

\*\*\*计算机科学学院 711 实验室

# PowerDesigner15.1 使用教程

图文并茂详解 PowerDesigner15.1 各模块功能的使用

幽灵柯南技术Blog: <http://enetq.blog.51cto.com/>

## 目录

1 PowerDesigner 简介.....	3
1.1 powerDesigner 建模工具简介 .....	3
1.2 PowerDesigner 主要包括以下几个功能部分 .....	4
1.3 PowerDesigner 的 5 种模型文件 .....	5
1.3.1 概念数据模型 (CDM) .....	5
1.3.2 逻辑数据模型 (LDM) .....	6
1.3.3 物理数据模型 (PDM) .....	6
1.3.4 面向对象模型 (OOM) .....	6
1.3.5 业务程序模型 (BPM) .....	7
1.4 模型区别 .....	7
1.4.1 对象转换.....	7
1.4.2 其它对比.....	7
2 概念模型设计.....	7
2.1 实体 .....	8
2.2 关系 .....	12
2.2.3 继承.....	15
2.2.4 关联 (Association) .....	16
3. 物理模型设计 1——表和主外键.....	18
3.1 主键 .....	20
3.2 外键.....	22
4. 物理模型设计 2——约束.....	24
4.1 唯一约束 .....	24
4.2 CHECK 约束.....	25
4.3 默认约束 .....	31
5. 物理模型设计 3——视图、存储过程和函数.....	32
5.1 视图 .....	32
5.2 存储过程和函数 .....	34
6. 定义和使用域 (Domain) .....	36
6.1 关于域 .....	36
7. 将 CDM 对象转换成 PDM 对象 .....	38
8. 使用物理数据模型创建数据库脚本 .....	42
9. 逆向工程.....	44
10. 生成数据库报表文件 .....	45

## 1 PowerDesigner 简介

PowerDesigner 是 Sybase 公司的 CASE 工具集，使用它可以方便地对管理信息系统进行分析设计，它几乎包括了数据库模型设计的全过程。利用 PowerDesigner 可以制作数据流程图、概念数据模型、物理数据模型，可以生成多种客户端开发工具的应用程序，还可为数据仓库制作结构模型，也能对团队设备模型进行控制。它可与许多流行的数据库设计软件，例如：PowerBuilder, Delphi, VB 等相配合使用来缩短开发时间和使系统设计更优化。

### 1.1 powerDesigner 建模工具简介

提高软件质量，缩短开发周期，并且使软件更能够适应业务需求的变化，以提高投资回报率，是每个企业所面临的、需要解决的关键问题。软件建模一直被认为是提高与有效控制软件质量的解决之道。近些年来为大家关注的主要是数据设计模型、对象模型、和业务流程模型。由于历史原因，面向数据架构，开发以及业务分析的建模工作总是被单独购买，彼此之间没有集成或共享信息。但是，企业不断需要更集成的建模套件，即集成化企业级建模工具，来支持在共享环境下，企业整个架构的不同方面的全面建模。

目前各主要的建模工具厂商如 Sybase PowerDesigner, IBM Rational Rose, Computer Associates 的 ERWin 等都在加强各自建模工具的融合与集成。PowerDesigner 经过近 20 年的发展，已经在原有的数据建模的基础上，形成一套完整的集成化企业级建模解决方案（如图所示）



图1：集成化企业级建模解决方案

PowerDesigner 系列产品提供了一个完整的建模解决方案，业务或系统分析人员，设计人员，数据库管理员 DBA 和开发人员可以对其裁剪以满足他们的特定的需要；而其模块化的结构为购买和扩展提供了极大的灵活性，从而使开发单位可以根据其项目的规模和范围来使用他们所需要的工具。PowerDesigner 灵活的分析 and 设计特性允许使用一种结构化的方法有效地创建数据库或数据仓库，而不要求严格遵循一个特定的方法学。PowerDesigner 提供了直观的符号表示使数据库的创建更加容易，并使项目

组内的交流和通讯标准化，同时能更加简单地向非技术人员展示数据库和应用的设计。

PowerDesigner 不仅加速了开发的过程，也向最终用户提供了管理和访问项目的信息的一个有效的结构。它允许设计人员不仅创建和管理数据的结构，而且开发和利用数据的结构针对领先的开发工具环境快速地生成应用对象和数据敏感的组件。开发人员可以使用同样的物理数据模型查看数据库的结构和整理文档，以及生成应用对象和在开发过程中使用的组件。应用对象生成有助于在整个开发生命周期提供更多的控制和更高的生产率。

PowerDesigner 是一个功能强大而使用简单工具集，提供了一个复杂的交互环境，支持开发生命周期的所有阶段，从处理流程建模到对象和组件的生成。PowerDesigner 产生的模型和应用可以不断地增长，适应并随着你的组织的变化而变化。

PowerDesigner 包含六个紧密集成的模块，允许个人和开发组的成员以合算的方式最好地满足他们的需要。这六个模块是：

1. PowerDesigner ProcessAnalyst，用于数据发现。
2. PowerDesigner DataArchitect，用于双层，交互式的数据库设计和构造。
3. PowerDesigner AppModeler，用于物理建模和应用对象及数据敏感组件的生成。
4. PowerDesigner MetaWorks，用于高级的团队开发，信息的共享和模型的管理。
5. PowerDesigner WarehouseArchitect，用于数据仓库的设计和实现。
6. PowerDesigner Viewer，用于以只读的、图形化方式访问整个企业的模型信息。

## 1.2 PowerDesigner 主要包括以下几个功能部分

### (1) DataArchitect

这是一个强大的数据库设计工具，使用 DataArchitect 可利用实体-关系图为一个信息系统

创建“概念数据模型”-CDM (Conceptual Data Model)。并且可根据 CDM 产生基于某一特定

数据库管理系统（例如：Sybase System 11）的“物理数据模型”-PDM(Physical Data Model)。

还可优化 PDM，产生为特定 DBMS 创建数据库的 SQL 语句并可以文件形式存储以便在其他时

刻运行这些 SQL 语句创建数据库。另外，DataArchitect 还可根据已存在的数据库反向生成

PDM, CDM 及创建数据库的 SQL 脚本。

## (2) ProcessAnalyst

这部分用于创建功能模型和数据流图，创建“处理层次关系”。

## (3) AppModeler

为客户/服务器应用程序创建应用模型。

## (4) ODBC Administrator

此部分用来管理系统的各种数据源。

## (5) XML 文件

可以使用工具生成相应的 XML 文件，或者用 XML 文件生成相关模型

# 1.3 PowerDesigner 的 5 种模型文件

## 1.3.1 概念数据模型 (CDM)

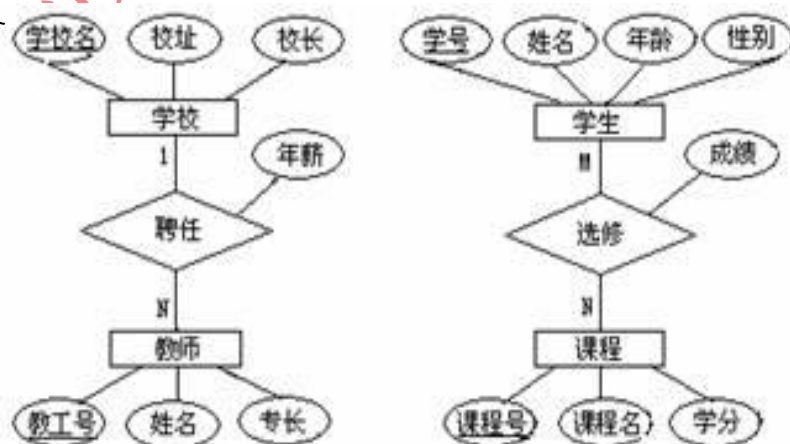
CDM 表现数据库的全部逻辑的结构, 与任何的软件或数据储藏结构无关。一个概念模型经常包括在物理数据库中仍然不实现的数据对象。它给运行计划或业务活动的数据库一个正式表现方式。

概念模型是对真实世界中问题域内的事物的描述, 不是对软件设计的描述。表示概念模型最常用的是“实体-关系”图。

E-R 图主要是由实体、属性和关系三个要素构成的。在 E-R 图中, 使用了下面几种基本的图形符号。

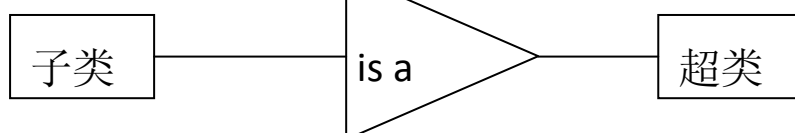
E/R 图三要素

- 实体, 矩形
- 属性, 椭圆形
- 关系, 菱形



关系：一对一关系，一对多关系，多对多关系。

E/R 图中的子类(实体)。



### 1.3.2 逻辑数据模型 (LDM)

在后期的 PowerDesigner 15 中，又引入了新的模型 如 LDM，多维数据模型等 而逻辑模型是概念模型的延伸，表示概念之间的逻辑次序，是一个属于方法层次的模型。具体来说，逻辑模型中一方面显示了实体、实体的属性和实体之间的关系，另一方面又将继承、实体关系中的引用等在实体的属性中进行展示。逻辑模型介于概念模型和物理模型之间，具有物理模型方面的特性，在概念模型中的多对多关系，在逻辑模型中将会以增加中间实体的一对多关系的方式来实现。

逻辑模型主要是使得整个概念模型更易于理解，同时又不依赖于具体的数据库实现，使用逻辑模型可以生成针对具体数据库管理系统的物理模型。逻辑模型并不是在整个步骤中必须的，可以直接通过概念模型来生成物理模型。

逻辑数据模型反映的是系统分析设计人员对数据存储的观点，是对概念数据模型进一步的分解和细化。

### 1.3.3 物理数据模型 (PDM)

PDM 叙述数据库的物理实现。藉由 PDM，你考虑真实的物理实现的细节。它进入帐户两个软件或数据储藏结构之内拿。你能修正 PDM 适合你的表现或物理约束。

物理模型是对真实数据库的描述。数据库中的一些对象如下：表，视图，字段，数据类型、长度、主键、外键、索引、是否可为空，默认值。

概念模型到物理模型的转换即是把概念模型中的对象转换成物理模型的对象。

### 1.3.4 面向对象模型 (OOM)

一个 OOM 包含一系列包，类，接口，和他们的关系。这些对象一起形成所有的（或部份）一个软件系统的逻辑的设计视图的类结构。一个 OOM 本质上是软件系统的一个静态的概念模型。

你使用 PowerDesigner 面向对象模型建立面向对象模型.(OOM) 你能纯粹地对象- 导向的靠模切目的建立一个 OOM，产生 Java 文件或者



PowerBuilder 文件,或你能使用一个来自 OOM 的物理数据模型 (PDM) 对象 , 来表示关系数据库设计分析。

### 1.3.5 业务程序模型 (BPM)

BPM 描述业务的各种不同内在任务和内在流程, 而且客户如何以这些任务和流程互相影响。

BPM 是从业务合伙人的观点来看业务逻辑和规则的概念模型, 使用一个图表描述程序, 流程, 信息和合作协议之间的交互作用。

信息流模型 (ILM)

ILM 是一个高层的信息流模型, 主要用于分布式数据库之间的数据复制。

## 1.4 模型区别

### 1.4.1 对象转换

对象	概念模型	逻辑模型	物理模型
实体	实体	实体	表
属性	属性	属性	字段
关系	关系(一对一, 一对多, 多对一)	关系	外键
关系	关系(一对多, 多对一)	实体	表(关系表) 例如订单和产品的关系是一对多, 这种关系确定为订单产品明细表。
关系	关系(多对多)	实体	表(关系表)

### 1.4.2 其它对比

对比项	概念模型	逻辑模型	物理模型
属性	不需完整定义实体属性	定义实体完整的属性	确定字段名, 长度, 数据类型, 是否可以为空, 初始值等
主键	无需确定主键	无需确定主键	确定主键

## 2 概念模型设计

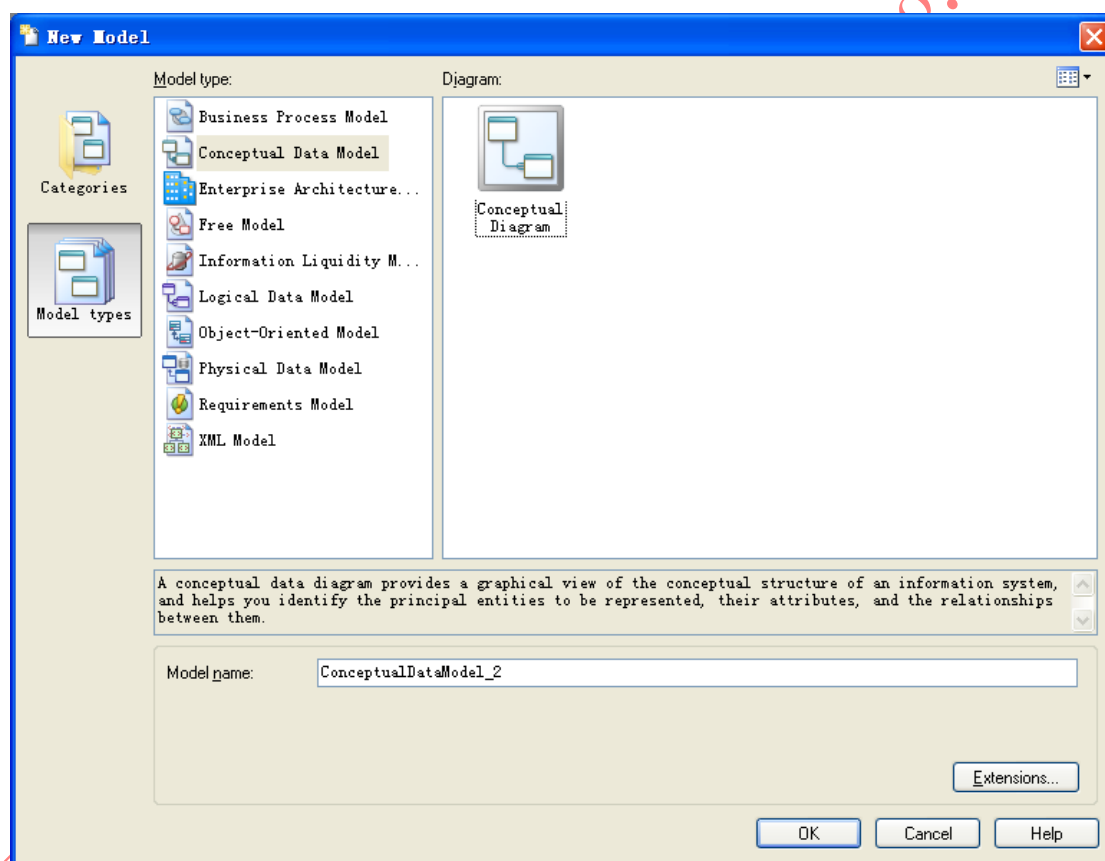
在概念模型中主要有以下几个操作和设置的对象: 实体 (Entity)、实体属性 (Attribute)、实体标识 (Identifiers)、关系 (Relationship)、继承 (Inheritance)、关联 (Association)、关联连接 (Association Link)。

## 2.1 实体

实体就是我们抽象出的对象，比如：一个选课系统中，学生、教师、班级、课程等等都是实体。实体属性就是一个实体中所包含的简单属性，比如学生实体，具有学号、姓名、生日、性别等属性。另外实体还有个标识用于唯一的标识出每个实体实例，比如学生实体中就可将学生的学号作为该实体的标识。标识可用是一个实体属性也可以是多个属性的结合。

在 PD 中新建一个新的概念模型如下：

1.



