备 4.传输介质

### 网络设备

#### 网卡（NIC）

又称网络适配器。用于计算机和网络电缆之间的物理连接。 网卡完成物理层和数据链路层的大部分功能。每块网卡都有一个唯一的地址，称为MAC地址或物理地址，采用十六进制数表示，共六个字节（48位）。前三个字节是厂家编码，后三个字节由各厂家自行指派。

#### 二、中继器(Repeater)

是工作在物理层设备。适用于完全相同的两类网络的互连，主要功能是通过对数据信号的复制、整形、放大再发送，来扩大网络传输的距离。

#### 三、集线器

具有中继器的功能，区别在于集线器能够提供多端口服务，也称多口中继器。集线器是物理层设备。集线器不具备交换机所具有的MAC地址表，所以它发送数据时是没有针对性的，而是采用广播方式发送。集线器所有端口是一个冲突域，所有的端口是一个广播域。

#### 四、交换机

也称多端口网桥，工作在数据链路层，能够识别帧的内容。 交换机在同一时刻可进行多个端口对之间的数据传输。每一端口都可视为独立的网段，连接在其上的网络设备独自享有全部的带宽，无须同其他设备竞争使用。交换机每一个端口是一个冲突域，所有的端口是一个广播域。

##### 1.二层交换

二层交换是以硬件的方式执行网桥的功能。通过MAC地址进行转发，同时将端口、所涉及的MAC地址及对应关系记录在地址表中。

##### 2.三层交换

三层交换技术就是：二层交换技术+三层转发技术，它解决了局域网中网段划分之后，网段中子网必须依赖路由器进行管理 的局面。三层交换机在对第一个数据流进行路由后，会产生一个MAC地址与IP地址的映射表，当同样的数据流再次通过时，将根据此表直接从二层交换而不是再次路由，从而消除了路由器进行路由选择而造成网络的延迟，提高了数据包转发的效率。

#### 五、路由器

路由器是网络层的互联设备，用于连接多个逻辑上分开的网络，所谓逻辑网络就是拥有独立网络地址的网络。 路由器的工作就是为经过路由器的每个数据帧寻找一条最佳传输路径，并将该数据帧有效地传送到目的站点。路由器每个端口是一个冲突域，每个端口是一个广播域。

### 网络协议与网络拓扑

#### 一、OSI网络参考模型

1.物理层

2.数据链路层

3.网络层

4.传输层

5.会话层

6.表示层

7.应用层

##### 1.物理层

是OSI参考模型的最低层或第一层。 物理层协议要解决的是主机、工作站等数据终端设备与通信 线路上通信设备之间的接口问题。 用4个技术特性来描述：

###### 机械特性

规定了DTE（数据终端设备）和DCE（数据通信设备）之间的连接器形式，包括连接器形状、几何尺寸、引线数目和排列方式等。

###### 电气特性 

规定了发送器和接收器的电气参数及其他有关电路的特征。如1和 0的各自电压的大小，每比特持续多少微秒等。电气特性决定了传送速率和传输距离。

###### 功能特性 

接口信号分为数据信号、控制信号和时钟信号。功能特性对接口各信号线的功能给出确切的定义，说明某些连线上出现的某一电压表示的意义。

###### 规程特性

规定了DTE和DCE之间各接口信号线实现数据传输的操作过程（操作顺序）。

##### 2.数据链路层(DataLink)

建立、维持和释放网络实体之间的数据链路，这种数据链路对 \网络层表现为一条无差错的信道。

数据链路层通常把流量控制和差错控制合并在一起。

数据链路层分为MAC（媒介访问控制层）和LLC（逻辑链路控制层）。

服务访问点为MAC地址

##### 3.网络层(Network)

属于通信子网，网络层解决的问题是路由选择、网络拥塞、异构网络互联等问题，其服务访问点为逻辑地址（网络地址）。 代表性协议有IP，IPX协议等。

##### 传输层( Transport)

实现发送端和接收端的端到端的数据分组传送，负责保证实现数据包无差错、按顺序、无丢失和无冗余的传输。服务访问点为端口。

代表性协议有TCP，UDP，SPX协议等。

##### 5.会话层( Session)

会话层主要功能是管理和协调不同主机上各种进程之间的通信（对话），即负责建立、管理和终止应用程序之间的会话。

##### 6.表示层(Prsentation)

表示层处理流经结点的数据编码的表示方式问题，以保证一个系统应用层发出的信息可被另一系统的应用层读出。 应用层可以根据其服务解释数据的含义。通常包括数据编码的约定、本地句法的转换、数据压缩与解压缩。例如，JPEG， ASCII，GIF，DES，MPEG等。

##### 7、应用层(Application)

直接为端用户服务，提供各类应用程序的接口和用户接口。例如，HTTP，Telnet，FTP，SMTP等。

#### 二、按拓扑结构分类

按拓扑结构可将计算机网络分为 

##### 总线型拓扑结构 

用单总线把各计算机连接起来。

优点：建网容易，增减节点方便，节省线路。

缺点：重负载时通信效率不高。

##### 星型拓扑结构 

每个终端或计算机都以单独（专用）的线路与中央设备相连 。中央设备一般是交换机（集线器）。

优点：结构简单，建网容易，延迟小，便于管理。

缺点：成本高，中心节点成为系统的瓶颈。

##### 环型拓扑结构 

所有计算机和接口设备连接成一个环，可以是单环，也可以是双环。环中信号是单向传输的。双环网络中两个环上信号的传输方向相反。 特别适合实时控制的局域网系统。

##### 树型拓扑结构 

节点组织成树状结构，具有层次性。 它具有较强的可折叠性，非常适用于构建网络主干，还能够有效地保护布线投资。这种拓扑结构的网络一般采用光纤作为网络主干，用于军事单位，政府单位等上、下界限相当严格和层次分明的部门。

##### 网状拓扑结构

每个节点至少有两条路径与其他节点相连。 

优点：可靠性高、可改善线路的信息流量分配、可选择最 佳路径，传输延迟小。 

缺点：控制复杂，线路成本高。

网状拓扑结构一般用于Internet骨干网上。

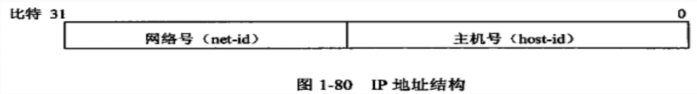
### IPv4

#### 一、网络地址及子网掩码

##### 1. IP地址结构及类别

###### （1）IP地址结构

IP地址是由32位二进制数，即4个字节组成的，由网络号和主机号两个字段组成。

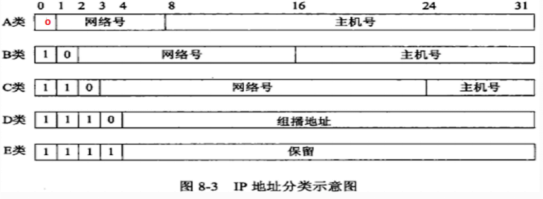


网络号的位数决定了可以分配的网络数（2 n）

主机号的位数决定了网络中最大主机个数（2m-2）

###### （2）IP地址分类

为了便于对IP地址进行管理，把IP地址分成为5类，即A类到E 类，目前大量使用的是A，B，C三类。



A类：0-127

B类：128-191

C类：192-223

D类：224-239

E类：240-255

##### 2. 特殊IP地址

IP定义了一套特殊地址格式，称为保留地址。

###### (1)网络地址。

主机号全0表示网络地址

###### (2)广播地址。

主机号全1表示广播地址

###### (3)子网掩码。

网络号部分全为1，主机号部分全为0；用于计算网络地址用（只需将IP地址和子网掩码做与操作，就可得到网络地址）

###### (4)保留地址。

为了满足内网的使用需求，保留了一部分不在公网使用的IP地址。

|  |  |  |  |
| --- | --- | --- | --- |
| **类别** | **IP地址范围** | **网络号** | **网络数** |
| A | 10.0.0.0-10.255.255.255 | 10 | 1 |
| B | 172.16.0.0-172.31.255.255 | 172.16-172.31 | 16 |
| C | 192.168.0.0-192.168.255.255 | 192.168.0-192.168.255 | 255 |

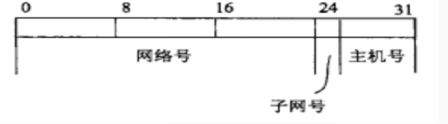
#### 二、子网及子网掩码

两级IP地址的缺点： 

IP地址空间的利用率低。 

给每一个物理网络分配一个网络号会使路由表变得太大因而使网络性能变坏。

在IP地址中增加一个subnet-id字段，将原来的主机号分为子网号和主机号两部分，使两级的IP地址变成为三级的IP地址。这叫作划分子网。划分子网是一个单位内部的事情。单位对外仍然表现为没有划分子网的网络

。

#### 三、路由汇聚

路由汇聚的是把一组路由汇聚为一个单个的路由广播。路由汇聚的最终结果和最明显的好处是缩小网络上的路由表的尺寸。

假设下面有4个网络:

172.18.129.0/24

172.18.130.0/24

172.18.132.0/24

172.18.133.0/24

进行路由汇聚，覆盖这四个网络的汇总地址：172.18.128.0/21

### IPv6

#### 一、 IPv6协议

IPv6协议，全称“互联网协议第6版”，即下一代的网际协议。

相对于IPv4来说，IPv6协议主要改进: 

扩展的地址。IPv6地址长度为128位。 

IPv6使用更小的路由表。 

简化的包头：IPv6头减少了字段的数量，提高选路效率。

流标志：IPv4对所有的包同等对待，路由器并不跟踪任意两台主机间发送的包。而IPv6中引入了流概念，可以对流 中的包进行高效处理。 

身份验证和保密：IPv6使用了两种安全性扩展，即IP身份验证头和IP封装安全性净荷。

##### 1.IPv6地址表示

IPv6地址的128位是以16位为一组，共分为8组，每组的16位转换为4位的十六进制数字，每组之间用冒号(：)分开。

2001: 0da8: d001: 0001: 0000: 0000: 0000: 0001

压缩表示： 2001:da8:d001:1:0:0:0:1

2001:da8:d001:1::1

##### 2.内嵌IPv4地址的IPv6地址

地址的第一部分使用十六进制表示，而IPv4部分采用十进制。这是过渡机制所用的IPv6地址特有的表示法。

如fe80:：200: 5efe: 58.20.27.60

IPv6提供两类嵌有IPv4地址的特殊地址：

0000:0000:0000:0000:0000:FFFF:xxxx:xxxx 或 0000:0000:0000:0000:0000:0000:xxxx:xxxx

##### 3.IPv6地址类型

IPv6仍有三种地址类型，分别是单播、多播（也称组播）、泛播（也称任意播）。在IPv6里广播不再使用。

###### (1)单播IPv6地址：

单播地址唯一标识一个IPv6节点的接口。

###### (2)多播IPv6地址：

多播地址标识一组IPv6节点的接口。

###### (3)泛播IPv6地址：

泛播地址指派给多个节点的接口。发送往泛播地址的数据包只会传递给其中的一个接口，一般是隔得最近的一个接口。

##### 4.IPv4向IPv6过渡

目前解决过渡问题基本技术主要有三种： 

双协议栈 

隧道技术 

NAT-PT

### 常用的网络协议

TCP/IP协议簇分为应用层、传输层、网际层和网络接口层四层



#### 一、ARP（地址解析协议）

IP地址是一个逻辑地址，不能被物理网络所识别。ARP给出了将主机的网络地址(IP地址)动态映射为MAC地址的方法。

#### 二、RARP（反向地址解析协议）

允许局域网的物理机器从网关服务器的ARP表或者缓存上请求其 IP 地址。常用于网络上的无盘工作站。

#### 三、DNS域名系统

方便用户访问互联网，而不用去记住能够被机器直接读取的IP数串。通过主机名，最终得到该主机名对应的IP地址的过程叫做域名解析（或主机名解析）。

DNS协议运行在UDP协议之上，端口号53。

每个域名对应的IP地址可以是一个，也可以是多个。

#### 四、DHCP（动态主机配置协议）

是一个局域网的网络协议，使用UDP协议工作， 给内部网络计算机自动分配IP地址、Gateway地址、DNS服务器地址等信息，并能够提升地址的使用率。

DHCP协议采用UDP作为传输协议，主机发送请求消息到DHCP服务器的67号端口，DHCP服务器回应应答消息给主机的68号端口。

DHCP的默认租约期是8天。

## 第6章 多媒体技术

### 多媒体基本概念

#### 一、媒体的分类

媒体可分为感觉媒体、表示媒体、表现媒体、存储媒体和传输媒体。

##### 1.感觉媒体

直接作用于人的感官，产生感觉（视、听、嗅、味、触觉）的媒体，语言、音乐、音响、图形、动画、数据、文字等 都是感觉媒体。

##### 2.表示媒体

是指用来表示感觉媒体的数据编码。如图像编码 (JPEG、MPEG)、文本编码(ASCII)和声音编码等。感觉媒体转换成表示媒体后，能够在计算机上进行加工处理和传输

##### 3.表现媒体

是进行信息输入或输出的媒体。如键盘、鼠标、扫描仪、话筒、数码相机、摄像机为输入表现媒体，显示器、打印机、扬声器、投影仪为输出表现媒体。

##### 4.存储媒体

是指用于存储表示媒体的物理实体。如硬盘、光盘等。

##### 5.传输媒体

是指传输表示媒体（即数据编码）的物理实体。如电缆、光缆等。

### 颜色空间

#### 一、颜色属性

视觉上的彩色可用下面参数来描述 

##### 色调：

人眼看到一种或多种波长的光时所产生的彩色感觉。 

##### 明度：

表示色所具有的亮度和暗度被称为明度

##### 饱和度：

指颜色的纯度，或者说是指颜色的深浅程度。

#### 二、颜色空间

三原色原理是指自然界的各种颜色光，都可由红(R)、绿(G)、蓝(B) 三种颜色按不同比例相配制而成。所以称这3种颜色为三基色。

(1) RGB颜色空间

(2) YUV颜色空间

数字化位通常采用Y:U:V-8:4:4或者Y:U:V-8:2:2

(3) CMY颜色空间

#### 三、图形与图像

##### 1．图像

图像也称为位图或点阵图，是由排成行列的像素组成的。

常把一幅位图图像作为一个点矩阵处理，矩阵中的一个元素（像素）对应图像的一个点。

##### 2．图形

图形也称矢量图形，是用一个指令集合来描述的，这些指令用来描述图中线条的形状、位置、颜色等各种属性和参数。特点是放大后图像不会失真，和分辨率无关。

##### 3．分辨率

图形（图像）的主要指标有分辨率、点距、色彩数（灰度）。

###### (1)分辨率：

可以分为屏幕分辨率和输出分辨率，单位：dpi（像 素每英寸）。 

水平分辨率是指水平方向（横向）总的像素点数。 

垂直分辨率是指垂直方向（坚向）总的像素点数。

分辨率越高，所包含像素越多，图像越清晰，印刷质量就越好。

###### (2)点距：

指两个像素之间的距离，一般来说，分辨率越高，则像素点距的规格越小，显示效果越好。

###### (3)深度：

是指显示或存储每个像素所用的位数。

### 数据压缩技术

数据压缩技术分为两大类：

无损压缩编码法，也称为冗余压缩法、熵编码法 

有损压缩编码法，也称为熵压缩法

注：把信息中排除了冗余后的平均信息量称为“信息熵”

#### 一、无损压缩法

去掉或减少了数据的冗余，这些冗余值可以重新插入到数据中，因此是可逆的，也是无失真压缩。

它通常使用的是统计编码技术，包括哈夫曼编码、算术编码、行程编码等。它的压缩率比较低，通常是2:1 --5:1

#### 二、有损压缩法

压缩了熵，会减少信息量，因此是不可逆的。它通常可以分为特征抽取和量化两大类。 

特征抽取包括基于模式的编码、分形编码等； 

量化包括零记忆量化、预测编码、直接映射、变换编码等方法。

其中，预测编码和变换编码是最常见的方法。

有损压缩能够达到较高的压缩比。对于声音可达4:1～8:1，对于动态的视频数据可达100:1～400:1。

#### 三、数据压缩标准

(1) JPEG (联合图像专家组) (2) MPEG(动态图像专家组) (3) DVI(数字视频接口) (4) H.261 (5) H.263

### 音频数据

#### 一、音频技术概述

##### 声音的三个要素： 

###### 音强：

即音量，是声音的强度，取决于声间的振幅。 

###### 音调：

由声音的频率决定。 

###### 音色：

又叫音品，指声音的感觉特性。不同的发声体由于 料、结构不同，发出声音的音色就不同。物体振动，除了一个基音外，还有许多不同频率（振动的速度）的泛音伴随，正是这些泛音决定了其不同的音色。

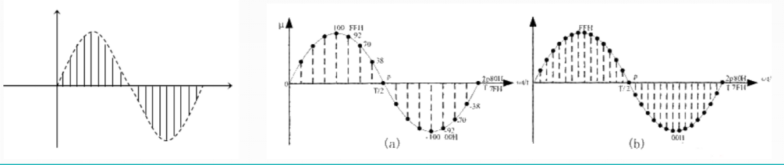
#### 二、音频数据存储和传输

模拟信号转化数字信号过程：采样、量化、编码

##### 1.采样

每隔一个时间间隔就在模拟声音的波形上取一个幅度值，这个间隔时间称为采样频率。

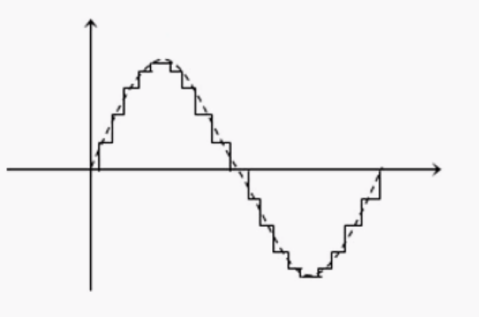
常用的采样频率为8kHz、11.025kHz、16kHz、22.05kHz(FM广播音质)、44.1kHz (CD音质)、48kHz (DVD音频或专业领域)，频率越高音质越好。采样频率不应低于声音信号最高频率的两倍。



