选择题40’

简答题3’\*2=6

操作题（数据库、语句）14’

分析\*4=40’

# 导论：

## 定义

软件=程序+文档+数据

程序=算法+数据结构

算法=操作+控制结构

程序是为实现特定目标或解决特定问题而用计算机语言编写的命令序列的集合

程序:由程序设计语言所描述的、能为计算机所识别、理解和处理的语句序列

程序设计语言具有良好、严格语法和语义

**数据**和**操作**是构成程序的两个基本要素

## 程序语言

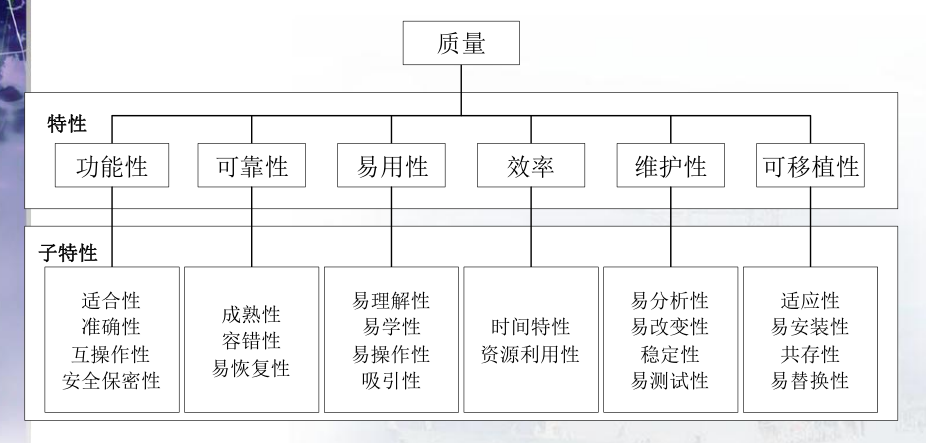
–面向机器:如汇编语言、机器语言等

–面向过程:如Fortran, Pascal, C等等

–面向对象:如C++、Java等等

–面向问题:如结构化查询语言SQL

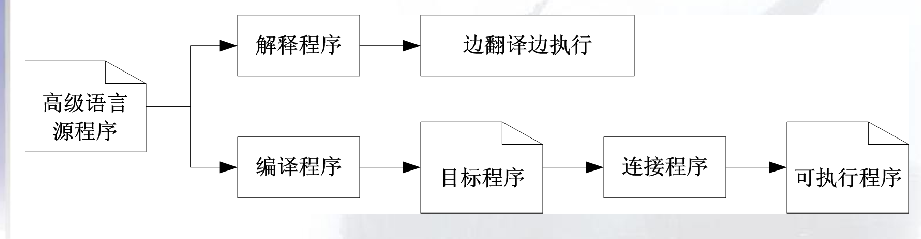
## 软件的质量属性



## 编译、解释

•高级语言程序必须经过翻译变成机器语言程序

•翻译有两种做法：编译和解释，相应的翻译工具分别叫做编译器和解释器



### 编译器工作原理

•第一步：词法分析（Lexical Analysis）

•第二步：语法分析（Syntax Analysis）

•第三步：语义分析（Semantic Analysis）

•第四步：中间代码生成

•第五步：优化

•第六步：代码生成

•连接

### 高级语言程序的解释执行

解释执行需要有一个解释器(Interpreter)，它先作词法分析，建立内部符号表；

•再作语法和语义分析，即以中间码建立语法树，并作类型检查。

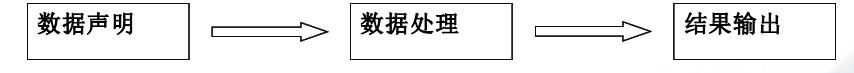
•完成检查后把每一语句压入执行堆栈，压入后立即解释执行

**解释器不大，工作空间也不大、能根据程序执行情况决定下一步做什么（人工智能经常是这样的）是它的优点，**

**•解释执行难于优化、效率较低，这是这类语言的致命缺点**

## Java面向对象VS结构化程序

### 传统的结构化程序设计（Structured Programming---SP）方法



自顶向下、分而治之的方法，将整个程序按功能划分为几个可独立编程的子过程模块，每一子模块完成指定的子任务，

**优点：**

–这种程序设计方法力求算法描述准确。

–对每一子过程模块容易进行程序正确性证明。

