

PowerDesigner 概念模型详解

一、概念模型的重要性

PowerDesigner 是最强大、最优秀的数据库建模工具，是 Sybase 公司最伟大的产品。从 9 一直用到现在，对 PD 的认识也是在逐步加深。

常常在工作中，看到大家用 PD，都是用来建几个表，实际上是做 PDM (Physical Data Model)，上来就干这个，实际上，这么用 PD，是对强悍的 PD 一种侮辱。PD 仅仅是这么玩的吗？

数据库设计的步骤是什么，难道上来稍稍想一下就搞个 PDM 出来？

下面简单回顾下大学课本里讲述的数据库设计的基本步骤：

1、需求分析

从系统需求中寻找一些概念性名词，并甄选，并对这些名词相关属性做了解，这部分是人工的，PD 做不了什么。

2、概念结构设计

针对甄选的名词进行分心，找出其中的关系（独立的、一对一、一对多、多对多、继承五种关系），并用 E-R 图描述出来，这是大学课本的做法。在 PD 中，这个过程可以用 CDM（概念模型）来描述，PDM 中实体概念模型表示方式比 E-R 更清晰，更好。

3、逻辑结构设计

实际上就是设计表的结构和表之间的主外关系等。这部分在 PD 中对应的是 PDM（物理模型），而 PD 中的物理模型一般都是直接从概念模型生成的。也就是说，只要你做好概念模型，物理模型就可以自动生成。

当然，这种生成结果一般都需要做一些调整和优化。

4、物理结构设计

有了 PDM，数据库的物理设计将不费吹灰之力，直接可以从 PDM 导出各种数据库系统的建库脚本。

5、数据库的建立和测试

这个过程也很简单，看看建库脚本的执行就知道了。不合理了重新修改 PDM，然后生成 sql 再来。

6、数据库运行和维护。

这个一般是 DBA 的事情了，比如时间长了，数据量大了，在某些列上加上索引，调优等等。

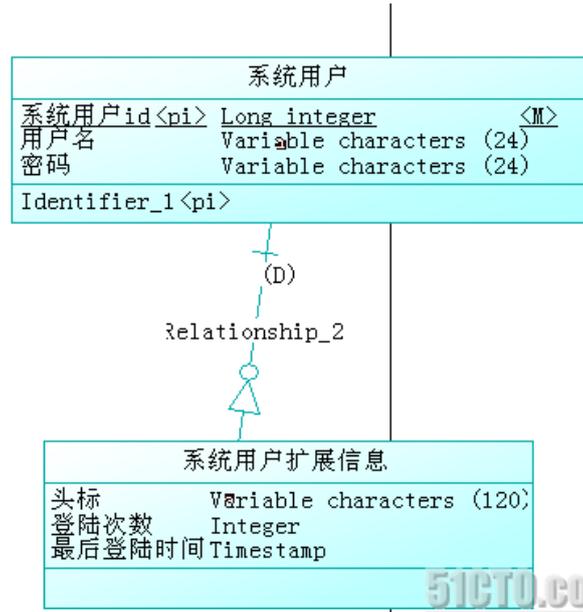
从中可以看到，一上来就建 PDM，是不合理的。实际上要求对概念模型有个透彻理解了才去做 PDM，这种理解可以不画图，但至少是心中有图。

做CDM (Conceptual Data Model) 概念模型的好处是交流容易，全世界通用，谁看了都明白。你难道能用PDM的外键关系去看数据关系吗，如果一个表上有多个外键，外键关联像蜘蛛网一样，就晕菜了，谁也看不明白！

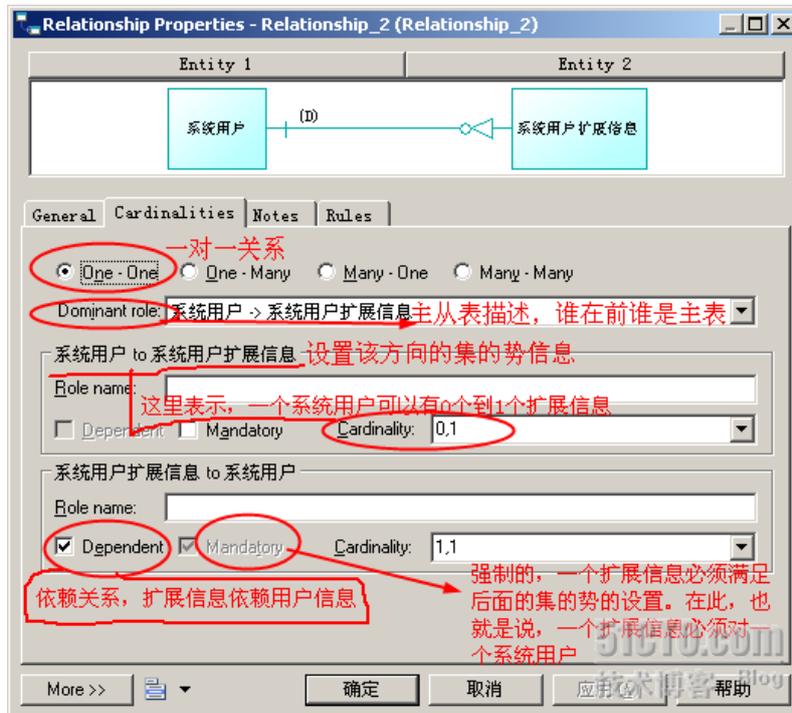
二、使用 PD 建立数据库概念模型

1、一对一 CDM

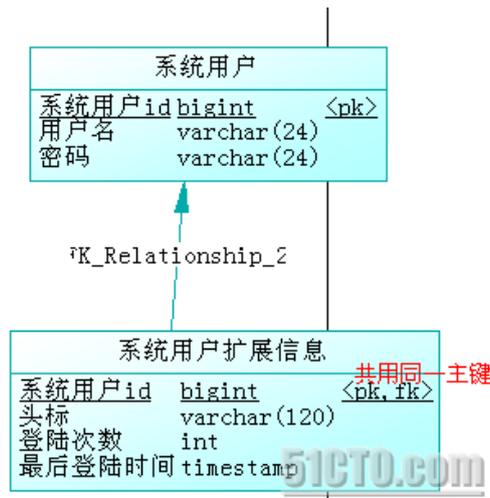
下图描述了一个系统用户对应一个扩展信息，也可以没有扩展信息。扩展信息依赖用户信息的存在。并且一个扩展信息只能有一个用户信息。



关系的设置:

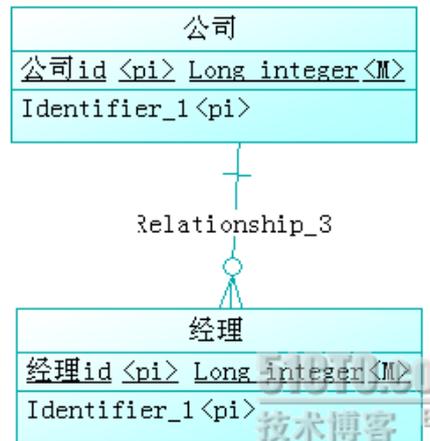


将其生成 PDM

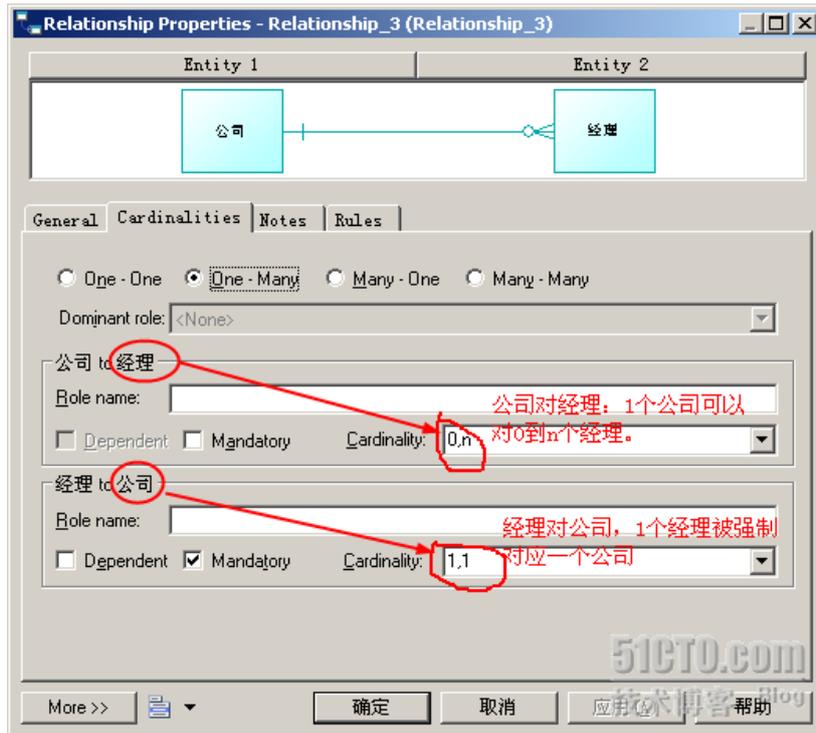


2、一对多 CDM

下图 C D M 描述了一个公司有多个经理的模型，当然一个公司也可以没有经理（老板是光杆司令）。但一个经理必须属于一个公司。



关系的设置:

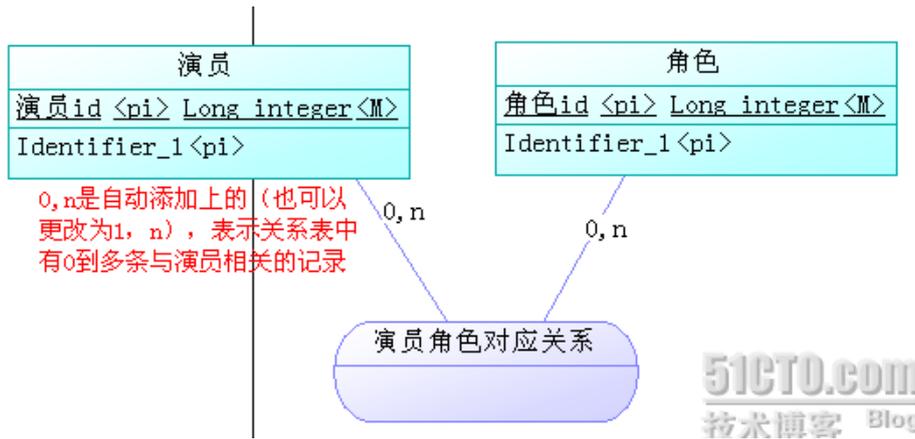


生成的 PDM

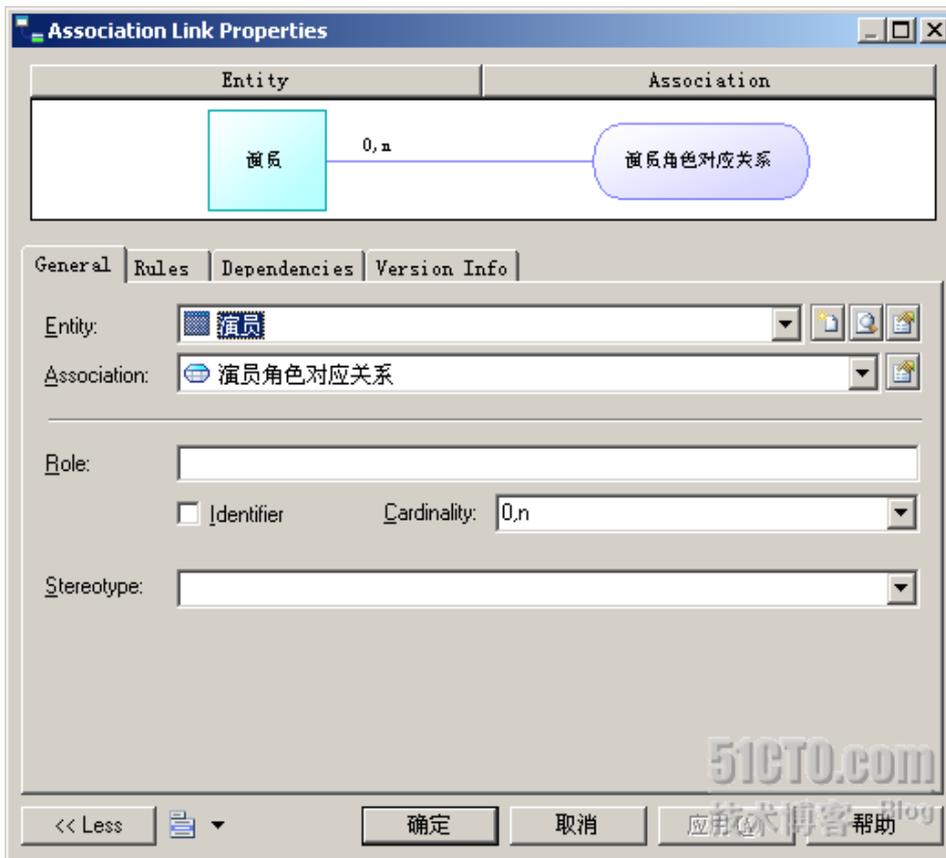


3、多对多 CDM

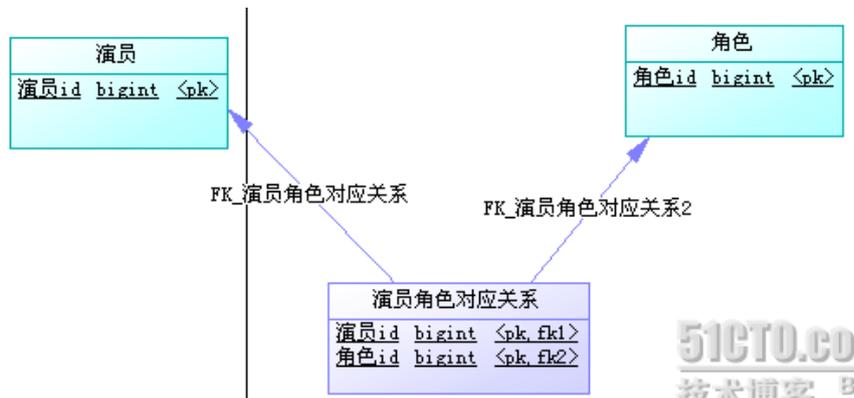
下面描述的是一个演员和角色的关系，一个演员可以演多个角色，一个角色可以由不同的演员来演，比如《红楼梦》的林妹妹，小时候找个演员 A 来演，长大后的形象由演员 B 来演。