##### 2.量化

就是把经过采样得到的瞬时值将其幅度离散，即用一组规定的电平，把瞬时抽样值用最接近的电平值来表示。

量化的级别通常用位(bit)来表示，位数越高则音质越好。



##### 3.编码

将声音数据写成计算机的数据格式。

每秒钟所需的存储量可由下式估算出：文件的字节数=采样频率(Hz)×采样位数×声道数÷8

#### 三、音频数据格式

(1) WAVE

(2) MOD

(3) Layer-3

(4) Real Audio

(5) CD Audio

(6) MIDI

(7) CMF

## 第7章 数据库技术基础

### 基本概念

#### 一、数据

是数据库中存储的基本对象，是描述事物的符号记录。

数据的种类：文本、图形、图像、音频、视频、学生的档案记录、货物的运输情况等。

#### 数据库(Database, DB)

数据库是长期储存在计算机内、有组织的、可共享的大量数据的集合。

##### 数据库的基本特征： 

数据按一定的数据模型组织、描述和储存 

可为各种用户共享 

冗余度较小 

数据独立性较高 

易扩展

#### 三、数据库管理系统(DBMS)

是位于用户与操作系统之间的一层数据管理软件，是一个大型复杂的软件系统 。

##### DBMS的用途 

科学地组织和存储数据 

高效地获取和维护数据

#### 四、数据库系统（Database System，DBS）

是计算机系统中引入数据库后的系统构成。

##### 数据库系统的构成 

数据库 

硬件平台 

软件（应用程序） 

数据库管理员

#### AHE`U8IQXR1ZV))V4W]TL{Y五、数据库系统的三级模式

数据库系统可以分为： 

外模式(子模式、用户模式) 

模式(概念模式、逻辑模式) 

内模式(存储模式)

##### 1.模式（概念模式、逻辑模式） 

数据库中全体数据的逻辑结构和特征的描述 

所有用户的公共数据视图，综合了所有用户的需求

一个数据库只有一个模式

##### 2.外模式（子模式、用户模式） 

数据库用户（包括应用程序员和最终用户）使用的局部数据的逻辑结构和特征的描述 

数据库用户的数据视图，是与某一应用有关的数据的逻辑表示

###### 外模式的地位：

介于模式与应用之间 

###### 模式与外模式的关系：

一对多 

外模式通常是模式的子集 

一个数据库可以有多个外模式。反映了不同的用户的应用需 求、看待数据的方式、对数据保密的要求

###### 外模式的用途 

保证数据库安全性的一个有力措施 

每个用户只能看见和访问所对应的外模式中的数据

##### 3.内模式（存储模式） 

是数据物理结构和存储方式的描述 

是数据在数据库内部的表示方式 

一个数据库只有一个内模式

##### 4.三个级别

与三级模式相对应，数据库系统可以划分为三个抽象级：

###### (1)用户级数据库：

对应于外模式，是用户看到和使用的数据库，又称用户视图。一个数据库可有多个不同的用户视图。

###### (2)概念级数据库：

对应于模式，是所有用户视图的最小并集，一个数据库应用系统只有一个DBA视图。

###### (3)物理级数据库：

对应于内模式，是数据库的低层表示，它描述数据的实际存储组织，是最接近于物理存储的，又称为内部视图。

### 数据模型

#### 一、数据模型的基本概念 

##### 1.概念数据模型 

E-R模型 

##### 2.基本数据模型 

层次模型 

网状模型 

关系模型 

面向对象数据模型

#### 二、数据模型的三要素 

数据结构 

数据操作 

数据的约束条件

#### 三、E-R模型

1976年, P.P.S.Chen提出E-R模型，用E-R图来描述概念数据模型。

观点: 世界是由一组称作实体的基本对象和这些对象之间的联系构成的。

##### 1.实体（Entity）

客观存在并可相互区别的事物称为实体。

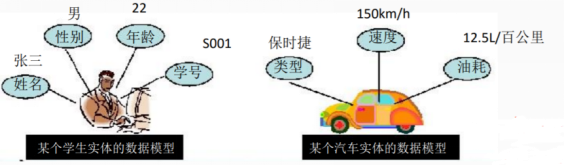
可以是具体的人、事、物或抽象的概念。 

客观实体：人,汽车,图书,…… 

抽象实体：帐户,贷款, ……

##### 2.属性（Attribute）

实体所具有的某一特性称为属性。一个实体可以由若干个属性来刻画。 例如：学生实体由学号、姓名、性别、出生年份、系、入学时间等属性组成。



###### 属性分类 

简单属性和复合属性 

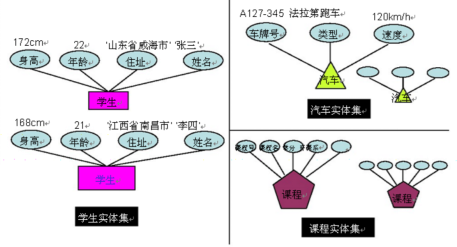
单值属性和多值属性 

NULL属性 

派生属性

##### 3.实体集(Entity Set)

具有相同类型和共享相同属性的实体的集合，如学生,课程。



##### 4.域（Domain）

属性的取值范围称为该属性的域。

##### 5.实体型（Entity Type）

用实体名及其属性名集合来抽象和刻画，同类实体称为实体型

例如：学生（学号、姓名、性别、出生年份、系、入学时间）

##### 6.码（Key）

唯一标识实体的属性集称为码。

例如：学号是学生实体的码。

##### 7. 联系（Relationship）

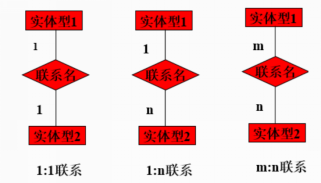
现实世界中事物内部以及事物之间的联系，在信息世界中反映为实体内部的联系和实体之间的联系

两个实体型间联系分为三类 

一对一联系（1:1） 

一对多联系（1:n） 

多对多联系（m:n）



###### （1）一对一联系

如果对于实体集A中的每一个实体，实体集B中至多有一个实体与之联系，反之亦然，则称实体集A与实体集B具有一对一联系。记为1:1。

例：班级与班长之间的联系

系与系主任之间的联系

部门与部门经理间的联系

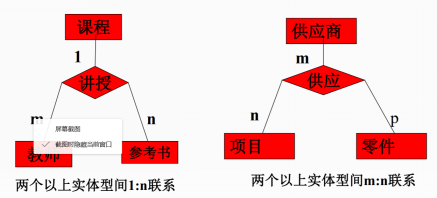
###### （2）一对多联系

如果对于实体集A中的每一个实体，实体集B中有n个实体（n≥0） 与之联系，反之，对于实体集B中的每一个实体，实体集A中至多只有一个实体与之联系，则称实体集A与实体集B有一对多联系记为1:n

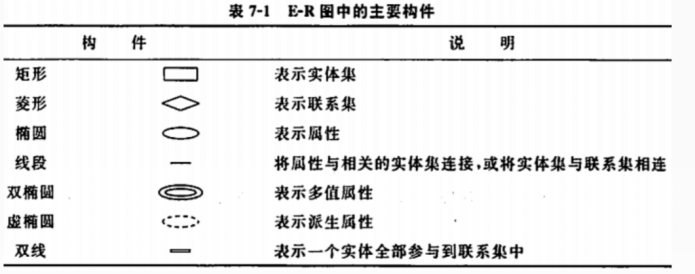
###### （3）多对多联系（m:n）

如果对于实体集A中的每一个实体，实体集B中有n个实体（n≥0）与之联系，反之，对于实体集B中的每一个实体，实体集A中也有 m个实体（m≥0）与之联系，则称实体集A与实体B具有多对多联系。记为m:n

###### （4）两个以上实体型之间一对多联系



##### 8.E-R图的组成要素



##### 9.E-R图建立过程

步骤1：确定实体类型。

步骤2：确定联系类型。

步骤3：把实体类型和联系类型组合成ER图

步骤4：确定实体类型和联系类型的属性。

步骤5：确定实体类型的关键码，在ER图中属于码的属性名下画一条横线。

#### 四、关系模型

在用户观点下，关系模型中数据的逻辑结构是一张二维表，由行和列组成。

用表格结构表达实体集，用外键（外码）表示实体间联系。



##### 优点： 

建立在严格的数学概念基础上 

概念单一，结构简单、清晰，用户易懂易用 

存取路径对用户透明，从而数据独立性、安全性好，简化数据库开发工作。 

##### 缺点：

由于存取路径透明，查询效率往往不如非关系数据模型。

### 数据库控制功能

#### 一、事务

DBMS运行的基本工作单位是事务。

事务是用户定义的一个数据库操作序列，这些操作序列要么全做要么全都不做，是一个不可分割的工作单位。

##### 事务的特性(ACID特性)：

(1)原子性（atomicity）

(2)一致性（consistency）

(3)隔离性（isolation）

(4)持续性（durability，永久性）

事务通常以BEGIN TRANSACTION（事务开始）语句开始，以 COMMIT或ROLLBACK语句结束。

COMMIT称为事务提交语句，表示事务执行成功地结束。

ROLLBACK称为事务回滚语句，表示事务执行不成功地结束，即把事务对数据库的修改进行恢复。

#### 二、故障恢复

##### 1.故障种类 

事务内部故障 

系统故障 

介质故障 

计算机病毒

##### 2.数据库恢复

数据库恢复原理：冗余

建立冗余数据常用技术： (1) 数据转储 (2) 登记日志文件

###### (1)数据转储的概念

转储即DBA定期地将整个数据库复制到磁带或另一个磁盘上保存起来的过程。这些备用的数据文本称为后备副本或后援副本。

###### (2)静态转储和动态转储 

静态转储：是在系统中无运行事务时进行的转储操作。 

动态转储：是指转储期间允许对数据库进行存取或修改。后备副本需配合日志文件。

###### (3)海量转储和增量转储 

海量转储：每次转储全部数据库。 

增量转储：每次只转储上一次转储后更新过的数据。

###### (4)日志文件的作用 

事务故障恢复和系统故障必须用日志文件。 

在动态转储方式中必须建立日志文件，后援副本和日志文件综合起来才能有效地恢复数据库。 

在静态转储方式中，也可以建立日志文件

###### (5)登记日志文件

应遵循两条原则：

1)登记的次序严格按并发事务执行的时间次序。

2)必须先写日志文件，后写数据库

为什么要先写日志文件:

写数据库和写日志文件是两个不同的操作 

在这两个操作之间可能发生故障 

如果先写了数据库修改，而在日志文件中没有登记下这个修改， 则以后就无法恢复这个修改了 

如果先写日志，但没有修改数据库，按日志文件恢复时只不过是多执行一次不必要的UNDO操作，并不会影响数据库的正确性

##### 3.故障恢复策略 

###### (1)事务故障的恢复 

事务故障: 是指事务运行过程中由于种种原因使事务未运行至正常终止点而夭折的情况。如：运算溢出、因死锁而被选中撤销该事务

事务故障的恢复:

反向扫描文件日志，查找该事务的更新操作。

对该事务的更新操作执行逆操作。

重复1)、2)直到遇此事务的开始标记。

###### (2)系统故障的恢复 

系统故障:是指造成系统停止运转的任何事件，使得系统要重新启动。例如，特定类型的硬件错误(CPU故障)、操作系统故障、突然停电等。

恢复操作： 

要撤销故障发生时未完成的事务 

重做(REDO)已完成的事务

系统故障的恢复是在重新启动时自动完成的，不需要用户干预。

###### (3)介质故障的恢复

介质故障:是指外存设备故障。如：磁盘损坏、强磁场干扰

介质故障的恢复:

1)装入最新的数据库后备副本，使数据库恢复到最近一次转储时的一致性状态。

2)装入相应的日志文件副本，重做已完成的事务。

##### 4.数据库镜像

根据DBA的要求，自动把整个数据库或其中的关键数据复制到另一个磁盘上。DBMS自动保证镜像数据与主数据的一致性。

作用： 

数据库的恢复 

用于并发操作

#### 三、并发操作

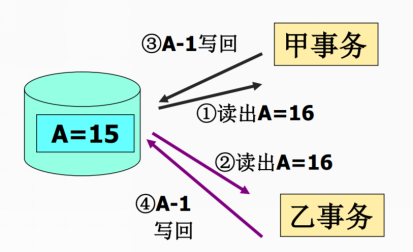
在多用户共享系统中，多个用户同时存取同一数据的情形称为并发操作。 

并发操作的优点: 增加了处理器和磁盘的可用性, 加大事务吞吐量。 

##### 1.并发操作可能带来的问题： 

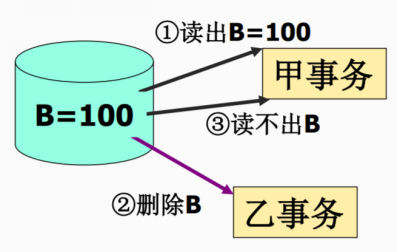
###### (1)更新丢失

T1、T2读入同一数据并修改，T2提交的结果破坏了T1提交的结果



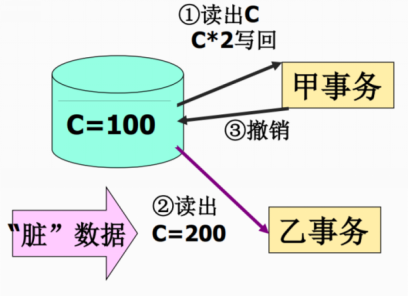
###### (2)不可重读

T1读数据后，T2执行更新操作，使T1无法再现前一次读取结果。



###### (3)读脏数据

T1修改数据并写回磁盘，T2读取同一数据后，T1被撤销即数据恢复原值，T2读的数据与DB中的不一致，称为“脏”数据。



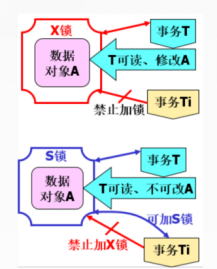
##### 2.封锁

封锁就是事务T在对某个数据对象（例如表、记录等）操作之前，先向系统发出请求，对其加锁，加锁后事务T就对该数据对象有了一定的控制，在事务T释放它的锁之前，其它的事务不能更新此数据对象。封锁是实现并发控制的一个非常重要的技术

###### （1）封锁类型 

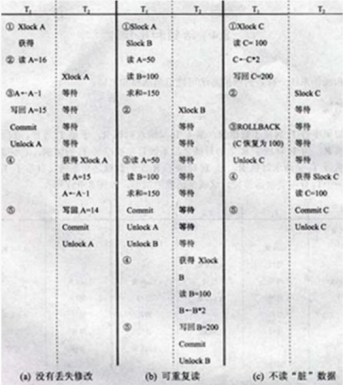
排它锁(X锁或写锁) 保证其他事务在T释放A上的锁之前，不能再修改A。 

共享锁(S锁或读锁) 保证其他事务可以读A，但在T释放A上的S锁之前，不能对A做任何修改。



###### （2）封锁协议

在运用X锁和S锁对数据对象加锁时，还需要约定一些规则，例如 何时申请X锁或S锁、持锁时间、何时释放等。称这些规则为封锁协议（Locking Protocol）。对封锁方式规定不同的规则，就形成了各种不同的封锁协议。

1)一级封锁协议

事务T在修改数据R之前必须先对 其加X封锁，直到事务结束才释 放。一级封锁协议可防止丢失修 改，并保证事务T是可恢复的。 在一级封锁协议中，如果仅仅是 读数据不对其进行修改，是不需 要加锁的，所以它不能保证可重 复读和不读“脏”数据。

2）二级封锁协议

在一级封锁的基础上，事务读数据的时候加s锁，读取之后释放。二级封锁协议可以防止丢失更新，读脏数据。

在二级封锁协议中，由于读完数据后即可释放S锁，所以它不能保证可重复读。

3）三级封锁协议

一级封锁协议加上事务T在读取数据R之前先对其加S锁，直到事务结束才释放。三级封锁协议可防止丢失修改、 防止读“脏”数据，还进一步防止了不可重复读。

4)两段锁协议

所有事务必须分两个阶段对数据项加锁和解锁。其中扩展阶段是在对任何数据进行读、写操作之前，首先要申请并获得对该数据的封锁；收缩阶段是在释放一个封锁之后，事务不能再申请和获得任何其他封锁。

Lock A, Read A, A:=A+100, Write A, Lock B, Unlock A, Read B, Unlock B, Commit

###### 3.封锁的粒度

封锁的粒度是被封锁数据目标的大小，在关系数据库中封锁粒度有属性值、属性值集、元组、关系、某索引项（或整个索引）、整个关系数据库、物理页（块）等几种。

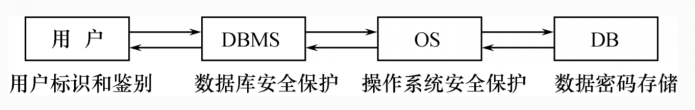
封锁粒度小则并发性高，但开销大；封锁粒度大则并发性低，但开销小。

#### 四、数据库的安全性

数据库系统的信息安全在技术上依赖于两种方式：

DBMS本身提供的用户身份识别、视图、使用权限控制、审计 等管理措施

依靠数据库的应用程序来实现对数据库访问进行控制和管理。



##### 1．用户认证

用户的身份认证是用户使用DBMS系统的第一个环节。

###### (1)口令认证

###### (2)强身份认证 

用户证书 

智能卡 

用户指纹识别

##### 2．用户角色

用户角色是具有相同操作权限的用户集合，不同角色的用户授予不同的数据管理和访问操作权限。

**一般可以将权限角色分为3类： **

数据库登录权限类 

资源管理权限类 

DBA权限类

##### 3．存取控制（数据授权）

数据库授权可以分为静态授权和动态授权。

静态授权是DBMS的隐性授权，也就是说用户（或DBA）对他自己拥有的信息是不需要有指定的授权动作就拥有全权管理和操作的权限的。

动态授权指数据对象的所有者或者DBA默认地拥有对数据的存取权，允许他们把这些权力授予其他的用户。

访问控制可以对用户访问的数据对象进行控制。

粒度从大到小分为4个层次：

(1)数据库级别

(2)表级，判断用户是否可以访问关系里面的内容。

(3)记录级（行级），判断用户是否能访问关系中的一行记录的内容。

(4)属性级

##### 4．数据库视图

是一个虚拟表，其内容由查询定义，其数据不进行实际存储，在对视图的数据进行操作时，系统根据视图的定义去操作与视图相关联的基本表。

使用视图可以实现下列功能：

(1)将用户限定在表中的特定行上。

(2)将用户限定在特定列上。

(3)将多个表中的列连接起来，使它们看起来像一个表。

(4)聚合信息而非提供详细信息。

##### 5．审计功能

身份认证是一种事前的防范措施，审计是一种事后监督的手段。审计作为一种安全检查的措施，会把系统的运行状况和用户访问数据库的行为记录以日志保存下来，该日志作为一种稽查用户行为的一种证据。

