1. 用过的数据库
2. MySQL SQL SERVER mangoDB
3. 数据库管理系统有哪些，用过什么

MySQL SQL SERVER mangoDB

1. 关系模型为什么分解

一个关系模式分解后，可以存放原来所不能存放的信息，通常称为“悬挂”的元组，这是实际所需要的，正是分解的优点。

1. E-R图：

实体-联系图

提供了表示实体类型、属性和联系的方法，用来描述现实世界的概念模型。它是描述现实世界关系概念模型的有效方法。是表示概念关系模型的一种方式。用“矩形框”表示实体型，矩形框内写明实体名称；用“椭圆图框”表示实体的属性，并用“实心线段”将其与相应关系的“实体型”连接起来；

用”菱形框“表示实体型之间的联系成因，在菱形框内写明联系名，并用”实心线段“分别与有关实体型连接起来，同时在”实心线段“旁标上联系的类型（1:1,1:n或m:n）。

1. 数据库安全

* 数据的安全性：

**保护数据库防止恶意的破坏和非法的存取**

防范对象：非法用户和非法操作

**系统运行安全和系统信息安全**

数据库安全性控制的常用方法：

-用户标识和鉴定

-存取控制

-视图（视图机制把要保密的数据对无权存取这些数据的用户隐藏起来，对数据提供一定程度的安全保护）

-审计

-密码存储

* 数据的完整性

防止数据库中存在不符合语义的数据，也就是防止数据库中存在不正确的数据

防范对象：不合语义的、不正确的数据

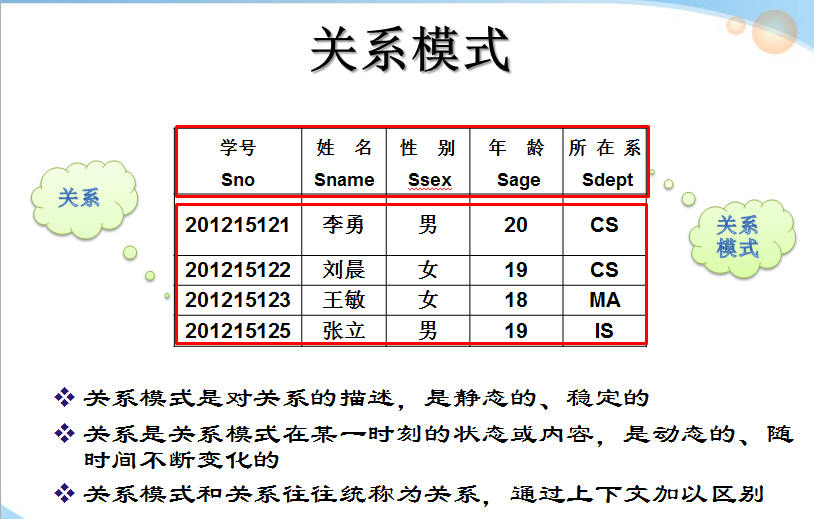
为维护数据库的完整性，DBMS必须：

* + 提供定义完整性约束条件的机制
  + 提供完整性检查的方法
  + 违约处理

1. 关系模型和关系模式的联系

关系模型。关系模型主要是用二维表格结构表达实体集，用外键表示实体间联系。;关系模型是由若干个关系模式组成的集合。

关系模式相当于记录类型。



1. 关系模型对关系的限制/关系模型的规范限制

关系模式是对关系的描述（有哪些属性，各个属性之间的依赖关系如何）

在关系模型中，对关系作了下列规范性限制：

(1)关系中每一个属性值都是不可分解的。

(2)关系中不允许出现相同的元组(没有重复元组)。

(3)由于关系是一个集合，因此不考虑元组间的顺序，即没有行序。

(4)元组中，属性在理论上也是无序的，但在使用时按习惯考虑列的顺序。

1. 元组为什么没有顺序，但是元组不能相同？

因为关系是一个元组的集合,而元组在集合中的顺序无关紧要。因此不考虑元组间的顺序,即没有行序。

1. 数据库通常用什么模型
2. 概念模型:按照用户的观点，对数据和信息进行建模。
3. 逻辑模型：按照计算机系统的特点，对数据库管理系统的实现
4. 关系模型：用二维表的方式，表示实体与实体之间的关系。