系统将出现一个工具栏如下，用于在设计面板中设计模型。

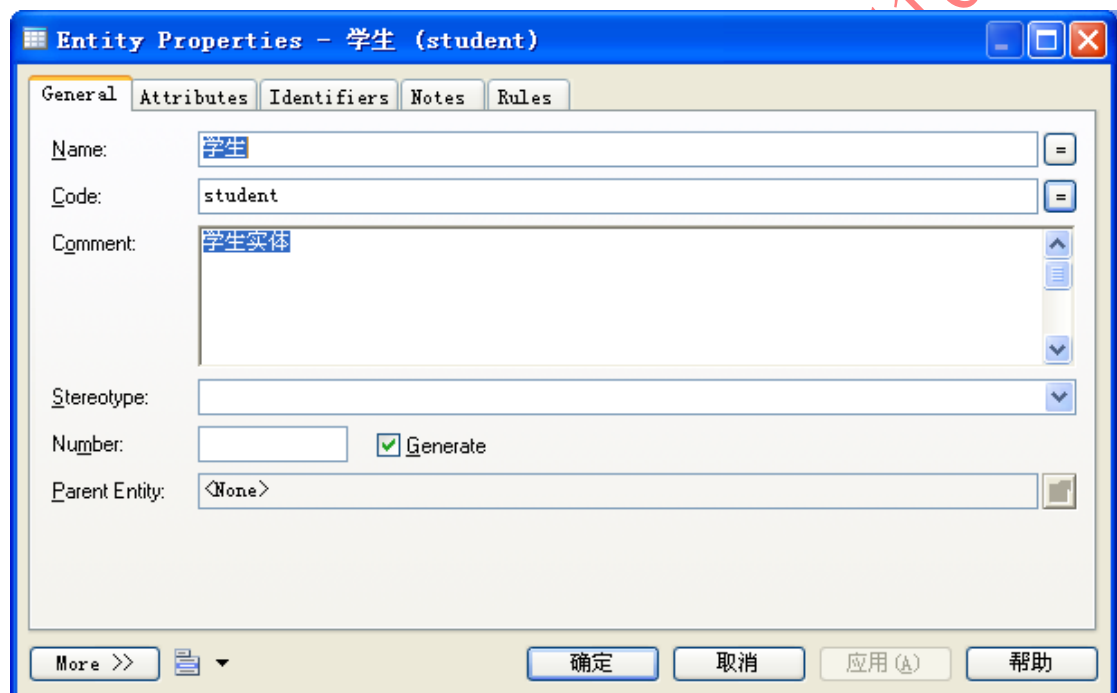


单击 Entity 图标，然后在设计主面板中单击一次便可添加一个实体。

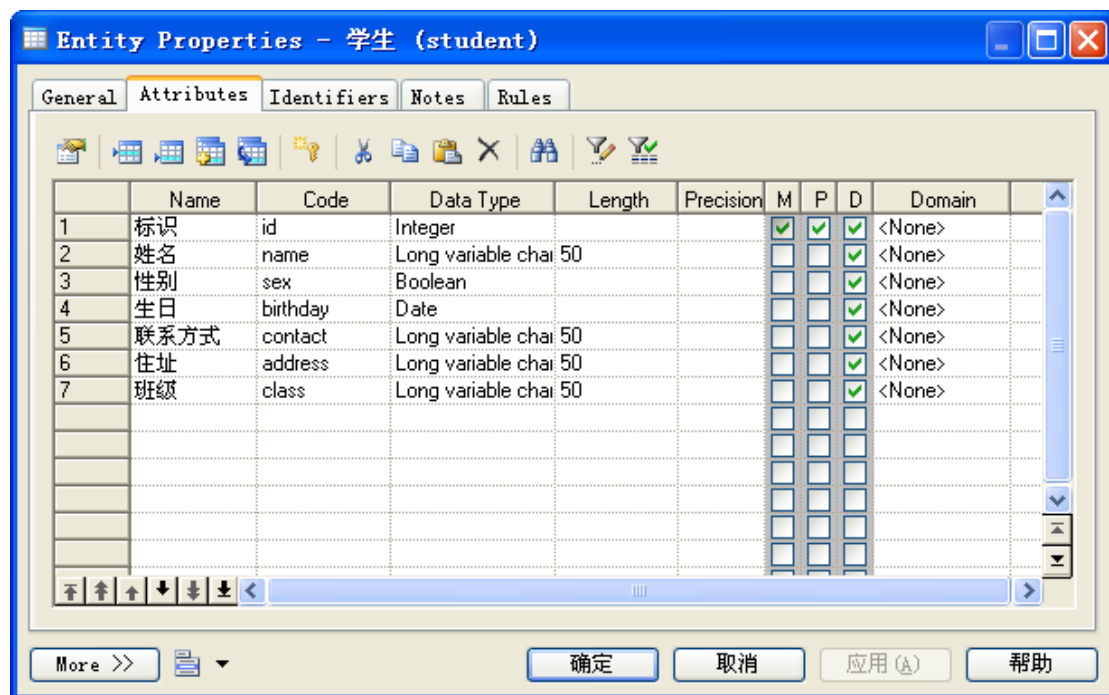




2. 再单击鼠标图标，即可切换回一般鼠标的模式。双击已经添加的实体，弹出实体属性设置对话框，在 General 中可以设置实体的 Name 和 Code 等属性，如图所示：CODE 是字段在数据库中的实际名称，一般用英文 NAME 是显示的名称，一般用中文，方便理解。

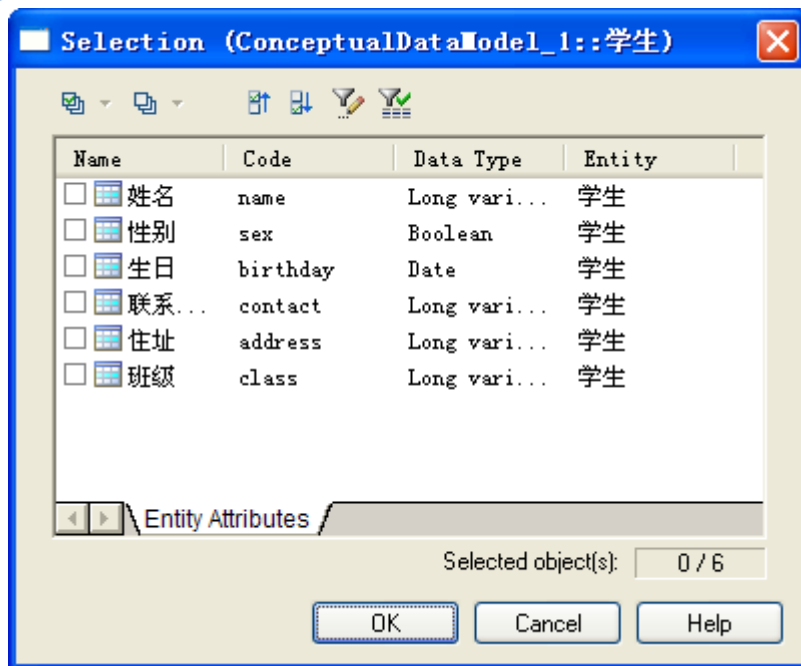


3. 切换到 Attributes 选项卡可以设置实体的属性 Name、Code、DataType



等，右边还有 3 个复选框，M 表示不能为空，P 表示是标识属性，D 表示在模型图中是否显示，如果在设置属性时直接选中 StudentID 的 P 复选框，系统将会自动生成该 Student 实体的一个 Identifier。如果不希望系统自动生成而是手动设置的话，那么切换到 Identifiers 选项卡，添加一行 Identifier，然后单击左上角的“属性”按钮，然后弹出的标识属性设置对话框中单击“添加行”按钮，选择该标识中使用的属性。例如将学号设置为学生实体的标识，具体操作如图所示！





设置完毕之后如图：

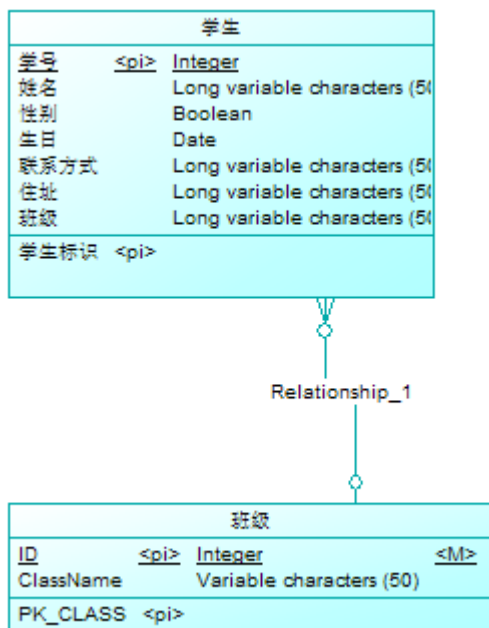
学生			
学号	<pi>	Integer	<M>
姓名		Long variable characters (50)	
性别		Boolean	
生日		Date	
联系方式		Long variable characters (50)	
住址		Long variable characters (50)	
班级		Long variable characters (50)	
学生标识	<pi>		

## 2.2 关系

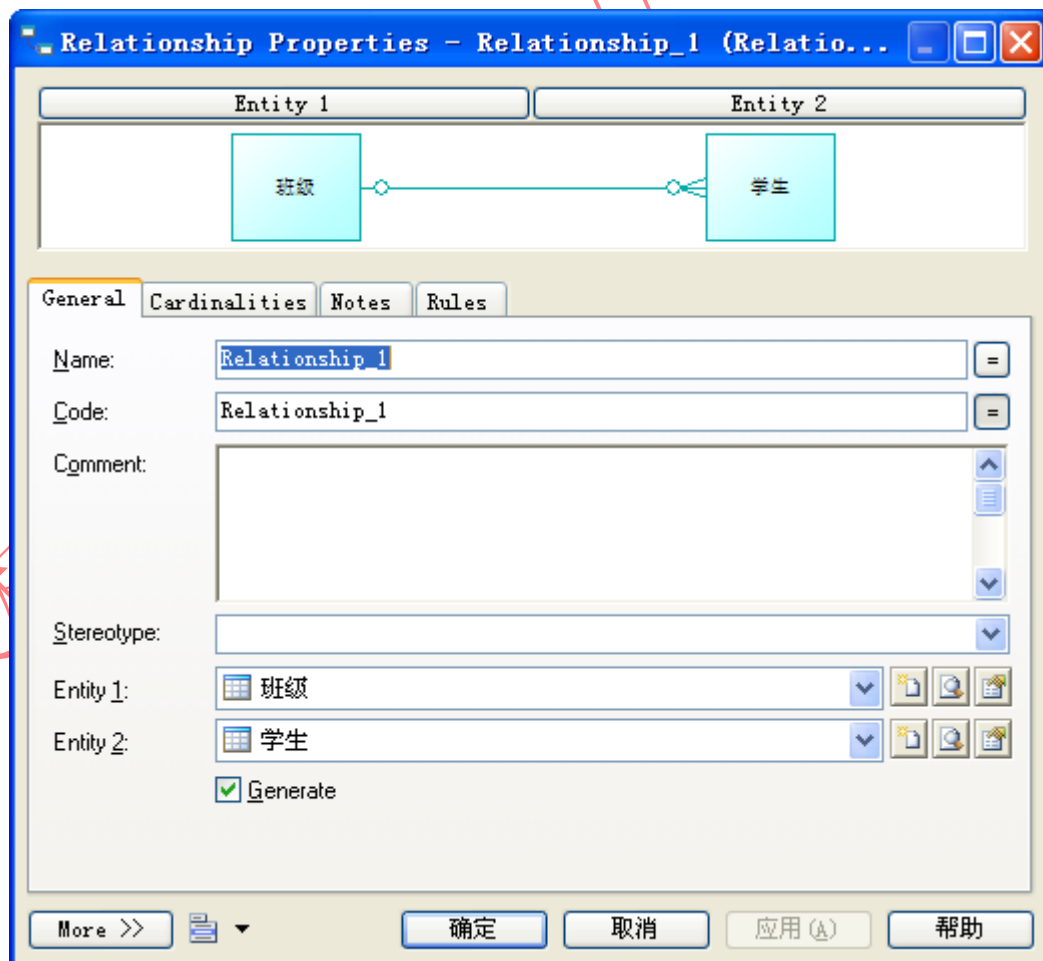
关系用于表示一个实体与另外一个实体之间的对应关系，分为一对一(1:1)，一对多(1:n)，多对一(n:1)，多对多(m:n) 4种对应关系，一对多和多对一其实就是方向相反，所以实际上就是一对一、一对多和多对多。最最常用的就是一对多，比如班级实体和学生实体就是一对多关系，一个班级有多个学生，一个学生只会属于一个班级。在设计面板中添加一个Class 实体，然后单击工具栏的Relationship 按钮，然后在Class 实体上单击一下，再到Student 上单击一下，就可以在Class 和Student 之间建立一对多关系，如图：



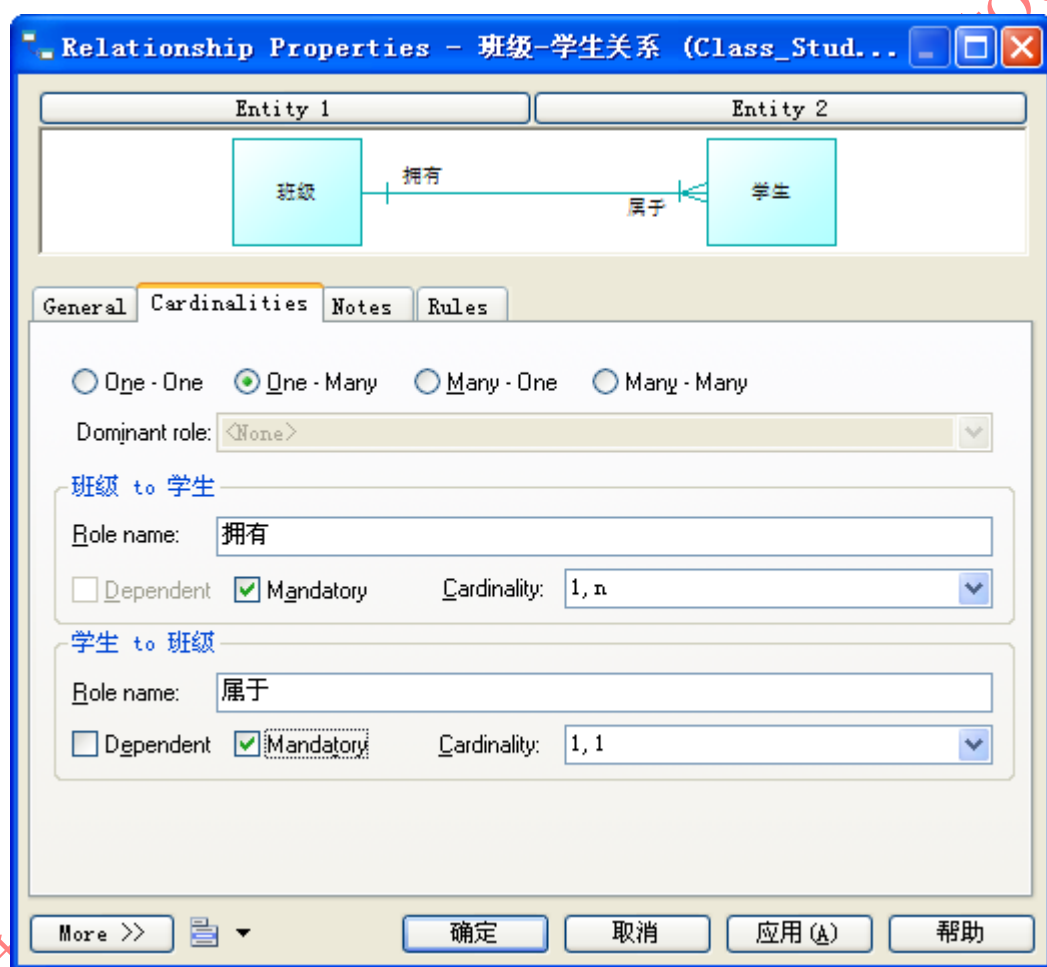
生成如图：



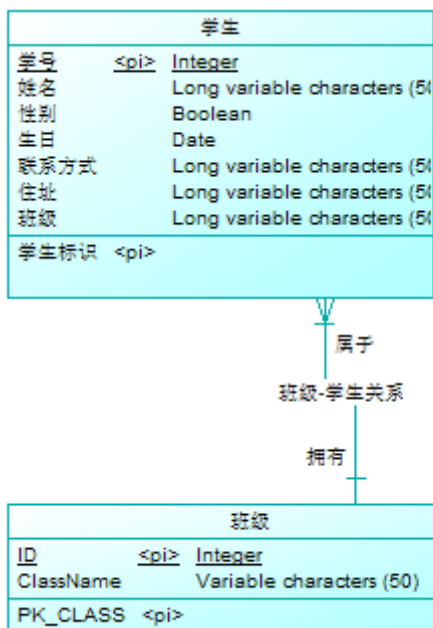
系统默认会给该关系一个命名 Relationship\_1，切换到鼠标指针模式，双击 Relationship\_1，即可打开关系的属性窗口，可以在 General 选项卡中



切换到 Cardinalities 选项卡：上面可以修改是一对多还是多对多之类的关系，下面的 Role name 没有多大的意义，就是在模型中显示文字，多的一方有 0, n 和 1, n 两种，也就是说一个班级可以对应多个学生，那么一个班级中最少是没有学生还是要至少存在一个学生，同样的一的一方有 0, 1 和 1, 1 两种，就是说一个学生是可以不属于任何班级呢还是必须属于某一个存在的班级，这里我们都选至少是 1，所以最终的设置界面如图：



点击确定后生成如图：



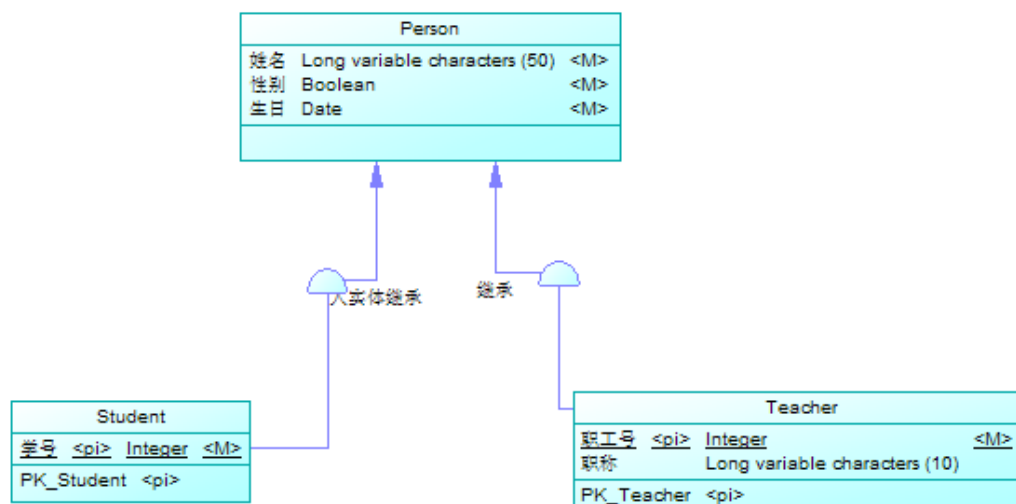
### 2.2.3 继承

在概念模型中的继承与在OO 模型中的继承是一回事，就是要抽象出一个实体，其他实体继承该实体后就拥有该实体的属性。同样以选课系统为例，现在有学生实体，有教师实体，其他他们都是人，具有人的属性，所以我们可以抽象出一个人的实体，人具有性别、生日、名字等属性，教师具有职称、工号等属性这是学生没有的，所以不能放在人实体中，学生具有学号属性是教师没有的。