**缺点：**

不稳定和多变，不能直接反映人类求解问题的思路。

程序代码可重用性差

维护程序的一致性困难

### 面向对象

#### 特性：

**封装：**隐藏属性、方法或实现细节的过程称为封装

**继承：**子类继承了父类的属性和方法

**多态：**在通过继承而派生出的一系列类中，可能存在一些名称相同，但实现过程和功能不同的方法（Method）。

类是具有相同属性和行为的一组对象的集合

类是概念模型，对象是真实模型

## public、private区别（修饰符）

public：该类或非该类均可访问

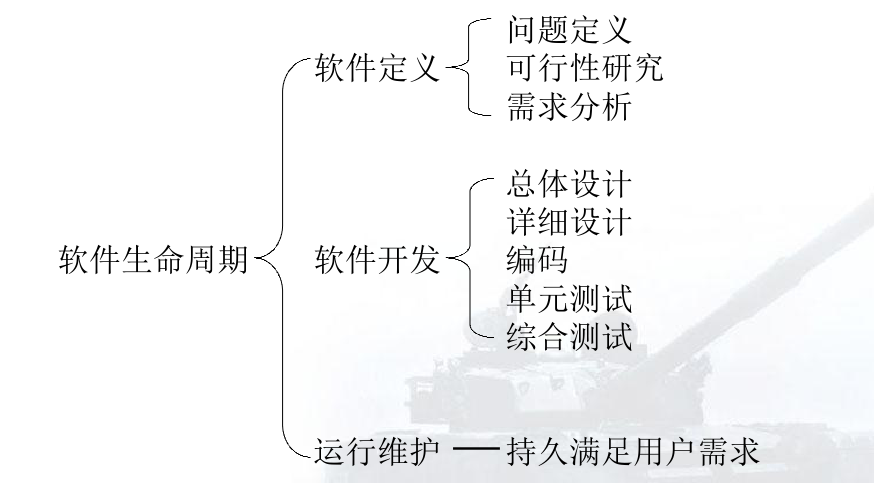
private：只有该类可以访问

# 软件工程：

通常把在软件生命周期全过程中使用的一整套技术方法的集合称为方法学(methodology)，也称为范型(paradigm)。

软件工程方法学包含3个要素：工具、方法和过程。

## 寿命周期：



现在的软件生命周期过程包括：早期：

**立项、需求分析、设计、编码、测试、交付、维护、退役**

又加入了：

**管理各种活动、质量保证、环境基础设施配置、文档管理等。**

## 过程模型特点、优点和不足

### –瀑布模型（Waterfall）

特点：

**文档驱动**的模型

•阶段间具有顺序性和依赖性

•推迟实现的观点

•质量保证的观点

问题：

实际项目很少按照该模型给出的顺序进行

•用户常常难以清楚地给出所有需求

•用户必须有耐心

•开发者常常被不必要地耽搁

### –原型模型（Prototype）

快速建立起来的可以在计算机上运行的程序，他所能完成的功能往往是最终产品能完成的功能的一个子集。

–扔掉原型

–增量原型

–演进原型

原型模型从需求收集开始。开发者和用户在一起定义软件的总体目标，标识出已知的需求，并规划出进一步定义的区域。

•然后是“快速设计”，快速设计集中于软件那些对用户可见部分的表示。“快速设计”

导致原型的建造。

•原型由用户评估，并进一步精化待开发软件的需求，逐步调整原型使其满足客户的要求。

同时开发者对将要做的事情有更好的理解，这个过程是**迭代的**。

原型仍是软件工程的一个有效范型。关键是如何定义一开始的游戏规则，即用户和开发者两方面必须达成一致：**原型被建造仅是为了定义需求，之后就该被抛弃**（或至少部分抛弃），实际的软件在充分考虑了质量和可维护性之后才被开发。