2016

1.DBMS的定义及组成

**数据（data) ：**是数据库中存储的基本对象

**数据库(Database，简称DB)：**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。

**数据库管理系统(DBMS)：**位于用户与操作系统之间的一层数据管理软件。是基础软件，是一个大型复杂的软件系统。如：SQL SERVER MySQL

DBMS是由两大部分组成：查询处理器和存储管理器。（1） 查询处理器有四个主要成分：DDL编译器、DML编译器、嵌入型DML的预编译器、查询运行核心程序。

（2） 存储管理器有四个主要成分：授权和完整性管理器、事务管理器、文件管理器、缓冲区管理器。

**数据库系统(DBS)：**在计算机系统中引入数据库后的系统。

2.简述数据模型：

在数据库中用数据模型这个工具来抽象、表示和处理现实世界中的数据和信息。通俗地讲数据模型就是现实世界的模拟。

数据模型应满足三方面要求：①能比较真实地模拟现实世界 ②容易为人所理解 ③便于在计算机上实现

数据模型分为两类：

* 概念模型
* 逻辑模型和物理模型 

逻辑模型主要包括网状模型、层次模型、关系模型、面向对象模型等，按计算机系统的观点对数据建模，用于DBMS实现。

物理模型是对数据最底层的抽象，描述数据在系统内部的表示方式和存取方法，在磁盘或磁带上的存储方式和存取方法。

什么是数据模型?

3.概念模型：也称信息模型，它是按用户的观点来对数据和信息建模，用于数据库设计。

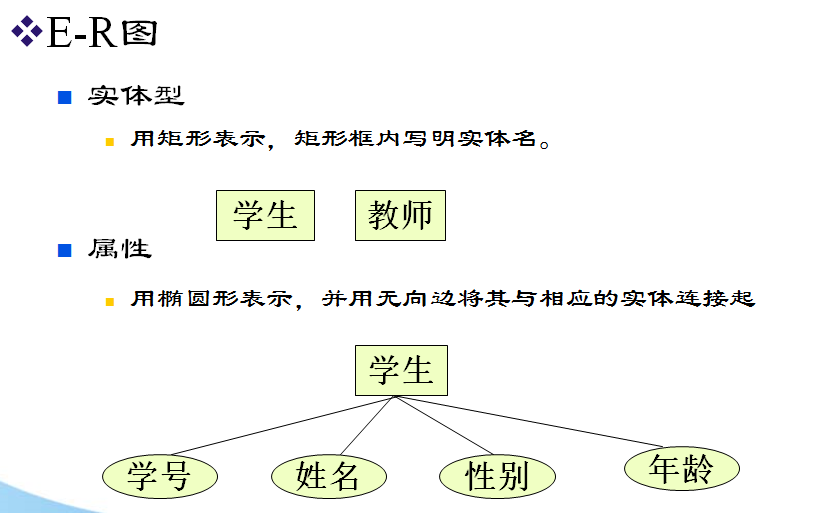
4.关系模型：用二维表的形式表示实体和实体间联系的数据模型。（行：元组 列：属性 一个关系对应通常说的一张表）

