## [分析与设计数据库模型的简单过程](https://www.cnblogs.com/studyzy/archive/2010/01/11/1643976.html)

近期在做一个业务系统的分析和数据模型设计，工作这几年也做过好几个项目的数据库模型的设计，期间也算是积累了一定的经验吧，这次有机会就写写我的数据库模型设计过程与方法。

在数据库设计中，设计的目标就是要建立E-R图（实体-关系图），在PowerDesigner中就是要建立概念模型或者逻辑模型。既然是实体-关系图，所以整个建模的核心就是围绕建立“实体”对象和找到实体之间的“关系”。实体分为两部分：标识（主键）和属性。标识是实体的一个或多个属性的组合，用于唯一的表标识出实体中的每一个数据。在确认一个实体的过程中，首先就是要确认实体的主键，只要找到了实体的主键，那么剩下的就是实体的属性。

### 1.确认核心实体

在建模过程中，首先需要对业务进行分析，知道我们的模型要表示怎么样的一个事情，从而确定我们模型的核心实体，找到了核心实体和其主键，那么剩下的工作就是以核心实体为中心进行实体关联的扩展和实体属性的抽象。一个数据库模型中一般会有1~2个实体作为整个模型的核心实体，核心实体一般都是一个名词，在整个业务过程中作为主语和宾语。所以总的来说，我们用一个主谓宾的句子来描述我们这个模型，那么基本就可以肯定，这句话中的主语和宾语就是核心实体，而通常谓语也是一个很核心的对象，该对象可能会产生一个实体来表示，也可能只是一个关联（Association）。通常数据库中数据量最大的表就是谓语对应的表。

以上说法可能比较抽象，用一两个简单的例子来说明。假设我们需要设计一个学生选课系统的数据库模型，那么首先就是要分析，我们这个系统是做什么的，记录什么的？“学生选课”！虽然只有4个字，但是已经完整的表达整个系统，从这样一个主谓宾的句子中，我们可以得出，整个模型的核心是“学生”（主语）和“课程安排”（宾语），谓词“选”表名了两个实体之间的核心关系。确定了核心的实体“学生”和“课程安排”，那么接下来就是要确定实体的主键和属性。“学生”实体的主键很容易确定，只要找到能够唯一标识每个学生的一个字段即可，所以我们可以使用“学号”来作为学生实体的主键，一个学校中每个学生的学号肯定是唯一的。“课程安排”这个实体的主键并没有那么明显的属性能够表示，对于无法找到明显的实体属性作为主键的情况下，我们需要创建一个专门的标识列（ID）用来标识实体中的每个实例。在数据库中最常见的ID就是自增列。这里我们可以设计“课程安排ID”作为课程实体的主键，每在数据库中增加一门课程，系统会自动为该课程分配一个自增的唯一整数来标识。

[](http://images.cnblogs.com/cnblogs_com/studyzy/WindowsLiveWriter/8b52fb11571c_10DB3/image_2.png)

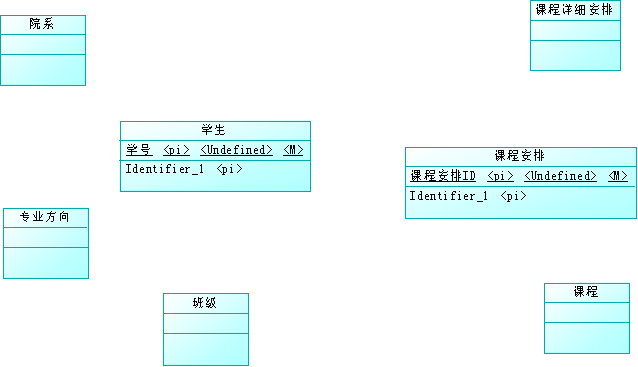
再比如一个要设计一个电子商务系统的数据库模型，首先一句话总结该系统就是“用户在网上购买商品”，所以这个系统的核心实体就是“用户”和“商品”。用户实体的主键是什么？用户的登录名是唯一的、邮箱是唯一的，都可以作为该实体的主键。但是在真实的电子商务系统中很少使用登录名或邮箱来作为主键，因为其中一个很重要的原因是登录名和邮箱都太长，而且长度不确定，所以在数据库中一般会设计一个自增的“用户ID”来作为用户的主键。商品实体的主键可以用商品的条形码来作为主键，确实可以这么做，但是同样的原因，条形码太长了，所以一般会用一个Int型的自增列“商品ID”作为商品的主键。