### –螺旋模型（Spiral）

**基本思想**

•使用原型及其他方法来尽量降低风险。

•框架活动：–用户通信——计划——风险分析——做工程——构造与发布——用户评审

**优点：**

–对可选方案和约束条件的强调有利于已有软件的**重用**，也有助于把软件质量作为软件开发的一个重要目标；

–减少了过多测试或测试不足；

–维护和开发之间并没有本质区别。

**特点：**

–**风险驱动的**

•主要适用于开发的大规模软件项目

### –敏捷开发

**瀑布模型的问题：**

基于低变化、高稳定性假设；

•需要在前期就定义完整准确的需求规格书；

•按计划工作，冻结需求，变更是风险；

•缺少用户的持续参与和反馈，低价值需求；

•产品推出周期长，与快速变化的市场不匹配；

•半成品堆积，迟到的集成和测试，后期风险高；

•竖井式文化，沟通浪费、流程等待、能力退化；

•依据流程和文档推动，命令控制式管理，主动和积极性不高……

•将软件开发等同于传统的大规模制造，

•违背了软件项目的本质。

#### •迭代开发，是一种构建软件的方式。

–软件的整个生命周期，一次由几个迭代组成，每次迭代都选取当前最重要的特性，生产出一个集成的、经过测试的子系统，是最终系统的一个子集。

–系统在一次次迭代中，伴随着新特性的出现而逐渐成长，就是增量开发。系统是不断“成长中”，而不是一直在“开发中”

#### 优点

1.频繁获取业务反馈，优先开发重要特性，更好地满足客户需求，提升产品的业务价值；

2.产品发布的早些，只是特性少些，能很好响应市场变化，产品快速上线；

3.允许甚至引发早期的变化，顺应软件项目的高变化本性；

4.利用时间盒原理，开发更加专注，生产力更高；

5.将大系统分解为小型有边界的迷你项目，复杂度和风险降低，更易成功；

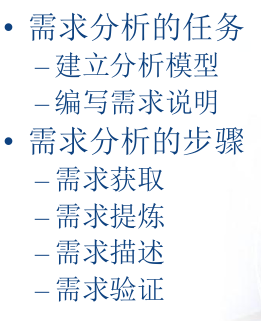
6.尽早和经常性测试，缺陷早发现，尽早和定期进行过程改进，质量提升；

7.团队从反复的成功中获得信心和满足，客户从可视化过程建立与团队的信任。

## 需求分析（环节）

软件的需求分析是开发期的第一个阶段。这个阶段的基本任务是：用户和分析人员双方共同来理解系统的需求，并将共同理解形成一份文件，即软件需求说明书。该阶段是面向用户问题的，它主要是对用户的业务活动进行分析，明确在用户的业务环境中软件系统应该“做什么”。

**一致性、完整性、可理解性、无二义性、可测试性**



### 功能/非功能需求

# 设计：

## 准则

•模块化:一个大的程序按功能或数据原则分解成为较小的模块,可以极大地降低程序的复杂性,有利于阅读、测试、修改

•结构化:每个模块内程序控制应是结构化的,以便于测试、查错和修改。要采用结构化程序设计语言,不用或少用GoTo语句

•数据隐藏：模块化实质也是局部化，凡与其它模块无关的数据尽可能作为局部量放在自己的模块内

•可测试性：程序的可测试性在设计程序结构时就得考虑，以免最后变得不可追踪，不可测试

•一致性：指程序编码完成后，行文风格一致(排列对齐，空行、边框、关键字都出现在同样位置)、表达方法尽量一致



**模块独立性准则**

•内聚性强，标志模块的独立性强；内聚性弱，标志模块的独立性差。

•耦合性强，标志互联的强，模块独立性差；耦合性弱，标志互连的弱，模块独立性强。

## 属性

除了功能、性能要求而外，程序设计还应该保证软件的质量,如:

•可靠性：交付时规格说明均满足，但时好时坏。

•安全性：由于使用不当导致所在系统崩溃。

•可维护性：修改一个小错；引发一串大错。

•可移植性：不同平台上不能用

•适应性：软、硬件平台中某部件升级或版本就不适应了。

•可测试性：除了使用现场其它环境不能作有意义的测试

# 算法和数据结构：

**算法的两要素**

算法由操作与控制结构两要素组成

## 定义：

解题过程的准确、完整的描述称作解该问题的算法

## 优劣

**正确性**

时间复杂性

空间复杂性

可读性和可操作性

## 常见算法及其特点

### 枚举法

特点：**算法简单，运算量大。**

•适用：可确定解的可数取值范围。

•注意：根据具体情况尽量确定出较小的枚举域，避免“不胜枚举”

### 迭代法

逼近

### 递归法

如果一个过程直接或间接地**调用它自身**，则称该过程是递归的

### 递推法

**由结果出发**

所谓递推法，它的数学公式也是递归的。只是在实现计算时与递归相反。从给定边界出发逐步迭代到达指定计算参数。

递推操作是提高递归函数执行效率最有效的方法，科技计算中最常见

### 回溯法

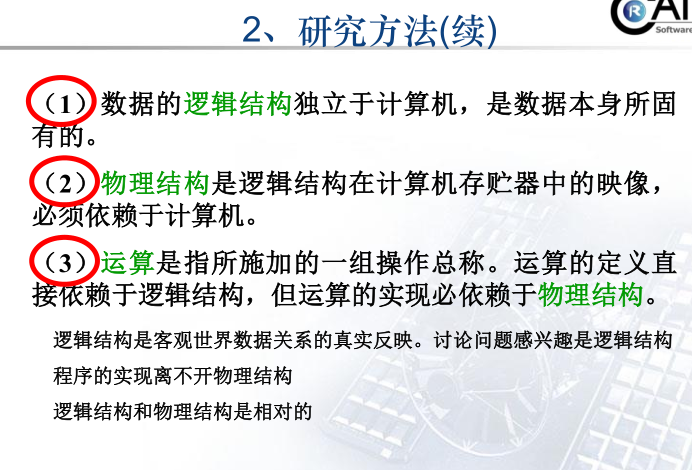
用于：

–寻找一组解

–求满足某些约束条件的最优解

回溯的本质：枚举、试探；尝试失败即返回。

## 数据结构



基础的数据结构分四类

### 1、表：元素是线性关系（连接）

1.顺序表

2.链表

**3.栈（后进先出）**

–栈是限定仅在表尾进行插入和删除运算的线性表，表尾称为栈顶(top)，表头称为栈底(bottom)