关系设置，多对多关系最简单了，一般不需要设置：

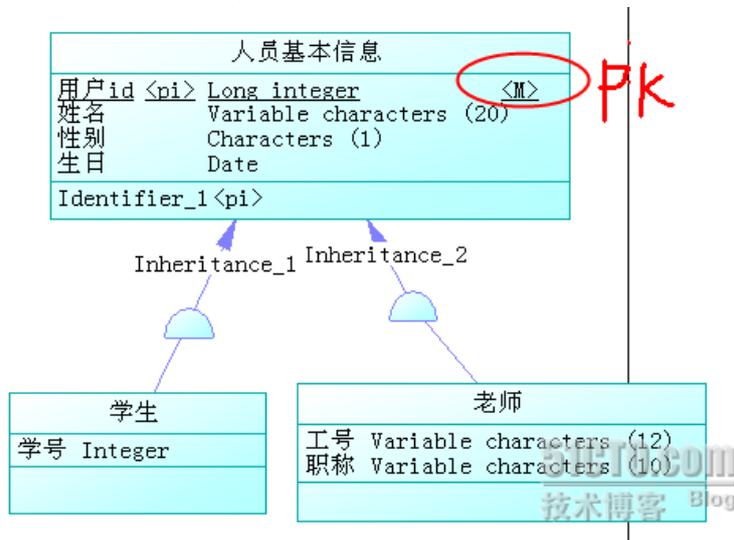


生成的 PDM 如下：



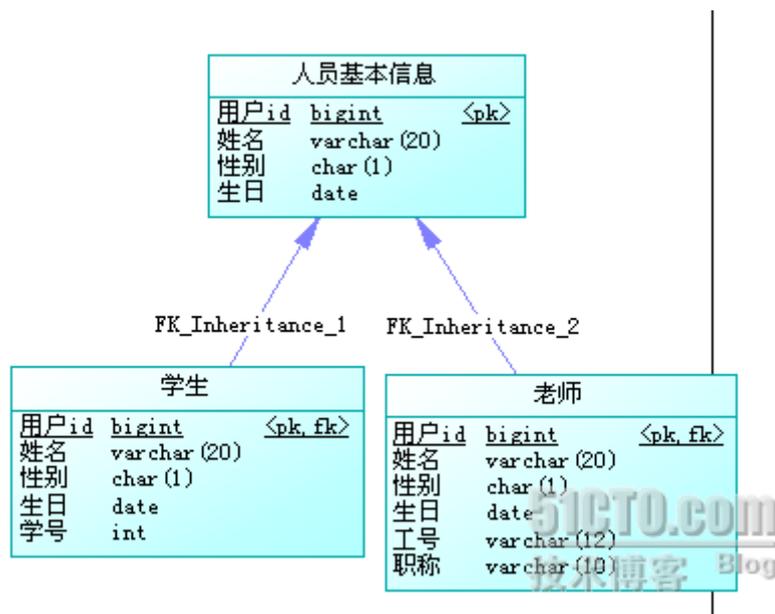
4、继承关系 CDM

下图描述的是一个继承关系，比如有一个教务系统，用户分学生老师，注册时候，老师和学生填写的信息有差异，但有公共信息。



关系配置，不需要，就集成关系，没啥好设置的。

生成的 PDM 如下：



也许你会发现，五个呢，怎么才四个，呵呵，单表就独立着和谁都没关系，还用画吗？

三、总结

1、数据库建模是系统设计中最重要一步，概念模型能很好的描述数据间的关系，还可以从概念模型精确生成符合一定标准范式的物理模型。

2、CDM 能描述出更细微的数据关系，比如是 0-n 还 1-n，这直接影响到数据业务上的约束，但是用 PDM 无法描述。CDM 为业务交流节约了沟通成本。

3、CDM 也为后来了解底层业务数据关系提供了依据，尤其是表很多很多时候，如果没有 CDM，那只有设计数据库的人知道底层的关系了。

4、如果表很多，分模块的情况，还可以讲 CDM 分包来管理，这样可以避免将所有的实体关系画到一张图中所带来阅读上烦恼。

5、PD 还有其他很多很强悍的功能，比如数据库反响到 PDM，PDM 导出脚本，PDM 导出 Java 模型对象、XML 模型。还可以生成 DAO 层的持久化代码，甚至 hbm 文件，还可以做业务流程建模、生成数据字典报表等等。但 PD 最擅长的就是 CDM-->PDM-->SQL，数据库反向工程，报表功能，用好这些就不错了。

使用PowerDesigner画E-R图详细教程

一、概念数据模型概述

数据模型是现实世界中数据特征的抽象。数据模型应该满足三个方面的要求：

- 1) 能够比较真实地模拟现实世界；
- 2) 容易为人所理解；
- 3) 便于计算机实现

概念数据模型也称信息模型，它以实体-联系(Entity-RelationShip,简称 E-R)理论为基础，并对这一理论进行了扩充。它从用户的观点出发对信息进行建模，主要用于数据库的概念级设计。

通常人们先将现实世界抽象为概念世界，然后再将概念世界转为机器世界。换句话说，就是先将现实世界中的客观对象抽象为实体(Entity)和联系(Relationship),它并不依赖于具体的计算机系统或某个 DBMS 系统，这种模型就是我们所说的 CDM;然后再将 CDM 转换为计算机上某个 DBMS 所支持的数据模型，这样的模型就是物理数据模型,即 PDM。

CDM是一组严格定义的模型元素的集合，这些模型元素精确地描述了系统的静态特性、动态特性以及完整性约束条件等，其中包括了数据结构、数据操作和完整性约束三部分。

- 1) 数据结构表达为实体和属性；
- 2) 数据操作表达为实体中的记录的插入、删除、修改、查询等操作；
- 3) 完整性约束表达为数据的自身完整性约束（如数据类型、检查、规则等）和数据间的参照完整性约束（如联系、继承联系等）；

二、实体、属性及标识符的定义

实体 (Entity)，也称为实例，对应现实世界中可区别于其他对象的“事件”或“事物”。例如，学校中的每个学生，医院中的每个手术。

每个实体都有用来描述实体特征的一组性质，称之为属性，一个实体由若干个属性来描述。如学生实体可由学号、姓名、性别、出生年月、所在系别、入学年份等属性组成。

实体集 (Entity Set) 是具体相同类型及相同性质实体的集合。例如学校所有学生的集合可定义为“学生”实体集，“学生”实体集中的每个实体均具有学号、姓名、性别、出生年月、所在系别、入学年份等性质。

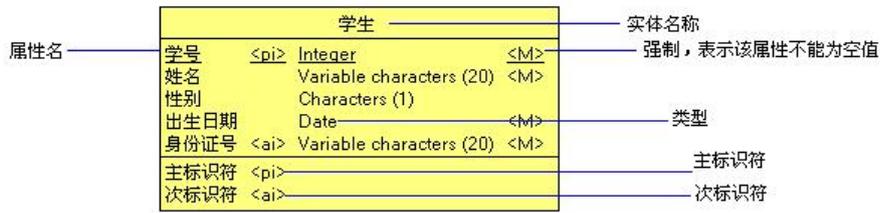
实体类型 (Entity Type) 是实体集中每个实体所具有的共同性质的集合，例如“患者”实体类型为：患者 {门诊号, 姓名, 性别, 年龄, 身份证号.....}。实体是实体类型的一个实例，在含义明确的情况下，实体、实体类型通常互换使用。

实体类型中的每个实体包含唯一标识它的一个或一组属性，这些属性称为实体类型的标识符 (Identifier)，如“学号”是学生实体类型的标识符，“姓名”、“出生日期”、“信址”共同组成“公民”实体类型的标识符。

