[2022.4.17]

文档标题作为文字占位，只需单击占位更换文本添加内容

摘要

[软考]

[日期及邮编地址]

[ID名称]

[系统架构师]

[计算机系统知识 4](#_Toc102912616)

[计算机硬件组成 4](#_Toc102912617)

[CPU中央处理单元 4](#_Toc102912618)

[运算器的四个重要寄存器： 4](#_Toc102912619)

[控制器的四个重要寄存器： 4](#_Toc102912620)

[数据表示 4](#_Toc102912621)

[原码 4](#_Toc102912622)

[反码 4](#_Toc102912623)

[补码 4](#_Toc102912624)

[移码 4](#_Toc102912625)

[浮点数 4](#_Toc102912626)

[奇偶校验码 4](#_Toc102912627)

[循环冗余校验码 5](#_Toc102912628)

[海明码 5](#_Toc102912629)

[计算机体系结构 5](#_Toc102912630)

[体系结构分类 5](#_Toc102912631)

[按处理机的数量分类： 5](#_Toc102912632)

[Flynn分类法： 5](#_Toc102912633)

[指令系统 6](#_Toc102912634)

[指令寻址方式（寻找指令） 6](#_Toc102912635)

[指令操作数的寻址方式（寻找操作数） 6](#_Toc102912636)

[指令系统分类 6](#_Toc102912637)

[指令流水线 6](#_Toc102912638)

[RISC中的流水线技术 6](#_Toc102912639)

[流水线时间计算 7](#_Toc102912640)

[存储系统 7](#_Toc102912641)

[两级存储 7](#_Toc102912642)

[局部性原理 7](#_Toc102912643)

[cache 7](#_Toc102912644)

[地址映射 8](#_Toc102912645)

[替换算法 9](#_Toc102912646)

[命中率及平均时间 9](#_Toc102912647)

[磁盘结构和参数 9](#_Toc102912648)

[磁盘调度算法 9](#_Toc102912649)

[先来先服务FCFS 9](#_Toc102912650)

[最短寻道时间优先 9](#_Toc102912651)

[扫描算法scan 9](#_Toc102912652)

[单向扫描调度算法CSCAN 9](#_Toc102912653)

[输入输出技术 9](#_Toc102912654)

[计算机和外设间的数据交互方式 9](#_Toc102912655)

[程序控制（查询）方式 9](#_Toc102912656)

[程序中断方式 9](#_Toc102912657)

[DMA方式（直接主存存取） 10](#_Toc102912658)

[总线结构 10](#_Toc102912659)

[总线（bus） 10](#_Toc102912660)

[内部总线 10](#_Toc102912661)

[系统总线 10](#_Toc102912662)

[外部总线 11](#_Toc102912663)

[计算机可靠性 11](#_Toc102912664)

[可靠性指标 11](#_Toc102912665)

[串并联系统可靠性 11](#_Toc102912666)

[串联系统 11](#_Toc102912667)

[并联系统 11](#_Toc102912668)

[N模冗余系统 11](#_Toc102912669)

计算机组成与结构

计算机系统知识

计算机硬件组成

计算机基本硬件系统五大组成部分：运算器、控制器、存储器、输入和输出设备

CPU中央处理单元

组成：由运算器、控制器、寄存器组合内部总线组成

功能：实现程序控制、操作控制、时间控制、数据处理功能

运算器的四个重要寄存器：

算术逻辑单元ALU（实现对数据的算术和逻辑运算）

累加寄存器AC（运算结果或源操作数的存放区）

数据缓冲寄存器DR（暂时存放内存的指令或数据）

状态条件寄存器PSW（保存指令运行结果的条件码内容，如溢出标志等）

控制器的四个重要寄存器：

指令寄存器IR（暂存CPU执行指令）

程序计数器PC（存放指令执行地址）

地址寄存器AR（保存当前CPU所访问的内存地址）

指令译码器ID（分析指令操作码）

CPU依据指令周期的不同阶段来区分二进制的指令和数据

数据表示

原码

一个数的正常二进制表示。最高位表示符号，0为正数，1为负数。

反码

正数的反码即原码，负数的反码是在原码的基础上，除符号位外，其他各位按位取反。数值0的反码也有两种形式。

补码

正数的补码即原码；负数补码是在原码的基础上，除符号位外，其他各位各位按位取反，而后末位+1，若有进位则进位。（即反码末位+1）

移码

用作浮点运算的阶码，无论正数负数，都是将该原码的补码的首位（符号位）取反得到移码。

浮点数

奇偶校验码

码距：在两个编码中，从A码到B码转换所需改变的位数称为码距，如A：00要转换为B：11，需要改变两位，所以码距为2。一般来说，码距越大越利于纠错和检错。

奇偶校验码：在编码中增加1位校验位来使编码中1的个数变为奇数（奇校验）或偶数（偶校验），从而使码距变为2。例如奇校验：编码中含有奇数个1，发送给接收方，接收方会计算收到的编码有多少个1，如果是奇数个，则无误，偶数个，则有误。