从学生实体到人实体如图：

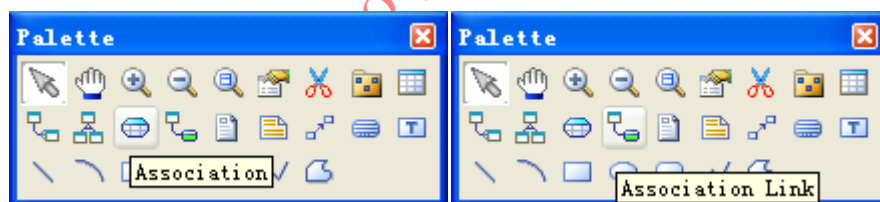




这里只是概念模型，在DBMS 中是没有继承这种说法的，所以在接下来的逻辑模型和物理模型中，系统就会将继承转换为实际的实体和表。这里只是概念模型，所以才有继承的说法。

#### 2.2.4 关联 (Association)

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。

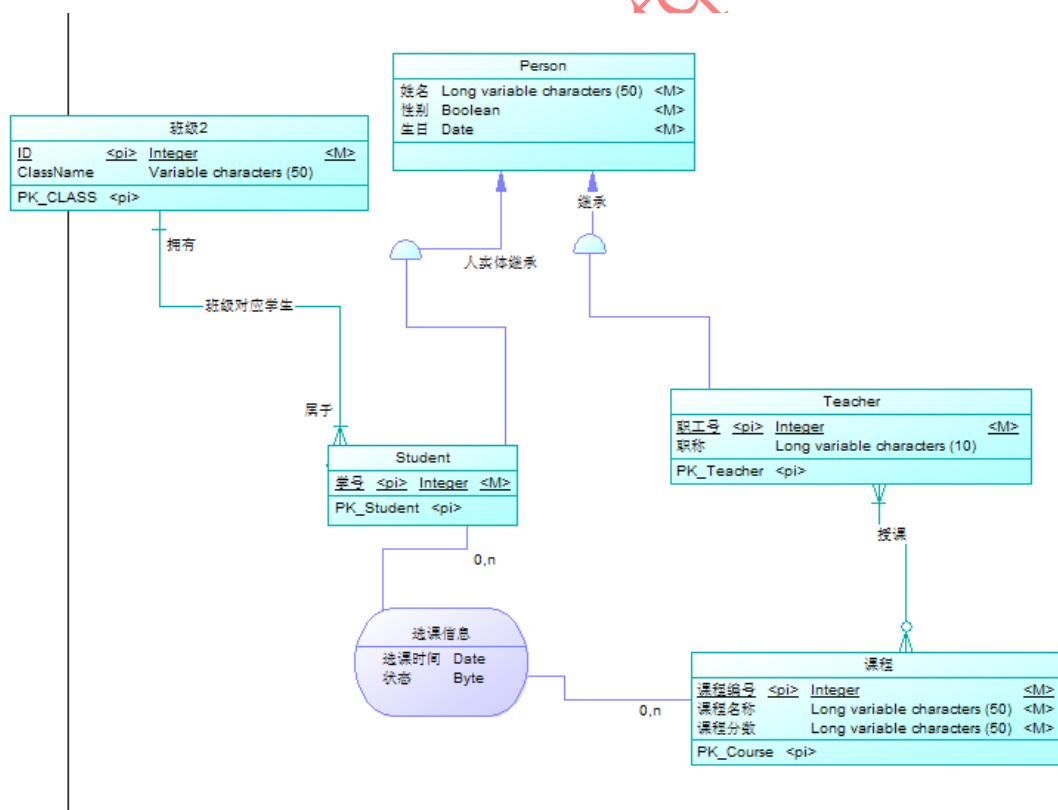


也就是说，实体和实体之间存在着关系（多对多），但是这种关系还存在其他的属性，这些属性如果作为一个明确的实体的实体来表示又不是很合适，所以就使用了 Association 来表达，这种关系之间一般是一个“事件”虚实体，也就是说是一个动词对应的实体。前面说的可能还是太抽象。

以实际的例子来说明：现在有了学生实体，有课



程实体，一个学生可以选择多门课程，一门课程有多个学生来上课，所以之间就存在一个“选课”的 Association，其中记录了学生选课的时间、选课的状态：一个学生会对多个选课结果选择多门课程，一个课程对应多个选课学生选课，所以学生实体和课程实体与选课的关系都是一对多，添加关联连接，一门课程可能太枯燥了，没有学生来选，所以课程对应选课可能是 0, n，一个学生可能学分够了这学期一门课都不选，所以学生对选课也可能是 0, n 的关系，所以我们最终形成的概念模型如图所示：



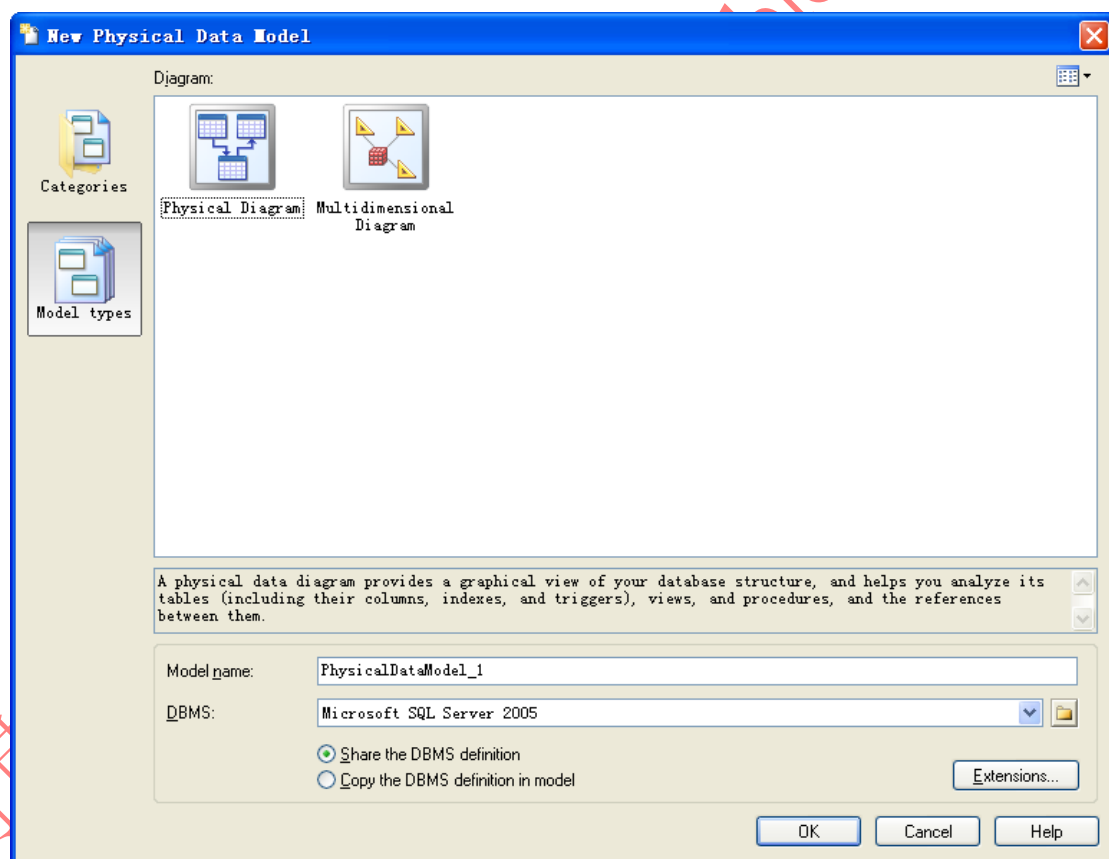
### 3. 物理模型设计 1——表 and 主外键

在PD 中建立物理模型由以下几种办法：

1. 直接新建物理模型。
2. 设计好概念模型，然后由概念模型生成物理模型。
3. 设计好逻辑模型，然后由逻辑模型生成物理模型。
4. 使用逆向工程的方法，连接到现有的数据库，由数据库生成物理模型。

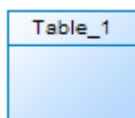
物理模型能够直观的反应出当前数据库的结构。在数据库中的表、视图、存储过程等数据库对象都可以在物理模型中进行设计。由于物理模型和数据库的一致性，接下来以数据库对象和物理模型对象的对应来一一介绍：

新建物理模型时需要指定物理模型对应的DBMS，这里我们使用SQL Server 2005，新建一个物理模型后，系统会显示一个专门用于物理模型设计的工具栏如图所示：



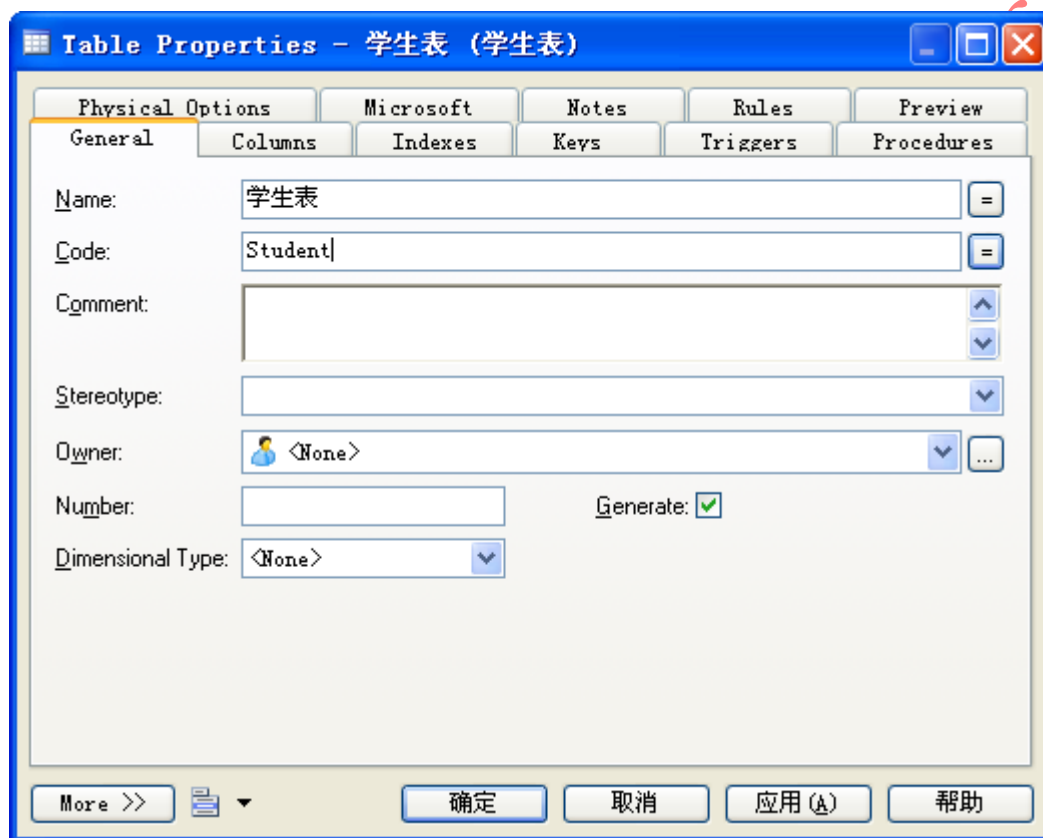
若要在物理模型中添加一个表，单击“表”按钮，然后再到模型设计面板中单击一次便可添加一个表，系统默认为表命名为Table\_n，这里的n 会随着添加

的表增多而顺序增加。添加的表是没有任何列的，如图所示：



单击工具栏的鼠标指针按钮，将鼠标切换回指针模式，然后双击一个表，系统将打开表属性窗口，在General 选项卡中可以设置表的Name、Code 等属性。

例如我们要新建一个学生表（Student），则可修改 Name 和 Code。



Name是在模型中显示的名称，Code 是生成数据库表的时候的实际表名。另外Name 中的内容还会作为SQL Server 中的表备注。单击Columns 切换到列选项卡，在下面的列表中可以添加表中的列。Name是模型上显示的名称，Code 是生成的实际的表名，后面的3 个复选框P 代办主键、F 代表外键，M 代表不能为空。为教室表设计了两个列，如图所示：



### 3.1 主键

在设计一个表时，一般情况下每个表都会有一个主键。在为表设置主键时有以下几种办法：

1. 在Columns 选项卡中，直接选中主键列的P 列复选框。
2. 选中一个列，然后单击工具栏中的“属性”按钮。在该窗口中可以设置该列的各种属性，当然也包括有一个很重要的复选框是“Identity”。选中Identity 复选框，则该列将自动成为主键列。

1. 在Columns 选项卡中，直接选中主键列的P 列复选框，这是最简单的方式。

2. 选中一个列，然后单击工具栏中的“属性”按钮，系统将弹出列属性窗口，在该窗口中可以设置该列的各种属性，当然也包括该列是否是主键。另外还有一个很重要的复选框是“Identity”。选中Identity 复选框则表示该列为自增列。



3. 切换到Keys 选项卡中，在其中添加一行命名为PK\_Student，然后单击工具栏的“属性”按钮，打开键属性窗口，在该窗口中切换到Columns 选项卡，单击添加列按钮，弹出列选择窗口，选中主键中应该包含的列，单击确定按钮即可完成主键的创建。另外需要注意的是，在建立主键时，系统会在主键上建立索引，索引分为聚集索引和非聚集索引，在“键属性”窗口的General 选项卡中可以设置该主键上建立的索引是聚集索引还是非聚集索引，如图所示：

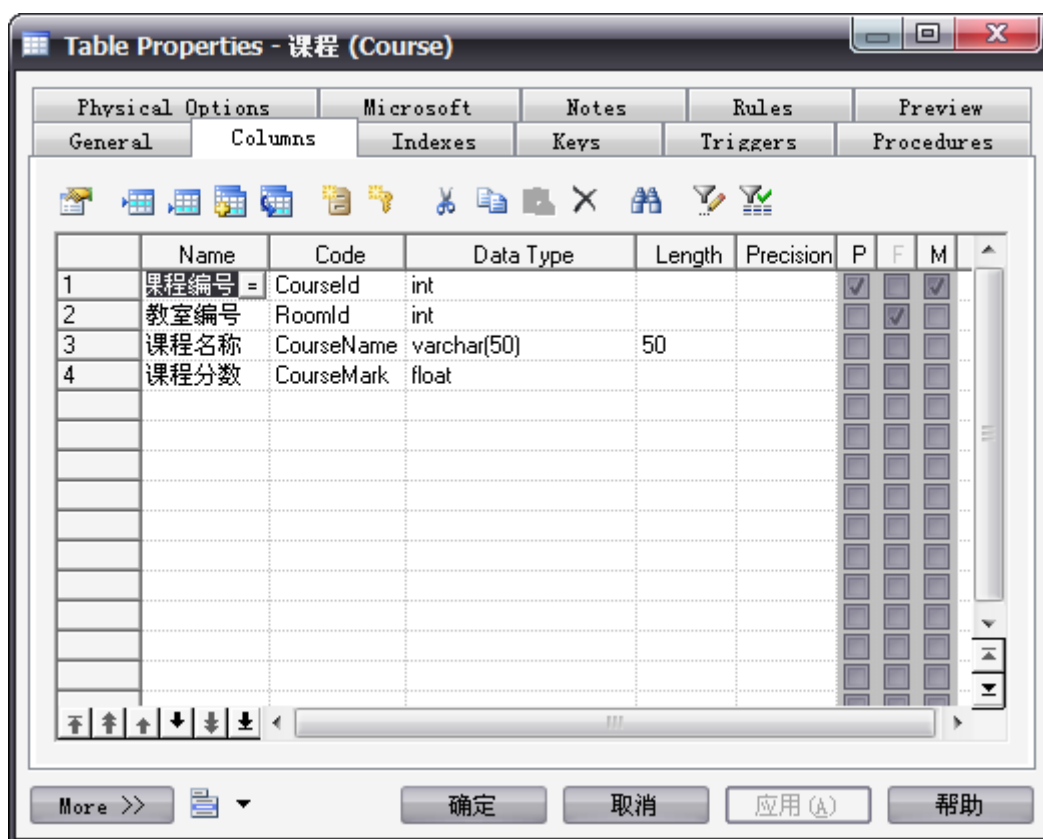


### 3.2 外键

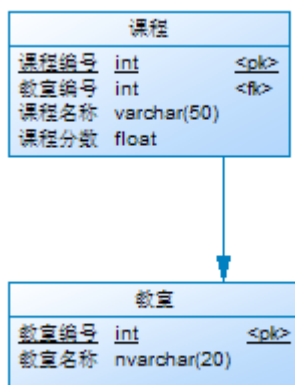
如果是由概念模型或者逻辑模型生成物理模型，那么外键是通过 Relationship 生成的，也可以通过工具栏中的 Reference 来实现两表之间的外键关系。假如一个课程只会在一个固定的教室上课，而一个教室会安排多个课程在不同的时间上课，所以教室和课程是一对多的关系，那么课程表中就需要添加 RoomID 列以形成外键列，具体操作方法就是在工具栏中单击“Reference”按钮，然后在设计面板中，课程表上按下鼠标左键，并拖拽到教师表中放开鼠标，这时如果课程表中没有 RoomID 列，系统会自动创建 RoomID 列并创建该列上的外键