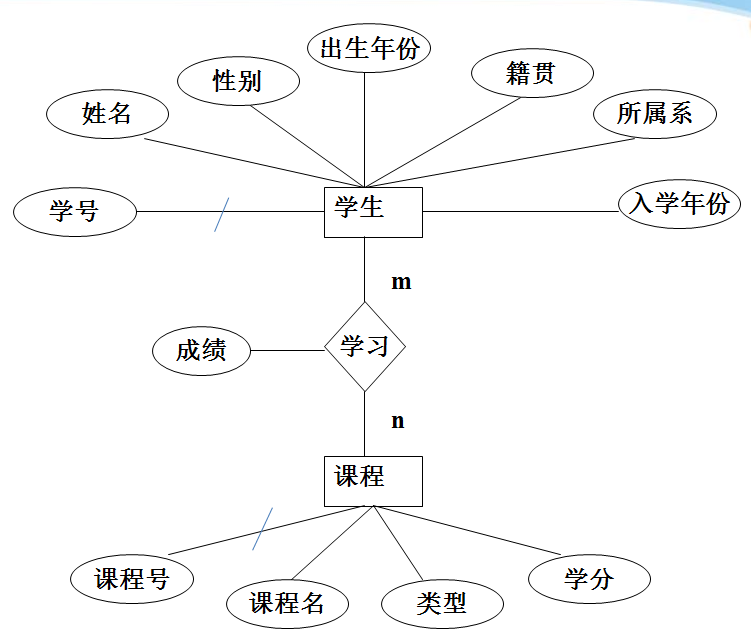
5.网状模型和关系模型的定义

层次数据模型是用树状<层次>结构来组织数据的数据模型。

用有向图表示实体和实体之间的联系的数据结构模型称为网状数据模型。

6.E-R图的相关内容,会判断它们之间的联系





6.数据库建立分为哪几个阶段

* 数据库设计分6个阶段
  + 需求分析：分析用户的需求，包括数据、功能和性能需求；
  + 概念结构设计：主要采用E-R模型进行设计，包括画E-R图；
  + 逻辑结构设计：通过将E-R图转换成表，实现从E-R模型到关系模型的转换；
  + 物理结构设计：主要是为所设计的数据库选择合适的存储结构和存取路径；
  + 数据库实施：包括编程、测试和试运行；
  + 数据库运行和维护：系统的运行与数据库的日常维护。

7.外编码在什么情况下为空

实体完整性规则是指若属性A是基本关系R的主属性，则属性A不能取空值。

若属性(或属性组)F是基本关系R的外码，它与基本关系S的主码Ks相对应(基本关系R和S不一定是不同的关系)，则对于R中每个元组在F上的值必须为：或者取空值(F的每个属性值均为空值)；或者等于S中某个元组的主码值。即属性F本身不是主属性，则可以取空值，否则不能取空值。

8.状态图定义UML

9．**E-R图**向**关系模型**的转换：

将实体、实体的属性和实体之间的联系转换为关系模式。

转换原则

* ⒈ 一个实体型转换为一个关系模式。
  + 关系的属性：实体型的属性
  + 关系的码：实体型的码

如关系模式：

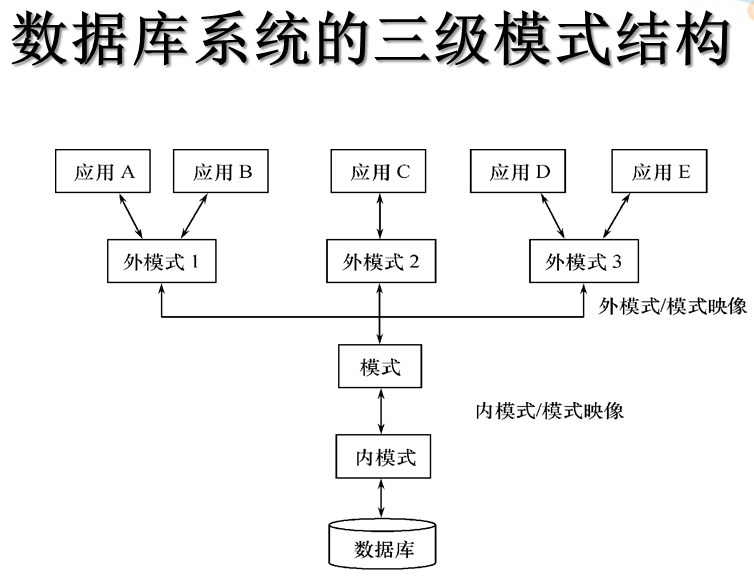
学生（学号，姓名，性别，专业，出生日期）

课程（课程编号，课程名称，课程类别，学分）

* 2.一个m:n联系转换为一个关系模式。选课（学号，课程号，成绩）
* 3.一个1:n联系可以转换为一个独立的关系模式，也可以与n端对应的关系模式合并。
* 4.一个1:1联系可以转换为一个独立的关系模式，也可以与任意一端对应的关系模式合并。
* 5.具有相同码的关系模式可合并