有些实体类型可以有几组属性充当标识符，选定其中一组属性作为实体类型的主标识符，其他的作为次标识符。

三、实体、属性及标识符的表达

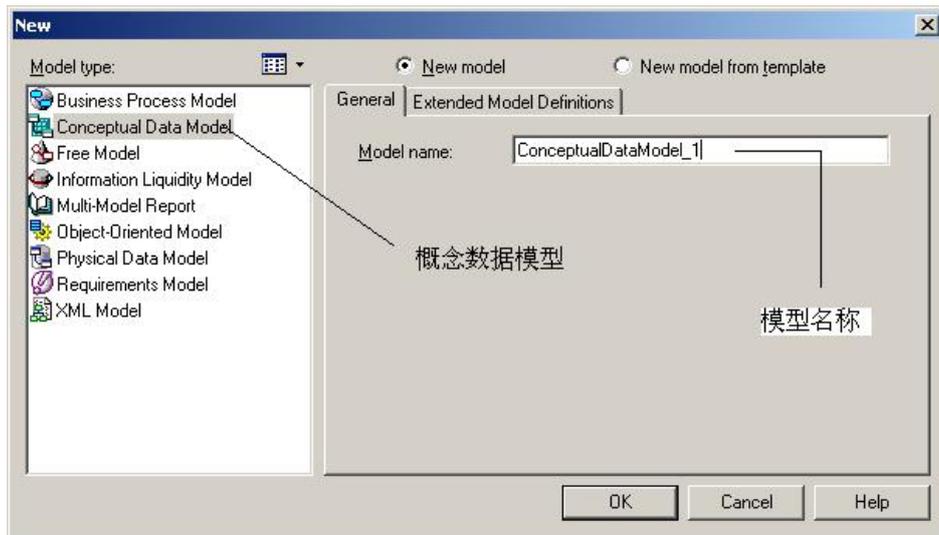
介绍 PowerDesigner 概念数据模型以及实体、属性创建。



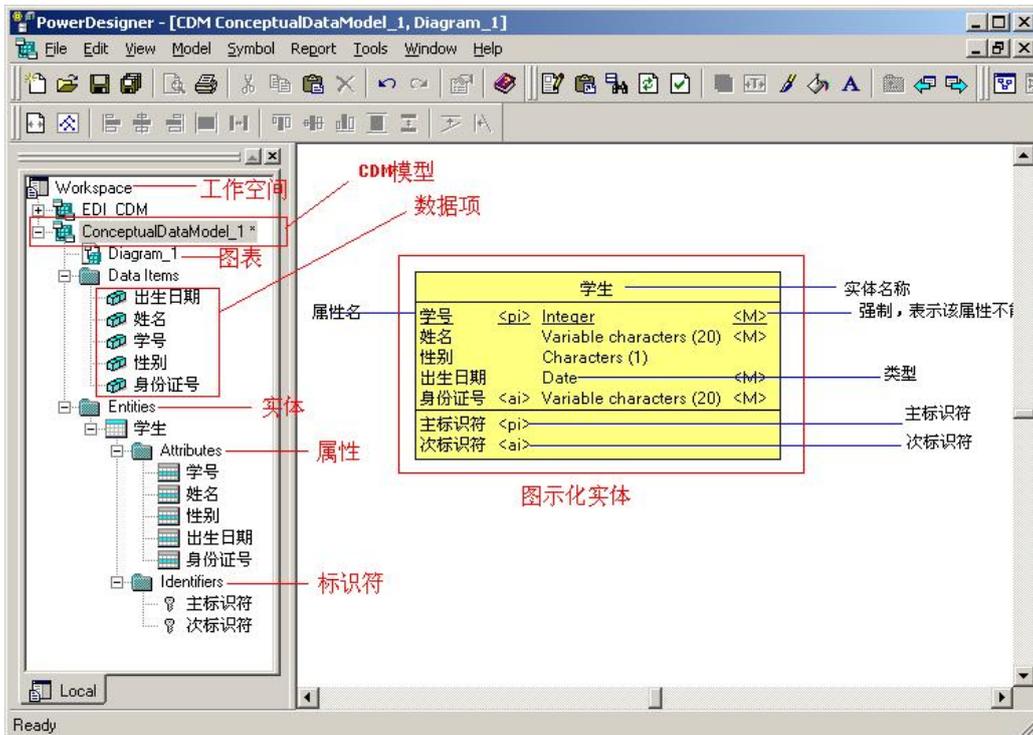
实体的表示方法

一、新建概念数据模型

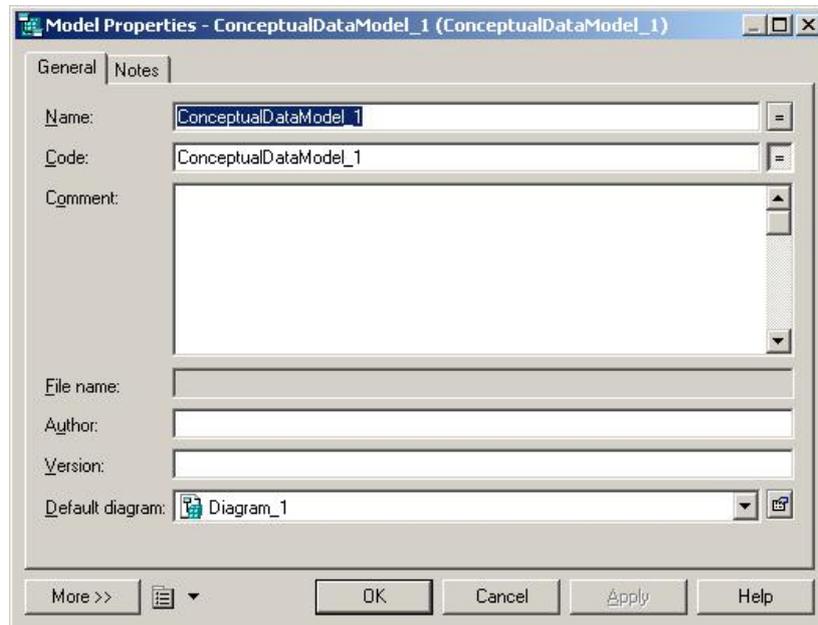
1) 选择 File-->New,弹出如图所示对话框,选择 CDM 模型(即概念数据模型)建立模型。



2) 完成概念数据模型的创建。以下图示,对当前的工作空间进行简单介绍。(以后再更详细说明)

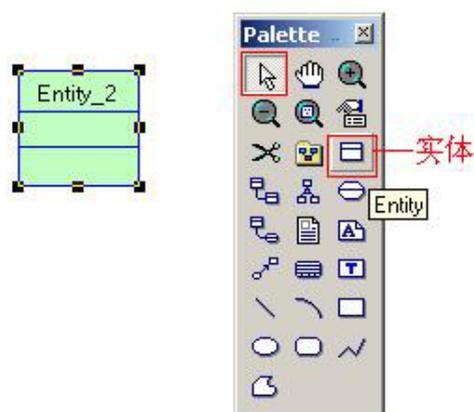


3)选择新增的 CDM 模型,右击,在弹出的菜单中选择“Properties”属性项,弹出如图所示对话框。在“General”标签里可以输入所建模型的名称、代码、描述、创建者、版本以及默认的图表等等信息。在“Notes”标签里可以输入相关描述及说明信息。当然再有更多的标签,可以点击 "More>>"按钮, 这里就不再进行详细解释。

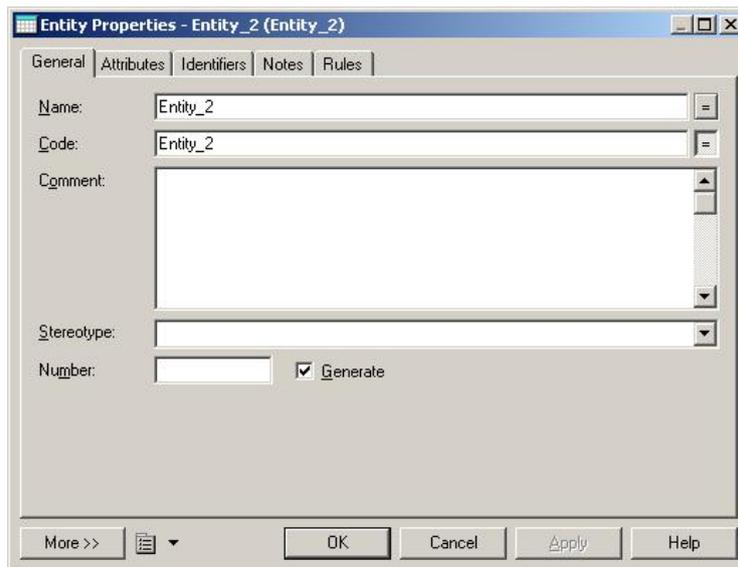


二、创建新实体

1) 在 CDM 的图形窗口中,单击工具选项版上的 Entity 工具,再单击图形窗口的空白处,在单击的位置就出现一个实体符号。点击 Pointer 工具或右击鼠标,释放 Entity 工具。如图所示