数据库系统的审计工作包括： 

设备安全审计 

操作审计 

应用审计 

攻击审计

##### 6. 数据加密

是防止数据库中的敏感信息在存储和传输过程中失密的有效手段。

### 数据仓库与数据挖掘

#### 一、数据仓库 

是一个用以更好地支持企业（或组织）决策分析处理的、面向主题的、集成的、不可更新的（相对稳定的）、随时间不断变化的数据集合。 

本质上和数据库一样，是长期储存在计算机内的、有组织、可共享的数据集合。

#### 二、 数据仓库的基本特征 

数据仓库的数据是面向主题的； 

数据仓库的数据是集成的； 

数据仓库的数据是不可更新的； 

数据仓库的数据是随时间不断变化的

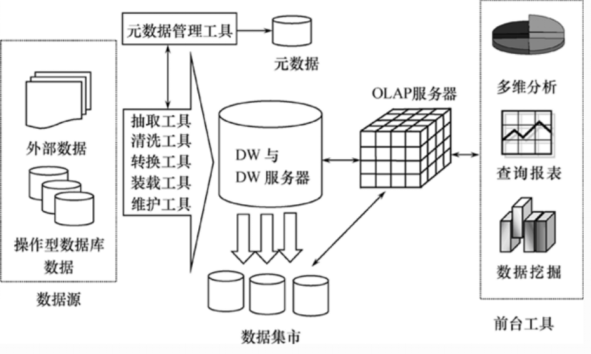
##### 数据仓库反映历史变化的属性主要表现

(1)数据仓库中的数据时间期限要远远长于传统操作型数据系统 中的数据时间期限。

(2)传统操作型数据系统中的数据含有“当前值”的数据，这些数据在访问时是有效的，当然数据的当前值也能被更新，但数据仓库中的数据仅仅是一系列某一时刻生成的复杂的快照。

(3)传统操作型数据系统中可能包含也可能不包含时间元素，如 年、月、日、时、分、秒等，而数据仓库中一定会包含时间元素。

#### 数据仓库系统的体系结构



数据仓库的后台工具：包括数据抽取、清洗、转换、装载和维护工具。 

数据仓库服务器：相当于数据库系统中的DBMS，负责管理数据仓库中数据的存储管理和数据存取，并给OLAP服务器和前台工具提供存取接口(如SQL查询接口) 

OLAP（On-Line Transaction Processing）服务器：透明地为前台工具和用户提供多维数据视图；

OLAP服务器则必须考虑物理上这些分析数据的存储问题 

前台工具：包括查询报表工具、多维分析工具、数据挖掘工具和分析结果可视化工具等

#### 四、数据挖掘的概念 

概念：数据挖掘是从大量数据中发现并提取隐藏在内的、人们事先不知道的但可能有用的信息和知识的一种新技术 

目的：帮助决策者寻找数据间潜在的关联，发现经营者被忽略的要素 

数据挖掘技术涉及数据库技术、人工智能技术、机器学习、统计分析等多种技术

#### 五、数据挖掘和传统分析方法的区别

本质区别：数据挖掘是在没有明确假设的前提下去挖掘信息，发现知识 。

数据挖掘所得到的信息应具有事先未知、有效和可实用3个特征

#### 六、数据挖掘的数据源 

##### 从数据仓库中来

优点：许多数据不一致的问题都较好地解决了，在数据挖掘时大大减少了清理数据的工作量

缺点：建立数据仓库是一项巨大的工程，耗时耗力 

##### 从数据库中来

如果只是为了数据挖掘，可以把一个或几个OLTP数据库导入一个只读的数据库中，然后在上面进行数据挖掘

#### 七、 数据挖掘的功能 

趋势和演变分析 

关联分析 

聚类 

概念描述 

偏差检测

#### 八、数据挖掘的流程

##### (1)问题定义

在开始数据挖掘之前最先的也是最重要的要求就是熟悉背景知识，弄清用户的需求

##### (2)建立数据挖掘库

要进行数据挖掘必须收集要挖掘的数据资源。一般建议把要挖掘的数据都收集到一个数据库中

##### (3)分析数据

分析数据就是通常所进行的对数据深入调查的过程。从数据集中找出规律和趋势，发现因素之间的相关性。

##### (4)调整数据

通过上述步骤的操作，对数据的状态和趋势有了进一步的了解， 这时要尽可能对问题解决的要求能进一步明确化、进一步量化。

##### (5)模型化

在问题进一步明确，数据结构和内容进一步调整的基础上，就可以建立形成知识的模型。

##### (6)评价和解释

## 第8章 关系数据库

### 概述

#### 一、关系模型

在用户观点下，关系模型中数据的逻辑结构是一张二维表，由行和列组成。

用表格结构表达实体集，用外键（外码）表示实体间联系。



##### 优点： 

建立在严格的数学概念基础上 

概念单一，结构简单、清晰，用户易懂易用 

存取路径对用户透明，从而数据独立性、安全性好，简化数据库开发工作。 

##### 缺点：

由于存取路径透明，查询效率往往不如非关系数据模型。

#### 二、基本概念

##### 1.域

是一组具有相同数据类型的值的集合。 

整数 

实数 

{‘男’，‘女’} 

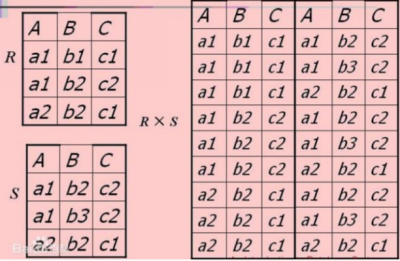
……………..

##### 2.笛卡尔积

给定一组域D1，D2，…，Dn，这些域中可以有相同的。 D1，D2，…，Dn的笛卡尔积为： D1×D2×…×Dn ＝ ｛（d1，d2，…，dn）｜di ∈ Di，i＝1，2，…，n｝ 

所有域的所有取值的一个组合 

不能重复



##### 3.关系

D1×D2×…×Dn的子集叫做在域D1，D2，…，Dn上的关系，表示为 R（D1，D2，…，Dn）

R：关系名

n：关系的目或度（Degree）

###### (1)关系的表示

关系也是一个二维表，表的每行对应一个元组，表的每列对应一个域

###### (2)属性

关系中不同列可以对应相同的域 为了加以区分，必须对每列起一个名字，称为属性（Attribute） n目关系必有n个属性

###### (3)候选码（Candidate key）

若关系中的某一属性组的值能唯一地标识一个元组，则称该属性组为候选码

###### (4)主码（Primary key）

若一个关系有多个候选码，则选定其中一个为主码

###### (5)关系的三种类型

(1)基本关系

(2)查询表

(3)视图表

###### 基本关系具有的性质：

(1)列是同质的，即每一列中的分量是同一类型的数据，来自同一个域。

(2)不同的列可出自同一个域，称其中的每一列为一个属性，不同的属性要给予不同的属性名。

(3)列的顺序无所谓，即列的次序可以任意交换。

###### 关系的描述

关系的描述称为关系模式，关系模式是一个五元组，形式化地表示为：

R（U，D，DOM，F）

R 关系名

U 组成该关系的属性名集合

D 属性组U中属性所来自的域

DOM 属性向域的映象集合

F 属性间的数据依赖关系集合

通常简记为R（A1，A2，…，An）。R为关系名， A1，A2....为属性名。

#### 二、完整性约束

##### (1)实体完整性(Entity integrity)

是指关系的主关键字不能重复也不能取“空值"。如主关键字是多个属性的组合，则所有主属性均不得取空值。

##### (2)参照完整性(Referential Iintigrity)

是定义建立关系之间联系的主关键字与外部关键字引用的约束条件。

##### (3)用户定义完整性(user defined integrity)

则是根据应用环境的要求和实际的需要，对某一具体应用所涉及的数据提出约束性条件。

### 关系运算

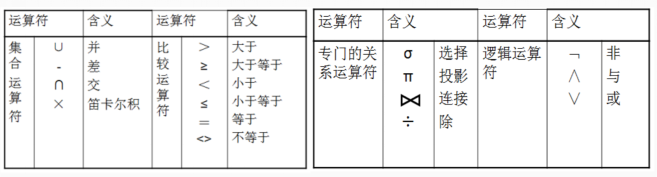
#### 一、常用的关系操作 

查询：选择、投影、连接、除、并、交、差 

数据更新：插入、删除、修改 查询的表达能力是其中最主要的部分 

传统的集合运算是二目运算，包括并、交、差、广义笛卡儿积 

专门的关系运算包括：选择、投影、连接、除4种运算。



##### 1.并

具有相同的目n（两个关系都有n个属性），R和S的并是由属于R 或属于S的元组组成的集合，记为RUS。

形式定义如下： R∪S = { t |t ∈ R∨t ∈S } 式中t是元组变量（下同）。 显然， RUS = SUR。

##### 2.差

关系R和S具有相同的目n，R和S的差是由属于R但不属于S的元组组成的集合，记为R-S。

##### 3.交

关系R和S具有相同的目n，R和S的交是由既属于R又属于S的元组组成的集合，记为R ∩ S。

形式定义如下： R∩S = { t|t ∈ R∧t ∈ S } R∩S = R – (R-S） R∩S = S – (S-R）

##### 4.笛卡儿积

R: n目关系，k1个元组 S: m目关系，k2个元组 R×S

列：（n+m）列元组的集合 

元组的前n列是关系R的一个元组 

后m列是关系S的一个元组

行：k1×k2行元组 

R×S = {tr ts |tr ∈ R ∧ ts ∈ S }

#### 二、关系运算

关系数据库还有一些专门的运算，主要有投影、选择、连接、除法和外连接。

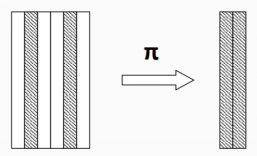
在关系代数中，由五种基本代数操作经过有限次复合的式子称为关系代数运算表达式。表达式的运算结果仍是一个关系。

##### 1.投影

投影操作从关系R中选择出若干属性列组成新的关系，该操作对关系进行垂直分割，消去某些列，并重新安排列的顺序，再删去重复元组。

记作： πA(R) = { t[A] | t ∈ R } 其中A为R的属性列。

投影之后不仅取消了原关系中的某些列，而且还可能取消某些元组（避免重复行）

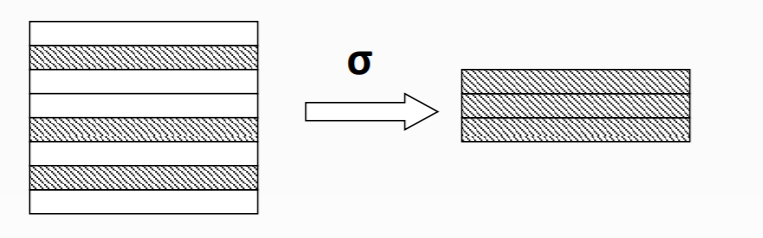


##### 2.选择

选择操作在关系R中选择满足给定条件的所有元组，

记作 σF(R) = {t|t ∈ R∧F(t)= '真'}

其中F表示选择条件，是一个逻辑表达式（逻辑运算符+算术表达式）。选择运算是从行的角度进行的运算。



##### 3.θ连接

θ连接从两个关系的笛卡儿积中选取属性间满足一定条件的元组记作：

_7]~BEF4[Z)HLBVO`CU%1YX

A和B：分别为R和S上度数相等且可比的属性组

θ：比较运算符 

一般连接 

等值连接 

自然连接

###### (1)一般连接

例：关系R和关系S 如下所示，一般连接 0$K{$WDMTQ%$ILYNKRZ[@71的结果如下：

![}@V3~9AY$](0~%@R}JC3VQA](data:image/png;base64,)

###### (2)等值连接（equijoin）

θ为“＝”的连接运算称为等值连接

等值连接的含义

从关系R与S的广义笛卡尔积中选取A、B属性值相等的那些元组， 即等值连接为：

ZDTE0(@ASFY``(B4Y)@B3QL

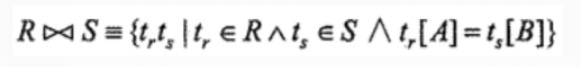
###### (3)自然连接（Natural join）

自然连接是一种特殊的等值连接 

两个关系中进行比较的分量必须是相同的属性组 

在结果中把重复的属性列去掉

自然连接的含义 R和S具有相同的属性组B



###### (4)外连接（OUTER JOIN）

关系R、S 进行自然连接时，如果把舍弃的元组也保存在结果关系中，而在其他属性上填空值(Null)，这种连接就叫做外连接。

###### (5)左外连接

关系R、S 进行自然连接时，如果只把左边关系R中要舍弃的元组保留就叫做左外连接(LEFT OUTER JOIN或LEFT JOIN)

###### (6)右外连接

关系R、S 进行自然连接时，如果只把右边关系S中要舍弃的元组保留就叫做右外连接(RIGHT OUTER JOIN或RIGHT JOIN)

##### 4.除法

给定关系R (X，Y) 和S (Y，Z)，其中X，Y，Z为属性组。

R中的Y与S中的Y可以有不同的属性名，但必须出自相同的域集。

R与S的除运算得到一个新的关系P(X)，

P是R中满足下列条件的元组在 X 属性列上的投影：

元组在X上分量值x的象集Yx包含S在Y上投影的集合，记作： R÷S = {tr [X] | tr ∈ R∧πY (S) ∈ Yx } Yx：x在R中的象集，x = tr [X]

### 查询优化

#### 一、选择操作的实现

例１ Select \* from student where <条件表达式>

<条件表达式> 的几种情况：

C1: 无条件；

C2: Sno=‘200215121’；

C3: Sage>20;

##### 1.简单的全表扫描方法

对查询的基本表顺序扫描，逐一检查每个元组是否满足选择条件，把满足条件的元组作为结果输出。 

对于小表，这种方法简单有效。 

对于大表，顺序扫描十分费时，效率很低

##### 2.索引(或散列)扫描方法

如果选择条件中的属性上有索引(例如Ｂ＋树索引或者Hash索引)，可以用索引扫描方法。

通过索引先找到满足条件的元组主码或元组指针，再通过元组指针直接在查询的基本表中找到元组。

以C2为例，Sno=‘200215121’，并且Sno上有索引。可以使用索引得到 Sno为‘200215121’元组的指针，然后通过元 组指针在 student 表中检索到该学生。

以C3为例，Sage>=28 , 并且Sage上有B+树索引。可以使用B+树索引找到Sage＝28的索引项，以此为入口点在B+树的顺序集上得到Sage>=28的所有元组指针，然后通过这些元组指针到student 表中检索所有年龄大于28的学生。

#### 二、连接操作的实现

连接操作是查询处理中最耗时的操作之一。

例２ SELECT \* FROM Student , SC WHERE Student.Sno ＝ SC.Sno

##### 算法1：嵌套循环方法

对外层循环(Student)的每一个元组(s)，检索内层循环(SC)中的每一个元组(sc)，并检查这两个元组在连接属性(Sno)上是否相等。如果满足连接条件，则串接后作为结果输，直到外层循环表中的元组处理完毕为止。

##### 算法2：排序－合并方法

适合连接的诸表已经排好序的情况。

步骤如下：

(1)如果连接的表没排好序，首先对两个表都按连接属性Sno排序。

(2)取Student表中第一个Sno，依次扫描SC表中具有相同Sno的元组； 把它们连接起来；

(3)当扫描到Sno不相同的第一个SC元组时，返回到Student表扫描它的下一个元组，再扫描SC表中具有相同Sno的元组，把它们连接起来。

#### 三、查询优化目标及步骤

##### 1.查询优化的目标

选择有效策略，求得给定关系表达式的值，使得查询代价最小

##### 2.优化的准则 

提早执行选择运算，目的：减少中间结果。

合并乘积与选择运算为连接运算，目的：避免扫描大的关系。

将投影运算与其他运算同时进行，目的：避免重复扫描关系。 

将投影运算与二目运算结合起来。目的：减少扫描关系的遍数。 

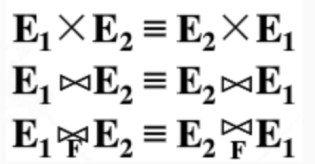
在执行连接前对关系适当地预处理：索引连接法，排序合并连 接法。

存储公共子表达式。目的：只需检索中间结果，不需重复计算

##### 3.关系代数表达式等价变换规则

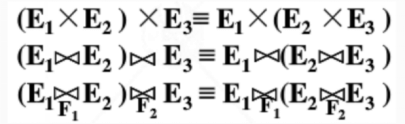
###### (1)连接、笛卡尔积交换律

设E1、E2为关系代数表达式，F为连接运算的条件。



###### (2)连接、笛卡尔积的结合律

设E1、E2 、E3为关系代数表达式，F1、F2为连接运算的条件。



###### (3)投影的串接定律

πA1,A2,…,An(πB1,B2,…,Bm(E)) ≡ πA1,A2,…,An(E) 属性组{A1,A2,…,An}是属性组{B1,B2,…,Bm}的子集。 例：πsno(πsno,sname(Stuent))≡ πsno(Stuent)

###### (4)选择的串接定律

σF1(σF2(E))≡σF1∧F2(E) 如：σJNO=‘J1’(σPNO=‘P1’(SPJ)) ≡ σJNO=‘J1’∧PNO=‘P1’(SPJ)

###### (5)选择与投影的交换律

σF (πA1,A2,…,An(E)) ≡ πA1,A2,…,An (σF (E)) 选择条件F只涉及属性A1,A2,…,An。

若F中有不属于A1,A2,…,An的属性B1,B2,…,Bm,则 πA1,A2,…,An (σF (E)) ≡ πA1,A2,…,An (σF (πA1,A2,…,An,B1,B2,…,Bm(E)))

###### (6)选择与笛卡尔积的交换律

1)若F中涉及的属性都是E1中的属性,则 σF (E1×E2 )≡σF (E1)×E2

2)若F=F1∧F2，且F1只涉及E1中的属性，F2只涉及E2中的属性，则 σF (E1×E2 )≡σF1(E1)×σF2(E2)

###### (7)选择与并的分配律

设E=E1∪E2，E1、E2有相同的属性名 σF(E1∪E2 )≡σF (E1)∪σF(E2)

###### (8)选择与差的分配律

设E1与E2有相同的属性名 σF(E1-E2)≡σF(E1)-σF(E2)

###### (9)选择对自然连接的分配律

σF (E1 ∞ E2 )≡σF (E1) ∞ σF (E2)

###### (10)投影与笛卡尔积的分配律

设A1 ,…,An是关系表达式E1的属性，B1 ,…,Bm是关系表达式E2 的属性

πA1,…,An,B1,…,Bm (E1×E2) ≡ πA1,…,An(E1)×πB1,…,Bm(E2)

###### (11)投影与并的分配律

设E1和E2具有相同的属性名 πA1,…,An(E1 ∪ E2) ≡ πA1,…,An(E1)∪πA1,…,An(E2)

### 关系数据库设计基础理论

#### 一、函数依赖

定义：设U{A1，A2，…，An}是属性集合，R(U)是U上的一个关系，x、y是U的子集。若对于R(U)下的任何一个可能的关系，均有x 的一个值对应于y的唯一具体值，称y函数依赖于x，记作x→y。 学生ID→学生姓名 学生姓名依赖于学号