–栈的物理存储可以用顺序存储结构，也可以用链式存储结构

**4.队列(先进先出)**

限定所有的插入只能在表的一端进行，而所有的删除都在表的另一端进行

### 2、树：元素间是非线性关系，连接不得有回路

结点子树个数为结点的度,结点度的最大值为该树的度

满二叉树：每一层上都含有最大结点数

完全二叉树：除最后一层外，每一层都取最大结点数，最后一层结点都集中在该层最左边的若干位置。

### 3、图：元素间非线性关系（连接），连接有回路

### 4、文件：记录的序列

## 查找、排序算法（原理）

特点，处理方式，描述表达，性质

### 顺序查找

用待查关键字值与线性表中各结点的关键字值逐个比较，直到找出相等的关键字值；或

找遍所有结点都找不到，即查找失败。

在下面两种情况下只能采取顺序查找

a.线性表为无序表（元素排列是无序的）；

b.即使是有序线性表，但采用的是链式存储结构

### 二分查找

要求线性表结点按关键字码值排好，且以顺序方式存储

前提：必须在具有顺序存储结构的有序表中进行

方法：用要查找的码值X与中间位置结点的关键码值W相比较：

(1)X=W，此时已经查找成功，查找结束

(2)X>W，表明X在表的后半部分，取后半部

分进行查找

(3)X<W，表明X在表的前半部分，取前半部

分进行查找

•二分法查找的优点是平均检索长度小，为log2n。

•二分法查找的缺点是它要求记录必须按关键字有序，并且只适用于顺序方式存储的顺序表。

### 冒泡排序

思想：小的浮起，大的沉底。从左端开始比较。

### 插入排序

**直接插入**：基本思想：从数组的第2号元素开始，顺序从数组中取出元素，并将该元素插入到其左端已排好序的数组的适当位置上。（抽出第J位aj，i从第J-1位开始，若大于则交换，直到小于）

### 选择排序

思想：首先从2~n个元素中选出关键字最小记录交换到第一个位置上。然后再从第3个到第n个元素中选出次小的记录交换到第二个位置上，依次类推。

## 二叉树遍历（操作题）

### 前序遍历二叉树(根左右)

### 中序遍历二叉树（左根右）

### 后序遍历二叉树（左右根）

## 图的遍历

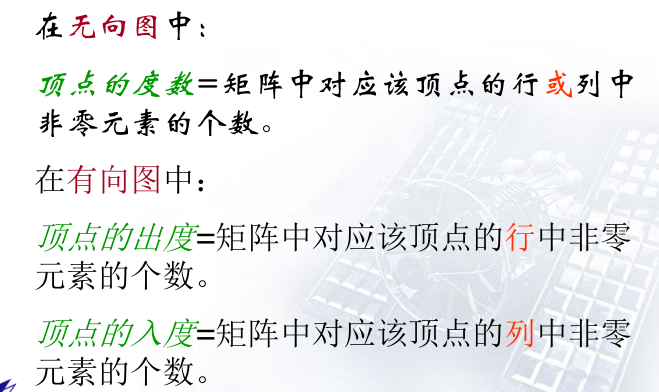
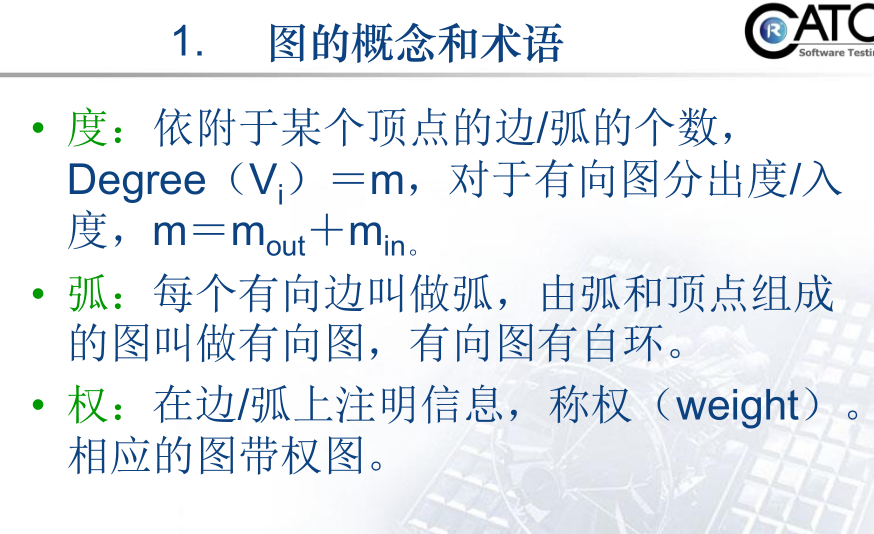
•常用G=(V,E)代表一个图，V是结点的有穷集合(非空)，E是边的有穷集合(E可为空集)。

•若一条边的结点对无序，则称无向图。(V1,V2)和(V2,V1)相同，有向图由顶点的非空有限集和边的有限集组成。

(V1,V2)和(V2,V1)表示不同边(见下页)