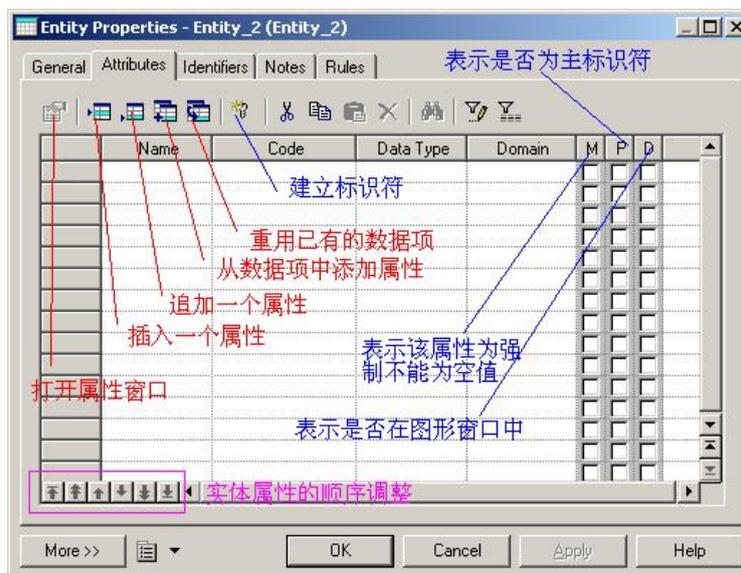


2) 双击刚创建的实体符号,打开下列图标窗口,在此窗口“General”标签中可以输入实体的名称、代码、描述等信息。



三、添加实体属性

1) 在上述窗口的“Attribute”选项标签上可以添加属性，如下图所示。



注意:

数据项中的“添加属性”和“重用已有数据项”这两项功能与模型中 Data Item 的 Unique code 和 Allow reuse 选项有关。

P 列表示该属性是否为主标识符;D 列表示该属性是否在图形窗口中显示;M 列表示该属性是否为强制的，即该列是否为空值。

如果一个实体属性为强制的，那么，这个属性在每条记录中都必须被赋值，不能为空。

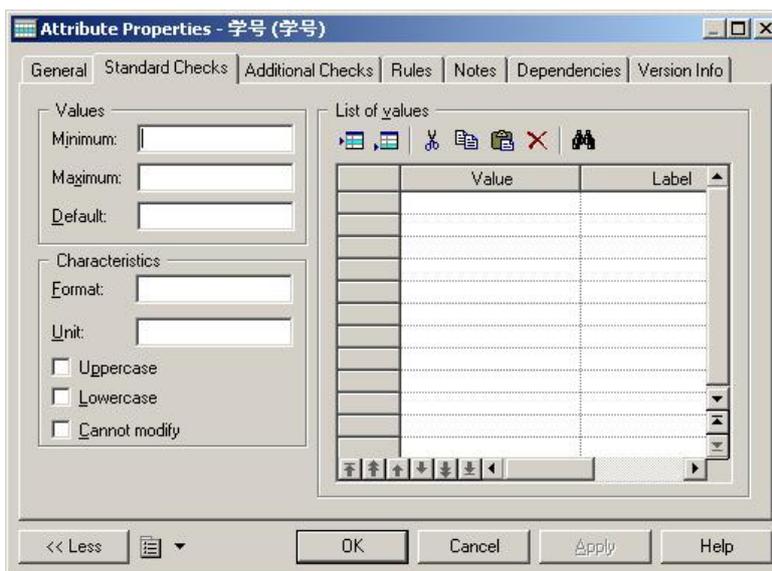
2) 在上图所示窗口中，点击插入属性按钮，弹出属性对话框，如下图所示。



注意：这里涉及到域的概念，即一种标准的数据结构，它可应用至数据项或实体的属性上

一、定义属性的标准检查约束

标准检查约束是一组确保属性有效的表达式。在实体属性的特性窗口，打开如图所示的检查选项卡。



在这个选项卡可以定义属性的标准检查约束，窗口中每项参数的含义，说明如下：

Minimum 属性可接受的最小数；

Maximum 属性可接受的最大数；

Default 属性不赋值时，系统提供的默认值；

Unit 单位，如公里、吨、元；

Format 属性的数据显示格式；

Lowercase 属性的赋值全部变为小写字母；

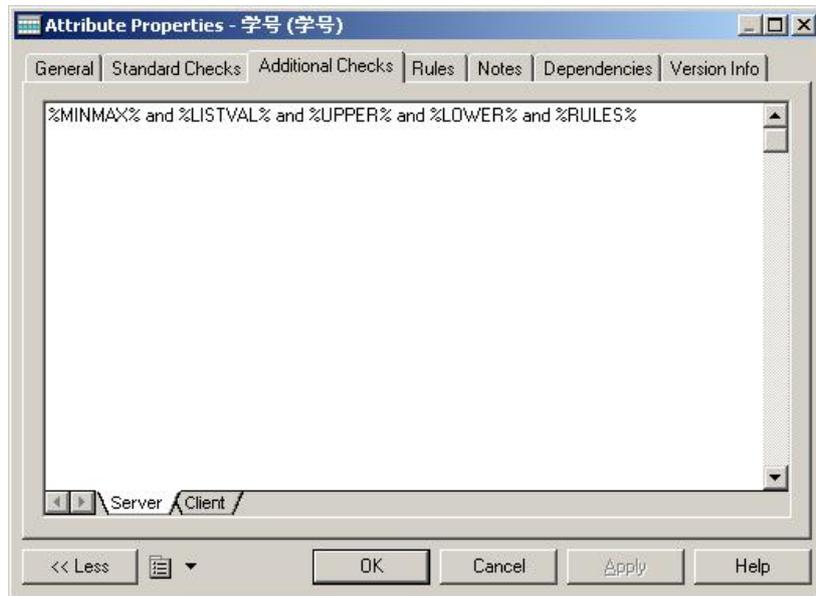
Uppercase 属性的赋值全部变为大写字母；

Cannot modify 该属性一旦赋值不能再修改；

List Of Values 属性赋值列表，除列表中的值，不能有其他的值 **Label** 属性列表值的标签

二、定义属性的附加检查

当 Standard checks 或 Rules 不能满足检查的要求时，可以在 Additional Checks 选项卡的 Server 子页上，通过 SQL 语句中使用%MINMAX%、%LISTVAL%、%RULES%、%UPPER%、%LOWER%几个变量来定义 Standard 和 Rule,如图所示



%MINMAX%、%LISTVAL%、%UPPER%、%LOWER%

在 Standard Check 中定义的 Minimum 和 Maximum、List values 、uppervalues、lowervalues%RULES%

