1. Pverview of the Design Process
   1. Design Phases
   2. Design Alternatives  
      避免Redundancy\Incompleteness，并在可行的若干设计中选出最适合的，这需要科学和好的“品味”
2. E-R 模型  
   通过建立实体到概念模型的映射，Entity-Relationship Model可以表达整个数据库的逻辑结构，很多数据库产品都采用E-R模型来表达数据库设计。E-R模型采用了三个基本概念：实体集、关系集和属性。
   1. 实体集  
      实体Entity相当于现实中的事物或对象，比如大学中的一个person就是一个entity，实体具有一系列属性，通过其中的某个或某几个属性可以唯一地确定这个实体，比如person\_id。实体可以是现实中具体的事物，比如person、book，也可以是抽象的事物，比如money、course等。  
      实体集是一系列具有相同属性实体的集合，比如某所大学的所有教师构成了一个称为instructors的实体集。  
      实体集可以是相交的，假如persons实体集为学校中的所有人，则一个person实体可能是一个instructor实体或者一个student实体，或者两者都不是。
   2. 联系集  
      联系是指多个实体间的相互关联。比如可以定义教师A与学生B之间的联系advisor，这一联系指明A是B的导师。联系集是相同类型联系的集合，是n>=2个实体集直接的数学关系。实体集之间的关联称为参与(participate)，联系也可以具有描述性属性(descriptive attribute)，实体集instructor和student之间的联系为advisor，可以将属性date与该联系关联起来，表示教师成为学生导师的日期。
   3. 属性  
      每个属性都有一个可取值的集合，称为属性的域(domain)或者值集(value set)。Course\_id的域可能是特定长度的所有字符串的集合，semester的域则可以是春夏秋冬四个季节。实体有多个属性，每个属性可以用属性名-值的键值对来描述，比如instructor的一个实体可以描述为：{(ID,7677),(name,Crick),(dept\_name, Biology)}
      1. 属性的分类
         1. 简单属性与复合属性。简单属性无法再被划分为更小的部分，复合属性却可以。比如name可以进一步划分为first\_name\middle\_name\last\_name。如果在一些场景下要使用整个属性，在另外的场景则只需要属性的一部分，则适合使用复合属性。复合属性有助于分组相关的属性，让建模更清晰。
         2. 单值属性和多值属性。  
            一般情况下一个属性只会有一个值，但有些多值的情况，比如phone\_number，一个人可能会有多个电话号码。多值属性用花括号区分，{phone\_number}，在适当的情况下，可以设置多值属性的上下界，比如设定phone\_number的数目为1-2。
3. 约束Constraints
   1. 映射基数(Mapping Cardinalities)  
      映射基数也称映射比率，表示一个实体统一联系集能关联的实体个数，常用于二元联系集。假设R为实体A-B之间的联系集，则映射基数有如下几种情况：
      1. 一对一(one to one)，A的实体至多与一个B实体关联，反之亦然。
      2. 一对多(one to many)，A的实体可以与任意多个B实体关联(>=0)，B实体则只能与至多一个A实体关联。
      3. 多对多(many to many)。A和B实体间相互关联的数目不受限制。
   2. 参与约束(Participation Constraints)  
      如果实体集E中的每个实体都参与到联系集R的至少一个联系中，则E在R中的参与是全部的(total)，否则为部分的(partial)。比如A-B为一对一映射，A中的实体数多于B，则A部分参与，B全部参与。
   3. 键
4. 移除实体集中的多余属性
5. 弱实体集
6. E-R模型转换为关系模型
   1. 转换具有简单属性的强实体集  
      假设E是具有简单描述性属性a1,a2..的强实体集，可以转换为具有同样属性的关系模型E，它的一个元祖对应实体集中的一个实体，主键也是实体集E的主键。
   2. 转换具有复杂属性的强实体集  
      对于复合属性，转换时需要为每个子属性创建单独的属性，比如将name拆分为first\_name, middle\_name, last\_name；对于多值属性，需要使用多个元祖来表示，比如ID为22的教师的phone\_number有555-1234与444—4321，则对应两条元祖(22, 555-1234)与(22, 444—4321)，这两个属性共同构成了主键。
   3. 转换联系集  
      联系集也转换为关系模型，所有参与联系的实体集的属性和联系集的描述性属性构成了转换的模型的属性。关于主键，对于多对多的二元联系，主键为参与联系的实体集的主键的并集；对于一对一的联系，可使用任意一方的主键；对于多对一的联系，选择“多”端的主键。
8. 扩展的E-R特性  
   虽然基本的E-R特性可以应对大部分数据库建模，但为了使用方便，也提供扩展的E-R特性
   1. 特化Specialization  
      有时一个实体集会保护若干子集，这些子集各自具有不同的属性。比如person可以包含employee和student子集，这些子集在person本身属性的基础上，附加了特殊的属性，employee会有salary属性，student会有score属性。特化就是在实体集内部分组的过程。  
      在E-R模型中特化用从特化实体指向被特化实体的空心箭头线表示，叫做ISA联系，既“is a”，比如从employee指向person，也既employee is a person。  
      有些实体集可以有不同的特化方式，employee可以特化为instructor、secretary，也可以特化为temporary\_employee、permanent\_employee，这样一个employee可能是一个instructor的同时也是一个permanent\_employee，这种一个实体可以属于多个实体集的特化称为重叠特化(overlapping specialization)，只能属于一个实体集的特化则称为不相交特化(disjoint specialization)。重叠特化用两个独立的空心箭头，不相交特化使用一个箭头。
   2. 概化(Generalization)  
      特化是在自顶向下(top-down)的设计过程中，显式地将初始实体分为一系列不同级别子实体的过程。设计过程也可以是自底向上(bottom-up)的，将一系列低级别的实体合成为具有基本属性的顶层实体，这便是概化，高层实体集与一个或多个低层实体集之间为包含关系。高层与低层实体集也可以分别称为超类(superclass)和子类(subclass)。
   3. 属性继承（Attribute Inheritance）  
      由特化和概化产生的高层和低层实体的一个重要特性是属性集成，高层实体集的属性被低层实体集继承，所以高层实体集的属性适用于低层实体集，但反之则不成立。
   4. 概化的约束
      1. 条件定义的  
         在条件定义的低层实体集中，根据实体是否满足某个特定的条件来确定成员资格。比如通过student.student\_type=granduate/undergranted来区分子实体是本科生还是研究生。
      2. 用户定义的  
         用户定义的低层实体集由数据库用户自行决定实体的划分。比如将教师分为几个教学组，employee的划分方式由用户决定。
      3. 完全性约束(completeness constraint)  
         全部概化\特化：每个高层实体必须属于一个低层实体集；  
         部分概化\特化:不是所有高层实体都有属于的低层实体集。  
         这在数据插入、删除上会带来区别。