只能检一位错

循环冗余校验码

可以校验多位，但是同样不能纠错

海明码

可以校验多位，能纠错，也是利用奇偶校验的形式

计算机体系结构

体系结构分类

按处理机的数量分类：

单处理系统：一个处理单元和其他设备集成

并行处理系统：两个以上的处理机互联

分布式处理系统：物理上远距离且松耦合的多计算机系统

Flynn分类法：

分类有两个因素，即指令流和数据流，指令流由控制部分处理，每一个控制部分处理一条指令流，多指令流就有多个控制部分

数据流由处理器来处理，每一个处理器处理一条数据流，多数据流就有多个处理器

至于主存模块是用来存储的，存储指令流或者数据流，因此无论是多指令流还是多数据流，都需要多个主存模块来存储，对于主存模块指令和数据都一样



多指令对应多控制部分，多数据对应多处理器

依据计算机特性，是由指令来控制数据的传输，因此一条指令可以控制一条或多条数据流，但一条数据流不能被多条指令控制，否则会出错，故多指令单数据MISD不可能。

现代多核处理器属于MIMD

指令系统

计算机指令的组成：一条指令由操作码和操作数两部分组成，操作码决定要完成的操作，操作数指参加运算的数据及其所在的单元地址。

计算机指令执行的过程：取指令—分析指令—执行指令三个步骤，首先将程序计数器PC中的指令地址取出，送入地址总线，CPU依据指令地址去内存中取出指令内容存入指令寄存器IR，而后由指令译码器进行分析，分析指令操作码，最后执行指令，取出指令执行所需的源操作数。