•n个顶点的无向图边的最大数目是n(n-1)/2。

•n个顶点的有向图边的最大数目为n2（双环且自环）



### 深度优先搜索遍历

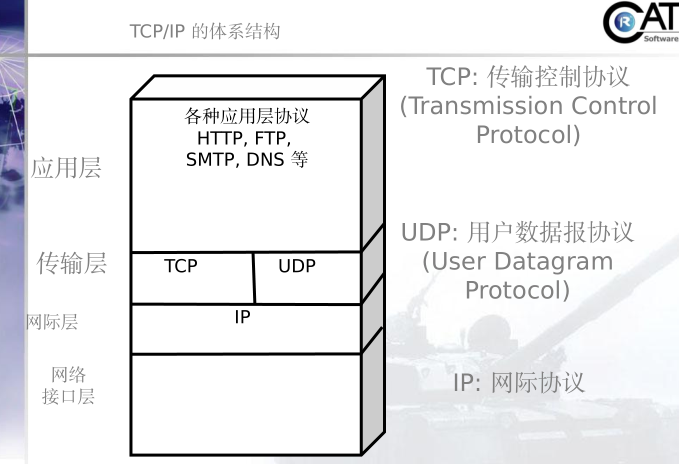
搜索与顶点i有边相连的下一个顶点j，若j未被访问过，则访问它，并将j的访问标记置为访问过，visited[j]=1，然后从j开始重复此过程，若j已访问，再看与i有边相连的其它顶点；

### 广度优先搜索遍历

依次访问与顶点i有边相连的所有顶点W1，W2，…，Wt；

# 网络：

## TCP/IP协议



### IP协议

•因特网的IP协议最重要，它**为分组在互连网中的发送、传输和接收制定了详尽的规则。**

•IP是Internet Protocol的缩写，中文标准译名是**“网际协议”**

**子网与子网掩码**

•将两个二进制数逻辑与（AND）运算后得出的结果即为网络部分

•将子网掩码取反再与IP地址逻辑与（AND）后得到的结果即为主机部分

**IP网络的重要特点**

•每一个分组独立选择路由。

•发往同一个目的地的分组，后发送的有可能先收到

（即可能不按顺序接收）。

•当网络中的通信量过大时，路由器就来不及处理分

组，于是要丢弃一些分组。

•因此，IP网络不保证分组的可靠地交付。

•IP网络提供的服务被称为：尽最大努力服务(best effort service)

### TCP协议

•因特网认为，IP协议没有必要提供可靠服务（这样做可以使网络简单、灵活性好、价格便宜）。

在计算机中增加TCP协议就可以实现计算机程序之间的可靠通信。

•TCP是Transmission Control Protocol的缩写，中文译名是**传输控制协议**。

•TCP协议驻留在用户计算机中，它的作用是**保证应用程序之间端到端的可靠通信**。

•TCP给要传送的每一个字节的数据都进行编号。

–接收端在收到数据后必须向发送端发送确认信息。

–若发送端在规定的时间内没有收到对方的确认，就重传这部分数据。

•当网络中的通信量过大时，TCP就告诉发送端要放慢发送数据。这叫做流量控制。

## 总结

•TCP协议保证计算机程序之间的、端到端的可靠交付。

•在TCP/IP的应用层协议使用的是**客户服务器方式**。

•客户和服务器都是进程（即软件）。客户是服务请求方，服务器是服务提供方。

•服务器有时也指“运行服务器软件”的机器。

# 数据库：

## 定义：

### 1.数据库

-数据库是相互关联的数据的集合，是对数据的科学组织与存储，以便对数据进行高效的查找与操作。

### 2．数据库管理系统（DBMS）

全称是Data Base Management System。

DBMS是一个进行数据库管理的软件集合，它对数据库进行统一的管理和控制，以保证数据库的安全性和完整性。

主要具备如下功能：

（1）数据库定义功能

对各级数据模式进行精确的描述

（2）数据操纵功能

对数据库中的数据进行追加、插入、修改、删除、检索等操作。

（3）系统运行控制功能

包括并发控制、安全性检查、完整性约束条件的检查和执行、数据库的内部维护等.