[](http://images.cnblogs.com/cnblogs_com/studyzy/WindowsLiveWriter/8b52fb11571c_10DB3/image_6.png)

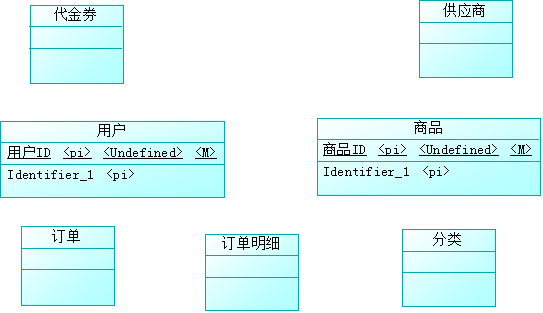
### 2.确认相关实体

在找到了核心实体后，接下来就是以核心实体为中心，找到相关的实体。相关实体一般来说就是和核心实体存在直接联系的实体，当然也有些相关实体是要经过另一个相关实体与核心实体关联。相关实体一般情况下都是名词。

以选课系统为例，与学生相关的实体是什么？班级、专业方向、院系等，与课程安排相关的实体是什么？课程、课程的详细安排、安排的教师等，所以我们可以将这些要关联到的实体都建立。

[](http://images.cnblogs.com/cnblogs_com/studyzy/WindowsLiveWriter/8b52fb11571c_10DB3/image20.png)

再看看前面说到的电子商务平台，核心实体是用户和商品，围绕用户，我们需要建立用户的“订单”（包括订单的明细）、用户的“代金券”等实体，围绕商品，我们需要建立商品的分类，商品的供应商等相关实体。于是我们的电子商务数据库模型变为：

[](http://images.cnblogs.com/cnblogs_com/studyzy/WindowsLiveWriter/8b52fb11571c_10DB3/image_26.png)

这一步并没有完成，一个实体可以没有属性，但是却不能没有主键，所以需要给所有相关实体添加主键，我们可以以简短的可以唯一标识实体的属性来作为主键，也可以使用自增的ID作为主键，在数据库中出于性能、快捷等方面的考虑，大部分实体都是以ID作为主键。

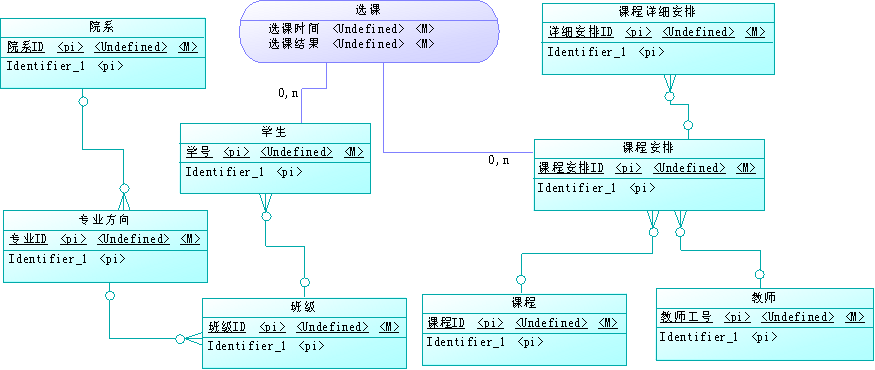
### 3.确认关联和关系

关联（Association）也是一种实体间的连接，在Merise模型方法学理论中，Association是一种用于连接分别代表明确定义的对象的不同实体，这种连接仅仅通过另一个实体不能很明确地表达，而通过“事件（Event）”连接来表示。