指令寻址方式（寻找指令）

顺序寻址：当执行一段程序时，是一条指令接着一条指令的顺序执行

跳跃寻址方式：指下一条指令的地址码不是由程序计数器给出，而是由本条指令直接给出。程序计数器的内容也必须相应改变，以便及时跟踪新的指令地址

指令操作数的寻址方式（寻找操作数）

立即寻址：指令的地址码字段指出的不是地址，而是操作数本身

直接寻址方式：在指令的地址字段中直接指出操作数在主存中的地址

间接寻址：指令地址码字段所指向的存储单元中存储的是操作数的地址

寄存器寻址：指令中的地址码是寄存器的编号

基址寻址：将基址寄存器的内容加上指令中的形式地址而形成操作数的有效地址，优点是可以扩大寻址能力

变址寻址：计算有效地址的方法与基址寻址方式很相似，它是将变址寄存器的内容加上指令中的形式地址而形成操作数的有效地址

指令系统分类

CISC：复杂指令系统，兼容性强，指令繁多，长度可变，由微程序实现

RISC：精简指令系统，指令少，使用频率接近，主要依靠硬件实现（通用寄存器、硬布线逻辑控制）



指令流水线

原理：将指令分成不同段，每段由不同的部分去处理，因此可以产生叠加的效果，所有的部件去处理指令的不同段

RISC中的流水线技术

超流水线：通过细化流水，增加级数和提高主频，使得在每个周期内完成一个甚至两个浮点操作，实质是以时间换空间

超标量：通过内装多条流水线来同时执行多个处理，其时钟频率虽然与一般流水接近，却有更小的CPI，实质是以空间换时间

超长指令字：与超标量类似，同时执行多条指令，不同在于超标量依靠硬件来实现并行处理的调度，超长指令字则充分发挥软件的作用，而使硬件简化性能提高

流水线时间计算

周期：指令分成不同执行段，其中执行时间最长的段为流水线周期

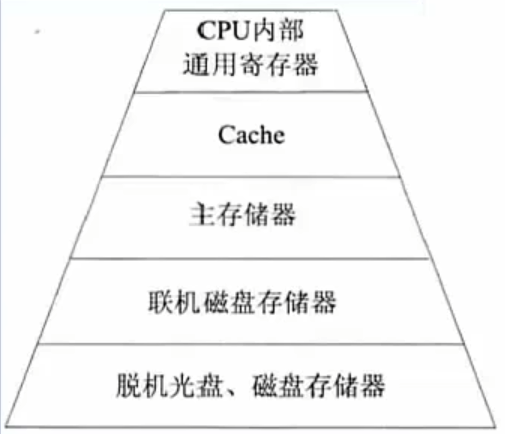
执行时间：1条指令总执行时间+（总指令条数-1）\*流水线周期

吞吐率：单位时间执行的指令条数。指令条数/流水线执行时间

加速比：使用流水线后的效率提升度，即比不使用流水线快了多少倍，越高表明流水线效率越高。不使用流水线执行时间/使用流水线执行时间

存储系统

计算机采用分级存储体系的主要目的是为了解决存储容量、成本和速度之间的矛盾问题



两级存储

cache—主存、主存—辅存（主存即内存）

局部性原理

总的来说，在cpu运行时，所访问的数据会趋向于一个较小的局部空间地址内，包括下面两个方面：

时间局部性原理

如果一个数据项正在被访问，那么在近期它很可能会被再次访问，即在相邻的时间里会访问同一个数据项

空间局部性原理

在最近的将来会用到的数据的地址和现在正在访问的数据地址很可能是相近的，即相邻的空间地址会被连续访问

cache

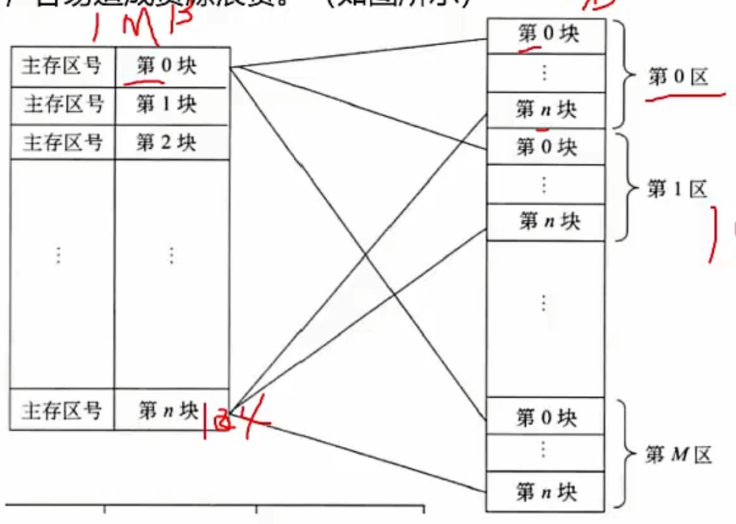
高速缓存cache用来存储当前最活跃的程序和数据，直接与cpu交互，位于cpu和主存之间，容量小，速度为内存的5-10倍，由半导体材料构成，其内容是主存内存的副本拷贝，对于程序员来说是不可见的

cache由控制部分和存储器组成，存储器存储数据，控制部分判断cpu要访问的数据是否在cache中，在则命中，不在则依据一定的算法从主存中替换

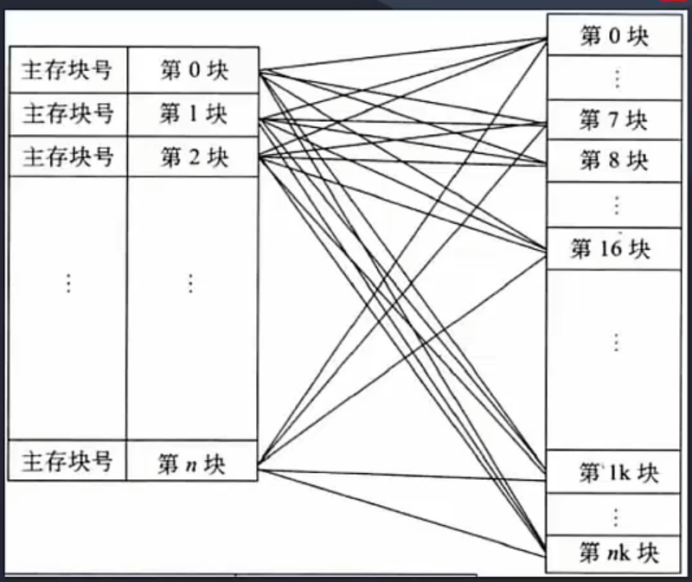
地址映射

在cpu工作时，送出的是主存单元的地址，而首先应从cache存储器中读写数据。这就需要将主存地址转换为cache存储器地址，这种地址的转换称为地址映像，由硬件自动完成映射，分为下列三种方法：