#### 二、其它函数依赖

##### 1.部分函数依赖

设R(U)是属性集U上的关系，x、y是U的子集，x’是x的真子集，若 x→y且x’→y，则称y部分依赖x，记作XP→Y。

(学生ID，所修课程ID)→学生姓名

学生ID→学生姓名 学生姓名依赖于关键字的一个主属性——学生ID，因此学生姓名部分函数依赖于(学生ID，所修课程ID)。

显然，当且仅当x为复合属性组时，才有可能出现部分函数依赖。

##### 2.完全函数依赖

设R(U)是属性集U上的关系，x、y是U的子集，x’是x的真子集。若对于R(U)的任何一个可能的关系，有x→y且x’!→y，则称y完全函数依赖于x，记作XY。

所谓完全依赖是指在依赖关系的决定项（即依赖关系的左项）中没有多余属性，有多余属性就是部分依赖。

学生ID，学生姓名，所修课程ID，课程名称，成绩

（学生ID，所修课程ID）→成绩

成绩既不能单独依赖于学生ID，也不能单独依赖于所修课程ID，因此成绩完全函数依赖于关键字。

##### 3.传递函数依赖

在关系模式R(U)中，设X，Y，Z是U的不同的属性子集，如果X确定Y、Y确定Z，且有X不包含Y，Y不确定X，（X∪Y）∩Z=空集合，则称Z传递函数依赖于X。

传递函数依赖会导致数据冗余和异常。传递函数依赖的Y和Z子集往往同属于某一个事物，因此可将其合并放到一个表中。

例如：学号确定宿舍、宿舍确定费用，且有学号不包含宿舍，宿舍不确定学号，符合传递函数依赖条件。

#### 三、规范化理论

设有一个关系模式R(SNAME , CNAME , TNAME , TADDRESS)，其属性分别表示学生姓名、课程名、任课教师姓名和任课教师地址。

这个模式存在下列存储异常的问题。

(1)数据冗余

(2)修改异常

(3)插入异常

(4)删除异常

把R分解成下列三个关系模式：R1(SNAME, CNAME)和R2（CNAME, TNAME）,R3(TNAME, TADDRESS)，则能消除上述的存储异常现象。

模式设计强调“每个联系单独表达”是一条重要的设计原则

#### 四、范式

##### 1.第一范式(1NF)

如果关系模式R的每个关系r的属性值都是不可分的原子值，那么称R是第一范式的模式，r是规范化的关系。

说明：在任何一个关系数据库中，第一范式是对关系模式的基本要求，不满足第一范式（1NF）的数据库就不是关系数据库

##### 2.第二范式(2NF)

若关系模式R是1NF，且每个非主属性完全函数依赖于候选键，那么称R是2NF模式。

简单的说，是表中的属性必须完全依赖于全部主键，而不是部分主键。所以只有一个主键的表如果符合第一范式，那一定是第二范式。

例如员工信息表中加上了员工编号（emp\_id）列，因为每个员工的员工编号是惟一的，因此每个员工可以被惟一区分。

##### 3.第三范式(3NF)

如果关系模式R是1NF，且每个非主属性都不传递依赖于R的候选码，则称R是3NF。

例如，存在一个部门信息表，其中每个部门有部门编号（dept\_id）、部门名称、部门简介等信息。那么在员工信息表中列出部门编号后就不能再将部门名称、部门 简介等与部门有关的信息再加入员工信息表中。如果不存在部门信息表，则根据第三范式（3NF）也应该构建它，否则就会有大量的数据冗余。

##### 4.BC范式(BCNF)

若关系模式R是1NF，且每个属性都不传递依赖于R的候选键，那么称R是BCNF模式。

相对于第三范式，BC范式的要求更加严格。第三范式只是要求R为第二范式且非主属性不传递依赖于R的候选键，而BC范式则是对R的每个属性都做要求。

在关系模式STJ（S，T，J）中，S表示学生，T表示教师，J表示课程。每一教师只教一门课。每门课由一名教师教，某一学生选定某门课，就确定了一个固定的教师。某个学生选修某个教师的课就确定了所选课的名称 ： (S，J)→T，(S，T)→J，T→J

## 第9章 SQL语言

### 数据库语言

#### 一、数据库语言概述 

##### 1.数据定义语言DDL

负责数据结构定义与数据库对象定义的语言，由CREATE、 ALTER与DROP三个语法组成。 

##### 2.数据操纵语言DML

实现对数据库的基本操作。如，对表中数据的SELECT 、INSERT、 DELETE和UPDATE。

#### 二、数据库语言的分类 

作为独立语言使用 

嵌入到高级语言中使用：嵌入式SQL和宿主语言

### 数据库定义

#### 一、创建表(CREATE TABLE)

语句格式： CREATE TABLE<表名>（<列名><数据类型>[列级完整性约束条件] [，<列名><数据类型>[列级完整性约束条件]]… [，<表级完整性约束条件>]）；

列级完整性约束条件有NULL（空）和UNIQUE（取值惟一）

#### 二、修改表和删除表

##### 1．修改表 (ALTER TABLE)

语句格式 : ALTER TABLE <表名>[ADD<新列名><数据类型> [完整性约束条件]] [DROP<完整性约束名>] [MODIFY<列名><数据类型>]；

##### 2．删除表 (DROP TABLE)

语句格式 : DROP TABLE <表名>

#### 三、创建和删除索引

##### 1．建立索引

语句格式 : CREATE [UNIQUE] [CLUSTER] INDEX <索引名> ON<表名>（<列名>[<次序>][，<列名>[<次序>]]…）；

说明：

(1)次序：可选ASC(升序)或DSC(降序) 默认值为ASC。

(2) UNIQUE：表明此索引的每一个索引值只对应惟一的数据记录。

(3) CLUSTER：表明要建立的索引是聚簇索引，索引项的顺序是 与表中记录的物理顺序一致的索引组织。

##### 2. 删除索引

语句格式： DROP INDEX <索引名>

#### 四、视图的创建和删除

##### 1．视图的创建

CREATE VIEW 视图名（列表名） AS SELECT 查询子句 [WITH CHECK OPTION];

创建视图必须遵循如下规定：

(1)子查询可以是任意复杂的SELECT语句，但通常不允许含有order by子句和DISTINCT短语。

(2) WITH CHECK OPTION表示执行UPDATE，INSTER或DELETE操作时 保证更新、插入或删除的行满足视图定义中的谓词条件（即子查 询中的条件表达式）。

(3)组成视图的属性列名或者全部省略或者全部指定。如果省略属 性列名，则隐含该视图由SELECT子查询目标列的主属性组成

##### 2．视图的删除

DROP VIEW 视图名

### 数据操作

#### 一、Select 语句

用于执行数据库的查询。

语句格式：

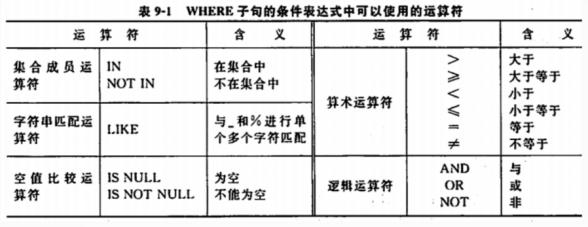
SELECT [ALL | DISTINCT] <目标列表达式> [ ,<目标列表达式>]…

FROM <表名或视图名> [ ,<表名或视图名>]

[WHERE<条件表达式>]

[GROUP BY<列名1>[HAVING<条件表达式>]]

[ORDER BY<列名2>[ASC|DESC]...]



##### 简单查询

例1:查询2012年计算机系“CS” 学生的学号、娃名及出生年份。

S(Sno,Sname,SD,Sage,Sex,Addr)

T(Tno,Tname,Tage,Sex)

C(Cno,Cname,Pcno)

SC(Sno,Cno,Grade)

SELECT Sno ,Sname ,2012-Sage FROM S WHERE SD='CS'

##### 连接查询

查询涉及两个以上的表，称为连接查询。

例2:检索选修了课程号为”C1”的学生号和学生姓名。

SELECT Sno, Sname FROM S, SC WHERE S.Sno=SC.Sno ADN SC.Cno='C1'

##### 3.子查询与聚集函数

###### （1）子查询

子查询也称嵌套查询。嵌套查询是指一个SELECT-FROM-WHERE 查询块可以嵌入另一个查询块之中。SQL中允许多重嵌套。

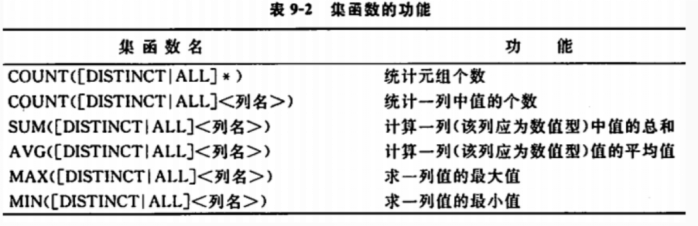
例3:采用嵌套查询来实现例2。

SELECT Sno ,Sname FROM S WHERE Sno IN (SELECT Sno FROM SC WHERE Cno='C1')

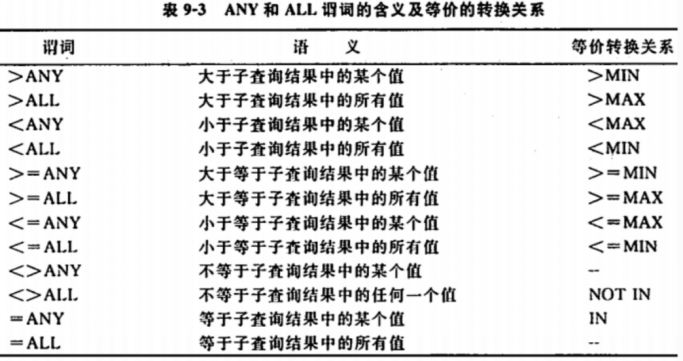
###### （2）聚集函数

聚集函数是以一个值的集合为输入，返回单个值的函数。

SQL提供了5个预定义的集函数：平均值AVG、最小值MIN、最大值MAX、求和SUM以及计数COUNT。



使用ANY和ALL谓词必须同时使用比较运算符，用集函数实现子查 询通常要比直接用ALL或ANY查询效率要高



##### 4.分组查询

###### （1）GROUP BY子句

例1：针对学生数据库中的SC关系，查询每个学生的平均成绩。

SELECT Sno，AVG(Grade) FROM SC GROUP BY Sno

###### （2）HAVING子句

在GROUP BY子句后面加一个HAVING子句，对分组设置过滤条件，(可以使用聚集函数)

注意，当元组含有空值时，记住以下两点： 

1. 空值在任何聚集操作中都会被忽视，如求和、求平均值和计数 都没有影响。如，COUNT(\*)是某个关系中所有元组数目之和， 但COUNT(A)却是A属性非空的元组个数之和。 
2. NULL值可以看作分组属性中的一个一般的值。例如，在SELECT A，AVG(B) FORM R中，当A的属性值为空时，就会统计A=NULL 的所有元组中B的均值。

##### 5.更名操作

###### AS子句

old-name AS new-name As子句既可出现在SELECT子句中，也可出现在FROM子句中。

例3：查询计算机学生的Sname和Sage，但Sname用"姓名"表示， Sage用"年龄"表示。 SELECT Sname AS 姓名，Sage AS 年龄 FROM S WHERE SD='CS'

##### 6.字符串操作

对于字符串进行的最通常的操作是使用操作符like的模式匹配 ， 使用两个特殊的字符来描述模式： 

“%” 匹配任意字符串 

“\_”匹配任意一个字符

模式是大小写敏感的。

%idge%

mary%

mary\_

##### 7.视图查询与更新

###### （1）Create view 语句创建视图

语法格式如下： Create view view\_name [(<列名1>，<列名2>，...)] As select\_statement [with check option]

select\_statement：用来创建视图的select语句，可在select语句 中查询多个表或视图。 

with check option：指出在视图上所进行的修改都要符合 select\_statement所指定的限制条件，这样可以确保数据修改后， 仍可通过视图看到修改的数据。

对select语句有以下限制： 

定义视图的用户必须对所参照的表或视图有查询（即可执行 select语句）权限。 

不能使用order by子句，除非有TOP语句。 

不能使用into子句。 

不能在临时表或表变量上创建视图。

###### （2）视图查询

将视图看成实体表进行查询即可

###### （3）视图删除

视图删除的语句如下：DROP VIEW 视图名

###### （4）视图更新

视图更新必须遵循以下规则：

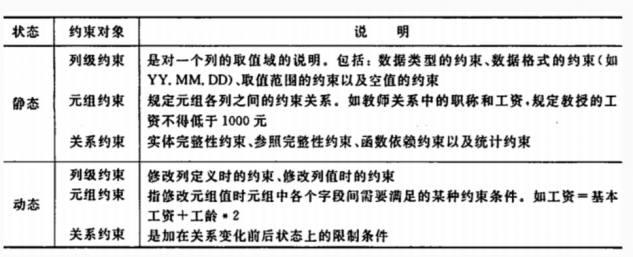
1. 从多个基本表通过连接操作导出的视图不允许更新。
2. 对使用了分组、集函数操作的视图不允许进行更新。
3. 如果视图是从单个基本表通过投影、选取操作导出的，则允许进行更新操作，且语法同基本表。

### SQL中的授权

#### 一、主键约束PRIMARY KEY

##### 1.完整性约束条件

完整性约束条件作用的对象有关系、元组和列



##### 2.完整性控制

完整性控制具有的功能：

定义功能、检测功能和处理功能 检查是否违背完整性约束的时机有两种： 

在一条语句执行完后立即检查称为立即执行约束； 

需要延迟到整个事务执行完后再检查称为延迟执行约束。

在关系系统中，最重要的完整性约束是实体完整性和参照完整性。

##### 3.实体完整性（使用“PRIMARY KEY”子句）

关系中只能有一个主键。

声明主键有两种方法：

1. 属性类型后加PRIMARY KEY 保留字；
2. 在属性列表最后增加一行，PRIMARY KEY（属性或属性组）

#### 二、外键约束FOREIGN KEY

参照完整性定义格式： FOREIGN KEY (属性名) REFERENCES 表名（属性名） [ON DELETE[CASCADE|SET NULL] 

FOREIGN KEY定义哪些列为外码， 

REFERENCES指明外码对应于哪个表的主码， 

ON DELETE CASCADE指明删除被参照关系的元组时，同时删除参照关系中的元组。 

SET NULL表示置为空值。

#### 三、属性值上的约束

属性值上的约束通过下面关键字进行： 

not null 不允许空值 

unique 唯一值 

check 属性值要满足指定条件

#### 四、全局约束CREATE ASSERTIONS

全局约束是指涉及到多个属性间的联系或多个不同关系间的联系约束。全局约束有两种：

##### 1.基于元组的检查子句

对单个关系的元组值加以约束。方法是在关系定义中的任何地方 加上关键字CHECK和约束条件。

例如：学生年龄在16-20岁之间 CHECK (Sage>=16 AND Sage<=20)

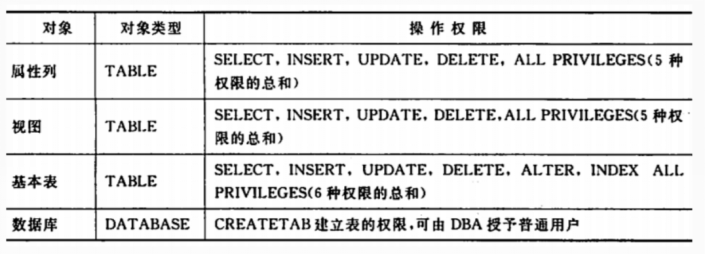
##### 2.基于断言的语法格式

格式：CREATE ASSERTION <断言名> CHECK (<条件>)

#### 五、授权(GRANT)与销权(REVOKE)

##### 1.授权语句格式

GRANT<权限>[,<权限>]… [ON<对象类型><对象名>] TO<用户>[，<用户]>]… [WITH GRANT OPTION]；



##### 2.收回权限语句格式

REVOKE<权限>[，<权限>]… [ON<对象类型><对象名>] FROM<用户>[，<用户]>]…；

### 创建与删除触发器

#### 一、概述

触发器(trigger)是SQL server 提供给程序员和数据分析员来保证数据完整性的一种方法，它是与表事件相关的特殊的存储过程，它 的执行不是由程序调用，也不是手工启动，而是由事件来触发。 

触发器经常用于加强数据的完整性约束和业务规则等 

触发器与存储过程的唯一区别是触发器不能执行EXECUTE语句调用，而是在用户执行Transact-SQL语句时自动触发执行。

#### 二、创建触发器

触发器的定义包括两个方面：

指明触发器的触发事件 

指明触发器的执行动作



事件的触发有两个时间： 

Before：事件发生之前触发 

After：事件发生之后触发

CREATE TRIGGER <触发器名> [{BEFORE |AFTER}]

{[DELETE|INSERT|UPDATE OF[列名清单]]}

ON 表名 [REFERENCING<临时视图名>]

[WHEN <触发条件>]

BEGIN

<触发动作>

END [触发器名]

#### 三、更改和删除触发器

##### 1.更改触发器

ALTER TRIGGER <触发器名> [{BEFORE|AFTER}]

{[DELETE|INSERT|UPDATEOF[列名清单]]}

ON表名|视图名

AS

BEGIN

要执行的SQL语句

END

##### 2.删除触发器

DROP TRIGGER删除指定的触发器，其语法格式如下：

DROP TRIGGER <触发器名>[,.....n]

## 第10章 系统开发和运行

### 软件工程基础

#### 一、软件危机与软件工程

##### 1、软件危机

##### 2、软件工程

软件工程是将系统化的、严格约束的、可量化的方法应用于软件的开发、运行和维护，即将工程化应用于软件。

软件工程方法学包括三个要素：方法、工具和过程。

#### 二、软件生命周期

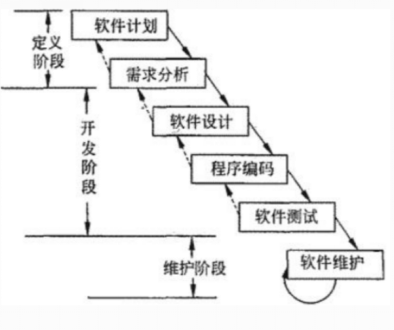
是指软件的产生直到报废的生命周期，包括：问题定义、可行性分析、需求分析、总体设计、详细设计、编码、测试、运行维护等阶段。