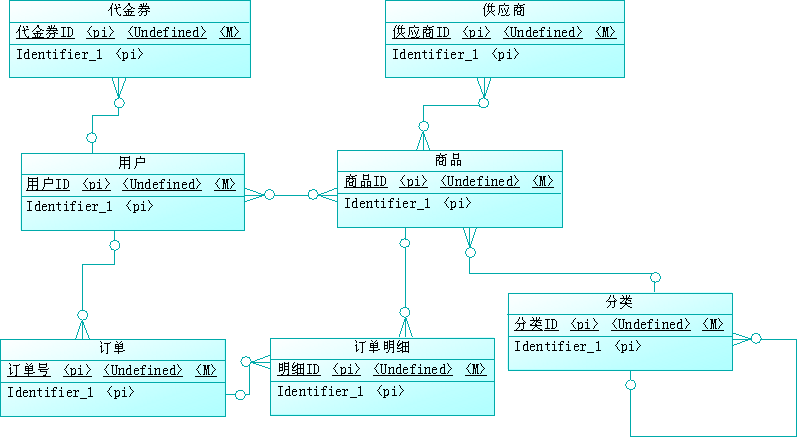
也就是说，实体和实体之间存在着关系（多对多），但是这种关系还存在其他的属性，这些属性如果如果作为一个明确的实体的实体来表示又不是很合适，所以就使用了Association来表达，这种关系之间一般是一个“事件”虚实体，也就是说是一个动词对应的实体。

以选课系统为例，“选课”这个动词就是需要用关联来表示，一个学生可以选择多个课程安排，一个课程安排会有多个学生来选，所以学生和课程安排之间是多对多的关系，但是学生选课时还需要记录学生的时间、选课是否成功等信息，所以需要使用关联来表示选课这个动作。

前面说到的多对多是实体之间的一种关系，两个实体之间存在4种关系：一对一、一对多、多对一和多对多。根据核心实体和相关实体之间的关系建立实体之间的关系，于是我们的选课系统数据库模型如图所示：

[](http://images.cnblogs.com/cnblogs_com/studyzy/WindowsLiveWriter/8b52fb11571c_10DB3/image46.png)

对于一个电子商务系统，分析其中的实体之间的关系，也可以得到类似的关系图。要表示用户对商品的收藏，也就用户和商品两个实体直接的直接关系，一个用户可以收藏多件商品，一个商品可以被多个用户收藏，所以用户和商品之间是多对多的关系。另外，商品分类和自身是一个一对多的关系，因为分类存在大分类和小分类，是一种层级关系，一个父级分类下面有多个小分类，一个小分类只会有一个父级分类，所以分类自身一对多。

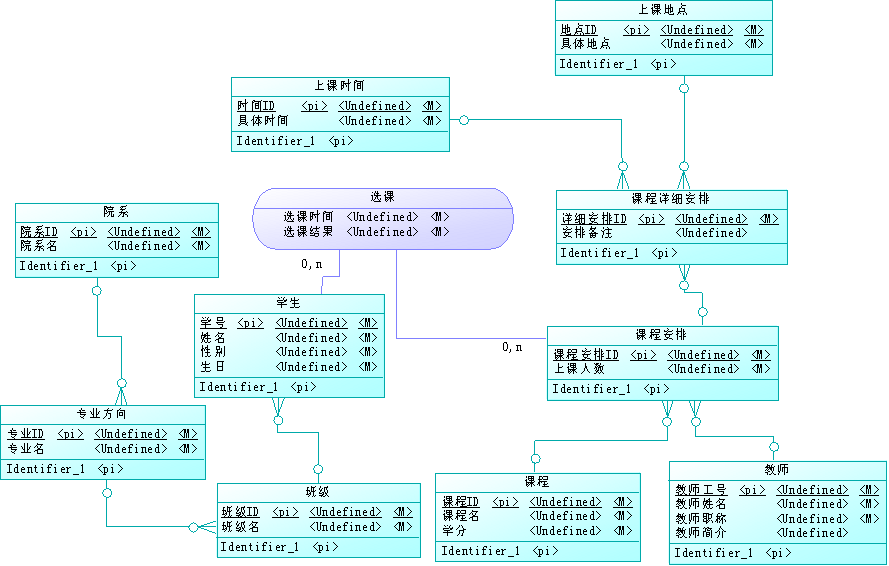
[](http://images.cnblogs.com/cnblogs_com/studyzy/WindowsLiveWriter/8b52fb11571c_10DB3/image_24.png)