10．外模式和内模式

数据库系统的三级模式结构：外模式 模式 内模式



* 模式：（也称逻辑模式）一个数据库只有一个模式。模式是数据库中全体数据的逻辑结构和特征的描述。是数据库系统模式结构的中间层。

模式的定义

1. 数据的逻辑结构（数据项的名字、类型、取值范围等）
2. 数据之间的联系
3. 数据有关的安全性、完整性要求

例如：在学生选课数据库模式中，包含学生记录、课程记录和学生选课记录

* 外模式：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。

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
    - 对模式中同一数据，在外模式中的结构、类型、长度、保密级别等都可以不同
* 外模式与应用的关系：一对多
  + - 同一外模式也可以为某一用户的多个应用系统所使用
    - 但一个应用程序只能使用一个外模式
* 内模式（也称存储模式）：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

11．物理独立性和逻辑独立性

数据库的二级映像功能与数据独立性：

* + 外模式／模式映像-----保证数据的逻辑独立性
* 当模式改变时，数据库管理员修改有关的外模式／模式映象，使外模式保持不变
* 应用程序是依据数据的外模式编写的，从而应用程序不必修改，保证了数据与程序的逻辑独立性，简称数据的逻辑独立性。

模式：描述的是数据的全局逻辑结构 外模式：描述的是数据的局部逻辑结构

* + 模式／内模式映像：

模式／内模式映象定义了数据全局逻辑结构与存储结构之间的对应关系。例如，说明逻辑记录和字段在内部是如何表示的

数据库中模式／内模式映象是唯一的

模式／内模式映像-----保证数据的物理独立性

* + 当数据库的存储结构改变了（例如选用了另一种存储结构），数据库管理员修改模式／内模式映象，使模式保持不变
  + 应用程序不受影响。保证了数据与程序的物理独立性，简称数据的物理独立性。

12．元组为什么没有顺序

因为关系是一个元组的集合,而元组在集合中的顺序无关紧要。因此不考虑元组间的顺序,即没有行序。

13．数据库恢复

* **数据库管理系统对故障的对策**
  + **DBMS提供恢复子系统**
  + **保证故障发生后，能把数据库从错误状态恢复到某一已知的正确状态，这就是数据库的恢复**
  + **保证事务ACID**

1. 数据库并发控制：

**并发控制就是要用正确的方式调度并发操作，使一个用户事务的执行不受其他事务的干扰，从而避免造成数据的不一致性**

-并发控制的主要技术：

封锁(Locking)：操作之前，先向系统发出请求，对数据对象加锁

时间戳(Timestamp)

乐观控制法

---

并发操作带来的数据不一致性：

-丢失修改（Lost Update）

-不可重复读（Non-repeatable Read）

-读“脏”数据（Dirty Read）：不正确的数据

--------补充--------

1触发器（Trigger）是用户定义在关系表上的一类由事件驱动的特殊过程

--由服务器自动激活

--可以进行更为复杂的检查和操作，具有更精细和更强大的数据控制能力

2数据依赖

一个关系内部属性与属性之间的约束关系

现实世界属性间相互联系的抽象

数据内在的性质

语义的体现

3范式是符合某一种级别的关系模式的集合

一个低一级范式的**关系模式**，通过**模式分解**可以转换为若干个高一级范式的关系模式的集合，这种过程就叫**规范化**

4“好”的关系模式：

不会发生插入异常、删除异常、更新异常，

数据冗余应尽可能少

5. **事务**

* + 一个数据库操作序列
  + 一个不可分割的工作单位
  + 恢复和并发控制的基本单位

事务的ACID特性：

原子性（Atomicity）

一致性（Consistency）

隔离性（Isolation）

持续性（Durability ）

6. **故障种类：**

* 事务内部的故障：如 银行转账事务，这个事务把一笔金额从一个账户甲转给另一个账户乙。
* 系统故障: 称为软故障，是指造成系统停止运转的任何事件，使得系统要重新启动。
* 介质故障: 称为硬故障，指外存故障。如：磁盘损坏
* 计算机病毒：