#### 三、软件开发模型

常见的开发模型有：瀑布模型、增量模型、螺旋模型、喷泉模型、智能模型、V模型、快速应用开发模型、构件组装模型、 敏捷方法和统一过程等。

##### 1.瀑布模型

瀑布模型也称为生命周期法，是结构化方法中最常用的开发模型， 它把软件开发的过程分为软件计划、需求分析、软件设计、程序 编码、软件测试和运行维护6个阶段。



###### 瀑布模型的优点：

1. 为项目提供了按阶段划分的检查点。
2. 当前一阶段完成后，只需要去关注后续阶段。
3. 它提供了一个模板，这个模板使得分析、设计、编码、测试和支持的方法可以在该模板下有一个共同的指导。

###### 瀑布模型的缺点：

(1)各个阶段之间产生大量的文档，极大地增加了工作量。

(2)由于开发模型是线性的，用户只有等到整个过程的末期才能见 到开发成果，从而增加了开发风险。

(3)不适应用户需求的变化，并且在需求分析阶段不可能完全获取。

(4)在软件开发前期未发现的错误传到后面的开发活动中时，可能 会扩散，进而可能会导致整个软件项目开发失败。

所以，瀑布模型适用于需求明确或很少变更的项目

##### 2.快速原型模型

快速原型是利用原型辅助软件开发的一种新思想。 经过简单快速分析，快速建造一个可以运行的软件原型，以便理解和澄清问题，使开发人员与用户达成共识，最终在确定的客户需求基础上开发客户满意的软件产品。

原型可以分为三类： 

###### 探索型原型

主要用于需求分析阶段，目的是要弄清用户的需求，并探索各种方案的可行性。它主要针对开发目标模糊，用户与开发人员对项目都缺乏经验的情况，通过对原型的开发来明确用户的需求。 

###### 实验型原型

主要用于设计阶段，考核实现方案是否合适，能否实现。对于大型系统，若对设计方案心中没有把握时，可通过这种原型来证实设计方案的正确性。

###### 演化型原型

主要用于及早向用户提交一个原型系统，该原型系统或者包含系统的框架，或者包含系统的主要功能，在得到用户的认可后，将原型系统不断扩充演变为最终的软件系统。它将原型的思想扩展到软件开发的全过程。

##### 3.演化模型

也称为变换模型，根据用户的基本需求，通过快速分析构造出一个初始可运行版本(原型)，然后根据用户在使用原型的过程中提出的意见和建议对原型进行改进，获得原型的新版本。重复这一过程，最终可得到令用户满意的软件产品。

快速原型模型是“抛弃式”的，演化模型是“渐进式”原型方法。

演化模型特别适用于对软件需求缺乏准确认识的情况。

###### 演化模型的优点：

(1)很早就可以验证是否符合产品需求。

(2)风险管理可以在早期就获得项目进程数据，可据此对后续的开发进度作出比较切实的估算。增加项目成功的机率。

(3)经验教训能反馈于本产品的下一个循环过程，提高质量效率。

(4)心理上，开发人员早日见到产品的雏型，是一种鼓舞。

(5)使用户可以在新的一批功能开发测试后，立即参加验证，以便提供非常有价值的反馈。

###### 演化模型的缺点

(1)产品需求在一开始并不完全弄清楚的话，会给总体设计带来困难及削弱产品设计的完整性，并影响产品性能的优化。

(2)如果缺乏严格的过程管理，这个生命周期模型可能退化为一种原始的无计划的“试－错－改”模式。

(3)用户接触开发中的尚未测试稳定的功能，可能对用户都产生负面的影响。

##### 4.增量模型

融合了瀑布模型的基本成分和原型实现的迭代特征，是第三种原型化开发方法，但它不是“抛弃式”的，也不是“渐进式”的。 增量模型把软件产品划分为一系列的增量构件，第一个增量往往是核心的产品，即第一个增量实现了基本的需求。客户对每一个增量的使用和评估都作为下一个增量发布的新特征和功能，这个过程在每一个增量发布后不断重复，直到产生了最终的完善产品。

增量模型与原型实现模型和其他演化方法一样，本质上是迭代的，但与原型实现不一样的是其强调每一个增量均发布一个可 操作产品。

增量模型的特点是引进了增量包的概念，无须等到所有需求都出来，只要某个需求的增量包出来即可进行开发。

###### 增量模型优点

(1) 人员分配灵活，初期不用太大投入。

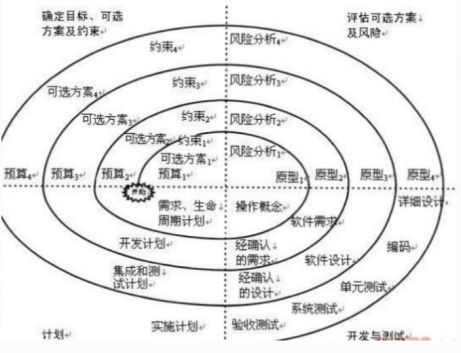
(2) 每隔一小段时间就提交用户部分功能，用户可以直观感受项目进展，及时试用产品功能。

(3)有利于风险的把控。

增量模型将功能细化、分别开发的方法适应于需求经常改变的软件开发过程

##### 5.螺旋模型

将瀑布模型和演化模型相结合，综合了两者的优点，并增加了风险分析。它以原型为基础，沿着螺线自内向外旋转，每旋转一圈 都要经过制订计划、风险分析、 实施工程及客户评价等活动，并开发原型的一个新版本。经过若干次螺旋上升的过程，得到最终的系统。



###### 螺旋模型的优点：

(1)设计上灵活，可以在项目的各个阶段进行变更；

(2)以小的分段来构建大型系统，使成本计算变得简单容易；

(3)客户始终参与每个阶段的开发，保证了项目不偏离正确方向；

(4) 随着项目推进，客户始终掌握项目的最新信息 , 从而能够和管 理层有效地交互。

###### 螺旋模型的缺点：

1. 需要具有相当丰富的风险评估经验和专门知识，如果未能够及时标识风险，势必造成重大损失；
2. 过多的迭代次数会增加开发成本，延迟提交时间。

##### 6.喷泉模型

是一种以用户需求为动力，以对象为驱动的模型，主要用于描述面向对象的软件开发过程，该模型认为软件开发过程自下而上的，各阶段是相互迭代和无间隙的。

无间隙是指在开发活动中，分析、设计和编码之间不存在明显的边界。

##### 7.基于构件的开发模型

将整个系统模块化，并在一定构件模型的支持下复用构件库中的一个或多个软件构件，通过组合手段高效率、高质量地构造应用 软件系统的过程。

基于构件的开发模型由软件的需求分析和定义、体系结构设计、构件库建立、应用软件构建以及测试和发布5个阶段组成。

###### 优点：

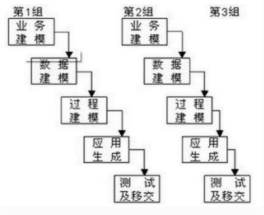
构件复用，提高了软件开发的效率。构件可由一方定义其规格说明，被另一方实现。然后供给第三方使用，构件组装模型允许多个项目同时开发，降低了费用，提高了可维护性，可实现分步提交软件产品。

###### 缺点：

缺乏通用的组装结构标准，因而引入了较大的风险。可重用性和软件高效性不易协调，需要精干的有经验的分析和开发人员。客户的满意度低，并且由于过分依赖于构件，所以构件库的质量影响着产品质量。

##### 8.快速应用开发模型（RAD）

是一个增量型的软件开发过程模型。强调极短的开发周期。RAD 模型是瀑布模型的一个“高速”变种，通过大量使用可复用构件，采用基于构件的建造方法赢得快速开发。如果需求理解得好且约束了项目的范围，随后是数据建模、过程建模、应用生成、测试及反复。



##### 9.敏捷开发

敏捷开发以用户的需求进化为核心，采用迭代、循序渐进的方法 进行软件开发。

###### 敏捷宣言 

个体和交互胜过过程和工具 

响应变化胜过遵循计划 

可以工作的软件胜过面面俱到的文档 

客户合作胜过合同谈判

###### 敏捷开发方法 

Scrum 

极限编程(XP) 

自适应软件开发 

水晶方法 

特征驱动开发(FDD)

###### 敏捷开发的原则 

每日立会 

小版本发布 

较少的文档 

合作为重 

客户参与 

自动化测试 

适应性计划调整 

结对编程 

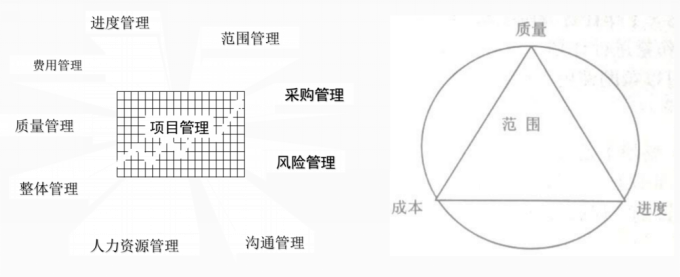
测试驱动开发 

持续集成 

重构

#### 四、项目管理

##### 1.项目管理内容



##### 2.软件项目估算

###### 1)软件规模估算方法 

估算的内容包括： 

软件规模的估算 

软件工作量的估算 

成本估算

LOC估算法（代码行估算法）

估算软件的代码行数。将软件项目划分为小的模块，通过历史数据、开发人员经验对LOC数进行估算。 

FP估算 FP（功能点）

是一种衡量工作量大小的单位。功能点=信息处理规模\*技术复杂度

信息处理规模： 

外部输入数 

外部输出数 

外部查询数 

内部逻辑文件数 

外部接口文件数

技术复杂度=0.65+调节因子

2)软件工作量估算

COCOMO模型是一种精确的、易于使用的工作量估算方法。

按其详细程度分为三级：

基本COCOMO模型：用已估算出来的代码行数(LOC)为自变量的经验函数计算软件开发工作量。 

中间COCOMO模型：在基本模型的基础上，再用涉及产品、硬件、人员、项目等方面的影响因素调整工作量的估算。

详细COCOMO模型：在中间模型的基础上，按模块层、子系统层、系统层，做出三张工作量因素分级表，供不同层次估算使 用。

##### 3.风险管理

是指对项目风险从识别到分析乃至采取应对措施等一系列过程，主要包括

编制风险管理计划 

风险识别 

定性风险分析 

定量风险分析 

风险应对计划的编制 

风险跟踪和监控

###### 1）识别项目中的风险方法 

头脑风暴法 

专家评估法 

风险检查表 

假设分析

###### 2）IT项目中常见的风险 

需求风险 

技术风险 

团队风险 

关键人员风险 

预算风险 

范围风险

##### 4.项目计划编排的方法与技术

###### 1）计划评审技术（PERT）

PERT是利用网络分析制定计划以及对计划予以评价的技术。它能协调整个计划的各道工序，合理安排人力、物力、时间、资金， 加速计划的完成。

PERT对各个项目活动的完成时间按三种情况估计：

(1)乐观时间：任何事情都顺利的情况下，完成某项工作的时间。

(2)最可能时间：正常情况下，完成某项工作的时间。

(3)悲观时间：最不利的情况下，完成某项工作的时间。

可算出每个活动的期望ti



根据P分布的方差计算方法，第i项活动的持续时间方差为：

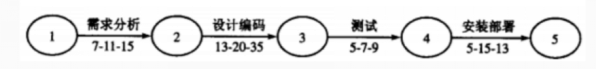


ai表示第i项活动的乐观时间

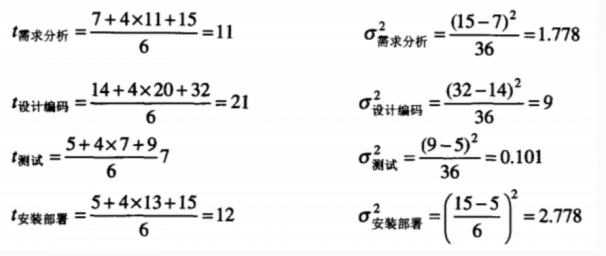
mi 表示第i项活动的最可能时间

bi 表示第i项活动的悲观时间。

例：某政府OA系统的建设可分解为需求分析、设计编码、测试、安装部署四个活动，各个活动依次进行，没有时间上的重叠，活动的 完成时间估计如图所示。

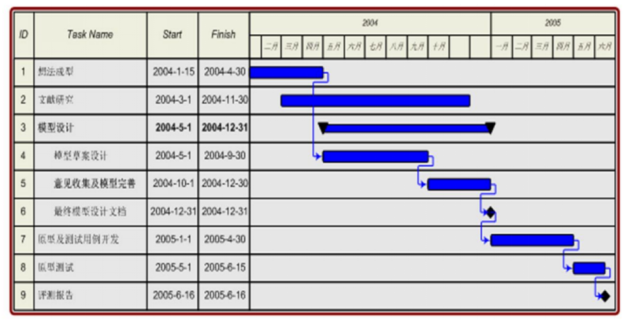


则各活动的期望工期和方差为？



###### 2)甘特图

甘特图也叫横道图，它以横线来表示每项活动的起止时间。



优点：

简单、明了、直观、易于编制的，因此到目前为止仍然是小型项目中常用的工具。



缺点：

不能系统地表达各项工作之间的复杂关系，难以进行定量的计算和分析，以及计划的优化，同时也没有指出影响 项目寿命周期的关键所在。

在大型工程项目中，它是高级管理层了解全局、基层安排进度时有用的工具。

###### 3)关键路径法(CPM)

是借助网络图和各活动所需时间（估计值)，计算每一活动的最早或最迟开始和结束时间。

CPM算法的核心思想是将工作分解结构（WBS)分解的活动按逻辑关系加以整合，统筹计算出整个项目的工期和关键路径。

每个活动有四个和时间相关的参数：

**最早开始时间（ES)。**某项活动能够开始的最早时间。

**最早结束时间（EF)。**某项活动能够完成的最早时间。 EF=ES+工期估计

**最迟结束时间（LF)。**为了使项目按时完成，某项工作必须完成的最迟时间。

**最迟开始时间（LS)。**为了使项目按时完成，某项工作必须开始的最迟时间。 LS=LF-工期估计

**CPM方法有两个规则**

规则1 :某项活动的最早开始时间必须相同或晚于直接指向这项活动的最早结束时间中的最晚时间

规则2 :某项活动的最迟结束时间必须相同或早于该活动直接指向的所有活动最迟开始时间的最早时间

根据以上规则，可以计算出工程的最早完工时间。

通过正向计算 （从第一个活动到最后一个活动）推算出最早完工时间，步骤如下：

( 1 )从网络图始端向终端计算。

( 2 )第一任务的开始为项目开始。

( 3 )任务完成时间为开始时间加持续时间。

( 4 )后续任务的开始时间根据前置任务的时间和搭接时间而定。

( 5 )多个前置任务存在时，根据最迟任务时间来定

通过反向计算（从最后一个活动到第一个活动）推算出最晚完工时间，步骤如下：

( 1 )从网络图终端向始端计算。

( 2 )最后一个任务的完成时间为项目完成时间。

( 3 )任务开始时间为完成时间减持续时间。

( 4 )前置任务的完成时间根据后续任务的时间和搭接时间而定。

( 5 ) 多个后续任务存在时，根据最早任务时间来定

最早开始时间和最晚开始时间相等的活动成为关键活动，关键活动串联起来的路径成为关键路径，关键路径的长度即为项目的工期。

一般可以通过两个原则来确认关键路径：

(1)总持续时间最长的线路称为关键线路

(2)总时差最小的工作组成的线路为关键线路



总时差是指一项工作在不影响总工期的前提下所具有的机动时间。 

总时差=该工作最迟完工时间-该工作最早完工时间 

总时差=该工作最迟开工时间-该工作最早开工时间

自由时差指一项工作在不影响其紧后工作最早开始时间的条件下，本工作可以利用的机动时间。 

自由时差=该工作的紧后工作最早开时间-该工作最早完工时间

### 系统分析基础

#### 一、结构化分析方法（SA）

SA是一种面向数据流的需求分析方法，它的基本思想是自顶向下逐层分析。

SA方法的分析结果包括：分层数据流图、数据字典、加工逻辑说明、补充材料。

##### （一）数据流图

###### 1.数据流图组成 

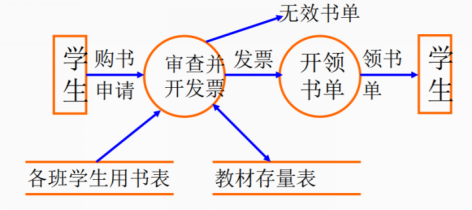
数据流 

加工 

数据存储 

外部实体





DFD不同于程序流程图。 

DFD可以表现大到整个系统，小到一个模块

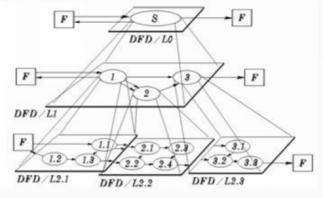
###### 2.分层数据流图的画法 

画系统的输入和输出。 

画系统的内部 

画加工的内部 

对第三步分解出来的DFD子图中的每个加工重复第三步的分解过程，直至圈中尚未分解的加工都足够简单（也就是说这种加工不必再分解）为止。



###### 3.对图和加工进行编号

1)父图与子图

如果父图中有n个加工，那么它可以有0～n张子图。见上图

2)编号 

顶层图只有一张，图中的加工也只有一个，所以不必编号。 

0层图只有一张，图中的加工号可以分别是0.1，0.2，…或者是1，2，… 

子图号就是父图中被分解的加工号。 

子图中的加工号由图号、圆点和序号组成。

3)实例

某考务处理系统有如下功能： 

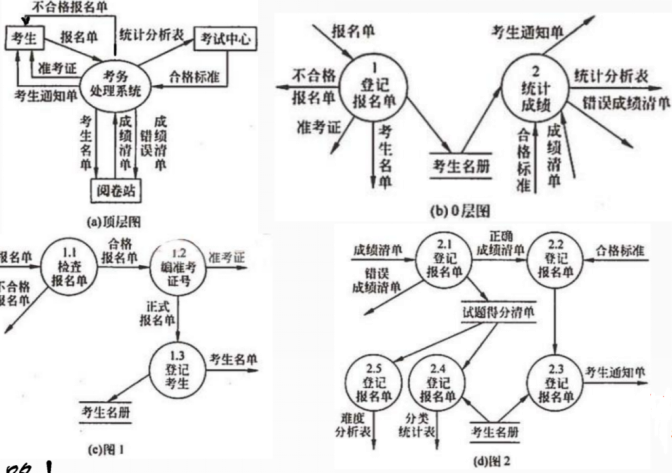
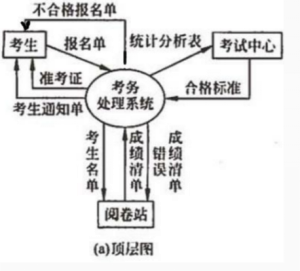
对考生送来的报名单进行检查； 