（4）系统维护功能

包括数据库初始数据的输入、转换功能，数据库的转储、恢复功能，数据库的重新组织功能和性能监视、分析功能等。

### 3．数据库系统

数据库系统是指拥有数据库技术支持的计算机系统，它可以实现有组织地、动态地存储大量相关数据，提供数据处理和信息资源共享服务。

### 4.数据库三类使用者

1、最终用户

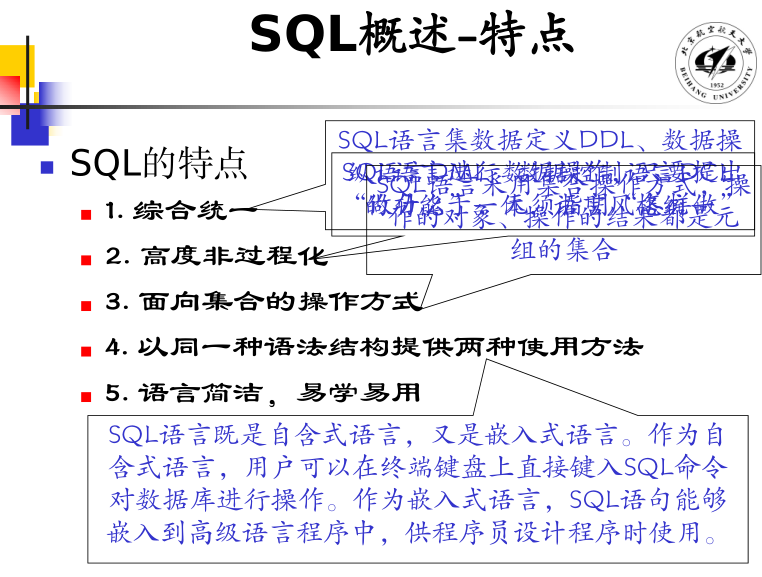
2、数据库应用系统开发人员

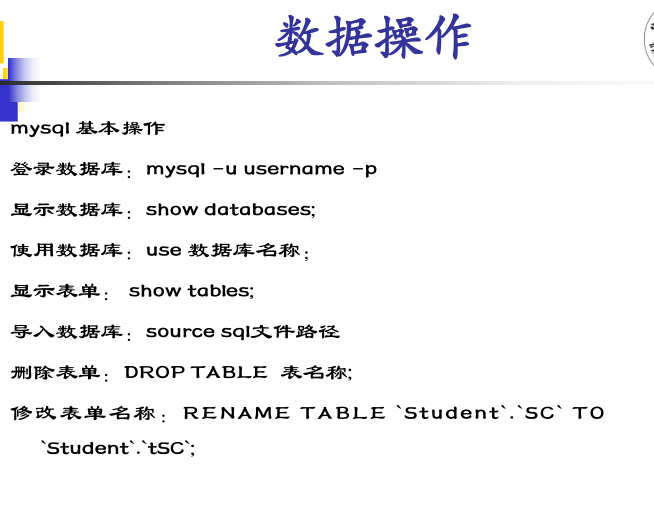
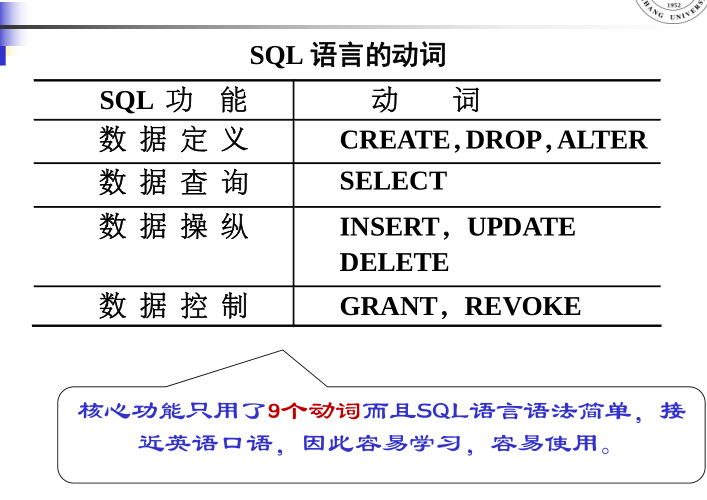
3、数据库管理员DBA