7. **恢复实现技术**

恢复操作的基本原理：冗余

利用存储在系统其它地方的冗余数据来重建数据库中已被破坏或不正确的那部分数据

* 恢复机制涉及的关键问题
  1. 如何建立冗余数据
     + 数据转储（backup）
     + 登录日志文件（logging）
  2. 如何利用这些冗余数据实施数据库恢复

8. 两段封锁协议(Two-Phase Locking，简称2PL)是最常用的一种封锁协议，理论上证明使用两段封锁协议产生的是可串行化调度。

指所有事务必须分两个阶段对数据项加锁和解锁

第一阶段是获得封锁，也称为扩展阶段。事务可以申请获得任何数据项上的任何类型的锁，但是不能释放任何锁。

第二阶段是释放封锁，也称为收缩阶段。事务可以释放任何数据项上的任何类型的锁，但是不能再申请任何锁。

--------------------------------------

1. PAD定义

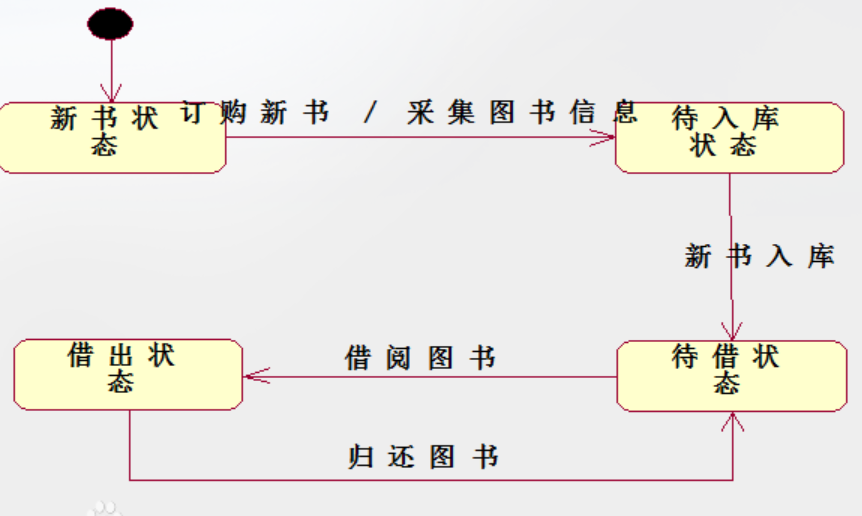
Pad-问题分析图

用于描述**软件详细设计**的图形表示工具。它用二维树形结构的图表示程序的控制流。

用PAD图表现程序逻辑，易读、易懂、易记。PAD图是**二维树型结构的图形**，程序从图中最左边上端的结点开始执行，自上而下，从左到右顺序执行；很容易将PAD图转换成程序代码。

1. 状态图UML

状态图(Statechart Diagram)是描述一个实体基于事件反应的动态行为，显示了该实体如何根据当前所处的状态对不同的事件做出反应。通常我们创建一个UML状态图是为了以下的研究目的：研究类、角色、子系统、或组件的复杂行为。



1. 软件配置的作用

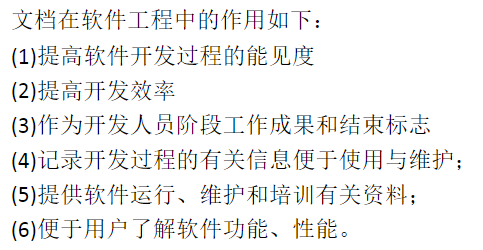
**软件配置管理**应用于整个软件工程过程。SCM活动的目标就是为了**标识变更、控制变更、确保变更正确实现并向其他有关人员报告变更。**从某种角度讲，SCM是一种标识、组织和控制修改的技术，**目的是使错误降为最小并最有效地提高生产效率。**

1. 什么叫软件工程

软件工程是：① 把系统的、规范的、可度量的途径应用于软件开发、运行和维护过程，也就是把工程应用于软件；② 研究①中提到的途径。

//////采用工程化的概念、原理、技术和方法**进行软件的开发和维护**，把经实践证明的科学的**管理措施**与最先进的**技术方法**结合起来。软件工程研究的目标是“以较少的投资获取高质量的软件”。