对合格的报名单生成准考证给考生； 

对阅卷站送来的成绩清单进行检查，并根据考试中心指定的合格标准审定合格者； 

制作考生通知单（内含成绩合格／不合格标志）送给考生； 

按地区、年龄、文化程度、职业和考试级别等进行成绩分类 统计和试题难度分析，产生统计分析表。



应注意的问题 

* 适当地为数据流、加工、数据存储及外部实体命名，名字应反映该成分的实际含义，避免使用空洞的名字。 
* 画数据流而不要画控制流。 
* 一个加工的输出数据流不应与输入数据流同名，即使它们的组成成分相同。 
* 允许一个加工有多条数据流流向另一个加工，也允许一个加工有两个相同的输出数据流流向两个不同的加工。
* 保持父图与子图平衡。也就是说，父图中某加工的输入输出数据流必须与它的子图的输入输出数据流在数量和名字上相同。 如果父图的一个输入（或输出）数据流对应于子图中几个输入（或输出）数据流，而子图中组成这些数据流的数据项全体正好是父图中的这一个数据流，那么它们仍然算是平衡的。 
* 在自顶向下的分解过程中，若一个数据存储首次出现时只与一个加工有关，那么这个数据存储应作为这个加工的内部文件而不用画出。
* 保持数据守恒：一个加工的所有输出数据流中的数据必须能从该加工的输入数据流中直接获得，或者是通过该加工能产 生的数据。 
* 每个加工必须既有输入数据流，又有输出数据流。 
* 在整套数据流图中，每个数据存储必须既有读的数据流，又有写的数据流。但在某一张子图中可能只有读没有写，或者只有写没有读

##### （二）数据字典（DD）

是对数据流程图各个组成部分的详细说明，是为数据流图中的每 个数据流、文件和加工，以及组成数据流或文件的数据项做出说明，是分析阶段的重要工具。 

数据流 

数据存储 

数据项 

加工

#### 二、UML分析与设计

UML又称统一建模语言或标准建模语言，是一个支持模型化和软件系统开发的图形化语言，它的作用域不仅支持面向对象的分析 与设计，还支持从需求分析开始的软件开发的全过程。

##### 建模的意义：

模型是对现实的简化，建模是为了更好地理解系统。 

模型帮助我们按照实际情况或需求对系统可视化；(掌握不了文字，画幅画代替) ； 

模型允许我们详细说明系统的结构、行为； 

模型给出了一个构造系统的模板； 

模型对我们作出的决策进行文档化。（先有文档，再有代码）

##### UML的特点： 

统一的标准，已经被OMG接受为标准建模语言 

面向对象，支持面向对象开发 

可视化，表示能力强 

独立于开发过程，可以适用于不同软件过程 

概念明确，表示简洁，结构清晰，容易学习掌握

##### （一）用例图（use case diagram）

描述一组用例、参与者（一种特殊的类）及它们之间的关系。

用例模型描述的是外部执行者所理解的系统功能，用于需求分析阶段。



###### 1.用例之间的关系

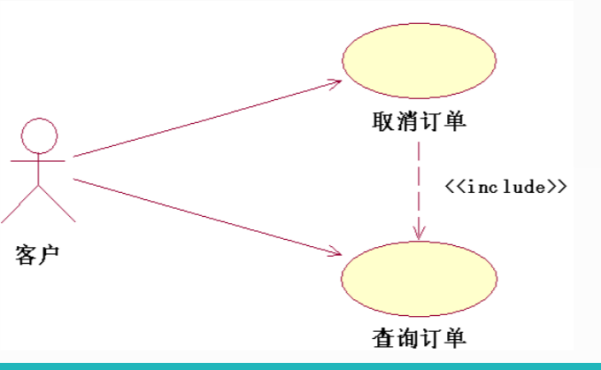
(1)包含（include） （是一种依赖关系，加了版型<<include>>） 

两个以上用例有共同功能，可分解到单独用例，形成包含依赖； 

箭头方向由基本用例指向被包含用例； 

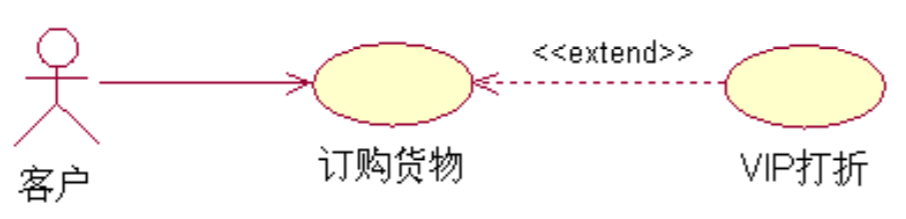
执行基本用例时，每次都必须调用被包含的用例； 

被包含用例也可以单独执行；



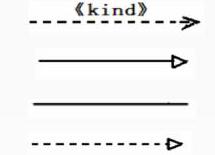
(2)扩展（extend） （是一种依赖关系，加了版型<<extend>>） 

如果一个用例明显地混合了两种或两种以上的不同场景，即根 据情况可能发生多种事情，则可以断定将这个用例分为一个主 用例和一个或多个辅用例进行描述可能更加清晰



##### （二）类图

描述一组类、接口、协作以及它们之间关系的图，用来显示系统中各个类的静态结构图。

类之间的关系（relationship）

依赖（dependency） 

泛化（generalization） 

关联（association） 

实现（realization）

###### 1.依赖（dependency） 

依赖关系是一种使用关系，特定事物的改变有可能会影响到使用该事物的其他事物。大多数情况下，依赖关系体现在某个类 的方法使用另一个类的对象作为参数。 

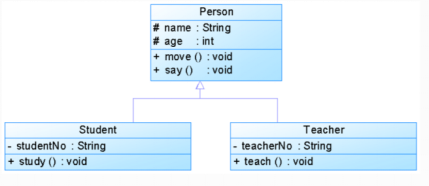
在UML中，依赖关系用带箭头的虚线表示，由依赖的一方指向被依赖的一方。



###### 2.泛化（generalization） 

一般元素和特殊元素之间的关系。 

泛化关系是继承关系的反关系，子类从父类中继承，父类是子类的泛化。



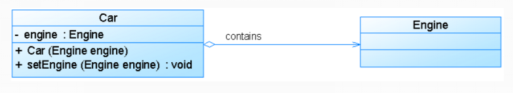
###### 3.关联（accociation） 

表示两个类之间存在某种语义上的联系。一个人为一家公司工作，一家公司有许多办公室。 

关联关系是所有关系中语义最弱的。可以分为聚合关系、组合关系。

（1）聚合关系

关联关系的一种特例，是强的关联关系。聚合表示类之间的关系 是整体与部分的关系，但整体与部分之间是可分离的，他们可以具有各自的生命周期。 在UML中，使用48T3{FHM`MC$OLI{Y(V1Q(F 表示聚合关系，空心菱形指向的是代表“整体”的类。

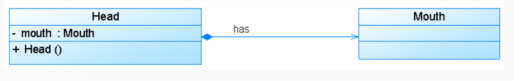


（2）组合关系

也是关联关系的一种特例，这种关系比聚合更强，也称为强聚合；

他同样体现整体与部分间的关系，但此时整体与部分是不可分的， 整体的生命周期结束也就意味着部分的生命周期结束。

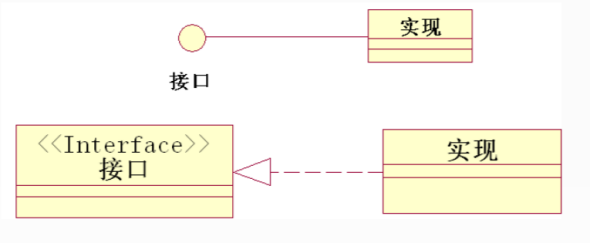
在UML中，使用带有实心菱形的实线P55N@NFIFU7BAWZJIWH8Y{T 表示组合关系。



###### 4.实现（realization） 

一个元素完成另外一个元素的操作功能，则二者构成实现关系。 

如接口类及其实现；接口是操作的集合，只声明了操作方法 （没有实现该方法），而由实现类具体定义实现部分。



##### （三）状态图（state chart diagram） 

状态图用来描述一个特定对象的所有可能状态及引起 状态转移的事件。 

它由状态、转移、事件和活动组成。 

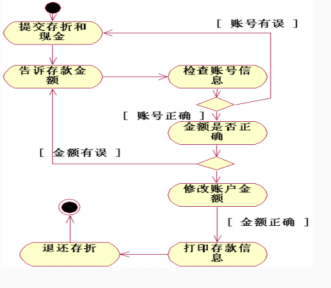
状态图给出了对象的动态视图

##### （四）活动图（activity diagram）

将进程或其他计算的结构展示为计算内部一步步的控制流和数据流。活动图专注于系统的动态视图。 

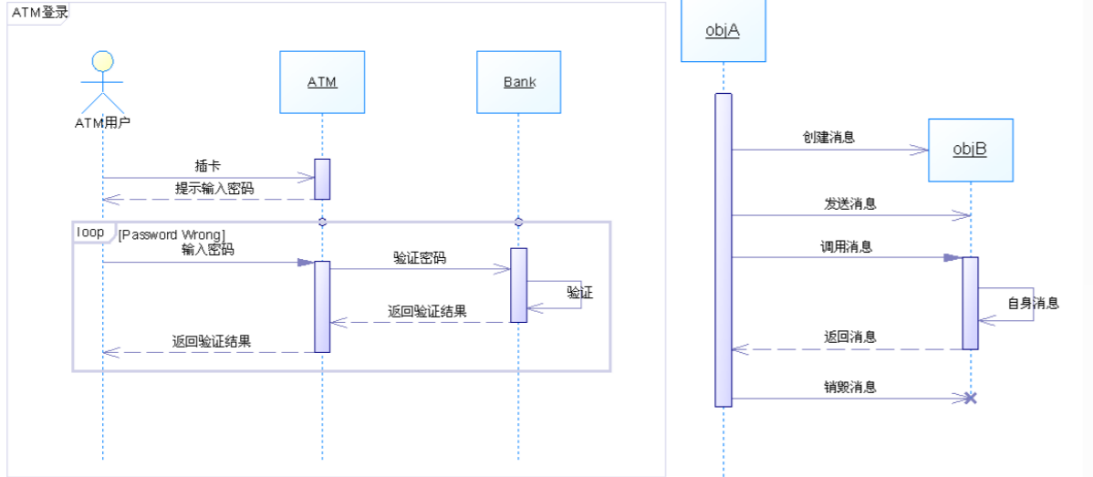
活动图侧重从行为的动作来描述 

状态图侧重从行为的结果来描述



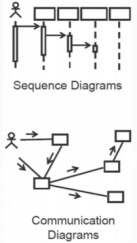
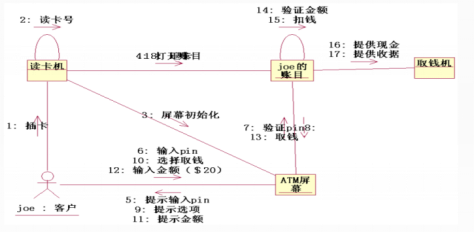
##### （五）顺序图（sequence diagram）

是一种交互图（interaction diagram），交互图展现了一 种交互，它由一组对象或角色以及它们之间可能发送的消息构成。交互图专注于系统的动态视图。顺序图是强调消息的时间次序的交互图。



##### （六）协作图（通信图，UML2.0后的名称）

是一种交互图，强调的是发送和接收消息的对象之间的组织结构。一个协作图显示了一系列的对象和在这些对象之间的 联系以及对象间发送和接收的消息。

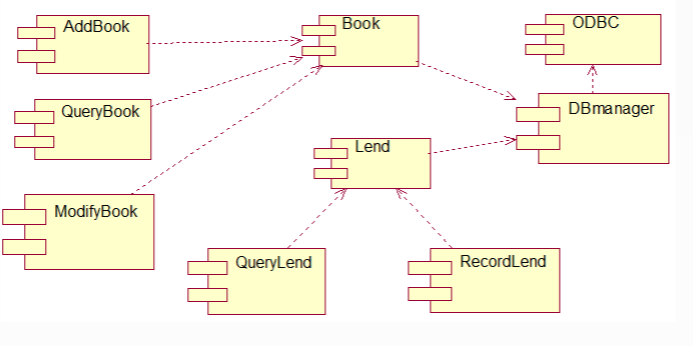


##### （七）构件图（component diagram） 

构件图是用来表示系统中构件与构件之间，类或接口与构 件之间的关系图。由源代码文件、二进制代码文件、可执 行文件或动态链接库 (DLL) 等构件构成，并通过依赖关系相连接。 

构件图用于表示系统的静态设计实现视图。 

是物理方面进行建模的两种图之一。

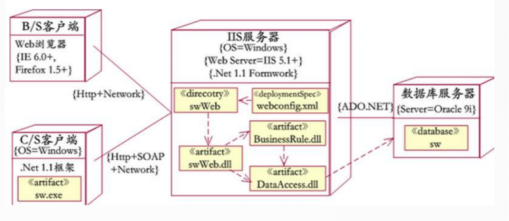


##### （八）部署图（deployment diagram） 

用来显示系统中软件和硬件的物理架构。 

从部署图中，可以了解到软件和硬件组件之间的物理关系以及处理节点的组件分布情况。 

是物理方面进行建模的两种图之一



##### 模型图分为二大类： 

静态模型（系统结构） 用例图、类图、对象图、构件图、部署图 

动态模型（系统行为） 状态图、活动图、顺序图、协作图

### 系统设计

#### 一、软件设计基本原则

##### （一）模块

是指执行某一特定任务的数据结构和程序代码。 

将模块的接口和功能定义为其外部特性 

将模块的局部数据和实现该模块的程序代码称为内部特性。

在模块设计时，最重要的原则就是实现信息隐蔽和模块独立

##### （二）信息隐蔽

将每个程序的成分隐蔽或封装在一个单一的设计模块中，并且尽可能少地暴露其内部的处理过程。

信息隐蔽可以提高软件的可修改性、可测试性和可移植性。

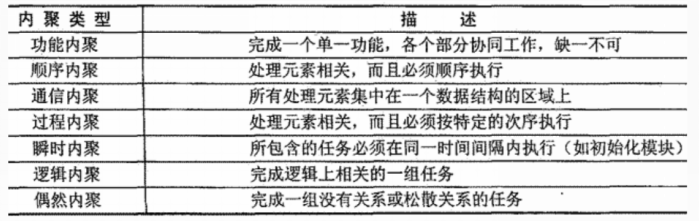
##### （三）模块独立

模块独立是指每个模块完成一个相对独立的特定子功能，并且与其他模块之间的联系最简单。

通常用耦合（模块之间联系的紧密程度）和内聚（模块内部各元素之间联系的紧密程度）两个标准来衡量，我们的目标是“高内 聚、低耦合”。

##### **（四）内聚**

指模块内部各元素之间联系的紧密程度。模块的内聚类型分为7种，根据内聚度从高到低的排序。



###### 1.功能内聚：

指模块内所有元素共同完成某一功能，联系紧密，缺一不可，是最强的内聚类型。

###### 2.顺序内聚：

指一个模块中各个处理元素都密切相关于同一功能且必须顺序执行，前一功能元素输出是下一功能元素的输入。即一个模块完成多个功能，这些模块又必须顺序执行。

###### 3.通信内聚：

指模块内所有处理元素都在同一个数据结构上操作，或者指各处理使用相同的输入数据或者产生相同的输出数据

###### 4.过程内聚：

构件或者操作的组合方式是，允许在调用前面的构件或操作之后，马上调用后面的构件或操作，即使两者之间没有数据进行传递。

###### 5.时间内聚：

把需要同时执行的动作组合在一起形成的模块为时间内聚模块，所有的动作需在同一个时间段内执行。

###### 6.逻辑内聚：

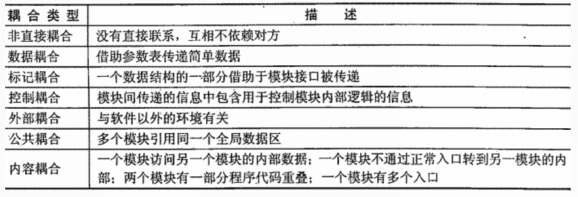
把几种相关的功能组合在一起，每次被调用时，由传送给模块参数来确定该模块应完成哪一种功能

###### 7.偶然内聚：

模块内各部分之间没有联系，或者有联系，这种联系也很松散，是内聚度最低的模块。

##### （五）耦合

指模块之间联系的紧密程度。模块的耦合类型分为7种，根据耦合度从低到高排序。



###### 1.非直接耦合：

两个模块之间没有直接关系，它们之间的联系完全是通过主模块的控制和调用来实现的。这种模块的耦合度最低、模块独立性最强。

###### 2.数据耦合：

指两个模块之间有调用关系，传递的是简单的数据值，相当于高级语言的值传递。

###### 3.标记耦合:

指两个模块之间传递的是数据结构，如高级语言中的数组名、记录名、文件名等这些名字即标记，其实传递的是这个数据结构的地址。

###### 4.控制耦合：

指一个模块调用另一个模块时，传递的是控制变量（如开关、标志等），被调模块通过该控制变量的值有选择地执行模块内某一功能。

###### 5.外部耦合：

一组模块都访问同一全局简单变量而不是同一全局数据结构，而且不是通过参数表传递该全局变量的信息。

###### 6.公共耦合：

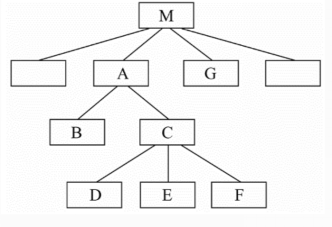
指一组模块都访问同一个公共数据环境，如全局数据结构，共享通信区。

###### 7.内容耦合:

一个模块直接使用另一个模块的内部数据，或通过非正常入口而转入另一个模块内部。是最差的耦合。

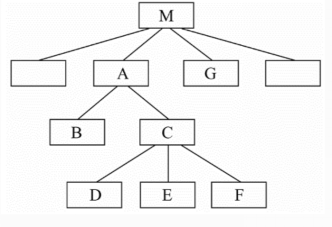
##### （六）深度

表示软件结构中控制的层数，它往往能粗略地标志一个系统的大小和复杂程度。如果层数过多则应该考虑是否有许多管理模块过分简单，能否适当合并。



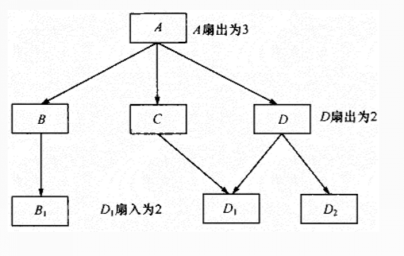
##### （七）宽度

是软件结构内同一个层次上的模块总数的最大值。宽度越大系统越复杂。对宽度影响最大的因素是模块的扇出。



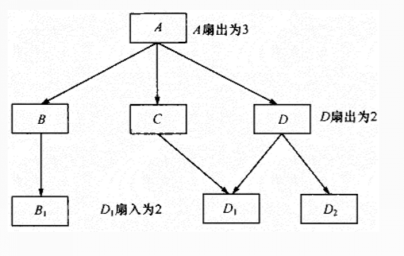
##### （八）扇出

模块的扇出是指一个模块直接控制(调用)的下层模块数目。扇出过大意味着模块过分复杂，需要控制和协调过多的下级模块；扇出过小也不好。设计得好的系统平均扇出是3或4。



##### （九）扇入

是指有多少个上级模块调用它，扇入越大则共享该模块的上级模块数目越多。



##### 在模块分解时需要注意： 

保持模块的大小适中 

尽可能减少调用的深度 

直接调用该模块的次数应该尽最多，但调用其他模块的次数则不宜过多(扇入大，扇出小)。好的软件设计结构顶层高扇出，中间扇出较少，底层高扇入。 

保证模块是单入口、单出口的 

模块的作用域应该在模块之内 

功能应该是可预测的

### 系统测试与维护

软件测试是指在规定的条件下对程序进行操作，以发现程序错误，衡量软件质量，并对其是否能满足设计要求进行评估的过 程。 

软件的正确性证明尚未得到根本的解决，软件测试仍是发现软件错误和缺陷的主要手段。

#### 一、软件测试基础

测试用例是由测试数据和预期结果构成的。 

* 为了发现程序中的错误，应竭力设计能暴露错误的测试用例 
* 好的测试用例是发现至今为止尚未发现的错误。 
* 一次成功的测试是发现了至今尚未发现的错误的测试

#### 二、软件测试准则 

* 应该尽早地、不断的进行测试，软件测试贯穿于开发过程的始终。 
* 所有测试都应该能追溯到用户需求。 
* 应该从小规模测试开始，并逐步进行大规模测试。 
* 应该远在测试之前就制定出测试计划。 
* 80%的错误可能出现在20%的程序模块中。 
* 应该由独立的第三方从事测试工作。
* 对非法和非预期的输入数据也要像合法数据一样编写测试用例。 
* 检查软件是否做了应该做的事仅是成功一半，另一半是看软件是否做了不该做的事。 
* 在规划测试时不要设想程序中不会查出错误。 
* 测试只能证明软件中有错误，不能证明软件中没有错误。