引用，如果已经存在 RoomID 列，则只添加外键引用，不会再添加新列。



生成如图所示:



切换到鼠标指针模式，双击箭头，系统将弹出引用的属性窗口，在属性窗口中可以设置该引用的Name、Code、关联的列、约束名、更新策略和删除策略

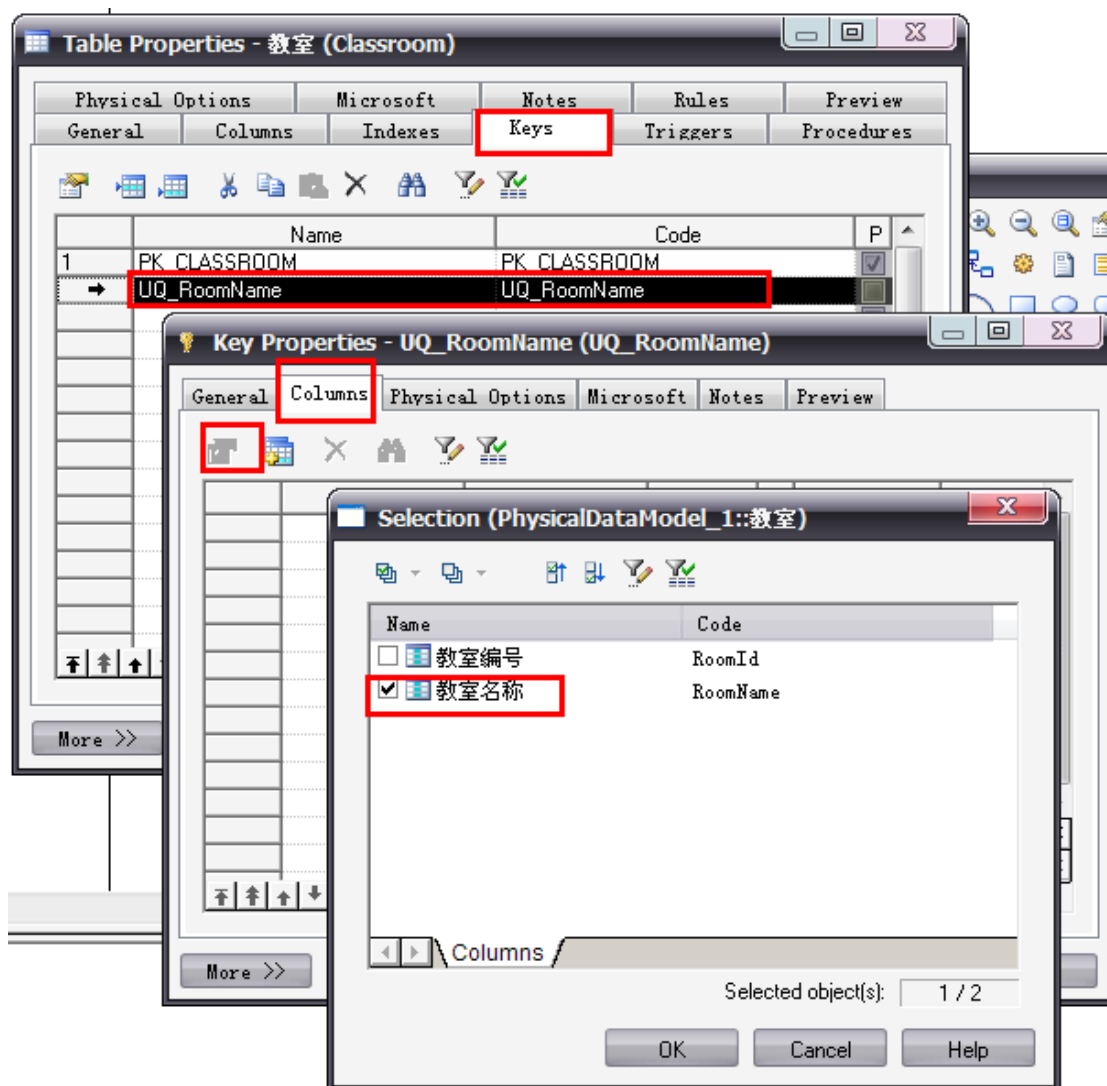
等。

## 4. 物理模型设计 2——约束

### 4.1 唯一约束

唯一约束与创建唯一索引基本上是一回事，因为在创建唯一约束的时候，系统会创建对应的一个唯一索引，通过唯一索引来实现约束。不过唯一约束更直观的表达了对应列的唯一性，使得对应索引的目的更加清晰，所以一般建议创建唯一约束而不是只创建唯一索引。在PD 中创建唯一约束的操作，以教室表来说，RoomID 是主键，必然是唯一的，RoomName 如果我们也要去必须是唯一的，那么具体操作如下：

在PD 的模型设计面板中，双击“教室”表，打开属性窗口，切换到“Keys”选项卡，可以看到里面有一行数据PK\_ClassRoom，这是主键约束。添加一行数据，命名为UQ\_RoomName，不能将右边的“P”列选上，然后单击工具栏的“属性”按钮，弹出UQ\_RoomName 的属性窗口，切换到列选项卡，单击增加列按钮，选择将RoomName 列添加到其中，然后单击确定即可完成唯一约束的添加。



这样系统就会自动创建唯一约束。

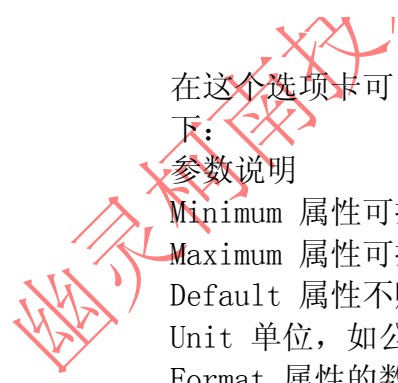
## 4.2 CHECK 约束

CHECK 分为列约束和表约束，列约束是只对表中的某一个列进行的约束可以在列的属性中进行设置，而表约束是对多个列进行的约束，需要在表的属性中进行设置（其实列约束也可以在表约束中设置）。

### 1. 标准 CHECK 约束

对于一些常用的 CHECK 约束，可以直接通过设置界面来完成。以班级表为例，ClassName 每个学校有自己的命名规则，假设这里规定 ClassName 必须以 2 开头，那么需要在 ClassName 列上定义 CHECK 约束，使得其满足命名规范。具

**ml**



## 参数说明

Maximum 属性可接受的最大数

Unit 单位, 如公里、吨、元

Lowercase 属性的赋值全部变为小写字母

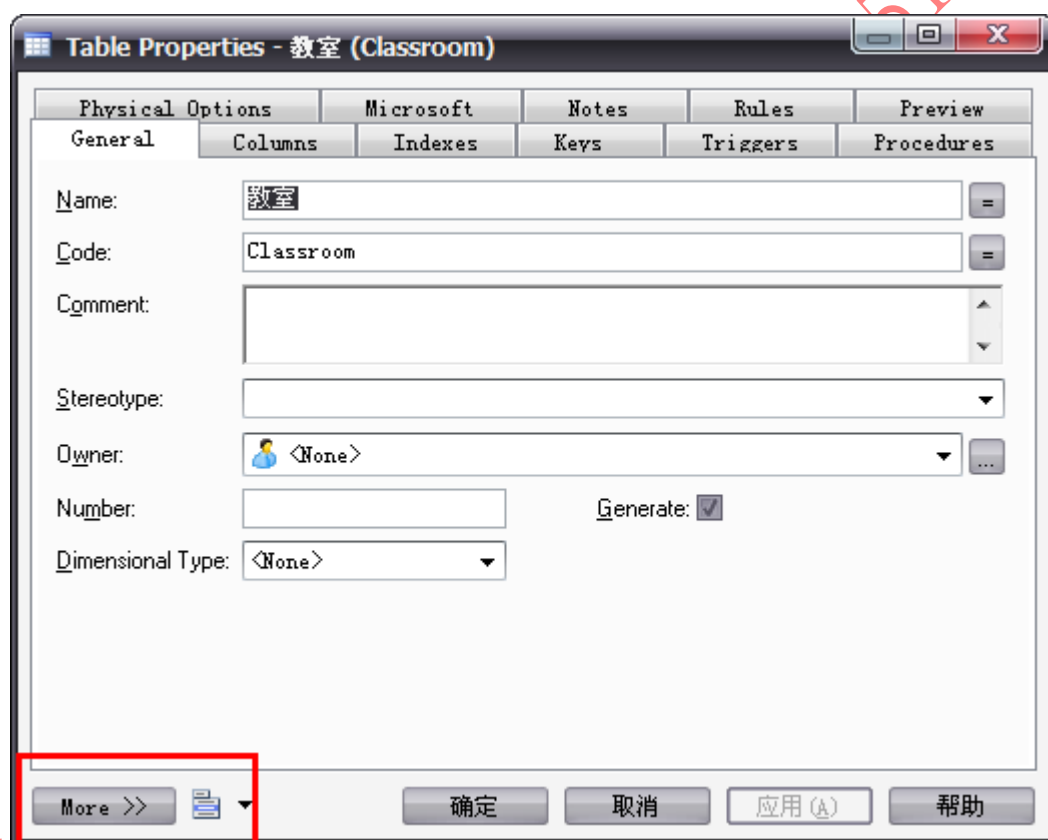
Cannot modify 该属性一旦赋值不能再修改

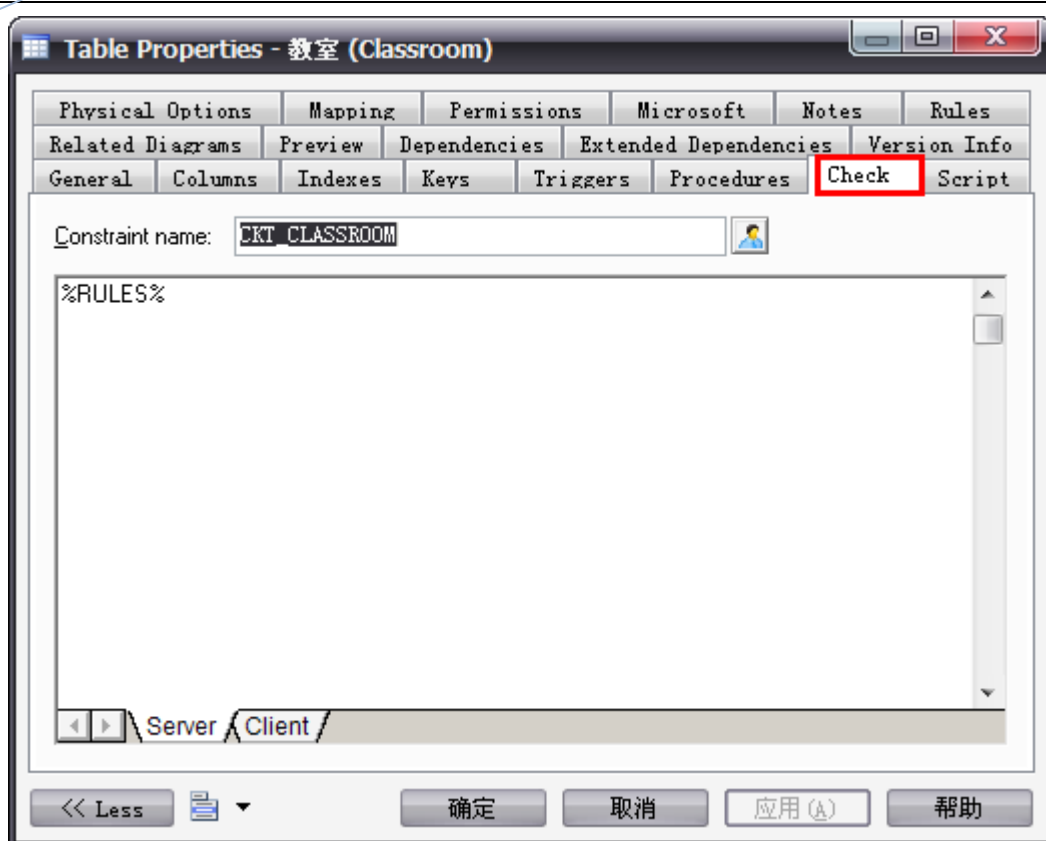
List Of Values 属性赋值列表，除列表中的值，不能有其他的值

Label 属性列表值的标签

## 2. 直接编写 SQL 语句的 CHECK 约束

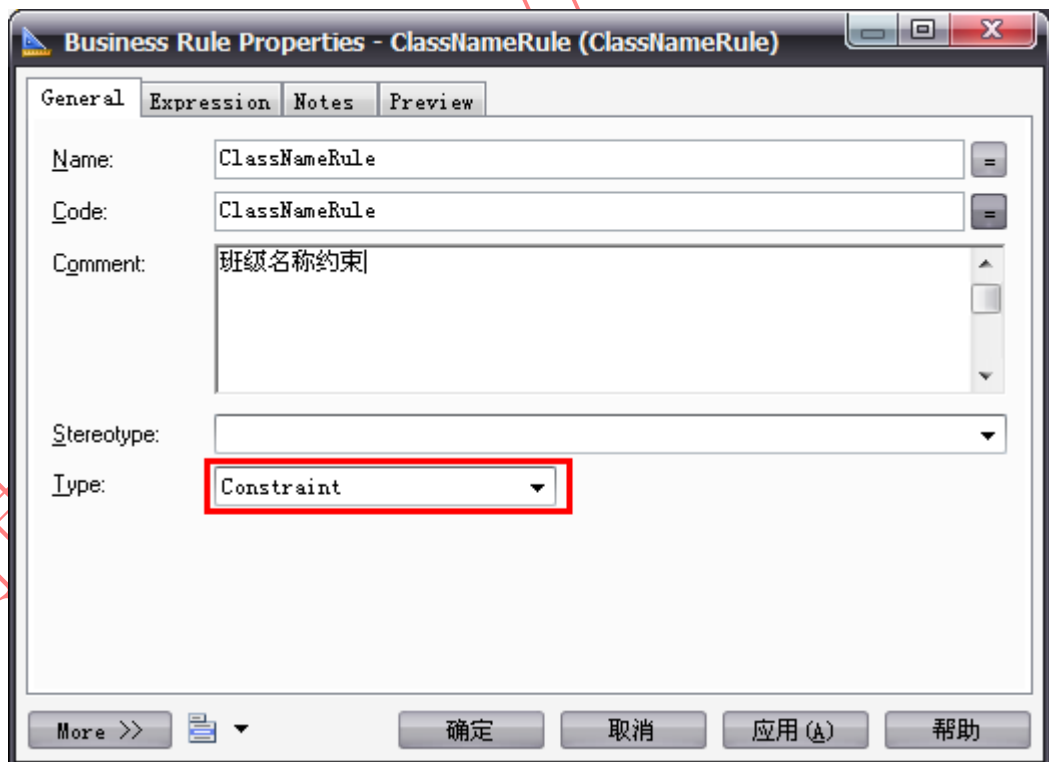
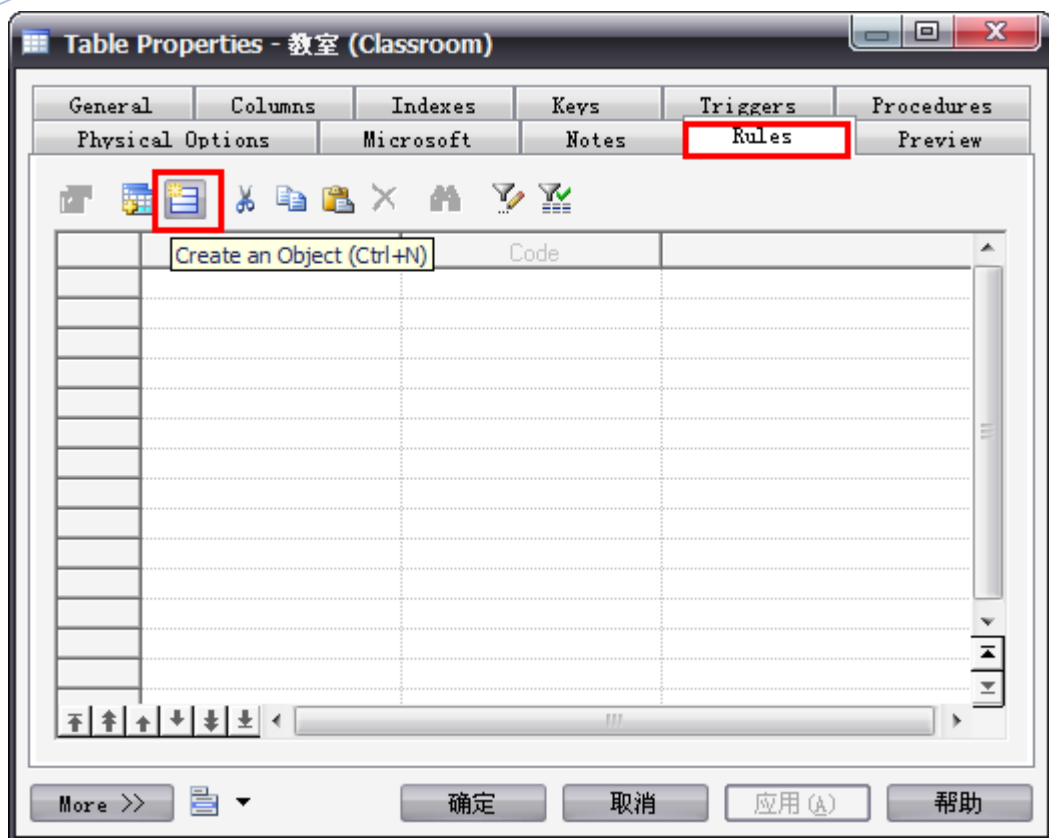
在前面弹出 ClassName 属性窗口中，单击左下角的“More”按钮，系统将弹出更多的选项卡，切换到“Additional Checks”选项卡，可以设置约束名和具体的约束内容，如图表级的 CHECK 约束与列级的 CHECK 约束设置类似，单击表属性窗口左下角的“More”按钮，切换到 Check 选项卡，设置 CHECK 约束的命名和 SQL 语句内容。