### 5、数据库系统的三级模式结构和二级映像

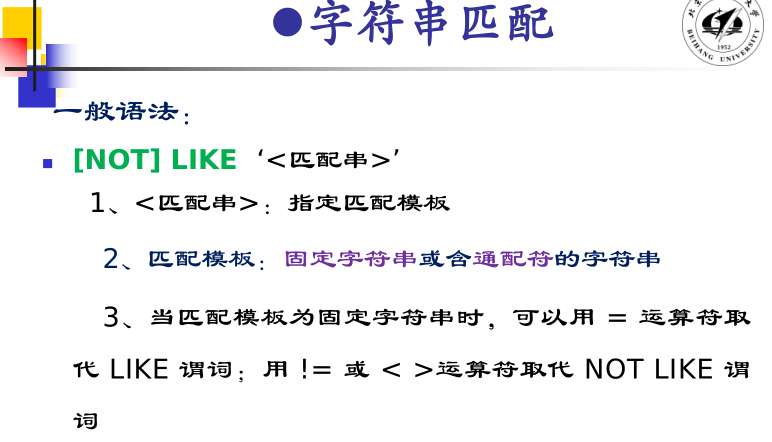
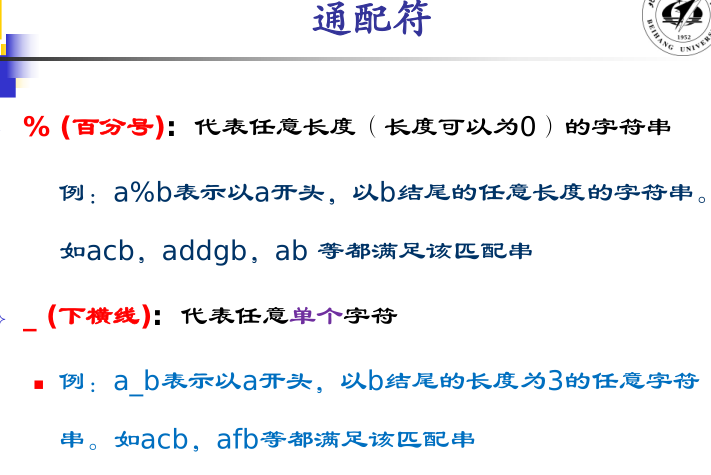
外模式、概念模式和内模式等，以及三种模式之间的二级映像。

## SQL数据库





### 有条件查询、统计



## ER图表达

实体型：矩形

属性：椭圆形

联系本身：菱形（1:1，1：n，n：m）

联系的属性：椭圆形

## 数据模型

分类：层次模型、网络模型、关系模型

数据模型是严格定义的一组概念的集合，这些概念精确地描述系统的静态特性、动态特性和完整性约束条件。

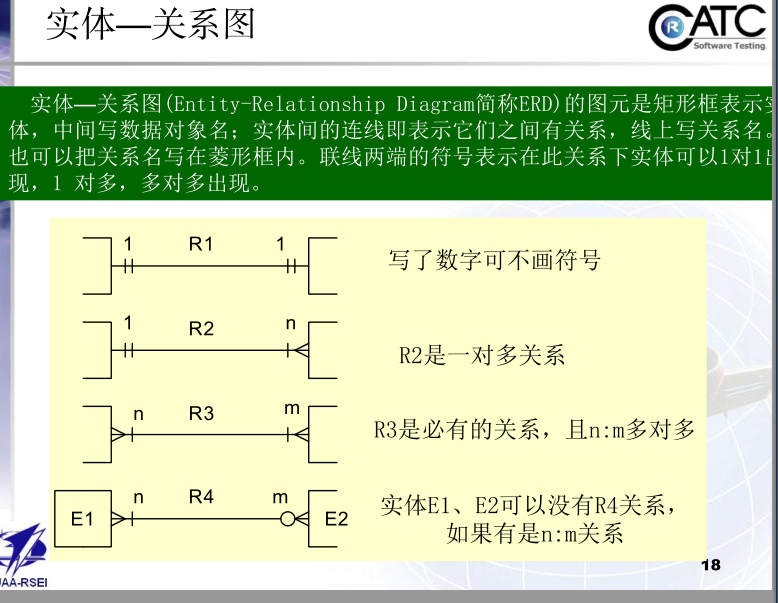
数据结构、数据操作、完整性约束

实体完整性：主键不为空

参照完整性：主键和外码的引用规则

用户定义的完整性：

## 关系图



## 表单（范式）

第一范式：每个属性值都是不可再分的最小数据单位

第二范式：消除部分函数依赖的关系，使全部外码都是完全依赖主键

第三范式：消除传递函数依赖的关系，删除依赖外码的属性