### 4.确认属性

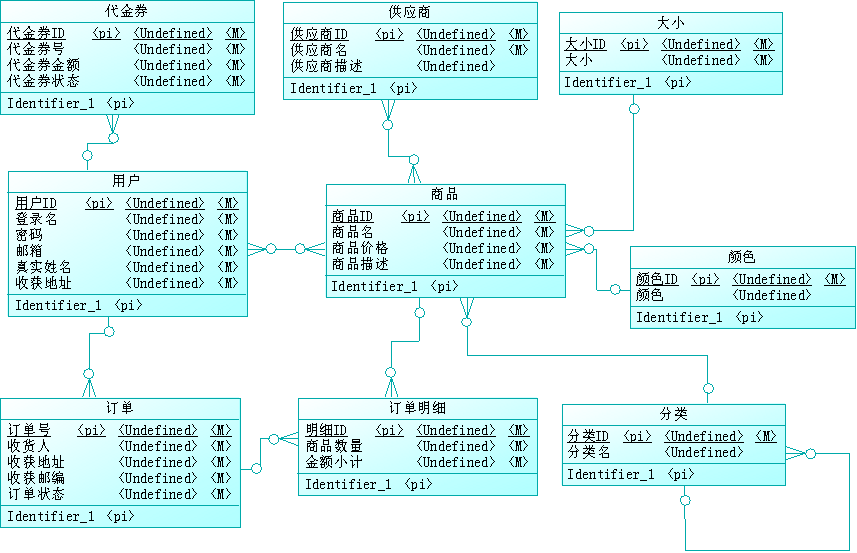
前面几步工作时最重要最核心的工作，接下来的工作就是要完善模型。首先需要的就是要将实体的属性补齐，实体的属性可以根据日常生活常识、用户提交的表单、用户需求调研等来确定。比如学生表，根据常识我们知道，学生会具有姓名、性别、生日等属性；课程会具有课程名、学分等属性；课程的详细安排会安排具体的时间、上课的地点等属性……在实际的企业应用中，大部分实体的属性时不可能通过常识来得到的，必须进行需求的调研，结合业务上的需求和实际中的表单、数据流等找到实体的属性。比如对于供应商这个实体，我们只知道供应商有编号，有名字，还有其他什么属性就必须得调研了。调研时我们知道企业新增加一个供应商时会填写一个新增供应商表，那么我们就可以拿到该表，更加表单的内容来设计供应商实体的属性。

### 5.范式化

在前面设计选课系统的数据模型时，对于选课的详细信息实体，会存在上课的时间、上课的地点等属性，但是仔细一考虑，这些属性如果直接放在该实体中，必然会形成数据重复，导致数据维护困难，不符合3范式的设计原则，所以应该将这些属性提出，作为单独的实体，于是，我们的选课系统的数据库模型就变为如图所示：

[](http://images.cnblogs.com/cnblogs_com/studyzy/WindowsLiveWriter/8b52fb11571c_10DB3/image1.png)

再说下电子商务系统的模型，里面最重要的一个实体“商品”会包含很多属性，比如大小、颜色、重量、卖价……，这其中，大小、颜色本身也可以作为实体抽取出来，以便于进行维护，所以我们的电子商务系统的模型便为：

[](http://images.cnblogs.com/cnblogs_com/studyzy/WindowsLiveWriter/8b52fb11571c_10DB3/image_22.png)

### 6.细节调整

现在整个模型已经基本上完成了，但是仍然有几个地方需要进一步的确认和调整：属性的数据类型和实体之间的关系。现在数据库模型中，所有的属性的数据类型都是Undefined，需要根据系统要求、业务需求和调研来确定每个属性的数据类型。但一般来说还是具有一定的规则可循：