### 3. 使用 Rule 创建约束

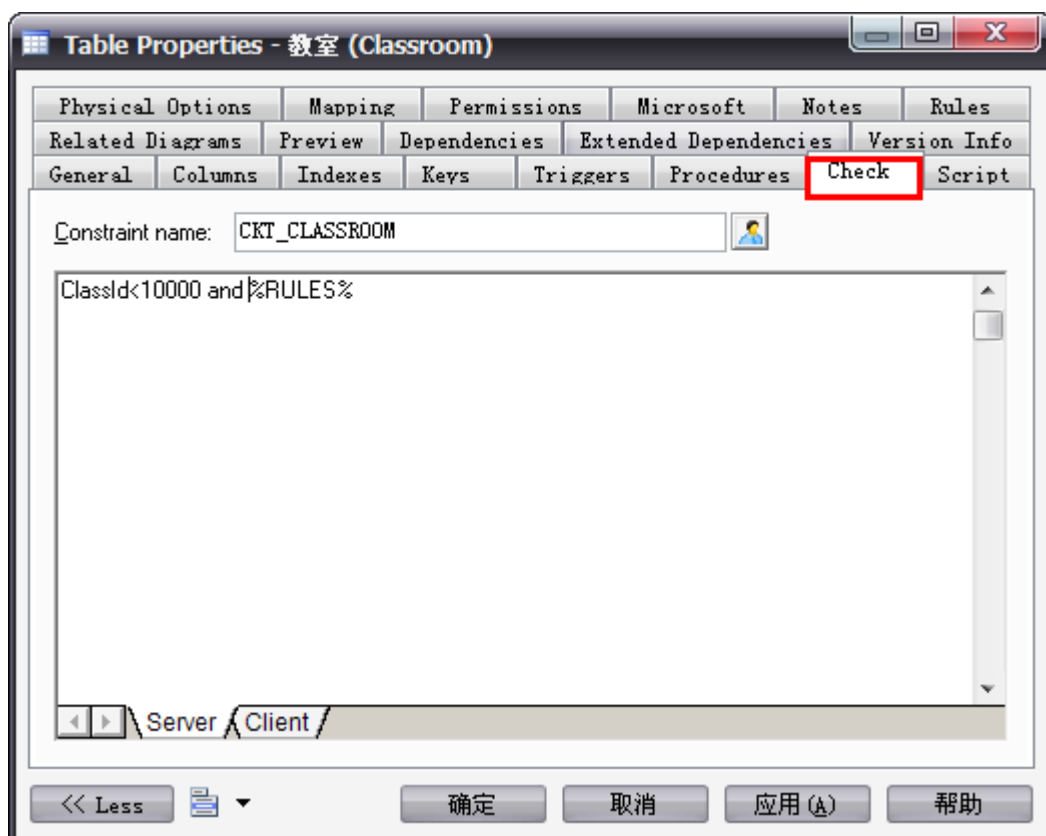
同样以班级名必须以 2 开头为例，通过 Rule 创建 CHECK 约束。首先需要创建一个 Rule，双击 Class 表，打开表的属性窗口，切换到 Rules 选项卡，单击“Create a Object”按钮，系统将打开一个业务规则属性窗口，修改规则名，并将规则的类型修改为 Constraint，如图所示：



然后切换到 Expression 选项卡，设置规则的内容为 “ClassName LIKE  
'2%' ”，单击确定按钮即可完成 Rule 的设置。切换到表属性的 Check 选项卡，



默认约束内容中的“%RULES%”就是用来表示 Rule 中设置的内容，如果我们还有一些其他的 CHECK 约束内容，不希望在 Rule 中设置，而是在 Check 选项卡中设置，那么只需要删除%RULES%将 CHECK 约束内容添加进去，也可以留%RULES%，然后在与%RULES%之间添加一个 and 即可。比如规定 ClassID 必须小于 10000，那么我们可以将 Check 内容设置如下：



生成的脚本如下：

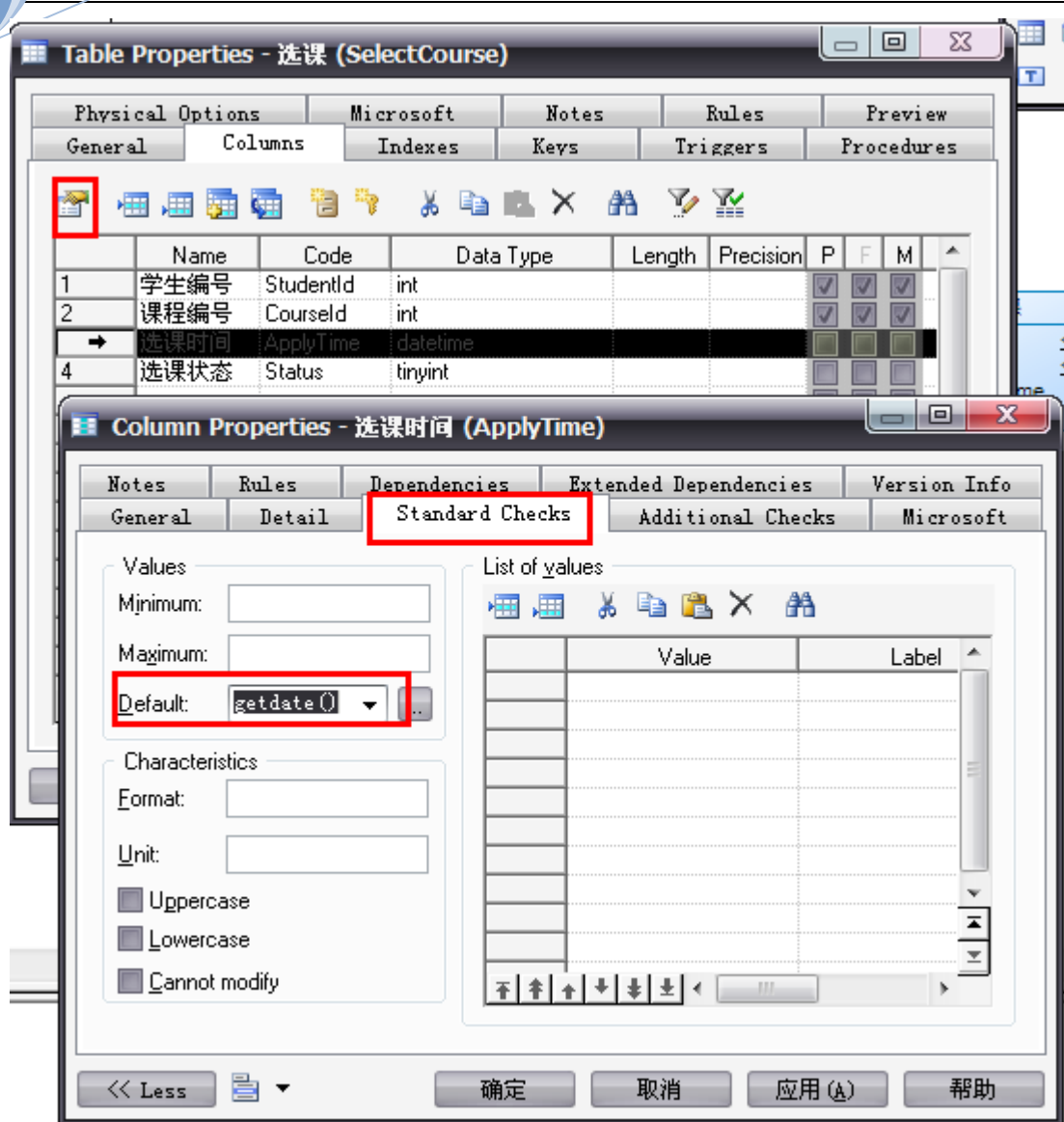
```
create table Class (  
    ClassID int not null,  
  
    ClassName varchar(20) not null,  
  
    constraint PK_CLASS primary key nonclustered (ClassID),  
  
    constraint CKT_CLASS check (ClassID<10000),
```

```
constraint ClassNameRule check (ClassName LIKE '2%')  
  
)  
  
go
```

可以看到，根据 Rule 生成的 CHECK 约束与在 Check 选项卡中设置的约束将分别创建一个约束，相互并不影响。

#### 4.3 默认约束

默认约束是用户在没有输入值的情况下，系统给出默认的值。最常用的是 CreateTime 字段，设置默认值为 getdate()，在用户创建一行数据时记录下创建时间。例如对于选课表，需要记录下选课的时间，则可以设置 ApplyTime 的默认值为 getdate() 函数。设置默认值约束的操作如下：双击选课表，打开表属性窗口，选择 ApplyTime 字段，单击工具栏的属性按钮，打开列的属性窗口，切换到 StandardChecks 选项卡，在 Default 下拉列表框中选择 getdate() 即可。

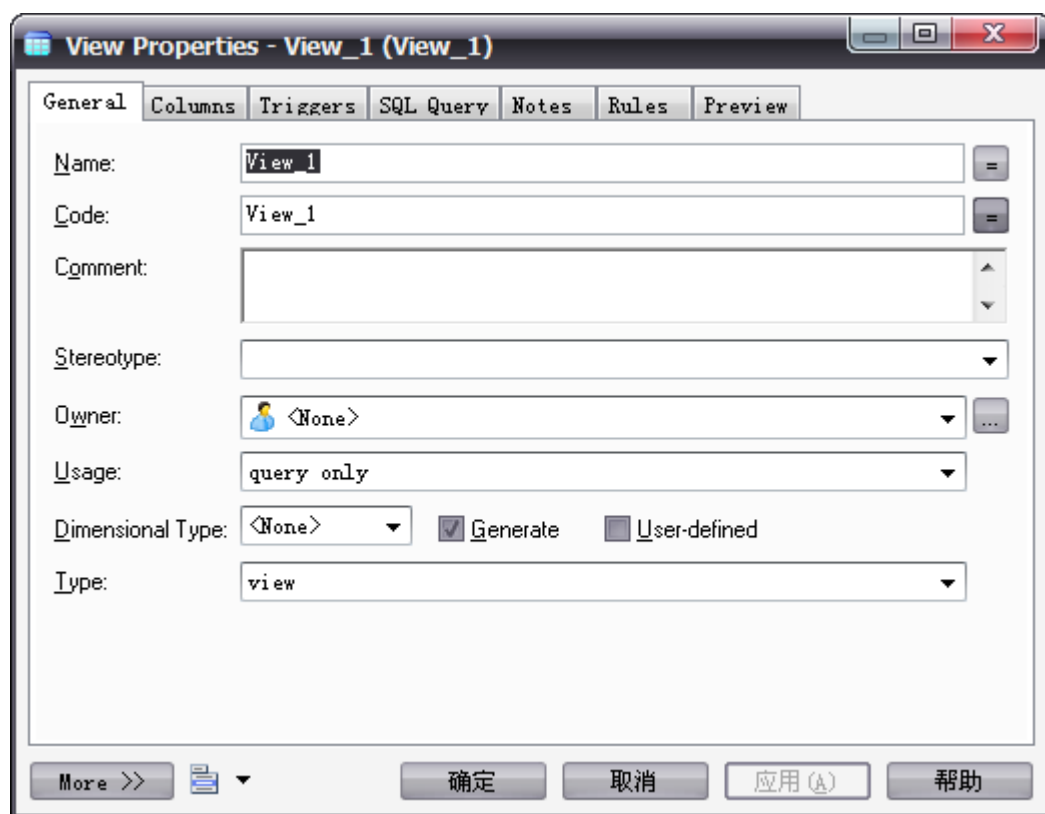
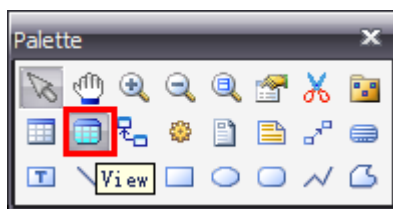


至此我们所有的约束在 PD 中的设置都介绍完了，下一篇将介绍视图、存储过程等数据库对象。

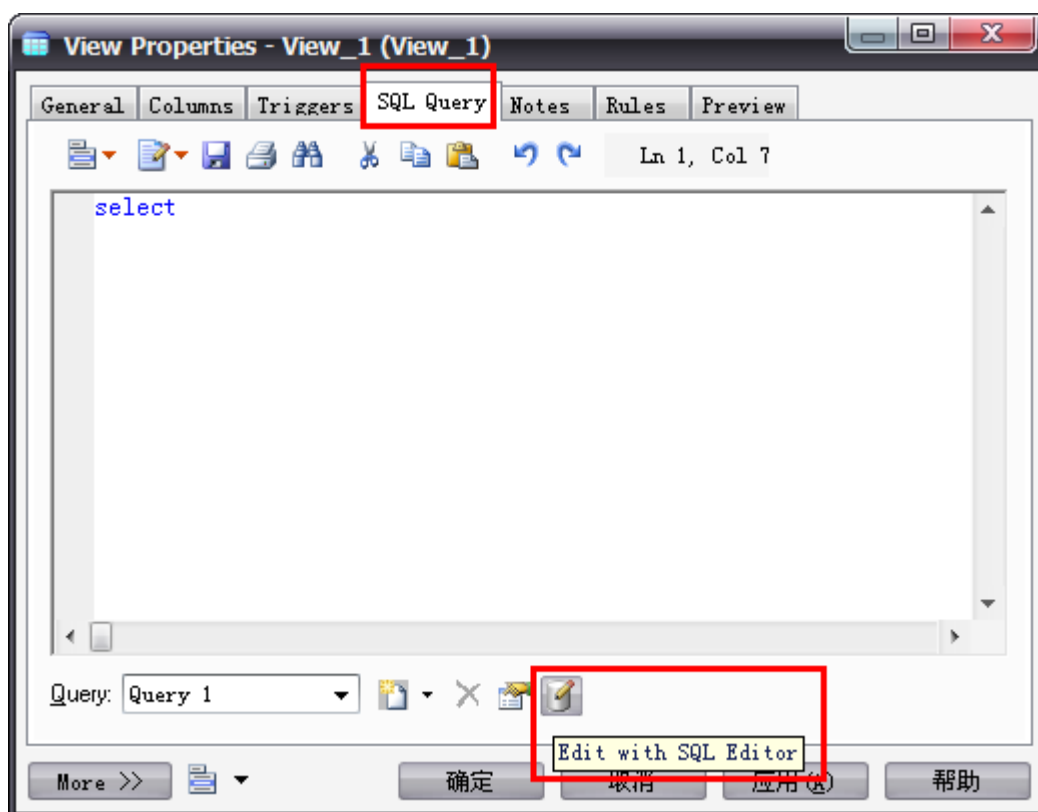
## 5. 物理模型设计 3——视图、存储过程和函数

### 5.1 视图

在 SQL Server 中视图定义了一个 SQL 查询，一个查询中可以查询一个表也可以查询多个表，在 PD 中定义视图与在 SQL Server 中定义查询相似。例如要创几个所有学生的所有选课结果的视图，那么在工具栏中选择视图按钮，然后在设计面板中单击鼠标一次便可添加一个空白的视图，切换到鼠标指针模式，双击该视图便可打开视图的属性窗口。在 General 选项卡中，可以设置视图的名字和其他属性。