#### 三、软件测试分类

从测试阶段分：单元测试、集成测试、确认测试和系统测试

从测试方法分：白盒测试、黑盒测试

回归测试是指修改了旧代码后，重新进行全部或部分以前的测试用例，以确认修改没有引入新的错误或导致其他代码产生错误。

#### 四、测试的阶段

根据测试的目的、阶段的不同，测试分为：单元测试、集成测试、确认测试、系统测试

##### （一）单元测试

单元测试又称为模块测试，是针对软件设计的最小单位（模块）进行正确性检验的测试工作。

目的在于检查每个程序单元能否正确实现详细设计说明中的模块功能、性能、接口和设计约束等要求，以及发现各模块内部可能 存在的各种错误。

单元测试通常由开发人员自己负责。 

单元测试要借助驱动模块（相当于用于测试模拟的主程序）和桩模块（子模块）来完成。 

单元测试的计划是在软件详细设计阶段完成的。 

单元测试一般使用白盒测试方法。

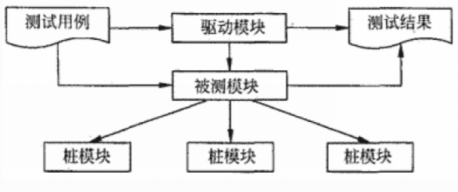
模块并不是一个独立的程序，在考虑测试模块时，同时要考虑它和外界的联系，用一些辅助模块去模拟与被测模块相联系的 其他模块。这些辅助模块分为两种：

###### 1.驱动模块：

相当于被测模块的主程序。它接收测试数据，把这些数据传送给被测模块，最后输出实测结果。

###### 2.桩模块：

用于代替被测模块调用的子模块。桩模块可以做少量的数据操作，不需要把子模块所有功能都带进来，但不允许什么事情也不做。



##### （二）集成测试

集成测试也称为组装测试、联合测试。它将已通过单元测试的模块集成在一起，主要测试模块之间的协作性。

从组装策略而言，可以分为一次性组装和增量式组装（包括自顶 向下、自底向上及混合式）两种。 

集成测试计划是在软件概要设计阶段完成的。 

集成测试一般采用黑盒测试方法。

自顶向下进行组装，不需要驱动模块。 

自底向上进行组装，不需要桩模块。 

在每个版本提交时，都需要进行“冒烟”测试，即对程序主要功能进行验证。冒烟测试也称为版本验证测试或提交测试。

##### （三）确认测试

确认测试也称为有效性测试，主要包括验证软件的功能、性能及其他特性是否与用户要求（需求）一致。

确认测试计划是在需求分析阶段完成的。

根据用户的参与程度，包括以下3种类型：

###### 1.内部确认测试：

由软件开发组织内部按软件需求说明书进 行测试。

###### 2.Alpha测试：

由用户在开发环境下进行测试。

###### 3.Beta测试：

由用户在实际使用环境下进行测试。

##### （四）系统测试

是将已经确认的软件、计算机硬件、外设、网络等其他元素结合在一起，进行信息系统的各种组装测试和确认测试，系统测试是针对整个产品系统进行的测试，目的是验证系统是否满足了需求规格的定义，找出与需求规格不符或与之矛盾的地方，从而提出更加完善的方案。

系统测试计划在系统分析阶段（需求分析阶段）完成。

系统测试的内容包括：功能测试、性能测试、健壮性测试、用户界面测试、安全性测试、安装与反安装测试

#### 五、 测试的类型

软件测试分为两大类：动态测试和静态测试。

##### （一）动态测试

动态测试指通过运行程序发现错误，分为：黑盒测试法、白盒测试法、灰盒测试法

###### 1.黑盒法

黑盒测试又称为功能测试或数据驱动测试。把被测试对象看成一个黑盒子，测试人员完全不考虑程序的内部结构和处理过程，只在软件的接口处进行测试，依据需求规格说明书，检查程序是否满足功能要求。

常用的黑盒测试用例的设计方法：等价类划分、边界值分析、错误推测、因果图

###### 2.白盒法

又称结构测试、透明盒测试、逻辑驱动测试或基于代码的测试。把测试对象看做一个打开的盒子，测试人员必须了解程序的内部结构和处理过程，以检查处理过程的细节为基础，对程序中尽可能多的逻辑路径进行测试，检验内部控制结构和数据结构是否有错，实际的运行状态与预期的状态是否一致。

常用的白盒测试用例设计方法有： 语句覆盖、判定覆盖、条件覆盖、条件判定覆盖、条件组合覆盖、路径覆盖等，发现错误的能力呈由弱至强的变化。 

语句覆盖每条语句至少执行一次。 

判定覆盖每个判定的每个分支至少执行一次。 

条件覆盖每个判定的每个条件应取到各种可能的值。 

判定/条件覆盖同时满足判定覆盖条件覆盖。 

条件组合覆盖每个判定中各条件的每一种组合至少出现一次。 

路径覆盖使程序中每一条可能的路径至少执行一次。

###### 3.灰盒法

灰盒测试是一种介于白盒测试与黑盒测试之间的测试，它关注输出对于输入的正确性，同时也关注内部表现，但这种关注不像白盒测试那样详细且完整，而只是通过一些表征性的现象、事件及标志来判断程序内部的运行状态。

灰盒测试结合了白盒测试和黑盒测试的要素，考虑了用户端、特定的系统知识和操作环境，在系统组件的协同性环境中评价应用软件的设计。

##### （二）静态测试

静态测试指被测试程序不在机器上运行，而采用人工检测和计算 机辅助静态分析的手段对程序进行检测。

静态分析中进行人工测试的主要方法：桌前检查( Desk Checking)、代码审查、代码走查

###### 1.桌前检查

由程序员自己检查自己编写的程序。程序员在程序通过编译之后，进行单元测试设计之前，对源程序代码进行分析、检验，并补充相关的文档，目的是发现程序中的错误。

###### 2.代码审查

代码审查是由若干程序员和测试员组成一个会审小组，通过阅读、讨论和争议，对程序进行静态分析的过程。代码审查分两步。 

第一步，小组负责人提前把设计规格说明书、控制流程图、程序文本及有关要求、规范等分发给小组成员，作为评审的依据。小组成员充分阅读这些材料。 

第二步，召开程序审查会。在会上，首先由程序员逐句讲解程序的逻辑。在此过程中，程序员或其他小组成员可以提出问题，展开讨论，审查错误是否存在。

###### 3.代码走查

代码走查与代码审查基本相同，其过程也分为两步。 

第一步，把材料先发给走查小组成员，认真研究程序，再开会。 

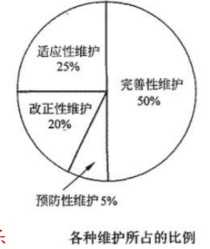
第二步，开会的程序与代码会审不同，不是简单地读程序和对照错误检查表进行检查，而是让与会者“充当”计算机。让测试用例沿程序的逻辑运行一遍，随时记录程序的踪迹，供分析和讨论使用。数据标准化、数据命名合理、文件格式转换、数据库格式转换等。

#### 六、软件维护

软件可维护性是指维护人员对该软件进行维护的难易程度，具体包括理解、改正、改动和改进软件的难易程度。

衡量程序可维护性的因素：可理解性、可测试性和可修改性等。

软件维护占整个软件生命周期的60%～80%，维护的类型有：改正性维护、适应性维护、完善性维护、预防性维护



###### 1.改正性维护

是指改正在系统开发阶段已发生而系统测试阶段尚未发现的错误。

###### 2.适应性维护

是指使用软件适应信息技术变化和管理需求变化而进行的修改。

###### 3.完善性维护

是为扩充功能和改善性能而进行的修改，主要是指对已有的软件系统增加一些在系统分析和设计阶段中没有规定的功能与性能特征。

###### 4.预防性维护

为了改进应用软件的可靠性和可维护性，为了适应未来的软硬件环境的变化，应主动增加预防性的新的功能，以使应用系统适应各类变化而不被淘汰。

##### 影响维护工作量的因素主要有：

系统大小、程序设计语言、系统年龄、数据库技术的应用、先进的软件开发技术

## 第11章 数据库设计

### 数据库设计概述

#### 一、数据库应用系统的生命周期

数据库规划 

需求描述与分析 

数据库设计与应用程序设计 

实现 

测试 

运行维护

#### 二、数据库设计一般策略 

自顶向下 

自底向上

#### 三、数据库设计的步骤 

用户需求分析 

概念结构设计 

逻辑结构设计 

物理结构设计

### 系统需求分析

#### 一、需求分析的任务、方法和目标 

信息要求 

处理要求 

系统要求

#### 二、需求分析阶段的文档 

需求说明文档 

数据流程图DFD 

数据字典DD

### 概念结构设计

#### 一、概念结构设计策略与方法

概念结构设计的策略通常有以下4种：

·自顶向下。

·自底向上。

·逐步扩张。

·混合策略。

E-R模型将现实世界的信息结构统一由实体、属性以及实体之间的联系来描述。

对现实事物抽象的3种方法： 

分类 

聚集 

概括

#### 二、用E-R方法建立概念模型

1.选择局部应用

2.逐一设计分E-R图

3.E-R图合并

#### 三、E-R图合并

##### （一）E-R图合并时的冲突

属性冲突 

命名冲突 

结构冲突

##### （二）E-R图合并时的优化 

实体类型的合并 

冗余属性的消除 

冗余联系的消除

### 逻辑结构设计

#### 逻辑结构设计的主要任务： 

确定数据模型 

将E-R图转换为指定的数据模型 

确定完整性约束 

确定用户视图

#### 一、E-R图向关系摸式的转换

E-R图是由实体、属性和联系三要素构成，而关系模型中只有惟一的结构--关系模式。

采用以下方法加以转换： 

实体向关系模式的转换 

联系向关系模式的转换

##### （二）实体向关系模式的转换 

每个实体类型转换成一个关系模式 

实体属性即为关系模式的属性 

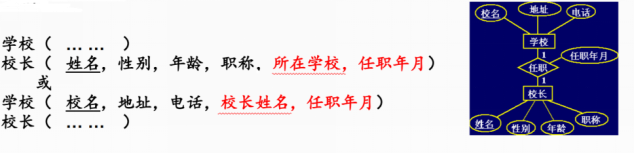
实体标识符即为关系模式的键

注意：得到的关系模式，有些可能会扩充属性

##### （三）联系向关系模式的转换 

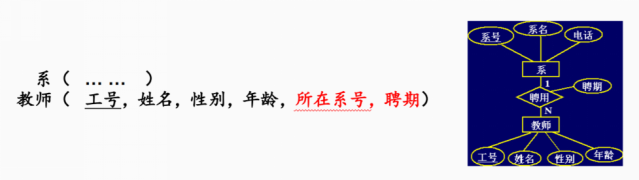
###### 1：1 联系 

1:1联系，联系两端的实体类型转成两个关系模式，在任一个关系模式中加入另一个关系模式的键（作为外键）和联系的属性



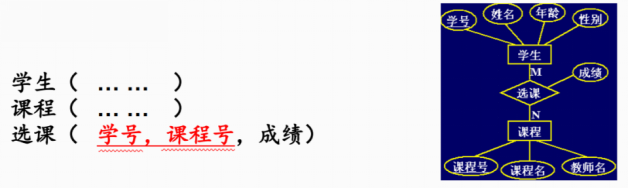
###### 1：N 联系 

1:N联系，在N端实体类型转换成的关系模式中，加入1端实体类型的键（作为外键）和联系的属性



###### M：N 联系

M:N联系，联系类型需转换为关系模式，属性为两端实体类型的键（分别作为外键）加上联系的属性，而键为两端实体键的组合



#### 二、关系模式的规范化

由E-R图转换得来的初始关系模式并不完全符合要求，还会有数据冗余或更新异常存在，需要经过进一步的规范化处理。步骤如下：

①根据语义确定各关系模式的数据依赖。

②根据数据依赖确定关系模式的范式。

③如果关系模式不符合要求，要根据关系模式的分解算法对其进行分解，达到3NF、BCNF或4NF。

④关系模式的评价及修正。

#### 三、确定完整性约束

根据规范化理论确定了关系模式之后，还要对关系模式加以约束，包括数据项的约束、表级约束及表间约束。可以参照soL 标准来确定不同的约束，如检查约束、主码约束以及参照完整性约束，以保证数据的正确性。

#### 四、用户视图的确定

根据数据流图及用户信息建立视图模式，提高数据的安全性和独立性。 

根据数据流图确定处理过程使用的视图 

根据用户类别确定不同用户使用的视图

### 数据库物理设计

#### 一、物理设计

根据数据库的逻辑结构来选定RDBMS（如Oracle、Sybase等），并设计和实施数据库的存储结构、存取方式等。

数据库的物理设计目标：

使设计出的物理数据库占用较少的存储空间 

对数据库的操作具有尽可能高的速度

#### 二、 物理设计工作步骤

物理设计应做以下工作： 

确定数据分布 

确定存储结构 

确定存取方式

##### （一）确定数据的分布 

按照数据应用的不同，确定数据的存放位置。 

从提高系统性能的角度确定数据的分布位置。 

数据库的数据备份、日志备份只在故障恢复时才使用，而且数据量很大，可以考虑放在磁带上 

多个磁盘驱动器或磁盘阵列的系统，将表和索引分别存放在不同的磁盘上，在查询时两个磁盘分别工作，读写快 

大的表分别放在不同的磁盘上，可以加快存取速度 

将日志文件和数据库对象（表、索引等）分别放在不同的磁可以改进系统性能。

##### （二）确定存储结构

存储结构指数据文件中记录之间的物理结构。

数据是以记录为单位存储的，可以是顺序存储、哈希存储、B+树存储等。要根据数据的处理要求和修改频度选定合适的物理结构。

###### 索引：

表中数据和相应存储位置的列表 

优点 大大的减少数据的查询时间 

缺点 每个索引都将需要占用一定的存储空间 对表中的数据进行增加、删除和修改，索引也要动态的维护

**在创建索引的时候，遵循以下的经验性原则：** 

在经常需要搜索的列上建立索引 

在主关键字上建立索引 

在经常用于连接的列上建立索引 

在经常需要根据范围进行搜索的列上建立索引 

在经常需要排序的列上建立索引 

在经常成为查询条件的列上建立索引

##### （三）确定存取方式

确定建立哪些存储路径以实现快速存取数据库中的数据。

DBMS提供的存取方法：索引方法、HASH法

## 第12章 数据库运行与管理

### 数据库系统的运行计划

#### 一、运行策略

运行策略要考虑正常运行策略和非正常运行策略两个方面。

正常运行策略是指在正常运行状态下的数据库执行策略。

**需要从以下几个方面考虑正常运行策略。**

1)系统运行对物理环境的要求

2)系统运行对人员的要求

3)数据库的安全性策略

4)数据库备份和恢复策略

非正常运行策略是指在特殊时期的数据库运行策略。

**非正常运行策略需要从以下几个方面考虑：**

1)突发事件的应对策略

2)高负载状态的应对策略

#### 二、数据库系统监控对象和监控方式

##### （一）数据库系统的监控对象 

性能监控 

故障监控 

安全监控

##### （二）数据库系统监控方式 

系统监控 

应用监控 

系统日志

### 数据库系统的运行和维护

#### 一、数据库维护

在数据库系统的运行过程中，可能会由于某些原因需要修改数据库的结构，称为数据库重构。

重构包括表结构的修改和视图的修改。

表结构的修改有数据列的增删与修改、约束的修改以及表的分解与合并。

由于DBMS具有一定的逻辑独立性，有些修改可能不需要修改应用程序：

(1)修改表中的属性列名或数据类型，必须修改使用该表的应用程序。因此应尽量少做这样的修改。

(2)增加和删除属性只须修改使用该列的应用程序。

(3)修改的约束如果是DBMS支持的约束，如主码约束、参照完整性约束和检查约束，一般不需要修改应用程序。复杂的约束可以通过修改触发器程序实现。

(4)表的分解可以通过建立与分解前的表同名的视图来避免修改应用程序。

在数据库重构过程中引入或修改视图，可能会影响数据的安全性，因此必须对视图进行评价和验证；保证不会因为数据库的重构而引起数据的泄密。文档必须与系统保持高度的一致性，否则会造成人为的困难和错误。数据库重构过程中的所有修改，必须在文档中体现出来。