直接映像：将cache存储器等分成块，主存也等分成块并编号。主存中的块与cache中的块对应关系是固定的，即二者块号相同才能命中。地址变换简单但不灵活，容易造成资源浪费



全相联映射：同样都等分成块并编号，主存中任意一块都与cache中任意一块对应。因此可以随意调入cache任意位置，但地址变换复杂，速度较慢。因为主存可以随意调入cache任意块，只有当cache满了才会发生冲突，是最不容易发生块冲突的映像方式



组组相连映射：前面两种方式的结合，将cache存储器先分块在分组，主存也同样先分块再分组，组间采用直接映像，即主存中组号与cache中组号相同的组才能命中，但是组内全相联映像，也即组号相同的两个组内的所有块可以任意调换

替换算法

目标就是使cache获得尽可能高的命中率。常用的算法有如下几种

随机替换算法：用随机数发生器产生一个要替换的块号，将该块替换出去

先进先出算法：将最先进入cache的信息块替换出去

近期最少使用算法：这种方法是将近期最少使用的cache中的信息块替换出去

优化替换算法：该方法必须先执行一次程序，统计cache的替换情况，有了这样的先验信息，在第二次执行该程序时便可以用最有效的方式来替换

命中率及平均时间

cpu读取的平均时间为：cache命中率\*cache读取时间+（1-cache命中率）\*主存读取时间

磁盘结构和参数

磁盘有正反两个盘面，每个盘有多个同心圆，每个同心圆是一个磁道，每个同心圆又被划分为多个扇区，数据就被存放在一个个扇区中

磁头首先要寻找到对应的磁道，然后等待磁盘进行周期旋转，旋转到指定的扇区，才能读取到对应的数据，因此，会产生寻道时间和等待时间。公式为：存取时间=寻道时间+等待时间。

磁盘调度算法

存取时间中寻道时间耗时最长，需要重点调度，有如下调度算法：

先来先服务FCFS

根据进程请求访问磁盘的先后顺序进行调度

最短寻道时间优先

请求访问的磁道与当前磁道最近的进程优先调度，使得每次的寻道时间最短。会产生“饥饿”现象，即远处进程可能永远无法访问

扫描算法scan

又称“电梯算法”，磁头在磁盘上双向移动，其会选择离磁头当前所在磁道最近的请求访问的磁道，并且与磁头移动方向一致，磁头永远都是从里向外或者从外向里一直移动完才调头，与电梯类似

单向扫描调度算法CSCAN

与scan不同的是，其只做单向移动，即只能从里向外或者从外向里

输入输出技术

计算机和外设间的数据交互方式

程序控制（查询）方式

cpu主动查询外设是否完成数据传输，效率极低。

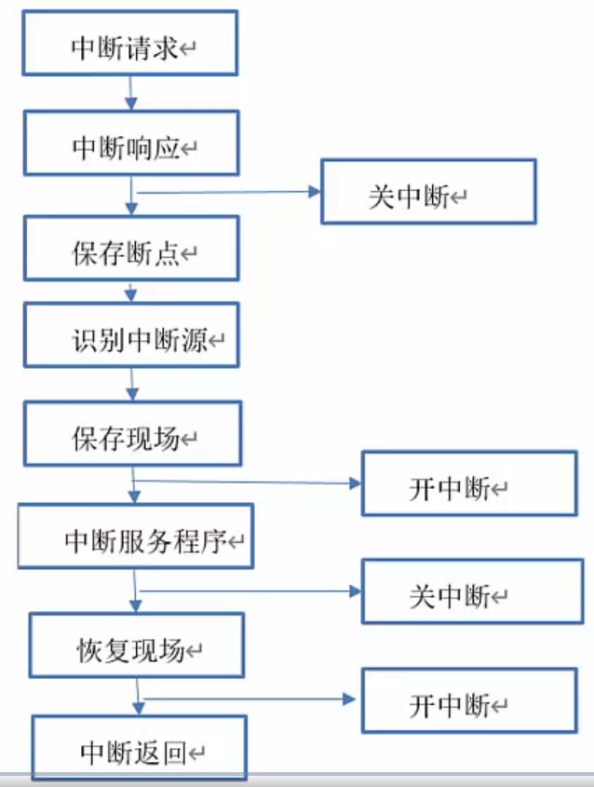
程序中断方式

外设完成数据传输后，向CPU发送中断，等待cpu处理数据，效率相对较高。中断响应时间指的是从发出中断请求到开始进入中断处理程序；中断处理时间：从中断处理开始到中断处理结束。中断向量提供中断服务程序的入口地址。多级中断嵌套，使用堆栈来保护断点和现场。