1. 文档在软件工程作用



1. 瀑布模型的优缺点: (需求分析 需求定义 概要设计 详细设计 实现 系统测试 验收测试 维护)

瀑布模型 优点：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。虽然有不少缺陷但比在软件开发中随意的状态要好得多。

缺点：(1) 实际的项目大部分情况难以按照该模型给出的顺序进行，而且这种模型的迭代是间接的，这很容易由微小的变化而造成大的混乱。

(2) 经常情况下客户难以表达真正的需求，而这种模型却要求如此，这种模型是不欢迎具有二义性问题存在的。

(3) 客户要等到开发周期的晚期才能看到程序运行的测试版本，而在这时发现大的错误时，可能引起客户的惊慌，而后果也可能是灾难性的。

1. 详细设计的基本任务:

(1)为每个模块进行**详细的算法设计**。用某种图形、表格、语言等工具将每个模块处理过程的详细算法描述出来。

    (2)为模块内的**数据结构进行设计**。对于需求分析、概要设计确定的概念性的数据类型进行确切的定义。

(3)**对数据结构进行物理设计**，**即确定数据库的物理结构**。物理结构主要指数据库的存储记录格式、存储记录安排和存储方法，这些都依赖于具体所使用的数据库系统。

----------

概要设计就是**设计软件的结构**，包括组成模块，模块的层次结构，模块的调用关系，每个模块的功能等等。同时，还要设计该项目的应用系统的总体数据结构和数据库结构，即应用系统要存储什么数据，这些数据是什么样的结构，它们之间有什么关系。

详细设计阶段就是为每个模块完成的功能进行具体的描述，要把功能描述转变为精确的、结构化的过程描述。

概要设计阶段通常得到软件结构图

详细设计阶段常用的描述方式有：流程图、N-S图、PAD图、伪代码等

1. 详细设计的步骤:

1.根据模块实现上层类

2.根据用例图实现具体子类

3.结合模块和用例图，实现具体类的属性和方法

4.根据接口划分模型(将抽象的传输转化为类)

5.根据模型写出model类

6.根据已经明确的各个类和数据库，画出类时序图

7.根据时序图和接口画出类数据流程图[如果有状态图和活动图，也需要参考]

1. 质量检测小组的作用:
2. 增量模型的优缺点

**快速原型模型**

优点：使用户能够感受到实际的系统，使开发者能够快速地构造出系统的框架。

缺点：产品的先天性不足，因为开发者常常需要做实现上的折中，可能采用不合适的操作系统或程序设计语言，以使原型能够尽快工作。

**增量模型**

优点：(1) 人员分配灵活，刚开始不用投入大量人力资源，当核心产品很受欢迎时，可增加人力实现下一个增量。

(2) 当配备的人员不能在设定的期限内完成产品时，它提供了一种先推出核心产品的途径，这样就可以先发布部分功能给客户，对客户起到镇静剂的作用。

缺点：(1) 至始至终开发者和客户纠缠在一起，直到完全版本出来。

(2) 适合于软件需求不明确、设计方案有一定风险的软件项目。

该模型具有一定的市场。

**螺旋模型**

优点：**对于大型系统及软件的开发**，这种模型是一个很好的方法。开发者和客户能够较好地对待和理解每一个演化级别上的风险。

缺点：(1) 需要相当的风险分析评估的专门技术，且成功依赖于这种技术。

(2) 很明显一个大的没有被发现的风险问题，将会导致问题的发生，可能导致演化 的方法失去控制。

(3) 这种模型相对比较新，应用不广泛，其功效需要进一步的验证。

该模型适合于大型软件的开发

1. 什么是静态测试

就是**不实际运行被测软件**，而只是静态地检查程序代码、界面或文档中可能存在的错误的过程。

包括对代码测试、界面测试和文档测试三个方面：

    对于代码测试，主要测试代码是否符合相应的标准和规范。

    对于界面测试，主要测试软件的实际界面与需求中的说明是否相符。

对于文档测试，主要测试用户手册和需求说明是否符合用户的实际需求。

--------

  动态测试（dynamic testing），指的是**实际运行被测程序**，输入相应的测试数据，检查实际输出结果和预期结果是否相符的过程，所以判断一个测试属于动态测试还是静态的，**唯一的标准就是看是否运行程序。**