在 Rules 特性窗口 Expression 选项卡中定义的有效性规则表达式

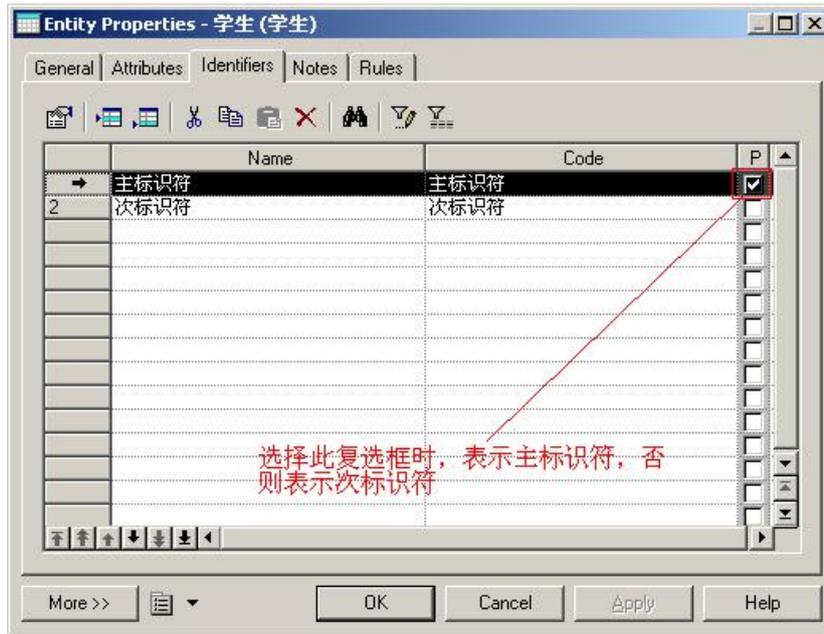
一、标识符

标识符是实体中一个或多个属性的集合，可用来唯一标识实体中的一个实例。要强调的是，CDM 中的标识符等价于 PDM 中的主键或候选键。

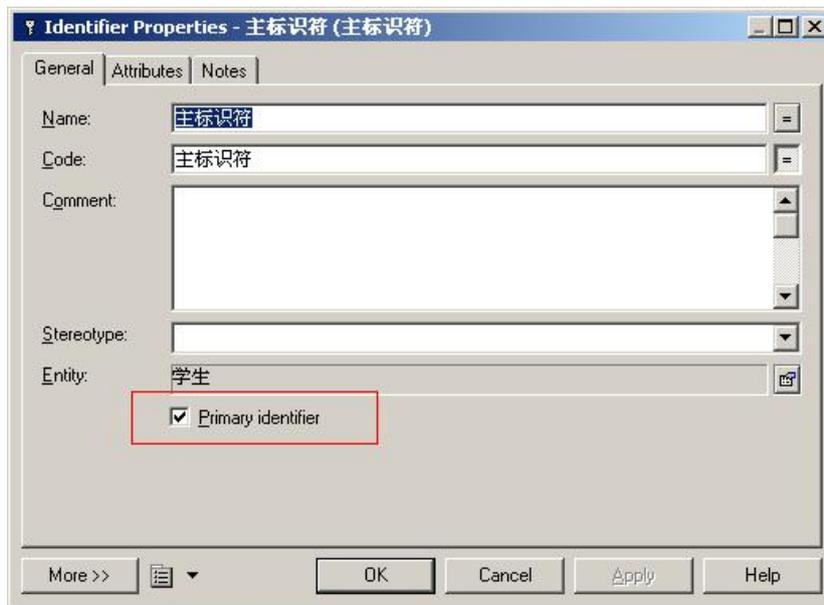
每个实体都必须至少有一个标识符。如果实体只有一个标识符，则它为实体的主标识符。如果实体有多个标识符，则其中一个被指定为主标识符，其余的标识符就是次标识符了。

二、如果定义主、次标识符

1) 选择某个实体双击弹出实体的属性对话框。在 Identifiers 选项卡上可以进行实体标识符的定义。如下图所示



2) 选择第一行“主标识符”，点击属性按钮或双击第一行“主标识符”，弹出属性对话框，如图所示



3) 选择“Attributes”选项卡，再点击“Add Attributes”工具，弹出如图所示窗口，选择某个属性作为标识符就行了。



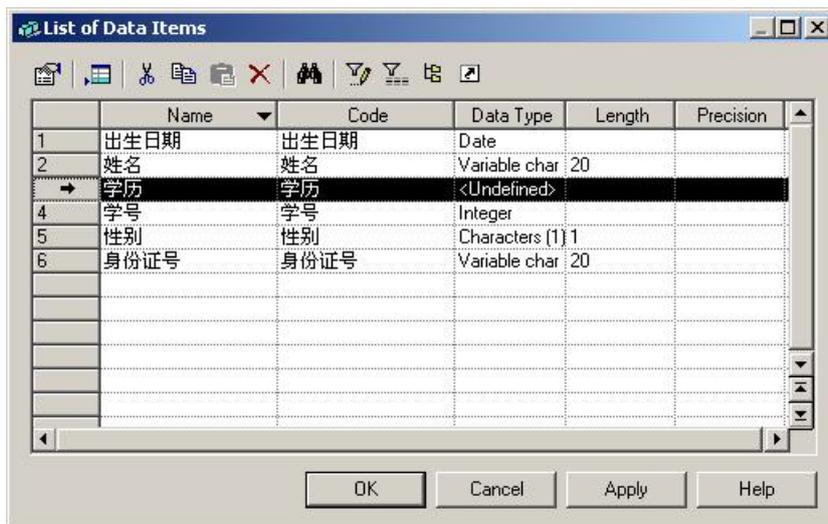
一、数据项

数据项（Data Item）是信息存储的最小单位，它可以附加在实体上作为实体的属性。

注意：模型中允许存在没有附加至任何实体上的数据项。

二、新建数据项

1) 使用“Model”---> Data Items 菜单，在打开的窗口中显示已有的数据项的列表，点击“Add a Row”按钮，创建一个新数据项，如图所示：



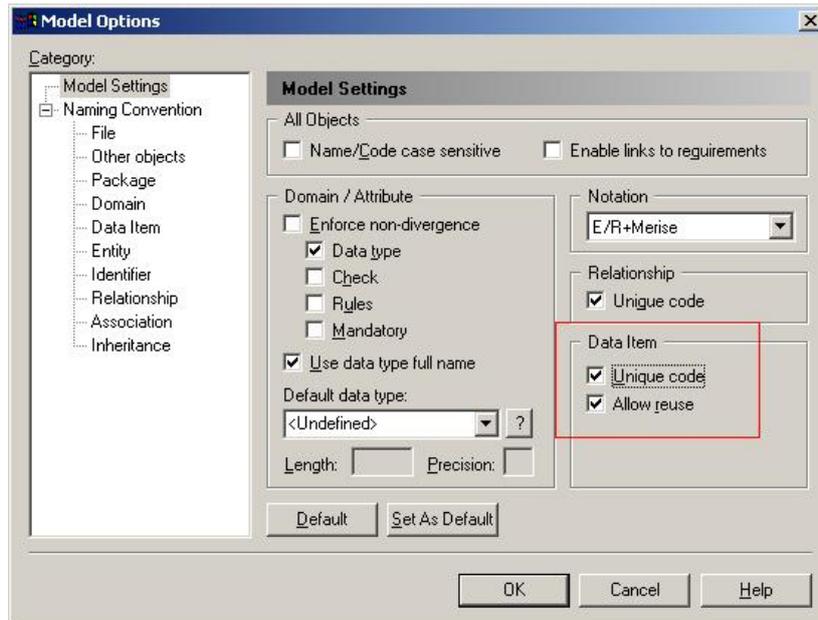
2) 当然您可以继续设置具体数据项的 Code、DataType、Length 等等信息。这里就不再详细说明了。

三、数据项的唯一性代码选项和重用选项

使用 Tools--->Model Options->Model Settings。在 Data Item 组框中定义数据项的唯一性代码选项 (Unique Code)与重用选项 (Allow Reuse)。

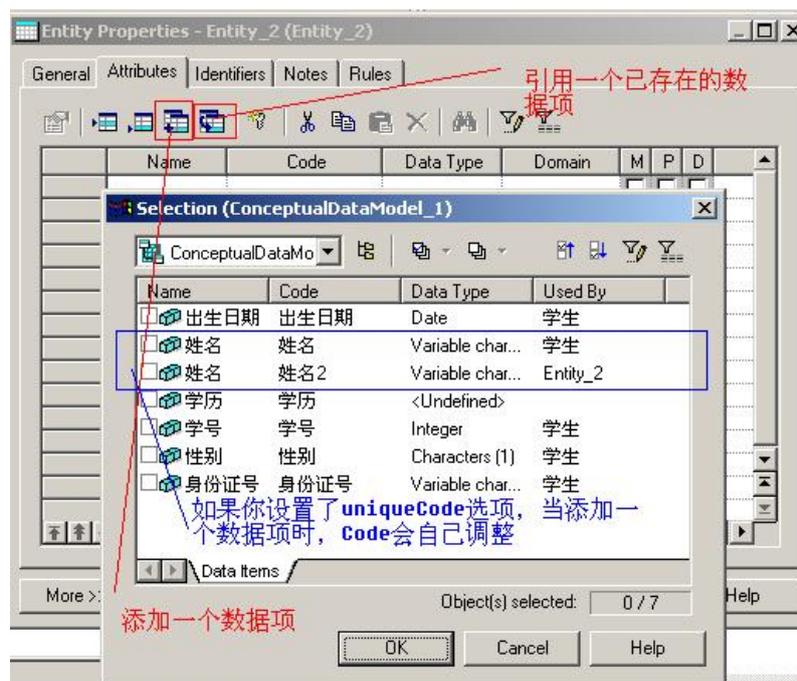
注意：

如果选择 Unique Code 复选框，每个数据项在同一个命名空间有唯一的代码，而选择 Allow reuse，一个数据项可以充当多个实体的属性。



四、在实体中添加数据项

- 1) 双击一个实体符号，打开该实体的属性窗口。
- 2) 单击 **Attributes** 选项卡，打开如下图所示窗口



注意：Add a Dataltem 与 Reuse a Dataltem 的区别在于：

Add a Dataltem 情况下，选择一个已经存在的数据项，系统会自动复制所选择的数据项。如果您设置了 UniqueCode 选项，那系统在复制过程中，新数据项的 Code 会自动生成一个唯一的号码，否则与所选择的数据项完全一致。