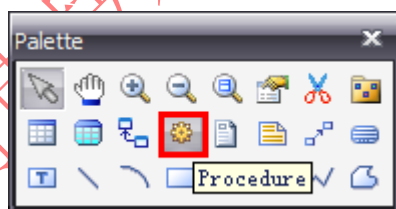
Usage 是表示视图是只读的视图还是可更新的视图，还有一个是 check option 选项，指定了 CHECK OPTION，也不能依据视图来验证任何直接对视图的基础表执行的更新。如果我们只创建一般的视图，那么就选择只查询选项。Dimensional Type 指定该视图表示的是维度还是事实，这个主要是在进行数据仓库多维数据建模时使用，一般情况下不需要指定。后面的两个复选框也不需要进行修改。Type 使用默认的 view 选项。切换到 SQL Query 选项卡，在文本框中可以设置视图定义的查询内容，建议直接先在 SSMS 中验证视图定义 SQL 语句的正确性，然后再将 SQL 语句复制粘贴到该文本框中。在定义视图时最好不要使用\*，而应该使用各个需要的列名，这样在视图属性的 Columns 中才能看到每个列。设计 SQL Query 如图所示。

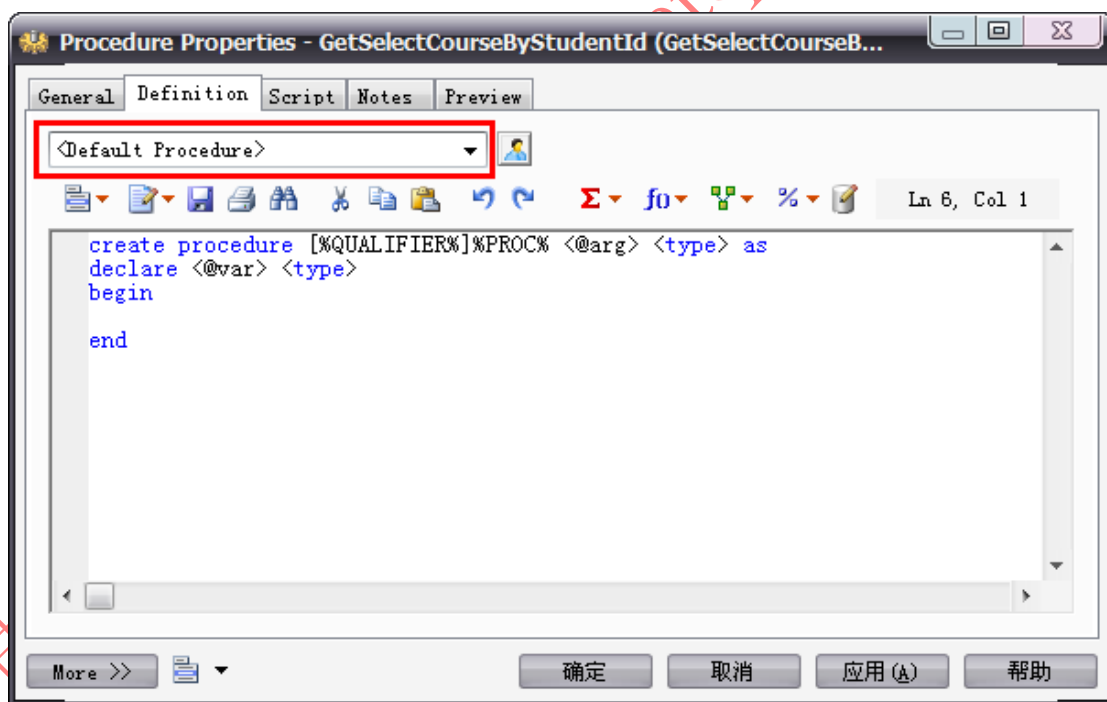
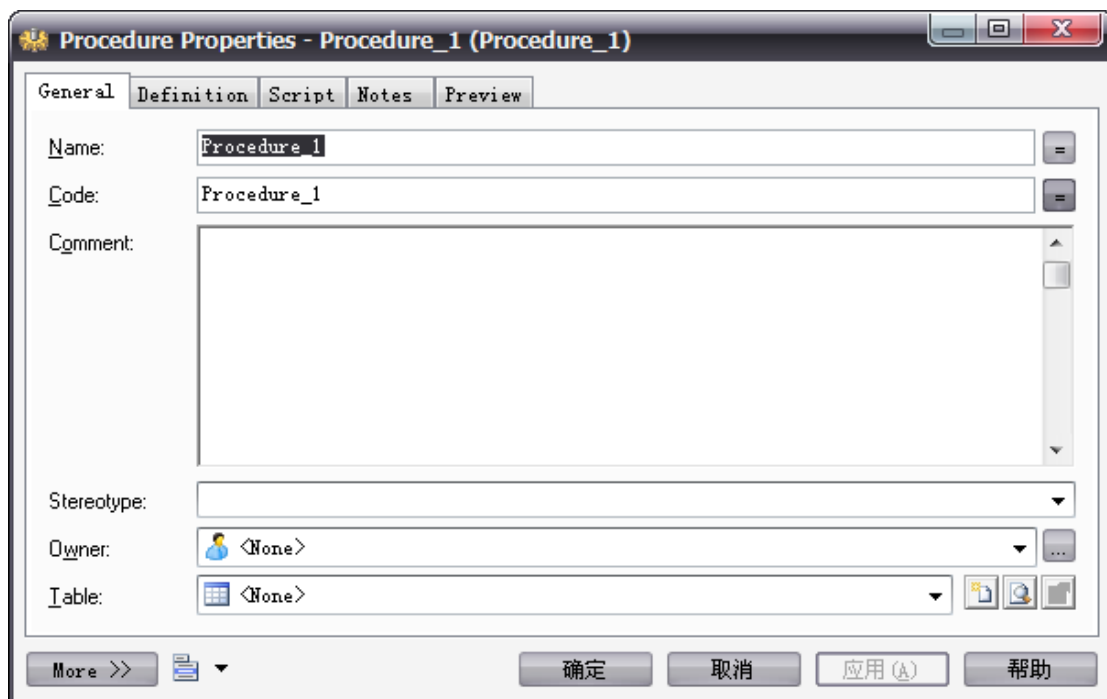


当然,也可以在PD 中使用自带的SQL 编辑器编写SQL 语句,单击右下角的“Edit with SQL Editor”按钮,即可弹出SQL Editor 编辑器,编写SQL 语句。

## 5.2 存储过程和函数

存储过程和用户自定义函数都是在同一个组件中设置的,在工具栏中单击 Procedure 按钮,然后在设计面板中单击一次便可添加一个Procedure。例如要创建一个存储过程根据学生的学号获得学生所选的课程,那么对于的操作如下:在指针模式下双击添加的Procedure,打开Procedure 属性窗口,在General 选项卡中可以设置该存储过程的名字。





在下面的SQL 语句中，可以将create procedure [%QUALIFIER%]%PROC%保留，其他的删除，根

据自己要创建的存储过程编写SQL 语句。

```
create procedure [%QUALIFIER%]%PROC%  
@StudentID int
```

```
as
```

```
begin
```

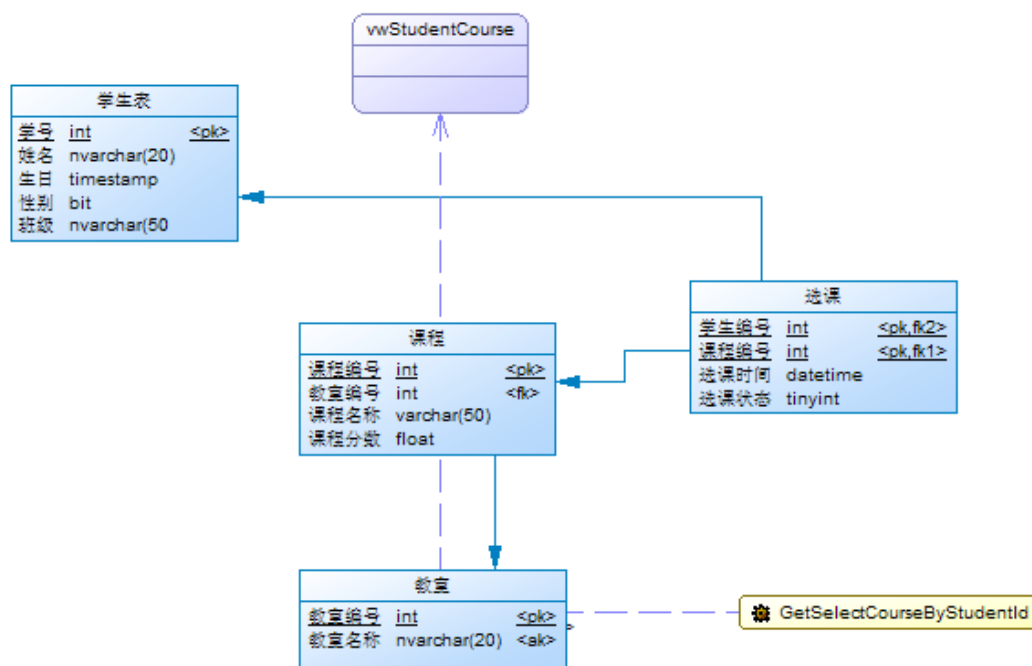
```
select CourseName
```

```

from vwStudentCourse
where StudentID=StudentID=@StudentID
end

```

单击确定按钮，系统会根据编写的 SQL 语句，将所使用的表、视图与存储过程关联起来，如图所示：



创建函数的过程与之类似，只是使用的是create function 而不是create Procedure 而已。

至此，最常见的数据库对象：表（表的约束）、视图、存储过程、函数等在PD 的创建已经介绍完了，接下

## 6. 定义和使用域 (Domain)

### 6.1 关于域

一个域定义能适用于多个数据项（Data Item）的标准数据结构。当修正一个域时，将更新全部与域关联的数据项目。当作任何变化的时候，导致数据一致化特性比较容易实现。

### 6.2 创建一个新的域

下面将会创建两个域，为钱数定义一个标准化的域，为模型中的百分比定义一个域。

1 选择菜单栏的模型域，Model > Domains。



域对话框的列表显示已存在的域。

2 点击增加一行(add a row)工具。

一支箭头在第一空行的开始处出现,一个默认的名字和代码被进入。

3 在名字列输入Amount。

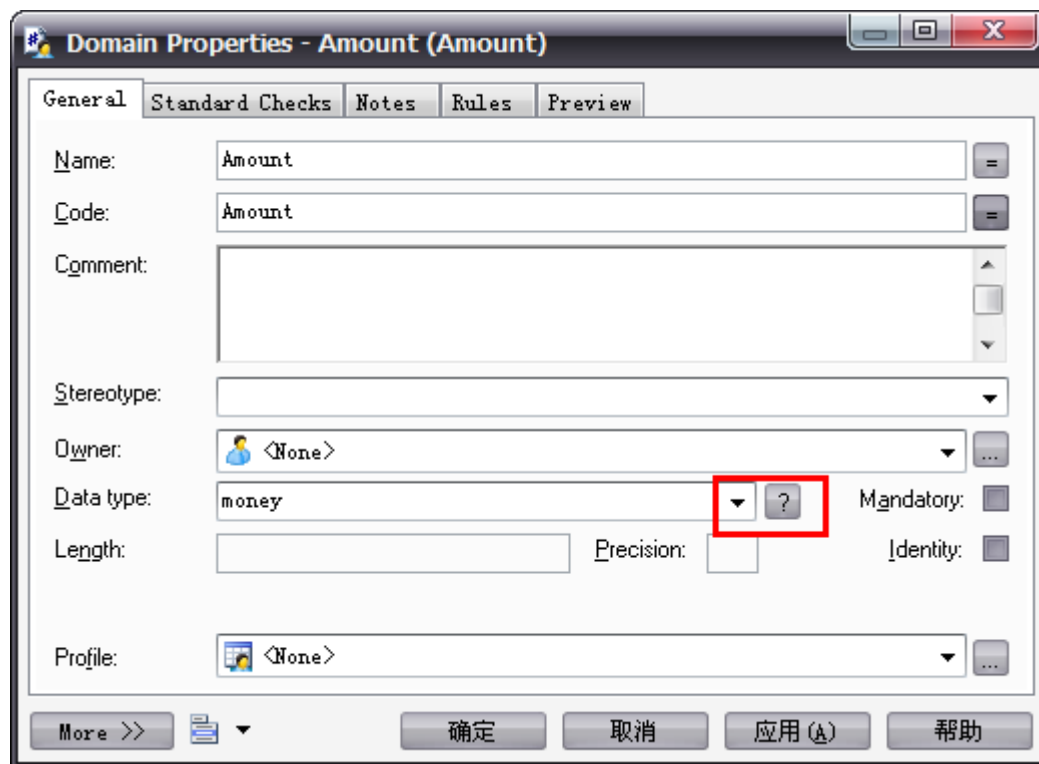
这是域的名字。相同的代码自动地在代码列中被进入。

4 点击应用。新创建的域被提交。

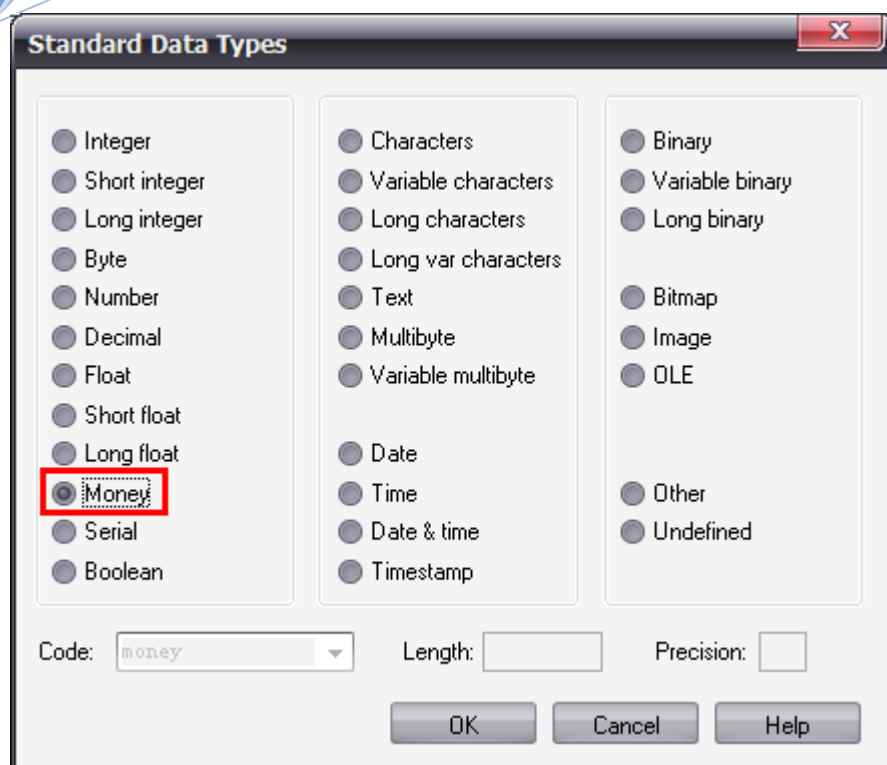
5 点击新的域行。

6 点击属性工具。

7 点击紧邻数据类型下拉列表的问号按钮。



8 点击Money单选按钮。



域现在有一个**Money**数据类型。一个**Money**数据类型用一个固定长度的数字来储存数字。然后，当应用这个域用来储存**Money**的数量数据项目（Data Item）的时候，将会使他们继承这一个数据类型。

9 在长度框中打字8。附到这个域的数据项目的最大的长度将会是8。

10 在精密框中打字2。附到这个域的数据项目在小数点之后能带二个位置。

11 点击OK。

回到域特性页。值MN8,2在数据类型列表框中出现。MN是一个**Money**数据类型的代码。8指出**Money**类型能有8个长度。2指出数量有2个小数位精度。