### 数据库系统的管理

#### 一、数据字典的管理

数据字典(data dictionary)是存储在数据库中的所有对象信息的知识库，存有用户及其权限信息、所有数据对象、表的约束条件和统计分析数据库的视图等信息。通常把数据字典中存储的数据称为元数据(metadata)。

#### 二、数据完整性维护和管理

对数据完整性的维护和管理采用两种方式：

(1)对于DBMS管理的约束，可通过修改数据库的定义，如增加或删除实体完整性约束、参照完整性约束以及检查约束来实现。

(2)对于应用程序实现的复杂的完整性约束，可通过分析和修改应用程序（通常是触发器程序）来实现。

#### 三、数据库的存储管理

(1)索引文件和数据文件分开存储，事务日志存储在高速设备上

(2)适时修改数据文件和索引文件的页面大小；

(3)定期对数据进行排序；

(4)增加必要的索引项。

#### 四、备份和恢复

针对备份计划的制定和实施，有以下建议：

(1)根据数据变更情况，设定合理的备份周期和备份时间；

(2)把事务日志文件保存在最稳定的存储设备上；

(3)定期在事务日志文件中加入检查点(checkpoint)。

#### 五、数据安全性管理

(1)建立网络级安全，重点是防火墙的设置；

(2)操作系统级安全，进行登录用户的管理；

(3)DBMS级安全，对访问数据库的用户进行密码验证

(4)角色和用户的授权管理；

(5)建立视图和存储过程加强安全性；

(6)使用审计功能，为追究非法入侵者的法律责任提供证据，发现安全漏洞。

### 性能调整

#### 一、SQL语句的编码检验

(1)尽可能地减少多表查询或建立物化视图

(2)以不相关子查询替代相关子查询；

(3)只检索需要的列；

(4)用带IN的条件子句等价替换OR子句；

(5)经常提交COMMIT，以尽早释放锁。

#### 二、表设计的评价

根据实际情况对表进行调整，调整的原则

(1)频繁地对两个相关表进行连接操作，考虑将其合并；

(2)频繁地访问表中的某一部分字段，考虑分解表，将该部分单独作为一个表；

(3)对于很少更新的表，引入物化视图。

#### 三、索引改进

(1)如果查询是瓶颈，则在关系上建立适当的索引。通常，在作为查询条件的属性上建立索引可以提高查询效率。

(2)如果更新是瓶颈，因每次更新都会重建表上的索引，引起效率的降低，则考虑删除某些索引。

(3)选择适当的索引类型。如果经常使用范围查询，则B树索引比散列索引更高效。

(4)将有利于大多数据查询和更新的索引设为聚簇索引。

#### 四、设备增强

(1)引入高速的计算机

(2)增加系统内存

(3)使用高速的网络设备

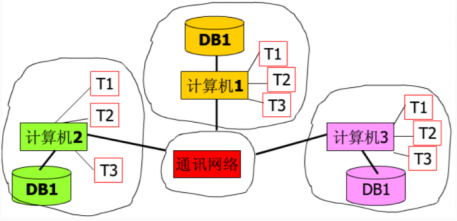
(4)使用高速的存储设备

## 第13章 数据库发展与新技术

### 分布式数据库

#### 一、分布式数据库的概念

分布式数据库是一个物理上分布在计算机网络的不同地点，而逻辑上又属于同一系统的数据集合。网络的每个站点的数据库都有自治能力，能够完成局部应用，同时每个站点的数据库又 属于整个系统，通过网络也可以完成全局应用。



#### 二、分布式数据库的特点 

数据的集中控制性 

数据独立性 

数据冗余可控性 

场地自治性 

存取的有效性

#### 三、数据分布 

集中式 

分割式 

复制式 

混合式

#### 四、数据分片 

水平分片 

垂直分片 

水平和垂直结合分片

**不论采用哪种分片，都要遵守如下规则 **

完备性条件 

可重构条件 

不相交性

#### (QDRAMEKQX3$59UP{62$)F7五、分布式数据库管理系统

分布式数据库管理系统组成： 

LDBMS（LOCAL DBMS）

GDBMS（GLOBLE DBMS） 

全局数据字典(GDD) 

通信管理(CM)

#### 六、分布式数据库查询和优化

与集中式数据库相比，分布式数据库查询还要考虑以下两方面：

(1)数据和信息均要通过通信线路进行传输，存在的延迟问题 将减慢整个查询的执行过程。

(2)网络中多处理器的存在提供了并行数据处理和传输的机会， 应充分利用以加快查询的速度。

查询优化： 

查询执行代价的优化 

查询响应时间的优化

#### 七、二阶段提交

两阶段提交保证分布式事务的原子性

数据库的一致性是由数据库的完整性约束实现

持久性则是通过commit日志来实现

把分布式事务的某一个代理指定为协调者(Coordinator)，所有其他代理称为参与者( Participant)。只有协调者才拥有提交或撤销事务的决定权，而其他参与者则各自负责在其本地数据库中执行写操作，并向协调者提出撤销或提交事务的意向。

##### 二阶段提交算法

两阶段提交的过程如下：

**第一阶段是表决阶段，目的是形成一个共同的决定。**

开始时，协调者在它的日志中写入一条开始提交的记录，再给所有参与者发送“准备提交”消息，并进入等待状态。当参与 者收到“准备提交”消息后，它检查是否能提交本地事务。如 果能提交，参与者在日志中写入一条就绪记录，并给协调者发送“建议提交”消息，然后进入就绪状态。否则，参与者写入撤销记录，并给协调者发送“建议撤销”消息。

如果某个节点做出“建议撤销”提议，由于撤销决定具有否决权（即单方面撤销），发出“建议撤销”的站点就可以直接忽略这个事务。

协调者收到所有参与者的回答后，就可以做出是否提交事务的决定。只要有一个参与者建议撤销，协调者就必须从整体上撤销整个分布式事务。因此写入一条撤销记录，并给所有参与者发送“全局撤销”消息，然后进入撤销状态。否则，写入提交记录，给所有的参与者发送“全局提交”消息，然后进入提交状态。

**第二阶段是执行阶段，目的是实现这个协调者的决定。**

根据协调者的指令，参与者或者提交事务，或者撤销事务，并给协调者发送确认消息。此时协调者在日志中写入一条事务结束记录并终止事务。

## 第14章 标准化和知识产权

### 标准化基础知识

#### 一、我国标准的级别

《中华人民共和国标准化法》将标准划分为4个层次：

(1)国家标准

(2)行业标准

(3)地方标准

(4)企业标准

除此之外还有国际标准，如：ISO(国际标准化组织)、IEC(国际电工委员会)、ITU(国际电信联盟)。

##### （一）国家标准

需要在全国范围内统一的技术要求，应当制定国家标准。国家标准由国家标准化管理委员会编制计划、审批、编号和发布。

国家标准代号为： 

GB：强制性国家标准 例：GB 18582-2001 

GB/T：推荐性国家标准 例：GB/T 30200-2013 

GB/Z：指导性国家标准 例：GB/Z 19579-2012 

GSB：国家实物标准 例：GSB G51001-94

国家标准在全国范围内适用，其他各级标准不得与之相抵触。国家标准是四级标准体系中的主体。

ANSI：美国国家标准协会标准

BS：英国国家标准

JIS：日本工业标准

##### （二）行业标准

对没有国家标准又需要在全国某个行业范围内统一的技术要求，可以制定行业标准，是专业性、技术性较强的标准。

行业标准由行业标准归口部门编制计划、审批、编号、发布和管理。

作为对国家标准的补充，当相应的国家标准实施后，该行业标准应自行废止。

部分行业的行业标准代号： 汽车-QC、 石油化工-SH、 化工-HG、 石油天然气-SY、有色金属-YS、 电子-SJ、 机械-JB、 轻工—QB、 船舶-CB、 核工业-EJ、 电力-DL、 商检-SN、 GJB：中华人民共和国军用标准

推荐性行业标准在行业代号后加／T，如:JB/T，为机械行业推荐 性标准，不加T为强制性标准。

##### （三）地方标准

对没有国家标准和行业标准而又需要在省、自治区、直辖市范围内统一要求，可以制定地方标准。

**地方标准的制定范围有： **

工业产品的安全、卫生要求； 

药品、兽药、食品卫生、环境保护、节约能源、种子等法律、法规的要求； 

其他法律、法规规定的要求。

地方标准由省、自治区、直辖市标准化行政主管部门统一编制计划、组织制定、审批、编号和发布，地方标准以DB开头。 

地方标准在本行政区域内适用，不得与国家标准和行业标准相抵触。 

国家标准、行业标准公布实施后，相应的地方标准即行废止。地方标准也分强制性与推荐性。

##### （四）企业标准

是对企业范围内需要协调、统一的技术要求、管理要求和工作要求所制定的标准。 

企业标准由企业制定，不得低于相应的国家标准或行业标准的要求，企业标准以Q开头。 

企业产品标准应在发布后30日内向政府备案。

#### 二、标准的有效期

自标准实施之日起，至标准复审重新确认、修订或废止的时间，称为标准的有效期，又称标龄。由于各国情况不同，标准有效期也不同。

以ISO标准为例，该标准每5年复审一次，平均标龄为4.92年。我国在国家标准管理办法中规定国家标准实施5年内要进行复审，即国家标准有效期为5年。

### 著作权法及实施条例

《中华人民共和国著作权法》及其实施条例，明确了保护文学、艺术和科学作品作者的著作权，以及与其相关的权益。

依据该法，我国不仅对文字作品，口述作品，音乐、戏剧、曲艺、舞蹈作品，美术、摄影作品，电影、电视、录像作品，工程设计、产品设计图纸及其说明、地图、示意图等图形作品给予保护，而且把计算机软件纳入著作权保护范围。

我国是世界上为数不多的明确将计算机软件作为著作权法保护客体的国家之一。

#### 一、著作权法主体

著作权法及实施条例的主体是指著作权关系人，通常包括著作权人、受让者两种。

(1)著作权人，又称为原始著作权人：是根据创作的事实进行确定的，依法取得著作权资格的创作、开发者。

(2)受让者，又称为后继著作权人：是指没有参与创作，通过著作权转移活动而享有著作权的人。

#### 二、著作权

(1)发表权

(2)署名权

(3)修改权

(4)保护作品完整权

(5)使用权、使用许可权和获取报酬权、转让权即以复制、表演、播放、展览、发行、摄制电影、电视、录像，或者改编、翻译、注释和编辑等方式使用作品的权利，以及许可他人以上述方式使用作品，并由此获得报酬的权利。

#### 三、著作权保护期限

作者的署名权、修改权、保护作品完整权的保护期不受限制。

公民的作品，其发表权、使用权和获得报酬权的保护期为作者终生及其死亡后50年，截止于作者死亡后第50年的12月31日；如果是合作作品，截止到最后死亡的作者死亡后第50年的12月31日。 法人或者其他组织的作品、著作权（署名权除外）由法人或者其他组织享有的职务作品，其发表权、使用权和获得报酬权的保护期为50年，截止于作品首次发表后第50年的12月31日，但作品自创作完成后50年内未发表的，著作权法不再保护。

#### 四、使用许可

对于出版者、表演者、录音录像制作者、广播电台、电视台而言，在下列情况下使用作品，可以不经著作权人许可、不向其支付报酬。但应指明作者姓名、作品名称，不得侵犯其他著作权。

(1)为个人学习、研究或欣赏，使用他人已经发表的作品。

(2)为介绍、评论某一个作品或说明某一个问题，在作品中适当引用他人已经发表的作品。

(3)为报道时事新闻，在报纸、期刊、广播、电视节目或新闻纪录影片中引用已经发表的作品。

(4)报纸、期刊、广播电台、电视台刊登或播放其他报纸、期刊、广播电台、电视台已经发表的社论、评论员文章。

(5)报纸、期刊、广播电台、电视台刊登或者播放在公众集会上发表的讲话，但作者声明不许刊登、播放的除外。

(6)为学校课堂教学或科学研究，翻译或者少量复制已经发表的作品，供教学或科研人员使用，但不得出版发行。

(7)国家机关为执行公务使用已经发表的作品。

(8)图书馆、档案馆、纪念馆、博物馆和美术馆等为陈列或保存版本的需要，复制本馆收藏的作品。

(9)免费表演已经发表的作品。

(10)对设置或者陈列在室外公共场所的艺术作品进行临摹、绘画、摄影及录像。

(11)将已经发表的汉族文字作品翻译成少数民族文字在国内出版发行。

(12)将已经发表的作品改成盲文出版

### 计算机软件保护条例

由于计算机软件也属于《中华人民共和国著作权法》保护的范围，因此在具体实施时，首先适用于《计算机软件保护条例》的条文规定，若是在《计算机软件保护条例》中没有规定适用条文的情况下，才依据《中华人民共和国著作权法》的原则和条文规定执行。

对软件著作权的保护只是针对计算机软件和文档，并不包括开发软件所用的思想、处理过程、操作方法或数学概念等，并且著作权人还需在软件登记机构办理登记。

#### 一、著作权人确定

根据《计算机软件保护条例》规定，软件开发可以分为合作开发、职务开发、委托开发三种形式。

##### （一）合作开发

对于由两个或两个以上的开发者或组织，合作开发的软件，著作权的归属根据合同约定确定。若无合同，则共享著作权。若合作开发的软件可以分割使用，那么开发者对自己开发的部分单独享有著作权，可以在不破坏整体著作权的基础上行使。

##### （二）职务开发

如果开发者在单位或组织中任职期间，所开发的软件符合以下条件，则软件著作权应归单位或组织所有。 

针对本职工作中明确规定的开发目标所开发的软件。 

开发出的软件属于从事本职工作活动的结果。 

使用了单位或组织的资金、专用设备、未公开的信息等物质、技术条件，并由单位或组织承担责任的软件。

##### （三）委托开发

如果是接受他人委托而进行开发的软件，其著作权的归属应由委托人与受托人签订书面合同约定；如果没有签订合同，或合同中未规定的，则其著作权由受托人享有。

由国家机关下达任务开发的软件，著作权的归属由项目任务书或合同规定，若未明确规定，其著作权应归任务接受方所有

#### 二、软件著作权

根据《计算机软件保护条例》规定，软件著作权人对其创作的软件产品，享有以下9种权利。

(1)发表权；即决定软件是否公之于众的权利。

(2)署名权：即表明开发者身份，在软件上署名的权利。

(3)修改权：即对软件进行增补、删节，或者改变指令、语句顺序的权利。

(4)复制权：即将软件制作一份或者多份的权利

(5)发行权：即以出售或者赠与方式向公众提供软件的原件或复制件的权利。

(6)出租权：即有偿许可他人临时使用软件的权利。

(7)信息网络传播权：即以信息网络方式向公众提供软件的权利。

(8)翻译权：即将原软件从一种自然语言文字转换成另一种自然语言文字的权利。

(9)使用许可权、获得报酬权、转让权。

软件著作权自软件开发完成之日起生效。

(1)著作权属于公民。著作权的保护期为作者终生及其死亡后的50年（第50年的12月31日）。对于合作开发的，则以最后死亡的作者为准。

(2)著作权属于单位。著作权的保护期为50年（首次发表后的第50年的12月31日），若50年内未发表的，不予保护。单位变更、终止后，其著作权由承受其权利义务的单位享有。

当得到软件著作权人的许可，获得了合法的计算机软件复制品后，复制品的所有人享有以下权利：

(1)根据使用的需求，将该计算机软件安装到设备中（计算机、 PDA等信息设备）。

(2)制作复制品的备份，以防止复制品损坏，但这些复制品不得通过任何方式转给其他人使用。

(3)根据实际的应用环境，对其进行功能、性能等方面的修改。但未经软件著作权人许可，不得向任何第三方提供修改后的软件

如果使用者只是为了学习、研究软件中包含的设计思想、原理，而以安装、显示和存储软件等方式使用软件，可以不经软件著作权人的许可，不向其支付报酬。

#### 三、侵权责任 

民事责任：停止侵害、消除影响、赔礼道歉、赔偿损失。 

行政责任：没收违法所得，没收复制品，处以罚款 

刑事责任

### 商标法及实施条例

#### 一、商标权

##### （一）商标的概念

商标指生产者及经营者为使自己的商品或服务与他人的商品或服务相区别，而使用在商品及其包装上或服务标记上的由文字、图形、字母、数字、三维标志和颜色组合，以及上述要素的组合所构成的一种可视性标志。

**以下标志不得作为商标： **

同国家、政府、国际组织名称相同、相似的 

同“红十字”、“红新月”的名称、标志相同或者近似的 

带有民族歧视性的，影响社会道德等性质的标志 

县级以上行政区划的地名或者公众知晓的外国地名

注册商标的有效期限为10年，自核准注册之日起计算。 

注册商标有效期满，需要继续使用的，应当在期满前6个月内申请续展注册； 

在此期间未能提出申请的，可以给予6个月的宽展期。 

宽展期满仍未提出申请的，注销其注册商标。 

每次续展注册的有效期为10年。

两个或者两个以上的申请人，在同一种商品或者类似商品上，分别以相同或者近似的商标在同一天申请注册的，各申请人应当自收到商标局通知之日起30日内提交其申请注册前在先使用该商标的证据。 

同日使用或者均未使用的，各申请人可以自收到商标局通知之日起30日内自行协商，并将书面协议报送商标局；

不愿协商或者协商不成的，商标局通知各申请人以抽签的方式确定一个申请人，驳回其他人的注册申请。 

商标局已经通知但申请人未参加抽签的，视为放弃申请，商标局应当书面通知未参加抽签的申请人。

### 专利法及实施细则

#### 一、专利权

包括发明、实用新型和外观设计三种：

**(1)发明：**是指对产品、方法或者其改进所提出的新的技术方案。科学发现不属于发明范畴。

**(2)实用新型：**是指对产品的形状、构造或者其结合所提出的适于实用的新的技术方案。

**(3)外观设计：**又称工业产品外观设计，是指对产品的形状、图案或者其结合以及色彩与形状、图案相结合所作出的富有美感并 适于工业上应用的新设计。

一份专利申请文件只能就一项发明创造提出专利申请。一项发明只授予一项专利，同样的发明申请专利，则按照申请时间的先后决定授予给谁。两个以上的申请人在同一日分别就同样的发明创造申请专利的，应当在收到国务院专利行政部门的通知后自行协商确定申请人。

##### （一）专利权的期限

发明专利权的期限为20年，实用新型专利权、外观设计专利权的期限为10年，均自申请日起计算。此处的申请日，是指向国务院专利行政主管部门提出专利申请之日。

##### （二）专利权人的义务

在保护期内，专利权人应该按时缴纳年费。在专利权保护期限内，如果专利权人没有按规定缴纳年费，或者以书面声明放弃其专利权，专利权可以在期满前终止。