Reuse a Dataltem 情况下，只引用不新增，就是引用那些已经存在的数据项，作为新实体的数据项。

一、联系

联系（Relationship）是指实体集这间或实体集内部实例之间的连接。

实体之间可以通过联系来相互关联。与实体和实体集对应，联系也可以分为联系和联系集，联系集是实体集之间的联系，联系是实体之间的联系，联系是具有方向性的。联系和联系集在含义明确的情况之下均可称为联系。

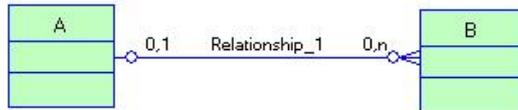
按照实体类型中实例之间的数量对应关系，通常可将联系分为 4 类，即一对一（ONE TO ONE）联系、一对多（ONE TO MANY）联系、多对一（MANY TO ONE）联系和多对多联系（MANY TO MANY）。

二、 建立联系

在 CDM 工具选项板中除了公共的工具外，还包括如下图所示的其它对象产生工具。

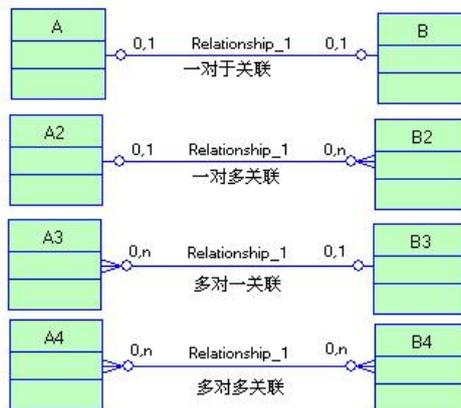


在图形窗口中创建两个实体后，单击“实体间建立联系”工具，单击一个实体，在按下鼠标左键的同时把光标拖至另一个实体上并释放鼠标左键，这样就在两个实体间创建了联系，右键单击图形窗口，释放 Relationship 工具。如下图所示



三、 四种基本的联系

即一对一（ONE TO ONE）联系、一对多（ONE TO MANY）联系、多对一（MANY TO ONE）联系和多对多联系（MANY TO MANY）。如图所示



四种基本的联系

四、 其他几类特殊联系

除了 4 种基本的联系之外，实体集与实体集之间还存在标定联系（Identify Relationship）、非标定联系（Non-Identify Relationship）和递归联系（Recursive Relationship）。

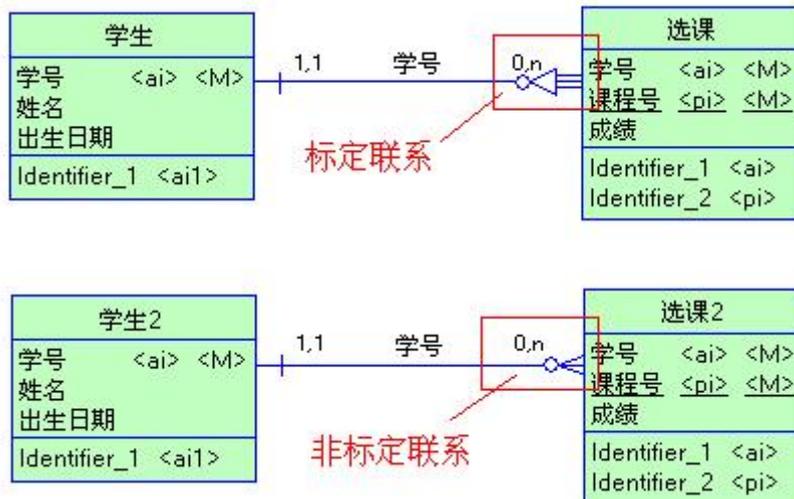
标定联系：

每个实体类型都有自己的标识符，如果两个实体集之间发生联系，其中一个实体类型的标识符进入另一个实体类型并与该实体类型中的标识符共同组成其标识符时，这种联系则称为标定联系，也叫依赖联系。反之称为非标定联系，也叫非依赖联系。

注意：

在非标定联系中，一个实体集中的部分实例依赖于另一个实例集中的实例，在这种依赖联系中，每个实体必须至少有一个标识符。而在标定联系中，一个实体集中的全部实例完全依赖于另一个实例集中的实例，在这种依赖联系中一个实体必须至少有一个标识符，而另一个实体却可以没有自己的标识符。没有标识符的实体用它所依赖的实体的标识符作为自己的标识符。

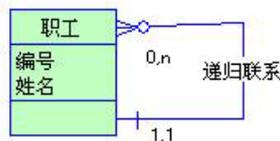
换句话来理解，在标定联系中，一个实体（选课）依赖 一个实体（学生），那么（学生）实体必须至少有一个标识符，而（选课）实体可以没有自己的标识符，没有标识符的实体可以用实体（学生）的标识符作为自己的标识符。



递归联系:

递归联系是实体集内部实例之间的一种联系，通常形象地称为自反联系。同一实体类型中不同实例集之间的联系也称为递归联系。

例如：在“职工”实体集中存在很多的职工，这些职工之间必须存在一种领导与被领导的关系。又如“学生”实体集中的实体包含“班长”子实体集与“普通学生”子实体集，这两个子实体集之间的联系就是一种递归联系。创建递归联系时，只需要单击“实体间建立联系”工具从实体的一部分拖至该实体的别一个部分即可。如图



五、 定义联系的特性

在两个实体间建立了联系后，双击联系线，打开联系特性窗口，如图所示。



六、 定义联系的角色名

在联系的两个方向上各自包含有一个分组框，其中的参数只对这个方向起作用，Role Name 为角色名，描述该方向联系的作用，一般用一个动词或动宾组表。

如：“学生 to 课目 ” 组框中应该填写“拥有”，而在“课目 To 学生”组框中填写“属于”。（在此只是举例说明，可能有些用词不太合理）。

七、 定义联系的强制性

Mandatory 表示这个方向联系的强制关系。选中这个复选框，则在联系线上产生一个联系线垂直的竖线。不选择这个复选框则表示联系这个方向上是可选的，在联系线上产生一个小圆圈。

八、 有关联系的基数

联系具有方向性，每个方向上都有一个基数。

举例，

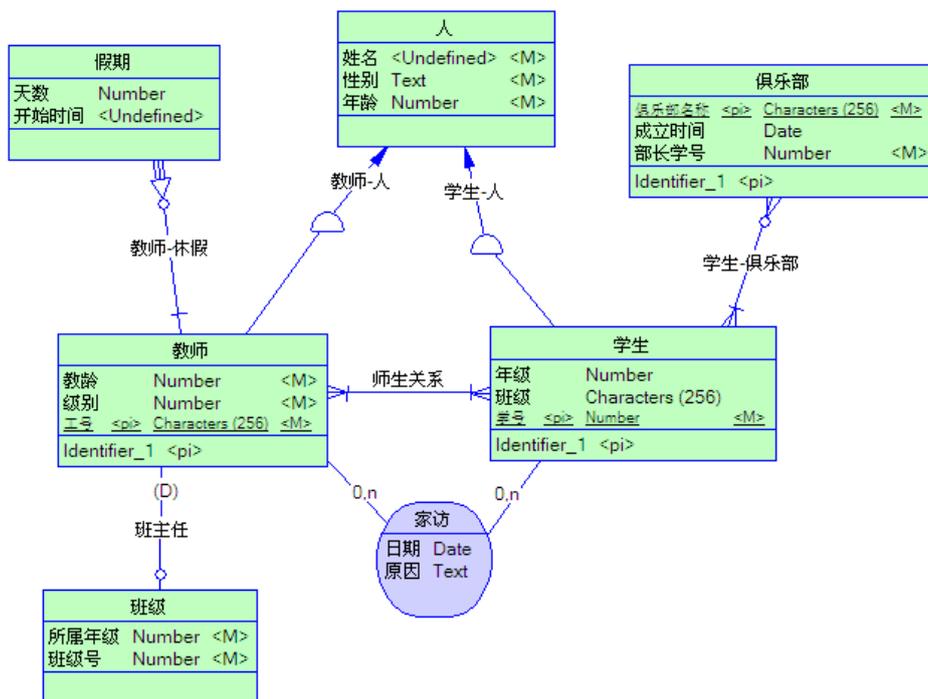
“系”与“学生”两个实体之间的联系是一对多联系，换句话说“学生”和“系”之间的联系是多对一联系。而且一个学生必须属于一个系，并且只能属于一个系，不能属于零个系，所以从“学生”实体至“系”实体的基数为“1,1”，从联系的另一方向考虑，一个系可以拥有多个学生，也可以没有任何学生，即零个学生，所以该方向联系的基数就为“0,n”，如图所示