12 点击OK，回到域的列表。

13 点击增加一行工具。

一支箭头在第一个空白行的开始处，一个默认的名字和代码被输入。

14 名称列键入Percent。

这是域的名称。相同的代码自动地在代码列中被进入。

15 点击应用。新创建的域被提交。

16 点击新的域行。一个箭头在行开始处出现。

17 点击属性工具。或在行开始处双击箭头。属性页为新创建的域。

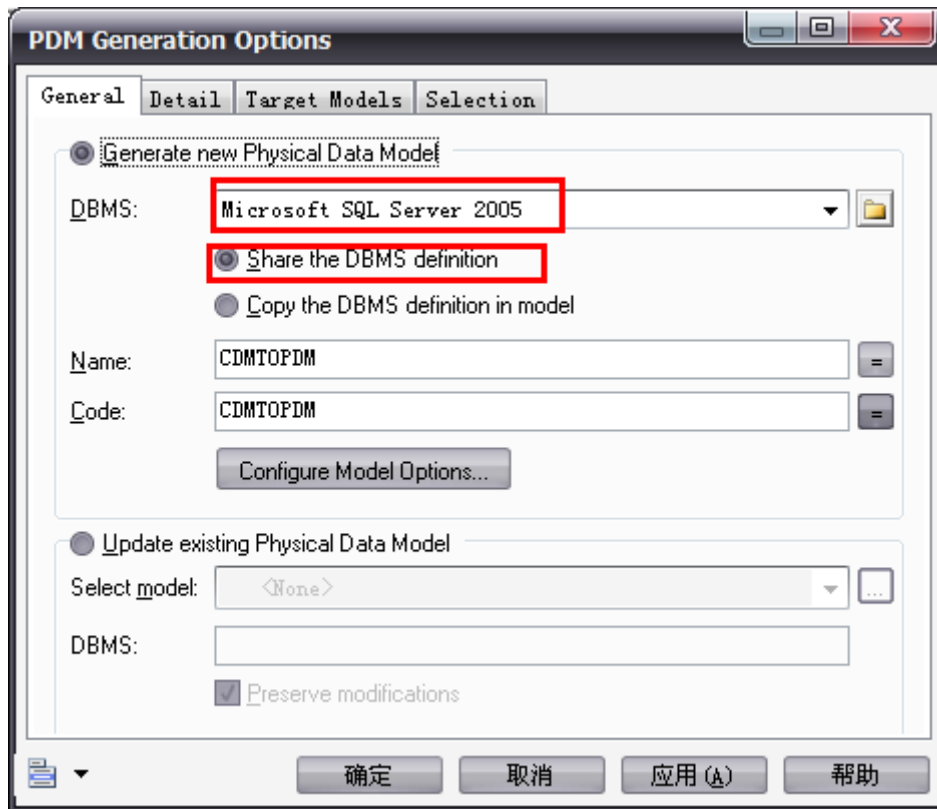
18 点击问号按钮旁边数据类型下拉列表框。

## 7. 将 CDM 对象转换成 PDM 对象

当从一个CDM生成PDM时，PowerDesigner将CDM中的对象和数据类型转换为PDM对象和当前DBMS支持的数据类型。同一个表中的两列不能有相同的名称，如果因为外键迁移而导致列名冲突，PowerDesigner会自动对迁移列重命名，新列名由原始实体名的前三个字母加属性的代码名组成。主标识符在生成

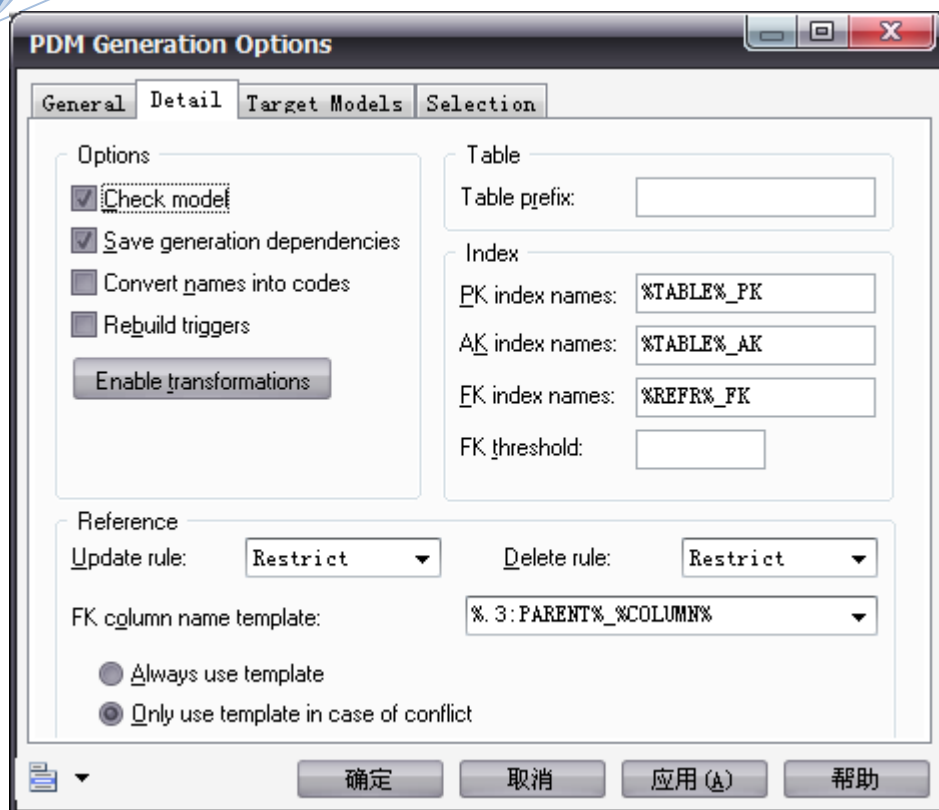
PDM中的主键和外键，非主标识符则对应生成候选键。在PDM中生成的键类型取决于CDM中用于定义一个Relationship的基数和依赖类型。

- 1 选择Tools—>Generate Physical Data Model。PDM生成选项对话框出现。
- 2 从数据库管理系统下拉列表框选择一种数据库系统系统。
- 3 选择 Share 单选按钮。



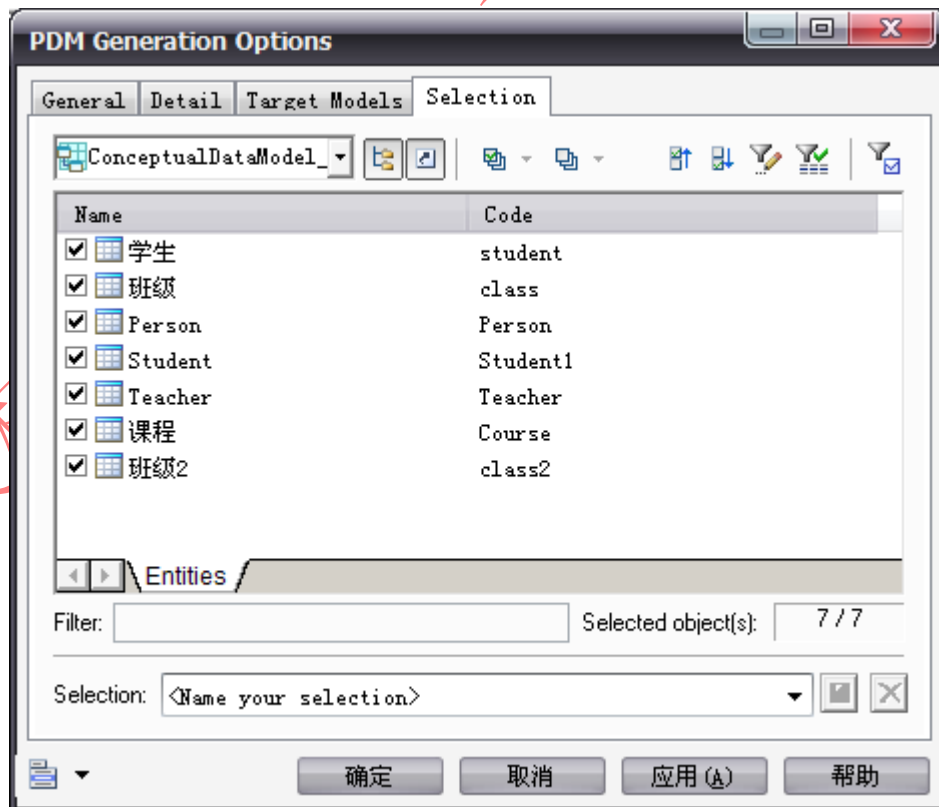
指出使用数据库管理系统库中储存的数据库管理系统定义文件。

- 4 点击detail页面。细节页出现。
- 5 挑选出来的或清除下列各项选项：

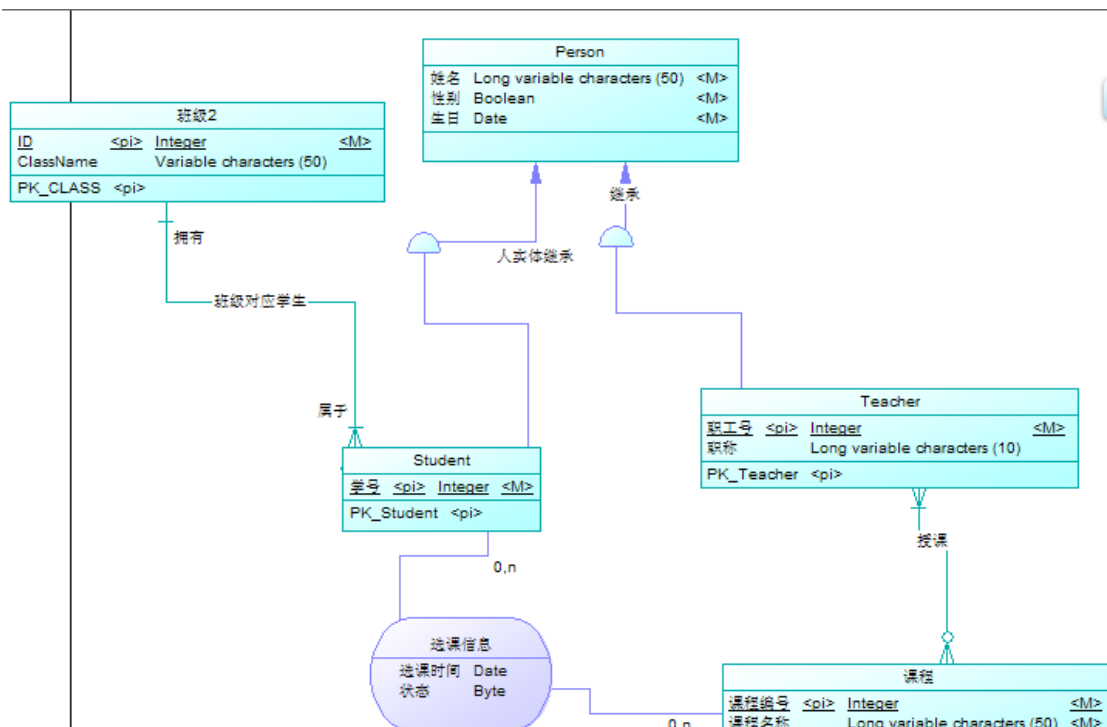


如果选择了Check Model，模型将会在生成之前被检查。Save GenerationDependencies 选项决定PowerDesigner 是否为每个模型的对象保存对象识别标签，这个选项主要用于合并由相同CDM生成的两个PDM。

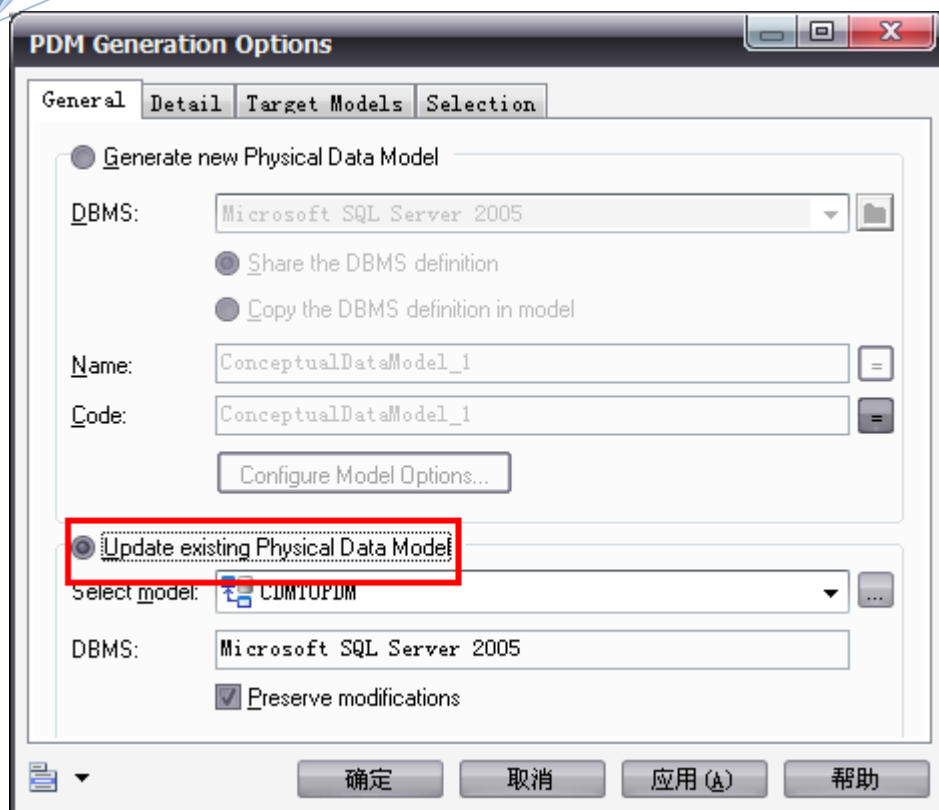
6 选择Selection 定位键。它列出 CDM 的所有对象。默认地，所有的对象被选择。



- 7 点击确定。
- 8 当程序已经完成生成时，关闭结果窗户。  
PDM 在模型的窗口中出现。



生成PDM后，可能还会对前面的CDM进行更改，若要将所做的更改与所生成的PDM保持一致，这时可以对已有PDM进行更新。这时操作也很简单，Tools->Generate Physical Data Model，在打开的PDM Generation Options窗口中选择Update existing Physical Data Model，并通过Select model下拉框选择将要更新的PDM。如下图：

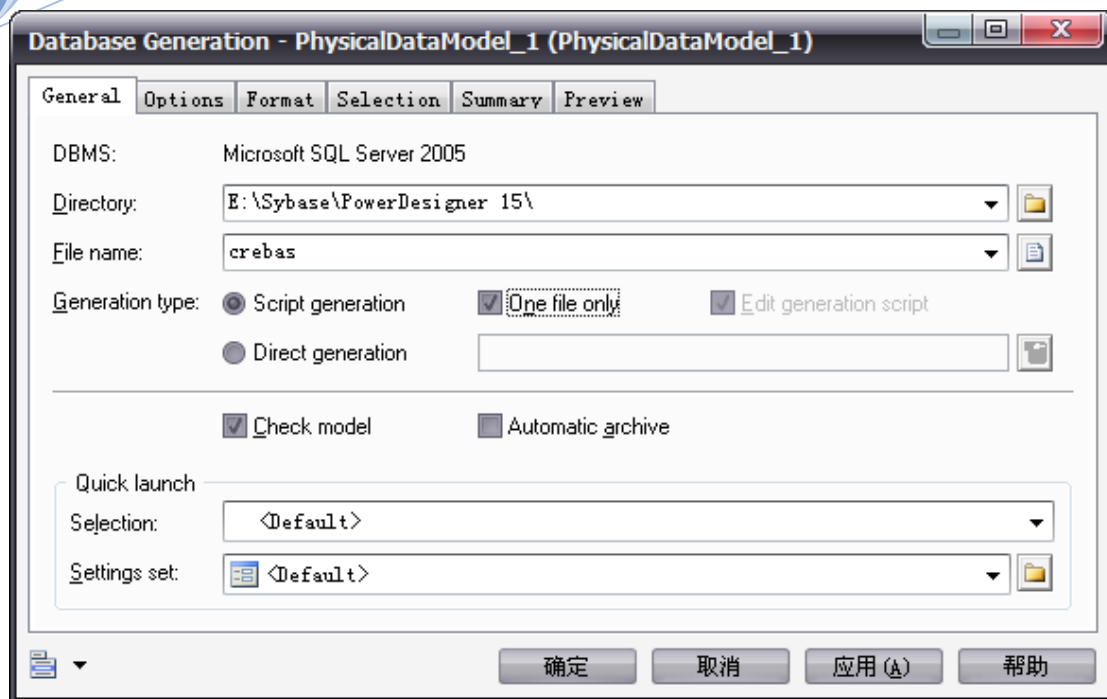


## 8. 使用物理数据模型创建数据库脚本

可以直接地从PDM产生一个数据库，或产生一个能在特定的数据库管理系统环境中运行的数据库脚本。默认是生成与PDM相同数据库的脚本，但是也可以产生其他数据库的脚本产生一个数据库创建脚本