DMA方式（直接主存存取）

cpu只需完成必要的初始化等操作，数据传输的整个过程都由DMA控制器来完成，无cpu参与，在主存和外设之间建立直接的数据通路，效率很高

在一个总线周期结束后，cpu会响应DMA请求开始读取数据，cpu响应程序中断方式请求是在一条指令执行结束时



总线结构

总线（bus）

指计算机设备和设备之间传输信息的公共数据通道。总线是连接计算机硬件内多种设备的通信线路，它的一个重要特这是由总线上的所有设备共享，因此可以将计算机系统内的多种设备连接到总线上

广义来说，任何连接两个以上电子元器件的导线都可以叫总线，通常分为以下三类

内部总线

内部芯片级别的总线，芯片与处理器之间通信的总线

系统总线

是板级总线，用于计算机内各部分之间的连接，具体分为数据总线（并行数据传输位数）、地址总线（系统可管理的内存空间大小）、控制总线（传送控制命令）（这三种总线是考试会考的，属于教材内知识点。内部、系统、外部是老师扩展的部分）。代表有ISA总线、EISA总线、PCI总线

外部总线

设备一级的总线，微机和外部设备的总线。代表有RS232（串行总线）、SCSI（并行总线）、USB（通用串行总线，即插即用，支持热拔插）。

计算机可靠性

可靠性指标

平均无故障时间MTTF=1/失效率

平均故障修复时间MTTR=1/修复率

平均故障间隔时间MTBF=MTTF+MTTR

系统可用性=MTTF/(MTTF+MTTR)\*100%

串并联系统可靠性

串联系统

一个设备不可靠整个系统崩溃，可靠性R=R1\*R2\*…\*Rn

并联系统

所有设备都不可靠整个系统才崩溃，可靠性R=1-(1-R1)\*(1-R2)\*…\*(1-Rn)

N模冗余系统

由N个（N=2n+1）相同的子系统和一个表决器组成，表决器把N个子系统中占多数相同结果的输出作为输出系统的输出。该系统中，只要有n+1个或以上子系统能正常工作，系统就能正常工作。（少数服从多数）

性能指标

计算机

时钟频率（主频）；运算速度；运算精度；内存的存储容量；存储器的存取周期；防护具处理速率PRD；吞吐率；各种响应时间；各种利用率；RASIS特性（即可靠性、可用性、可维护性、完成性、安全性）；平均故障响应时间；兼容性；性价比

路由器

设备吞吐量；端口吞吐量；丢包率；时延；时延抖动

网络

设备级性能指标；网络级的性能指标；应用级性能指标；用户级性能指标；吞吐量

操作系统

系统可靠性；系统吞吐量；系统响应时间；系统资源利用率；可移植性

数据管理系统

分为数据库本身和管理系统：数据库大小；数据库中表的数量；单个表的大小；表中允许记录的行数量；单个记录的大小；表上允许的索引数量；数据库所允许的索引数量；最大并发事务处理能力；负责均衡能力；最大连接数