CDM 是大多数开发者使用 PD 时最先创建的模型，也是整个数据库设计最高层的抽象。CDM 是建立在传统的 ER 图模型理论之上的，ER 图中有三大主要元素：实体型，属性和联系。其中实体型对应到 CDM 中的 Entity，属性对应到 CDM 中每个 Entity 的 Attribute，在概念上基本上是一一对应的。但在联系上，CDM 有了比较大的扩展，除了保留 ER 图原有的 Relationship 概念之外，还增加了 Association, Inheritance 两种实体关系，下面就让我们分别看看这些关系的用法和之间的区别（下图中被标红的工具栏按钮就是用来向实体中添加这些关系的）。



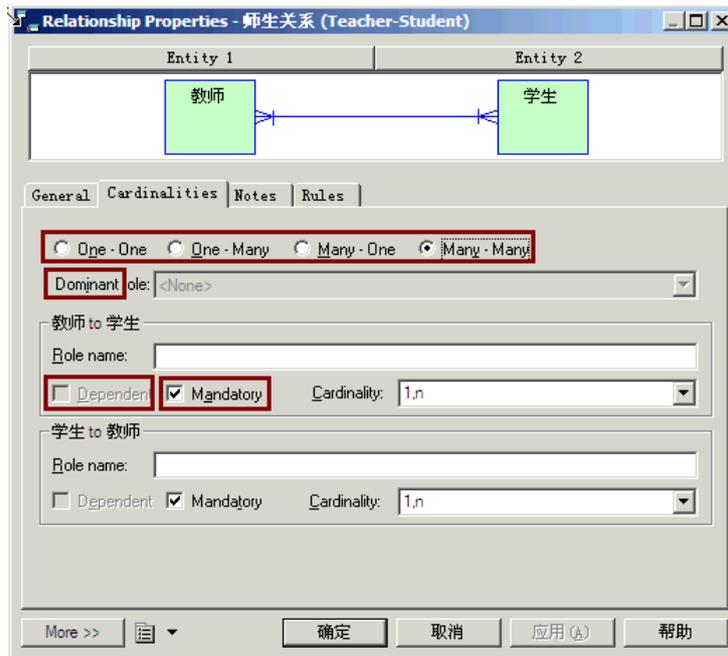
另外，在介绍所有这些 CDM 中的元素之前，笔者先给出一个很简单的 CDM 图，是对我们最最熟悉的学校场景的一个建模，下文中提到的所有概念在图中都有体现，大家在看下文的时候可以对照着来看：



一. Relationship(联系)

先给出 PD 手册里对联系的定义：“A relationship is a link between entities. For example, in a CDM that manages human resources, the relationship Member links the entities Employee and Team, because employees can be members of teams. This relationship expresses that each employee works in a team and that each team has employees.” 可见，也许联系的概念真的太简单了吧，所以反而不那么好表述，所以 PD 的文档里也是用一个例子来说明出现了什么样的情况我们就认为两个实体间是有联系的。

当我们提起实体间联系的时候，最先想到的恐怕是 one to one, one to many 和 many to many 这三种联系类型，这些联系类型也是大家最熟悉的。笔者对 ER 图原本的概念并不精通，但在 CDM 中，联系还有另外三个可以设置的属性：mandatory(强制性联系), dependent(依赖性联系/标定关联) 和 dominant(统制联系)。这些属性对后面 PDM 的生成都有比较大的影响，需要我们一一有所了解。它们都是在联系的属性控制面板中设定的，见下图：



1.mandatory

联系是否具有强制性，指的是实体间是不是一定会出现这种联系；或者换句话说，当我们在谈及一个联系的应用场景的时候，联系对应的那两个实体型的实体实例的个数可不可能为零。也许这样的解释还是有点抽象，让我们举两个联系的例子，一个是对两边的实体都有强制性的，另一个则不然。

(1) 教师--学生 联系

这个联系首先是一个多对多联系，因为每个老师可以教多个学生，每个学生也都有多个老师来负责他们的学业。同时，这个联系对教师和学生都是强制性的，也就是说，不存在任何一个老师，他不负责任何一个学生的教学；也不存在任何一个学生，他没有任何一个任课老师。

(2) 学生--俱乐部 联系

这个联系也是一个多对多关系，但它对学生这个实体型而言就不是强制的（Optional,可选的）。每个俱乐部都有至少一个学生参加，但并不是每个学生都要去参加俱乐部的活动。完全可以有一些学生，他们什么俱乐部都没参加。

上面的例子主要是从概念的角度来区分了 mandatory 和 optional 的区别。实际上如果把这个模型对应到我们最后生成的表，如果 A-B 间的联系对 A 是 mandatory 的话，那么如果在 A 里面如果包含 B 的外键，这个外键不能为空值，反之可以为空值。后面我们谈到 PDM 和实际数据库的时候，大家会看到这一点。

2.dependent

每一个 Entity 型都有自己的 Identifier，如果两个 Entity 型之间发生关联时，其中一个 Entity 型的 Identifier 进入另一个 Entity 型并与该 Entity 型中的 Identifier 共同组成其 Identifier 时，这种关联称为标定关联,也叫依赖性关联(dependent relationship)。一个 Entity 型的 Identifier 进入另一个 Entity 型后充当其非 Identifier 时，这种关联称为非标定关联,也叫非依赖关联。

概念的定义说起来还是有些拗口，说白了其实就是主-从表关系，从表要依赖于主表。比如在我们系统里要记录教师休假的情况，有一个实体型 Holiday，其属性包括休假的开始时间和天数，每次有教师休假的时候，都要在这个表留下记录。从我们的场景描述中可以看到，实体型假期必须依附于实体型教师，即对于每一个假期实例，必须指向某一个教师实例。

对于依赖型联系，必须注意它不可能是一个多对多联系，在这个联系中，必须有一个作为主体的实体型。一个 dependent 联系的从实体可以没有自己的 identifier。

3.dominant

上图中所有标红的部分是我们最应该关注的内容，因为他们都是由于我们对实体型间的关系的定义而产生的，下面给出一些简单的说明。

1. “师生关系”和“学生俱乐部”这两个表是由于我们的多对多关系而产生的。
2. “假期”表的“工号”字段是由于我们将教师-假期关系指定为 **dependent** 而产生的。
3. “班级”表的“工号”字段是由于我们将教师-班级关系制定为 **dominant** 而产生的。
4. “家访”表中的“工号”和“学号”字段是由于家访是教师和学生实体型的 **association** 而产生的。

另外，记得我们在提到 **dominant** 属性的时候说过，一个没指定 **dominant** 方向的一对一联系将产生两个引用，下面我们就把原本的 CDM 中的教师-班级关系进行一个小小的修改，去掉这个 **relationship** 的 **dominant** 定义，那么最终产生的 PDM 中教师表和班级表将互相包含对方的主键(由于我们的班级表没有自己的主键，所以只能在班级表中看到多出来的列)，截图如下：