1 选择Database—> Generate Database。

数据库生成对话框出现。 它显示生成参数。 默认参数已经被选择。



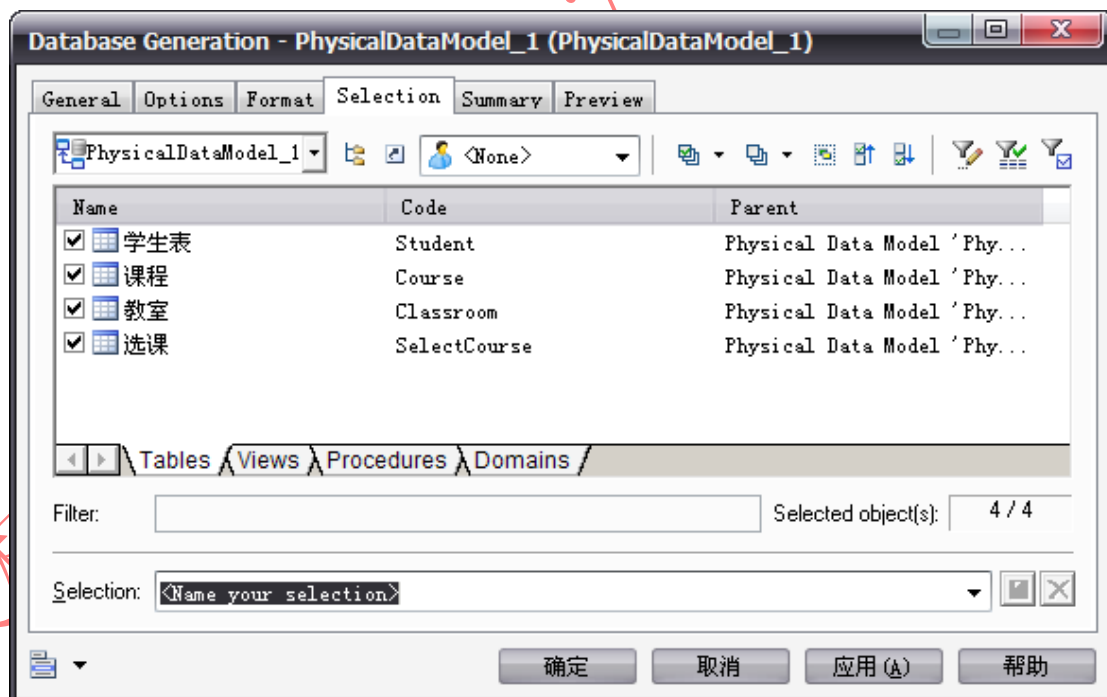
2 SQL的文件名称框中键入脚本文件名。

3 在目录框中，输入或选择一条路径。

4 选择生成脚本的按钮。

5 选择仅仅生成一个文件。

6 点击Selection 定位键。



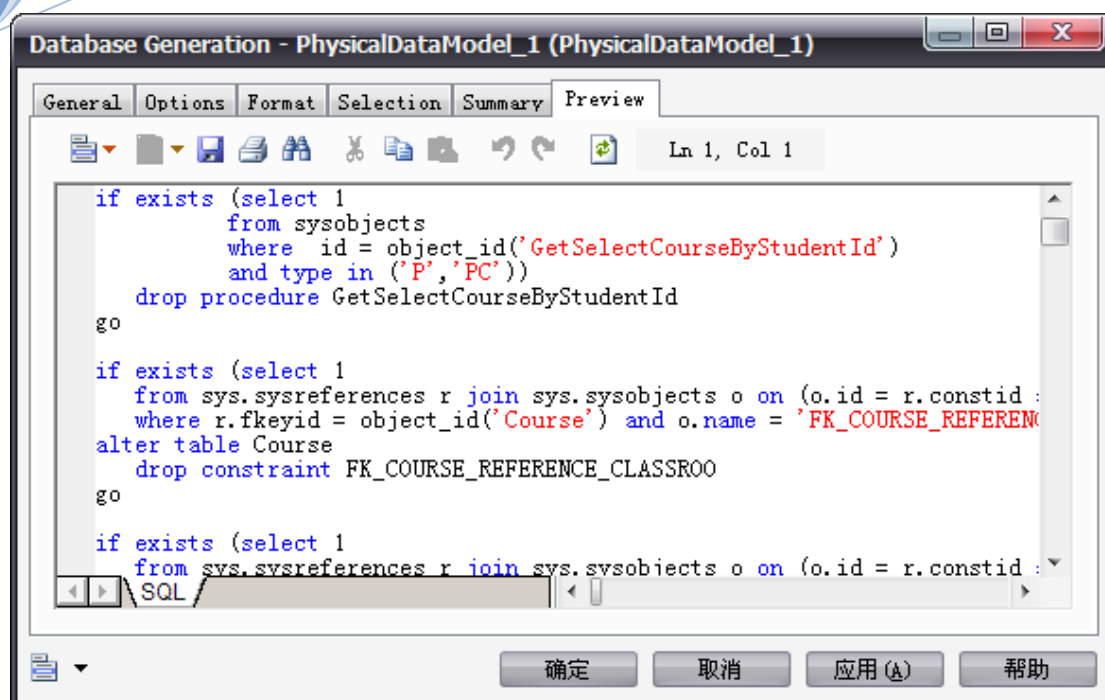
7 点击底部表定位键。

表页列出模型中选择可用的所有数据库表。

8 全部点击选择工具。这选择所有的表复选框。

9 切换Views和Domains选择需要的视图和域。

10 点击Preview，可预览脚本。

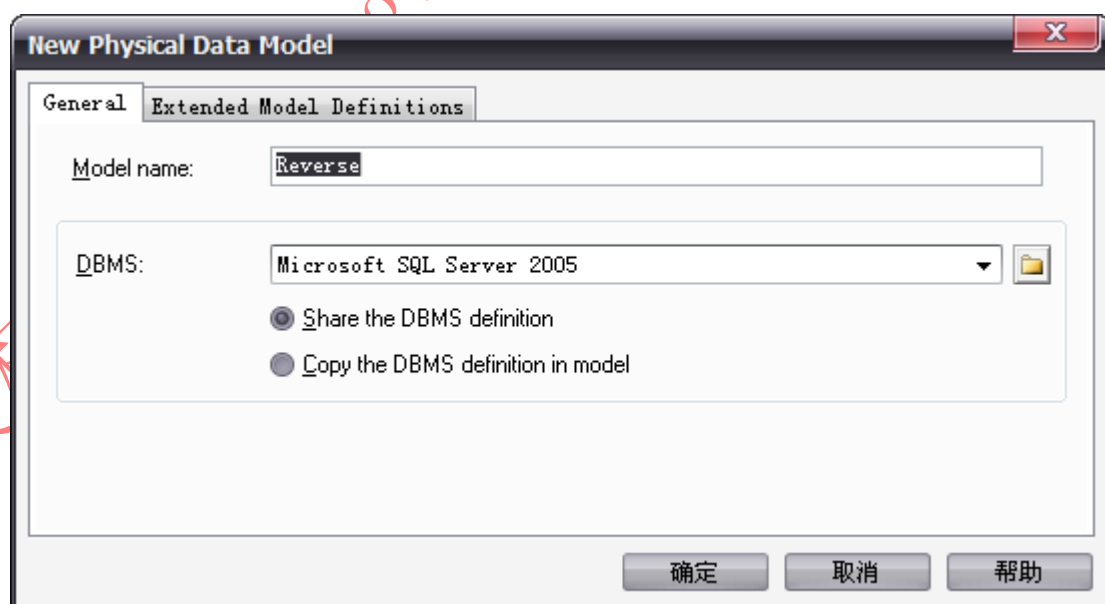


## 9. 逆向工程

Powerdesigner可以逆向工程已存在的数据库，加进新的PDM之内。数据来源可以是脚本文件或一个ODBC数据源。当你逆向工程使用脚本的时候，可以使用单一脚本文件或多个脚本文件。

### □ 从脚本文件逆向工程到PDM

1 选择File->Reverse Engineer->Database 显示新的物理数据模型的对话框。



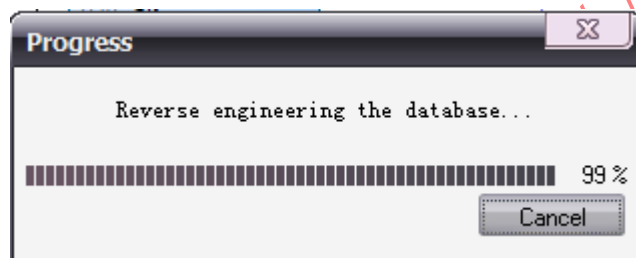
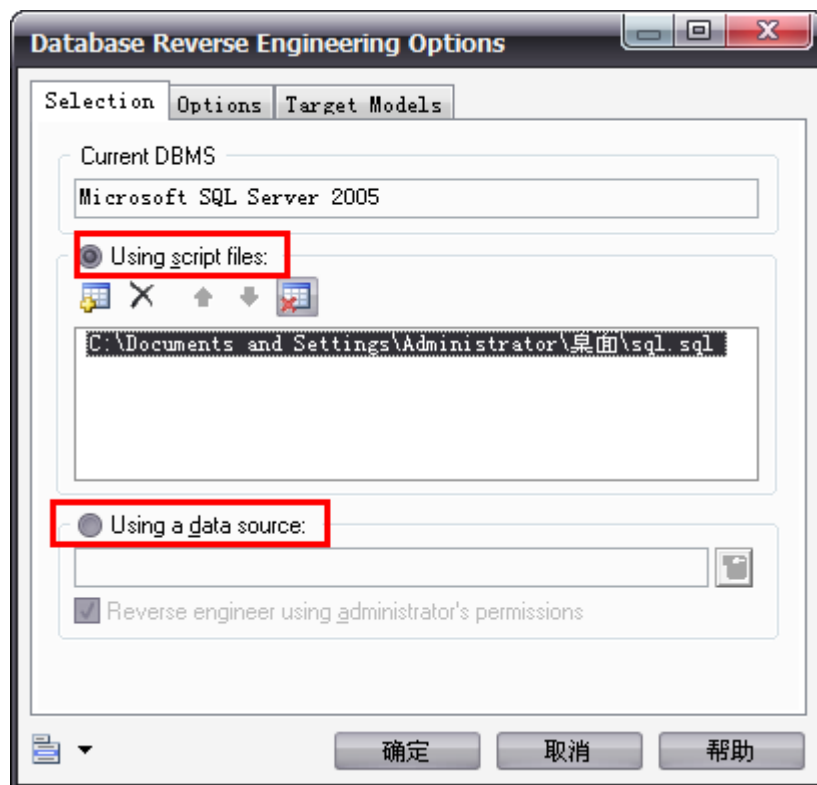
2 输入数据模型名称。

3 选择下拉列表框的一个数据库管理系统。

4 点击确定。数据库逆向工程对话框出现。可以有两种选择。一种是使用sql脚



本文件，另一种是直接使用数据源。

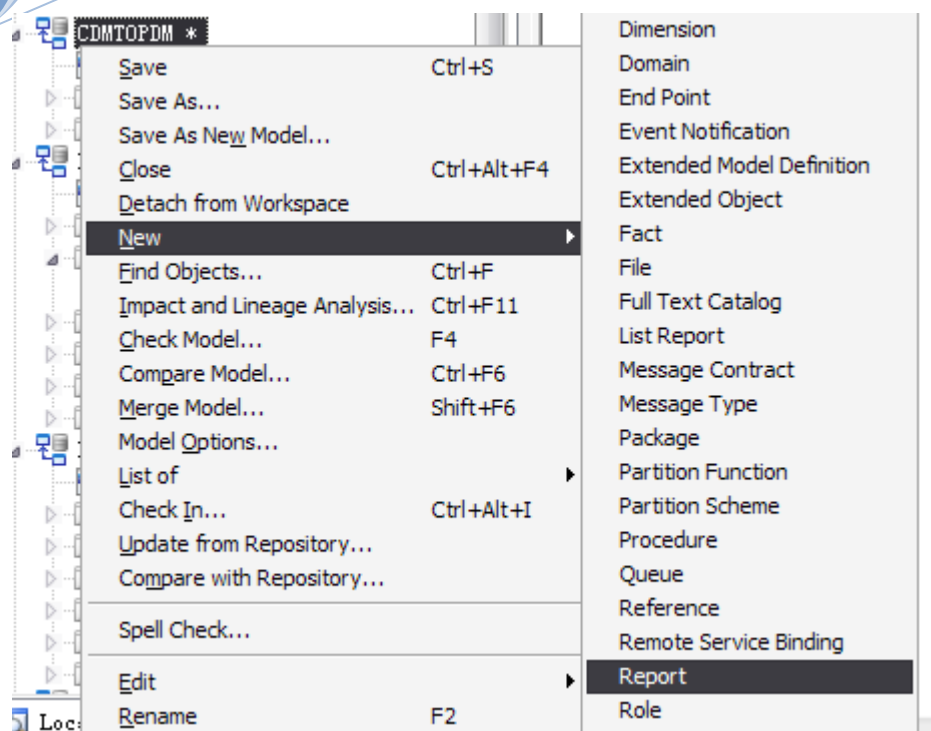


## 10. 生成数据库报表文件

当设计出数据库以后，有一份正规的DOC文件是很必要的，下面讲解如何生成报表。

### 1. 新建立报表文件

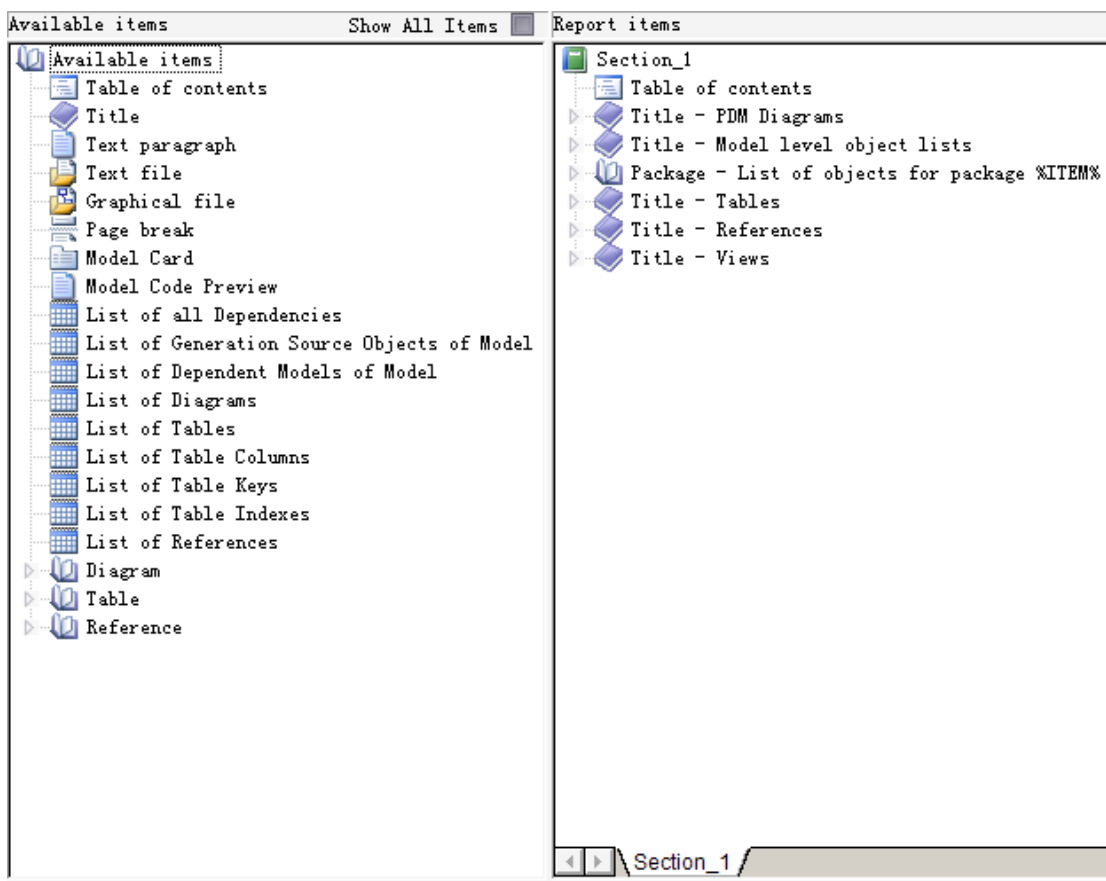
右键单击模型，选择New->Report，弹出如下界面



输入报表名称，选择报表模板。

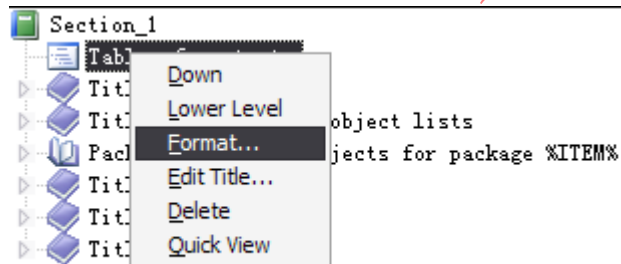


1. 拖拉需要显示的部分到报表输出选项

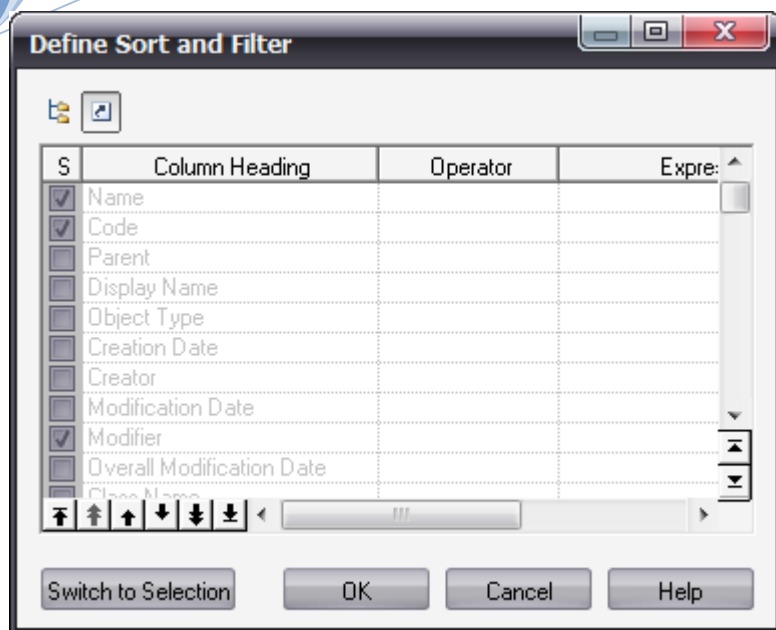


### 3. 修改输出选项的内容

在内容上单击右键，选择Format，可以修改相应项目的标题、字体、格式等。



### 4. 修改列的宽度和需要显示的内容



5.生成RTF或者其他的类型的报表文件.