    黑盒测试有可能是动态测试（运行程序，看输入输出），也有可能是静态测试（不运行，只看界面）

    白盒测试有可能是动态测试（运行程序并分析代码结构），也有可能是静态测试（不运行程序，只静态察看代码）

    动态测试有可能是黑盒测试（运行，只看输入输出），也有可能是白盒测试 （运行并分析代码结构）

静态测试有可能是黑盒测试（不运行，只察看界面），也有可能是白盒测试（不运行，只察看代码）

1. 黑盒、白盒

**白盒测试：是通过程序的源代码进行测试**而不使用用户界面。这种类型的测试需要从代码句法发现内部代码在算法，溢出，路径，条件等等中的缺点或者错误，进而加以修正。

**黑盒测试：又被称为功能测试**、数据驱动测试或基于规格说明的测试，**是通过使用整个软件或某种软件功能来严格地测试,** 而并没有通过检查程序的源代码或者很清楚地了解该软件的源代码程序具体是怎样设计的。

1. 数据字典：
2. 详细设计的步骤及基本任务
3. PAD定义
4. 快速原形模型定义
5. 软件质量定义：

概括地说，软件质量就是"软件与明确的和隐含的定义的需求相一致的程度"。具体地说，软件质量是软件符合明确叙述的功能和性能需求、文档中明确描述的开发标准、以及所有专业开发的软件都应具有的隐含特征的程度。

-----

软件可靠性：

软件可靠性是软件产品在规定的条件下和规定的时间区间完成规定功能的能力。软件可靠性是关于软件能够满足需求功能的性质，软件不能满足需求是因为软件中的差错引起了软件故障。软件可靠性直接影响着软件质量，如果一个软件的可靠性不高的话，那么说明这个软件产品的质量是有问题的。只有软件可靠性足够高，才能让人家质量更好。

1. 文档在软件中的作用
2. 单元测试的内容：**是指对软件中的最小可测试单元进行检查和验证。**对于单元测试中单元的含义，一般来说，要根据实际情况去判定其具体含义，如C语言中单元指一个函数，图形化的软件中可以指一个窗口或一个菜单等。总的来说，单元就是人为规定的最小的被测功能模块。
3. 阿拉法和贝塔测试定义与区别：

α测试

       α测试是用户在开发环境下的测试，或者是开发内部的用户在模拟实际环境下的测试；

       Alpha 测试(α测试)是由一个用户**在开发环境下进行的测试**，也可以是公司内部的用户在**模拟实际操作环境下进行的受控测试**，Alpha测试不能由程序员或测试员完成。Alpha测试发现的错误，可以在测试现场立刻反馈给开发人员，由开发人员及时分析和处理。目的是评价软件产品的功能、可使用性、可靠性、性能和支持。尤其注重产品的界面和特色。Alpha测试可以从软件产品编码结束之后开始，或在模块(子系统)测试完成后开始，也可以在确认测试过程中产品达到一定的稳定和可靠程度之后再开始。有关的手册(草稿)等应该在Alpha测试前准备好。

β测试

       Beta测试是由软件的一个或多个用户**在实际使用环境下进行的测试**；

       Beta测试(β测试)是软件的多个用户在一个或多个用户的实际使用环境下进行的测试。开发者通常不在测试现场，Beta测试不能由程序员或测试员完成。因而，Beta测试是在开发者无法控制的环境下进行的软件现场应用。在Beta测试中，由用户记下遇到的所有问题，包括真实的以及主管认定的，定期向开发者报告，开发者在综合用户的报告后，做出修改，最后将软件产品交付给全体用户使用。Beta测试着重于产品的支持性，包括文档、客户培训和支持产品的生产能力。只有当Alpha测试达到一定的可靠程度后，才能开始Beta测试。由于Beta测试的主要目标是测试可支持性，所以Beta测试应该尽可能由主持产品发行的人员来管理。