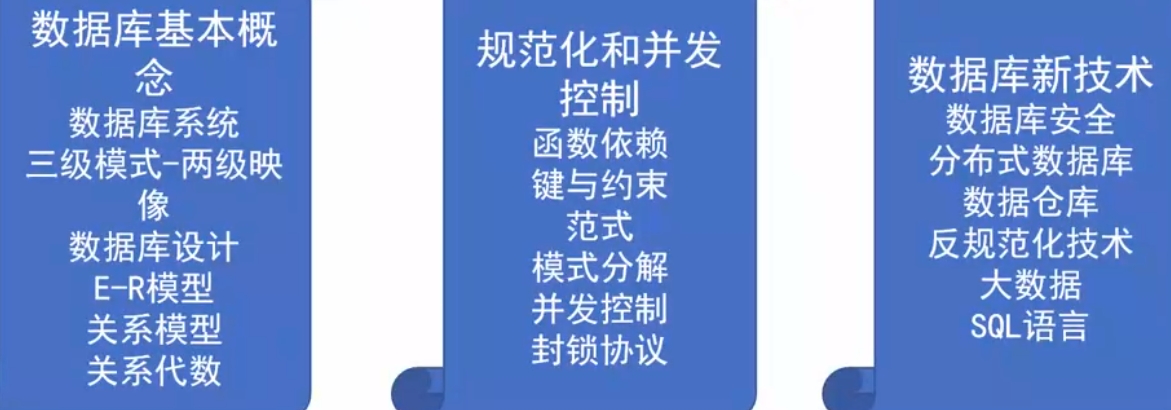
Web服务器

最大并发连接数；相应延迟；吞吐量

数据库系统

历年真题考情：3-5分左右。

第二版更新：主要改动在数据设计，新增NoSQL在案例专题。



数据：

描述失误的符号记录。种类：文本、图形、图像、音频、视频、学生的档案记录等。

数据库DB：

是长期存储在计算机内、有组织、可共享的大量数据的集合。

数据库的基本特征：

数据可按一定的数据模型组织、描述和存储。

可为各种用户共享。

冗余度较小。

数据独立性比较高。

易扩展。

数据库系统DBS

由四部分组成：数据库、硬件、软件、人员（操作数据库的人、DBA等）

数据库管理系统DBMS

实现对共享数据有效的组织、管理和存取。

包括数据定义、数据库操作、数据库运行管理、数据的存储管理、数据库的建立和维护等。

考点一：三级模式—两级映像

内模式

管理如何存储物理的数据，对应具体物理存储文件。

模式

又称概念模式，通常使用的基本表，根据应用、需求将物理数据划分成一张张的表。

外模式

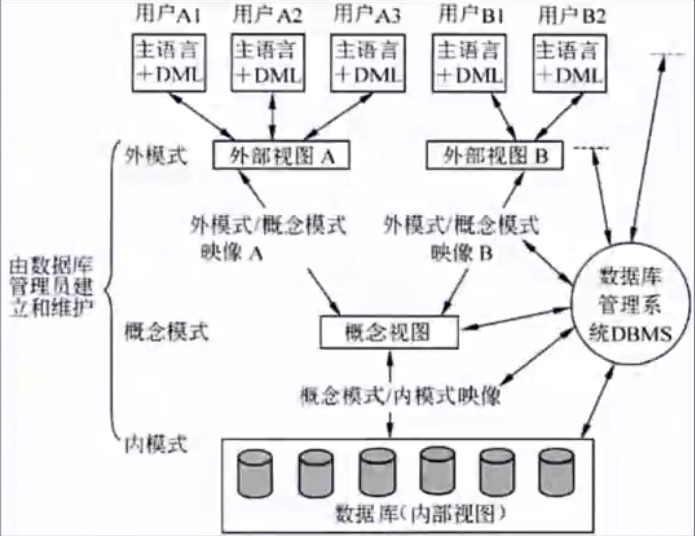
对应数据库中的视图级别，将表进行一定的处理后再提供给用户使用。

外模式—模式映像

是表和视图之间的映射，存在于概念级和外部级之间。若表中数据发生了修改，只需要修改此映射而无需修改应用程序。

模式—内模式映像

是表和数据的物理存储之间的映射，存在于概念级和内部级之间，若修改了数据存储方式，只需要修改此映射，而不需要去修改应用程序



数据库设计

考点：1.步骤。2.产出物。3.内容。

需求分析

分析数据存储的要求，产出物有数据流图、数据字典、需求说明说。获得用户对系统的三个要求：信息要求、处理要求、系统要求。

概念结构设计

设计E-R图，即实体-联系图。工作步骤包括：选择局部应用、逐一设计分E-R图、E-R图合并。E-R图在合并时可能会有3种冲突：

属性冲突：同一属性可能会存在于不同的分E-R图中。

命名冲突：相同意义的属性，在不同的E-R图中有不同的命名，或是名称相同的属性在不同的分E-R图中代表着不同的意义。

结构冲突：统一实体在不同分E-R图中有不同的属性，同一对象在某一分E-R图中被抽象为实体而在另一分E-R图中又被抽象为属性。

逻辑结构设计

将E-R图转换为关系模式（数据表）。工作步骤包括：确定数据模型、将E-R图转换为指定的数据模型、确定完整性约束和确定用户视图。

物理设计

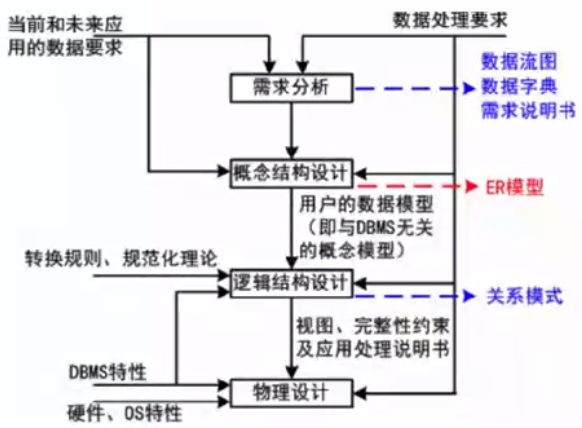
步骤包括：确定数据分布、存储结构和访问方式。

数据库实施阶段