自增ID用Integer型，如果数据量会特别特别大的话，可以使用长整型。

涉及到金额的用Money类型。

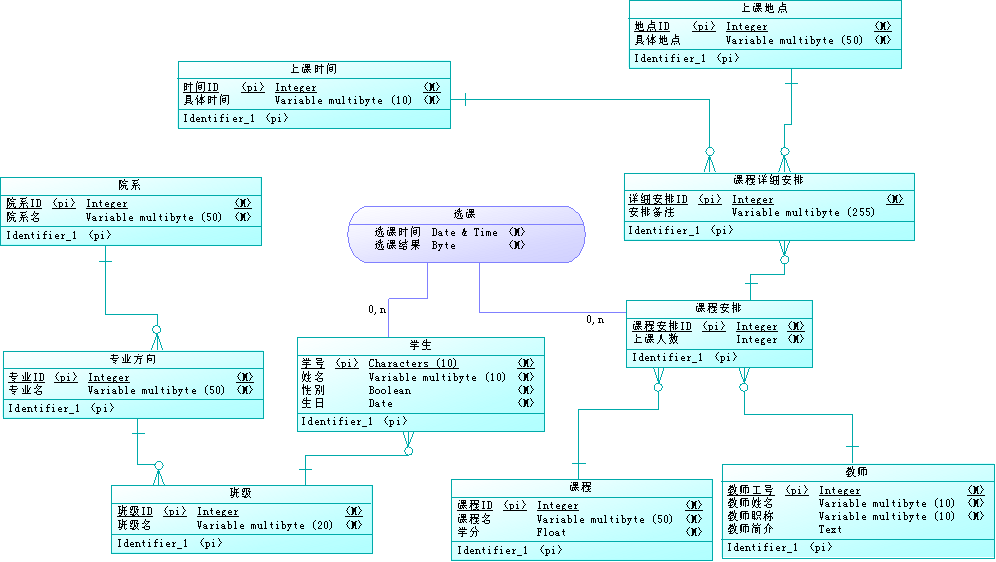
涉及字符串的确定该属性中是否有可能出现中文，如果有中文出现的，用variable multibyte，没有中文出现那就用Characters或者variable Characters。

如果是枚举类型的，用Byte。

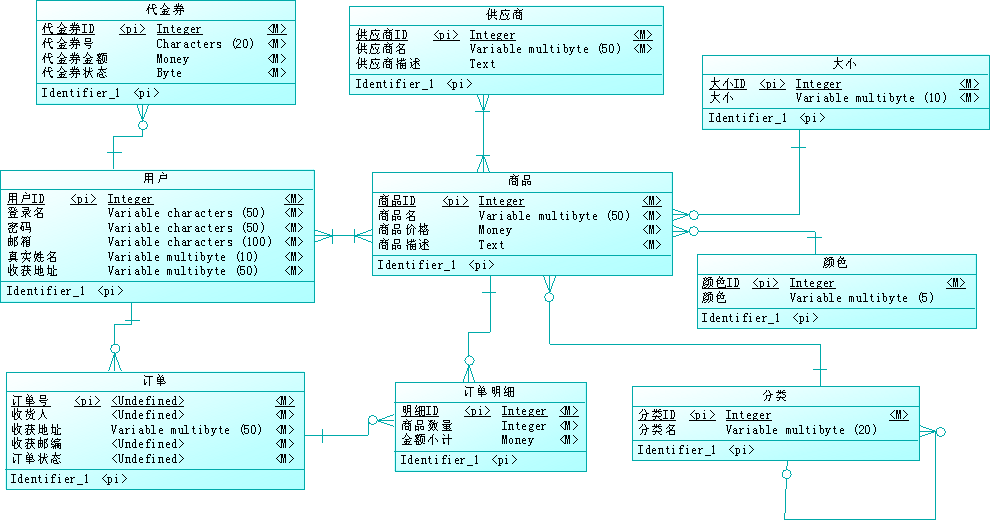
日期和时间类型的，确定是要用日期还是用时间，或者两者都需要记录。

具有小数的用float类型。

按照实际情况将模型中的每个属性的数据类型进行修改。另外就是实体之间的关系，在默认情况下，添加的实体关系是一对多的关系，另外也可能存在一对一或者多对多的关系，除了这些关系外，另外还需要确定对应的关系实体是否是必须的。一对多中，一这部分就存在0,1 和1,1两种情况；多的部分存在0,n和1,n两种情况。最常见的情况是1,1:0,n，也就是说多的一端肯定会对应一个一的实体，而一的一端可以对应0到多个实体。

[](http://images.cnblogs.com/cnblogs_com/studyzy/WindowsLiveWriter/8b52fb11571c_10DB3/image_10.png)

再比如电子商务系统，确定该数据库模型中每个实体属性的数据类型，然后修改实体之间的关系，将必须存在值对应的地方修改为1,1或者1,n即可。

[](http://images.cnblogs.com/cnblogs_com/studyzy/WindowsLiveWriter/8b52fb11571c_10DB3/image_12.png)

通过以上几步操作，我们可以建立完整的数据库概念模型，主要应该关注在实体的建立（核心就是要找到实体的主键）和实体关系的建立（核心就是找到实体直接是一对多还是多对多或者一对一），只要把这两点做好，那么整个模型的框架就搭建好了。