两者区别

       Alpha测试由开发人员或测试人员在场，可随时记录下错误和使用中出现的问题；

       Beta测试开发人员和测试人员都不在场。

1. 集成和系统测试的区别：

**集成测试**在系统测试之前，单元测试完成之后系统集成的时候进行测试。集成测试主要是针对程序内部结构进行测试，特别是对程序之间的接口进行测试。集成测试对测试人员的编写脚本能力要求比较高。测试方法一般选用黑盒测试和白盒测试相结合。

**系统测试**最主要的就是功能测试，测试软件《需求规格说明书》中提到的功能是否有遗漏，是否正确的实现。做系统测试要严格按照《需求规格说明书》，以它为标准。测试方法一般都使用--黑盒测试..

（单元测试-集成测试-系统测试-验收测试）

1. 软件生命周期：是软件的产生直到报废或停止使用的生命周期。

**软件生命周期模型**是跨越整个生存期的系统开发、运作和维护所实施的全部过程、活动和任务的**结构框架**。

1. 软件工程的**基本特征：**

(1) 软件工程**关注于大型程序(软件系统)的构造**

(2) 软件工程的中心课题是**分解问题，**控制复杂性

(3) 软件是经常变化的，开发过程中必须考虑软件将来可能的变化

(4) 开发软件的效率非常重要，因此，软件工程的一个重要课题就是，**寻求开发与维护软件的更好更有效的方法和工具**

(5) **和谐地合作**是开发软件的关键

(6) **软件必须有效地支持它的用户**

1. 系统模型图和数据流图的定义：

**数据流图：它以图形的方式描绘数据在系统中流动和处理的过程。（**数据流图不是传统的流程图或框图，数据流也不是控制流。数据流图是从数据的角度来描述一个系统，而框图是从对数据进行加工的工作人员的角度来描述系统。**）**显示系统将输入和输出什么样的信息，数据如何通过系统前进以及数据将被存储在何处。

1. 软件配置管理的作用
2. 软件危机定义：软件危机是指在计算机软件开发、使用与维护过程中遇到的一系列严重问题和难题。它包括两方面：如何开发软件，已满足对软件日益增长的需求；如何维护数量不断增长的已有软件。
3. 软件维护：在软件已经交付使用之后，为了改正错误或满足新的需要而修改软件的过程。目的：通过必要的维护工作使得系统持久的满足用户的需要。
4. 虚继承作用:
5. 私有属性和保护属性
6. 虚函数和重载函数的联系区别
7. 数据字典
8. 数据保护
9. define声明一个常量
10. 数据保护四个方面
11. C＋＋的特性
12. DDL
13. C++静态成员
14. 派生函数和析构函数的执行顺序
15. 简单介绍开发的管理信息系统
16. C和JAVA的区别
17. 用new创建单精度数组array，其长度为100
18. 提高软件可维护性的措施
19. 有参数和无参数的同名函数是否等价
20. 什么是析构函数以及析构函数的作用
21. 虚函数和基函数
22. 继承有哪些种类
23. 写一个#define的语句
24. 软件的控制类型，有什么共同特点
25. 虚函数用什么关键字表示
26. 什么是静态数据成员，用哪个关键字表示
27. 自己做过的项目
28. 重载函数的三种定义

注：多看看毕设，必问！！！！

2016

主要是每本书前面的理论知识

**C++面向对象程序设计**

1. 虚继承的作用
2. 虚函数的定义
3. 静态多样性
4. 重载函数
5. 虚函数与重载函数的联系区别
6. 类继承的定义
7. 析构函数的定义和作用
8. 有参数和无参数的同名函数是否等价
9. 类声明是否可以初始化
10. 说出三个典型的面向对象的语言
11. 面向对象和面向过程语言有什么区别
12. C++的特性

大学时学的什么专业课程，毕设内容要看看

如果学的软件测试一定要把当时的大概工具什么记清楚，学姐我的**惨痛教训！！！！**