根据逻辑设计和物理设计阶段的结果建立数据库，编制与调试应用程序，组织数据入库，并进行试运行。

数据库运行和维护阶段

数据库应用系统经过试运行即可投入运行，但该阶段需要不断的对系统进行评价、调整与修改。



数据模型

关系模型

是二维表的形式表示的实体-联系模型，是将实体-联系模型转换而来的，经过开发人员设计的；

概念模型（E-R图）

是从用户的角度进行建模的，是现实世界到信息世界的第一抽象，是真正的实体-联系模型。

网状模型

表示实体类型及其实体之间的联系，一个事物和另外几个都有联系，形成一张网。

面向对象模型

是采用面向对象的方法设计数据库，以对象为单位，每个对象包括属性和方法，具有类和继承等特点。

数据模型三要素

数据结构

所研究的对象类型的集合

数据操作

对数据库中各类对象的实例允许执行的操作的集合

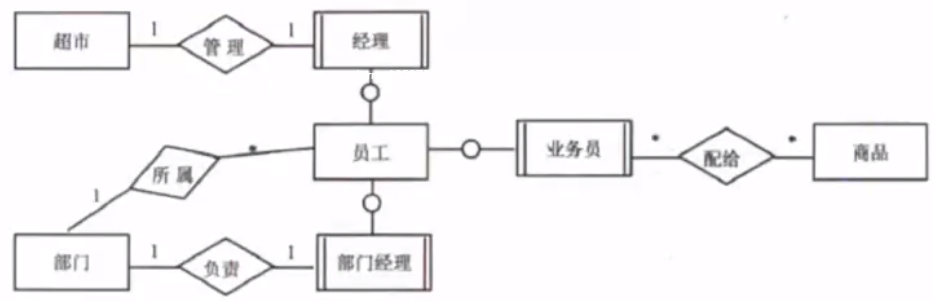
数据的约束条件

一组完整性规则的集合

E-R模型

用E-R图来描述概念数据模型。

在E-R模型中，使用椭圆表示属性（一般没有）、长方形表示实体、菱形表示联系，联系的两端要填写联系类型。



实体：客观存在并可相互区别的事物。可以是具体的人、事、物或抽象概念。如人、汽车、图书、账户、贷款。

弱实体和强实体：弱实体依赖于强实体的存在而存在，上图中“经理”、“部门经理”属于弱实体（长方形内部有两个竖线）。

实体集：具有相同类型和共享相同属性的实体的集合，如学生、课程。

属性：实体所具有的特性。

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。

域：属性的取值范围称为该属性的域。

码（key、主键）：唯一标识实体的属性集。

联系：现实世界中事物内部以及事物之间的联系，在E-R图中反映为实体内部的联系和实体之间的联系。

联系类型：一对一1:1、一对多1：N、多对多M：N。（M、N、\*都能表示多）

关系模型

关系模型中数据的逻辑结构是一张二维表，由行列组成。用表格结构表达实体集，用外键标识实体间的联系。

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。



E-R模型转换为关系模型（原则）

每个实体都对应一个关系模式（必须是强实体）

联系分为三种

1:1联系中，联系可以放到任意的两端实体中，作为一个属性（要保证1：1的两端关联），也可以转换为一个单独的关系模式。

1：N的联系中，联系可以单独作为一个关系模式，也可以在N端加入1端实体的主键。

M：N的联系中，联系必须作为一个单独的关系模式，其主键是M和N端的联合主键。

关系代数